

Diplomarbeit

Webbasiertes User-Management des Virtuellen Unix Labors

© 10. Juni 2003

Fachbereich: Informatik

Verfasser: Stefan Zimmermann

Betreuer: Prof. Jürgen Sauer

Erklärung

1. Mir ist bekannt, dass die Diplomarbeit als Prüfungsleistung in das Eigentum des Freistaats Bayern übergeht. Hiermit erkläre ich mein Einverständnis, dass die Fachhochschule Regensburg diese Prüfungsleistung die Studenten der Fachhochschule Regensburg einsehen lassen darf, und dass sie die Abschlussarbeit unter Nennung meines Namens als Urheber veröffentlichen darf.
2. Ich erkläre hiermit, dass ich diese Diplomarbeit selbständig verfasst, noch nicht anderweitig für andere Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen und Hilfsmittel benützt sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Regensburg, den 10. Juni 2003

.....
Unterschrift

Vorwort

Thema der Diplomarbeit ist die Benutzerverwaltung des HWP-Projekts "Virtuellen Unix Labors" auf der Basis einer Three-Tier Architektur. Hauptbestandteile sind hierbei der Applikations-Server in der Mittelschicht, sowie die Integration und Implementierung des Datenbank-Backends.

Dem Projekt liegt eine Problematik zu Grunde, mit der sich die Diplomarbeit einleitend befasst. Die Architektur in Kapitel 2 zeigt das eigentliche Konzept des Vorhabens, das später im Kapitel 3 im Detail verdeutlicht wird. Auftretende Probleme im Hinblick auf Sicherheit werden dabei nicht außer Acht gelassen.

Damit das Resultat des vorliegenden Schriftstücks Buchdruckqualität hat, wurde es in \LaTeX verfaßt. Dabei handelt es sich um ein Satzprogramm, das nahezu alle Aufgaben lösen kann, die bisher dem traditionellen Beruf des Setzers vorbehalten waren [Kopka (2002)]. Doch noch wichtiger als die Buchdruckqualität ist die Weiterverwendung. Da das \LaTeX -Dokument im ASCII-Format vorliegt, können Teile der Diplomarbeit ohne Weiteres weiterverarbeitet und in die Dokumentation des "Virtuellen Unix Labors" aufgenommen werden.

Mein besonderer Dank gilt ANDREAS NEUMEIER, der immer ein offenes Ohr für meine Fragen hatte, und mir mit vielen Tipps und Tricks zur Seite stand. Ebenso bedanke ich mich bei HUBERT FEYERER, der mir als Administrator der FH-Regensburg einen Arbeitsrechner zur Verfügung stellte und in Sachen NetBSD eine große Hilfe war. Eine weitere wichtige Person ist HERR PROF. JÜRGEN SAUER als Betreuer der Diplomarbeit, bei dem ich mich auch recht herzlich bedanke. Ohne ihn wäre diese Diplomarbeit nicht möglich gewesen.

Stefan Zimmermann, 10. Juni 2003

Inhaltsverzeichnis

Erklärung	iii
Vorwort	v
Inhaltsverzeichnis	x
1 Grundlagen	1
1.1 Hintergründe	1
1.2 Vorhaben	1
1.3 Thick Client	3
1.4 Three-Tier Architektur	3
1.4.1 Client Tier – Thin Client	4
1.4.2 Middle Tier – Web und Applikations-Server	5
1.4.3 Database Tier	5
1.5 Verwendete Software	6
1.5.1 Web Server: Apache	6
1.5.2 Applikations-Server: PHP	7
1.5.3 Datenbank: PostgreSQL	7
2 Architektur	9
2.1 Neuanmeldung	9
2.2 Login	12
2.3 Erstellen von Übungen	13
2.4 Buchen der Übungen	14
2.5 Ablauf der Übung	16
3 Programmierung	19
3.1 Probleme zwischen den Versionen	19
3.1.1 Globale Variablen – Ein Schicherheitsrisiko ?	19
3.1.2 Pufferung des Ausgabestroms	21
3.2 Seitenaufbau	23

3.2.1	Menü	26
3.2.2	Seiteninhalt	29
3.3	Session-Handling	34
3.4	Benutzerverwaltung	38
3.4.1	Benutzerprofil anlegen	38
3.4.2	Login-Vorgang	41
3.4.3	Benutzerdaten im Anwenderbereich	42
3.4.4	Benutzerdaten im Administrationsbereich	45
3.5	Buchungssystem	51
3.5.1	Buchen von Übungen im Anwenderbereich	51
3.5.2	Verwalten der Buchungen im Anwenderbereich	60
3.5.3	Verwalten der Buchungen im Administrationsbereich	64
4	Installation	69
4.1	Aufbau der Datenbank	69
4.2	Aufbau des Web-Frontend	70
4.3	Säubern der Datenbank — Cron	72
4.4	Vom <i>www-user</i> zu startende Scripte	73
5	Weiterentwicklung	75
5.1	Erzeugen eines neuen Menüpunkts	75
5.2	Aufbau eines neuen Moduls	78
5.2.1	Session-Handling	78
5.2.2	Error-Handling	80
5.2.3	Aufruf weiterer Scripte	82
6	Stolpersteine auf dem Weg zum Erfolg	85
6.1	Keine Verbindung zu PostgreSQL — <i>pg_hba.conf</i>	85
6.2	Nicht existierende Methode — <i>pg_query()</i>	86
6.3	Der Unterschied zwischen <i>char</i> und <i>varchar</i> bei PostgreSQL	86
6.4	Das Problem mit den globalen Variablen	88
6.5	Serverseitige Pufferung des Ausgabestroms	89
6.6	Session-Handling ohne <i>--enable-trans-sid</i>	90
6.7	Fazit	90
A	Aufbau des Systems	91
A.1	Web Server Apache	91
A.1.1	Installation	91
A.1.2	Konfiguration und Testbetrieb	92
A.2	Script Sprache PHP	94

A.2.1	Installation	94
A.2.2	Konfiguration	95
A.2.3	Testbetrieb	96
A.3	DBMS PostgreSQL	97
A.3.1	Installation	97
A.3.2	Client Authentifizierung	99
A.3.3	Arbeiten mit der Datenbank	100
B	Die Datenbankstruktur	103
C	Datensicherung	111
D	Funktionsverzeichnis	115
E	Programmcode	125
E.1	Basis-Skripte zum Seitenaufbau	125
E.2	Module im Defaultbereich	151
E.2.1	Home	151
E.2.2	Login	153
E.2.3	Profil anlegen	160
E.2.4	Info	171
E.2.5	Bestätigen	173
E.2.6	Error	175
E.3	Module im Anwenderbereich	177
E.3.1	Home	178
E.3.2	Benutzerdaten	180
E.3.3	Buchung vornehmen	193
E.3.4	Buchungen einsehen	229
E.3.5	Error	243
E.4	Module im Administrationsbereich	246
E.4.1	Home	246
E.4.2	Benutzerdaten	247
E.4.3	Buchungen	277
E.4.4	Error	294
E.5	Module im Dozentenbereich	295
E.5.1	Home	295
E.5.2	Benutzerdaten	296
E.5.3	Error	296
F	Beigefügte CD	297

Abkürzungsverzeichnis	300
Tabellenverzeichnis	301
Bilderverzeichnis	304
Quellcodeverzeichnis	307
Literaturverzeichnis	309
Stichwortverzeichnis	311

Kapitel 1

Grundlagen

1.1 Hintergründe

Die an der Hochschule vorhandenen Computer-Labore bieten jedem Studenten die Chance sich mit dem Betriebssystem Unix auseinander zusetzen. Sie können unter ihrer Benutzerkennung auf diesen Systemen uneingeschränkt arbeiten, indem ihnen eine Vielzahl von Entwicklungswerkzeuge zur Verfügung gestellt werden. Leider gibt es keine Möglichkeit, lernwillige Studentinnen oder Studenten tiefer in das Betriebssystem eingreifen zu lassen, wie z. B. im Rahmen der Systemadministrations-Vorlesung.

Das heißt: Sie haben keine Rechte, systemweit Software zu installieren oder Konfigurationen zu ändern. Der Sinn des Informatik-Studiums besteht aber gerade darin, tiefer in solchen Computersystemen zu arbeiten, als es einfachen Benutzern erlaubt ist.

1.2 Vorhaben

Es bietet sich dennoch Studenten die Vorlesung 'Systemadministration' an, bei der sie in die Tiefen des Unix vordringen und etliche Tricks und Kniffe lernen, wie solch ein System eingestellt wird. Leider gibt es nur eingeschränkte Möglichkeiten das Vermittelte auszuprobieren bzw. anzuwenden. Deshalb wurde nun ein Projekt ins Leben gerufen, welches den Studentinnen und Studenten erlaubt, Unix auf Administratorebene zu betrachten und zu bedienen.

Geplant ist ein Computer-System, das die notwendigen Rechner für die Übungen automatisch aufsetzt, den Studenten Zugang gewährt, die Übungen über-

wacht und am Ende auswertet. Damit jeder Student die gleichen Übungsbedingungen erhält, müssen alle eventuell gemachten Änderungen am System durch Deinstallation rückgängig gemacht werden. Eine automatische Neuinstallation spart jedoch diesen Aufwand und entlastet somit den Systemverwalter.

Abbildung 1.1 zeigt die Zusammenhänge des Networks.

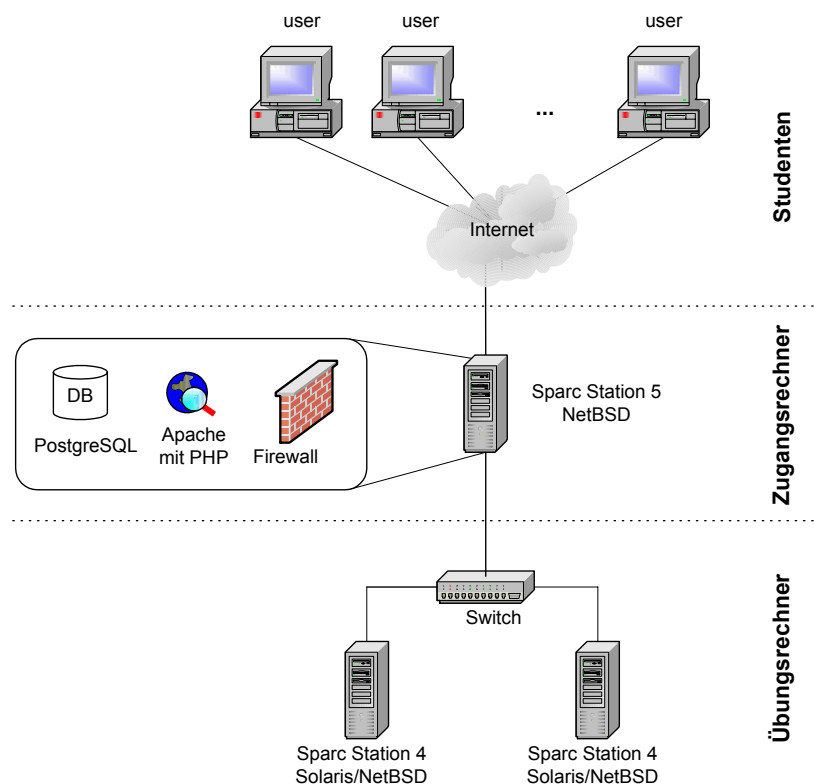


Abbildung 1.1: Geplante Netzwerkstruktur

Mit dieser Netzwerkstruktur soll nun die Möglichkeit geschaffen werden, einem Benutzer root-Zugriff auf die ganz unten in Abbildung 1.1 stehenden Übungsrechner zu gewähren. Dort können sie dann in einem nach außen abgeschotteten lokalen Netz alles ausprobieren, ohne den Produktivbetrieb im FH-Netz zu stören.

Damit sich die übenden Studenten nicht gegenseitig behindern, lässt die Firewall - gesteuert über die Sparc Station 5 - nur einen Benutzer gleichzeitig zu. Das Hauptaugenmerk dieser Arbeit ist darum der Zugangsrechner basierend auf dem Betriebssystem NetBSD/sparc. Auf ihm wird in erster Linie die Benutzerverwaltung realisiert.

1.3 Thick Client

Vernetzte PCs mit lokal installierten Anwendungen werden gerne als "Fat Client" oder "Thick Client" bezeichnet. Sie benutzen alle die gleiche Software, was zu hohen Betriebskosten durch Software-Wartung und Installation führt. Weiterhin entsteht ein großer Supportbedarf, der durch die Unverträglichkeiten der Anwendungen mit Treibern immer komplexer werdender Betriebssysteme hervorgerufen wird [Thi (2001)]. Aus diesem Grund kann man einen Trend weg vom "fetten" Client mit hoher Funktionalität erkennen, und tendiert deshalb eher in Richtung Thin-Client. Der Vorteil des "dünnen" Client besteht darin, dass er nur mehr als Anzeigeeinheit oder Display verwendet wird. Die eigentlichen Aufgaben dazu erledigt ein zentraler Applikations-Server in der Mittel-Schicht der Three-Tier Architektur. In der Diplomarbeit wurde dieser Ansatz gewählt.

1.4 Three-Tier Architektur

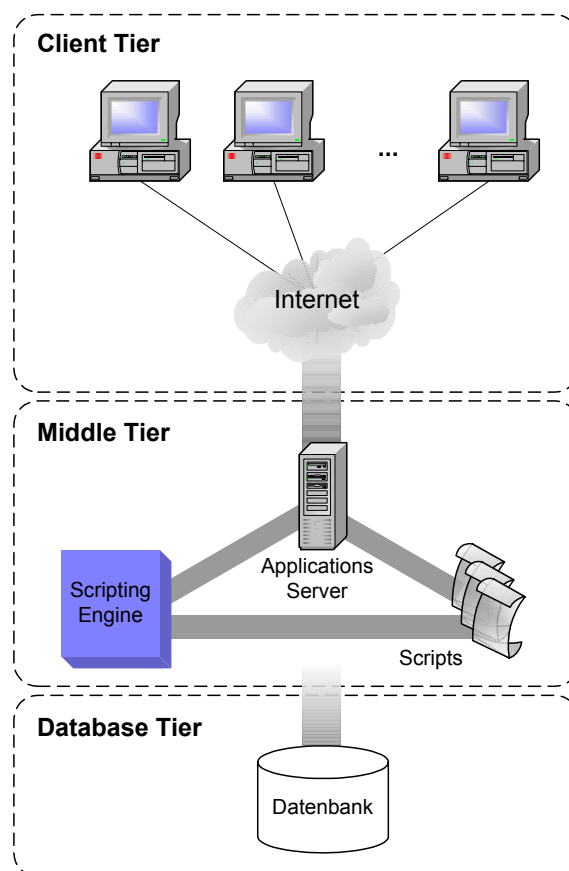


Abbildung 1.2: Three Tier Architektur

Da das Internet immer weiter wächst und die standardisierte Web-Browser-Software auf einer immer breiteren Basis von Betriebssystemen läuft, bedient man sich auch hier dieser Technologie. Die aufzubauende Anwendung arbeitet dabei über drei Ebenen der Applikations-Logik.

In der Fachsprache "Three-Tier Architektur"¹ [Williams and Lane (2002)] genannt, gliedert sich der Aufbau in "Client Tier", "Middle Tier" und "Database Tier" (siehe Abbildung 1.2). Sehr oft laufen die Dienste dieser Schichten auf jeweils verschiedenen Rechnern. Bei diesem Projekt jedoch werden die beiden unteren Ebene (Web Service und Datenbank) auf einem Computer gefahren (siehe Abbildung 1.1). Die Beschreibung des Systemaufbaus findet sich in Anhang A.

1.4.1 Client Tier – Thin Client

Die Client-Schicht in der Three-Tier Architektur ist gewöhnlich der Web Browser des Endanwenders. Diese Software fungiert als Thin Client, was soviel bedeutet, als dass beim Anwender so wenig Applikations-Logik wie möglich steckt. Software wie zum Beispiel für Datenbank-Zugriffe, Zugangskontrolle, Verschlüsselung oder Ähnliches kann verzichtet werden. Der Browser sendet nur HTTP-Anfragen an den Server und zeigt die darauf erhaltenen Ergebnisse an. Dies geschieht überwiegend als HTML-Dokument.

Das Drei-Schicht-Modell bedeutet in diesem Fall, dass auf der Seite des Clients keinerlei Installation oder Konfiguration nötig sind. Jeder Benutzer, der einen Web Browser hat, kann ohne Aufwand sofort auf die Applikation zugreifen. Es ist auch nicht notwendig ein bestimmtes Betriebssystem oder eine besondere Hardware zu verwenden. Die gewünschten Anwendungen werden automatisch über einen Applikations-Server angesprochen und ihre Ergebnisse an unzählige unterschiedliche und geographisch unabhängige Benutzer verteilt [Williams and Lane (2002)]. Dazu interagiert der Client mit der Mittelschicht, die eine Schnittstelle zur Datenbank bildet.

¹Tree-Tier kommt aus dem englischen und bedeutet drei-Schicht. In einer drei-Tier-Architektur wird die Anwendung in drei Schichten aufgeteilt.

1.4.2 Middle Tier – Web und Applikations-Server

In der Regel liegt der Großteil der Applikations-Logik in der Mittelschicht. Der Client sammelt Daten vom Anwender und präsentiert die Ergebnisse. Die Datenbank empfängt und speichert Informationen. Die restliche Funktionalität liegt in der Mittelschicht. Sie verbindet alle anderen Schichten miteinander und steuert die Struktur und den Inhalt der Daten, die dem Anwender angezeigt werden. Desweiteren kümmert sie sich um die Aufbereitung der Informationen des Users, indem sie diese in spezielle Anfragen (queries) formatiert und der Datenbank vorlegt. Die Middle Tier verflechtet also die Datenbank mit dem Web [Williams and Lane (2002)].

1.4.3 Database Tier

Die Datenbankschicht ist die Basis jeder Datenbank-Applikation. Um eine funktionierende Datenbank-Anwendung zu entwickeln ist ein vernünftiges Design der erste Schritt zum Erfolg. Für die Diplomarbeit wurde das in Abbildung 1.3 Datenbankschema entworfen.

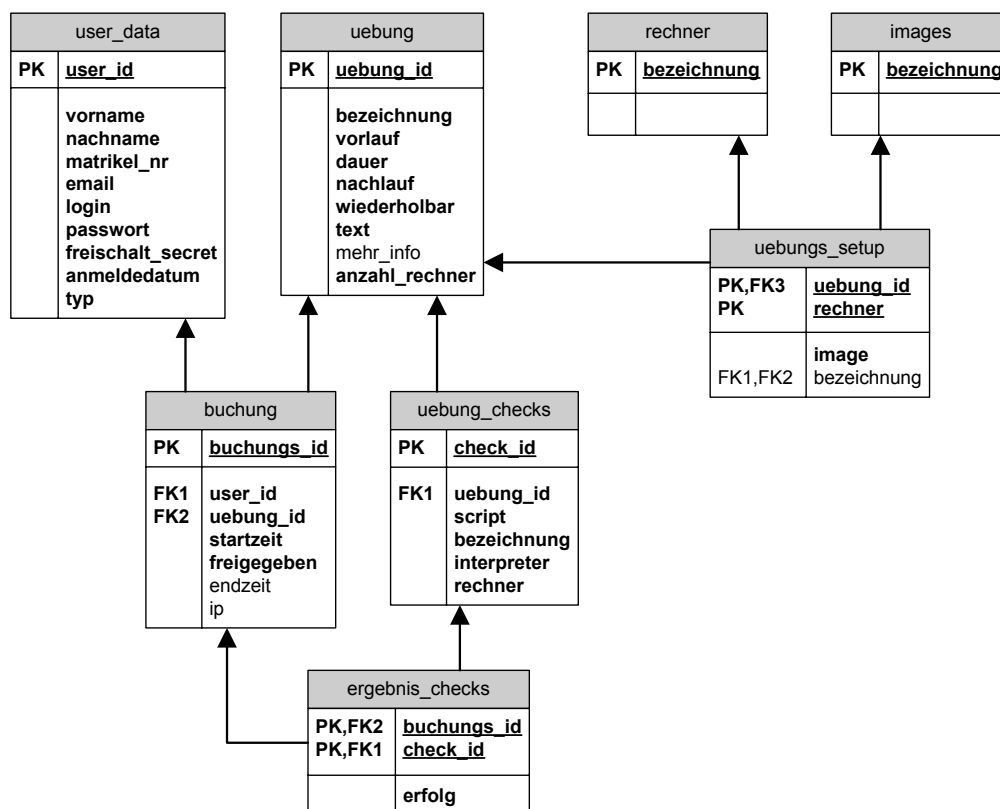


Abbildung 1.3: Datenbankstruktur

Die genaue Beschreibung der Datenbankstruktur sowie dessen SQL-Script zur Erzeugung finden sich in Anhang B.

In einer Three-Tier Architektur [Williams and Lane (2002)] hält die Datenschicht alle erforderlichen Daten. Das Datenbank-Management-System empfängt Informationen und speichert diese. Bei der Sprache innerhalb dieser dritten Schicht handelt es sich oft um SQL. Sie formuliert die Anfragen an die Datenbank. Die darunter liegende Architektur ist für die meisten Anwender uninteressant.

SQL ist die standard-relationale Datenbank-Sprache. Nahezu alle relationalen Datenbank-Systeme - einschließlich PostgreSQL - unterstützen SQL. Die Sprache ist **das** Tool zum Erzeugen und Verwalten einer Datenbank. Alle Anfragen an die Datenbank werden in SQL formuliert. In der Tat ist SQL ein vollständiges Werkzeug, das alle Aspekte im Umgang mit Datenbank-Management berücksichtigt.

1.5 Verwendete Software

1.5.1 Web Server: Apache

Bei dieser Arbeit werden in der Mittelschicht als Komponenten der Web Server Apache mit der Scriptsprache PHP verwendet.

Apache basiert auf dem NCSA² Web Server, der von Rob McCool an der Universität von Illinois entwickelt wurde. Anfang 1995 war er der am meisten verwendete Server. Nachdem Rob McCool das NCSA verlassen hatte, versuchten andere Leute den Web Server zu verbessern. Es wurden immer mehr Patches hinzugefügt und man erhielt einen *gepatchten* Server (engl.: "a patchy server") - woraus dann der Name Apache entstand. Im Dezember 1995 hatte Apache den NCSA Web Server verdrängt und war zum meistgenutzten Web Server geworden [Eilebrecht (1998)].

Apache ist ein Programm, das unter einem geeigneten Multitasking-Betriebssystem läuft. Unabhängig von diesem System, macht es nichts anderes, als an den in der Config-Datei angegebenen IP-Adressen und TCP-Ports zu 'horchen'.

² National Center for Supercomputing Applications,
Universität von Illinois, USA; <http://www.ncsa.uiuc.edu/>

Geht eine Anforderung an einem gültigen Port ein, teilt sich der Masterprozess und behandelt die jeweilige Anfrage. Dabei wendet er Regeln an, die in der Konfigurationsdatei enthalten sind, und führt die entsprechenden Aktionen aus [Laurie and Laurie (1999)].

Der Apache ist "Open-Source"³, d. h. man kann sich den Quellcode des Programms kostenlos besorgen und es einfach auf seinem System kompilieren.

1.5.2 Applikations-Server: PHP

Der Web Server verfügt über eine spezielle Schnittstelle und kann darüber verschiedenste Module einbinden. Dazu gehören unter anderem Sprachmodule wie `mod_perl` oder das hier verwendete `PHP`.

PHP wurde ursprünglich entworfen, um einen dynamischen Web-Inhalt zu erzeugen und ist dazu immer noch am besten geeignet [Lerdorf and Tatro (2002)]. Es ist eine mächtige Sprache, die sich in statisches HTML einbinden lässt - ohne dass der Client etwas davon merkt. PHP arbeitet vollständig auf der Seite des Servers und ist die Basissprache dieser Arbeit.

Die genaue Beschreibung zur Installation der Komponenten in der Mittel-Schicht findet sich in Anhang A.

1.5.3 Datenbank: PostgreSQL

Das bei diesem Projekt verwendete Datenbank Management System (DBMS) ist PostgreSQL. Dabei handelt es sich um ein objekt-relationales System, das nahezu alle SQL Konstrukte versteht.

PostgreSQL wird seit 1977 in unterschiedlichen Formen entwickelt. Begonnen hat alles mit einem Projekt namens Ingres an der 'University of California' in Berkeley. 1986 wurde der Ingres-Code zu einem objekt-relationalen Datenbank-System weiter entwickelt. Es erhielt den Namen Postgres. Im Zuge eines Open Source Projekts bekam Postgres eine erweiterte Funktionalität und wurde zu PostgreSQL umbenannt [Worsley and Drake (2002)].

³ Open Source bedeutet, der Quell-Code des Produkts ist frei verfügbar. Man muss bei Verwendung der Software keine Gebühren an den Hersteller entrichten, und kann bei Bedarf den Code auf seine Bedürfnisse umschreiben und verändern.

PostgreSQL wird heute weitgehend als modernste Open Source Datenbank betrachtet. Sie bietet eine Fülle von Eigenschaften die gewöhnlich nur in kommerziellen Datenbanken wie Oracle zu finden sind. Dazu gehören unter anderem:

- Objekt-relationales DBMS
PostgreSQL bietet deklarative SQL-Anweisungen, Multi-Versions Kontrolle, Multi-User Support, Transaktionen, Query Optimierung, Vererbung und Arrays
- Übergreifende SQL Unterstützung
PostgreSQL arbeitet mit dem Herzstück von SQL99 und umfasst dabei eine Vielzahl von Eigenschaften aus SQL92
- Große Erweiterbarkeit
PostgreSQL unterstützt benutzerdefinierte Operatoren, Funktionen, Zugriffsmethoden und Datentypen
- Process-per-user Architektur
Es gibt einen Masterprozess, der sich bei Bedarf aufspaltet, um zusätzliche Verbindungen für jeden Client zu bieten
- Flexible API
PostgreSQL bietet Entwicklern Schnittstellen zu den Programmiersprachen Python, Perl, PHP, TCL, Pascal, Java/JDBC, Ruby, ODBC, C/C++ und Pike
- Prozedurale Sprache
PostgreSQL unterstützt die Sprache PL/pgSQL. Sie ist vergleichbar mit PG/SQL von Oracle. Weiterhin bietet PostgreSQL die Möglichkeit Perl, Python oder TCL als 'embedded language' zu verwenden.
- ...

Anders als MySQL ist PostgreSQL portabler und läuft auch auf NetBSD/sparc. Eine Beschreibung der Installation und Konfiguration zu PostgreSQL findet sich in Anhang [A.3](#).

Kapitel 2

Architektur

Die Grundlage des 'Virtuellen Unix Labors' ist eine vernünftige Verwaltung und Aufbereitung der von den Benutzern eingegebenen Daten. Dazu wurde eine Datenbankstruktur entworfen, in der alle Anwender gespeichert sind und eine Middle-Tier entwickelt, welche die Datensätze verarbeitet.

Die Einträge der Mitglieder von Hand zu bearbeiten ist sehr aufwendig, deshalb wird sehr viel Wert auf Automatisierung gelegt. Dieses Kapitel diskutiert nun speziell das Konzept zum Aufbau der Projekt-Architektur.

2.1 Neuanmeldung

Einer Person, welche in diesem Labor erstmals arbeiten möchte, soll es ohne großen Aufwand möglich sein, eine Benutzerkennung zu erhalten. Dazu wird ein grafisches Web-Frontend entworfen, das die nötigen Formularitäten übernimmt. Abbildung 2.1 zeigt den programm-technischen Ablauf, wie die Registrierung eines neuen Benutzers vonstatten geht.

Anfangs erhält der noch nicht angemeldete Anwender ein Formular, in das er seine persönlichen Daten einträgt. Wird die Eingabe bestätigt (submit), startet eine Validierung, die jegliche Eingaben auf ihre Korrektheit überprüft. Dabei erzeugen leer gelassenen Felder einen Fehler. Doppelt vergebene eMail-Adressen oder nicht übereinstimmende Passwörter werden ebenfalls abgewiesen. Sofern die eMail-Adresse keine gültige Form hat, bekommt man ebenfalls eine Beschwerde.

Nachdem das Formular keine Fehler mehr aufweist, generiert ein Algorithmus

eine Zeichenfolge (secret), die später zur Identifizierung des Anmeldenden verwendet wird. Der Applikations Server packt nun diese Kennung in eine eMail und sendet sie als Bestätigung an den nun registrierten Benutzer. Schließlich nimmt die Tabelle "benutzer" der Datenbank (siehe Datenbankstruktur in Anhang B) alle Informationen auf und speichert sie.

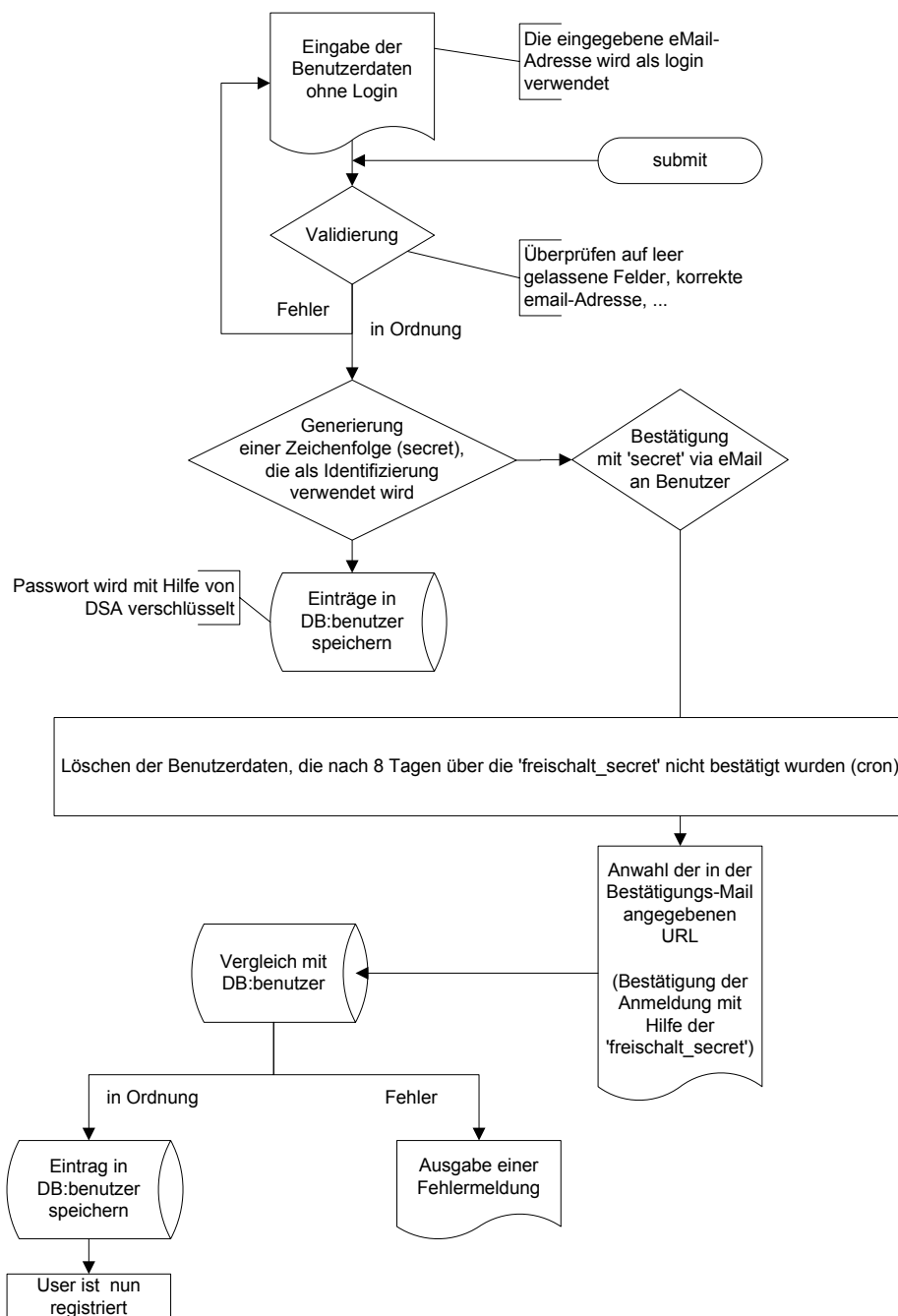


Abbildung 2.1: Ablaufplan der Neuanmeldung (Profil erstellen)

Um Sicherheit zu gewährleisten, wird das gewünschte Passwort nicht im Klartext in der Datenbank abgelegt. Es wird mit Hilfe des Algorithmus DES¹ verschlüsselt, die nicht mehr zurückgewandelt werden kann.

Nach erfolgreichem Ablauf dieser Prozedur ist die jetzt angemeldete Person bereits eingeloggt und befindet sich in ihrem persönlichen Profil. Dort kann sie ihre Benutzerdaten ändern, und sich z. B. ein neues Passwort geben oder seinen Login zu ändern (standardmäßig ist die eMail-Adresse als Login gesetzt).

Ziel des Projekts ist aber auch, die Datenbank sauber zu halten. Dies bedeutet, dass 'pseudo-user', die sich mit falschen Daten angemeldet haben, die Datenbank nicht unnötig belasten. Aus diesem Grund wird der Anwender über ein eMail angewiesen, sein erstelltes Profil zu bestätigen. Dazu ist die ihm übertragene Zeichenfolge (secret) nötig. Wird sie vom User übermittelt, erfolgt eine erneute Validierung. Dabei überprüft der Server die Übereinstimmung der secret und der eMail-Adresse mit den gespeicherten Daten. Ist alles in Ordnung, wird ein Flag gesetzt, welches das neue Anwender-Profil als gültig akzeptiert.

Nicht bestätigte Einträge werden bei einem Alter von mehr als acht Tagen unwiderruflich aus der Datenbank gelöscht. Um diese Aufgabe kümmert sich der Dämon 'cron', der im Hintergrund läuft und zeitgesteuert Kommandos ausführt. Wie 'cron' eingerichtet wird, findet sich im Kapitel [4.3](#) – Säubern der Datenbank.

¹ **D**ata **E**ncryption **S**tandard - Datenverschlüsselungs-Standard von 1974

2.2 Login

Jeder registrierte Anwender besitzt bestimmte Rechte, die es ihm erlauben sich im System zu bewegen. Dazu gibt es drei Berechtigungsstufen:

- user
- admin
- dozent

Dem *user* (normaler Anwender) ist es erlaubt, seine eigenen Daten zu verändern. Er darf sich beispielsweise einen neuen Login-Namen geben oder ein anderes Paßwort setzen. Des weiteren kann er Übungen buchen, die er zu einem festgelegten Zeitraum bearbeiten möchte. Diese Buchungen dürfen von ihm eingesehen werden. Ebenfalls hat er zum Abrufen seiner Ergebnisse eine Berechtigung.

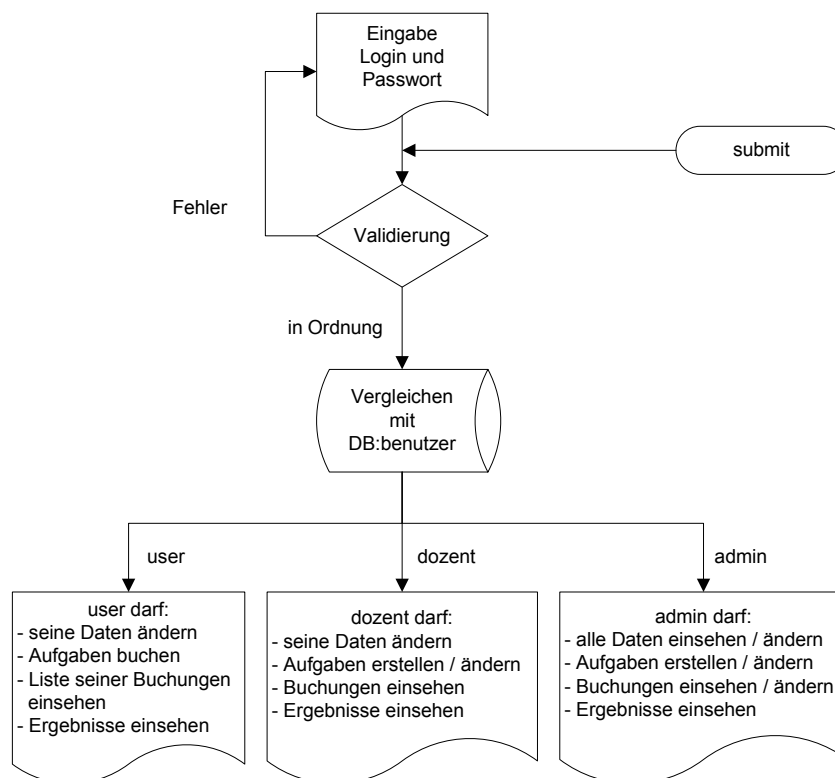


Abbildung 2.2: Ablaufplan des Logins

Als *admin* stehen einem wesentlich mehr Rechte zur Verfügung. Personen mit dieser Berechtigung dürfen alle Benutzerdaten einsehen und können diese bei Bedarf auch ändern. Die Übungen, welche die *user* bearbeiten sollen, werden

vom Administrator erstellt und ins System eingehängt. Es ist ihm erlaubt, bereits getätigte Buchungen zu verändern und er hat zum Schluss natürlich vollen Handlungsspielraum über die Auswertung der Übungen.

Der *dozent* steht in etwa zwischen den beiden. Er kann neue Übungsaufgaben ins System eingeben und darf bestehende Übungen ändern oder löschen. Jedoch bekommt er keinen Zugriff auf die Profile der *user* und hat keine Rechte Buchungen zu verändern.

Login

Bitte geben Sie hier Ihre Daten ein.

Login

Passwort

Sie sind noch nicht registriert !?
Dann muß für Sie zuerst ein [ein Profil](#) angelegt werden.

Abbildung 2.3: Screenshot des Login-Formulars

Damit niemand im 'Virtuellen Unix Labor' seine Berechtigungen überschreitet, ist eine Identifikation nötig. Dies geschieht, wie in Abbildung 2.2 gezeigt, über das normale Login. Der Anwender gibt seine Kennung mit dem dazu passenden Passwort in das dafür vorgesehene Formular ein und sendet (submit) die Daten an den Server. Abbildung 2.3 zeigt einen Screenshot des Login-Formulars. Während der Validierung wird das Passwort verschlüsselt und mit dem in der Datenbank gespeicherten Gegenstück verglichen. Ist das Passwort falsch, so wird der Anfragende abgewiesen. Bei Korrektheit gibt die Datenbank Auskunft über die Berechtigungsstufe des Benutzers und lässt ihn dementsprechend auf die jeweiligen Menüpunkte zugreifen.

2.3 Erstellen von Übungen

Die Übungsaufgaben, welche den Anwendern zu Verfügung stehen sollen, müssen zuvor entworfen und dem System zugeführt werden. Berechtigt dazu ist nur ein Administrator. Über den in Kapitel 3.9 beschriebenen Login-Vorgang

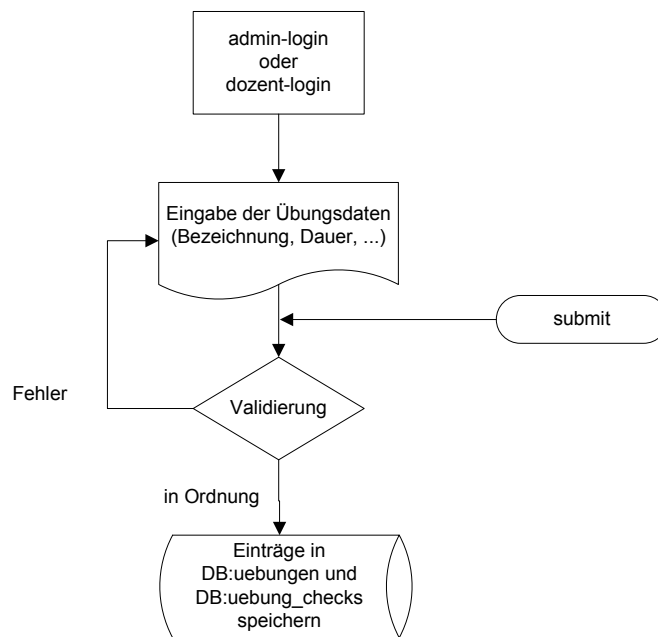


Abbildung 2.4: Ablaufplan zum Erstellen der Übungen

gelangt die befugte Person zu einem Menüpunkt, welcher die Verwaltung der Übungsaufgaben zulässt. An dieser Stelle kann der Admin die bereits bestehenden Übungen betrachten und sie gegebenenfalls abändern.

Weitaus wichtiger ist die Eingabe neuer Übungen. Dazu steht ein Formular zur Verfügung, in das alle relevanten Daten der neuen Übungsaufgaben eingetragen werden müssen. Das zu Grunde liegende "Design des Virtuellen Unix Labors" findet sich unter <http://smaug.fh-regensburg.de/~feyrer/design> [Feyrer (2003)]. Nachdem alle Angaben, wie in Abbildung 2.4, über submit an den Server gegangen sind, startet die Validierung. Dabei werden alle Informationen auf ihre korrekte Form geprüft und der Datenbank übergeben.

2.4 Buchen der Übungen

Die zur Verfügung stehenden Übungen können auf den Übungsrechnern größtenteils nur unter der Kennung *root* ausgeführt werden. Damit sich dabei nicht verschiedene Anwender gegenseitig behindern, lässt das Verwaltungssystem nur eine Person gleichzeitig zu. Um dies koordinieren zu können, ist eine rechtzeitige Reservierung der Übungen nötig. Den Ablauf zum Buchen einer Übung zeigt Abbildung 2.5.

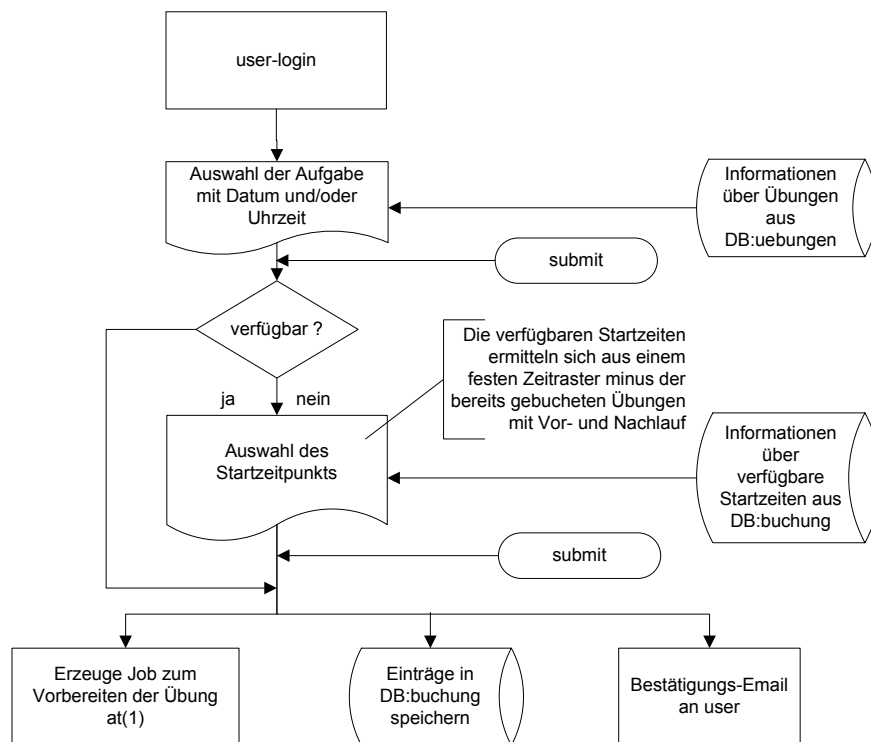


Abbildung 2.5: Ablaufplan zum Buchen der Übungen

Dazu muß sich der *user* zuerst mit Hilfe des Login-Vorgangs (siehe Kapitel 3.9) identifizieren. Danach gelangt er über den Menüpunkt "Übungen buchen" zu einer Auswahl von Übungen, welche die Middle-Tier aus der Datenbank gelesen hat. Der Benutzer entscheidet sich für eine Übung und gibt das Datum, die Uhrzeit oder beides an, wann er sie bearbeiten würde.

Hat sich der Anwender nicht konkret für einen Zeitraum entschieden oder ist der gewählte Zeitabschnitt bereits vergeben, werden die noch zur Verfügung stehenden Zeitfenster im Browser angezeigt. Nachdem der Benutzer eine endgültige Startzeit ausgesucht und diese bestätigt hat, ist die Buchung abgeschlossen.

Der Server veranlasst zum Schluss noch die Erzeugung eines `at(1)`-Jobs², welcher sich um die Vorbereitung des Computer-Systems zum gebuchten Zeitpunkt kümmert. Weiterhin werden alle notwendigen Informationen der Reservierung in der Datenbank abgelegt und der Anwender erhält eine BestätigungseMail.

² Das Kommando `at(1)` verarbeitet Kommandos zu einem festgelegten späteren Zeitpunkt.

2.5 Ablauf der Übung

Das Bearbeiten der gebuchten Übung gestaltet sich etwas umfangreicher (siehe dazu "Kurs-Engine" im Design-Dokument unter <http://smaug.fh-regensburg.de/~feyrer/design>) [Feyrer (2003)]. Wie in Kapitel 2.4 erwähnt, darf immer nur ein Benutzer auf die Übungsrechner zugreifen. Um dies realisieren zu können, schottet eine Firewall die Übungsrechner vom Internet ab. Abbildung 1.1 auf Seite 2 zeigt die bereits besprochene Netzwerkstruktur. Aus diesem Grund wird die Firewall dynamisch konfiguriert. Und zwar so, dass nur eine bestimmte Quell-IP-Adresse akzeptiert wird. Der Inhaber dieser IP-Adresse ist der Benutzer, der die Übung bearbeiten möchte, und nur er wird über die Firewall auf das Testsystem zugelassen. Alle anderen Computer werden abgewiesen.

Damit die Übung fehlerfrei abläuft, ist einiger Aufwand nötig, der in Abbildung 2.6 grafisch dargestellt ist.

Am Anfang steht natürlich wieder die Identifizierung des *users* via Login. Der Menüpunkt "Übung bearbeiten" im Web-Frontend gibt Auskunft darüber, ob eine Übungsaufgabe reserviert wurde, und diese zur Bearbeitung freigegeben ist. Die Informationen dazu gewinnt der Applikations-Server aus der Datenbank. Liegt keine Reservierung vor oder ist die Vorbereitung der Übung noch nicht abgeschlossen, so wird der Benutzer mit einer entsprechenden Meldung abgewiesen.

Bei einer bereits freigegebenen Übung übermittelt der Browser des Anwenders die IP-Adresse des Client-Rechners an die Middle-Tier. Sie kann gegebenenfalls über ein Formular geändert bzw. neu eingegeben werden.

Diese IP-Adresse wird benötigt, um die Firewall für den Benutzer freizuschalten. Dazu wird die Firewall so umkonfiguriert, damit nur die vom Anwender eingehende Adresse beim Übungssystem akzeptiert wird. Alle anderen Benutzer, welche versuchen darauf Zugriff zu bekommen, werden abgewiesen. Dadurch wird erreicht, dass nur die eine berechnete Person ungestört arbeiten kann, und andere *user* nicht mit ihr kollidieren.

Nach erfolgreichem Ablauf des Deployment-at(1)-Jobs, der beim Buchen der Übung erzeugt wurde (siehe Kapitel 2.4), zeigt das Flag 'bereit' in der Daten-

bank an, dass der Zugang bereitgestellt ist. An dieser Stelle startet erneut ein `at(1)`-Job, welcher die Übung überwacht. Denn es ist möglich, dass der Anwender das Entwicklungssystem beschädigt und dann kein Zugriff mehr gewährleistet ist. Deshalb muss zu dieser Zeit ein Reset durchgeführt werden, bzw. ist es notwendig, das Image des Betriebssystems wieder herzustellen. Desweiteren wird ein `wall(1)`-Job³ erzeugt, der dem Benutzer Informationen über seinen aktuellen zeitlichen Stand vermittelt. Es ist z. B. geplant, eine Meldung auszugeben, wenn dem Anwender nur noch fünf oder zehn Minuten zum Abschluss der Übung bleiben.

Der Benutzer kann nun die Übungsaufgabe in Angriff nehmen, nachdem alle zum Start der Übung notwendigen Server-Aktivitäten abgeschlossen sind. Jetzt erst bekommt er die Aufgabenstellung zu Gesicht und beginnt mit der Bearbeitung.

Bei der Bereitstellung der Übung über den Dozenten oder Administrator wurde eine maximale Bearbeitungszeit vorgegeben. Ist diese verstrichen, wird der Anwender automatisch ausgeloggt. Er kann aber bereits vor diesem Timeout, über einen Schalter in seinem Browser, seine Aktivitäten vorzeitig beenden. Im Anschluss zeigt ihm eine Informations-Seite, dass nach einer Nachlaufzeit das Ergebnis in Form eines Reports abgeholt werden kann.

Die Abläufe für die Nacharbeit sind in einem 'Übungs-Ende'-Script verankert (siehe Abbildung 2.6). Zuerst muß die userspezifische Konfiguration der Firewall rückgängig gemacht werden. Weiterhin wird das Flag 'bereit' in der Datenbank zurück gesetzt. Die in der Datenbank gespeicherten Check-Skripten analysieren danach die Ausführungen des Benutzers auf bestimmte Kriterien. Die dabei erhaltenen Ergebnisse werden für spätere Zugriffe in der Datenbank gesichert. Zu guter Letzt bekommt der Anwender die Informationen über die abgehandelte Übung per eMail zugesandt.

³ Die Anweisung `wall(1)` gibt Nachrichten an alle eingeloggten *user* aus.

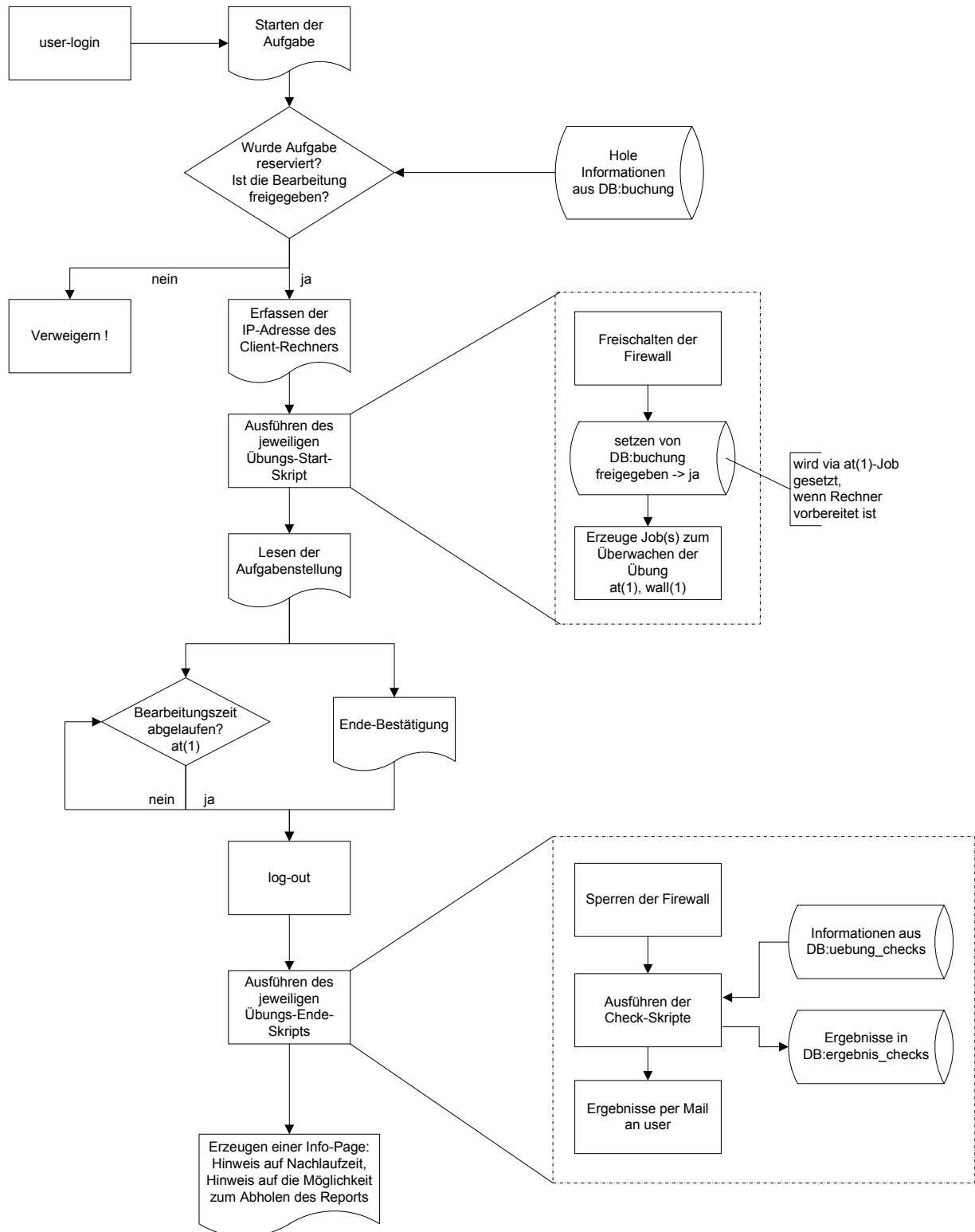


Abbildung 2.6: Ablauf der Übung

Kapitel 3

Programmierung

Wie bereits in einem früheren Kapitel erwähnt wurde, handelt es sich bei der Programmierung des Applikationsservers um die Script-Sprache PHP. Sie ist in der Lage, Web-Inhalte für den Client zu erstellen und wird deshalb zur Erzeugung der Benutzeroberfläche verwendet.

3.1 Probleme zwischen den Versionen

3.1.1 Globale Variablen – Ein Sicherheitsrisiko ?

Da die Entwicklung von PHP nicht still steht und sich die Eigenschaften der Sprache langsam ändern, stößt man immer öfters auf Probleme. Diese betreffen in erster Linie die Portierbarkeit des Codes. So wurde zum Beispiel mit der Version 4.0 die Möglichkeit geschaffen globale Variablen zu verwenden. Dazu ist in der Initialisierungs-Datei `php.ini` das Flag `register_globals` auf `on` gesetzt.

```
register_globals = On
```

Häufig werden globale Variablen verwendet, um Zugriffe auf Hash-Inhalte einfacher zu handhaben. In PHP existieren standardmäßig mehrere Hash-Tabellen:

- `$HTTP_ENV_VARS`
- `$HTTP_GET_VARS`
- `$HTTP_POST_VARS`
- `$HTTP_COOKIE_VARS`
- `$HTTP_SERVER_VARS`

Um auf dessen Inhalte zuzugreifen, gibt man in der Regel die Tabelle mit einem Schlüsselwert an. Im nachfolgenden Beispiel wird im Hash `'$HTTP_SERVER_VARS'` nach dem Eintrag zum Schlüsselwort `'DOCUMENT_ROOT'` gesucht.

```
<?php
    echo $HTTP_SERVER_VARS[ 'DOCUMENT_ROOT' ];
?>
```

Verwendet man nun globale Variablen, kann direkt auf den Inhalt, auf den der Schlüssel `'$DOCUMENT_ROOT'` verweist, zugegriffen werden.

```
<?php
    echo $DOCUMENT_ROOT;
?>
```

Das scheint die Programmierung um einiges zu vereinfachen. Doch was passiert, wenn das selbe Schlüsselwort in verschiedenen Hash-Tabellen verwendet wird?

Betrachten wir folgende Situation:

Damit ein Benutzer auf bestimmte Inhalte eines Servers zugreifen kann, muss er sich /indexauthentifizieren authentifizieren. Um zu vermeiden, dass dieser Prozess der Authentifikation bei jeder Anfrage erneut stattfindet, wird der User anfangs einmal überprüft und das Ergebnis für die komplette Sitzung gespeichert. In der Hash-Tabelle `'$HTTP_SESSION_VARS'` existiert somit ein Schlüssel `'auth'`, welcher auf `'wahr'` oder `'falsch'` gesetzt ist. Er verkörpert die Zuverlässigkeit des Benutzers. Ruft der Anwender nun Informationen ab, so überprüft PHP innerhalb einer gültigen Sitzung, mit Hilfe globaler Variablen, den Schlüssel-Wert `'auth'`.

```
<?php
    ...
    if ($auth == true)
        // authentifiziert
    ...
?>
```

Ein Anwender, dessen Session-Variable `'auth'` auf `'false'` steht, bekommt keinen Zugriff auf die gewünschten Informationen. Deshalb simuliert er im Web-Browser einen Aufruf der Methode `GET`.

```
http:// ... /index.php?auth=true
```

Dieser Aufruf erzeugt in der Hash-Tabelle '`$HTTP_GET_VARS`' einen Schlüssel mit Namen '`auth`', welcher als Inhalt den Wert '`true`' hält.

Da globale Variablen verwendet werden, kann auch hier direkt auf den Inhalt des Schlüssels '`auth`' zugegriffen werden. Welches Ergebnis an dieser Stelle die vorherige PHP-Abfrage liefert, ist nicht mehr sicher.

```
<?php
...
if ($auth == true)
    // authentifiziert ?
    // $auth aus $HTTP_SESSION_VARS oder
    // $auth aus $HTTP_GET_VARS ?
...
?>
```

Das Problem wurde erkannt. In der Dokumentation zu PHP [\[php \(2002\)\]](#) heißt es, dass aus Sicherheitsgründen das Flag `register_globals` in der Initialisierungs-Datei `php.ini` ab Version 4.2 als 'deprecated' — abgelehnt — beschrieben wird.

Sicherer und auch unproblematischer beim Verwenden des selben Programm-Codes auf verschiedenen PHP-Versionen ist deshalb folgende Zeile:

```
<?php
...
if ($HTTP_SESSION_VARS['auth'] && !$HTTP_GET_VARS['auth'])
    // authentifiziert
...
?>
```

Sie greift direkt auf den gewählten Hash zu. So gibt es keine Verwechslungen.

3.1.2 Pufferung des Ausgabestroms

Nicht nur, dass sich viele verschiedene PHP-Versionen auf den unterschiedlichsten Servern finden lassen, jede Installation ist auch noch anders konfiguriert. So gibt es in der Initialisierungs-Datei `php.ini` einen Eintrag, welche den Puffer

des Ausgabestroms steuert.

Um eine besser Leistung zu erhalten, wird dieser Puffer gerne aktiviert und auf 4 kByte gesetzt.

```
output_buffering = 4096
```

Jedoch resultiert daraus der Nachteil, dass mancher PHP-Code nicht mehr auf andere PHP-Versionen portiert werden kann.

PHP bietet die Möglichkeit zu jedem Zeitpunkt HTTP-Header-Informationen zu erzeugen. Allerdings dürfen diese nur in einer bestimmten Reihenfolge an den Client übertragen werden.

Wie in Abbildung 3.1 zu sehen ist, steht der Header – mit allen Informationen über das Dokument – vor dem eigentlichen Inhalt. Wird der HTTP-Body bereits übertragen, können keine Fragmente oder gar ein kompletter Header nicht mehr gesendet werden.

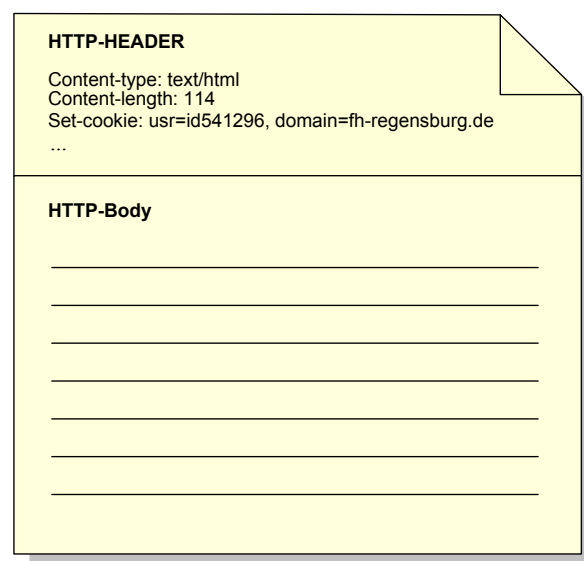


Abbildung 3.1: HTTP-Dokument

Bei einem aktiven Ausgabe-Puffer wird, wie der Name sagt, jegliche Ausgabe, vor der Übertragung in einen Puffer geschrieben. Die komplette HTML-Seite wird so zuerst am Server zusammengestellt und erst zum Schluss an den Client übermittelt. Deshalb ist es somit möglich, auf Seiten des Servers den HTTP-Header erst später zu erzeugen, als den Inhalt selbst.

Darum können auf diesem Weg an jeder Stelle im PHP-Code Cookies oder sogar Session erzeugt werden. Dieser Vorgang ist zwar nicht verboten, aber verkörpert einen sehr schlechten Programmier-Stil.

Bei abgeschaltetem Ausgabe-Puffer

```
output_buffering = 0
```

oder

```
output_buffering = Off
```

Ist dieses Verfahren nicht mehr möglich. Die Ausgabe des Servers kann vorher nicht "sortiert" werden. Darum ist darauf zu achten, dass der Seitenaufbau korrekt von statten geht.

Für gewöhnlich sollte jedes PHP-Programm auf jedem PHP-Server lauffähig sein — egal wie er konfiguriert ist. Dies ist immer der Fall, wenn sich an die Regeln gehalten wird. Die meisten Erleichterungen führen jedoch zu einem nicht mehr portierbaren Programm-Code.

3.2 Seitenaufbau

Bei dieser Arbeit handelt es sich in erster Linie um die Benutzerverwaltung des "Virtuellen Unix Labors". Das bedeutet, dass alle wichtigen Informationen der Anwender in einer Datenbank gehalten und grafisch administriert werden. Deshalb wurde ein relativ modulares Front-End gewählt, so dass in späteren Weiterentwicklungen viele Funktionalitäten als einzelne Pakete in das System integriert werden können.

Der Aufbau der Web-Seite ist darum an ein festes Design gebunden. Alle verwendeten und zur Funktion beitragenden Module greifen in keinsten Weise auf das Seitenlayout zu und können es auch nicht verändern. Abbildung 3.2 zeigt den schematischen Aufbau der Seite.

Die dazu startende Datei `index.php` ist auszugsweise im Listing 3.1 zu sehen. Jeder Verweis auf einen anderen Inhalt ruft immer wieder diese Datei oder `index_session.php` auf. Bei `index_session.php` handelt es sich um einen

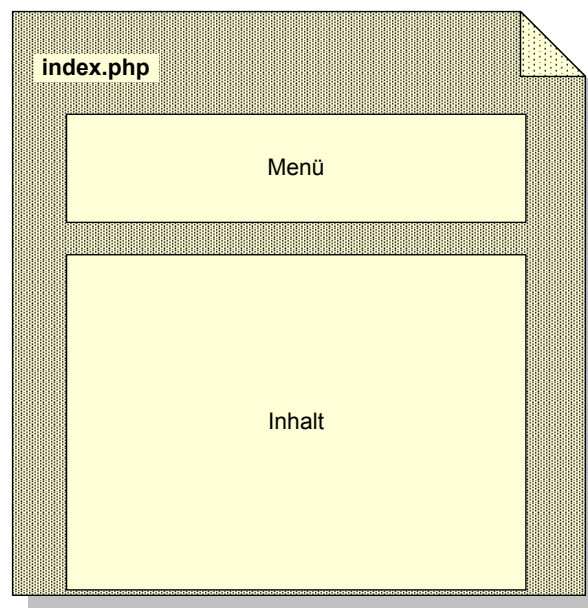


Abbildung 3.2: Prinzipieller Seitenaufbau — `index.php`

ähnlichen Aufbau wie `index.php`, allerdings wurde sie um das Session-Handling erweitert. Nähere Ausführungen aber dazu in Kapitel 3.3.

Werfen wir einen Blick auf Listing 3.1 – `index.php`. Zeile 4-7 liest aus der Umgebungsvariable den Inhalt des Schlüssels `menu` bzw. den Inhalt des Schlüssels `content` in Zeile 10-13. Diese Werte werden etwas später im Quellcode zum Aufbau der Seite benötigt.

Eine kleine Hilfestellung zum Lesen der Umgebungsvariable gibt dabei die in der Datei `functions.php` enthaltenen Funktion `getValue($method, $key)`. Ihren Quellcode finden Sie im Anhang E.1 im Listing E.7.

Zurück zu Listing 3.1 – `index.php`. Zeile 4 versucht den Inhalt aus der Hash-Tabelle `GET` zu lesen. Wird kein Ergebnis erzielt, so findet sich der Wert im Hash `POST` – Zeile 7. Analog dazu belegen wir die Variable `$content`.

In Zeile 17 beginnt nun der Aufbau der HTML-Seite. Nachdem die ersten Zeilen des "Bodys" an den Client gesendet wurden, findet der Server in Zeile 24 einen kurzen PHP-Abschnitt, der das Menü einbinden soll. Der Code in Zeile 31 lässt zu guter letzt den Inhalt in die Seite einhängen. Mehr dazu in den nachfolgenden Kapiteln 3.2.1 und 3.2.2.

Listing 3.1: index.php

```
1 <?php
2 ...
3 // hole Informationen ueber Menu
4 $menu = getValue("GET", "menu");
5
6 if (empty($menu))
7     $menu = getValue("POST", "menu");
8
9 // hole Informationen ueber Link
10 $content = getValue("GET", "content");
11
12 if (empty($content))
13     $content = getValue("POST", "content");
14 ...
15 ?>
16
17 <html>
18 <head>
19     <title>Virtuelles UNIX labor</title>
20     ...
21     <link rel="stylesheet" href="vulab.css" type="text/css">
22 </head>
23 <body ...
24     <?php
25         if (!empty($menu))
26             echo menu($menu);
27         else
28             echo menu("default");
29     ?>
30 ...
31 <?php
32
33     if (!empty($content))
34     {
35         $contentLink = getContentLink($content, "default");
36
37         if(!empty($contentLink))
38             include $contentLink;
39         else
40             include "notFound.php";
41     }
42     else
43     {
44         $contentLink = getContentLink("home", "default");
45
46         if(!empty($contentLink))
47             include $contentLink;
48         else
49             include "notFound.php";
50     }
51     ?>
52 ...
53 </body>
54 </html>
```

3.2.1 Menü

Der Aufbau des Menüs ist etwas kompliziert. Um die Funktion `menu($userType)` der Funktions-Sammlung `functions.php` zu untersuchen, sollte man erst die Datei-Struktur des Projektes verstanden haben. Abbildung 3.3 zeigt den modularen Aufbau.

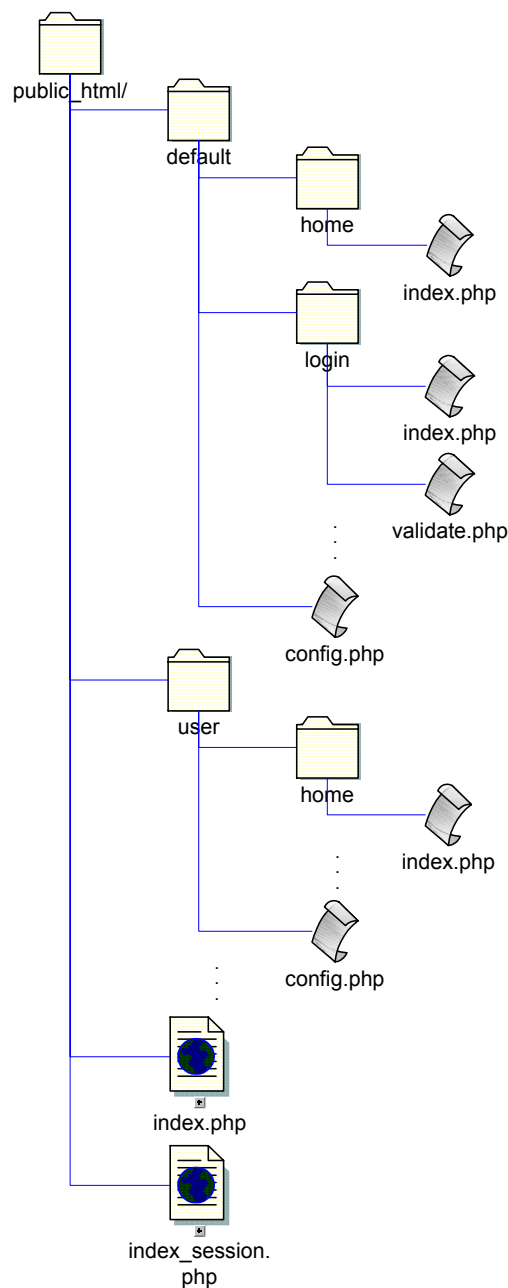


Abbildung 3.3: Datei-Struktur des Projekts

Neben den Dateien `index.php` und `index_session.php` befinden sich noch weitere wichtige Scripte im Wurzel-Verzeichnis `public_html`, welche hier aber

nicht näher erwähnt werden. Ferner findet man Ordner wie `user`, `admin`, `dozent` und `default`. Sie bezeichnen den Benutzertyp. Innerhalb dieser Ordner entdeckt man weitere Unterverzeichnisse. Sie repräsentieren die eigentlichen Module. Das zu jedem Benutzertyp gehörende PHP-Script `config.php` beschreibt dessen Inhalt. In der untersten Ebene der Verzeichnis-Struktur muss immer eine `index.php` liegen. Sie verkörpert den Einsprungpunkt in das jeweilige Modul.

Wie in Abbildung 3.3 ersichtlich ist, gibt er für den `user` ein Modul `home`. Weiterhin enthält der Typ `default` neben dem gleichnamigen Modul `home` eines mit Namen `login`. Abbildung 3.3 gibt allerdings nur auszugsweise die Inhalte wieder, da hier lediglich das Prinzip erklärt werden soll.

In dem Verzeichnis `default` findet sich eine Datei mit Namen `config.php`. Sie dient zur Konfiguration des `default`-Menüs. Listing 3.2 zeigt den Aufbau mit allen Einträgen.

Listing 3.2: Konfigurationsdatei `default/config.php`

```
1 <?php
2
3 // Menue-Eintraege in Reihenfolge ihres Auftretens
4 $items = array();
5 $items[1] = "home";
6 $items[2] = "login";
7 $items[3] = "Informationen";
8
9 // Zuweisung der Verzeichnisse auf die Menue-Namen
10 // $access[Menue-Name] = Verzeichniss-Name
11 $access = array();
12 $access["Informationen"] = "info";
13 $access["home"] = "home";
14 $access["login"] = "login";
15 $access["firstLogin"] = "firstLogin";
16 $access["bestaetigen"] = "bestaetigen";
17 $access["error"] = "error";
18
19 // Zuweisung der Beschreibung auf die Menue-Namen
20 // $description[Menue-Name] = Beschreibung
21 $description = array();
22 $description["home"] = "Link auf diese Seite";
23 $description["login"] = "Einloggen des Benutzers in das " .
24     "virtuelle Unix Labor";
25 $description["Informationen"] = "Informationen &uuml;ber das " .
26     "virtuelle Unix Labor";
27 ?>
```

Die Variable `$items` in den Zeilen 5-7 setzt die Menü-Einträge in der Reihenfolge ihres Auftretens beim Client von links nach rechts. Über den Hash `$access` (Zeile 12-17) wird die Zuweisung des jeweiligen Menüpunkts auf das entspre-

chende Modul bzw. dessen Verzeichnis-Namen vorgenommen. Desweiteren kann eine Beschreibung der einzelnen Menüeinträge erfolgen. Sie werden hier in die Hash-Tabelle `$description` in Zeile 22-26 eingegeben, damit später z. B. das Modul `home` darauf zurückgreifen kann. Abbildung 3.4 zeigt das eingebettete Menü im Punkt `home`.



Abbildung 3.4: Eingebettetes Menü des Moduls `home`

In Zeile 26 bzw. in Zeile 28 des Listings 3.1 `index.php` auf Seite 25 wird nun die Funktion `menu($userType)` aufgerufen. Der `$userType` gelangt in unserem Beispiel über die Umgebungsvariable `GET` zum Server. Dem Benutzertyp entsprechend werden dann die jeweiligen Menü-Punkte in die HTML-Seite eingetragen. Wenn kein `$userType` gelesen werden kann, wird stattdessen `default` verwendet.

Diesen Benutzertyp nimmt die Funktion `menu($userType)` auf und sucht sich danach die dazugehörige Konfigurations-Datei `config.php`. Listing E.7 in Kapitel E.1 zeigt ab Zeile 525 wie dies geschieht.

3.2.2 Seiteninhalt

Da nun das Menü zur Verfügung steht, sollten die dadurch verfügbar gewordenen Inhalte auch angezeigt werden können. Das Basis-PHP-Script `index.php` (siehe Seite 25) im Wurzelverzeichnis `public_html` liest im PHP-Block ab Zeile 31 den entsprechenden Seiteninhalt ein.

Wurde ein bestimmtes Modul über das Menü angefordert, so ist die Variable `$content` nicht leer. Die Überprüfung `!empty($content)` in Zeile 33 liefert "true". Deshalb erfolgt in Zeile 35 der Aufruf `getContentLink($content, "default")`.

Das Script speichert den so erhaltenen Pfad in `$contentLink` und hängt den Inhalt – so fern verfügbar – in das Design ein (Zeile 38). Tritt dabei ein Fehler auf, so wählt PHP den Inhalt von `notFound.php` und zeigt an, dass die Seite nicht gefunden wurde.

Sollte aus irgend einem Grund die Variable `$content` anfangs nicht belegt sein, holt Zeile 44 des `index`-Scripts den Pfad auf das Moduls "home".

Was bei dem Aufruf der Funktion `getContentLink($content, $userType)` in Zeile 35 bzw. 44 des `index`-Scripts 3.1 auf Seite 25 passiert, zeigt Listing 3.3.

Listing 3.3: `functions.php: getContentLink($content, $userType)`

```
1 function getContentLink($content, $userType)
2 {
3     $link = "";
4     $file = "";
5
6     if (file_exists($absoluteProjectPath . $userType . "/config.php"))
7     {
8         include $absoluteProjectPath . $userType . "/config.php";
9
10        $pos=(int) strpos($content, '/');
11
12        if ($pos == FALSE)
13        {
14            if (array_key_exists($content, $access))
15            {
16                $subdir = $access[$content];
17                $file = $absoluteProjectPath . $userType . "/" . $subdir . "/index.php";
18            }
19        }
20        else
21            $file = $absoluteProjectPath . $userType . "/" . $content . ".php";
22    }
```

```

23     if (file_exists($file))
24         $link = $file;
25     }
26     else
27         $link = $absoluteProjectPath . "noConfig.php";
28
29     return $link;
30 }

```

Im Funktionskopf sind zwei Variablen enthalten. Während `$content` den Namen des gewünschten Moduls enthält, handelt es sich bei der Variable `$userType` um den Typ, der mit identischem Namen den übergeordneten Ordner benennt (vgl. Abbildung 3.6 bzw. 3.3).

Wie die Datei-Struktur der beiden Abbildungen zeigt, ergibt sich der Zugriff auf das Modul in Form von `$userType/$content/index.php`

Abbildung 3.5 verdeutlicht den internen Ablauf der Funktion `getContentLink()` anhand eines Blockdiagramms.

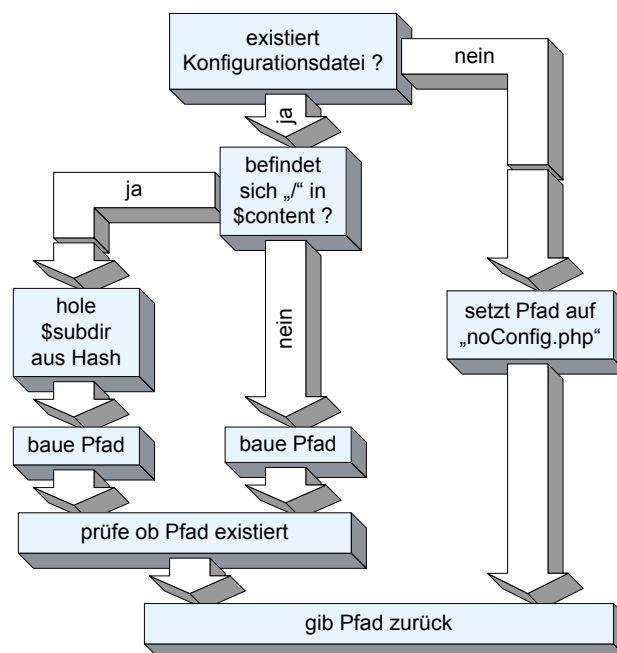


Abbildung 3.5: Blockdiagramm zum Ablauf der Funktion `getContentLink($content, $userType)`

Nachdem die Funktion angelaufen ist, lädt PHP in Zeile 8 die dem `$userType` entsprechende Konfigurationsdatei. Danach sucht das Script in der Variable `$content` die erste Position des evtl. enthaltenen Zeichens `/` (Zeile 10). In Zeile

12 wird anschließend ausgewertet, ob das Zeichen "/" auch wirklich enthalten ist. Liefert `$pos` den Wert `FALSE`, so kommt dieses Zeichen nicht vor. Es handelt sich deshalb um einen Modulnamen, welcher speziell aufgelöst werden muss.

Dazu sieht PHP in Zeile 14 in die Hash-Tabelle `$access`, welche über die Konfigurationsdatei verfügbar wurde, und kontrolliert, ob die Tabelle einen Schlüssel mit diesem Modulnamen enthält. Ist das der Fall, wird der Inhalt, auf den dieser Schlüssel verweist, in `$subdir` geschrieben (Zeile 16). Danach setzt sich der Pfad zusammen, welcher auf die `index`-Datei des gewünschten Moduls zeigt.

```
$userType/$subdir/index.php
```

Handelt es sich beim Inhalt der Variable `$content` um eine Pfadangabe, muss anders vorgegangen werden. In Zeile 21 wird der Verweis auf die jeweilige Datei direkt zusammengesetzt.

Zeile 23 überprüft zur Sicherheit, ob diese Datei auch wirklich existiert und kopiert den Pfad schließlich in die resultierende Variable `$link`, die letztendlich zurückgegeben wird.

Konnte aus irgendeinem Grund die Konfigurationsdatei nicht gefunden werden, liefert PHP in Zeile 27 den Pfad auf die Datei `noConfig.php`, welche eine entsprechende Fehlermeldung enthält.

Nachfolgende Beispiele zeigen anhand der Verzeichnis-Struktur in Abbildung 3.6, wie die Funktion `getContentLink` arbeitet.

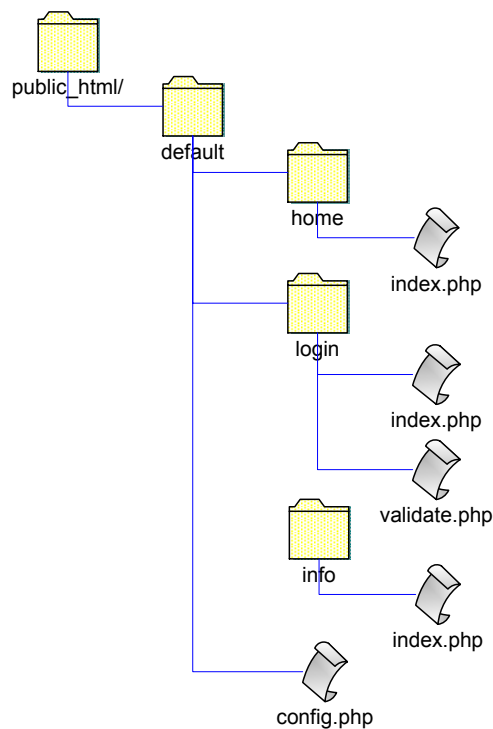


Abbildung 3.6: Verzeichnis-Struktur die das Beispiel verwendet

Ausgangssituation der Beispiele:

```

$access["home"] = "home";
$access["login"] = "login";
$access["Informationen"] = "info";

```

Beispiel 1: Ein Modul wird geladen

Eingebettet wird das Modul "Informationen" des Typs "default".

```
getContentLink("Informationen", "default");
```

Variablen sind gesetzt:

```

→ $content = "Informationen"
→ $userType = "default"

```

In "Informationen" ist kein "/" enthalten.

```

⇒ $subdir = "info"
⇒ $file = "default/info/index.php"

```

Überprüfen: Datei existiert !!

```
⇒ $link = "default/info/index.php"
```

Beispiel 2: Direktzugriff auf ein Script

Es soll das Script "login/validate.php" geladen werden.

```
getContentLink("login/validate", "default");
```

Variablen sind gesetzt:

```
→ $content = "login/validate"
```

```
→ $userType = "default"
```

In "login/validate" findet sich ein "/".

```
⇒ $file = "default/login/validate.php"
```

Überprüfen: Datei existiert !!

```
⇒ $link = "default/login/validate.php"
```

3.3 Session-Handling

Es ist nicht möglich, die gesamte Funktionalität des Projekts in **eine** HTML-Seite oder **ein** PHP-Script zu stecken. Zahlreiche Dateien spielen zusammen, um dieses komplexe Vorhaben aufzubauen. Dazu müssen oft Informationen über mehrere Scripte hinweg erhalten bleiben. Um die Daten nicht immer von einem Script zum nächsten übertragen zu müssen, speichert man diese Informationen temporär auf dem Server.

Allerdings werden die Formulare des "Virtuellen Unix Labors" nicht nur von einer Person verwendet. Verschiedene Anwender erzeugen unterschiedliche temporäre Dateien auf dem Server, welche immer wieder identifiziert werden müssen. Deshalb bekommen diese zwischengespeicherten Dateien eine eindeutige Bezeichnung in Form einer Buchstaben-Zahlen-Kombination — die sog. Session-ID.

Allein diese Kennung reicht aus, um Zugriff auf alle gespeicherten, temporären Informationen zu bekommen. Darum wird zum scriptübergreifenden Austausch der Daten nur die Session-ID benötigt. Dieses Verfahren nennt man Session-Handling.

Da alle Session-Informationen im Header des zu übertragenden HTTP-Dokuments stehen müssen, darf vor dem Erzeugen der Session keine HTML-Ausgabe erfolgen. Dies wurde bereits in Kapitel 3.1.2 – Pufferung des Ausgabestroms – angesprochen. Abbildung 3.1 auf Seite 22 zeigt anschaulich den Aufbau eines HTTP-Dokuments.

Aus diesem Grund wurde speziell für das Session-Handling eine eigene Start-Datei entwickelt — das Script `index.session.php`. Es kann eine bestehende Sitzung übernehmen, aber auch bei Bedarf eine Neue erzeugen. Des weiteren erkennt dieses Script Manipulationen, die einen Benutzer unberechtigterweise auf Daten zugreifen lassen, und geht dagegen vor. Wie dies geschieht kann man Listings 3.4 entnehmen.

Zu Beginn liest PHP in Zeile 6 bzw. 8 die Session-ID. Sie wird von PHP standardmäßig mit `$PHPSESSID` bezeichnet. Zeile 12 startet die Sitzung. War zu diesem Zeitpunkt keine `$PHPSESSID` verfügbar, wird eine neue Session-ID generiert und diese auf die Variable `$PHPSESSID` zurückgeschrieben. Eine bestehende Sit-

zung wird weiter verwendet, wenn vor dem Aufruf `session_start()`; eine Session-ID gelesen worden konnte.

Listing 3.4: index_session.php

```

1 <?php
2     ...
3
4     // hole Session-Variable
5     if (isset($_GET["PHPSESSID"]))
6         $PHPSESSID = $_GET["PHPSESSID"];
7     elseif (isset($_POST["PHPSESSID"]))
8         $PHPSESSID = $_POST["PHPSESSID"];
9
10    // hole vorhandene Session
11    // falls keine Session verfuegbar, erzeuge neue Session
12    session_start();
13    $PHPSESSID=session_id();
14
15    // hole Session-Variablen
16    if (isset($_HTTP_SESSION_VARS["login"]))
17        $login = $_HTTP_SESSION_VARS["login"];
18
19    if (isset($_HTTP_SESSION_VARS["auth"]))
20        $auth = $_HTTP_SESSION_VARS["auth"];
21    else
22        $auth = false;
23
24    // hole Informationen ueber Menu
25    ...
26
27    // URL zum Logout
28    $logoutUrl = getLocation("logout.php") .
29        "?PHPSESSID=" . $PHPSESSID;
30
31    // falls keine Informationen ueber Seiten-Inhalt vorhanden,
32    // erfolgt ein logout
33    if (empty($menu) || empty($content))
34    {
35        header ("Location: " . $logoutUrl );
36        exit;
37    }
38
39    // falls der user nicht authentifiziert ist, erfolgt ein logout
40    if ($menu != "default" && $auth == FALSE)
41    {
42        header ("Location: " . $logoutUrl );
43        exit;
44    }
45
46    // falls jemand eingeloggt ist und auf ein nicht-berechtigtes
47    // Menue zugreifen moechte, erfolgt ein logout
48    if (!empty($login))
49    {
50        // baue Verbindung zur Datenbank auf
51        $connection = getConnection();
52

```

```

53 // frage nach userType
54 $sql = "select userType from userdata where login='" .
55     $login . "'";
56 $result = pg_exec($connection, $sql);
57
58 // holt die erste Zeile des Resultats
59 $array = fetchArray($result, 0);
60
61 // ueberpruefe ob angefordertes Menue mit Berechtigung
62 // ueberein stimmt
63 if (!$auth || $array['usertype'] != $menu)
64 {
65     header ("Location: " . $logoutUrl );
66     exit;
67 }
68 }
69 ?>
70
71 <html>
72     ...
73 </html>

```

Nun sollte auf jeden Fall eine Session verfügbar sein. Innerhalb dieser Sitzung werden nun in Zeile 17 und 20 benötigte Session-Variablen, falls sie verfügbar sind, ausgelesen und lokal zugänglich gemacht.

Ab Zeile 28 behandelt die Datei `index.session.php` verschiedene Sicherheits-Probleme, welche unter Umständen auftreten können. Dazu gehört, dass Personen, welche unberechtigtweise auf dem Portal des "Virtuellen Unix Labors" arbeiten, abgewiesen werden. Deshalb stellt PHP in Zeile 28 den Pfad auf das `logout`-Script zur Verfügung. Anschließend werden alle Sicherheits-Überprüfungen vorgenommen.

In der Regel kann ein Benutzer den Aufbau der Seite nur über die URL manipulieren, da die mit der URL übergebenen Parameter den Inhalt bestimmen. Denn sie entscheiden, ob es sich dabei um den Seiten-Inhalt eines Administrators oder dem eines einfachen Benutzers handelt. Zeile 48 des Listings 3.4 – `index.session.php` prüft dazu zuerst, ob die gewünschten Parameter auch wirklich verfügbar sind.

Ist eine Person nicht authentifiziert (`$auth = FALSE`), darf sie sich nur im Bereich `default` aufhalten. Zeile 40 kontrolliert das. Zum Schluß fragt das Script noch bei der Datenbank an und kontrolliert die Benutzer-Berechtigung zum Zugriff auf das jeweilige Menü (Zeile 48 bis 65).

Nachdem alle Überprüfungen die Sicherheit des Benutzers bestätigt haben, wird ihm der Zugriff auf das Menü und den damit enthaltenen Inhalt gestattet.

Sollte ein Anwender, durch Beeinflussung der URL im Browser, versuchen ein nicht autorisiertes Menü zu laden, so ruft eine der Zeilen 35, 42 oder 65 im Listing 3.4 das PHP-Script `logout.php` auf, welches im nachfolgenden Listing 3.5 zu sehen ist.

Dieses kurze Script kümmert sich darum, dass die bestehende Sitzung "zerstört" und der Benutzer ausgeloggt wird.

Dazu holt sich PHP in Zeile 4 die Session-ID um die Sitzung zuerst ordnungsgemäß zu übernehmen (Zeile 9). Danach werden in Zeile 14 alle Session-Variablen gelöscht, wobei eine Zeile zuvor, zur Sicherheit, der komplette Session-Hash überschrieben wurde. Anschließend löscht PHP in Zeile 15 die bestehende Sitzung und lädt schließlich wieder das Login-Formular.

Listing 3.5: `logout.php`

```
1 <?php
2     require_once 'functions.php';
3
4     $PHPSESSID = getValue("GET", "PHPSESSID");
5     ...
6
7     if (!empty($PHPSESSID))
8     {
9         session_start();
10        ...
11
12        $HTTP_SESSION_VARS = array();
13        session_unset();
14        session_destroy();
15    }
16
17    $string = getLocation("index.php");
18    header ("Location: " . $string . "?menu=default&content=login");
19
20 ?>
```

3.4 Benutzerverwaltung

Wie bei den meisten Online-Anbietern, benötigt man auch beim "Virtuellen Unix Labor" eine Zugangs-Berechtigung. Die Formulare die dazu nötig sind, werden von PHP-Scripten generiert. Diese sind als Module in das bereits erklärte Grundgerüst mit Session-Handling eingebettet und greifen direkt auf die darunter liegende Datenbank zu.

Eine Reihe weiterer Scripte dienen zur Auswertung der vom Anwender eingetragener Daten. Dazu gilt es eine Vielzahl von möglichen Fehler abzufangen und den Benutzer zu benachrichtigen, um später alle Informationen mit der Datenbank abzugleichen.

3.4.1 Benutzerprofil anlegen

Damit ein Anwender ein "Konto" beim "Virtuellen Unix Labor" bekommt, wird von ihm verlangt, zuerst seine persönlichen Daten in einem dafür vorgesehenen Formular zu hinterlassen. Wir sprechen hier vom "Anlegen eines neuen Profils". Abbildung 3.7 zeigt, wie sich das Formular dem Benutzer präsentiert.

Um den Anwender nicht zu verwirren, baut sich für ihn scheinbar immer wieder die selbe Seite auf. Allerdings werden bei Bedarf Fehlermeldungen in roter Farbe über dem Formular ausgegeben. Dabei läuft vieles im Hintergrund ab und bleibt dem Benutzer verborgen. Das dazu aufgerufene Modul "FirstLogin" befindet sich im Verzeichnis `default/firstLogin/` und enthält die Scripten

- `index.php`
- `validate.php`.

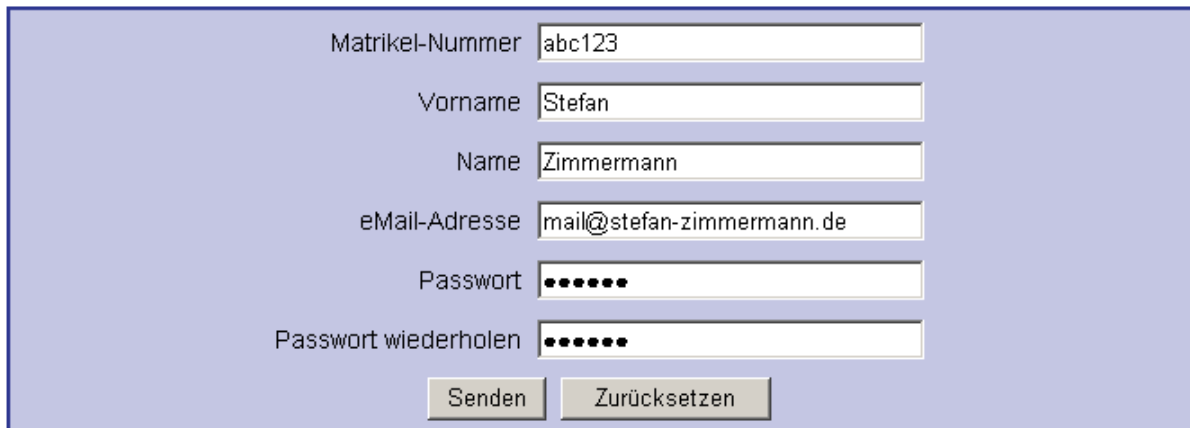
Nach Bestätigung des Buttons "Senden", übermittelt der Client alle Daten des Formulars an den Validierer. Dabei handelt es sich um das PHP-Script `default/firstLogin/validate.php`, welches alle im Formular enthaltenen Eingaben überprüft.

Entsprechen die übertragenen Daten nicht der gewünschten Form, so werden die Benutzereingaben und die entsprechenden Meldungen in der Session gespeichert. Die danach geladenen HTML-Seite unterscheidet sich nur in der Ausgabe der Fehlermeldung von der bereits bekannten Eingabemaske. Dazu

Profil anlegen

Bitte geben Sie hier Ihre Daten ein.
Es sind alle Felder auszufüllen.

Die Matrikel-Nummer darf keine Buchstaben oder andere Zeichen enthalten.



Matrikel-Nummer	<input type="text" value="abc123"/>
Vorname	<input type="text" value="Stefan"/>
Name	<input type="text" value="Zimmermann"/>
eMail-Adresse	<input type="text" value="mail@stefan-zimmermann.de"/>
Passwort	<input type="password" value="....."/>
Passwort wiederholen	<input type="password" value="....."/>
<input type="button" value="Senden"/> <input type="button" value="Zurücksetzen"/>	

Nachdem Sie Ihr Profil erfolgreich angelegt wurde, können Sie bereits in den geschützten Bereich eintreten. Desweiteren wird Ihnen eine Bestätigung per eMail zugesand. Mit ihr erhalten Sie eine Geheimnummer, die Sie einmalig zur Identifikation Ihres Profils verwenden. Ohne diese Identifizierung wird Ihr Zugang nach 8 Tagen ungültig.

Abbildung 3.7: Formular zum Erstellen eines neuen Profils

greift das Script `default/firstLogin/index.php` auf die bestehende Sitzung zu und liest alle vom Validierer geschriebenen Informationen.

Geprüft wird unter anderem, ob das gesetzte Login nur einmal existiert:

```
SELECT * FROM benutzer WHERE login='mail@stefan-zimmermann.de'
```

Und auch die Matrikel-Nummer darf noch nicht vorhanden sein:

```
SELECT * FROM benutzer WHERE matrikel_nr='123'
```

Das Ergebnis der SQL-Abfragen darf in beiden Fällen keinen Datensatz zurück liefern. Akzeptiert das Script `default/firstLogin/validate.php` alle Eingaben des Anwenders, verschlüsselt PHP das gewünschte Passwort und überträgt schließlich mit dem SQL-Statement

```

INSERT INTO benutzer (
    matrikel_nr,
    vorname,
    nachname,
    email,
    login,
    password,
    anmeldedatum,
    freischalt_secret)
VALUES (
    '12345',
    'Stefan',
    'Zimmermann',
    'mail@stefan-zimmermann.de',
    'mail@stefan-zimmermann.de',
    'Z28TIt7BYZ9Ws',
    'now',
    '76vhcwIuq0fmM'
)

```

alle Informationen in die Datenbank. Abbildung 3.8 veranschaulicht den Vorgang.

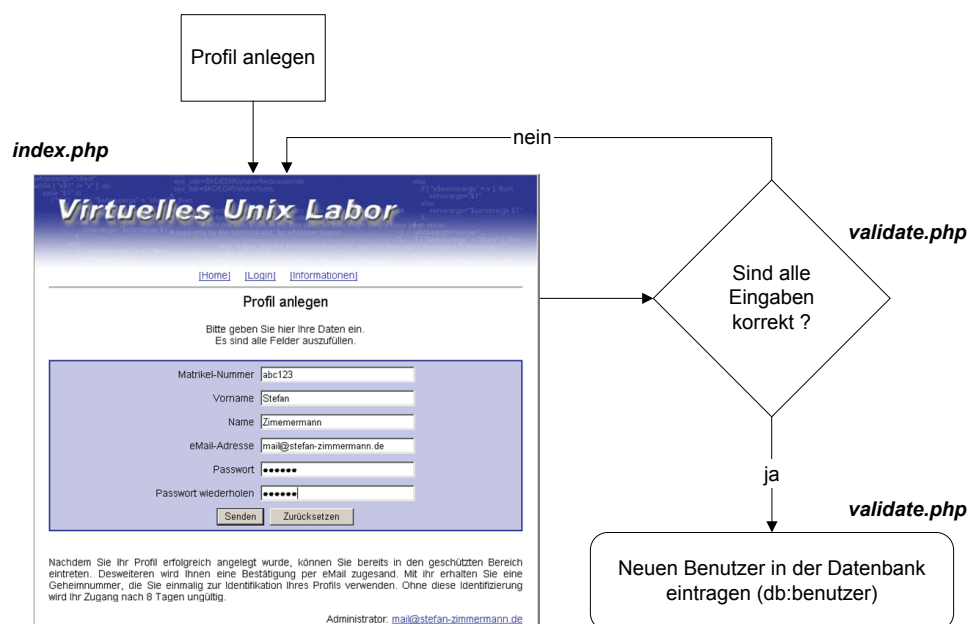


Abbildung 3.8: Ablaufdiagramm zum Erstellen eines neuen Profils

3.4.2 Login-Vorgang

Ein bereits registrierter Benutzer erreicht über den Menüpunkt "Login" das Login-Formular. Diese Eingabemaske wird vom gleichnamigen Modul "Login" erzeugt, welches sich im Verzeichnis `default/login/` befindet und die PHP-Scripten

- `index.php`
- `validateLogin.php`

enthält.

Die Datei `default/login/index.php` erzeugt die in Abbildung 3.9 dargestellte Maske, welche der Benutzer auszufüllen hat. Die im Anschluß an den Validierer `default/login/validateLogin.php` gesendeten Daten werden mit den Einträgen der Datenbank verglichen.

Login

Bitte geben Sie hier Ihre Daten ein.

Login

Passwort

Sie sind noch nicht registriert !?
Dann muß für Sie zuerst ein [ein Profil](#) angelegt werden.

Abbildung 3.9: Formular zum Einloggen eines Benutzers

Dabei fragt das SQL-Statement

```
SELECT password, typ FROM benutzer where login='stefan'
```

nach dem verschlüsselten Passwort des Benutzers "stefan". Ist kein Datensatz vorhanden, wird eine Meldung generiert. Konnte ein cryptische Passwort aus der Datenbank gelesen werden, verschlüsselt PHP das vom Benutzer eingegebene Passwort und vergleicht es mit dem Crypt aus der Datenbank.

Im Fehlerfall transportiert das Script, welches die Überprüfung durchführt, alle Informationen über die Session zurück an das `index.php`-Script. Der Anwender

darf dann seine Eingaben korriegieren.

Stimmen Login und Passwort überein, wird der Benutzertyp und der Login in der aktuellen Sitzung verewigt. Somit ist keine weitere Authentifizierung mehr notwendig, da PHP die notwendigen Informationen jederzeit aus der Session lesen kann. Abbildung 3.10 zeigt den Vorgang anhand eines Diagramms.

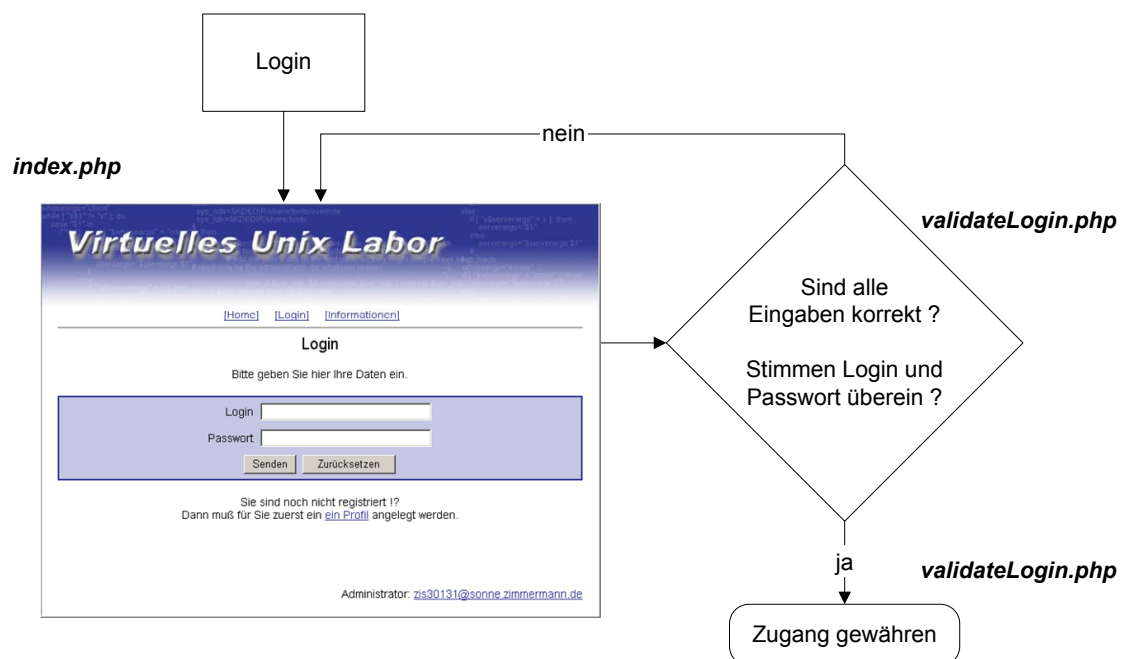


Abbildung 3.10: Ablaufdiagramm zum Einlogg-Vorgang

3.4.3 Benutzerdaten im Anwenderbereich

Beim Anlegen eines neuen Profils, wurde die eMail-Adresse des Benutzers als Login-Name eingetragen. Über den Menüpunkt "Benutzerdaten" kann der Login und auch alle anderen persönlichen Daten verändert werden. Das dazu verwendete Modul beinhaltet die Scripte

- index.php
- validateUserData.php
- userData2.php
- updated.php.

Damit die bereits gesetzten Informationen für den Anwender sichtbar werden, liest das SQL-Statement

```
SELECT * FROM benutzer WHERE login='user'
```

seine Daten aus der Datenbank. Das PHP-Script `user/userdata/index.php` setzt anschließend die erhaltenen Informationen in das dafür vorgesehene Formular ein. Abbildung 3.11 zeigt die erhaltene Maske.

Benutzerdaten

In diesem Bereich können Sie die bei uns gespeicherten persönlichen Daten ändern.

Vorname	<input type="text" value="test"/>
Name	<input type="text" value="user"/>
eMail-Adresse	<input type="text" value="test@user.de"/>
Login	<input type="text" value="user"/>
Passwortfeld nur ausfüllen, wenn neues Passwort gesetzt werden soll !	
neues Passwort	<input type="password"/>
Passwort wiederholen	<input type="password"/>
<input type="button" value="Werte übernehmen"/> <input type="button" value="Zurücksetzen"/>	

Abbildung 3.11: Formular zum Verändern der Benutzerdaten

Nachdem die geänderten Daten an den Server übertragen wurden, startet das Script `user/userdata/validateUserData.php`. Es überprüft alle Eingaben auf korrekte Form und kontrolliert mit dem SQL-Statement

```
SELECT * FROM benutzer WHERE login='user' AND matrikel_nr!='56226'
```

ob das Login bereits vorhanden ist.

Sind dabei Probleme aufgetreten, überträgt PHP die Meldungen mit Hilfe der bestehenden Session, an das Script `user/userdata/userData2.php`. Dieses Script baut wiederum dem Anwender eine Eingabemaske, welche der in Abbildung 3.11 gleicht.

Akzeptiert PHP alle Änderungen, startet ein Update auf die Datenbank.

```
UPDATE benutzer SET vorname='test',
    nachname='user',
    email='test@user.de',
    login='user'
WHERE matrikel_nr='56226'
```

Danach zeigt letztendlich das Script `user/userdata/updated.php` die in Abbildung 3.12 dargestellte Bestätigung.

Ihre Daten wurden erfolgreich geändert.

Abbildung 3.12: Bestätigung der veränderten Benutzerdaten

Zusammengefasst kann man den kompletten Ablauf in Abbildung 3.13 betrachten.

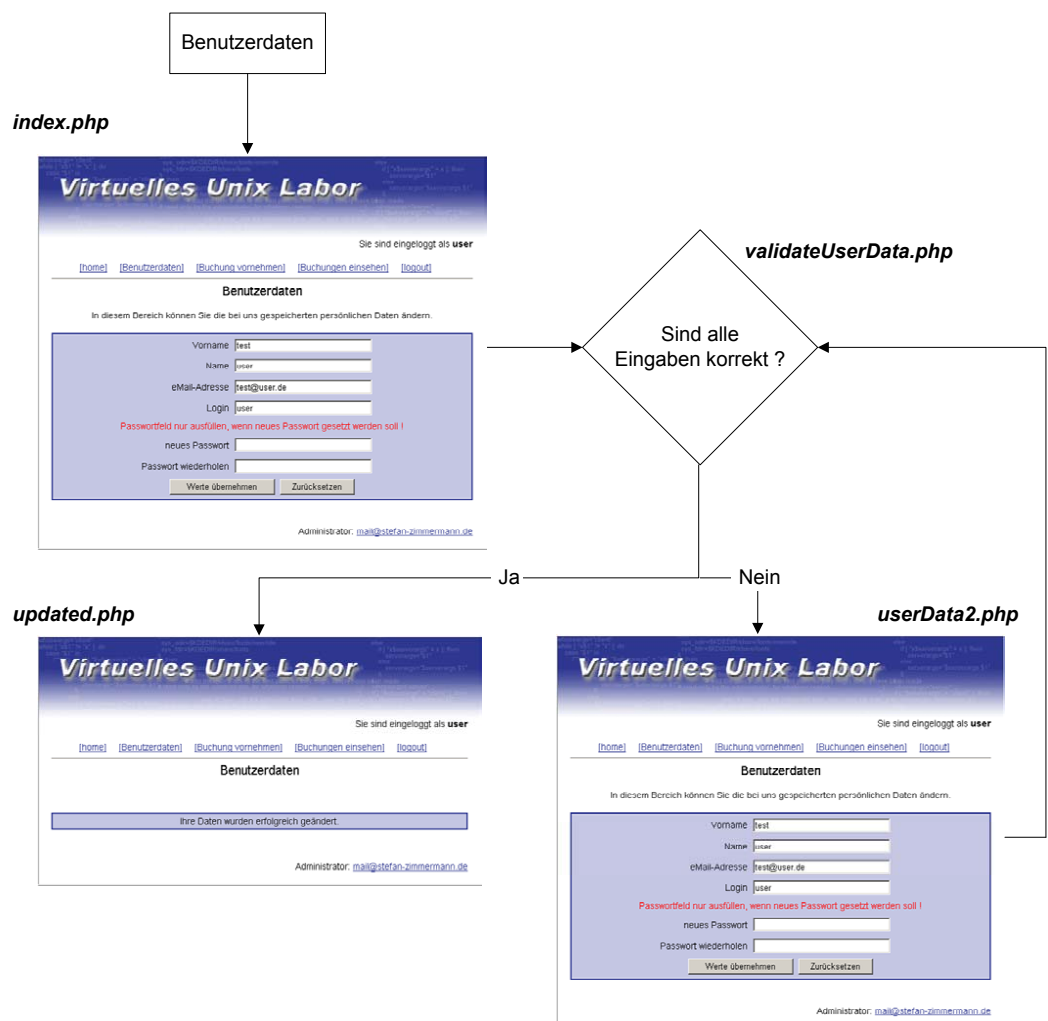


Abbildung 3.13: Ablaufdiagramm zum Verändern der Benutzerdaten

3.4.4 Benutzerdaten im Administrationsbereich

Als Administrator hat man die Möglichkeit, alle Daten der im "Virtuellen Unix Labor" existierenden Benutzer, einzusehen und zu verändern. Wählt der Administrator deshalb den Menüpunkt "Benutzerdaten", wird das zugehörige Modul mit den Scripten

- `index.php`
- `adminData2.php`
- `adminData3.php`
- `validateData.php`
- `updated.php`
- `delete.php`
- `delete2.php`
- `deleteMessage.php`
- `deleted.php`

gestartet. Wie in Abbildung 3.14, gibt PHP eine Liste aller Benutzer wieder. Die Informationen erhält das Script `admin/userdata/index.php` mit Hilfe der SQL-Abfrage

```
SELECT * FROM benutzer ORDER BY lower(nachname), lower(vorname)
```

Weiterhin bietet die entstandene Web-Seite die Möglichkeit nach Stichwörtern zu suchen und die Datensätze zu sortieren.

Mit dem Klick auf die Schaltfläche "Profil editieren", wie es Abbildung 3.15 zeigt, bekommt der Administrator auf den jeweiligen Datensatz Zugriff. Dazu wird die Matrikel-Nummer via POST an den Server übertragen.

Hat sich der Administrator zum Editieren eines Profils entschieden, öffnet Script `admin/userdata/userData2.php` den Zugang auf die Datenbank und liest mit

```
SELECT * FROM benutzer WHERE matrikel_nr='123456'
```

Benutzerdaten einsehen / ändern

Stichwort-Suche:

Vorhandene User: 5
| 1-5 |











MatrikelNr	Name	Vorname	Login	Type		
2	dozent	test	dozent	dozent		
17	Feyrer	Hubert	hubertf	admin		
123456	Feyrer	Hubert	hubertf@gmx.de	user		
1	user	test	user	user		
12345	Zimmermann	Stefan	zis30131	admin		

Abbildung 3.14: Verwaltung der Benutzer





die Informationen des jeweiligen Benutzers. Die sich dabei aufbauende HTML-Seite veranschaulicht Abbildung 3.16. An dieser Stelle können nun Daten im Formular geändert werden. Bevor sich allerdings die Datenbank aktualisiert, prüft das PHP-Script `admin/userdata/validate.php` die Korrektheit der Daten.

Mit Hilfe der SQL-Statements

```
SELECT * FROM benutzer WHERE matrikel_nr='123456'
      AND matrikel_nr!='123456'
```

und

```
SELECT * FROM benutzer WHERE login='hubertf@gmx.de'
      AND matrikel_nr!='123456'
```

123456	Feyrer	Hubert	hubertf@gmx.de	user		
1	user	test	user	user		

Profil editieren

Abbildung 3.15: Klick auf Schaltfläche 'Profil editieren'

wird kontrollieren, ob die gewünschte Matrikel-Nummer bzw. der gewünschte Login-Name bereits existiert. Ist dies der Fall, überträgt die bestehende Session

Meldungen an das Script `admin/userdata/userData3.php`. Dem Administrator zeigt sich danach eine Maske wie in Abbildung 3.16 und er kann seine Eingaben korrigieren.

Benutzerdaten

In diesem Bereich können Sie die gespeicherten persönlichen Daten ändern.

MatrikelNr	<input type="text" value="123456"/>
Vorname	<input type="text" value="Hubert"/>
Name	<input type="text" value="Feyrer"/>
eMail-Adresse	<input type="text" value="hubertf@gmx.de"/>
Login	<input type="text" value="hubertf@gmx.de"/>
Benutzer-Typ ?	Admin <input type="radio"/> Dozent <input type="radio"/> User <input checked="" type="radio"/>
Anmeldung bestätigt ?	Ja <input checked="" type="radio"/> Nein <input type="radio"/> <input type="text" value="bestaetigt"/>
Passwortfeld nur ausfüllen, wenn neues Passwort gesetzt werden soll !	
neues Passwort	<input type="text"/>
Passwort wiederholen	<input type="text"/>
<input type="button" value="Werte übernehmen"/> <input type="button" value="Zurücksetzen"/>	

Abbildung 3.16: Profil editieren

Nachdem alle Überprüfungen erfolgreich abgelaufen sind, aktualisiert das Script `admin/userdata/validate.php` durch die nachfolgende SQL-Anweisung die Datenbank.

```
UPDATE benutzer SET matrikel_nr='123456',  
    vorname='Hubert',  
    nachname='Feyrer',  
    email='hubertfeyrer.de',  
    login='hubertf',  
    typ='admin',  
    freischalt_secret='bestaetigt'  
WHERE matrikel_nr='123456'
```

Die Liste aller Benutzer bietet weiterhin dem Administrator die Möglichkeit einen Anwender des "Virtuellen Unix Labors" zu entfernen. Dazu klickt er auf die Schaltfläche "Profil löschen", wie es Abbildung 3.17 zeigt.



Abbildung 3.17: Klick auf Schaltfläche 'Profil löschen'

Das dann aufgerufenen Script `admin/userdata/delete.php` enthält einen Sicherheitsmechanismus. Es prüft, ob die Matrikel-Nummer des zu löschenden Benutzers mit der eigenen übereinstimmt. Ist dies der Fall, darf der Datensatz in keinsten Weise verworfen werden. Das würde zu einem "logout" führen und der Administrator kann sich nie wieder anmelden. Deshalb leitet PHP zum Script `admin/userdata/deleteMessage.php` weiter, welches die in Abbildung 3.18 dargestellte Meldung ausgibt.

Achtung: Bei dem zu löschenden Profil handelt es sich um Ihr Eigenes !!
Dieser Datensatz kann nicht gelöscht werden.

Abbildung 3.18: Das eigene Profil kann nicht gelöscht werden

Tritt der Sicherheitsmechanismus nicht in Kraft, holt das Script `admin/userdata/delete.php` mit Hilfe des SQL-Statements

```
SELECT * FROM benutzer WHERE matrikel_nr='123456'
```

den zu löschende Datensatz. Weiterhin vergewissert sich PHP, ob die Daten wirklich verworfen werden sollen. In Abbildung 3.19 kann man die HTML-Seite sehen, welche sich dem Administrator präsentiert.

Datensatz löschen

Der folgende Datensatz wird unwiederruflich gelöscht !!

MatrikelNr	Name	Vorname	login
123456	Feyrer	Hubert	hubertf@gmx.de

Wollen Sie fortfahren ?

Abbildung 3.19: Soll das Profil gelöscht werden ?

Wird die Nachfrage bestätigt, startet der Server das Script `admin/userdata/delete2.php`. Mit der SQL-Anweisung

```
DELETE FROM benutzer WHERE matrikel_nr='123456'
```

wird schließlich der Benutzer mit all seinen Daten aus der Datenbank gelöscht. Den gesamten Ablauf kann man nochmals in [Abbildung 3.20](#) betrachten.

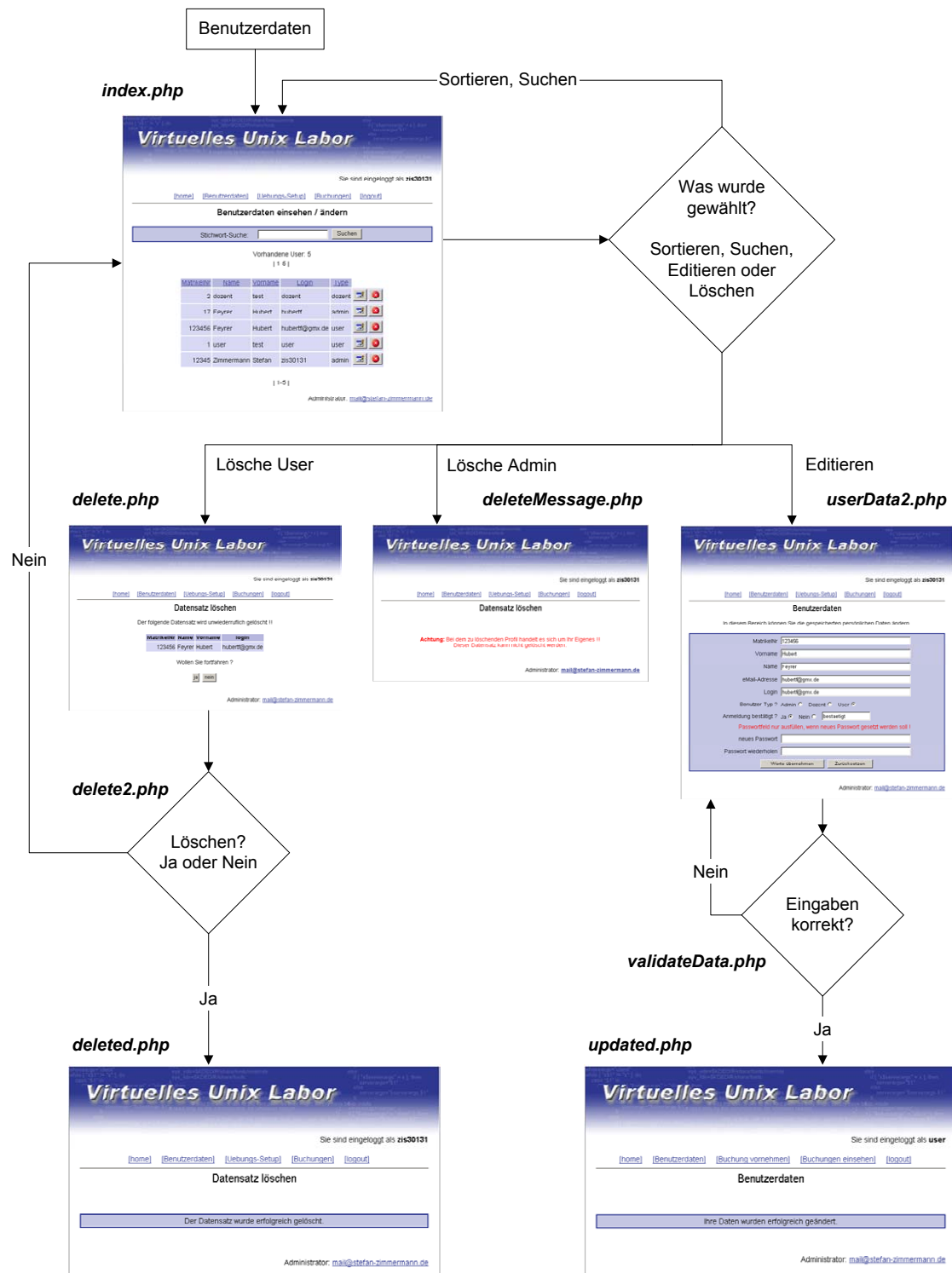


Abbildung 3.20: Ablaufdiagramm zum Verändern der Benutzerdaten als Adminsitrator

3.5 Buchungssystem

Das Projekt "virtuelles Unix Labor" wurde ins Leben gerufen, um jedem Anwender Arbeitszeiten auf den Übungsrechnern zur Verfügung zu stellen. Der Zugangsrechner (siehe Abbildung 1.1 auf Seite 2) verwaltet diese Zeiten und sorgt dafür, dass sich die Benutzer auf den Übungsrechnern frei und ungestört bewegen können.

Die Verwaltung variabler Rechnerzeiten ist ein sehr komplexes Thema, deshalb wurde ein System mit festen Zeitfenstern gewählt. Egal ob der Administrator oder Dozent 60 oder gar 120 Minuten für den Ablauf der Übung wählt, es steht immer ein Zeitfenster von drei Stunden zur Verfügung.

Für den Anwender bleibt nun die Aufgabe, sich für einen Zeitraum zu entscheiden zu dem er eine bestimmte Übung durchführen möchte.

3.5.1 Buchen von Übungen im Anwenderbereich

Eingeloggt im "virtuelles Unix Labor", wählt der User den Menüpunkt "Buchung vornehmen". Daraufhin startet das Modul "Buchen", dessen Scripten sich im Projektverzeichnis `./user/buchen/` befinden. Dieses Modul umfasst die Dateien:

- `index.php`
- `calender.php`
- `buchen2.php`
- `buchen3.php`
- `validate.php`
- `zeigeBuchung.php`
- `speichereBuchung.php`
- `execute.php`

Gestartet wird das Script `index.php`, welches das Script `calender.php` einbettet und einen Kalender (siehe Abbildung 3.21) generiert. Er bietet die Möglichkeit einen Tag auszuwählen. Der Mausklick auf den gewünschten Tag erzeugt

eine Datenbankanfrage, welche nach den Buchungen dieses Tages fragt.

Hinweis: Als Beispiel wurde die Übung "NIS" für den 26. Mai 2003 um 9:00 Uhr gewählt. Wir gehen davon aus, dass heute der 22. Mai 2003 ist und spielen das Szenario anhand dieser Daten durch.

```
SELECT startzeit FROM buchungen WHERE datum='5/26/2003'
```

```
startzeit
-----
12:00:00
18:00:00
(2 rows)
```

PHP vergleicht nun die bereits gebuchten Startzeiten dieses Tages mit den fest definierten Zeitfenstern und entfernt diese bei Übereinstimmung aus dem statischen Array. Der nachfolgende Auszug aus der Datei `user/buchen/index.php` – Listing 3.6 – zeigt das Array mit den eingetragenen Startzeiten.

Listing 3.6: Auszug aus `user/buchen/index.php`

```
1 <?php
2     ...
3
4     // feste Zeitfenster
5     $startzeit = array();
6     $startzeit[0] = "00:00:00";
7     $startzeit[1] = "03:00:00";
8     $startzeit[2] = "06:00:00";
9     $startzeit[3] = "09:00:00";
10    $startzeit[4] = "12:00:00";
11    $startzeit[5] = "15:00:00";
12    $startzeit[6] = "18:00:00";
13    $startzeit[7] = "21:00:00";
14
15    ...
16 ?>
```

Nachdem das Array `$startzeit` aktualisiert wurde, baut sich beim Anwender die in Abbildung 3.21 dargestellte Auswahl auf.

Da die Startzeiten 12:00 Uhr und 18:00 Uhr fehlen, entschließt sich der Benutzer zum Beginn der Übung um 9:00 Uhr und klickt auf die Schaltfläche "weiter".

Abbildung 3.21: Auswahl der verfügbaren Startzeiten des 26. Mai 2003

Startzeit und Datum werden nun an den Server übermittelt und mit Hilfe des Scripts `buchen2.php` ausgewertet. Dazu erkennt PHP zuerst, ob überhaupt eine Zeit gewählt wurde. Danach entscheidet sich, ob die gewählte Uhrzeit auch wirklich in der Zukunft liegt.

Das SQL-Statment

```
SELECT age( now(), '5/26/2003 9:00'::timestamp - max(vorlauf) )
        AS alter FROM uebungen
```

liefert das Ergebnis:

```
        alter
-----
-3 days -14:26:33.19855
(1 row)
```

PostgreSQL ermittelt dadurch, welches Alter die ausgesuchte Zeit im Bezug auf jetzt (`now()`) hat. Dabei wird vom Zeitstempel (`timestamp`) die maximale Vorlaufzeit aller Übungen subtrahiert, da der Zugangsrechner diese Zeit zusätzlich benötigt, um die Übung vorzubereiten. Das sich in diesem Fall ergebene Alter von -3 Tagen bedeutet, dass der Zeitpunkt 3 Tage in der Zukunft liegt.

Somit ist die Eingabe korrekt und es wird das Script `buchen3.php` aufgerufen. Dieses Script hat die Aufgabe, dem Anwender eine Auswahl an Übungen zu

zeigen, von denen er schließlich eine auswählen darf.

Die Liste der möglichen Übungen findet sich, wie bei allen anderen Daten, in der Datenbank. Entsprechend dem jeweiligen User wird bei der Datenbank angefragt.

```
SELECT * FROM uebungen WHERE uebung_id NOT IN (
    SELECT uebungen.uebung_id
        FROM uebungen, buchungen, benutzer
        WHERE benutzer.user_id=buchungen.user_id
        AND uebungen.uebung_id=buchungen.uebung_id
        AND uebungen.wiederholbar='f'
        AND benutzer.login='user'
    )
```

uebung_id	bezeichnung	vorlauf	dauer	nachlauf	wiederholbar
nfs	Aufsetzen von NFS Client und Server	00:45:00	01:30:00	00:15:00	t
nis	Aufsetzen von NIS Client und Server	00:45:00	01:30:00	00:15:00	t
apache	Aufsetzen eines Apache Servers	00:30:00	01:00:00	00:30:00	t
pruefung	Verwalten von Benutzern mit Hilfe von NIS	00:45:00	01:00:00	00:15:00	f
pruefung2	Einrichten eines Apache Servers mit SSL	00:45:00	01:00:00	00:15:00	f

(5 row)

	<u>Kurzbezeichnung</u>	<u>Bezeichnung</u>	<u>Dauer</u>	<u>wiederholbar</u>
<input type="radio"/>	apache	Aufsetzen eines Apache Servers	01:00	ja
<input type="radio"/>	nfs	Aufsetzen von NFS Client und Server	01:30	ja
<input checked="" type="radio"/>	nis	Aufsetzen von NIS Client und Server	01:30	ja
<input type="radio"/>	pruefung	Verwalten von Benutzern mit Hilfe von NIS	01:00	nein
<input type="radio"/>	pruefung2	Einrichten eines Apache Servers mit SSL	01:00	nein

Abbildung 3.22: Auswahl der verfügbaren Übungen

Der SELECT nach dem NOT IN sucht alle nicht wiederholbaren Übungen zu dem entsprechenden Benutzer. Letztendlich erscheint keine nicht wiederholbare Übung (wiederholbar='f'), wenn sie bereits gebucht wurde. Abbildung 3.22 zeigt die grafisch aufbereitete Auswahl.

Nachdem in unserem Fall die Übung "NIS" gewählt wurde, prüft das Script validate.php die Eingabe.

Die SELECT-Anweisung

```
SELECT * FROM uebungen WHERE uebung_id='nis'
```

zeigt nochmals die ausgesuchten Werte.

uebung_id	bezeichnung	vorlauf	dauer	nachlauf	wiederholbar
nis	Aufsetzen von NIS Client und Server	00:45:00	01:30:00	00:15:00	t

(1 row)

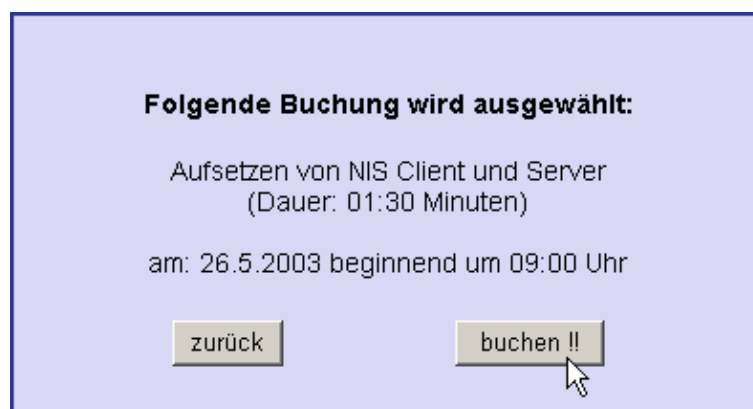


Abbildung 3.23: Ausgabe aller gewählten Daten

Der Anwender muss nun seine Daten bestätigen. Er klickt auf buchen (siehe Abbildung 3.23), um den gesamten Vorgang abzuschließen.

Für den User ist die Buchung somit abgeschlossen. Der Server jedoch hat jetzt erst seine Arbeit zu verrichten. Bevor er alle Benutzereingaben in die Datenbank schreibt, findet nochmals eine Überprüfung statt. Sie stellt sicher, dass für den User zur gewählten Zeit (26. Mai 2003, 9:00 Uhr) noch keine Buchung in der Datenbank steht.

```
SELECT buchungs_id FROM buchungen, benutzer
WHERE datum='5/26/2003'
AND startzeit='9:00'
AND buchungen.user_id=benutzer.user_id
AND benutzer.login='user'
```

```
buchungs_id
-----
(0 row)
```

Dies geschieht mit dem Hintergrund, dass die Übung nicht ein zweites Mal gebucht wird, falls der Benutzer "reload" im Browser wählt und der Server die Daten erneut erhält.

In unserem Fall sendet der Benutzer die Daten zum ersten Mal. Man erkennt es daran, dass es keine `buchungs_id` zur ausgesuchten Zeit gibt. In die Datenbank darf nun ein neuer Eintrag erfolgen.

```
INSERT INTO buchungen (  
  user_id,  
  uebung_id,  
  datum,  
  startzeit  
)  
VALUES (  
  (SELECT user_id FROM benutzer WHERE login='user'),  
  'nis',  
  '5/26/2003',  
  '9:00'  
)
```

Beim "Virtuellen Unix Labor" handelt es sich um ein automatisch konfigurierendes Übungssystem, dass nur von Benutzereingaben gesteuert wird. Die gerade getätigte Buchung bezieht sich auf eine Übung, welche erst in vier Tagen geplant ist. Sie wird deshalb nicht sofort ausgeführt, sondern an einen `at-Job`¹ übergeben, der sich um die termingerechte Vorbereitung der Übungsrechner kümmert.

Das Script `deploy`, welches sich im Projektverzeichnis `./bin/` befindet, ist zuständig für das korrekte Aufsetzen der Images auf den verwendeten Übungsrechnern. Um welche Rechner es sich handelt und welche Images benötigt werden, erfährt das Script über die jeweilige Buchungsnummer (`buchungen.buchungs_id`).

¹ `at` ist ein Unix-Programm, welches Programme oder Scripten zu einem festgelegten, späteren Zeitpunkt ausführt.

```
SELECT buchungs_id FROM buchungen, benutzer
WHERE datum='5/26/2003'
AND startzeit='19:00'
AND buchungen.user_id=benutzer.user_id
AND benutzer.login='user'
```

```
buchungs_id
-----
          17
(1 row)
```

Dem at-Job wird so das Script `deploy` mit der Buchungsnummer als Parameter übergeben. Dieser benötigt allerdings nicht die Startzeit, für die sich der Anwender entschieden hat. Da das System die Übung erst vorbereitet muss, kommt die Vorlaufzeit der Übung zum Einsatz.

Die Datenbank erledigt die Rechenarbeit und subtrahiert die Vorlaufzeit der Übung "NIS" von der vom Benutzer gewählten Startzeit.

```
SELECT ('5/26/2003 9:00'::timestamp - vorlauf)
AS startzeit FROM uebungen WHERE uebung_id='nis'
```

```
startzeit
-----
2003-05-26 08:15:00
(1 row)
```

Weiterhin veranlasst ein SQL-Statement die Zerlegung der Startzeit in einfach zu handhabende Werte.

```
SELECT
    date_part('hour', '2003-05-26 8:15:00'::timestamp) AS hour,
    date_part('minute', '2003-05-26 8:15:00'::timestamp) AS minute,
    date_part('day', '2003-05-26 8:15:00'::timestamp) AS day,
    date_part('month', '2003-05-26 8:15:00'::timestamp) AS month,
    date_part('year', '2003-05-26 8:15:00'::timestamp) AS year
```

```
hour | minute | day | month | year
-----+-----+-----+-----+-----
    17 |      15 |   20 |     5 | 2003
(1 row)
```

Mit Hilfe dieser Angaben startet PHP einen Systemaufruf, welche den `at`-Job startet und das Script `deploy` in eine Warteschlange einträgt.

```
echo "deploy 17" | at 8:15 5/26/2003 2>&1
```

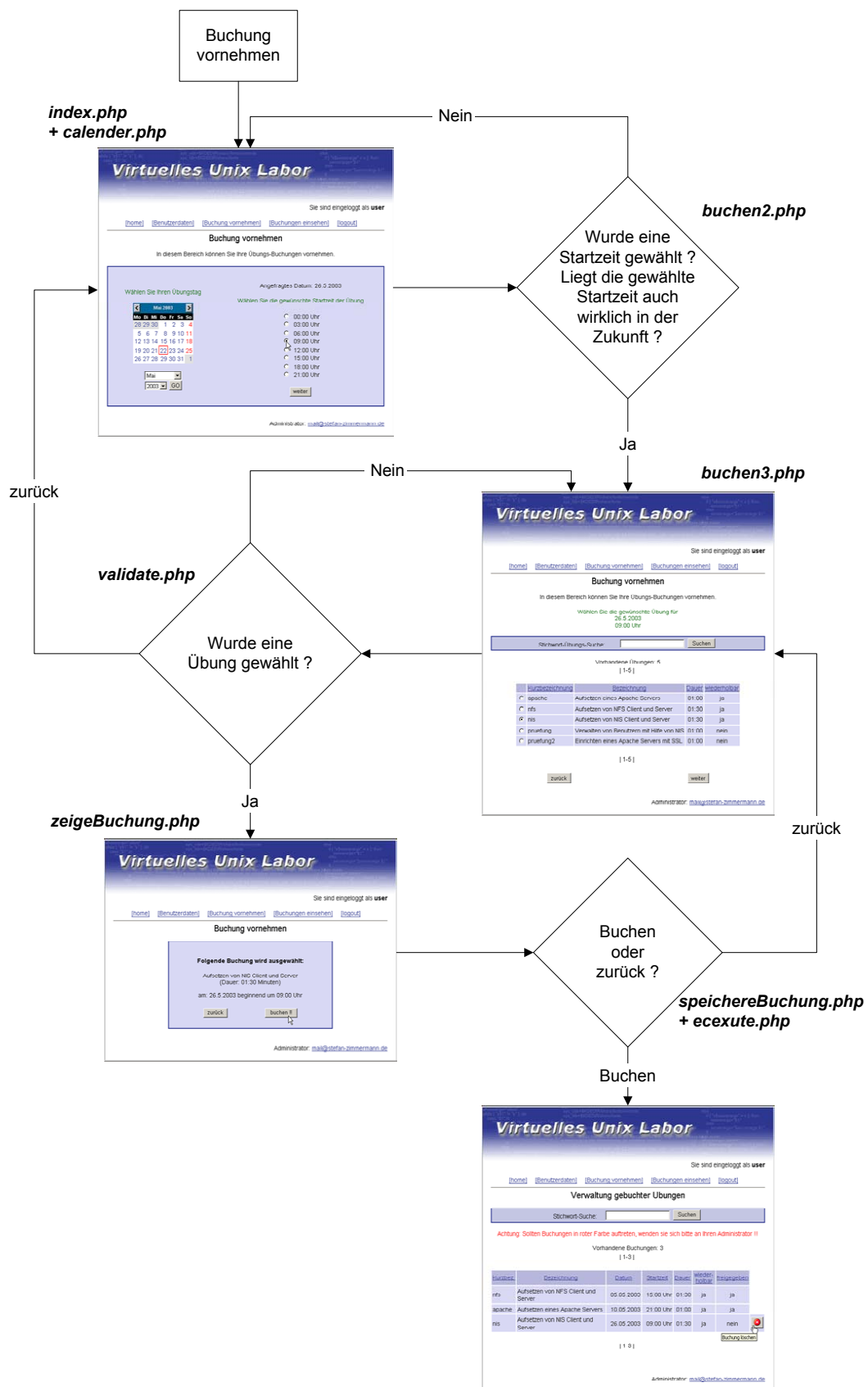
Eine Ausgabe von `at` bekommt man nur durch Umleiten des `stderr` nach `stdout`. Dies erreicht man durch Anhängen des Konstrukts `2>&1`. Somit erhält man folgende Ausgabe.

```
Job 50 will be executed using /bin/sh
```

Die Meldung wird anschließend von PHP gelesen und die Job-Nummer extrahiert. Diese Nummer ist der einzige Ankerpunkt des Auftrags `deploy`. Jeglicher spätere Zugriff ist allein über diese Job-Nummer möglich. Deshalb wird sie zur Buchung in der Datenbank hinterlegt.

```
UPDATE buchungen SET at_id='50'
  WHERE datum='5/26/2003'
  AND startzeit='9:00'
  AND buchungen.user_id=benutzer.user_id
  AND benutzer.login='user'
```

Die Buchung ist damit abgeschlossen. Zusammengefasst zeigt [Abbildung 3.24](#) den gesamten Ablauf des Buchungsvorgangs.



3.5.2 Verwalten der Buchungen im Anwenderbereich

Jeder Benutzer kann seine getätigten Buchungen über den Menüpunkt "Buchungen einsehen" im Anwenderbereich begutachten. Dafür ist das Modul "Buchungen" zuständig und hat seine Scripten im Verzeichnis `./user/buchungen/`. Folgende Scripten setzen sich zu diesem Modul zusammen:

- `index.php`
- `delete.php`
- `delete2.php`
- `deleted.php`

Die Einsprungsdatei `index.php` stellt die Buchungen des jeweiligen Benutzers in Form einer Tabelle dar. Sollte die Liste der Buchungen zu lang sein, bietet ein Eingabefeld die Möglichkeit nach Stichwörtern zu suchen. Die dadurch generierte SQL-Abfrage ist allerdings etwas länglich. Abbildung 3.25 zeigt die grafische Aufbereitung der Daten.


Verwaltung gebuchter Übungen

Stichwort-Suche:

Achtung: Sollten Buchungen in roter Farbe auftreten, wenden sie sich bitte an Ihren Administrator !!

Vorhandene Buchungen: 2
| 1-2 |

<u>Kurzbez.</u>	<u>Bezeichnung</u>	<u>Datum</u>	<u>Startzeit</u>	<u>Dauer</u>	<u>wiederholbar</u>	<u>freigegeben</u>
nis	Aufsetzen von NIS Client und Server	12.04.2003	12:00 Uhr	01:30	ja	ja
nis	Aufsetzen von NIS Client und Server	26.05.2003	09:00 Uhr	01:30	ja	nein



Buchung löschen

Abbildung 3.25: Verwaltung gebuchter Übungen

```

SELECT * FROM buchungen, uebungen, benutzer
WHERE uebungen.uebung_id=buchungen.uebung_id
AND buchungen.user_id=benutzer.user_id
AND benutzer.login='user'
AND (
    upper(buchungen.uebung_id) LIKE upper('%nis%')
    OR upper(bezeichnung) LIKE upper('%nis%')
    OR upper(freigegeben) LIKE upper('%nis%')
    OR datum LIKE '%nis%'
    OR startzeit LIKE '%nis%'
    OR dauer LIKE '%nis%'
)
ORDER BY datum, startzeit

```

Sofern der Auftrag (at-Job), welcher der Buchung zugrunde liegt, noch nicht durchgeführt wurde, kann die Buchung vom Benutzer rückgängig gemacht werden. Man erkennt das leicht an der Schaltfläche "Buchung löschen". Bei Buchungen deren Übung bereits in der Vergangenheit liegt, verschwindet diese Schaltfläche.

Aus Sicherheitsgründen fragt das Script `delete.php` nach dem Klick auf den "Buchung löschen" Button, ob der Benutzer dies auch wirklich möchte. Dabei hat PHP über die GET-Methode die Buchungs-Nummer `buchungs_id` und die at-Job-Nummer `at_id` erhalten. Diese Buchungs-Nummer verwendet nun SQL, um die zu löschende Buchung, wie in Abbildung 3.26, dem Benutzer anzuzeigen.

Buchung löschen

Die folgende Buchung wird unwiederruflich gelöscht !!

Buchungs_ID	Kurzbezeichnung	Datum	Startzeit	Dauer	freigegeben
17	nis	2003-05-26	09:00	01:30	nein

Wollen Sie fortfahren ?

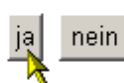


Abbildung 3.26: Buchung löschen

```
SELECT * FROM buchungen, uebungen, benutzer
WHERE uebungen.uebung_id=buchungen.uebung_id
AND buchungen.user_id=benutzer.user_id
AND buchungs_id=17
```

```
buchungs_id | uebung_id | datum      | startzeit | dauer      | freigegeben
-----+-----+-----+-----+-----+-----
          17 | nis       | 2003-05-26 | 09:00:00 | 01:30:00 | nein
(1 row)
```

Das danach gestartete Script `delete2.php` überprüft letztendlich wie sich der Anwender entschieden hat. Soll die Buchung rückgängig gemacht werden, löscht das `delete2.php` zuerst den at-Job über das System-Kommando

```
at -r 50
```

und entfernt danach den Datensatz aus der Datenbank.

```
DELETE FROM buchungen WHERE buchungs_id='17'
```

Den kompletten Vorgang kann man schematisch in [Abbildung 3.27](#) betrachten.

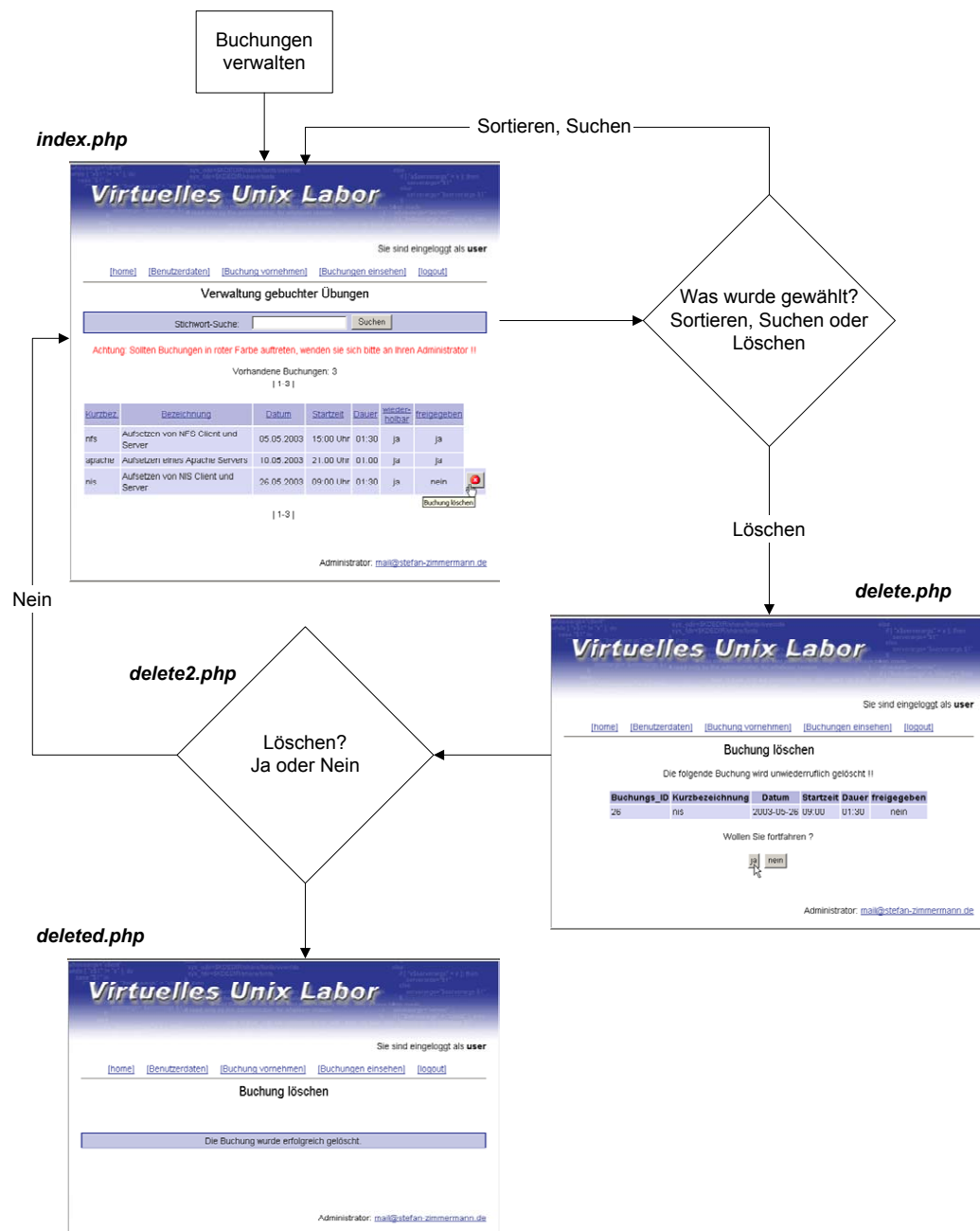


Abbildung 3.27: Ablaufdiagramm zum Verwalten der Buchungen im Anwenderbereich

3.5.3 Verwalten der Buchungen im Administrationsbereich

Im Gegensatz zu jedem Benutzer, der nur Einblick in seine eigenen Buchungen hat, kann der Administrator oder Dozent alle gebuchten Übungen kontrollieren. Ebenso wie im Anwenderbereich, ist es hier auch möglich, nach Stichworten zu suchen. In Abbildung 3.28 sucht das Script `index.php` im Verzeichnis `./admin/buchungen/` nach "nis". Das Modul "Buchungen" umfasst folgende Sripten:

- `index.php`
- `delete.php`
- `delete2.php`
- `deleted.php`
- `show.php`

Verwaltung gebuchter Übungen

Stichwort-Suche:

Achtung: Bei Buchungen in roter Farbe, war das Erstellen des at-Jobs fehlerhaft !!

Vorhandene Buchungen: 3
| 1-3 |







ID	Login	Kurzbez.	Bezeichnung	Datum	Startzeit	Dauer	freigegeben	at_id	
12	user	nis	Aufsetzen von NIS Client und Server	12.04.2003	12:00 Uhr	01:30	ja		 
17	user	nis	Aufsetzen von NIS Client und Server	26.05.2003	09:00 Uhr	01:30	nein	50	 
20	hugo	nis	Aufsetzen von NIS Client und Server	29.05.2003	18:00 Uhr	01:30	nein	53	 

Abbildung 3.28: Verwaltung gebuchter Übungen im Administrationsbereich

Das SQL-Statment dazu sieht wie folgt aus:

```
SELECT * FROM buchungen, uebungen, benutzer
WHERE uebungen.uebung_id=buchungen.uebung_id
AND buchungen.user_id=benutzer.user_id
AND (
    upper(buchungen.uebung_id) LIKE upper('%nis%')
    OR upper(bezeichnung) LIKE upper('%nis%')
    OR upper(freigegeben) LIKE upper('%nis%')
    OR datum LIKE '%nis%'
    OR int4(buchungs_id) LIKE '%nis%'
    OR int4(at_id) LIKE '%nis%'
    OR startzeit LIKE '%nis%'
    OR dauer LIKE '%nis%'
)
ORDER BY datum, startzeit
```

buchungs_id	user_id	uebung_id	bezeichnung
12	3	nis	Aufsetzen von NIS Client und Server
17	3	nis	Aufsetzen von NIS Client und Server
20	6	nis	Aufsetzen von NIS Client und Server

datum	startzeit	dauer	freigegeben	at_id
2003-04-12	12:00:00	01:30:00	ja	
2003-05-26	09:00:00	01:30:00	nein	50
2003-05-29	18:00:00	01:30:00	nein	53

(3 rows)

Am rechten Rand jeder Buchung erscheinen zwei Buttons. Sie sind jeweils mit einem Hyperlink hinterlegt und rufen somit verschiedene Scripten zum Anzeigen und Löschen der Daten auf. In Abbildung 3.29 wird die "at-Job einsehen" Schaltfläche gewählt.

17	user	nis	Aufsetzen von NIS Client und Server	26.05.2003	09:00 Uhr	01:30	nein	50	
----	------	-----	-------------------------------------	------------	-----------	-------	------	----	---------------------------------------------------------------------------------------

Abbildung 3.29: Klick auf Schaltfläche 'at-Job einsehen'

Das dadurch gestartete Script `show.php` bekommt über die `GET`-Methode die in der Datenbank gespeicherte at-Job-Nummer. Über diese Referenz kann der Systembefehl

```
at -c 50 2>&1
```

den in der Warteschlange stehenden Auftrag mit allen dazugehörigen Informationen auslesen. In Abbildung 3.30 erkennt man am Ende der Ausgabe das beim Buchen eingetragenen `deploy 17`.

Kommando:
`at -c 50 2>&1`

Ausgabe:
`#!/bin/sh
atrun uid=1001 gid=1000
mail root 0
umask 22
LD_LIBRARY_PATH=; export LD_LIBRARY_PATH
HOME=; export HOME
PATH=/sbin:/bin:/usr/sbin:/usr/bin; export PATH
cd /home/zis30131/public_html/vulab\230503/user/buchen || {
echo 'Execution directory inaccessible' >&2
exit 1
}
/home/zis30131/public_html/vulab\230503/bin/deploy 17`

[zurück](#)

Abbildung 3.30: Informationen zum at-Job einsehen

Der Administrator oder Dozent hat höhere Rechte, als ein User. Deshalb darf er auf alle Buchungen zugreifen und sie gegebenenfalls auch löschen. Um eine gebuchte Übung aus der Datenbank permanent zu entfernen, wird die Schaltfläche "Buchung löschen", wie es Abbildung 3.31 zeigt, betätigt.



17	user	nis	Aufsetzen von NIS Client und Server	26.05.2003	09:00 Uhr	01:30	nein	50	
20	hugo	nis	Aufsetzen von NIS Client und Server	29.05.2003	18:00 Uhr	01:30	nein	53	

Abbildung 3.31: Klick auf Schaltfläche 'Buchung löschen'

Mit Hilfe der `GET`-Methode gelangen ebenfalls Informationen über die Buchungsnummer (`buchungs_id`) und die at-Job-Nummer (`at_id`) zum Script `delete.php`.

Zur Sicherheit erzeugt dieses Script eine Abfrage, ob die Buchung auch wirklich gelöscht werden soll. Zusätzlich stellt das Script Informationen dar, die es aus der Datenbank erhält. Abbildung 3.32 zeigt die Ausgabe.

```
SELECT * FROM buchungen, uebungen, benutzer
WHERE uebungen.uebung_id=buchungen.uebung_id
AND buchungen.user_id=benutzer.user_id
AND buchungs_id=17
```

buchungs_id	user_id	uebung_id	datum	startzeit	dauer	freigegeben	at_id
17	3	nis	2003-05-26	09:00:00	01:30:00	nein	50

(1 row)

Buchung löschen

Die folgende Buchung wird unwiederruflich gelöscht !!

Buchungs_ID	Login	Kurzbezeichnung	Datum	Startzeit	Dauer	freigegeben	at_id
17	user	nis	2003-05-26	09:00:00	01:30:00	nein	50

Wollen Sie fortfahren ?

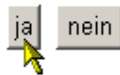


Abbildung 3.32: Soll die gebuchte Übung gelöscht werden ?

Ob nun auf Button "Ja" oder "Nein" geklickt wird – beide verweisen auf das Script `delete2.php`. Diese überprüft die Wahl und löscht bei Bedarf den at-Job mit dem System-Aufruf

```
at -r 50
```

aus der Warteschlange. Das SQL-Statement

```
DELETE FROM buchungen WHERE buchungs_id='17'
```

entfernt schließlich den Datensatz aus der Datenbank. Den gesamten Ablauf faßt Abbildung 3.33 nochmals zusammen.

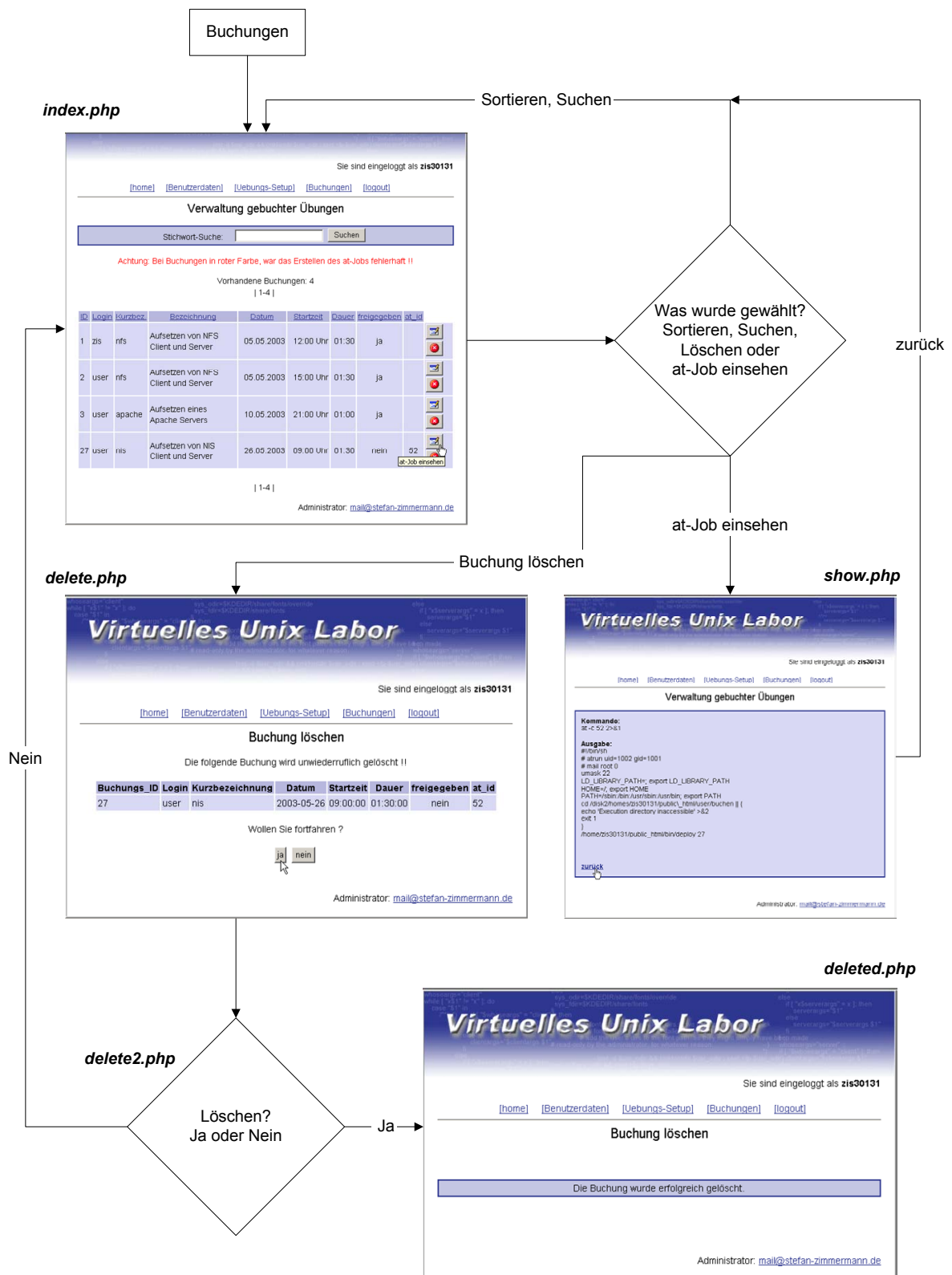


Abbildung 3.33: Ablaufdiagramm zum Verwalten der Buchungen im Administrationsbereich

Kapitel 4

Installation

Nachdem der Zugangsrechner, wie in Anhang [A](#), für das 'Virtuelle Unix Labor' vorbereitet wurde, geht es nun um die letzten Schritte zur funktionierenden Benutzerverwaltung.

4.1 Aufbau der Datenbank

Davon ausgehend, dass innerhalb PostgreSQL ein Benutzer `vulab` mit der dazugehörigen Datenbank `vulab` existiert, benötigt man den Inhalt des Verzeichnisses `backup` der beiliegenden CD.

Dieser Ordner enthält unter anderem die Dateien:

- `Makefile`
- `vulab.sql`
- `dropdb.sql`

Das `Makefile` bezieht sich auf den Zugangsrechner mit der IP-Adresse `194.95.108.11` und muss bei Bedarf angepasst werden.

Zum Anlegen der Tabellen in der Datenbank wird nun `make` aufgerufen. Zusätzlich trägt das Script einige vorgegebene Werte in die Tabellen ein, so dass sofort damit gearbeitet werden kann.

```
cd backup
make createTables
```

4.2 Aufbau des Web-Frontend

Die Benutzerverwaltung besteht im Großen und Ganzen aus vielen PHP-Skripten, die auch alle auf der beiliegenden CD im Verzeichnis `public_html` enthalten sind. Dieser Ordner muss auf den Zugangsrechner kopiert werden.

```
cp -R /mnt/cdrom/public_html ${HOME}/
```

Danach sollte das Verzeichnis `public_html` ausführbare Rechte für die Gruppe `others` erhalten. Dies ist Voraussetzung, damit der Web-Server darauf zugreifen kann.

```
chmod o+x ${HOME}/public_html
```

Die PHP-Skripte sind so aufgebaut, dass sie evtl. auftretende Fehler mitprotokollieren. Dazu muss der Ordner `log` im Verzeichnis `public_html` alle Rechte besitzen, da dort der Benutzer `www` Dateien anlegt.

```
chmod 777 ${HOME}/public_html/log
```

Um nun letztendlich das grafische Frontend des 'virtuellen Unix Labors' starten zu können, sollte noch die Konfigurationsdatei `config.php` kontrolliert bzw. angepasst werden. Wichtig ist dabei die Konfiguration der *eMail* und der *Datenbank*. Das nachfolgende Listing 4.1 zeigt einen Teil der aktuellen Einstellungen.

```
vi ${HOME}/public_html/config.php
```

Listing 4.1: Auszug aus der Konfigurationsdatei `config.php`)

```
<?php
2   ...
4   //////////////////////////////////////
   // eMail-Konfiguration
6
   $server = "smaug.fh-regensburg.de";
8   $adminName = "Administrator";
   $adminMail = "mail@stefan-zimmermann.de";
10  $replyName = "Stefan Zimmermann";
   $replyMail = "mail@stefan-zimmermann.de";
12  $Cc = "";
   $Bcc = "";
14
   //////////////////////////////////////
16  // Datenbank-Konfiguration
18
   $pg_host = "smaug.fh-regensburg.de";
   $pg_port = 5432;
```



```

20  $pg_db = "vulab";
    $pg_login = "vulab";
22  $pg_pass = "vulab";

24  ...

?>

```

Nach diesen Einstellungen sollte sich nun im Web-Browser unter der URL
 linebreak *http://smaug.fh-regensburg.de/~zis30131/index.php* die erste Seite
 des 'Virtuellen Unix Labors', wie in Abbildung 4.1, zeigen.



Abbildung 4.1: Startseite des 'Virtuellen Unix Labors'

4.3 Säubern der Datenbank — Cron

Erfahrungsgemäß gibt es immer wieder Personen, welche sich mit gefälschten Daten einen Zugang zum 'virtuellen Unix Labor' verschaffen wollen. Um zu verhindern, dass dadurch unnötige Einträge die Datenbank überfluten, wird bei der Erstanmeldung eine Bestätigungsnummer per eMail versandt. Diese Nummer ist in der Datenbank gespeichert (`benutzer:freischalt_secret`) und wird bei Bestätigung des Accounts mit dem String `bestaetigt` überschrieben.

Konnte, durch die Angabe falscher Daten, die eMail nicht zugestellt, und somit die Bestätigung nicht durchgeführt werden, bleibt der Code in der Datenbank bestehen (z. B. `stnv08wvsAHJM`). Das Script [4.2](#) `cleanDB.sh` identifiziert die nicht bestätigten Datensätze und löscht sie.

Um die Säuberung zu automatisieren, übernimmt `cron`¹ diese Aufgabe. Dazu befindet sich das Script `cleanDB.sh` im Ordner `cron` im Homeverzeichnis. Damit es ausgeführt werden kann, benötigt es spezielle Rechte. Sie werden mit folgendem Aufruf gesetzt.

```
% chmod 755 ${HOME}/cron/cleanDB.sh
```

Die Übergabe an `cron` erfolgt mittels:

```
% crontab -e
```

Im gestarteten Editor sollte nun folgender Eintrag vorgenommen werden:

#Minute	Stunde	Tag	Monat	Wochentag	Script
00	00	*	*	*	<code>\$(HOME)/cron/cleanDB.sh</code>

Zwischen den Angaben müssen immer Leerzeichen bzw. ein Tabulator stehen. Im angegebenen Fall wird das Script `cleanDB.sh` an jedem Tag der Woche um 00:00 Uhr gestartet.

Der beinhaltete SQL-Aufruf löscht alle Benutzer der Datenbank, welche nicht bestätigt haben und älter als acht Tage sind.

¹Bei `cron` handelt es sich um einen Unix-Dämon, welcher zeitgesteuert, immer wiederkehrende Aufgaben erledigt.

Listing 4.2: SQL-Script zum Säubern der Datenbank (cleanDB.sh)

```
2  #!/bin/sh
4  # Fuehrt einen SQL-Befehl in PostgreSQL aus, der alle Datensätze,
   # die älter als 8 Tage sind und nicht bestaetigt haben, löscht.
6
   # -c : nur ein Befehl und dann exit
8  # -d : ReferenzDB um eine Verbindung zur DB herzustellen
   # -U : der Benutzer, unter dem der Befehl ausgeführt wird
10
12  psql -U vulab \
      -c "DELETE FROM benutzer WHERE anmeldedatum <= ( now() - 8 ) \
14      AND freischalt_secret <> 'bestaetigt'" -d vulab
```

4.4 Vom www-user zu startende Scripte

Ebenfalls befinden sich auf der CD im Verzeichnis `public_html/bin` die Scripte, welche PHP als *www-user* startet. Dazu gehört unter anderem das Script `deploy`. Es sollte bereits, wie im vorhergehenden Kapitel [4.2](#), im Zuge des Aufbau des Web-Frontends, auf den Zugangsrechner kopiert worden sein.

Auch diese Dateien benötigen ausführbare Rechte:

```
chmod o+x ${HOME}/public_html/bin/*
```

Die auf der CD enthaltene Datei `deploy` ist allerdings nur ein dummy-Script, welches die Rechner- und Image-Namen für die benötigte Installation der Übungsrechner ausgibt. Dieses Script sollte später ersetzt werden.

Kapitel 5

Weiterentwicklung

Die Diplomarbeit umfasst neben der Benutzerverwaltung eine Umgebung, die es erlaubt, zusätzliche Funktionalitäten als weitere Module einzubinden. Dazu ist etwas Hintergrundwissen notwendig, damit die bereits vorhandenen Variablen richtig eingesetzt werden.

In den nachfolgenden Ausführungen geht hervor, wie ein neues Modul anhand eines Beispiels aufgebaut und dem System zugänglich gemacht wird.

5.1 Erzeugen eines neuen Menüpunkts

Das neue Modul "Beispiel" soll dem Anwender-Typ "user" zur Verfügung stehen. Deshalb kümmern wir uns als erstes um einen neuen Menü-Eintrag, über den die Funktionalität des hinzukommenden Moduls erreicht wird.

Dazu muss die Konfigurationsdatei `config.php` im Verzeichnis `user/` editiert werden.

```
% cd ${HOME}/public_html/user/  
% vi config.php
```

Indem Zeile 9 hinzugefügt wird, enthält das Array `$items` den Namen des neuen Menüpunkts. Die Ziffern, welche als Schlüsselwerte des Arrays `$items` fungieren, legen die Reihenfolge des Auftretens der Menüpunkte von links nach rechts fest. An Stelle 5 erscheint so der Eintrag "Beispiel", wie es auch Abbildung 5.1 zeigt.

Durch das Hinzufügen der Zeilen 34 und 35 kann auf der Startseite im Anwen-
derbereich (Abbildung 5.1) eine Beschreibung des Moduls angezeigt werden.

Listing 5.1: user/config.php — Hinzufügen eines neuen Moduls

```

<?php
2
// Menue-Eintraege in Reihenfolge ihres Auftretens
4 $items = array();
  $items[1] = "home";
6 $items[2] = "Benutzerdaten";
  $items[3] = "Buchung vornehmen";
8 $items[4] = "Buchungen einsehen";
  $items[5] = "Beispiel";
10 $items[7] = "logout";

12 // Zuweisung der Verzeichnisse auf die Menue-Namen
  // $access[Menue-Name] = Verzeichniss-Name
14 $access = array();
  $access["error"]           = "error";
16 $access["home"]           = "home";
  $access["Benutzerdaten"]   = "userdata";
18 $access["Buchung vornehmen"] = "buchen";
  $access["Buchungen einsehen"] = "buchungen";
20 $access["Beispiel"]       = "beispiel";
  // keine Zuweisung fuer logout notwendig
22 // $access["logout"] = "logout";

24 // Zuweisung der Beschreibung auf die Menue-Namen
  // $description[Menue-Name] = Beschreibung
26 $description = array();
  $description["home"]       = "Link auf diese Seite";
28 $description["Benutzerdaten"] = "Hier koennen Sie Ihre " .
    "persoennlichen Daten andaern";
30 $description["Buchung vornehmen"] = "Reservieren Sie sich " .
    "Aufgaben, die Sie demnaechst bearbeiten wollen";
32 $description["Buchungen einsehen"] = "Ueberpruefen Sie Ihre " .
    "getriggerten Buchungen";
34 $description["Beispiel"]       = "Bei diesem Punkt handelt es " .
    "sich lediglich um ein Beispiel.";
36 $description["logout"]        = "Verlassen Sie ihr Profil";

38 ?>

```

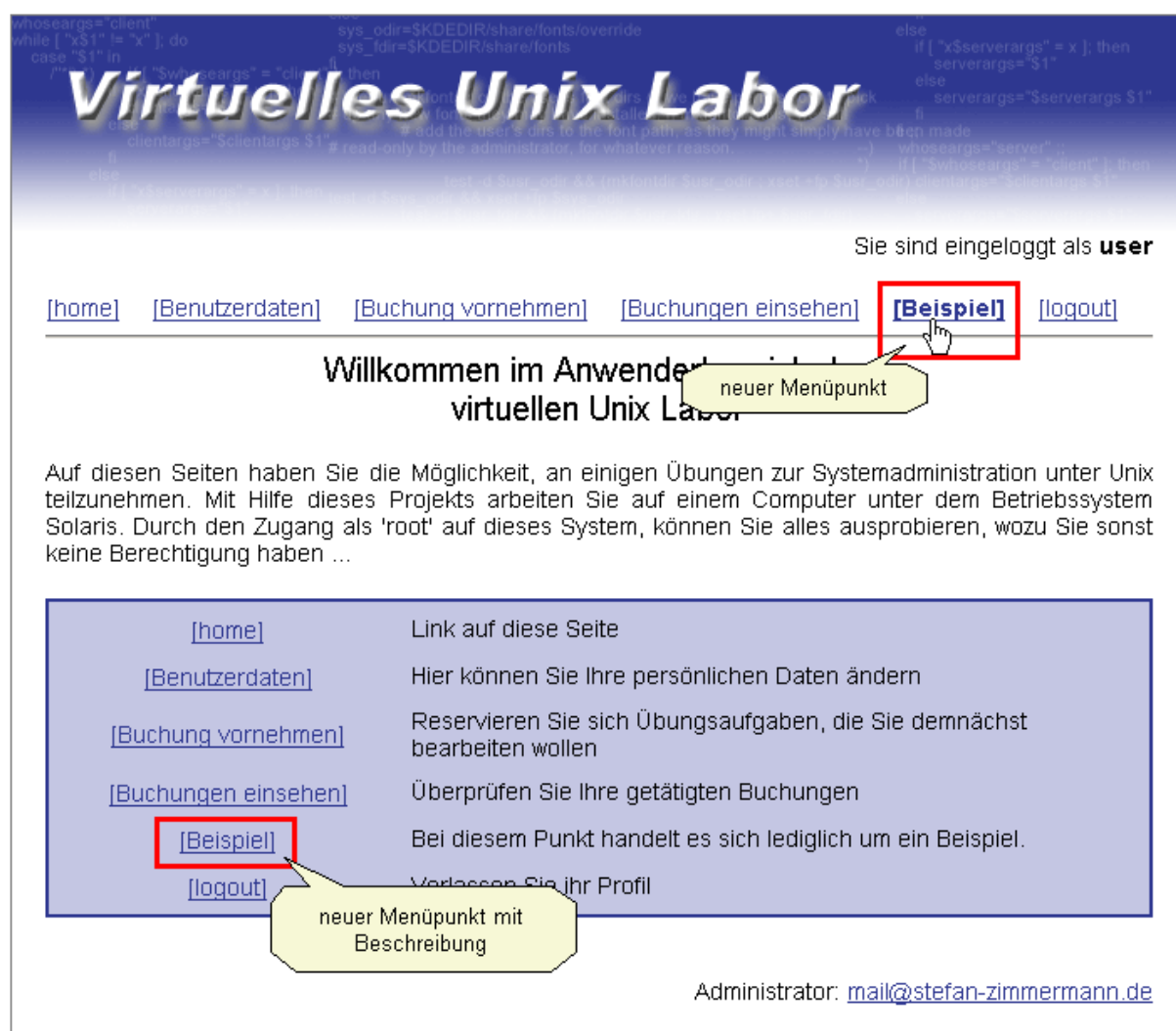


Abbildung 5.1: Hinzufügen eines neuen Menüpunkts im Anwenderbereich

Damit letztendlich beim Mausklick auf den neuen Menüpunkt auch der Inhalt des Moduls geladen wird, muss der Eintrag von Zeile 20 in der Datei `config.php` vorgenommen werden. Diese Zuweisung gibt Aussage darüber, in welchem Verzeichnis die Startdatei `index.php` gesucht wird.

Passend zur Zeile 20 sollte dann natürlich auch das benannte Verzeichnis mit der Datei `index.php` existieren.

```
% cd ${HOME}/public_html/user/
% mkdir beispiel
% cd Beispiel
% echo "Diese ist ein Beispiel" > index.php
```

Falls alle beschriebenen Punkte fehlerfrei ausgeführt worden sind, sollte nun nach Aufruf des Moduls "Beispiel" die Darstellung von Abbildung 5.2 sichtbar sein.

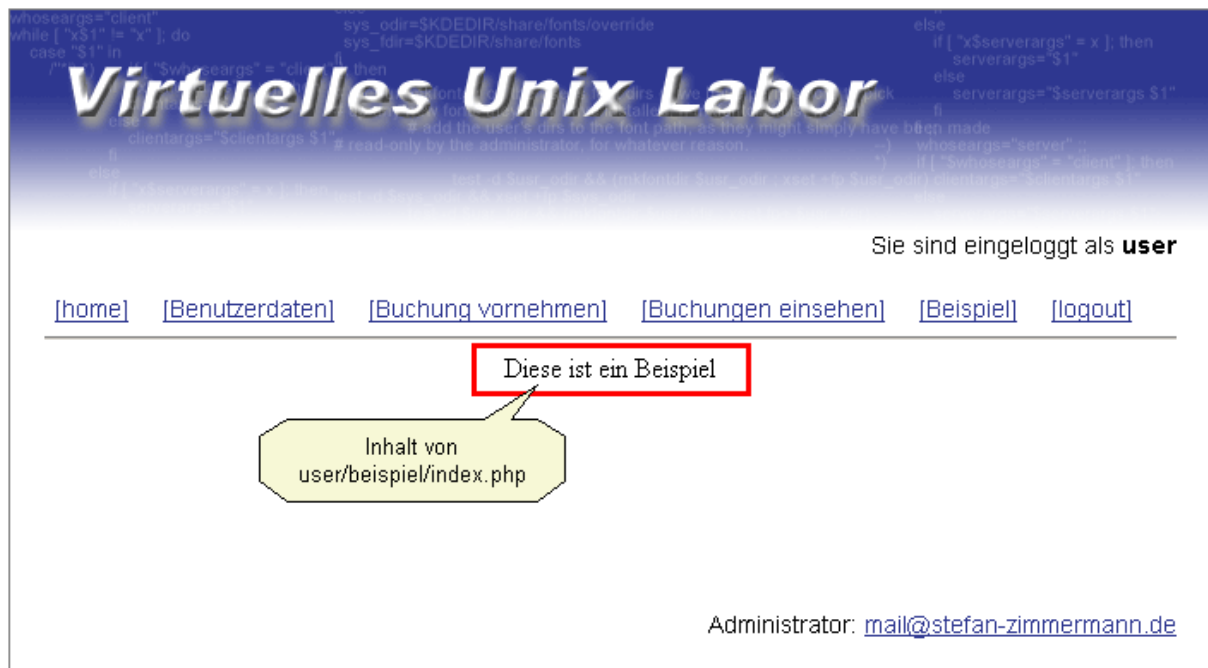


Abbildung 5.2: Inhalt des neuen Moduls 'Beispiel'

5.2 Aufbau eines neuen Moduls

Auf die Ausgabe von HTML-Tags wie `<HTML>` `<BODY>` `</BODY>` `</HTML>` kann an dieser Stelle verzichtet werden. Die Module werden, wie bereits im Kapitel 3.2 – Seitenaufbau beschrieben, in eine vorgefertigte HTML-Seite eingehängt.

5.2.1 Session-Handling

Besonders wichtig, um auf bereits verfügbare Variablen zugreifen zu können, ist das Initialisieren der Session. Die dazu benötigte Variable `$PHPSESSID` sollte dabei in der Hash-Tabelle POST bzw. GET enthalten sein.

Empfehlenswert ist es, generell den Code aus Listing 5.2 am Anfang jedes Scripts einzuhängen.

Listing 5.2: Code-Beispiel zum Session-Handling

```
<?php
2
   // hole Session-Variable
4   if (isset($_HTTP_GET_VARS["PHPSESSID"]))
       $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
6   elseif (isset($_HTTP_POST_VARS["PHPSESSID"]))
       $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];
8
   // hole vorhandene Session
10  session_start();
   $PHPSESSID=session_id();
12
   // falls Session verfuegbar, hole Session-Variablen
14  if (!empty($PHPSESSID))
  {
16      if (isset($_HTTP_SESSION_VARS["projectPath"]))
          $projectPath = $_HTTP_SESSION_VARS["projectPath"];
18
20      if (isset($_HTTP_SESSION_VARS["projectPath"]))
          $absoluteProjectPath =
              $_HTTP_SESSION_VARS["absoluteProjectPath"];
22
24      if (isset($_HTTP_SESSION_VARS["menu"]))
          $menu = $_HTTP_SESSION_VARS["menu"];
  }
26
   require_once $absoluteProjectPath . "functions.php";
28
   ...
30 ?>
```

Nachdem die `if`-Bedingungen aus Zeile 4 bis 7 die Session-ID eingelesen haben, kann `session_start()` in Zeile 10 die Sitzung neu initialisieren.

Die nächsten `if`-Abfragen ab Zeile 14 prüfen schließlich die Verfügbarkeit bestimmter Session-Variablen und schreiben diese auf lokale Variablen zurück. In der Hash-Tabelle der Session sollten deshalb folgende Schlüssel vorhanden sein.

- projectPath
Der Pfad auf das Projektverzeichnis, welcher für die URL verwendet wird (z. B. /~zis30131/vulab/).
- absoluteProjectPath
Der absolute Pfad auf das Projektverzeichnis (z. B. /home/zis30131/public.html/vulab/).
- menu
Der Benutzer-Typ, bzw. das verwendete Menü.
- auth
Ein Flag, das die erfolgreiche Authentifizierung anzeigt.
- login
Der eingeloggte Benutzer.

Zeile 27 verwendet bereits eine über die Session erhaltenen Variable. Das PHP-Script bindet an dieser Stelle die Datei `functions.php` ein, welche dem Entwickler viele Funktion zur Verfügung stellt.

5.2.2 Error-Handling

Um zu verhindern, dass evtl. auftretende PHP-Fehler dem Anwender nicht zu Gesicht kommen aber dem Administrator langfristig erhalten bleiben, wird das Code-Fragment aus Listing 5.3 verwendet.

Listing 5.3: Code-Beispiel zum Error-Handling

```
<?php
2     ...

4     // Fehlerbehandlung
    ob_start('ob_gzhandler');
6     set_error_handler('handleErrors');
    $scriptName = $menu . "/buchen/delete2.php";
8
    ...
10 ?>
```

Die Anweisung `ob_start(. . .)` in Zeile 5 ist zwar nicht so zwingend, kann sich aber als äußerst hilfreich erweisen. Viele Browser unterstützen Komprimierungsverfahren für Web-Seiten. D. h. der Server komprimiert den HTML-Code der Seite und sendet ihn an den Client. Dieser entpackt ihn und stellt die Informationen dar. Um eine automatische Kompression zu erhalten, wird `ob_start('ob_gzhandler')` verwendet. Zusätzlich kann diese Funktion, im Falle eines auftretenden Fehlers, Ausgaben des Scripts bis zur vollständigen Beendigung unterdrücken.

Die auftretenden Fehler gibt PHP standardmäßig als Text an den Client weiter, so dass dieser die Meldungen im Browser lesen kann. Diese Form der Fehlerbenachrichtigung sollte nach Möglichkeit unterdrückt werden. Dazu beinhaltet die Datei `functions.php` eine Funktion `handleErrors(. . .)` (siehe Anhang D). Durch diese Funktion ist ein Error-Handler definiert, welcher alle Fehlermeldungen aufbereitet und in ein log-File schreibt.

Damit jedoch die Funktion `handleErrors(. . .)` aufgerufen wird, muss der Error-Handler mit `set_error_handler()` gesetzt werden. Dies geschieht in Zeile 6.

Ein Problem von PHP ist es, die genaue Herkunft des aufgetretenen Fehlers zu analysieren. Aus diesem Grund fragt der Error-Handler nach einer Variablen `$scriptName`, welche den aktuelle Scriptnamen als Text enthält (Zeile 7). Dieser Text wird später in der log-Datei vermerkt, um eine längere Fehlersuche zu vermeiden.

5.2.3 Aufruf weiterer Scripte

In den wenigsten Fällen steckt die gesamte Funktionalität des Moduls in einer PHP-Datei. Zum Datenaustausch werden weitere Scripte benötigt, die in einer bestimmten Form aufgerufen werden sollten.

Die Übergabe bestimmter Parameter ist dabei zwingend erforderlich. Dazu gehören auf jeden Fall die drei Variablen:

- PHPSESSID
Die Session-ID zum Identifizieren der Sitzung.
- menu
Der Benutzer-Typ, bzw. das verwendete Menü.
- content
Das gewünschte Modul, bzw. ein spezielles Script innerhalb des Moduls.

Bei der Verwendung dieser Variablen ist darauf zu achten, dass die aufzurufende Datei in der Regel das Basis-Script `index_session.php` ist.

Aufruf via Link

Ein angezeigter Link, welcher ein Modul aufruft, hat demnach wie folgt auszu-sehen:

```
<a href="/~zis30131/vulab/index_session.php?
    menu=user&content=Beispiel&
    PHPSESSID=859b14022df5cba0e35149b8aa6ad807">
Link auf das Modul Beispiel</a>
```

Im PHP-Code könnte das so aussehen:

```
<a href="<?php
    echo $projectPath . "index_session.php?menu=" . $menu .
    "&content=Beispiel&PHPSESSID=" . $PHPSESSID ?>">
Link auf das Modul Beispiel</a>
```

Man beachte: Der Inhalt der Variablen `content` (hier: *Beispiel*), muss mit dem Eintrag im Array `items` der Datei `user/config.php` (siehe Listing 5.1 übereinstimmen. Die Variablen `$projectPath` und `$menu` wurden, wie in Listing 5.2

beschrieben, aus der Session gelesen.

Dieser beschriebene Aufruf des Moduls "Beispiel" setzt allerdings voraus, dass keine Header-Informationen übertragen werden. Es ist lediglich auf die Übertragung von reiner Text beschränkt, da das Script `index_session.php` bereits einen Header gesendet hat. Dieses Problem wurde zuvor in Kapitel 3.1.2 — Pufferung des Ausgabestroms angesprochen.

Ein Script, welches benötigt wird, um Header-Informationen (z. B. Cookies) an den Client zu senden, darf nicht über die Datei `index_session.php` aufgerufen werden. Dazu wird die Referenz des Links direkt auf das auszuführende Script gesetzt.

```
<a href="<?php
    echo $projectPath .
    "/user/beispiel/validate.php?PHPSESSID=" .
    $PHPSESSID ?>">
```

Link auf das Check-Script

Um nicht immer auf die Einsprungsdatei `user/beispiel/index.php` angewiesen sein zu müssen, besteht eine zusätzliche Möglichkeit ein PHP-Script aufzurufen.

```
<a href="<?php
    echo $projectPath . "index_session.php?menu=" . $menu .
    "&content=beispiel/script2&PHPSESSID=" . $PHPSESSID ?>">
```

Link auf ein weiteres Script

Bei der Angabe `beispiel/script2` handelt es sich um eine direkte Pfadangabe. Sie wird von `index_session.php` aufgelöst und zusammen mit der Variable `menu` verarbeitet.

Aufruf via Formular

Der Aufruf eines Scripts über eine Formular-Schaltfläche sieht ähnlich aus. Die benötigten Variablen werden allerdings als sog. hidden-Fields übergeben.

Listing 5.4: Beispiel eines Formulars mit versteckten Feldern

```

2 <form action="<?php echo $projectPath . "index_session.php" ?>" method="post">
    <input type="hidden" name="menu" value="<?php echo $menu ?>">
4    <input type="hidden" name="content" value="beispiel/script2">
    <?php
6        if (!empty($PHPSESSID))
            echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" . $PHPSESSID . \">\n";
8    ?>
    <input type="text" name="find" size="20" value="<?php echo $find; ?>">&nbsp;
10    <input type="submit" name="" value="Suchen"> &nbsp;
</form>

```

Das Listing 5.4 zeigt den Aufruf des Scripts `user/beispiel/script2.php` über die Datei `index_session.php`.

Zusammenfassung

Grundsätzlich bestehen drei Möglichkeiten zum Aufruf eines weiteren PHP-Scripts. Tabelle 5.1 fasst die verschiedenen Fälle zusammen.

URI	menu	content	Beschreibung
<code>index_session.php</code>	<code>user</code>	Beispiel	Direkter Aufruf des Moduls über den Einsprungpunkt <code>user/beispiel/index.php</code> .
<code>index_session.php</code>	<code>user</code>	<code>beispiel/script2</code>	Aufruf des Scripts <code>user/beispiel/script2.php</code> .
<code>validate.php</code>	–	–	Aufruf des Scripts <code>validate.php</code> .

Tabelle 5.1: Zusammenfassung der Parameterarten

Kapitel 6

Stolpersteine auf dem Weg zum Erfolg

Zu Beginn der Diplomarbeit waren mir viele Komponenten des Projekts unbekannt. Ich hatte noch keine Erfahrung im Umgang mit der Programmiersprache PHP. Das Datenbank-Management-System PostgreSQL kannte ich nur dem Namen nach und von \LaTeX wusste ich gerade so viel, dass dies nichts mit dem Kunststoff zu tun hatte.

So war es voraussehbar, dass mit diesem Projekt einiges an Arbeit auf mich zu kommen würde, und ich viele Schwierigkeiten zu meistern hatte. Dieses Kapitel widme ich deshalb jenen Problemen, welche mir das Leben schwer machten.

6.1 Keine Verbindung zu PostgreSQL — `pg_hba.conf`

Eine Bedingung der Diplomarbeit war es, alle Daten der Benutzerverwaltung in einer PostgreSQL-Datenbank zu halten. Da es sich bei der Programmiersprache um PHP handelte, musste PHP die Verbindung zu PostgreSQL aufnehmen.

Doch leider war die zuständige PHP-Funktion `pg_connect(...)` erfolglos. Sie sollte laut Dokumentation ein Connection-Objekt zurück liefern, doch es ergab sich immer nur `FALSE`.

Nach weiteren Recherchen fand ich heraus, dass PostgreSQL nicht richtig konfiguriert war. Der Fehler lag in der Datei `pg_hba.conf`. Sie ermöglicht die Authentifikation zwischen dem PostgreSQL-Server und der Client-Anwendung. In der Datei `pg_hba.conf` stehen eine Reihe von Einträgen, welche die zugreifenden Hosts mit ihren Rechten versehen [Worsley and Drake (2002)].

Jedes Mal, wenn sich PHP auf die Datenbank verbinden möchte, wird die Datei `pg_hba.conf` gelesen und im Falle eines passenden Eintrags der Zugang gewährt.

Also wurde der Host in die Datei `/usr/pkg/pgsql/data/pg_hba.conf` eingetragen.

```
host          all          194.95.108.11      255.255.255.255    md5
```

Doch PHP konnte die Verbindung nicht herstellen. Durch Zufall stellte ich fest, dass, wider der Dokumentation, die Datei `pg_hba.conf` nicht bei jeder Anfrage gelesen wird, sondern ein Neustart von PostgreSQL die Rechte der Hosts in den Server einbindet.

6.2 Nicht existierende Methode — `pg_query()`

Nachdem eine Verbindung zur Datenbank bestand, konnte das erste SQL-Statement abgesetzt werden. Doch die in PHP dokumentierte Funktion `pg_query(...)` existierte nicht.

```
Fatal error: Call to undefined function: pg_query()
```

Langes Suchen ergab, dass die ehemalige Funktion `pg_exec()` zwar durch `pg_query()` ersetzt wurde, aber aus Gründen der Kompatibilität immer noch verfügbar ist. Leider findet sich zur alten Funktion `pg_exec()` keine Dokumentation mehr.

6.3 Der Unterschied zwischen `char` und `varchar` bei PostgreSQL

Wer glaubt `"admin" == "admin"`, der irrt !!

Das Verhalten des folgenden Code-Fragments (Listing 6.1) widersprach jeglicher Logik. Vor allem das Ergebnis der `if`-Bedingung von Zeile 21 machte keinen Sinn.

Listing 6.1: Auszug zur Berechtigungs-Prüfung

```

<?php
2      ...

4      // baue Verbindung zur Datenbank auf
$conn_string = "host=$pg_host dbname=$pg_db user=$pg_login password=$pg_pass";
6      $connection = pg_connect($conn_string);

8      // frage nach Benutzer-Typ
$sql = "select typ from benutzer where login='" . $login . "'";
10     $result = executeStatement($connection, $sql);

12     // holt die erste Zeile des Resultats
$array = pg_fetch_array($result, 0);
14

16     // Debugging
echo "menu: $menu<br>\n";
echo "auth: $auth<br>\n";
18     echo "array[typ]: " . $array['typ'] . "<br>\n";

20     // ueberpruefe ob angefordertes Menue mit Berechtigung ueberein stimmt
if (!$auth || $array['typ'] != $menu)
22     {
        header ("Location: " . getLocation("logout.php") . "?PHPSESSID=" . $PHPSESSID);
24
        exit;
26     }

28     ...
?>

```

Die zur Überprüfung von Manipulationen eingesetzte Abfrage (Zeile 21), löste jedes Mal aus, und führte zu einem ungewollten Logout.

Doch wo lag der Fehler ?

Die Zeilen 16 bis 18, welche zum Prüfen der Variablen eingefügt wurde, ergaben:

```

menu: admin
auth: 1
array(typ): admin

```

Wieso ist denn "admin" nicht gleich "admin" ?

Die Lösung fand sich im HTML-Code:

Listing 6.2: HTML-Code der fraglichen Ausgabe

```
1 menu: admin<br>
2 auth: 1<br>
3 array(typ): admin <br>
```

Der Datentyp `char` in der PostgreSQL-Datenbank verursachte die vielen Leerzeichen. Nachdem schließlich alle Datentypen in `varchar` geändert waren, funktionierte die gewünschte Abfrage.

6.4 Das Problem mit den globalen Variablen

Viele Fachbücher über PHP machen sich das Leben leicht, indem sie von einer PHP-Konfiguration ausgehen, in der globale Variablen aktiviert sind. Im Grunde genommen ist der Gedanke gar nicht so verkehrt. Einige Provider verwenden tatsächlich globale Variablen. Trotzdem bin ich der Meinung, man sollte keine Sonderfälle als gegeben hinnehmen, sondern eher das Standardverfahren beschreiben. Außerdem stellen globale Variablen, wie bereits in Kapitel 3.1.1 erwähnt, ein Sicherheitsrisiko dar.

Doch wie es so ist, versuchte ich zuerst den Beispielen der Fachliteratur zu folgen und war schließlich frustriert als der Zugriff auf die Variable der URL

```
http://smaug.fh-regensburg.de/~zis30131/testScript.php?name=hugo
```

mit `<?php echo $name ?>` nicht funktionierte.

Nach einigen Nachforschungen ergab sich die Lösung:

Der Inhalt der Variable "versteckte" sich im Hash `$HTTP_GET_VARS['name']`.

6.5 Serverseitige Pufferung des Ausgabestroms

Ebenfalls, wie das Problem mit den globalen Variablen, wurde auch die serverseitige Pufferung des Ausgabestroms schon in Kapitel 3.1.2 angesprochen.

Der PHP-Code, mit dem ich begann, generierte erst verschiedene Ausgaben, bis es zu einer `if`-Bedingung kam. Diese traf dann die Entscheidung, eine neue HTML-Seite zu laden. Ein kleine Beispiel dafür findet sich in Listing 6.3.

Listing 6.3: Beispiel für schlechten PHP-Code

```
<?php
2
...
4 echo "<html><body>\n";
...
6
8 if (count($errors) > 0)
{
    header ("Location: " . $projectPath .
10         "index_session.php?menu=default&content=login&PHPSESSID=" .
        session_id());
12 }
else
14 {
    echo "...";
16 }
18
...
20 echo "</body></html>\n";
...
22 ?>
```

Da auf dem aktuellen Web-Server in der Initialisierungs-Datei `php.ini` der Eintrag `output_buffering = 4096` vorhanden war, welche die Größe des Ausgabepuffer setzt, wurde das Problem nicht erkannt. Die Pufferung führte zu einer Reorganisation der Ausgaben, welche letztendlich in richtiger Reihenfolge an den Client gesendet wurden. Doch als ich die PHP-Dateien auf einen Server portiert hatte, welcher über kein "Output Buffering" verfügte, ging gar nichts mehr.

Warning: Cannot send session cache limiter - headers already sent

Die Folge daraus, war ein komplettes Umstrukturieren des Programm-Codes. Die Header-Informationen werden nun vor der Ausgabe des generierten HTML-Code erzeugt.

6.6 Session-Handling ohne `--enable-trans-sid`

Über das Thema Session-Handling haben sich die Autoren der PHP-Bücher auch nur mäßig ausgelassen. Es gehen viele von einer PHP-Engine aus, in der beim Kompilieren der Parameter `--enable-trans-sid` gesetzt war. Eine so erzeugte Version des PHP-Interpreter kümmert sich automatisch um die Weitergabe der benötigten Session-ID. Daß man sich ohne diese "transiente Session-ID" um den Erhalt der Sitzung selbst kümmern müsse, wird oft verschwiegen.

6.7 Fazit

Alles in allem war für mich diese Diplomarbeit eine Herausforderung. Ich hatte mit vielen, mir noch unbekannten Komponenten zu tun. Deshalb denke ich, es ist ganz normal, wenn sich anfangs die Problem häufen. Doch wie schon ein Sprichwort sagt: "Aus Fehlern lernt man".

Anhang A

Aufbau des Systems

Wie in Kapitel 1.2 beschrieben, ist die Basis des Projekts die Sparc Station 5. Auf ihr laufen Dienste wie Web Server und Datenbank. Das zugrunde liegende Betriebssystem ist NetBSD¹.

Ausgehen von einer bereits bestehenden Installation von NetBSD, werden zusätzlich die Pakete Apache, PHP und PostgreSQL benötigt. Diese Installationen halten sich relativ einfach, da der Quell-Code nicht wie bei anderen Unix-Systemen erst kompiliert werden muss. Für NetBSD stehen bereits übersetzte Binärdateien der Produkte zur Verfügung, die dem Betriebssystem lediglich hinzugefügt werden müssen. Dies geschieht mit dem Befehl `pkg_add`. Vorsichtigen Administratoren steht es jedoch frei, die Pakete trotzdem selbst zu kompilieren. Die Packages-Sourcen stehen zur Verfügung und können wie üblich mit `make - make install` übersetzt werden.

Zum Installieren von Apache geht man z. B. wie folgt vor:

```
# cd /usr/pkgsrc/www/apache
# make
# make install
```

A.1 Web Server Apache

A.1.1 Installation

Das Paket `apache-1.3.26nb4` beinhaltet alle notwendigen Dateien für eine funktionierende Apache-Distribution. Um sie auf NetBSD zu installieren muss le-

¹ <http://www.netbsd.org>

diglich im Verzeichnis mit der Datei `apache-1.3.26nb4.tgz` der Befehl

```
% pkg_add apache-1.3.26nb4
```

einggegeben werden.

Anschließend kopiert man mit

```
% cp /usr/pkg/etc/rc.d/apache /etc/rc.d/apache
```

das Script `/usr/pkg/etc/rc.d/apache` nach `/etc/rc.d/apache`.

Schließlich muss noch in der Konfigurations-Datei `/etc/rc.conf` die Zeile

```
apache=yes
```

hinzugefügt werden.

Der Web Server startet nun bei jedem Neustart des Systems automatisch.

Um Apache von Hand zu starten, benutzt man die Anweisung

```
% /etc/rc.d/apache start
```

oder

```
% /etc/rc.d/apache restart
```

Um ihn zu stoppen, gibt man

```
% /etc/rc.d/apache stop
```

ein.

A.1.2 Konfiguration und Testbetrieb

Alle notwendigen Einstellungen des Apache befinden sich in der Konfigurationsdatei `/usr/pkg/etc/httpd/httpd.conf`. Sie sind bereits durch die Installation voreingestellt und müssen an dieser Stelle nicht verändert werden.

Nachdem der Web Server erfolgreich gestartet wurde, kann man ihn nun über einen Web Browser auf seine Funktionalität testen. Dazu gibt man im Browser einfach die URL² (Adresse) des Servers an. Dies geschieht in unserem Fall mit

```
http://localhost/
```

² **U**niversal **R**esource **L**ocator

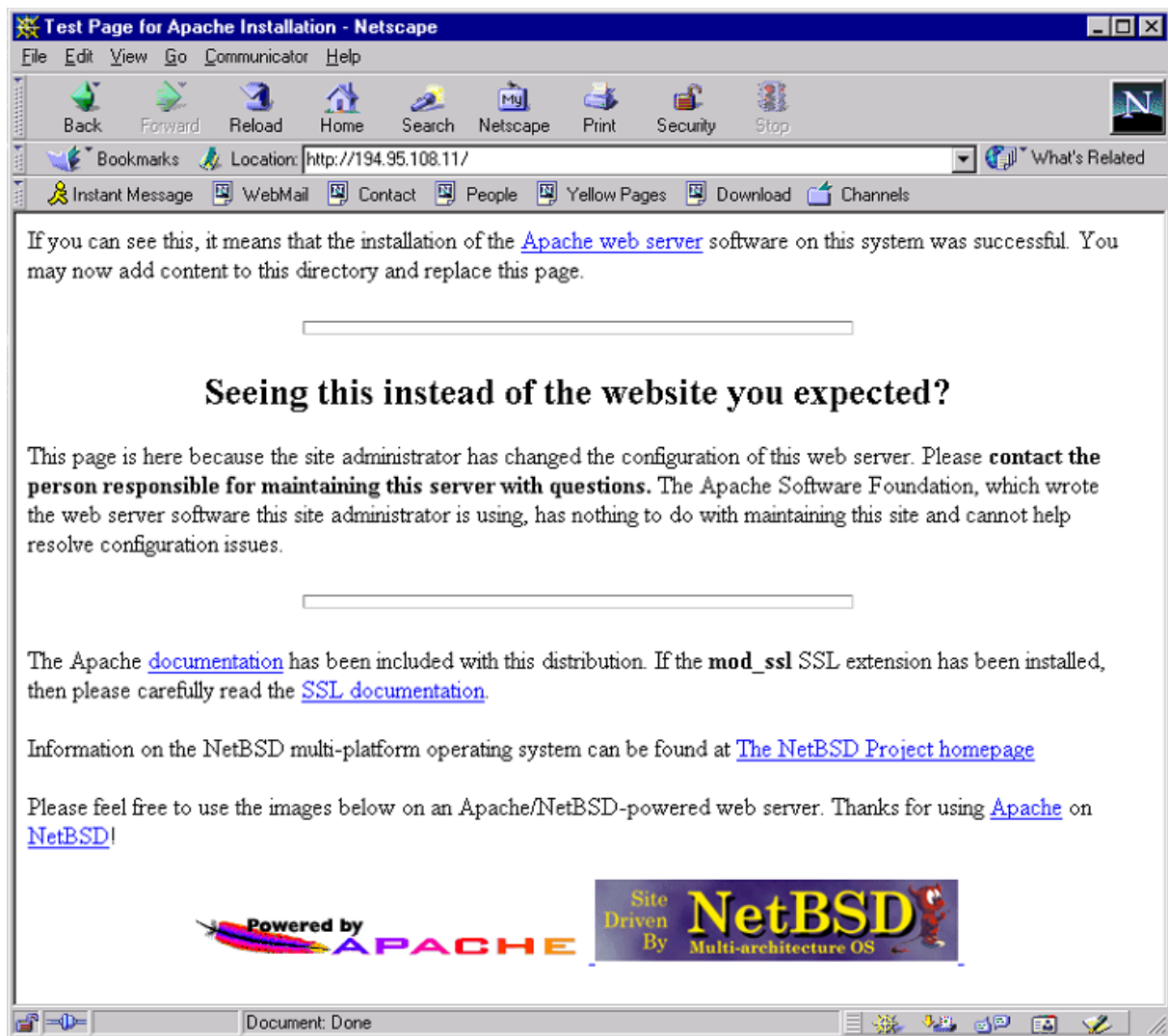


Abbildung A.1: Testseite des Apache Web Server

auf dem selben Rechner oder mit

`http://IP-Adresse/`

auf einer fremden Maschine.

In unserem Fall handelt es sich dabei um die IP-Adresse 194.95.108.11.

Nach Auswahl von

`http://194.95.108.11/`

wird wie in Abbildung A.1 im Browser eine Testseite des Apache Web Server angezeigt.

A.2 Script Sprache PHP

A.2.1 Installation

Die Installation der Programmiersprache PHP verhält sich ähnlich wie die von Apache. Dazu werden als Basis zwei Pakete benötigt:

- `php-4.1.2`
Basispaket der Scriptsprache PHP4
- `ap-php-4.1.2`
Apache Modul für PHP4

Befindet man sich im Verzeichnis mit den genannten Dateien gibt man Folgendes auf der Kommandozeile ein:

```
% pkg_add php-4.1.2
% pkg_add ap-php-4.1.2
```

PHP wird nun ins System eingebunden. Allerdings reicht dies nicht aus, um den gewünschten Erfolg zu erzielen. Da PHP mit dem im Anschluss (Anhang [A.3](#)) installierten Datenbank Management System kommunizieren soll, verlangt PHP eine Schnittstelle zur Datenbank. Sie wird mit dem Paket `php-pgsql-4.1.2` geliefert. Zusätzlich verwenden wir für das Projekt noch weitere PHP Module. Sie finden sich in folgenden Paketen:

- `php-pgsql-4.1.2`
PHP4 Erweiterung für PostgreSQL Datenbanken
- `php-pcre-4.1.2`
PHP4 Erweiterung für Perl-kompatible reguläre Ausdrücke
- `php-session-4.1.2nb1`
PHP4 Erweiterung für Session-Handling

Auch sie werden dem Betriebssystem NetBSD hinzugefügt:

```
% pkg_add php-pgsql-4.1.2
% pkg_add php-pcre-4.1.2
% pkg_add php-session-4.1.2nb1
```


A.2.2 Konfiguration

Nach der Installation von PHP auf dem Web-Server muss dieser für die Ausführung von PHP-Programmen angepasst werden. Dazu gibt es mehrere Möglichkeiten.

PHP kann als ein autonomes CGI³ Script direkt in den Server oder über die native Server API⁴ (SAPI) eingebunden werden. Zusätzlich es auch möglich PHP so zu konfigurieren, damit es als Java Servlet Engine fungiert [Lerdorf and Tatroe (2002)].

Die CGI Version ist zwar wesentlich zuverlässiger, aber auch langsamer als die SAPI Implementierung, da sie bei jeder Anfrage erneut geladen wird. Dagegen wird die SAPI Implementierung nur einmal geladen und erzeugt bei jeder Anfrage einen neuen Prozess. Obwohl Letzteres effizienter ist, kann diese feste Kopplung an den Server, falls Speicherlöcher auftreten, den kompletten Dienst abschießen [Lerdorf and Tatroe (2002)].

In Hinsicht auf die Geschwindigkeit, ist zu diesem Projekt die Entscheidung auf die SAPI Implementierung gefallen. Dazu müssen in der Konfigurationsdatei des Apache `httpd.conf` im Verzeichnis `/usr/pkg/etc/httpd` folgende Zeilen ergänzt werden:

```
LoadModule php4_module lib/httpd/mod_php4.so
AddModule mod_php4.c
AddType application/x-httpd-php .php
```

Diese Einträge sind nötig, damit der Web Server das PHP als Apache Modul lädt. Die zusätzlich installierten PHP Module werden über die Initialisierungsdatei `php.ini` im Verzeichnis `/usr/pkg/etc` eingebunden. Dazu sind folgende Einträge nötig:

```
extension=pgsql.so
extension=pcre.so
extension=session.so
```

Da PHP standardmäßig keine Fehlermeldungen ausgibt, wird dadurch das Debuggen sehr erschwert. Deßhalb ist es sinnvoll sich zu vergewissern, ob in der

³ Common Gateway Interface

⁴ Application Programming Interface

Datei `php.ini` das Flag `display_errors` auf `on` gesetzt ist.

```
display_errors = On
```

Anmerkung: Aus Gründen der Sicherheit sollte diese Funktion im Produktionsbetrieb wieder abgeschaltet werden.

Um die Änderungen wirksam zu machen, ist ein Neustart von Apache notwendig.

```
% /etc/rc.d/apache restart
```

A.2.3 Testbetrieb

Bevor mit der Installation fortgeschritten wird, wollen wir uns von der ordnungsgemäßen Funktionalität des Web Servers überzeugen. Dazu benötigen wir ein Web Verzeichnis, in welches die Testseite gespeichert wird. Dieser Ordner heißt `public_html` und liegt im Home-Verzeichnis.

```
% cd
% mkdir public_html
% chmod 755 .
% chmod 755 public_html
```

Mit `chmod 755` (change file modes) werden die Zugriffsrechte auf das Web Verzeichnis für alle freigegeben.

Nun wechseln wir in das Verzeichnis `public_html` und erstellen dort eine Datei mit Namen `phpinfo.php`. Sie soll den Web Server anweisen, Informationen über die installierte PHP Version auszugeben.

```
% cd public_html
% echo "<?php phpinfo() ?>" > phpinfo.php
% chmod 644 phpinfo.php
```

Wird im Web Browser jetzt als URL

```
http://194.95.108.11/~zis30131/phpinfo.php
```



Abbildung A.2: Informationsseite über PHP

angegeben, erscheint eine Informationsseite zur installierten PHP Version wie es Abbildung A.2 zeigt. Dabei entspricht `zis30131` der Benutzerkennung, unter der die Testdatei erzeugt wurde.

A.3 DBMS PostgreSQL

A.3.1 Installation

Um das Datenbank Management System PostgreSQL zu installieren, wird das Paket `postgresql-7.2.1` benötigt.

Es beinhaltet die Einzelpakete:

- `postgresql-lib-7.2.1`
PostgreSQL database headers and libraries
- `postgresql-client-7.2.1`
PostgreSQL database client programs
- `postgresql-server-7.2.1`
PostgreSQL database server programs
- `postgresql-docs-7.2.1`
PostgreSQL database system documentation

Befindet man sich im Verzeichnis, in der die Datei `postgresql-7.2.1.tgz` liegt, startet die Installation von PostgreSQL automatisch, wenn

```
% pkg_add postgresql-7.2.1
```

einggegeben wird.

Jetzt muss das Script `/usr/pkg/etc/rc.d/pgsql` nach `/etc/rc.d/pgsql` kopiert werden.

```
% cp /usr/pkg/etc/rc.d/pgsql /etc/rc.d/pgsql
```

Schließlich ergänzt man noch die Konfigurations-Datei `/etc/rc.conf` um die Zeilen

```
pgsql=yes  
pgsql_flags="-i"
```

Bei einem Neustart des Systems startet nun der PostgreSQL Multi-User Server `postmaster` automatisch.

Um `postmaster` von Hand zu starten, gibt man

```
% /etc/rc.d/pgsql start
```

bzw.

```
% /etc/rc.d/pgsql restart
```

ein.

Um ihn zu stoppen, wird

```
% /etc/rc.d/pgsql stop
```

verwendet.

A.3.2 Client Authentifizierung

Die Client Authentifizierung ist ein zentrales Merkmal von PostgreSQL [Worsley and Drake (2002)]. Ohne sie wäre ein Fernzugriff nicht möglich bzw. es würde blindlings jedem User Zugriff auf die Datenbank gewährt und jeder könnte Daten lesen oder sie sogar gleich ändern.

Versucht eine Anwendung eine Verbindung zu PostgreSQL aufzubauen, müssen der Benutzername und der Datenbankname angegeben werden. Das Passwort ist optional und hängt von der Art der verwendeten Authentifizierung ab. In der Regel wird bei lokalen Zugriffen auf ein Passwort verzichtet.

Bei einer aufbauenden Verbindung wird zuerst die Datei `pg_hba.conf` gelesen und danach die Maschine überprüft, die den Aufbau wünscht. Die Kriterien zur Authentifizierung des anfragenden Rechners stehen in `pg_hba.conf`.

PostgreSQL bietet zahlreiche unterschiedliche Arten der Client Authentifikation. Um sie in `pg_hba.conf` zu spezifizieren, müssen die Einträge von folgender Form sein:

```
host DBNAME IP_ADDRESS ADDRESS_MASK AUTH_TYPE [AUTH_ARGUMENT]
```

Damit nun Änderungen vorgenommen werden können, ist der root-Account notwendig.

```
% su
#
```

An dieser Stelle benötigt man kein Passwort mehr um auf den PostgreSQL-User zu wechseln.

```
# su postgres
```

Die Datei `pg_hba.conf` findet sich im Verzeichnis `/usr/pkg/postgresql/data`. Da in diesem speziellen Fall die md5-Authentifikation verwendet wird, muss folgender Eintrag in `pg_hba.conf` ergänzt werden:

```
host    all    194.95.108.11    255.255.255.255    md5
```

Dabei steht 194.95.108.11 für die IP-Adresse des Computers auf dem PostgreSQL installiert ist.

Damit auch lokale Zugriffe akzeptiert werden können wird zusätzlich

```
local   all                                     trust
host    all    127.0.0.1          255.255.255.255    trust
```

in `pg_hba.conf` eingetragen.

Da die Änderungen zu diesem Zeitpunkt noch nicht wirksam sind, ist ein Neustart des Postmaster Dienstes notwendig.

```
% /etc/rc.d/pgsql restart
```

A.3.3 Arbeiten mit der Datenbank

Um mit der Datenbank arbeiten zu können, muss natürlich erst im Datenbank Management System eine Datenbank erzeugt werden. Dazu wird auch ein Benutzer benötigt, der Rechte auf die selbe besitzt. Da anfangs nur der PostgreSQL-User `pgsql` existiert, muss einem bestehenden Unix-User PostgreSQL-Rechte zugewiesen werden.

Dazu wechselt man in den PostgreSQL-User:

```
% su pgsql
```

Mit der Anweisung

```
% createuser --pwprompt --encrypted username
```

wird ein neuer Benutzer angelegt. Dabei verlangt das Script `createuser` nach einem Passwort und möchte im Anschluß wissen, ob dieser neue Benutzer Datenbanken und neue Anwender erstellen darf.

Mit dem Befehl

```
% dropuser username
```

wird der Benutzer `username` aus der Verwaltungsdatenbank entfernt.

Eingeloggt als der gerade erzeugte PostgreSQL-User, kann nun eine Datenbank im Management System angelegt werden. Dies geschieht mit der Anweisung

```
% createdb dbname
```

oder als nicht-PostgreSQL-User von einem entfernten Rechner

```
% createdb --host 194.95.108.11  
--username username --password dbname
```

Wobei 194.95.108.11 wieder die IP-Adresse der Maschine mit dem aktiven PostgreSQL Dienst ist. Als *username* wird der mit *createuser* erzeugte Benutzername verwendet, und *dbname* ist der Name der anzulegenden Datenbank.

Gelöscht wird eine bestehende Datenbank mit dem Befehl

```
% dropdb dbname
```

oder

```
% dropdb --host 194.95.108.11  
--username username --password dbname
```

Für das Projekt "Virtuelles Unix Labor" verwenden wir den Benutzer *vulab* und die dazugehörige PostgreSQL Datenbank *vulab*.

```
% createuser vulab  
% createdb --host 194.95.108.11  
--username vulab --password vulab
```


Anhang B

Die Datenbankstruktur

Die Abspeicherung der Daten erfolgt nach einem festen Schema. Die Grundlage des Datenmodells sind einfache Tabellen. Sie bilden eine hierarchiefreie Zusammenstellung von Datenfeldern.

Um die Daten besser verwalten zu können ist eine redundanzfreie Speicherung notwendig. D.h., keine Daten dürfen doppelt vorkommen. Dazu wurde das in Abbildung B.1 gezeigte Design entworfen. Nachfolgendes SQL-Script erstellt die gewünschten Tabellen in der in Anhang A.3.3 erzeugten Datenbank vuLab.

Listing B.1: SQL-Script zur Erzeugung der Datenbankstruktur

```
2 -----  
3 -- SQL-Script fuer PostgreSQL  
4 -- zum Aufbau der Datenbankstruktur  
5 -- zur Verwaltung der Daten des  
6 -- Virtuellen Unix Labors  
7 -----  
8  
9  
10 DROP TABLE ergebnis_checks;  
11 DROP TABLE buchungen;  
12 DROP SEQUENCE buchungen_Buchungs_id_seq;  
13 DROP TABLE uebungs_checks;  
14 DROP SEQUENCE uebungs_checks_check_id_seq;  
15 DROP TABLE uebung_setup;  
16 DROP TABLE uebungen;  
17 DROP TABLE benutzer;  
18 DROP SEQUENCE benutzer_user_id_seq;  
19 DROP TABLE rechner;  
20 DROP TABLE images;
```



```

22 CREATE TABLE rechner (
    bezeichnung varchar(30) NOT NULL,
24     PRIMARY KEY (bezeichnung),
    UNIQUE (bezeichnung)
26 );

28 CREATE TABLE images (
    bezeichnung varchar(150) NOT NULL,
30     PRIMARY KEY (bezeichnung),
    UNIQUE (bezeichnung)
32 );

34 CREATE TABLE uebungen (
    uebung_id varchar(40) NOT NULL,
36     bezeichnung varchar(150) NOT NULL,
    vorlauf time NOT NULL,
38     dauer time NOT NULL,
    nachlauf time NOT NULL,
40     wiederholbar boolean NOT NULL,
    text varchar(150) NOT NULL,
42     mehr_info varchar(150),
    PRIMARY KEY (uebung_id),
44     UNIQUE (uebung_id)
);

46 CREATE TABLE uebung_setup (
48     uebung_id varchar(40) NOT NULL,
    rechner varchar(30) NOT NULL,
50     image varchar(150) NOT NULL,
    CONSTRAINT pk_uebung_setup
52         PRIMARY KEY (uebung_id, rechner),
    CONSTRAINT fk_uebung
54         FOREIGN KEY (uebung_id)
        REFERENCES uebungen (uebung_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
56     CONSTRAINT fk_rechner
        FOREIGN KEY (rechner)
60         REFERENCES rechner (bezeichnung)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
62     CONSTRAINT fk_image
        FOREIGN KEY (image)
64         REFERENCES images (bezeichnung)
        ON DELETE CASCADE
        ON UPDATE CASCADE
66 );

70 CREATE TABLE uebungs_checks (
    check_id serial NOT NULL,
72     uebung_id varchar(80) NOT NULL,
    script varchar(150) NOT NULL,
74     bezeichnung varchar(150) NOT NULL,
    interpreter varchar(30) NOT NULL,
76     rechner varchar(30) NOT NULL,
    PRIMARY KEY (check_id),
78     UNIQUE (check_id),

```

```
CONSTRAINT fk_uebung
    FOREIGN KEY (uebung_id)
    REFERENCES uebungen (uebung_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
CONSTRAINT fk_rechner
    FOREIGN KEY (rechner)
    REFERENCES rechner (bezeichnung)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE buchungen (
    buchungs_id serial NOT NULL,
    user_id int NOT NULL,
    uebung_id varchar(40) NOT NULL,
    datum date NOT NULL,
    startzeit time NOT NULL,
    freigegeben varchar(30) DEFAULT 'nein' NOT NULL,
    endzeit time,
    at_id int,
    ip varchar(20),
    PRIMARY KEY (buchungs_id),
    UNIQUE (buchungs_id),
    CONSTRAINT fk_uebung
        FOREIGN KEY (uebung_id)
        REFERENCES uebungen (uebung_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    CONSTRAINT fk_user_id
        FOREIGN KEY (user_id)
        REFERENCES benutzer (user_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE ergebnis_checks (
    buchungs_id int NOT NULL,
    check_id int NOT NULL,
    erfolg boolean NOT NULL,
    CONSTRAINT pk_ergebnis_checks
        PRIMARY KEY (buchungs_id,
                     check_id),
    CONSTRAINT fk_check_id
        FOREIGN KEY (check_id)
        REFERENCES uebungs_checks (check_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    CONSTRAINT fk_buchungs_id
        FOREIGN KEY (buchungs_id)
        REFERENCES buchungen (buchungs_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);
```

Das unter `vulab.sql` gespeicherte Script kann mit dem Befehl `psql` und dem Parameter `-f dateiname` direkt ausgeführt werden.

```
% psql -f dateiname -d datenbank
```

Dabei ist `dateiname` der Dateiname des SQL-Script und `datenbank` die Datenbank im Management System PostgreSQL.

Um von einem externen Rechner aus das SQL-Script auszuführen, muß die Anweisung erweitert werden.

```
% psql -h host -p port -U user -f dateiname -d datenbank
```

Der Parameter `-h` gibt dabei den `host` an, auf dem PostgreSQL läuft. Der `port` des Datenbank Management Systems wird mit Hilfe `-p` spezifiziert und `-U` legt den `user` fest. Das SQL-Script wird mit `-f dateiname` angegeben und `-d` bestimmt die `datenbank`.

Für das Projekt "Virtuelles Unix Labor" wird unter dem Benutzer `vulab` und der dazugehörige PostgreSQL Datenbank `vulab` das SQL-Script `vulab.sql` verwendet.

```
% psql -h 194.95.108.11 -p 5432 -U vulab  
-f vulab.sql -d vulab
```

Die Bedeutung der einzelnen Feldnamen der Tabellen werden nun nachfolgend explizit aufgeführt.

Tabelle: benutzer	
user_id	Eine laufende Nummer
login	Die Benutzerkennung. Bei der Initialisierung wird hier die eMail-Adresse eingetragen.
matrikel_nr	Die Matrikelnummer
vorname	Der Vorname
nachname	Der Nachname
email	Die eMail-Adresse
passwort	Das DES-verschlüsselte Passwort
anmeldedatum	Das Datum der Erstanmeldung
typ	Ein Feld zum Überprüfen, welche Rechte der Benutzer hat
freischalt_secret	Eine Zeichenkombination mit der sich der Benutzer identifizieren muss. Sie wird für den ersten Login benötigt.

Tabelle B.1: Benutzer

Tabelle: rechner	
bezeichnung	Der Name des Computers

Tabelle B.2: Rechner

Tabelle: images	
bezeichnung	Der Name des Images

Tabelle B.3: Images

Tabelle: buchungen	
buchungs_id	Eine laufende Nummer
user_id	Die Nummer des jeweiligen Benutzers, der eine Buchung vorgenommen hat.
uebung_id	Die Nummer der Übung, die gebucht wurde.
datum	Das Datum an dem die Übung durchgeführt wird.
startzeit	Die Uhrzeit markiert den Beginn der Übung.
freigegeben	Ein Flag, das anzeigt, ob die gewünschte Übung bereitgestellt wurde. (Dazu gehört unter anderem, dass aufsetzen des Image.
endzeit	Die Uhrzeit, wann die Übung beendet wurde.
at_id	Die Nummer des at-Jobs der gestartet wurde.
ip	Die IP-Adresse, von wo aus die Übung bearbeitet wird.

Tabelle B.4: Buchungen

Tabelle: uebungen	
uebung_id	Die Kurzbezeichnung der Übung
bezeichnung	Die Bezeichnung der Übung
dauer	Die maximale Dauer der Übung ohne Vor- und Nachlauf (in Form von Stunde:Minute).
vorlauf	Die Zeit, die benötigt wird, um die Übung vorzubereiten (in Form von Stunde:Minute).
nachlauf	Die Zeit, die benötigt wird, um das Ergebnis auszuwerten (in Form von Stunde:Minute).
wiederholbar	Ein Flag, das anzeigt, ob der Benutzer die Übung wiederholen darf.
text	Der Pfad, der auf die Übungsbeschreibung verweist (bspw. nfs.html).
mehr_info	Der Pfad, der auf weitere Informationen verweist (bspw. nfs_info.html).

Tabelle B.5: Übungen

Tabelle: uebungs_checks	
check_id	Eine laufende Nummer
uebung_id	Die Nummer der Übung, auf die der Check ausgeführt wird.
bezeichnung	Der Titel des Checks
script	Der Pfad, der auf die Scripdatei verweist, die den Check durchführt (bspw. nfs-check).
interpreter	Der für das Check-Script verwendete Interpreter (bspw. /bin/perl)
rechner	Der Name des Computers, auf dem der Check ausgeführt wird.

Tabelle B.6: Übungs_Checks

Tabelle: ergebnis_checks	
buchungs_id	Die Nummer der zugehörigen Buchung
check_id	Die Nummer des Checks, der durchgeführt wurde.
erfolg	Ein Flag das anzeigt, ob der Check das gewünschte Ergebnis bestätigt hat.

Tabelle B.7: Ergebnis_Checks

Tabelle: uebungs_setup	
uebung_id	Die Kurzbezeichnung der jeweiligen Übung
rechner	Der Name des jeweiligen Computers.
image	Das für die Übung benötigte Image.

Tabelle B.8: Übungs_Setup

Anhang C

Datensicherung

Es ist hilfreich, in regelmäßigen Abständen eine Sicherung der Informationen aus der verwendeten Datenbank `vu1ab` durchzuführen. Im Fehlerfall kann somit eine Restaurierung aller Daten erfolgen.

Eine Hilfestellung dazu gibt das `Makefile` – siehe Listing C.2. Es befindet sich mit allen dazu gehörigen Dateien im Ordner `backup` im Homeverzeichnis. Der Aufruf `make [Parameter]` startet den Vorgang.

```
% make
Es wurde kein Parameter angegeben.
make param

param:
    createDB                erzeugt Datenbank
    removeDB                loescht DB
    createTables            legt Tabellen an
    removeTables            loescht Tabellen
    backup ['BACKUP=filename.sql'] erzeugt sql-Script
    backup-tar              erzeugt sql-Script tar-komprimiert
    backup-tgz              erzeugt sql-Script tgz-komprimiert
    restore ['BACKUP=filename.sql'] restauriert DB aus sql-Datei
    restore-tar             restauriert DB aus tar-Datei
    restore-tgz             restauriert DB aus tgz-Datei
```

Zum Löschen aller Tabellen (`make removeTables`) startet das SQL-Script `dropdb.sql`, welches weiter unten als Listing C.1 dargestellt ist.

Alle Tabellen des 'virtuellen Unix Labors' lassen sich mit dem Aufruf

```
% make createTables
```

herstellen. Dabei wird das Script [B.2](#) – `vu1ab.sql` ausgeführt, welches im Anhang [B](#) – Datenbankstruktur beschrieben ist.

Alle für `make` verfügbaren Parameter sind eigentlich selbsterklärend. Die wahrscheinlich häufigste Methode zum Sichern der Daten wird

```
% make backup-tgz
```

mit seinem Gegenstück

```
% make restore-tgz
```

sein. Es erzeugt ein Backup der Datenbank `vu1ab` und komprimiert es als 'tgz'.

Listing C.1: SQL-Script zum Löschen der Tabellen (`dropdb.sql`)

```
2  -----
   -- SQL-Script fuer PostgreSQL
4  -- zum Loeschen der Datenbankstruktur
   -- des Virtuellen UNIX Labors
6  -----

8  DROP TABLE ergebnis_checks;
   DROP TABLE buchungen;
10 DROP SEQUENCE buchungen_Buchungs_id_seq;
   DROP TABLE uebungs_checks;
12 DROP SEQUENCE uebungs_checks_check_id_seq;
   DROP TABLE uebung_setup;
14 DROP TABLE uebungen;
   DROP TABLE benutzer;
16 DROP SEQUENCE benutzer_user_id_seq;
   DROP TABLE rechner;
18 DROP TABLE images;
```

Listing C.2: Makefile zur Sicherung der Daten

```

2 # Makefile to create, backup and restore the postgresQL database vulab

4 PGSQL      = psql                # postgresQL client
DB           = vulab              # Datenbank
6 REFERENCE  = template1          # Referenz-Datenbank
INIT         = vulab.sql          # sql-Script zum Anlegen der Tabellen
8 DROP       = dropdb.sql         # sql-Script zum Loeschen der Tabellen
HOST         = 194.95.108.11      # Host mit dem DBMS
10 PORT      = 5432               # Port des DBMS
USER         = vulab              # User des DBMS
12 BACKUP     = backup.sql         # Backup-SQL-Datei
BACKUP_TAR   = backup.sql.tar     # Backup-SQL-Datei tar-komprimiert
14 BACKUP_GZIP = backup.sql.tgz   # Backup-SQL-Datei tar.gz-komprimiert

16 default:
    @echo "Es wurde kein Parameter angegeben."
18    @echo "make param"
    @echo "param:"
20    @echo " createdB                erzeugt Datenbank"
    @echo " removeDB                loescht DB"
22    @echo " createTables            legt Tabellen an"
    @echo " removeTables            loescht Tabellen"
24    @echo " backup ['BACKUP=filename.sql'] erzeugt sql-Script"
    @echo " backup-tar                erzeugt sql-Script tar-komprimiert"
26    @echo " backup-tgz                erzeugt sql-Script tgz-komprimiert"
    @echo " restore ['BACKUP=filename.sql'] restauriert DB aus sql-Datei"
28    @echo " restore-tar                restauriert DB aus tar-Datei"
    @echo " restore-tgz                restauriert DB aus tgz-Datei"
30
32 removedB:
    dropdb $(DB)

34 createdB:
    createdb --host ${HOST} --username=vulab --password $(DB)
36
38 createTables:
    $(PGSQL) -h $(HOST) -p $(PORT) -U $(USER) -f $(INIT) $(DB)

40 removeTables:
    $(PGSQL) -h $(HOST) -p $(PORT) -U $(USER) -f $(DROP) $(DB)
42
44 backup:
    pg_dump -v -C -d -f $(BACKUP) $(DB)

46 backup-tar:
    pg_dump -v -C -d -F t -f $(BACKUP_TAR) $(DB)
48
48 backup-tgz:
    pg_dump -v -C -d -F c -f $(BACKUP_GZIP) $(DB)
50
52 restore:
    $(PGSQL) -h $(HOST) -p $(PORT) -U $(USER) -f $(BACKUP) $(DB)
54
56 restore-tar:
    pg_restore -v -C -d $(REFERENCE) $(BACKUP_TAR)

```

```
58 restore-tgz:  
    pg_restore -v -C -d $(REFERENCE) $(BACKUP_GZIP)
```

Anhang D

Funktionsverzeichnis

Der Anhang enthält eine alphabetische Kurzbeschreibung der Funktionen, welche sich in der Datei `functions.php` befinden.

Um diese Funktionen zu verwenden, muss im jeweiligen PHP-Script der Eintrag

```
require_once 'functions.php';
```

vorgenommen werden.

Beschreibung der Funktionen aus `functions.php`

- **checkPass**

```
checkPass($plain, $crypt)
```

Vergleicht ein verschlüsseltes Passwort mit einem Passwort in Klartext.

Parameter:

`$plain`: Das Passwort in Klartext.

`$crypt`: Das verschlüsselte Passwort.

Rückgabe:

Ein boolescher Wert: `TRUE` oder `FALSE`

Beispiel:

```
$success = checkPass("dozent", "vMvWUS2ZysWKc");  
echo $success; ⇒ TRUE
```

- **createPass**

```
createPass($plain)
```

Erzeugt ein verschlüsseltes Passwort aus dem Klartext `$plain`.

Parameter:

`$plain`: Das Passwort in klartext.

Rückgabe:

Das verschlüsselte Passwort als `String`.

Beispiel:

```
$pass = createPass("dozent");  
echo $pass;  $\Rightarrow$  vMvWUS2ZysWK
```

- **cutTime**

```
cutTime($time)
```

Schneidet die Sekunden der Variable `$time` ab.

Parameter:

`$time`: Die Uhrzeit als `String`.

Rückgabe:

Die Uhrzeit als `String`.

Beispiel:

```
$time = cutTime("12:45:00");  
echo $pass;  $\Rightarrow$  12:45
```

- **executeStatement**

```
executeStatement($connection, $sql, $error)
```

Wendet eine SQL-Anweisung auf eine bestehende PostgreSQL-Datenbank-Verbindung an.

Parameter:

`$connection`: Ein PHP-Objekt, welches über `getConnection()` erzeugt wurde.

`$sql`: Die SQL-Anweisung als String.

`$error`: Die im Fehlerfall auszugebende Meldung als String.

Rückgabe:

Das Ergebnis in Form eines PHP-Objekts.

Beispiel:

```
$connection = getConnection();  
$sql = "select * from userdata";  
$error = "Es ist ein Fehler aufgetreten."  
$result =  
    executeStatement($connection, $sql, $error);
```

- **fetchArray**

```
fetchArray($result, $pos)
```

Extrahiert einen Datensatz aus dem mit `executeStatement()` erhaltenen `$result`.

Parameter:

`$result`: Ein PHP-Objekt, welches der Aufruf `executeStatement()` erzeugt hat.
`$pos`: Die Nummer des gewünschten Datensatz als `int`.

Rückgabe:

Den Datensatz in Form eines `arrays`.

Beispiel:

```
$connection = getConnection();  
$sql = "select * from userdata";  
$error = "Es ist ein Fehler aufgetreten."  
$result =  
    executeStatement($connection, $sql, $error);  
$array =  
    fetchArray($result, $pos);
```

- **getConnection**

```
getConnection()
```

Baut eine Verbindung zur PostgreSQL-Datenbank auf. Dazu werden die Daten aus der `config.php`-Datei im Wurzelverzeichnis `public_html` verwendet.

Rückgabe:

Der Datenbank-Zugang in Form eines PHP-Objekts.

Beispiel:

```
$connection = getConnection();
```


- **getContentLink**

```
getContentLink($content, $userType)
```

Erzeugt den Pfad auf den Einsprungpunkt des jeweiligen Menüs.

Parameter:

<code>\$content:</code>	Der Modul- oder Dateiname, welche geladen werden soll (String).
<code>\$userType:</code>	Der Type als String. <ul style="list-style-type: none">◇ default◇ user◇ dozent◇ admin

Rückgabe:

Den Pfad auf die gewünschte Datei als String.

Beispiel:

```
$path = getContentLink("home", "user");  
⇒ user/home/index.php
```

- **getLocation**

```
getLocation($link)
```

Liefert die URL (ohne Host) der Datei `$link`.
Die Funktion wird verwendet, um Dateien zu adressieren,
welche in Unterverzeichnissen liegen.

Parameter:

`$link`: Der relativer Verweis auf die Datei als `String`.

Rückgabe:

Die URL auf die gewünschte Datei als `String`.

Beispiel:

Aufruf aus Datei `http://smaug.fh-regensburg.de/vulab/index.php`

```
$path = getLocation("default/index.php");  
echo $path;  
⇒ vulab/default/index.php
```

- **getSubdirs**

```
getSubdirs($dir)
```

Liefert ein `array` mit allen Unterverzeichnissen von `$dir`.

Parameter:

`$dir`: Das Start-Verzeichnis als `String`.

Rückgabe:

Ein `array`, welches alle Unterverzeichnisse enthält.

Beispiel:

```
$array = getSubdirs("user");  
print_r ($array);  
⇒ Array ( (0) => home (1) => userdata )
```

- **getValue**

```
getValue($method, $key)
```

Extrahiert Werte aus den Hash-Tabellen GET und POST.

Parameter:

\$method: Der Name des Hash als String.

◇ GET

◇ POST

\$key: Der gewünschte Schlüssel als String.

Rückgabe:

Der Inhalt als String.

Beispiel:

Aufruf aus Datei *.../index.php?name=Hugo*

```
$value = getValue("GET", "name");
```

```
echo ($value); ⇒ Hugo
```

- **handleErrors**

```
handleErrors($error, $message, $filename, $line)
```

Übernimmt die Fehlerbehandlung.

Es wird eine log-Datei angelegt, in der alle aufgetretenen Fehler mitprotokolliert werden.

Parameter:

\$error: Der Name des Hash als String.

\$message: Der gewünschte Schlüssel als String.

\$filename: Der gewünschte Schlüssel als String.

\$line: Der gewünschte Schlüssel als String.

Beispiel:

Aufruf aus Datei *index.php*:

```
set_error_handler('handleErrors');
```

```
$scriptName = 'index.php';
```

- **home**

```
home( $userType )
```

Erzeugt eine Beschreibung der Menüpunkte.

Parameter:

`$method`: Der Typ als `String`.

Rückgabe:

Ein `String` der HTML-Code enthält.

Er spiegelt die Beschreibung der Menüpunkte wieder.

Beispiel:

```
home( "user" );  $\Rightarrow$  siehe Abbildung 3.4 auf Seite 28
```

- **makeSecret**

```
makeSecret( )
```

Erzeugt eine eindeutige Seriennummer.

Rückgabe:

Die Seriennummer als `String`.

Beispiel:

```
$secret = makeSecret( );  $\Rightarrow$  vMvWUS2ZysWK
```

- **makeSeed**

```
makeSeed( )
```

Generiert einen Startwert zur Erzeugung von Passwörtern basierend auf der aktuellen Uhrzeit.

Rückgabe:

Der Wert als `float`.

Beispiel:

```
$secret = makeSeed( );  $\Rightarrow$  1050138774.6
```

- **menu**

```
menu( $userType )
```

Erzeugt das Menü.

Parameter:

\$method: Der Typ als String.

Rückgabe:

Ein String der HTML-Code enthält.
Er spiegelt das Menü wieder.

Beispiel:

```
menu( "user" );
```

⇒ siehe Abbildung [3.4](#) auf Seite [28](#)

- **showGetVar**

```
showGetVar( )
```

Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle GET aus.

Beispiel:

```
showGetVar( );
```

```
⇒ KEY      VALUE
   ...      ...
```

- **showPostVar**

```
showPostVar( )
```

Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle POST aus.

Beispiel:

```
showPostVar( );
```

```
⇒ KEY      VALUE
   ...      ...
```

- **showServerVar**

`showServerVar()`

Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle SERVER aus.

Beispiel:

```
showServerVar();  
⇒ KEY          VALUE  
   DOCUMENT_ROOT /usr/pkg/share/httpd/htdocs  
   ...           ...
```

- **showSessionVar**

`showSessionVar()`

Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle SESSION aus.

Beispiel:

```
showSessionVar();  
⇒ KEY      VALUE  
   ...     ...
```

Anhang E

Programmcode

E.1 Basis-Scripte zum Seitenaufbau

Listing E.1: config.php

```
<?php
2
   /*
4   * config.php
   */
6
   //////////////////////////////////////
8   // Error Handling

10  $log_errors = TRUE;
   $log_path = "log/";
12
   //////////////////////////////////////
14  // eMail-Konfiguration

16  $server = "sonne.zimmermann.de";
   $adminName = "Administrator";
18  $adminMail = "zis30131@sonne.zimmermann.de";
   $replyName = "Stefan Zimmermann";
20  $replyMail = "zis30131@sonne.zimmermann.de";
   $Cc = "";
22  $Bcc = "";

24  //////////////////////////////////////
   // Datenbank-Konfiguration

26
   $pg_host = "sonne.zimmermann.de";
28  $pg_port = 5432;
   $pg_db = "vulab";
30  $pg_login = "vulab";
   $pg_pass = "vulab";
32
   //////////////////////////////////////
34  // Variablen zum blaettern in den Datensatzen
```

```
36 // Wieviele Datensätze anzeigen ?
    $number = 10;
38
    // Wieviele Seiten-Links anzeigen ?
40    $linkNumber = 6;
    ?>
```

Listing E.2: index.php

```
<?php
2
    /*
4     * index.php
    */
6
    $debug = false;
8
    // extrahiert den Pfad
10    // /home/zis30131/public_html/phpinfo.php
    // --> /home/zis30131/public_html/
12    function getPath($file)
    {
14        $pos=(int) strrpos($file, '/');
        $s=substr($file, 0, $pos);
16        $string=$s . "/";

18        return $string;
    }
20

    $projectPath = getPath($_SERVER_VARS['PHP_SELF']);
22    $absoluteProjectPath = getPath($_SERVER_VARS['PATH_TRANSLATED']);

24    require_once $absoluteProjectPath . 'functions.php';
    require_once $absoluteProjectPath . 'config.php';
26

    // Fehlerbehandlung
28    ob_start('ob_gzhandler');
    set_error_handler('handleErrors');
30    $scriptName = "index.php";

32    // hole Session-Variable
    if (isset($_GET["PHPSESSID"]))
34        $PHPSESSID = $_GET["PHPSESSID"];
    elseif (isset($_POST["PHPSESSID"]))
36        $PHPSESSID = $_POST["PHPSESSID"];

38    // falls Session vorhanden, schliesse sie
    if (isset($PHPSESSID))
40    {
        session_start();
42
        $_SESSION = array();
44        session_unset();
```



```

        session_destroy();
46
        unset($_SESSION);
48    }

    // hole Informationen ueber Menu
    $menu = getValue("GET", "menu");
52

    if (empty($menu))
54        $menu = getValue("POST", "menu");

    if (empty($menu))
56        $menu = "default";
58

    // hole Informationen ueber Link
    $content = getValue("GET", "content");
60

    if (empty($content))
62        $content = getValue("POST", "content");
64

    // ohne Login darf nur das default-Menue aufgerufen werden
    if ($menu != "default")
66        $menu = "default";
68

?>
70
<html>
72 <head>
    <title>Virtuelles Unix Labor</title>
74    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    <link rel="stylesheet" href="vulab.css" type="text/css">
76 </head>

78 <body bgcolor="#E5E5E5" text="#000000"
    leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">
80    <table width="700" cellpadding="0" cellspacing="0"
        align="center" height="100%" border="2" bgcolor="white">
82        <tr><td valign="top">
            <table width="700" cellpadding="0" cellspacing="0"
84                align="center" height="100%" bgcolor="#FFFFFF" border="0">
                <tr>
86                    <td height="110"></td>
                </tr>
88                <tr height="30">
                    <td align="center" valign="middle" height="30">
90                        <table border="0" cellpadding="0" cellspacing="0"
                            width="100%" height="100%">
92                            <tr><td width="20">&nbsp;</td><td valign="top" align="center">

                                <!-- Hier das Menue einsetzen -->
94
                                <?php
96
                                    if (!empty($menu))
                                        echo menu($menu);
98
                                    else
                                        echo menu("default");
100
                                ?>

```

```

104         <hr>
        </td><td width="20">&nbsp;</td></tr>
106     </table>
    </td>
108 </tr>
    <tr>
110         <td align="center" valign="top">
            <table border="0" cellpadding="0" cellspacing="0"
112                 width="100%" height="100%">
                <tr><td width="20">&nbsp;</td><td valign="top" align="center">
114
                    <!-- Hier den Inhalt einsetzen -->
116
                    <?php
118
                        if ($debug)
120                        {
                            showPostVar();
122                            showGetVar();
                        }
124
                        if (!empty($content))
126                        {
                            $contentLink = getContentLink($content, "default");
128                            if(!empty($contentLink))
                                include $contentLink;
                            else
                                include "notFound.php";
132                        }
                        else
134                        {
                            $contentLink = getContentLink("home", "default");
136                            if(!empty($contentLink))
                                include $contentLink;
                            else
138                                include "notFound.php";
                        }
140                    ?>
142
                </td><td width="20">&nbsp;</td></tr>
            </table>
        </td>
146    </tr>
    <tr>
148        <td align="center">
            <table border="0" cellpadding="0" cellspacing="0"
150                 width="100%" height="100%">
                <tr>
152                    <td width="20">&nbsp;</td>
                    <td valign="bottom" align="right">
154                        <p class="text">
<?php
156                echo "<br>\n";
                echo $adminName . ": <a href='mailto:" . $adminMail .
158                    "?subject=Virtuelles Unix Labor'" . $adminMail . "</a>\n";
                ?>
160            <p>

```

```

162         </td><td width="20">&nbsp;</td>
163     </tr>
164 </table>
165 </td>
166 </tr>
167 </table>
168 </td></tr>
169 </table>
170 </body>
171 </html>

```

Listing E.3: index_session.php

```

1 <?php
2
3     /*
4      * index_session.php
5      */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ---
12    // ???
13    //
14    // POST
15    // ----
16    // ???
17    //
18    // SESSION
19    // -----
20    // nach dem ersten Aufruf:
21    // $HTTP_SESSION_VARS['projectPath']
22    // $HTTP_SESSION_VARS['absoluteProjectPath']
23    // ???
24
25    $debug = false;
26
27    // extrahiert den Pfad
28    // /home/zis30131/public_html/phpinfo.php
29    // --> /home/zis30131/public_html/
30    function getPath($file)
31    {
32        $pos=(int) strrpos($file, '/');
33        $s=substr($file, 0, $pos);
34        $string=$s . "/";
35
36        return $string;
37    }
38
39    // hole Session-Variable
40    if (isset($_GET["PHPSESSID"]))
41        $PHPSESSID = $_GET["PHPSESSID"];

```

```

42     elseif (isset($_POST["PHPSESSID"]))
43         $PHPSESSID = $_POST["PHPSESSID"];
44
45     // hole vorhandene Session
46     // falls keine Session verfuegbar, erzeuge neue Session
47     session_start();
48     $PHPSESSID=session_id();
49
50     // Variablen in der Session registrieren
51     if (!session_is_registered("projectPath"))
52         session_register("projectPath");
53
54     if (!session_is_registered("absoluteProjectPath"))
55         session_register("absoluteProjectPath");
56
57     $projectPath = getPath($_SERVER_VARS['PHP_SELF']);
58     $_SESSION_VARS['projectPath'] = $projectPath;
59
60     $absoluteProjectPath = getPath($_SERVER_VARS['PATH_TRANSLATED']);
61     $_SESSION_VARS['absoluteProjectPath'] = $absoluteProjectPath;
62
63     require_once $absoluteProjectPath . 'functions.php';
64     require_once $absoluteProjectPath . 'config.php';
65
66     // Fehlerbehandlung
67     ob_start('ob_gzhandler');
68     set_error_handler('handleErrors');
69     $scriptName = "index_session.php";
70
71     if ($debug)
72     {
73         echo $projectPath . "<br>";
74         echo $absoluteProjectPath . "<br>";
75     }
76
77     // hole Session-Variablen
78     if (isset($_SESSION_VARS["login"]))
79         $login = $_SESSION_VARS["login"];
80
81     if (isset($_SESSION_VARS["auth"]))
82         $auth = $_SESSION_VARS["auth"];
83     else
84         $auth = false;
85
86     if ($debug)
87         if (isset($_SESSION))
88             showSessionVar();
89
90     // hole Informationen ueber Menu
91     $menu = getValue("GET", "menu");
92
93     if (empty($menu))
94         $menu = getValue("POST", "menu");
95
96     if (empty($menu))
97         $menu = "default";
98
99     // hole Informationen ueber Link

```

```
100     $content = getValue("GET", "content");
102
103     if (empty($content))
104         $content = getValue("POST", "content");
105
106     // URL zum Logout
107     $logoutUrl=getLocation("logout.php") . "?PHPSESSID=" . $PHPSESSID;
108
109     // falls keine Informationen ueber Seiten-Inhalt vorhanden, erfolgt ein logout
110     if (empty($menu) || empty($content))
111     {
112         if ($debug)
113             echo $logoutUrl;
114         else
115             header ("Location: " . $logoutUrl );
116
117         exit;
118     }
119
120     // falls der user nicht authentifiziert ist, erfolgt ein logout
121     if ($menu != "default" && $auth == FALSE)
122     {
123         if ($debug)
124             echo $logoutUrl;
125         else
126             header ("Location: " . $logoutUrl );
127
128         exit;
129     }
130
131     // falls jemand eingeloggt ist und auf ein nicht-berechtigtes
132     // Menu zugreifen moechte, erfolgt ein logout
133     if (!empty($login))
134     {
135         // baue Verbindung zur Datenbank auf
136         $connection = getConnection();
137
138         // frage nach userType
139         $sql = "select typ from benutzer where login='" . $login . "'";
140
141         $msg = "Beim Zugriff auf die Datenbank ist ein Fehler " .
142             "aufgetreten.<br>Wenden Sie sich an Ihren Administrator.";
143
144         $result = executeStatement($connection, $sql, $msg);
145
146         if ($result)
147         {
148             // holt die erste Zeile des Resultats
149             $array = fetchArray($result, 0);
150
151             if ($debug)
152             {
153                 echo "menu: $menu<br>";
154                 echo "auth: $auth<br>";
155                 echo "array[typ]: " . $array['typ'] . "<br>";
156                 print_r ($array);
157                 echo "<br>";
158             }
159         }
160     }
```

```

158      // ueberpruefe ob angefordertes Menue mit Berechtigung
160      // ueberein stimmt
161      if (!$auth || $array['typ'] != $menu)
162      {
163          if ($debug)
164              echo $logoutUrl;
165          else
166              header ("Location: " . $logoutUrl );
167
168          exit;
169      }
170    }
171  }
172  ?>
173
174  <html>
175  <head>
176      <title>Viruelles Unix Labor</title>
177      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
178      <link rel="stylesheet" href="vulab.css" type="text/css">
179  </head>
180
181  <body bgcolor="#E5E5E5" text="#000000"
182      leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">
183      <table width="700" cellspacing="0" cellpadding="0"
184          align="center" height="100%" border="2" bgcolor="white">
185          <tr><td valign="top">
186              <table width="700" cellspacing="0" cellpadding="0"
187                  align="center" height="100%" bgcolor="#FFFFFF" border="0">
188                  <tr>
189                      <td height="110"></td>
190                  </tr>
191                  <tr height="30">
192                      <td align="center" valign="middle" height="30">
193                          <table border="0" cellpadding="0" cellspacing="0"
194                              width="100%" height="100%">
195                              <tr><td width="20">&nbsp;</td><td align="top" align="center">
196
197                                  <!-- Hier das Menue einsetzen -->
198
199                                  <?php
200                                      if (!empty($menu))
201                                          echo menu($menu);
202                                      else
203                                          echo menu("default");
204                                  ?>
205
206                                  <hr>
207                              </td><td width="20">&nbsp;</td></tr>
208                          </table>
209                      </td>
210                  </tr>
211                  <tr>
212                      <td align="center" valign="top">
213                          <table border="0" cellpadding="0" cellspacing="0"
214                              width="100%" height="100%">
215                              <tr><td width="20">&nbsp;</td><td align="top" align="center">

```

```

216         <!-- Hier den Inhalt einsetzen -->
218
219         <?php
220
221             if ($debug)
222             {
223                 if (isset($PHPSESSID))
224                     echo "PHPSESSID=" . $PHPSESSID . "<br>";
225                 showGetVar();
226                 showPostVar();
227             }
228
229             if (!empty($content))
230             {
231                 $contentLink = getContentLink($content, $menu);
232                 if(!empty($contentLink))
233                     include $contentLink;
234                 else
235                     include "notFound.php";
236             }
237             else
238             {
239                 $contentLink = getContentLink("home", $menu);
240                 if(!empty($contentLink))
241                     include $contentLink;
242                 else
243                     include "notFound.php";
244             }
245         ?>
246
247         </td><td width="20">&nbsp;</td></tr>
248     </table>
249 </td>
250 </tr>
251 <tr>
252     <td align="center">
253         <table border="0" cellpadding="0" cellspacing="0"
254             width="100%" height="100%">
255             <tr>
256                 <td width="20">&nbsp;</td>
257                 <td valign="bottom" align="right">
258                     <p class="text">
259 <?php
260     echo "<br>\n";
261     echo $adminName . ": <a href='mailto:" . $adminMail .
262         "?subject=Virtuelles Unix Labor'" . $adminMail . "</a>\n";
263 ?>
264
265                 <p>
266                     </td><td width="20">&nbsp;</td>
267                 </tr>
268             </table>
269     </td>
270 </tr>
271 </table>
272 </td></tr>
273 </table>
274 </body>

```

```
274 </html>
```

Listing E.4: logout.php

```
<?php
2
3     /*
4     * logout.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['PHPSESSID']
15
16    $debug = false;
17
18    // hole Session-Variable
19    if (isset($HTTP_GET_VARS["PHPSESSID"]))
20        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
21    elseif (isset($HTTP_POST_VARS["PHPSESSID"]))
22        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
23
24    if (!empty($PHPSESSID))
25    {
26        session_start();
27
28        if (isset($HTTP_SESSION_VARS["projectPath"]))
29            $projectPath = $HTTP_SESSION_VARS["projectPath"];
30
31        if (isset($HTTP_SESSION_VARS["projectPath"]))
32            $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
33
34        require_once $absoluteProjectPath . "functions.php";
35
36        // nach session_unset() verschwinden evtl. auch lokale Variablen
37        $ProjectPath = $projectPath;
38
39        if ($debug)
40            showSessionVar();
41
42        // Fehlerbehandlung
43        ob_start('ob_gzhandler');
44        set_error_handler('handleErrors');
45        $scriptName = "logout.php";
46
47        // zerstoere Session
48        $HTTP_SESSION_VARS = array();
49        session_unset();
50        session_destroy();
```



```
    }  
52     if (!$debug)  
        header ("Location: " . $ProjectPath . "index_session.php?menu=default&content=login");  
54  
?>
```

Listing E.5: noConfig.php

```
<?php  
2  
    /*  
4     * noConfig.php  
    */  
6  
?>  
8  
<div align="center">  
10     <p class="head">  
        <br>Es ist ein Fehler aufgetreten  
12     </p>  
</div>  
14 <div align="center">  
    <p class="error">  
16     Die benoetigte Konfigurationsdatei (config.php) ist nicht verfuegbar.  
    </p>  
18 </div>
```

Listing E.6: notFound.php

```
<?php  
2  
    /*  
4     * notFound.php  
    */  
6  
?>  
8  
<div align="center">  
10     <p class="head">  
        <br>Es ist ein Fehler aufgetreten  
12     </p>  
</div>  
14 <div align="center">  
    <p class="error">  
16     Die angeforderte Seite ist leider nicht verfuegbar.  
    </p>  
18 </div>
```

Listing E.7: functions.php

```

2  <?php
3
4  /*
5   * functions.php
6   */
7
8  // Verfuegbare Variablen
9  // =====
10 //
11 // $projectPath
12 // $absoluteProjectPath
13
14 include $absoluteProjectPath . 'config.php';
15
16 // Uebernimmt die Fehlerbehandlung.
17 // Es wird eine log-Datei angelegt, in der alle
18 // aufgetretenen Fehler mitprotokolliert werden.
19 function handleErrors ($error, $message, $filename, $line)
20 {
21     global $log_errors, $log_path, $scriptName;
22     global $absoluteProjectPath, $projectPath;
23
24     $file = $absoluteProjectPath . $log_path . "/vulab.log";
25     $dir = $absoluteProjectPath . $log_path;
26
27     $msg = "";
28
29     // schreibe in Log-Datei
30     if ($log_errors == TRUE)
31     {
32         // pruefe ob die Log-datei existiert
33         if (!file_exists($file))
34         {
35             clearstatcache();
36             $handle = @fopen ($file, "a+");
37
38             if ($handle == FALSE)
39             {
40                 $msg = "Die Log-Datei <i>$file</i> kann nicht " .
41                     "angelegt werden.<br>Stellen Sie sicher das " .
42                     "Verzeichnis <i>$dir</i> <b>Schreibrechte " .
43                     "f&uuml;r den Web-User</b> hat.<br>";
44                 echo $msg;
45             }
46             else
47             {
48                 fclose($handle);
49             }
50         }
51
52         // pruefe ob die Log-Datei Schreibrechte hat
53         if (is_writable($file))
54         {
55             if (filesize($file) > 102400)
56             {
57                 rename($file, $file . (string) time());
58                 clearstatcache();

```

```

58     }
60     $handle = fopen ($file, "a+");
62     // schreibe in die Log-Datei
63     if ($handle)
64     {
65         $output = date("r"). ": from $scriptName\n\t" .
66             "$message in line $line of $filename\n";
68         fwrite($handle, $output);
69         fclose($handle);
70     }
71 }
72 else
73 {
74     $msg = "In die Log-Datei <i>$file</i> kann nicht " .
75         "geschrieben werden.<br>Stellen Sie sicher das " .
76         "die Datei <i>$file</i>, <br>sowohl auch das " .
77         "Verzeichnis <b>Schreibrechte f&uuml;r den " .
78         "Web-User</b> hat.";
79     echo $msg;
80 }
81 }
82 else // falls nicht in die Log-Datei geschrieben wird ...
83 {
84     echo "<html><body>";
85     echo "<b>$message</b> in line $line of <i>$filename</i><br>";
86     echo "</body></html>";
87 }
88 }
89
90 // Extrahiert Werte aus den Hash-Tabellen GET und POST
91 function getValue($method, $key)
92 {
93     global $HTTP_GET_VARS, $HTTP_POST_VARS;
94
95     $value = "";
96
97     if (strtolower($method) == "get")
98         if (isset($HTTP_GET_VARS[$key]))
99             $value = $HTTP_GET_VARS[$key];
100
101     if (strtolower($method) == "post")
102         if (isset($HTTP_POST_VARS[$key]))
103             $value = $HTTP_POST_VARS[$key];
104
105     return $value;
106 }
107
108 // Liefert die URL (ohneHost) der Datei $link.
109 // Die Funktion wird verwendet, um Dateien zu adressieren,
110 // welche in Unterverzeichnissen liegen.
111 function getLocation($link)
112 {
113     global $HTTP_SERVER_VARS;

```

```

116     $pos=(int) strpos($HTTP_SERVER_VARS["PHP_SELF"], '/');
117     $s=substr($HTTP_SERVER_VARS["PHP_SELF"], 0, $pos);
118     $string=$s . "/" . $link;

120     return $string;
121 }
122
123 // BauteineVerbindungzurPostgreSQL-Datenbankauf.
124 // Dazu werden die Daten aus der config.php-Datei im
125 // Wurzelverzeichnis public_html verwendet.
126 function getConnection()
127 {
128     global $debug;
129     global $absoluteProjectPath, $projectPath;
130
131     $index_session = $projectPath . "index_session.php";
132     $index = $projectPath . "index.php";
133
134     // falls PostgreSQL-Funktionen nicht verfuegbar sind
135     if (!function_exists("pg_connect"))
136     {
137         $msg = "Die aktuelle PHP-Enging bietet leider keine " .
138             "Schnittstelle zu PostgreSQL.<br>" .
139             "Bitte wenden Sie sich an Ihren Administrator.";
140         if (!$debug)
141             header ("Location: " . $index .
142                 "?menu=default&content=error&msg=" . $msg);
143         else
144             echo "Location: " . $index .
145                 "?menu=default&content=error&msg=" . $msg . "<br>";
146
147         exit;
148     }
149
150     include $absoluteProjectPath . 'config.php';
151
152     $conn_string = "host=$pg_host dbname=$pg_db user=$pg_login password=$pg_pass";
153     $connection = pg_connect ($conn_string);
154
155     if (!$connection)
156     {
157         $msg = "Es konnte keine Verbindung zur Datenbank " .
158             "hergestellt werden.<br>Bitte versuchen Sie " .
159             "es zu einem spaeteren Zeitpunkt nochmal.";
160
161         if (!$debug)
162             header ("Location: " . $index .
163                 "?menu=default&content=error&msg=" . $msg);
164         else
165             echo "Location: " . $index .
166                 "?menu=default&content=error&msg=" . $msg . "<br>";
167
168         exit;
169     }
170
171     return $connection;
172 }

```

```

174 // Wendet eine SQL-Anweisung auf eine bestehende
175 // PostgreSQL-Datenbank-Verbindung an.
176 function executeStatement($connection, $sql, $error)
177 {
178     global $debug, $PHPSESSID;
179     global $HTTP_SESSION_VARS;
180     global $absoluteProjectPath, $projectPath;
181
182     if (isset($HTTP_SESSION_VARS['menu']))
183         $menu = $HTTP_SESSION_VARS['menu'];
184     else
185         $menu = "default";
186
187     $result = pg_exec($connection, $sql);
188
189     if (!$result)
190     {
191         $index_session = $projectPath . "index_session.php";
192         $index = $projectPath . "index.php";
193
194         if (!empty($PHPSESSID))
195             if (!$debug)
196                 header ("Location: " . $index_session .
197                     "?menu=" . $menu . "&content=error&msg=" . $error .
198                     "&PHPSESSID=" . $PHPSESSID);
199             else
200                 echo "Location: " . $index_session .
201                     "?menu=" . $menu . "&content=error&msg=" . $error .
202                     "&PHPSESSID=" . $PHPSESSID . "<br>";
203         else
204             if (!$debug)
205                 header ("Location: " . $index .
206                     "?menu=" . $menu . "&content=error&msg=" . $error);
207             else
208                 echo "Location: " . $index .
209                     "?menu=" . $menu . "&content=error&msg=" . $error . "<br>";
210         exit;
211     }
212
213     return $result;
214 }
215
216 // Extrahiert einen Datensatz aus dem mit executeStatement()
217 // erhaltenen $result.
218 function fetchArray($result, $pos)
219 {
220     global $debug, $PHPSESSID;
221     global $HTTP_SESSION_VARS;
222     global $absoluteProjectPath, $projectPath;
223
224     if (isset($HTTP_SESSION_VARS['menu']))
225         $menu = $HTTP_SESSION_VARS['menu'];
226     else
227         $menu = "default";
228
229     $array = pg_fetch_array($result, $pos);
230
231     if (!$array)

```

```

232 {
233     $index_session = $projectPath . "index_session.php";
234     $index = $projectPath . "index.php";
235
236     $msg = "Fehler beim Zugriff auf den gewünschten Datensatz";
237
238     if (!empty($PHPSESSID))
239         if (!$debug)
240             header ("Location: " . $index_session .
241                 "?menu=" . $menu . "&content=error&msg=" . $msg .
242                 "&PHPSESSID=" . $PHPSESSID);
243         else
244             echo "Location: " . $index_session .
245                 "?menu=" . $menu . "&content=error&msg=" . $msg .
246                 "&PHPSESSID=" . $PHPSESSID . "<br>";
247
248     else
249         if (!$debug)
250             header ("Location: " . $index .
251                 "?menu=" . $menu . "&content=error&msg=" . $msg);
252         else
253             echo "Location: " . $index .
254                 "?menu=" . $menu . "&content=error&msg=" . $msg . "<br>";
255
256     exit;
257 }
258
259 return $array;
260 }
261
262 // Vergleicht ein verschluesselltes Passwort mit einem Passwort in klartext.
263 function checkPass($plain, $crypt)
264 {
265     $success = false;
266     $salt = substr($crypt, 0, 2);
267     $pass = crypt($plain, $salt);
268
269     if ($crypt == $pass)
270         $success = true;
271
272     return $success;
273 }
274
275 // Erzeugt ein verschluesselltes Passwort aus dem Klartext $plain.
276 function createPass($plain)
277 {
278     $salt = substr($plain, 0, 2);
279     $pass = crypt($plain, $salt);
280
281     return $pass;
282 }
283
284 // Generiert einen Startwert zur Erzeugung von Passwoertern
285 // basierend auf der aktuellen Uhrzeit.
286 function makeSeed()
287 {
288     list($usec, $sec) = explode(' ', microtime());
289     return (float) $sec + ((float) $usec * 100000);

```

```
290 }
292 // Erzeugt eine eindeutige Seriennummer.
function makeSecret()
294 {
    $number = "";
296     $salt = "";
    $sn = "";
298
    srand(makeSeed());
300
    for ($i = 0; $i < 10; $i++)
302         $number .= rand(0, 9);
304
    for ($i = 0; $i < 2; $i++)
        $salt .= rand(0, 9);
306
    $sn = crypt($number, $salt);
308
    return $sn;
310 }
312 // Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle SERVER aus.
function showServerVar()
314 {
    echo "<p></p>\n";
316
    echo "<table border=2>\n";
    echo "<tr>\n";
    echo "<th>KEY</th>\n";
320     echo " <th>VALUE</th>\n";
    echo "</tr>\n";
322
    foreach ($_SERVER as $key => $value)
324     {
        echo "<tr>\n";
326         echo "<td>$key</td>\n";
        echo "<td>$value</td>\n";
328         echo "</tr>\n";
    }
330
    echo "</table>\n";
332     echo "<p></p>\n";
}
334
// Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle GET aus.
336 function showGetVar()
{
338     echo "<p></p>\n";
340
    echo "<table border=2>\n";
    echo "<tr>\n";
342     echo "<th>KEY</th>\n";
    echo " <th>VALUE</th>\n";
344     echo "</tr>\n";
346
    foreach ($_GET as $key => $value)
    {
```

```

348         echo "<tr>\n";
349         echo "<td>$key</td>\n";
350         echo "<td>$value</td>\n";
351         echo "</tr>\n";
352     }
353
354     echo "</table>\n";
355     echo "<p></p>\n";
356 }
357
358 // Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle POST aus.
359 function showPostVar()
360 {
361     echo "<p></p>\n";
362
363     echo "<table border=2>\n";
364     echo "<tr>\n";
365     echo "<th>KEY</th>\n";
366     echo " <th>VALUE</th>\n";
367     echo "</tr>\n";
368
369     foreach ($_POST as $key => $value)
370     {
371         echo "<tr>\n";
372         echo "<td>$key</td>\n";
373         echo "<td>$value</td>\n";
374         echo "</tr>\n";
375     }
376
377     echo "</table>\n";
378     echo "<p></p>\n";
379 }
380
381 // Gibt einen HTML-TABLE mit den Inhalten der Hash-Tabelle SESSION aus.
382 function showSessionVar()
383 {
384     echo "<p></p>\n";
385
386     echo "<table border=2>\n";
387     echo "<tr>\n";
388     echo "<th>KEY</th>\n";
389     echo " <th>VALUE</th>\n";
390     echo "</tr>\n";
391
392     foreach ($_SESSION as $key => $value)
393     {
394         echo "<tr>\n";
395         echo "<td>$key</td>\n";
396         echo "<td>$value</td>\n";
397         echo "</tr>\n";
398     }
399
400     echo "</table>\n";
401     echo "<p></p>\n";
402 }
403
404 // Liefert ein array mit allen Unterverzeichnissen von $dir.
405 function getSubdirs($dir)

```



```
406 {
    chdir($dir);
408
    if ($handle = opendir("."))
410    {
        while (false !== ($file = readdir($handle)))
412        {
            if ($file != "." && $file != ".." && is_dir($file))
414            {
                $links[] = $file;
416            }
        }
418
        closedir($handle);
420    }

    chdir("..");
422

    return $links;
424 }
426
// Erzeugt das Menue.
428 function menu($userType)
{
430     global $PHPSESSID, $HTTP_SESSION_VARS;
    global $absoluteProjectPath, $projectPath;
432
    if (file_exists($absoluteProjectPath . $userType . "/config.php"))
434        include $absoluteProjectPath . $userType . "/config.php";
    else
436    {
        include $absoluteProjectPath . "default/config.php";
438        $userType = "default";
    }
440
    $string = "";
442
    $index_session = $projectPath . "index_session.php";
444    $index = $projectPath . "index.php";

446    if (!empty($HTTP_SESSION_VARS["login"]))
    {
448        $string = $string . "<div align=\"right\">\n";
        $string = $string . "<p class=\"text\">";
450        $string = $string . "Sie sind eingeloggt als ";
        $string = $string . "<b>" . $HTTP_SESSION_VARS["login"] . "</b>";
452        $string = $string . "</p>\n</div>\n";
    }
454

    foreach ($items as $item)
456    {
        if (strtolower($userType) == "default" &&
458            strtolower($item) == "home" ||
            strtolower($item) == "info" ||
460            strtolower($item) == "informationen" )

462            $path = $index;
        else
```



```

522 }

524 // Erzeugt eine Beschreibung der Menuepunkte.
function home($userType)
526 {
    global $PHPSESSID, $HTTP_SESSION_VARS;
528     global $absoluteProjectPath, $projectPath;

530     if (file_exists($absoluteProjectPath . $userType . "/config.php"))
        include $absoluteProjectPath . $userType . "/config.php";
532     else
        include $absoluteProjectPath . "default/config.php";

534     $string = "<table border=\"0\" cellpadding=\"2\" cellspacing=\"5\">\n";

536     $index_session = $projectPath . "index_session.php";
538     $index = $projectPath . "index.php";

540     foreach ($items as $item)
        {
542         if ($userType == "default" && $item == "home" || $item == "info")
            $path = $index;
544         else
            $path = $index_session;

546         $string = $string . "<tr>\n";

548         $string = $string . "<td width=\"200\" align=\"center\">\n";
550             $string = $string . "<a href=\"" . $path;
552             $string = $string . "?menu=" . $userType;
554             $string = $string . "&content=" . $item;

556             if (!empty($PHPSESSID))
                $string = $string . "&PHPSESSID=" . $PHPSESSID;

558             $string = $string . "\">";
560             $string = $string . "[" . $item . "]";
562             $string = $string . "</a>";
564             $string = $string . "\n</td>\n";

566             $string = $string . "<td align=\"left\">\n";
568             $string = $string . "<p class=\"text\">\n";
570             if (array_key_exists($item, $description))
                $string = $string . $description[$item];
572             else
                $string = $string . "keine Beschreibung gefunden";
574             $string = $string . "\n</p></td>\n";

576             $string = $string . "</tr>\n";
        }

    $string = $string . "</table>\n";

    return $string;
}

578 // Schneidet die Sekunden der Variable $time ab.
// 12:00:00 --> 12:00

```

```
580 function cutTime($time)
581 {
582     $pos = (int) strrpos($time, ':');
583     $string = substr($time, 0, $pos);
584
585     return $string;
586 }
587
588 ?>
```

Listing E.8: vulab.css

```
2  /*
3   * vulab.css
4   */
5
6  .head {
7      color: #000000;
8      font-size: 20px;
9      font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular
10 }
11 .border {
12     background-color: #C4C6E3;
13     border: solid 2px #2e3790
14 }
15 a:link {
16     color: #2e3790;
17     font-weight: normal;
18     font-size: 14px;
19     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
20     text-decoration: underline
21 }
22 a:hover {
23     color: #2e3790;
24     font-weight: bold;
25     font-size: 14px;
26     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
27     text-decoration: underline
28 }
29 a:active {
30     color: #4b529f;
31     font-weight: bold;
32     font-size: 14px;
33     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
34     text-decoration: underline
35 }
36 a:visited {
37     color: #2e3790;
38     font-weight: normal;
39     font-size: 14px;
40     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
41     text-decoration: underline
42 }
```

```
44 a.nocss:link {
    font-weight: normal;
    font-size: 14px;
46    font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
    text-decoration: none
48 }
a.nocss:hover {
50     font-weight: normal;
    font-size: 14px;
52     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
    text-decoration: none
54 }
a.nocss:active {
56     font-weight: normal;
    font-size: 14px;
58     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
    text-decoration: none
60 }
a.nocss:visited {
62     font-weight: normal;
    font-size: 14px;
64     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
    text-decoration: none
66 }
a.rot:link {
68     color: #FF0000;
    font-weight: normal;
70     font-size: 14px;
    font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
72     text-decoration: none
}
a.rot:hover {
74     color: #FF0000;
    font-weight: normal;
76     font-size: 14px;
    font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
78     text-decoration: none
80 }
a.rot:active {
82     color: #FF0000;
    font-weight: normal;
84     font-size: 14px;
    font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
86     text-decoration: none
}
88 a.rot:visited {
    color: #FF0000;
90     font-weight: normal;
    font-size: 14px;
92     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
    text-decoration: none
94 }
.code {
96     font-family: "Courier New",
    Courier, mono;
98     font-size: 12px;
    font-weight: normal;
100    color: #000000;
```

```

    left: 20px; clip:
102     rect(    );
        cursor: hand}
104 .error {
        color: #ff0200;
106     font-size: 14px;
        font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular
108     }
    .green {
110         color: #006400;
        font-size: 14px;
112         font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular
    }
114 .text {
        color: #000;
116         font-size: 14px;
        font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular
118     }
    .block {
120         font-size: 14px;
        font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
122         text-align: justify
    }
124 table.vulab {
        font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
126         border : solid 2px #2e3790;
        font-size : 14px;
128         background-color : #d8d8f4;
        color: #000;
130     }
    th.vulab {
132         font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
        color: #000;
134         font-size: 14px;
        background-color: #B7B7DE
136     }
    td.vulab {
138         font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
        background-color: #d8d8f4;
140         line-height : 130%;
        font-size : 14px;
142         color : #000 ;
    }
144 td.error {
        font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
146         background-color: #d8d8f4;
        line-height : 130%;
148         font-size : 14px;
        color : #FF0200 ;
150     }

152 /*****
    * calender
154 *****/

156 .monthname
{
158     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;

```

```

160     BACKGROUND-COLOR: #006699;
161     COLOR: #cccccc;
162     FONT-SIZE: 8pt;
163     FONT-WEIGHT: bold
164 }
165 .weekdayname
166 {
167     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
168     BACKGROUND-COLOR: #000000;
169     COLOR: #ffffff;
170     FONT-SIZE: 8pt;
171     FONT-WEIGHT: bold;
172     TEXT-ALIGN: center
173 }
174 .weekday
175 {
176     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
177     BACKGROUND-COLOR: #ffffff;
178     COLOR: #000000;
179     FONT-SIZE: 8pt;
180     TEXT-ALIGN: right;
181     text-decoration : none;
182 }
183 .sat
184 {
185     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
186     BACKGROUND-COLOR: #ffffff;
187     COLOR: #666666;
188     FONT-SIZE: 8pt;
189     text-decoration : none;
190     TEXT-ALIGN: right
191 }
192 .sun
193 {
194     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
195     BACKGROUND-COLOR: #ffffff;
196     COLOR: #ff0000;
197     FONT-SIZE: 8pt;
198     TEXT-ALIGN: right;
199     text-decoration : none;
200 }
201 .holiday
202 {
203     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
204     FONT-SIZE: 8pt;
205     TEXT-ALIGN: right;
206     text-decoration : none;
207 }
208 .notthismonth
209 {
210     font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;
211     BACKGROUND-COLOR: #e1e1e1;
212     COLOR: #999999;
213     FONT-SIZE: 8pt;
214     TEXT-ALIGN: right;
215     text-decoration : none;
216 }
217 input, select, option

```

```
218 {  
    font-family: Arial, Helvetica, Geneva, Swiss, SunSans-Regular;  
    FONT-SIZE: 10pt;  
220 }  
#today  
222 {  
    border : 1px solid Red;  
224 }
```


E.2 Module im Defaultbereich

Listing E.9: default/config.php

```
2  <?php
   /*
4   * default/config.php
   */
6
   // Menue-Eintraege in Reihenfolge ihres Auftretens
8  $items = array();
   $items[1] = "Home";
10  $items[2] = "Login";
   $items[3] = "Informationen";
12
   // Zuweisung der Verzeichnisse auf die Menue-Namen
14  // $access[Menue-Name] = Verzeichniss-Name
   $access = array();
16  $access["Informationen"] = "info";
   $access["home"] = "home";
18  $access["login"] = "login";
   $access["Home"] = "home";
20  $access["Login"] = "login";
   $access["firstLogin"] = "firstLogin";
22  $access["bestaetigen"] = "bestaetigen";
   $access["error"] = "error";
24
   // Zuweisung der Beschreibung auf die Menue-Namen
26  // $description[Menue-Name] = Beschreibung
   $description = array();
28  $description["Home"] = "Link auf diese Seite";
   $description["Login"] = "Einloggen des Benutzers in das " .
30       "virtuelle Unix Labor";
   $description["Informationen"] = "Informationen &uuml;ber das " .
32       "virtuelle Unix Labor";
34  ?>
```

E.2.1 Home

Listing E.10: default/home/index.php

```
2  <?php
   /*
4   * default/home/index.php
   */
```

```
6      // Verfuegbare Variablen
8      // =====
9      //
10     // GET
11     // ----
12     // $HTTP_GET_VARS['menu']
13     // $HTTP_GET_VARS['content']
14     // $HTTP_GET_VARS['msg']
15
16     $debug = false;
17
18     require_once "functions.php";
19
20     // Fehlerbehandlung
21     ob_start('ob_gzhandler');
22     set_error_handler('handleErrors');
23     $scriptName = "default/home/index.php";
24
25     if ($debug)
26     {
27         echo "Post:<br>";
28         showPostVar();
29         echo "Get:<br>";
30         showGetVar();
31     }
32     ?>
33
34     <div align="center">
35         <p class="head">
36             Willkommen im <br>
37             virtuellen Unix Labor
38         </p>
39     </div>
40     <div align="left">
41         <p class="block">
42             Auf diesen Seiten haben Sie die M&ouml;glichkeit, an einigen &Uuml;bungen
43             zur Systemadministration unter Unix teilzunehmen. Mit Hilfe dieses Projekts
44             arbeiten Sie auf einem Computer unter dem Betriebssystem Solaris.
45             Durch den Zugang als 'root' auf dieses System, k&ouml;nnen Sie alles
46             ausprobieren, wozu Sie sonst keine Berechtigung haben ...
47         </p>
48     </div>
49
50     <div align="center">
51         <p class="border">
52             <?php
53                 echo home("default");
54             ?>
55         </p>
56     </div>
```

E.2.2 Login

Listing E.11: default/login/index.php

```

1 <?php
2
3     /*
4      * default/login/index.php
5      */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['PHPSESSID']
15    //
16    // SESSION
17    // -----
18    // beim ersten Aufruf:
19    // $HTTP_SESSION_VARS['projectPath']
20    // $HTTP_SESSION_VARS['absoluteProjectPath']
21    // danach:
22    // $HTTP_SESSION_VARS['errors']
23    // $HTTP_SESSION_VARS['formVars']
24    // $HTTP_SESSION_VARS['auth']
25    // $HTTP_SESSION_VARS['login']
26    // $HTTP_SESSION_VARS['menu']
27
28    $debug = false;
29
30    // hole Session-Variable
31    if (isset($HTTP_GET_VARS["PHPSESSID"]))
32        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
33    elseif (isset($HTTP_POST_VARS["PHPSESSID"]))
34        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
35
36    // hole vorhandene Session
37    session_start();
38    $PHPSESSID=session_id();
39
40    // falls Session verfuegbar, hole Session-Variablen
41    if (!empty($PHPSESSID))
42    {
43        if (isset($HTTP_SESSION_VARS["errors"]))
44            $errors = $HTTP_SESSION_VARS["errors"];
45
46        if (isset($HTTP_SESSION_VARS["formVars"]))
47            $formVars = $HTTP_SESSION_VARS["formVars"];
48
49        if (isset($HTTP_SESSION_VARS["auth"]))
50            $HTTP_SESSION_VARS["auth"] = false;
51
52        if (isset($HTTP_SESSION_VARS["login"]))

```

```

        $HTTP_SESSION_VARS["login"] = "";
54
        if (isset($HTTP_SESSION_VARS["projectPath"]))
56            $projectPath = $HTTP_SESSION_VARS["projectPath"];

58            if (isset($HTTP_SESSION_VARS["projectPath"]))
                $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
60    }

62    require_once $absoluteProjectPath . "functions.php";

64    // Fehlerbehandlung
    ob_start('ob_gzhandler');
66    set_error_handler('handleErrors');
    $scriptName = "default/login/index.php";
68

    if ($debug)
70    {
        echo "Session: " . session_id() . "<br>";
72        print_r($HTTP_SESSION_VARS);
        echo "<br>";
74        echo "Post:<br>";
        print_r($HTTP_POST_VARS);
76        echo "<br>";
        echo "Get:<br>";
78        print_r($HTTP_GET_VARS);
        echo "<br>";
80    }

82    function getFormVars($fieldName)
    {
84        global $formVars;
        $value = "";

86
        if (isset($formVars[$fieldName]))
88            $value = $formVars[$fieldName];

90        return $value;
    }

92
    function getError($fieldName)
94    {
        global $errors;

96
        $msg = "";

98
        if (isset($errors[$fieldName]))
100            $msg = "<p class=\"error\">" . $errors[$fieldName] . "</p>";

102        return $msg;
    }

104
    function getErrors()
106    {
        global $errors;
108        $msg = "";

110        if (isset($errors))

```

```

112         {
113             $msg .= "<p class=\"error\">\n";
114
115             foreach ($errors as $value)
116             {
117                 $msg .= $value . "<br>\n";
118             }
119
120             $msg .= "</p>\n";
121         }
122
123         return $msg;
124     }
125 }
126
127 <div align="center">
128     <p class="head">Login</p>
129     <p class="text">Bitte geben Sie hier Ihre Daten ein.</p>
130
131     <?php echo getErrors(); ?>
132
133     <form name="" action="<?php echo $projectPath .
134         'default/login/validateLogin.php' ?>" method="post">
135
136     <?php
137         if (!empty($PHPSESSID))
138             echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" .
139                 $PHPSESSID . "\">\n";
140     ?>
141
142     <div align="center">
143         <p class="border">
144             <table border="0" cellpadding="3" cellspacing="3">
145                 <tr>
146                     <td align="right">
147                         <p class="text">Login</p>
148                     </td>
149                     <td>
150                         <input type="text" name="login" size="30"
151                             value="<?php echo getFormVars('login') ?>">
152                     </td>
153                 </tr>
154                 <tr>
155                     <td align="right">
156                         <p class="text">Passwort</p>
157                     </td>
158                     <td>
159                         <input type="password" name="password" size="30"
160                             value="<?php echo getFormVars('password') ?>">
161                     </td>
162                 </tr>
163                 <tr>
164                     <td colspan="2" align="center">
165                         <p class="text">
166                             <input type="submit" name="" value="Senden">&nbsp;
167                             <input type="reset" value="Zurücksetzen">
168                         </p>
169                     </td>
170                 </tr>
171             </table>
172         </p>
173     </div>
174
175     </div>
176 </div>

```

```

170         </tr>
        </table>
        </p>
172    </div>
</form>
174 <p class="text">
    Sie sind noch nicht registriert !?<br>
176    Dann mu&szlig; f&uuml;r Sie zuerst ein
    <a href="<?php echo $HTTP_SERVER_VARS['PHP_SELF'] .
178        "?menu=default&content=firstLogin&PHPSESSID=" .
        $PHPSESSID ?> ">ein Profil</a>
180    angelegt werden.
    </p>
182    <p class="text">
    </p>
184 </div>

```

Listing E.12: default/login/validateLogin.php

```

<?php
2
    /*
4     * default/login/validateLogin.php
    */
6
    // Verfuegbare Variablen
8    // =====
    //
10   // POST
    // ----
12   // $HTTP_POST_VARS['login']
    // $HTTP_POST_VARS['password']
14   //
    // SESSION
16   // -----
    // beim ersten Aufruf:
18   // $HTTP_SESSION_VARS['projectPath']
    // $HTTP_SESSION_VARS['absoluteProjectPath']
20   // danach:
    // $HTTP_SESSION_VARS['errors']
22   // $HTTP_SESSION_VARS['formVars']
    // $HTTP_SESSION_VARS['auth']
24   // $HTTP_SESSION_VARS['login']
    // $HTTP_SESSION_VARS['menu']
26
    $debug = false;
28
    function setSessionVars()
30    {
        global $errors, $formVars, $auth, $login;
32        global $menu;
        global $HTTP_SESSION_VARS;
34
        $HTTP_SESSION_VARS["errors"] = $errors;

```

```

36     $HTTP_SESSION_VARS["formVars"] = $formVars;
37     $HTTP_SESSION_VARS["auth"] = $auth;
38     $HTTP_SESSION_VARS["login"] = $login;
39     $HTTP_SESSION_VARS["menu"] = $menu;
40 }

42 // hole Session-Variable
43 if (isset($HTTP_GET_VARS["PHPSESSID"]))
44     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
45 else
46     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];

48 // hole vorhandene Session
49 session_start();
50 $PHPSESSID=session_id();

52 // falls Session verfuegbar, hole Session-Variablen
53 if (!empty($PHPSESSID))
54 {
55     if (isset($HTTP_SESSION_VARS["projectPath"]))
56         $projectPath = $HTTP_SESSION_VARS["projectPath"];

58     if (isset($HTTP_SESSION_VARS["projectPath"]))
59         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
60 }

62 require_once $absoluteProjectPath . "functions.php";

64 // Fehlerbehandlung
65 ob_start('ob_gzhandler');
66 set_error_handler('handleErrors');
67 $scriptName = "default/login/validateLogin.php";

68
69 if ($debug)
70 {
71     echo "Session: " . session_id() . "<br>";
72     showSessionVar();
73     echo "Post:<br>";
74     showPostVar();
75     echo "Get:<br>";
76     showGetVar();
77 }

78
79 if ($debug)
80 {
81     echo "Session: " . session_id() . "<br>";
82     print_r($HTTP_SESSION_VARS);
83     echo "<br>";
84     echo "Post:<br>";
85     print_r($HTTP_POST_VARS);
86     echo "<br>";
87     echo "Get:<br>";
88     print_r($HTTP_GET_VARS);
89     echo "<br>";
90 }

92 // Variablen in der Session registrieren
93 if (!session_is_registered("errors"))

```

```

94     session_register("errors");

96     if (!session_is_registered("formVars"))
        session_register("formVars");

98

100    if (!session_is_registered("auth"))
        session_register("auth");

102    if (!session_is_registered("login"))
        session_register("login");

104

106    if (!session_is_registered("menu"))
        session_register("menu");

108    // Auslesen der Formulardaten
    foreach($HTTP_POST_VARS as $varname => $value)
110        $formVars[$varname] = trim($value);

112    // Session-Variablen
    $errors = array();
114    $auth = false;
    $login = "";
116    $menu = "default";

118    if (!empty($HTTP_POST_VARS))
    {
120        // Ueberpruefung auf leere Felder
        if (empty($formVars['login']) || empty($formVars['passwort']))
122        {
            $errors[] = "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.";
124
            if ($debug)
                echo "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.<br>";
126        }

128        // Es sind Fehler vorhanden
        if (count($errors) > 0)
130        {
            setSessionVars();

132            if (!$debug)
                header ("Location: " . $projectPath .
134                    "index_session.php?menu=default&content=login&PHPSESSID=" .
                    session_id());
136            else
                echo ("Location: " . $projectPath .
138                    "index_session.php?menu=default&content=login&PHPSESSID=" .
                    session_id());
140            exit;
142        }
144        // alle Eingaben sind richtig erfolgt
        else
146        {
            // baue Verbindung zur Datenbank auf
148            $connection = getConnection();

150            // frage nach login
            $sql = "select passwort, typ from benutzer where login='" .

```



```

152         $formVars['login'] . "'";
154
155         $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
156             "Wenden Sie sich an Ihren Administrator.";
157
158         $result = executeStatement($connection, $sql, $msg);
159
160         // es darf nur ein login verfuegbar sein
161         if (pg_numrows($result) != 1)
162         {
163             $errors[] = "Login fehlgeschlagen.<br>Evtl. ist Ihre " .
164                 "eMail-Adresse als Login gesetzt.";
165
166             if ($debug)
167                 echo "Login fehlgeschlagen.<br>Evtl. ist Ihre " .
168                     "eMail-Adresse als Login gesetzt.<br>";
169         }
170         else
171         {
172             // holt die erste Zeile des Resultats
173             // enthaellt password und typ
174             $array = fetchArray($result, 0);
175
176             // $array["password"] haellt das verschluesselte Passwort
177             if (checkPass($formVars['password'], $array["password"]))
178             {
179                 // password korrekt
180                 $auth = true;
181                 $login = $formVars['login'];
182
183                 // $array['typ'] haellt Informationen ueber admin, dozent oder user
184                 $menu = $array['typ'];
185
186                 setSessionVars();
187                 session_unregister("formVars");
188
189                 if (!$debug)
190                     header ("Location: " . $projectPath .
191                         "index_session.php?menu=$menu&content=home&PHPSESSID=" .
192                         session_id());
193                 else
194                     echo ("Location: " . $projectPath .
195                         "index_session.php?menu=$menu&content=home&PHPSESSID=" .
196                         session_id());
197                 exit;
198             }
199             else
200             {
201                 $errors[] = "Login fehlgeschlagen.<br>Evtl. ist Ihre " .
202                     "eMail-Adresse als Login gesetzt.";
203
204                 if ($debug)
205                     echo "Login fehlgeschlagen.<br>Evtl. ist Ihre eMail-Adresse " .
206                         "als Login gesetzt.<br>";
207             }
208         }
209
210         if (count($errors) > 0)

```

```

210         {
211             setSessionVars();
212
213             if (!$debug)
214                 header ("Location: " . $projectPath .
215                     "index_session.php?menu=default&content=login&PHPSESSID=" .
216                     session_id());
217             else
218                 echo ("Location: " . $projectPath .
219                     "index_session.php?menu=default&content=login&PHPSESSID=" .
220                     session_id());
221             exit;
222         }
223     }
224 }
?>

```

E.2.3 Profil anlegen

Listing E.13: default/firstLogin/index.php

```

<?php
2
3     /*
4     * default/firstLogin/index.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // POST
11    // ----
12    // $HTTP_POST_VARS['matrikelNr']
13    // $HTTP_POST_VARS['vorname']
14    // $HTTP_POST_VARS['name']
15    // $HTTP_POST_VARS['mail']
16    // $HTTP_POST_VARS['password']
17    // $HTTP_POST_VARS['secondPassword']
18    // $HTTP_POST_VARS['PHPSESSID']
19    //
20    // SESSION
21    // -----
22    // beim ersten Aufruf:
23    // $HTTP_SESSION_VARS['projectPath']
24    // $HTTP_SESSION_VARS['absoluteProjectPath']
25    // danach:
26    // $HTTP_SESSION_VARS['errors']
27    // $HTTP_SESSION_VARS['formVars']
28    // $HTTP_SESSION_VARS['auth']
29    // $HTTP_SESSION_VARS['login']
30    // $HTTP_SESSION_VARS['menu']

```

```
32     $debug = false;

34     // hole Session-Variable
    if (isset($_HTTP_GET_VARS["PHPSESSID"]))
36         $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
    else
38         $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];

40     // hole vorhandene Session
    session_start();
42     $PHPSESSID=session_id();

44     // falls Session verfuegbar, hole Session-Variablen
    if (!empty($PHPSESSID))
46     {
        if (isset($_HTTP_SESSION_VARS["errors"]))
48             $errors = $_HTTP_SESSION_VARS["errors"];

50         if (isset($_HTTP_SESSION_VARS["formVars"]))
            $formVars = $_HTTP_SESSION_VARS["formVars"];
52
        if (isset($_HTTP_SESSION_VARS["projectPath"]))
54             $projectPath = $_HTTP_SESSION_VARS["projectPath"];

56         if (isset($_HTTP_SESSION_VARS["projectPath"]))
            $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
58
        if (isset($_HTTP_SESSION_VARS["auth"]))
60             $_HTTP_SESSION_VARS["auth"] = false;

62         if (isset($_HTTP_SESSION_VARS["login"]))
            $_HTTP_SESSION_VARS["login"] = "";
64     }

66     require_once $absoluteProjectPath . "functions.php";

68     // Fehlerbehandlung
    ob_start('ob_gzhandler');
70     set_error_handler('handleErrors');
    $scriptName = "default/firstLogin/index.php";
72
    if ($debug)
74     {
        echo "Session: " . session_id() . "<br>";
76         showSessionVar();
        echo "Post:<br>";
78         showPostVar();
        echo "Get:<br>";
80         showGetVar();
    }
82

    // extrahiert vorhandene Formular-Variablen
84     function getFormVars($fieldName)
    {
86         global $formVars;

88         $value = "";
```

```

90     if (isset($formVars[$fieldName]))
91         $value = $formVars[$fieldName];
92
93     return $value;
94 }
95
96 // bettet eine Fehlermeldung in HTML ein
97 function getError($fieldName)
98 {
99     global $errors;
100
101     $msg = "";
102
103     if (isset($errors[$fieldName]))
104         $msg = "<p class=\"error\">" . $errors[$fieldName] . "</p>";
105
106     return $msg;
107 }
108
109 // bettet alle Fehlermeldungen in HTML ein
110 function getErrors()
111 {
112     global $errors;
113
114     $msg = "";
115
116     if (isset($errors))
117     {
118         $msg .= "<p class=\"error\">\n";
119
120         foreach ($errors as $value)
121             $msg .= $value . "<br>\n";
122
123         $msg .= "</p>\n";
124     }
125
126     return $msg;
127 }
128 ?>
129
130 <div align="center">
131     <p class="head">Profil anlegen</p>
132     <p class="text">
133         Bitte geben Sie hier Ihre Daten ein.<br>
134         Es sind alle Felder auszuf&uuml;llen.
135     </p>
136
137 <?php
138     echo getErrors();
139
140     if ($debug)
141     {
142         showGetVar();
143         showPostVar();
144         showSessionVar();
145     }
146
147 ?>

```

```

148     <form name="" action="<?php echo $projectPath .
150         'default/firstLogin/validate.php' ?>" method="post">
<?php
152         if (!empty($PHPSESSID))
            echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" .
154                 $PHPSESSID . \">\n\"";
?>
156     <div align="center">
        <p class="border">
158         <table border="0" cellpadding="3" cellspacing="3">
            <tr>
160                 <td align="right">
                    <p class="text">Matrikel-Nummer</p>
162                 </td>
                    <td><input type="text" name="matrikelNr" size="30"
164                        value="<?php echo getFormVars('matrikelNr') ?>"></td>
            </tr>
166            <tr>
                <td align="right">
168                    <p class="text">Vorname</p>
                </td>
                <td><input type="text" name="vorname" size="30"
170                    value="<?php echo getFormVars('vorname') ?>"></td>
            </tr>
172            <tr>
                <td align="right">
174                    <p class="text">Name</p>
                </td>
                <td><input type="text" name="name" size="30"
176                    value="<?php echo getFormVars('name') ?>"></td>
            </tr>
180            <tr>
                <td align="right">
182                    <p class="text">eMail-Adresse</p>
                </td>
                <td><input type="text" name="mail" size="30"
184                    value="<?php echo getFormVars('mail') ?>"></td>
            </tr>
186            <tr>
                <td align="right">
188                    <p class="text">Passwort</p>
                </td>
                <td><input type="password" name="password" size="30"
192                    value="<?php echo getFormVars('password') ?>"></td>
            </tr>
194            <tr>
                <td align="right">
196                    <p class="text">Passwort wiederholen</p>
                </td>
                <td><input type="password" name="secondPasswort" size="30"
198                    value="<?php echo getFormVars('secondPasswort') ?>"></td>
            </tr>
200            <tr>
                <td colspan="2" align="center">
202                    <p class="text">
                        <input type="submit" name="" value="Senden">&nbsp;
204

```

```

206         <input type="reset" value="Zurücksetzen">
           </p>
208
           </td>
210       </tr>
       </table>
212     </p>
     </div>
214 </form>
<p class="block"><br>
216     Nachdem Sie Ihr Profil erfolgreich angelegt wurde,
     k&ouml;nnen Sie bereits in den gesch&uuml;tzten Bereich eintreten.
218     Desweiteren wird Ihnen eine Best&auml;tigung per eMail zugesand.
     Mit ihr erhalten Sie eine Geheimnummer, die Sie einmalig zur
220     Identifikation Ihres Profils verwenden.
     Ohne diese Identifizierung wird Ihr Zugang nach 8 Tagen ung&uuml;ltig.
222 </p>
</div>

```

Listing E.14: default/firstLogin/validate.php

```

<?php
2
    /*
4     * default/firstLogin/validate.php
    */
6
    // Verfuegbare Variablen
    // =====
    //
10   // POST
    // ----
12   // $HTTP_POST_VARS['matrikelNr']
    // $HTTP_POST_VARS['vorname']
14   // $HTTP_POST_VARS['name']
    // $HTTP_POST_VARS['mail']
16   // $HTTP_POST_VARS['passwort']
    // $HTTP_POST_VARS['secondPasswort']
18   // $HTTP_POST_VARS['PHPSESSID']
    //
20   // SESSION
    // -----
22   // beim ersten Aufruf:
    // $HTTP_SESSION_VARS['projectPath']
24   // $HTTP_SESSION_VARS['absoluteProjectPath']
    // danach:
26   // $HTTP_SESSION_VARS['errors']
    // $HTTP_SESSION_VARS['formVars']
28   // $HTTP_SESSION_VARS['auth']
    // $HTTP_SESSION_VARS['login']
30   // $HTTP_SESSION_VARS['menu']
32
    $debug = false;

```

```
34 // hole Session-Variable
35 if (isset($_HTTP_GET_VARS["PHPSESSID"]))
36     $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
37 else
38     $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];
39
40 // hole vorhandene Session
41 session_start();
42 $PHPSESSID=session_id();
43
44 // falls Session verfuegbar, hole Session-Variablen
45 if (!empty($PHPSESSID))
46 {
47     if (isset($_HTTP_SESSION_VARS["projectPath"]))
48         $projectPath = $_HTTP_SESSION_VARS["projectPath"];
49
50     if (isset($_HTTP_SESSION_VARS["projectPath"]))
51         $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
52 }
53
54 if ($debug)
55 {
56     echo $projectPath . "<br>";
57     echo $absoluteProjectPath . "<br>";
58 }
59
60 require_once $absoluteProjectPath . "functions.php";
61 require_once $absoluteProjectPath . "config.php";
62
63 // Fehlerbehandlung
64 ob_start('ob_gzhandler');
65 set_error_handler('handleErrors');
66 $scriptName = "default/firstLogin/validate.php";
67
68 if ($debug)
69 {
70     echo "Session: " . session_id() . "<br>";
71     showSessionVar();
72     echo "Post:<br>";
73     showPostVar();
74     echo "Get:<br>";
75     showGetVar();
76 }
77
78 // Variablen in der Session registrieren
79 if (!session_is_registered("errors"))
80     session_register("errors");
81
82 if (!session_is_registered("formVars"))
83     session_register("formVars");
84
85 if (!session_is_registered("auth"))
86     session_register("auth");
87
88 if (!session_is_registered("login"))
89     session_register("login");
90
91 if (!session_is_registered("menu"))
```

```

92     session_register("menu");

94     // Session-Variablen
    $errors = array();
96     $auth = false;
    $login = "";
98     $menu = "default";

100    function sendMail($vorname, $name, $mail, $matrikelNr, $secret)
    {
102        global $debug, $server, $adminName, $adminMail;
        global $replyName, $replyMail, $Cc, $Bcc;
104        global $projectPath;

106        $url = "http://" . $server . $projectPath . "index.php";
        $url .= "?menu=default&content=bestaetigen&mail=";
108        $url .= $mail;
        $url .= "&secret=";
110        $url .= $secret;

112        $urlLogin = "http://" . $server . $projectPath . "index.php";
        $urlLogin .= "?menu=default&content=login";
114
        // Empfaenger
116        $to = $vorname . " " . $name . "<" . $mail . ">";

118        // Betreff
        $subject = "Anmeldung beim virtuellen Unix Labor";
120
        // Nachricht
122        $message = "Willkommen " . $vorname . " " . $name . " !!<br><br>\n";
        $message .= "Sie haben sich soeben beim 'virtuellen Unix Labor' ";
124        $message .= "registriert.<br><br>\n";
        $message .= "Ihre Registrierungs-Daten sind:<br><br>\n";
126        $message .= "MatrikelNr: " . $matrikelNr . "<br>\n";
        $message .= "Vorname: " . $vorname . "<br>\n";
128        $message .= "Name: " . $name . "<br>\n";
        $message .= "eMail: " . $mail . "<br>\n";
130        $message .= "login: " . $mail . "<br><br>\n";
        $message .= "Ihre Registrierungs-Nummer ist:<br>\n";
132        $message .= "<b>" . $secret . "</b><br><br>\n";
        $message .= "Bitte benutzen Sie folgende URL um Ihre Anmeldung zu bestaetigen.<br>\n";
134        $message .= "<p><a href='" . $url . "'>" . $url . "</a></p>\n";
        $message .= "Ohne diese Bestaetigung werden Ihre Daten nach acht Tagen ";
136        $message .= "geloescht.<br><br>\n";
        $message .= "Sie koennen Ihre persoenlichen Daten jederzeit ueber den ";
138        $message .= "Menue-Punkt 'Benutzerdaten' ";
        $message .= "anpassen. Bitte benutzen Sie folgende URL um sich einzuloggen.<br>\n";
140        $message .= "<p><a href='" . $urlLogin . "'>" . $urlLogin . "</a></p>\n";
        $message .= "Ihr " . $adminName . "<br>\n";
142
        /* To send HTML mail, you can set the Content-type header. */
144        $headers = "MIME-Version: 1.0\r\n";
        $headers .= "Content-type: text/html; charset=iso-8859-1\r\n";
146
        /* additional headers */
148        $headers .= "From: virtuelles Unix Labor <" . $adminMail . ">\r\n";
        $headers .= "Reply-to: " . $replyName . " <" . $replyMail . ">\r\n";

```



```

150     $headers .= "Cc: " . $Cc . "\r\n";
151     $headers .= "Bcc: " . $Bcc . "\r\n";
152
153     /* and now mail it */
154     if (!$debug)
155         mail($to, $subject, $message, $headers);
156
157     if ($debug)
158     {
159         echo "<br><br>Mail : <br>\n";
160         echo $to . "<br>\n";
161         echo $headers . "<br>\n";
162         echo $subject . "<br>\n";
163         echo $message . "<br>\n";
164     }
165 }
166
167 function setSessionVars()
168 {
169     global $errors, $formVars, $auth, $login, $menu;
170     global $HTTP_SESSION_VARS;
171
172     $HTTP_SESSION_VARS["errors"] = $errors;
173     $HTTP_SESSION_VARS["formVars"] = $formVars;
174     $HTTP_SESSION_VARS["auth"] = $auth;
175     $HTTP_SESSION_VARS["login"] = $login;
176     $HTTP_SESSION_VARS["menu"] = $menu;
177 }
178
179 // Kontrolliert, ob ein Eintrag die maximale
180 // Laenge ueberschreitet
181 // $name: Bezeichnung des Feldes
182 // $value: Inhalt des Feldes
183 // $length: maximale Feldlaenge
184 function checkFieldLength($name, $value, $length)
185 {
186     global $errors, $debug;
187
188     if (strlen($value) > $length)
189     {
190         $errors[] = "Das Feld ". $name . " kann maximal " .
191             $length . " Zeichen aufnehmen.";
192         if ($debug)
193             echo "Das Feld ". $name . " kann maximal " .
194                 $length . " Zeichen aufnehmen.<br>\n";
195     }
196 }
197
198 // Auslesen der Formulardaten
199 foreach($HTTP_POST_VARS as $varname => $value)
200     $formVars[$varname] = trim($value);
201
202 // Session-Variablen
203 $errors = array();
204 $auth = false;
205 $login = "";
206
207 $mailDomain = substr( strstr($formVars['mail'], '@'), 1 );

```

```

208     if ($debug)
210         echo "Domain : " . $mailDomain . "<br>";

212     $validEmailExpr =
213         "[0-9a-z~!#$%&_~]([.]?[0-9a-z~!#$%&_~])*".
214         "@[0-9a-z~!#$%&_~]([.]?[0-9a-z~!#$%&_~])*".
215         "\.[0-9a-z~!#$%&_~]([0-9a-z~!#$%&_~])*$";

216     $validNumExpr = "[0-9]*$";

218     $validNumLength = "[0-9]{1,10}$";

220     if (!empty($HTTP_POST_VARS))
221     {
222         // Ueberpruefung auf leere Felder
223         if ( ( empty($formVars['matrikelNr']) && !is_numeric($formVars['matrikelNr']) ) ||
224             empty($formVars['name']) ||
225             empty($formVars['vorname']) ||
226             empty($formVars['mail']) ||
227             empty($formVars['passwort']) ||
228             empty($formVars['secondPasswort'])
229         )
230         {
231             $errors[] = "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.";
232             if ($debug)
233                 echo "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.<br>";
234         }

235         // ueberpruefung auf Nummern-Felder
236         if (!ereg($validNumExpr, $formVars['matrikelNr']))
237         {
238             $errors[] = "Die Matrikel-Nummer darf keine Buchstaben oder " .
239                 "andere Zeichen enthalten.";
240             if ($debug)
241                 echo "Die Matrikel-Nummer darf keine Buchstaben oder " .
242                     "andere Zeichen enthalten.<br>";
243         }

244         // ueberpruefung auf maximal 10 Stellen
245         elseif (!ereg($validNumLength, $formVars['matrikelNr']))
246         {
247             $errors[] = "Die MatrikelNr ist auf 10 Ziffern beschr&uuml;nkt.";
248             if ($debug)
249                 echo "Die MatrikelNr ist auf 10 Ziffern beschr&uuml;nkt.<br>";
250         }

251         // ueberpruefung auf korrekte email-Form
252         if (!ereg($validEmailExpr, $formVars['mail']))
253         {
254             $errors[] = "Ihre eingegebene Email-Adresse hat keine g&uuml;ltige Form.";
255             if ($debug)
256                 echo "Ihre eingegebene Email-Adresse hat keine g&uuml;ltige Form.<br>";
257         }

258         // ueberpruefung auf korrektes Passwort
259         if ($formVars['passwort'] != $formVars['secondPasswort'])
260         {

```

```

266         $errors[] = "Ihre beiden angegebenen Passw&ouml;rter stimmen " .
                "nicht &uuml;berein.";
268         if ($debug)
                echo "Ihre beiden angegebenen Passw&ouml;rter stimmen " .
270                 "nicht &uuml;berein.<br>";
    }

272
    checkFieldLength ("Vorname", $formVars["vorname"], 50);
274    checkFieldLength ("Name", $formVars["name"], 50);
    checkFieldLength ("eMail", $formVars["mail"], 80);
276    checkFieldLength ("Passwort", $formVars["passwort"], 50);
    checkFieldLength ("Passwort wiederholen", $formVars["secondPasswort"], 50);
278
    // Es sind Fehler vorhanden
280    if (count($errors) > 0)
    {
282        setSessionVars();

284        if (!$debug)
            header ("Location: " . $projectPath .
286                "index_session.php?menu=default&content=firstLogin&PHPSESSID=" .
                session_id());
        else
288            echo ("Location: " . $projectPath .
290                "index_session.php?menu=default&content=firstLogin&PHPSESSID=" .
                session_id());
292        exit;
    }
294    // alle Eingaben sind richtig erfolgt
    else
296    {
        if ($debug)
298            echo "versuche Verbindung zur Datenbank zu oeffnen ...<br>\n";

        // baue Verbindung zur Datenbank auf
        $connection = getConnection();
302        if ($debug)
            echo "Connection: " . $connection . "<br>\n";
304
        $sql = "select * from benutzer where login='" . $formVars['mail'] . "'";
306
        $msg = "Bei der Ueberpruefung der eMail-Adresse ist ein Fehler aufgetreten.<br>".
308                "Wenden Sie sich an Ihren Administrator.";

        $result = executeStatement($connection, $sql, $msg);
310
        if ($debug)
            echo "Result: " . $result . "<br>\n";
312
        // email darf es noch nicht geben
        if (pg_numrows($result) > 0)
        {
318            $errors[] = "Diese eMail-Adresse wird bereits von einer " .
                "anderen Person verwendet.";
320
            if ($debug)
322                echo "Diese eMail-Adresse wird bereits von einer " .
                    "anderen Person verwendet.";

```

```

324     }

326     // frage nach matrikelNr
327     $sql = "select * from benutzer where matrikel_nr='" .
328         $formVars['matrikelNr'] . "'";

330     $msg = "Bei der Ueberpruefung des Logins ist ein Fehler aufgetreten.<br>".
331         "Wenden Sie sich an Ihren Administrator.";

332

333     $result = executeStatement($connection, $sql, $msg);

334

335     if ($debug)
336         echo "Result: " . $result . "<br>\n";

337
338     // matrikelNr darf es noch nicht geben
339     if (pg_numrows($result) > 0)
340     {
341         $errors[] = "Unter dieser Matrikelnummer ist bereits " .
342             "eine anderer Person registriert.";

343
344         if ($debug)
345             echo "Unter dieser Matrikelnummer ist bereits " .
346                 "eine anderer Person registriert.<br>";
347     }

348
349     // Es sind Fehler vorhanden
350     if (count($errors) > 0)
351     {
352         setSessionVars();

353
354         if (!$debug)
355             header ("Location: " . $projectPath .
356                 "index_session.php?menu=default&content=firstLogin&PHPSESSID=" .
357                 session_id());
358         else
359             echo ("Location: " . $projectPath .
360                 "index_session.php?menu=default&content=firstLogin&PHPSESSID=" .
361                 session_id());
362         exit;
363     }

364     // Eintag in die Datenbank
365     else
366     {
367         if ($debug)
368             echo "Es sind alle Eingaben erfolgt.<br>\n";

369
370         $cryptPass = createPass($formVars['passwort']);
371         $secret = makeSecret();

372
373         $sql = "insert into benutzer ";
374         $sql .= "(matrikel_nr, vorname, nachname, email, login, ";
375         $sql .= "passwort, anmeldedatum, freischalt_secret) ";
376         $sql .= "values ";
377         $sql .= "(" . $formVars['matrikelNr'] . ", ' " .
378             $formVars['vorname'] . ", ' " .
379             $formVars['name'] . ", ' " .
380             $formVars['mail'] . ", ' " .
381             $formVars['mail'] . ", ' " .

```

```

382         $cryptPass . " ', '" .
383         "now" . " ', '" . // 'now' ist PostgreSQL-Funktionalitaet
384                             //und liefert den aktuellen Timestamp
385         $secret . "')";
386
387     if ($debug)
388         echo $sql . "<br>\n";
389
390     $msg = "Bei Schreiben in die Datenbank ist ein Fehler aufgetreten.<br>".
391           "Wenden Sie sich an Ihren Administrator.";
392
393     if (!$debug)
394         $result = executeStatement($connection, $sql, $msg);
395
396     if ($debug)
397         echo $result;
398
399     // korrekt authentifiziert
400     $auth = true;
401     $login = $formVars['mail'];
402     $formVars['login'] = $formVars['mail'];
403     $menu = "user";
404
405     // sende eMail
406     sendMail($formVars['vorname'], $formVars['name'], $formVars['mail'],
407             $formVars['matrikelNr'], $secret);
408
409     setSessionVars();
410     session_unregister("formVars");
411
412     if (!$debug)
413         header ("Location: " . $projectPath .
414               "index_session.php?menu=user&content=home&PHPSESSID=" .
415               session_id());
416     else
417         echo ("Location: " . $projectPath .
418               "index_session.php?menu=user&content=home&PHPSESSID=" .
419               session_id());
420     exit;
421 }
422 }
423 }
424 ?>

```

E.2.4 Info

Listing E.15: default/info/index.php

```

1 <?php
2
3 /*
4  * default/info/index.php

```

```

        */
6
    // Verfuegbare Variablen
8    // =====
    //
10   // GET
    // ----
12   // $HTTP_GET_VARS['menu']
    // $HTTP_GET_VARS['content']
14   // $HTTP_GET_VARS['msg']

16   $debug = false;

18   require_once "functions.php";

20   // Fehlerbehandlung
    ob_start('ob_gzhandler');
22   set_error_handler('handleErrors');
    $scriptName = "default/info/index.php";
24

    if ($debug)
26   {
        echo "Post:<br>";
28        showPostVar();
        echo "Get:<br>";
30        showGetVar();
    }
32 ?>

34 <div align="center">
    <p class="head">
36        Informationen über das<br>
        virtuellen Unix Labor
38    </p>
</div>
40 <div align="left">
    <p class="text">
42        Das Virtuelle Unix Labor ist ein interaktives Kurssystem für den Bereich
        Unix-Systemverwaltung. Nach Anmeldung werden Übungsrechner für den Student
44        vorbereitet, er erhält vollen root-Zugriff und kann ohne Einschränkungen die
        vorgegebene Übungen absolvieren. Am Ende überprüft das System ob/welche Teile
46        der Übung korrekt ausgeführt wurden, und erstellt eine Auswertung. Der Zugriff
        erfolgt über das Internet mittels Web-Browser sowie gängiger Unix-Clients (ssh,
48        telnet, ftp).
    </p>
50
    <p class="text">
52        Weitere Informationen zum Virtuellen Unix Labor finden Sie
        <a href="http://www.feyrer.de/vulab/">hier</a>.
54    </p>
</div>

```

E.2.5 Bestätigen

Listing E.16: default/bestaetigen/index.php

```

1  <?php
2
3      /*
4       * default/bestaetigen/index.php
5       */
6
7      // Verfuegbare Variablen
8      // =====
9      //
10     // GET
11     // ----
12     // $HTTP_GET_VARS['menu']
13     // $HTTP_GET_VARS['content']
14     // $HTTP_GET_VARS['mail']
15     // $HTTP_GET_VARS['secret']
16
17     $debug = false;
18
19     require_once "functions.php";
20
21     // Fehlerbehandlung
22     ob_start('ob_gzhandler');
23     set_error_handler('handleErrors');
24     $scriptName = "default/bestaetigen/index.php";
25
26     if ($debug)
27     {
28         echo "Post:<br>";
29         showPostVar();
30         echo "Get:<br>";
31         showGetVar();
32     }
33 ?>
34
35 <div align="center">
36     <p></p>
37     <p class="head">Best&auml;tigung des Profils</p>
38 </div>
39 <div align="center">
40     <br><br>
41     <p class="border">
42         <table>
43             <tr>
44                 <td>
45                     <div align="center">
46
47
48 <?php
49     foreach($HTTP_GET_VARS as $varname => $value)
50         $formVars[$varname] = trim($value);
51
52     $connection = getConnection();

```

```

54      // frage nach mail und secret
55      $sql = "SELECT * FROM benutzer WHERE email='" . $formVars['mail'] .
56            "' AND freischalt_secret='" . $formVars['secret'] . "'";

58      $msg = "Beim Ueberpruefen der Registrierungs-Nummer ist ein Fehler aufgetreten.<br>".
59            "Wenden Sie sich an Ihren Administrator.";

60
61      $result = executeStatement($connection, $sql, $msg);
62
63      if ($result)
64      {
65          // es darf nur ein Ergebnis geben
66          if (pg_numrows($result) != 1)
67              echo "<p class=\"error\">Diese URL ist nicht bzw. nicht mehr g&uuml;ltig.</p>";
68          else
69          {
70              $sql = "UPDATE benutzer SET freischalt_secret='bestaetigt' WHERE email='" .
71                    $formVars['mail'] . "'";
72
73              $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
74                    "Wenden Sie sich an Ihren Administrator.";
75
76              $result = executeStatement($connection, $sql, $msg);
77
78              if ($result)
79                  echo "<p class=\"text\">Ihr Profil wurde erfolgreich best&auml;tigt.</p>";
80              else
81              {
82                  echo "<p class=\"error\">Bei der Best&auml;tigung ist ein Datenbank-Fehler " .
83                        "aufgetreten.<br>\n" .
84                        "Wenden Sie sich an Ihren Administrator.</p>";
85                  exit;
86              }
87          }
88      }
89      else
90      {
91          echo "<p class=\"error\">Es konnte keine Verbindung zur Datenbank " .
92                "hergestellt werden.<br>\n" .
93                "Bitte versuchen sie es zu einem sp&auml;teren Zeitpunkt nochmal.</p>";
94      }
95  ?>
96
97      </div>
98
99      </td>
100
101      </tr>
102
103      </table>
104
105      </p>
106
107  </div>

```


E.2.6 Error

Listing E.17: default/error/index.php

```
<?php
2
3     /*
4     * default/error/index.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['msg']
15    // $HTTP_GET_VARS['PHPSESSID']
16    //
17    // POST
18    // ----
19    // ???
20    //
21    // SESSION
22    // -----
23    // beim ersten Aufruf:
24    // $HTTP_SESSION_VARS['projectPath']
25    // $HTTP_SESSION_VARS['absoluteProjectPath']
26    // ???
27
28    $debug = false;
29
30    // hole Session-Variable
31    if (isset($HTTP_GET_VARS["PHPSESSID"]))
32        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
33    elseif (isset($HTTP_POST_VARS["PHPSESSID"]))
34        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
35
36    // hole vorhandene Session
37    session_start();
38    $PHPSESSID=session_id();
39
40    // falls Session verfuegbar, hole Session-Variablen
41    if (!empty($PHPSESSID))
42    {
43        if (isset($HTTP_SESSION_VARS["projectPath"]))
44            $projectPath = $HTTP_SESSION_VARS["projectPath"];
45
46        if (isset($HTTP_SESSION_VARS["projectPath"]))
47            $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
48    }
49
50    require_once $absoluteProjectPath . "functions.php";
51
52    // Fehlerbehandlung
```

```

54  ob_start('ob_gzhandler');
    set_error_handler('handleErrors');
    $scriptName = "default/error/index.php";

56
    if ($debug)
58  {
        echo "Session: " . session_id() . "<br>";
60        showSessionVar();
        echo "Post:<br>";
62        showPostVar();
        echo "Get:<br>";
64        showGetVar();
    }
66  ?>

68  <div align="center">
    <br>
70    <p></p>
    <p class="head">Fehler im vuLab</p>
72  </div>
    <div align="center">
74        <p class="error">
            <table>
76                <tr>
                    <td>
78                        <div align="center">
80
82  <?php
        foreach($HTTP_GET_VARS as $varname => $value)
            $formVars[$varname] = trim($value);

84
        echo "<p class=\"error\">";
86        echo $formVars["msg"];
        echo "</p>";
88  ?>

90                        </div>
                    </td>
92                </tr>
            </table>
94        </p>
    </div>

```

E.3 Module im Anwenderbereich

Listing E.18: user/config.php

```
<?php
2
/*
4  * user/config.php
  */
6
// Menue-Eintraege in Reihenfolge ihres Auftretens
8 $items = array();
  $items[1] = "home";
10 $items[2] = "Benutzerdaten";
  $items[3] = "Buchung vornehmen";
12 $items[4] = "Buchungen einsehen";
  $items[7] = "logout";
14
// Zuweisung der Verzeichnisse auf die Menue-Namen
16 // $access[Menue-Name] = Verzeichniss-Name
  $access = array();
18 $access["error"] = "error";
  $access["home"] = "home";
20 $access["Benutzerdaten"] = "userdata";
  $access["Buchung vornehmen"] = "buchen";
22 $access["Buchungen einsehen"] = "buchungen";
  // keine Zuweisung fuer logout notwendig
24 // $access["logout"] = "logout";

26 // Zuweisung der Beschreibung auf die Menue-Namen
  // $description[Menue-Name] = Beschreibung
28 $description = array();
  $description["home"] = "Link auf diese Seite";
30 $description["Benutzerdaten"] = "Hier koennen Sie Ihre " .
    "persoennlichen Daten aendern";
32 $description["Buchung vornehmen"] = "Reservieren Sie sich Uebungsaufgaben, " .
    "die Sie demnaechst bearbeiten wollen";
34 $description["Buchungen einsehen"] = "Ueberpruefen Sie Ihre " .
    "getriggerten Buchungen";
36 $description["logout"] = "Verlassen Sie ihr Profil";
38 ?>
```

E.3.1 Home

Listing E.19: user/home/index.php

```

1  <?php
2
3      /*
4       * user/home/index.php
5       */
6
7      // Verfuegbare Variablen
8      // =====
9      //
10     // GET
11     // ----
12     // $HTTP_GET_VARS['menu']
13     // $HTTP_GET_VARS['content']
14     // $HTTP_GET_VARS['PHPSESSID']
15     //
16     // SESSION
17     // -----
18     // $HTTP_SESSION_VARS['projectPath']
19     // $HTTP_SESSION_VARS['absoluteProjectPath']
20     // $HTTP_SESSION_VARS['errors']
21     // $HTTP_SESSION_VARS['auth']
22     // $HTTP_SESSION_VARS['login']
23     // $HTTP_SESSION_VARS['menu']
24
25     $debug = false;
26
27     // hole Session-Variable
28     if (isset($HTTP_GET_VARS["PHPSESSID"]))
29         $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
30     else
31         $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
32
33     // hole vorhandene Session
34     session_start();
35     $PHPSESSID=session_id();
36
37     // falls Session verfuegbar, hole Session-Variablen
38     if (!empty($PHPSESSID))
39     {
40         if (isset($HTTP_SESSION_VARS["projectPath"]))
41             $projectPath = $HTTP_SESSION_VARS["projectPath"];
42
43         if (isset($HTTP_SESSION_VARS["absoluteProjectPath"]))
44             $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
45
46         if (isset($HTTP_SESSION_VARS["menu"]))
47             $menu = $HTTP_SESSION_VARS["menu"];
48     }
49
50     require_once $absoluteProjectPath . "functions.php";
51
52     // Fehlerbehandlung

```

```

54  ob_start('ob_gzhandler');
    set_error_handler('handleErrors');
    $scriptName = $menu . "/home/index.php";

56
    if ($debug)
58  {
        echo "Session: " . session_id() . "<br>";
60        showSessionVar();
        echo "Post:<br>";
62        showPostVar();
        echo "Get:<br>";
64        showGetVar();
    }
66  ?>

68  <div align="center">
    <p class="head">
70        Willkommen im Anwenderbereich des<br>
        virtuellen Unix Labor
72    </p>
</div>
74  <div align="left">
    <p class="block">
76        Auf diesen Seiten haben Sie die Möglichkeit, an
        einigen &Uuml;bungen zur Systemadministration unter
78        Unix teilzunehmen. Mit Hilfe dieses Projekts
        arbeiten Sie auf einem Computer unter dem Betriebssystem
80        Solaris. Durch den Zugang als 'root' auf dieses System,
        k&ouml;nnen Sie alles ausprobieren, wozu Sie sonst keine
82        Berechtigung haben ...
    </p>
84  </div>

86  <?php

88      // baue Verbindung zur Datenbank auf
      $connection = getConnection();

90
      // frage nach Bestaetigung
92      $sql = "SELECT freischalt_secret FROM benutzer WHERE login='" .
          $HTTP_SESSION_VARS['login'] . "'";

94
      if ($debug)
96          echo "SELECT freischalt_secret FROM benutzer WHERE login='" .
              $HTTP_SESSION_VARS['login'] . "'<br>";
98
      $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
          "Wenden Sie sich an Ihren Administrator.";

100
      $result = executeStatement($connection, $sql, $msg);

102
      // holt die erste Zeile des Resultats
      $array = fetchArray($result, 0);

104
      if ($array['freischalt_secret'] != "bestaetigt")
106      {
          $sql = "SELECT 8-(current_date-anmeldedatum) " .
108              "AS time_left FROM benutzer WHERE login='" .
110

```

```

112     $HTTP_SESSION_VARS['login'] . "' " ;
113
114     $result = executeStatement($connection, $sql, $msg);
115
116     // holt die erste Zeile des Resultats
117     $array = fetchArray($result, 0);
118     $time_left = $array['time_left'];
119
120     echo "<div align='center'>\n";
121     echo "<table width='80%' border='0' cellpadding='2' cellspacing='5'>";
122     echo "<tr>";
123     echo "<td valign='top'>";
124     echo "<p class='error'>\n";
125     echo "Warnung:</p></td>\n";
126     echo "<td>";
127     echo "<p class='error'>\n";
128     echo "Sie haben ihren Account noch nicht " .
129         "best&auml;tigt.<br>Verwenden Sie dazu den " .
130         "bei der Anmeldung per eMail erhaltenen " .
131         "Link um Ihr Profil permanent freizuschalten.<br>" .
132         "Ihr Profil wird sonst in $time_left Tagen " .
133         "gel&ouml;scht.</p></td>";
134     echo "\n</tr></table>\n";
135     echo "</div>\n";
136 }
137
138 <div align="center">
139 <p class="border">
140 <?php
141     echo home("user");
142 <?>
143 </p>
144 </div>

```

E.3.2 Benutzerdaten

Listing E.20: user/home/index.php

```

<?php
2
3     /*
4     * user/userdata/index.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']

```

```
14 // $HTTP_GET_VARS['PHPSESSID']
15 //
16 // SESSION
17 // -----
18 // $HTTP_SESSION_VARS['projectPath']
19 // $HTTP_SESSION_VARS['absoluteProjectPath']
20 // $HTTP_SESSION_VARS['errors']
21 // $HTTP_SESSION_VARS['auth']
22 // $HTTP_SESSION_VARS['login']
23 // $HTTP_SESSION_VARS['menu']
24
25 $debug = false;
26
27 // hole Session-Variable
28 if (isset($HTTP_GET_VARS["PHPSESSID"]))
29     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
30 else
31     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
32
33 // hole vorhandene Session
34 session_start();
35 $PHPSESSID=session_id();
36
37 // falls Session verfuegbar, hole Session-Variablen
38 if (!empty($PHPSESSID))
39 {
40     if (isset($HTTP_SESSION_VARS["projectPath"]))
41         $projectPath = $HTTP_SESSION_VARS["projectPath"];
42
43     if (isset($HTTP_SESSION_VARS["absoluteProjectPath"]))
44         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
45
46     if (isset($HTTP_SESSION_VARS["errors"]))
47         $errors = $HTTP_SESSION_VARS["errors"];
48
49     if (isset($HTTP_SESSION_VARS["formVars"]))
50         $formVars = $HTTP_SESSION_VARS["formVars"];
51
52     if (isset($HTTP_SESSION_VARS["menu"]))
53         $menu = $HTTP_SESSION_VARS["menu"];
54 }
55
56 require_once $absoluteProjectPath . "functions.php";
57
58 // Fehlerbehandlung
59 ob_start('ob_gzhandler');
60 set_error_handler('handleErrors');
61 $scriptName = $menu . "/userdata/index.php";
62
63 if ($debug)
64 {
65     echo "Session: " . session_id() . "<br>";
66     showSessionVar();
67     echo "Post:<br>";
68     showPostVar();
69     echo "Get:<br>";
70     showGetVar();
71 }
```

```

72     function getFormVars($fieldName)
73     {
74         global $formVars;
75
76         $value = "";
77
78         if (isset($formVars[$fieldName]))
79             $value = $formVars[$fieldName];
80
81         return $value;
82     }
83
84     function getErrors()
85     {
86         global $errors;
87
88         $msg = "";
89
90         if (isset($errors))
91         {
92             $msg .= "<p class=\"error\">\n";
93
94             foreach ($errors as $value)
95                 $msg .= $value . "<br>\n";
96
97             $msg .= "</p>\n";
98         }
99
100        return $msg;
101    }
102
103    // baue Verbindung zur Datenbank auf
104    $connection = getConnection();
105
106    $sql = "SELECT * FROM benutzer WHERE login='" . $login . "'";
107
108    $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
109          "Wenden Sie sich an Ihren Administrator.";
110
111    $result = executeStatement($connection, $sql, $msg);
112
113    if ($debug)
114    {
115        echo "SQL: " . $sql . "<br>";
116        echo "Result: " . $result . "<br>";
117    }
118
119    // holt die erste Zeile des Resultats
120    $array = fetchArray($result, 0);
121
122    foreach($array as $varname => $value)
123        $formVars[$varname] = trim($value);
124
125    if (array_key_exists("passwort", $formVars))
126        unset($formVars['passwort']);
127
128    if (array_key_exists("secondPasswort", $formVars))

```



```

130     unset($formVars['secondPasswort']);

132     if ($debug)
        foreach($formVars as $x => $y)
134         echo "$x -> $y <br>\n";

136     ?>

138 <div align="center">
    <p class="head">Benutzerdaten</p>
140    <p class="text">
        In diesem Bereich k&ouml;nnen Sie die bei uns
142        gespeicherten pers&ouml;nlichen Daten &auml;ndern.
    </p>

144    <?php echo getErrors(); ?>

146    <form name="" action="<?php echo $projectPath . $menu .
148        '/userdata/validateUserData.php' ?>" method="post">
        <div align="center">
150            <p class="border">
<?php
152                if (!empty($PHPSESSID))
                    echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" .
154                    getValue('GET', 'PHPSESSID') . "\">\n";
?>

156                <input type="hidden" name="matrikelNr"
                    value="<?php echo $formVars['matrikel_nr'] ?>">
158                <input type="hidden" name="login" value="<?php echo $formVars['login'] ?>">
<table border="0" cellpadding="3" cellspacing="3">
160     <tr>
        <td align="right">
162             <p class="text">Vorname</p>
        </td>
164         <td><input type="text" name="vornameNeu" size="30"
            value="<?php echo getFormVars('vorname') ?>"></td>
166     </tr>
    <tr>
168         <td align="right">
            <p class="text">Name</p>
170         </td>
        <td><input type="text" name="nameNeu" size="30"
172            value="<?php echo getFormVars('nachname') ?>"></td>
    </tr>
174     <tr>
        <td align="right">
176             <p class="text">eMail-Adresse</p>
        </td>
178         <td><input type="text" name="mailNeu" size="30"
            value="<?php echo getFormVars('email') ?>"></td>
180     </tr>
    <tr>
182         <td align="right">
            <p class="text">Login</p>
184         </td>
        <td><input type="text" name="loginNeu" size="30"
186            value="<?php echo getFormVars('login') ?>"></td>
    </tr>

```

```

188         <tr>
189             <td colspan="2" align="right">
190                 <p class="error">Passwortfeld nur ausfüllen,
191                     wenn neues Passwort gesetzt werden soll !</p>
192             </td>
193         </tr>
194         <tr>
195             <td align="right">
196                 <p class="text">neues Passwort</p>
197             </td>
198             <td><input type="password" name="passwortNeu" size="30"
199                 value="<?php echo getFormVars('passwort') ?>"></td>
200         </tr>
201         <tr>
202             <td align="right">
203                 <p class="text">Passwort wiederholen</p>
204             </td>
205             <td><input type="password" name="secondPasswortNeu"
206                 size="30" value="<?php echo getFormVars('secondPasswort') ?>">
207             </td>
208         </tr>
209         <tr>
210             <td colspan="2" align="center">
211                 <p class="text">
212                     <input type="submit" name="" value="Werte übernehmen">&nbsp;
213                     <input type="reset" value="Zurücksetzen">
214                 </p>
215             </td>
216         </tr>
217     </table>
218     </p>
219 </div>
220 </form>
</div>

```

Listing E.21: user/home/userData2.php

```

<?php
2
3     /*
4     * user/userdata/userData2.php
5     */
6
7     // Verfügbare Variablen
8     // =====
9     //
10    // POST
11    // ----
12    // $_HTTP_GET_VARS['matrikelNr']
13    // $_HTTP_GET_VARS['login']
14    // $_HTTP_GET_VARS['vornameNeu']
15    // $_HTTP_GET_VARS['nameNeu']
16    // $_HTTP_GET_VARS['mailNeu']
17    // $_HTTP_GET_VARS['loginNeu']

```

```
18 // $HTTP_GET_VARS['passwortNeu']
19 // $HTTP_GET_VARS['secondPasswortNeu']
20 // $HTTP_GET_VARS['PHPSESSID']
21 //
22 // SESSION
23 // -----
24 // $HTTP_SESSION_VARS['projectPath']
25 // $HTTP_SESSION_VARS['absoluteProjectPath']
26 // $HTTP_SESSION_VARS['errors']
27 // $HTTP_SESSION_VARS['auth']
28 // $HTTP_SESSION_VARS['login']
29 // $HTTP_SESSION_VARS['menu']
30 // $HTTP_SESSION_VARS['formVars']
31
32 $debug = false;
33
34 // hole Session-Variable
35 if (isset($HTTP_GET_VARS['PHPSESSID']))
36     $PHPSESSID = $HTTP_GET_VARS['PHPSESSID'];
37 else
38     $PHPSESSID = $HTTP_POST_VARS['PHPSESSID'];
39
40 // hole vorhandene Session
41 session_start();
42 $PHPSESSID=session_id();
43
44 // falls Session verfuegbar, hole Session-Variablen
45 if (!empty($PHPSESSID))
46 {
47     if (isset($HTTP_SESSION_VARS["projectPath"]))
48         $projectPath = $HTTP_SESSION_VARS["projectPath"];
49
50     if (isset($HTTP_SESSION_VARS["projectPath"]))
51         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
52
53     if (isset($HTTP_SESSION_VARS["errors"]))
54         $errors = $HTTP_SESSION_VARS["errors"];
55
56     if (isset($HTTP_SESSION_VARS["formVars"]))
57         $formVars = $HTTP_SESSION_VARS["formVars"];
58
59     if (isset($HTTP_SESSION_VARS["menu"]))
60         $menu = $HTTP_SESSION_VARS["menu"];
61 }
62
63 require_once $absoluteProjectPath . "functions.php";
64
65 // Fehlerbehandlung
66 ob_start('ob_gzhandler');
67 set_error_handler('handleErrors');
68 $scriptName = $menu . "/userdata/userData2.php";
69
70 if ($debug)
71 {
72     echo "Session: " . session_id() . "<br>";
73     showSessionVar();
74     echo "Post:<br>";
75     showPostVar();
```

```

76     echo "Get:<br>";
77     showGetVar();
78 }
79
80 function getFormVars($fieldName)
81 {
82     global $formVars;
83
84     $value = "";
85
86     if (isset($formVars[$fieldName]))
87         $value = $formVars[$fieldName];
88
89     return $value;
90 }
91
92 function getError($fieldName)
93 {
94     global $errors;
95
96     $msg = "";
97
98     if (isset($errors[$fieldName]))
99         $msg = "<p class=\"error\">" . $errors[$fieldName] . "</p>";
100
101     return $msg;
102 }
103
104 function getErrors()
105 {
106     global $errors;
107
108     $msg = "";
109
110     if (isset($errors))
111     {
112         $msg .= "<p class=\"error\">\n";
113
114         foreach ($errors as $value)
115             $msg .= $value . "<br>\n";
116
117         $msg .= "</p>\n";
118     }
119
120     return $msg;
121 }
122 ?>
123
124 <div align="center">
125     <p class="head">Benutzerdaten</p>
126     <p class="text">
127         In diesem Bereich k&ouml;nnen Sie die bei uns gespeicherten
128         pers&ouml;nlichen Daten &auml;ndern.<br>
129     </p>
130
131     <?php echo getErrors(); ?>
132
133     <form name="" action="<?php echo $projectPath . $menu .

```

```

134         '/userdata/validateUserData.php' ?>" method="post">
135     <div align="center">
136         <p class="border">
137     <?php
138         if (!empty($PHPSESSID))
139             echo "<input type=\"hidden\" name=\"PHPSESSID\"
140                 value=\"\" . getValue('GET', 'PHPSESSID') . \">\n\";
141     ?>
142
143     <input type="hidden" name="matrikelNr"
144         value="<?php echo getFormVars('matrikelNr') ?>">
145     <input type="hidden" name="login" value="<?php echo $formVars['login'] ?>">
146     <table border="0" cellpadding="3" cellspacing="3">
147         <tr>
148             <td align="right">
149                 <p class="text">Vorname</p>
150             </td>
151             <td><input type="text" name="vornameNeu" size="30"
152                 value="<?php echo getFormVars('vornameNeu') ?>"></td>
153         </tr>
154         <tr>
155             <td align="right">
156                 <p class="text">Name</p>
157             </td>
158             <td><input type="text" name="nameNeu" size="30"
159                 value="<?php echo getFormVars('nameNeu') ?>"></td>
160         </tr>
161         <tr>
162             <td align="right">
163                 <p class="text">eMail-Adresse</p>
164             </td>
165             <td><input type="text" name="mailNeu" size="30"
166                 value="<?php echo getFormVars('mailNeu') ?>"></td>
167         </tr>
168         <tr>
169             <td align="right">
170                 <p class="text">Login</p>
171             </td>
172             <td><input type="text" name="loginNeu" size="30"
173                 value="<?php echo getFormVars('loginNeu') ?>"></td>
174         </tr>
175         <tr>
176             <td colspan="2" align="right">
177                 <p class="error">Passwortfeld nur ausfüllen,
178                     wenn neues Passwort gesetzt werden soll !</p>
179             </td>
180         </tr>
181         <tr>
182             <td align="right">
183                 <p class="text">neues Passwort</p>
184             </td>
185             <td><input type="password" name="passwortNeu" size="30"
186                 value="<?php echo getFormVars('passwortNeu') ?>"></td>
187         </tr>
188         <tr>
189             <td align="right">
190                 <p class="text">Passwort wiederholen</p>
191             </td>

```

```

192         <td><input type="password" name="secondPasswortNeu" size="30"
           value="<?php echo getFormVars('secondPasswortNeu') ?>"></td>
194     </tr>
195     <tr>
196         <td colspan="2" align="center">
197             <p class="text">
198                 <input type="submit" name="" value="Werte &uuml;bernehmen">&nbsp;
199                 <input type="reset" value="Zur&uuml;cksetzen">
200             </p>
201         </td>
202     </tr>
203 </table>
204 </p>
205 </div>
206 </form>
</div>

```

Listing E.22: user/home/validateUserData.php

```

<?php
2
    /*
4     * user/userdata/validateUserData.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // POST
11    // ----
12    // $HTTP_GET_VARS['matrikelNr']
13    // $HTTP_GET_VARS['login']
14    // $HTTP_GET_VARS['vornameNeu']
15    // $HTTP_GET_VARS['nameNeu']
16    // $HTTP_GET_VARS['mailNeu']
17    // $HTTP_GET_VARS['loginNeu']
18    // $HTTP_GET_VARS['passwortNeu']
19    // $HTTP_GET_VARS['secondPasswortNeu']
20    // $HTTP_GET_VARS['PHPSESSID']
21    //
22    // SESSION
23    // -----
24    // $HTTP_SESSION_VARS['projectPath']
25    // $HTTP_SESSION_VARS['absoluteProjectPath']
26    // $HTTP_SESSION_VARS['errors']
27    // $HTTP_SESSION_VARS['auth']
28    // $HTTP_SESSION_VARS['login']
29    // $HTTP_SESSION_VARS['menu']
30    // $HTTP_SESSION_VARS['formVars']
31
32    $debug = false;
33
34    // hole Session-Variable
35    if (isset($HTTP_GET_VARS["PHPSESSID"]))

```

```

36     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
    else
38         $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];

40     // hole vorhandene Session
    session_start();
42     $PHPSESSID=session_id();

44     // falls Session verfuegbar, hole Session-Variablen
    if (!empty($PHPSESSID))
46     {
        if (isset($HTTP_SESSION_VARS["projectPath"]))
48             $projectPath = $HTTP_SESSION_VARS["projectPath"];

        if (isset($HTTP_SESSION_VARS["projectPath"]))
50             $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];

52         if (isset($HTTP_SESSION_VARS["menu"]))
54             $menu = $HTTP_SESSION_VARS["menu"];
    }

56     require_once $absoluteProjectPath . "functions.php";

58     // Fehlerbehandlung
    ob_start('ob_gzhandler');
    set_error_handler('handleErrors');
62     $scriptName = $menu . "/userdata/validateUserData.php";

64     if ($debug)
    {
66         echo "Session: " . session_id() . "<br>";
        showSessionVar();
68         echo "Post:<br>";
        showPostVar();
70         echo "Get:<br>";
        showGetVar();
72     }

74     function setSessionVars()
    {
76         global $errors, $formVars, $login;
        global $HTTP_SESSION_VARS;

78         $HTTP_SESSION_VARS["errors"] = $errors;
        $HTTP_SESSION_VARS["formVars"] = $formVars;
80         $HTTP_SESSION_VARS["login"] = $login;
82     }

84     // Kontrolliert, ob ein Eintrag die maximale
    // Laenge ueberschreitet
    // $name: Bezeichnung des Feldes
    // $value: Inhalt des Feldes
    // $length: maximale Feldlaenge
    function checkFieldLength($name, $value, $length)
90     {
        global $errors, $debug;

92         if (strlen($value) > $length)

```

```

94     {
95         $errors[] = "Das Feld ". $name . " kann maximal " .
96             $length . " Zeichen aufnehmen.";
97         if ($debug)
98             echo "Das Feld ". $name . " kann maximal " .
99                 $length . " Zeichen aufnehmen.<br>\n";
100     }
101 }
102
103 // Variablen in der Session registrieren
104 if (!session_is_registered("errors"))
105     session_register("errors");
106
107 if (!session_is_registered("formVars"))
108     session_register("formVars");
109
110 if (!session_is_registered("login"))
111     session_register("login");
112
113 // Session-Variablen
114 $errors = array();
115 $login = $_HTTP_SESSION_VARS['login'];
116
117 foreach($_HTTP_POST_VARS as $varname => $value)
118     $formVars[$varname] = trim($value);
119
120 setSessionVars();
121
122 $validEmailExpr =
123     "^[0-9a-z~!#%&_-]([.]?[0-9a-z~!#%&_-])*" .
124     "@[0-9a-z~!#%&_-]([.]?[0-9a-z~!#%&_-])*" .
125     "\.[0-9a-z~!#%&_-]([0-9a-z~!#%&_-])*$";
126
127 if (!empty($_POST))
128 {
129     // ueberpruefung auf leere Felder
130     if (empty($formVars['nameNeu']) ||
131         empty($formVars['vornameNeu']) ||
132         empty($formVars['mailNeu']) ||
133         empty($formVars['loginNeu']))
134     {
135         $errors[] = "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.";
136         if ($debug)
137             echo "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.<br>";
138     }
139
140     // ueberpruefung auf korrekte email-Form
141     if (!eregi($validEmailExpr, $formVars['mailNeu']))
142     {
143         $errors[] = "Ihre eingegebene Email-Adresse hat keine g&uuml;ltige Form.";
144         if ($debug)
145             echo "Ihre eingegebene Email-Adresse hat keine g&uuml;ltige Form.<br>";
146     }
147
148     // ueberpruefung auf korrektes Passwort
149     if ($formVars['passwortNeu'] != $formVars['secondPasswortNeu'])
150     {

```



```

152     $errors[] = "Ihre beiden angegebenen Passw&ouml;rter " .
153         "stimmen nicht &uuml;berein.";
154     if ($debug)
155         echo "Ihre beiden angegebenen Passw&ouml;rter " .
156             "stimmen nicht &uuml;berein.<br>";
157     }
158
159     checkFieldLength ("Vorname", $formVars["vornameNeu"], 50);
160     checkFieldLength ("Name", $formVars["nameNeu"], 50);
161     checkFieldLength ("eMail", $formVars["mailNeu"], 80);
162     checkFieldLength ("Login", $formVars["loginNeu"], 80);
163     checkFieldLength ("Passwort", $formVars["passwortNeu"], 50);
164     checkFieldLength ("Passwort wiederholen", $formVars["secondPasswortNeu"], 50);
165
166     // Es sind Fehler vorhanden
167     if (count($errors) > 0)
168     {
169         //$_SESSION_VARS['errors'] = $errors;
170         setSessionVars();
171
172         if (!$debug)
173             header ("Location: " . $projectPath . "index_session.php?menu=" . $menu .
174                 "&content=userdata/userData2&PHPSESSID=" . session_id());
175         else
176             echo ("Location: " . $projectPath . "index_session.php?menu=" . $menu .
177                 "&content=userdata/userData2&PHPSESSID=" . session_id());
178         exit;
179     }
180     // alle Eingaben sind richtig erfolgt
181     else
182     {
183         // baue Verbindung zur Datenbank auf
184         $connection = getConnection();
185         if ($debug)
186             echo "Connection: " . $connection . "<br>\n";
187
188         $sql = "SELECT * FROM benutzer WHERE login='" . $formVars['loginNeu'] .
189             "' AND matrikel_nr!='" . $formVars['matrikelNr'] . "'";
190
191         $msg = "Bei der Ueberpruefung des Logins ist ein Fehler aufgetreten.<br>".
192             "Wenden Sie sich an Ihren Administrator.";
193
194         $result = executeStatement($connection, $sql, $msg);
195         if ($debug)
196             echo "Result: " . $result . "<br>\n";
197
198         // es darf diesen login noch nicht geben
199         if (pg_numrows($result) > 0)
200         {
201             $errors[] = "Dieser login wird bereits von einer " .
202                 "anderen Person verwendet.";
203             if ($debug)
204                 echo "Dieser login wird bereits von einer " .
205                     "anderen Person verwendet.<br>";
206         }
207
208         // Es sind Fehler vorhanden
209         if (count($errors) > 0)

```

```

210     {
211         //$_SESSION_VARS['errors'] = $errors;
212         setSessionVars();
213
214         if (!$debug)
215             header ("Location: " . $projectPath . "index_session.php?menu=" . $menu .
216                 "&content=userdata/userData2&PHPSESSID=" . session_id());
217         else
218             echo ("Location: " . $projectPath . "index_session.php?menu=" . $menu .
219                 "&content=userdata/userData2&PHPSESSID=" . session_id());
220         exit;
221     }
222     // Eintrag in die Datenbank
223     else
224     {
225         if ($debug)
226             echo "Es sind alle Eingaben erfolgt.<br>\n";
227
228         $cryptPass = createPass($formVars['passwortNeu']);
229
230         $sql = "UPDATE benutzer SET ";
231         $sql .= "vorname='" . $formVars['vornameNeu'] . "', ";
232         $sql .= "nachname='" . $formVars['nameNeu'] . "', ";
233         $sql .= "email='" . $formVars['mailNeu'] . "', ";
234         $sql .= "login='" . $formVars['loginNeu'] . "' ";
235
236         if (!empty($formVars['passwortNeu']))
237             $sql .= ", passwort='" . $cryptPass . "' ";
238
239         $sql .= "WHERE matrikel_nr='" . $formVars['matrikelNr'] . "' ";
240
241         if ($debug)
242             echo "SQL: " . $sql . "<br>\n";
243
244         $msg = "Bei Schreiben in die Datenbank ist ein Fehler aufgetreten.<br>".
245             "Wenden Sie sich an Ihren Administrator.";
246
247         $result = executeStatement($connection, $sql, $msg);
248
249         if ($debug)
250             echo $result;
251
252         // neuen Login setzen
253         $login = $formVars['loginNeu'];
254         setSessionVars();
255
256         if (!$debug)
257             header ("Location: " . $projectPath . "index_session.php?menu=" .
258                 $menu . "&content=userdata/updated&PHPSESSID=" . session_id());
259         else
260             echo ("<br>Location: " . $projectPath . "index_session.php?menu=" .
261                 $menu . "&content=userdata/updated&PHPSESSID=" . session_id());
262         exit;
263     }
264 }
265 }
266 ?>

```

Listing E.23: user/home/updated.php

```
<?php
2
3     /*
4      * user/userdata/updated.php
5      */
6
7     ?>
8
9     <div align="center">
10         <p></p>
11         <p class="head">Benutzerdaten</p>
12     </div>
13     <div align="center">
14         <br><br>
15         <p class="border">
16             <table>
17                 <tr>
18                     <td>
19                         <div align="center">
20                             <p class="text">
21                                 Ihre Daten wurden erfolgreich ge&uuml;ndert.
22                             </p>
23                         </div>
24                     </td>
25                 </tr>
26             </table>
27         </p>
28     </div>
```

E.3.3 Buchung vornehmen

Listing E.24: user/buchen/index.php

```
<?php
2
3     /*
4      * user/buchen/index.php
5      */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // POST
11    // ----
12    // $HTTP_POST_VARS['PHPSESSID']
13    // $HTTP_POST_VARS['day']
14    // $HTTP_POST_VARS['month']
15    // $HTTP_POST_VARS['year']
16    //
17    // GET
```

```

18 // ----
19 // $HTTP_GET_VARS['PHPSESSID']
20 // $HTTP_GET_VARS['menu']
21 // $HTTP_GET_VARS['content']
22 // $HTTP_GET_VARS['day']
23 // $HTTP_GET_VARS['month']
24 // $HTTP_GET_VARS['year']
25 // $HTTP_GET_VARS['time']
26 //
27 // SESSION
28 // -----
29 // $HTTP_SESSION_VARS['projectPath']
30 // $HTTP_SESSION_VARS['absoluteProjectPath']
31 // $HTTP_SESSION_VARS['errors']
32 // $HTTP_SESSION_VARS['auth']
33 // $HTTP_SESSION_VARS['login']
34 // $HTTP_SESSION_VARS['menu']
35
36 $debug = false;
37
38 // hole Session-Variable
39 if (isset($HTTP_GET_VARS["PHPSESSID"]))
40     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
41 else
42     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
43
44 // hole vorhandene Session
45 session_start();
46 $PHPSESSID=session_id();
47
48 // falls Session verfuegbar, hole Session-Variablen
49 if (!empty($PHPSESSID))
50 {
51     if (isset($HTTP_SESSION_VARS["projectPath"]))
52         $projectPath = $HTTP_SESSION_VARS["projectPath"];
53
54     if (isset($HTTP_SESSION_VARS["projectPath"]))
55         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
56
57     if (isset($HTTP_SESSION_VARS["errors"]))
58         $errors = $HTTP_SESSION_VARS["errors"];
59
60     if (isset($HTTP_SESSION_VARS["menu"]))
61         $menu = $HTTP_SESSION_VARS["menu"];
62 }
63
64 require_once $absoluteProjectPath . "functions.php";
65
66 // Fehlerbehandlung
67 ob_start('ob_gzhandler');
68 set_error_handler('handleErrors');
69 $scriptName = $menu . "/buchen/index.php";
70
71 if ($debug)
72 {
73     echo "Session: " . session_id() . "<br>";
74     showSessionVar();
75     echo "Post:<br>";

```

```
76     showPostVar();
77     echo "Get:<br>";
78     showGetVar();
79 }
80
81 function getErrors()
82 {
83     global $errors;
84     $msg = "";
85
86     if (isset($errors))
87     {
88         $msg .= "<p class=\"error\">\n";
89
90         foreach ($errors as $value)
91         {
92             $msg .= $value . "<br>\n";
93         }
94
95         $msg .= "</p>\n";
96     }
97
98     return $msg;
99 }
100
101 // feste Zeitfenster
102 $startzeit = array();
103 $startzeit[] = "00:00:00";
104 $startzeit[] = "03:00:00";
105 $startzeit[] = "06:00:00";
106 $startzeit[] = "09:00:00";
107 $startzeit[] = "12:00:00";
108 $startzeit[] = "15:00:00";
109 $startzeit[] = "18:00:00";
110 $startzeit[] = "21:00:00";
111
112 // Datum der Uebung
113 $requestDay = "";
114 $requestMonth = "";
115 $requestYear = "";
116
117 if (isset($_HTTP_GET_VARS['day']))
118     $requestDay = $_HTTP_GET_VARS['day'];
119 elseif (isset($_HTTP_POST_VARS['day']))
120     $requestDay = $_HTTP_POST_VARS['day'];
121
122 if (isset($_HTTP_GET_VARS['month']))
123     $requestMonth = $_HTTP_GET_VARS['month'];
124 elseif (isset($_HTTP_POST_VARS['month']))
125     $requestMonth = $_HTTP_POST_VARS['month'];
126
127 if (isset($_HTTP_GET_VARS['year']))
128     $requestYear = $_HTTP_GET_VARS['year'];
129 elseif (isset($_HTTP_POST_VARS['year']))
130     $requestYear = $_HTTP_POST_VARS['year'];
131
132 if (isset($_HTTP_GET_VARS['time']))
133     $requestTime = $_HTTP_GET_VARS['time'];
```

```

134     elseif (isset($HTTP_POST_VARS['time']))
135         $requestTime = $HTTP_POST_VARS['time'];
136
137     if ($debug)
138     {
139         echo "day: " . $requestDay . "<br>";
140         echo "month: " . $requestMonth . "<br>";
141         echo "year: " . $requestYear . "<br>";
142         echo "time: " . $requestTime . "<br>";
143     }
144
145     // baue Verbindung zur Datenbank auf
146     $connection = getConnection();
147
148     // aktualisiere Zeitfenster
149     if (!empty($requestDay) && !empty($requestMonth) && !empty($requestYear))
150     {
151         // formatiere Datum
152         $datum = $requestMonth . "/" . $requestDay . "/" . $requestYear;
153
154         // frage nach belegten Zeiten
155         $sql = "SELECT startzeit FROM buchungen WHERE datum='" . $datum . "'";
156
157         $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
158             "Wenden Sie sich an Ihren Administrator.";
159
160         $result = executeStatement($connection, $sql, $msg);
161
162         if (pg_numrows($result) > 0)
163         {
164             $belegteStartzeit = array();
165
166             // hole bereits gebuchte Zeiten
167             for ($i = 0; $i < pg_numrows($result); $i++)
168             {
169                 $tmp = fetchArray($result, $i);
170                 $belegteStartzeit[$i] = $tmp[0];
171             }
172
173             // loesche vergangene Zeiten
174             // aus den Zeitfenstern
175             foreach ($belegteStartzeit as $value)
176             {
177                 $pos = array_search($value, $startzeit);
178                 unset( $startzeit[$pos] );
179             }
180         }
181     }
182
183     // frage nach ännst-ömglicher Startzeit
184     $sql = "SELECT " .
185         "date_part('hour', now()-max(uebungen.vorlauf) ) AS hour, " .
186         "date_part('minute', now()-max(uebungen.vorlauf) ) AS minute, " .
187         "date_part('second', now()-max(uebungen.vorlauf) ) AS second, " .
188         "date_part('day', now()-max(uebungen.vorlauf) ) AS day, " .
189         "date_part('month', now()-max(uebungen.vorlauf) ) AS month, " .
190         "date_part('year', now()-max(uebungen.vorlauf) ) AS year";

```

```

192 $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
    "Wenden Sie sich an Ihren Administrator.";
194
196 $result = executeStatement($connection, $sql, $msg);
198
196 $time = fetchArray($result, 0);
198
200 $localdate = mktime(0, 0, 0, $time['month'], $time['day'], $time['year']);
200 $localtime = mktime($time['hour'], $time['minute'],
    $time['second'], $time['month'], $time['day'], $time['year']);
202 $bookdate = mktime( 0, 0, 0, $requestMonth, $requestDay, $requestYear);

204 // liegt der Buchungstag in der Vergangenheit,
    // dann sind keine Zeitfenster verfuegbar
206 if ($bookdate < $localdate)
    {
208     if ($debug)
        echo "Vergangenheit <br>";
210
212     // loesche vergangene Zeiten
        // aus den Zeitfenstern
        foreach ($startzeit as $value)
214     {
            $pos = array_search($value, $startzeit);
216             unset( $startzeit[$pos] );
        }
218     }

220 // zeige keine vergangenen Zeitfenster fuer den
    // Zeitpunkt heute
222 elseif ($bookdate == $localdate)
    {
224     if ($debug)
        echo "heute <br>";
226
228     // loesche vergangene Zeiten
        // aus den Zeitfenstern
        foreach ($startzeit as $value)
230     {
            if ( $localtime >= mktime( cutTime($value), 0, 0,
232                $requestMonth, $requestDay, $requestYear) )
            {
234                $pos = array_search($value, $startzeit);
                unset( $startzeit[$pos] );
236            }
        }
238     }

240 echo "<p class=\"head\">Buchung vornehmen</p>\n";
    echo "<p class=\"text\">\n";
242 echo "In diesem Bereich k&ouml;nnen Sie Ihre &Uuml;bungs-Buchungen vornehmen.\n";
    echo "</p>\n";
244
246 echo getErrors();

248 echo "<p></p>\n";
    echo "<table class=\"vulab\" width=100%>\n";
    echo "<tr><td width=\"40%\" align=\"center\">\n";

```

```

250     echo "<p>&nbsp;</p>\n";
251     echo "<p class=\"green\">\n";
252     echo "W&auml;hlen Sie Ihren &Uuml;bungstag\n";
253     echo "</p>\n";
254
255     //////////////////////////////////////
256     // Calender
257
258     echo "<!-- Calender -->\n";
259
260     include $absoluteProjectPath . $menu . "/buchen/calender.php";
261
262     // Fehlerbehandlung
263     $scriptName = $menu . "/buchen/index.php";
264
265     //////////////////////////////////////
266
267     echo "</td><td width=\"60%\" align=\"center\" valign=\"top\">\n";
268
269     if (!empty($requestDay) && !empty($requestMonth) && !empty($requestYear))
270     {
271         echo "<!-- Zeit-Auswahl -->\n";
272
273         echo "<form name=\"\" action=\"\" . $projectPath .
274             \"index_session.php?menu=\" . $menu .
275             \"&content=buchen/buchen2\" method=\"post\">\n";
276
277         if (!empty($PHPSESSID))
278             echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" .
279                 $PHPSESSID . \"\">\n";
280         if (!empty($requestDay))
281             echo "<input type=\"hidden\" name=\"day\" value=\"\" .
282                 $requestDay . \"\">\n";
283         if (!empty($requestMonth))
284             echo "<input type=\"hidden\" name=\"month\" value=\"\" .
285                 $requestMonth . \"\">\n";
286         if (!empty($requestYear))
287             echo "<input type=\"hidden\" name=\"year\" value=\"\" .
288                 $requestYear . \"\">\n";
289         if (!empty($requestTime))
290             echo "<input type=\"hidden\" name=\"time\" value=\"\" .
291                 $requestTime . \"\">\n";
292
293         echo "<p>&nbsp;</p>\n";
294         echo "<p class=\"text\">Angefragtes Datum: \" . $requestDay .
295             \" . \" . $requestMonth . \" . \" . $requestYear . \"</p>\n";
296         echo "<p></p>\n";
297
298         echo "<p class=\"green\">\n";
299         echo "W&auml;hlen Sie die gew&uuml;nschte Startzeit der &Uuml;bung\n";
300         echo "</p>\n";
301
302         // Liste mit Radio-Buttons
303         echo "<p class=\"text\">";
304
305         foreach ($startzeit as $value)
306         {
307             if (!empty($requestTime))

```



```

308     {
          if ($requestTime == cutTime($value))
            echo "<input type=\"radio\" name=\""time\" value=\"$" .
              cutTime($value) . "\" checked />&nbsp;&nbsp;&nbsp;" .
              cutTime($value) . " Uhr<br>\n";
          else
            echo "<input type=\"radio\" name=\""time\" value=\"$" .
              cutTime($value) . "\"/>&nbsp;&nbsp;&nbsp;" .
              cutTime($value) . " Uhr<br>\n";
        }
      else
        echo "<input type=\"radio\" name=\""time\" value=\"$" .
          cutTime($value) . "\"/>&nbsp;&nbsp;&nbsp;" .
          cutTime($value) . " Uhr<br>\n";
    }

if (sizeof($startzeit) == 0)
{
  echo "<p class=\"error\">";
  echo "!!! keine Zeitfenster überfgbar !!!";
  echo "</p>\n";
}

echo "</p>\n";

echo "<input type=\"submit\" name=\"\" value=\"weiter\"/>\n";
echo "</form>\n";
}

echo "</td></tr>\n";
echo "</table>\n";
?>
```

Listing E.25: user/buchen/calender.php

```

1 <?php
2
3     /*
4     * user/buchen/calender.php
5     */
6
7     // hole Session-Variable
8     if (isset($_HTTP_GET_VARS["PHPSESSID"]))
9         $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
10    else
11        $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];
12
13    // hole vorhandene Session
14    session_start();
15    $PHPSESSID=session_id();
16
17    // falls Session verfuegbar, hole Session-Variablen
18    if (!empty($PHPSESSID))

```

```

{
20     if (isset($_HTTP_SESSION_VARS["projectPath"]))
        $projectPath = $_HTTP_SESSION_VARS["projectPath"];
22
        if (isset($_HTTP_SESSION_VARS["projectPath"]))
24            $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];

26        if (isset($_HTTP_SESSION_VARS["menu"]))
            $menu = $_HTTP_SESSION_VARS["menu"];
28    }

30    require_once $absoluteProjectPath . "functions.php";

32    // Fehlerbehandlung
    ob_start('ob_gzhandler');
34    set_error_handler('handleErrors');
    $scriptName = $menu . "/buchen/calender.php";
36

    /* Language config of weekdays
38
    $cfgMonday = "Montag";
40    $cfgTuesday = "Dienstag";
    $cfgWednesday = "Mittwoch";
42    $cfgThursday = "Donnerstag";
    $cfgFriday = "Freitag";
44    $cfgSaturday = "Samstag";
    $cfgSunday = "Sonntag";*/

46
    $cfgMonday = "Mo";
48    $cfgTuesday = "Di";
    $cfgWednesday = "Mi";
50    $cfgThursday = "Do";
    $cfgFriday = "Fr";
52    $cfgSaturday = "Sa";
    $cfgSunday = "So";
54

    /* Language config of monthnames*/
56    $cfgMonthname = array("", "Januar",
        "Februar",
58        "M&auml;rzt",
        "April",
60        "Mai",
        "Juni",
62        "Juli",
        "August",
64        "September",
        "Oktober",
66        "November",
        "Dezember");

68
function WriteDaydetailLink($intLinkDay, $intLinkMonth, $intLinkYear,
    $strValue, $strLinkClass)
{
72    global $projectPath, $PHPSESSID, $menu;

74    $strToday = "";

76    if (date("d.m.Y") == date("d.m.Y",

```

```

        mktime(0,0,0,$intLinkMonth,$intLinkDay,$intLinkYear) ) )
78     $strToday = " id=\"today\"";

80     echo "<td class=\"\$strLinkClass\"\$strToday>";

82     if ($strLinkClass == "sun")
        echo "<a class=\"rot\" href=\"\" . $projectPath . "index_session.php?menu=" .
84         $menu . "&content=buchen/index&PHPSESSID=" . $PHPSESSID .
            "&day=$intLinkDay&month=$intLinkMonth&year=$intLinkYear\" " .
86         "class=\"\$strLinkClass\">";
    else
88         echo "<a class=\"nocss\" href=\"\" . $projectPath . "index_session.php?menu=" .
            $menu . "&content=buchen/index&PHPSESSID=" . $PHPSESSID .
90         "&day=$intLinkDay&month=$intLinkMonth&year=$intLinkYear\" " .
            "class=\"\$strLinkClass\">";
92     echo $strValue."</a></td>";
}

94 function DayAdd($intNumOfDays)
{
96     global $day, $month, $year;
    return date("j", mktime(0,0,0,$month,$day + $intNumOfDays,$year));
98 }

100 $daycount=01;
$day=01;          /*First day of each month  date("d");*/
102
103 if (isset($_HTTP_GET_VARS['month']))
104     $month = $_HTTP_GET_VARS['month'];
elseif (isset($_HTTP_POST_VARS['month']))
106     $month = $_HTTP_POST_VARS['month'];

108 if (isset($_HTTP_GET_VARS['year']))
    $year = $_HTTP_GET_VARS['year'];
elseif (isset($_HTTP_POST_VARS['year']))
110     $year = $_HTTP_POST_VARS['year'];
112

113 if (!isset($month) && !isset($year))
114 {
    $month=date("n");    /*This month */
116     $year=date("Y");
}

118 $previousyear = $year;
120 $nextyear = $year;

122 $nextmonth = $month + 1;
if ($nextmonth > 12)
124 {
    $nextmonth = 1;
126     $nextyear = $year + 1;
}

128 $previousmonth = $month - 1;
if ($previousmonth < 1)
130 {
    $previousmonth = 12;
132     $previousyear = $year - 1;
134 }

```

```

136 $off=0;

138 while (checkdate($month,$daycount,$year)):
    $daycount++;
140 endwhile;

142 $monthname = $cfgMonthname[$month];

144 /* Create a table with days of the week headers */
echo "<n<table border=\"0\" cellspacing=\"0\" cellpadding=\"2\"><n<tr>";
146 echo "<td colspan=\"7\" class=\"monthname\"><n\";
echo "<table border=\"0\" width=\"100%\" cellspacing=\"0\" cellpadding=\"0\"><n\";
148 echo "
    <tr><td class=\"monthname\" align=\"left\"><a href=\"\" . $projectPath .
        "index_session.php?menu=" . $menu . "&content=buchen/index&PHPSESSID=" .
150 $PHPSESSID . "&month=$previousmonth&year=$previousyear\">\" .
        "<img border=\"0\" src=\"\" . $projectPath . $menu .
152 "/buchen/images/previous.gif\" width=\"15\" height=\"19\"></td><n\";
echo "
    <td class=\"monthname\" align=\"center\" nowrap>$monthname $year</td><n\";
154 echo "
    <td class=\"monthname\" align=\"right\"><a href=\"\" . $projectPath .
        "index_session.php?menu=" . $menu . "&content=buchen/index&PHPSESSID=" .
156 $PHPSESSID . "&month=$nextmonth&year=$nextyear\"><img border=\"0\" src=\"\" .
        $projectPath . $menu . "/buchen/images/next.gif\" width=\"15\" height=\"19\">\" .
158 "</td></tr><n\";
echo "</table><n</td><n</tr>";
160 echo "<tr><td class=\"weekdayname\">$cfgMonday</td>\" .
    "<td class=\"weekdayname\">$cfgTuesday</td><n\";
162 echo "
    <td class=\"weekdayname\">$cfgWednesday</td>\" .
    "<td class=\"weekdayname\">$cfgThursday</td><n\";
164 echo "
    <td class=\"weekdayname\">$cfgFriday</td>\" .
    "<td class=\"weekdayname\">$cfgSaturday</td><n\";
166 echo "
    <td class=\"weekdayname\">$cfgSunday</td><n\";
echo "</tr>";

168
/* Start the table data and make sure the number of days does not exceed
170 the total for the month - $date will always be one more than the total
number of days in the month */
172 echo "<n<tr>";
while ($day<$daycount):
174
    if ($day == '01' and date('l', mktime(0,0,0,$month,$day,$year)) == 'Monday')
176
    {
        WriteDaydetailLink($day, $month, $year, $day, "weekday");
178        $off= '01';
    }

180 elseif ($day == '01' and date('l', mktime(0,0,0,$month,$day,$year)) == 'Tuesday')
    {
182        WriteDaydetailLink(DayAdd(-1), $previousmonth,
            $previousyear, DayAdd(-1), "notthismonth");
184
        WriteDaydetailLink($day, $month, $year, $day, "weekday");
186        $off= '02';
    }

188 elseif ($day == '01' and date('l', mktime(0,0,0,$month,$day,$year)) == 'Wednesday')
    {
190        for ($i=1; $i<=2; $i++)
        {
192            WriteDaydetailLink(DayAdd(-3 + $i), $previousmonth,

```

```

194         $previousyear, DayAdd(-3 + $i), "notthismonth");
196     }
198     WriteDaydetailLink($day, $month, $year, $day, "weekday");
199     $off= '03';
200 }
201 elseif ($day == '01' and date('l', mktime(0,0,0,$month,$day,$year)) == 'Thursday')
202 {
203     for ($i=1; $i<=3; $i++)
204     {
205         WriteDaydetailLink(DayAdd(-4 + $i), $previousmonth,
206             $previousyear, DayAdd(-4 + $i), "notthismonth");
207     }
208     WriteDaydetailLink($day, $month, $year, $day, "weekday");
209     $off= '04';
210 }
211 elseif ($day == '01' and date('l', mktime(0,0,0,$month,$day,$year)) == 'Friday')
212 {
213     for ($i=1; $i<=4; $i++)
214     {
215         WriteDaydetailLink(DayAdd(-5 + $i), $previousmonth,
216             $previousyear, DayAdd(-5 + $i), "notthismonth");
217     }
218     WriteDaydetailLink($day, $month, $year, $day, "weekday");
219     $off= '05';
220 }
221 elseif ($day == '01' and date('l', mktime(0,0,0,$month,$day,$year)) == 'Saturday')
222 {
223     for ($i=1; $i<=5; $i++)
224     {
225         WriteDaydetailLink(DayAdd(-6 + $i), $previousmonth,
226             $previousyear, DayAdd(-6 + $i), "notthismonth");
227     }
228     WriteDaydetailLink($day, $month, $year, $day, "sat");
229     $off= '06';
230 }
231 elseif ($day == '01' and date('l', mktime(0,0,0,$month,$day,$year)) == 'Sunday')
232 {
233     for ($i=1; $i<=6; $i++)
234     {
235         WriteDaydetailLink(DayAdd(-7 + $i), $previousmonth,
236             $previousyear, DayAdd(-7 + $i), "notthismonth");
237     }
238     WriteDaydetailLink($day, $month, $year, $day, "sun");
239     $off = '07';
240 }
241 else
242 {
243     $classname = "weekday";
244     if (date('l', mktime(0,0,0,$month,$day,$year)) == 'Sunday')
245     { $classname = "sun"; }
246
247     if (date('l', mktime(0,0,0,$month,$day,$year)) == 'Saturday')
248     { $classname = "sat"; }
249 }
250

```

```

        WriteDaydetailLink($day, $month, $year, $day, $classname);
252 }
    $day++;
254 $off++;

256 if ($off>7)
    {
        echo "</tr>\n<tr>";
258     $off='01';
    }
260 endwhile;

262 $day-- ;
    $i = 1 ;
264 if ($off != 1)
    {
266     while ($off < 8)
        {
            WriteDaydetailLink(DayAdd($i), $nextmonth, $nextyear, DayAdd($i), "notthismonth");
268             $off++ ;
                $i++ ;
270         }
    }
272 echo "</table>";

274 // Form with elements to select a month and a year
?>
276 <form action='<?php echo $projectPath . "index_session.php?menu=" .
    $menu . "&content=buchen/index&PHPSESSID=" . $PHPSESSID ?>' method=post>
278 <p><select size="1" name="month">
    <?
280         for ($i=1; $i<=12; $i++)
            {
282
                echo "<option value=".$i;
284                 if (date("n") == $i) echo " selected ";
                echo ">".$cfgMonthname[$i]."</option>\n";
286             }
    ?>
288 </select><br>
    <select size="1" name="year">
290     <?
        $strSelected = "";
292     for ($i=1990; $i<=2010; $i++)
        {
294         echo "<option value=".$i;
            if ($i == date("Y")) echo " selected";
296         echo ">".$i."</option>";
        }
    ?>
298 </select> <input type="submit" value="GO"></p>
300 </form>

```

Listing E.26: user/buchen/buchen2.php

```
<?php
2
3    /*
4    * user/buchen/buchen2.php
5    */
6
7    // Verfuegbare Variablen
8    // =====
9    //
10   // POST
11   // ----
12   // $HTTP_POST_VARS['PHPSESSID']
13   // $HTTP_POST_VARS['day']
14   // $HTTP_POST_VARS['month']
15   // $HTTP_POST_VARS['year']
16   // $HTTP_POST_VARS['time']
17   //
18   // GET
19   // ----
20   // $HTTP_GET_VARS['menu']
21   // $HTTP_GET_VARS['content']
22   //
23   // SESSION
24   // -----
25   // $HTTP_SESSION_VARS['projectPath']
26   // $HTTP_SESSION_VARS['absoluteProjectPath']
27   // $HTTP_SESSION_VARS['errors']
28   // $HTTP_SESSION_VARS['auth']
29   // $HTTP_SESSION_VARS['login']
30   // $HTTP_SESSION_VARS['menu']
31
32   $debug = false;
33
34   // hole Session-Variable
35   if (isset($HTTP_GET_VARS["PHPSESSID"]))
36       $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
37   else
38       $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
39
40   // hole vorhandene Session
41   session_start();
42   $PHPSESSID=session_id();
43
44   // falls Session verfuegbar, hole Session-Variablen
45   if (!empty($PHPSESSID))
46   {
47       if (isset($HTTP_SESSION_VARS["projectPath"]))
48           $projectPath = $HTTP_SESSION_VARS["projectPath"];
49
50       if (isset($HTTP_SESSION_VARS["projectPath"]))
51           $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
52
53       if (isset($HTTP_SESSION_VARS["errors"]))
54           $errors = $HTTP_SESSION_VARS["errors"];
55
56       if (isset($HTTP_SESSION_VARS["menu"]))
57           $menu = $HTTP_SESSION_VARS["menu"];
```

```

58     }

60     require_once $absoluteProjectPath . "functions.php";

62     // Fehlerbehandlung
63     ob_start('ob_gzhandler');
64     set_error_handler('handleErrors');
65     $scriptName = $menu . "/buchen/index.php";
66
67     if ($debug)
68     {
69         echo "Session: " . session_id() . "<br>";
70         showSessionVar();
71         echo "Post:<br>";
72         showPostVar();
73         echo "Get:<br>";
74         showGetVar();
75     }
76
77     function getErrors()
78     {
79         global $errors;
80         $msg = "";
81
82         if (isset($errors))
83         {
84             $msg .= "<p class=\"error\">\n";
85
86             foreach ($errors as $value)
87             {
88                 $msg .= $value . "<br>\n";
89             }
90
91             $msg .= "</p>\n";
92         }
93
94         return $msg;
95     }
96
97     if (isset($_HTTP_GET_VARS['day']))
98         $requestDay = $_HTTP_GET_VARS['day'];
99     elseif (isset($_HTTP_POST_VARS['day']))
100         $requestDay = $_HTTP_POST_VARS['day'];
101
102     if (isset($_HTTP_GET_VARS['month']))
103         $requestMonth = $_HTTP_GET_VARS['month'];
104     elseif (isset($_HTTP_POST_VARS['month']))
105         $requestMonth = $_HTTP_POST_VARS['month'];
106
107     if (isset($_HTTP_GET_VARS['year']))
108         $requestYear = $_HTTP_GET_VARS['year'];
109     elseif (isset($_HTTP_POST_VARS['year']))
110         $requestYear = $_HTTP_POST_VARS['year'];
111
112     if (isset($_HTTP_GET_VARS['time']))
113         $requestTime = $_HTTP_GET_VARS['time'];
114     elseif (isset($_HTTP_POST_VARS['time']))
115         $requestTime = $_HTTP_POST_VARS['time'];

```



```

116     if ($debug)
117     {
118         echo "day: " . $requestDay . "<br>";
119         echo "month: " . $requestMonth . "<br>";
120         echo "year: " . $requestYear . "<br>";
121         echo "time: " . $requestTime . "<br>";
122     }
123
124     function setSessionVars()
125     {
126         global $errors;
127         global $HTTP_SESSION_VARS;
128
129         $HTTP_SESSION_VARS["errors"] = $errors;
130     }
131
132     // Variablen in der Session registrieren
133     if (!session_is_registered("errors"))
134         session_register("errors");
135
136     // Session-Variable
137     $errors = array();
138
139     //////////////////////////////////////
140     // Pruefen auf Fehler
141
142     if (!empty($HTTP_POST_VARS))
143     {
144         // pruefen auf leere Eingabe
145         if (empty($requestDay) || empty($requestMonth) ||
146             empty($requestYear) || empty($requestTime))
147         {
148             $errors[] = "Es wurde keine Startzeit gew&auml;hlt.";
149             if ($debug)
150                 echo "Es wurde keine Startzeit gew&auml;hlt.<br>";
151         }
152
153         // Es sind Fehler vorhanden
154         if (count($errors) > 0)
155         {
156             setSessionVars();
157
158             $header = "Location: " . $projectPath . "index_session.php?" .
159                 "menu=" . $menu . "&content=buchen/index&PHPSESSID=" . session_id() .
160                 "&day=" . $requestDay .
161                 "&month=" . $requestMonth .
162                 "&year=" . $requestYear;
163
164             if (isset($requestTime))
165                 $header = $header . "&time=" . $requestTime;
166
167             if (!$debug)
168                 header ($header);
169             else
170                 echo ($header);
171             exit;
172         }
173     }

```

```

174 ///////////////////////////////////////////////////////////////////
175 // pruefen, ob die gewaehlte Zeit in der Vergangenheit liegt
176
177 // SQL-Abfrage zur Startzeit - jetzt
178
180 $sql = "SELECT age( now(), ' " . $requestMonth .
181        "/" . $requestDay . "/" . $requestYear . " " .
182        $requestTime . "'::timestamp - max(vorlauf)) as alter from uebungen";
183
184 if ($debug)
185 {
186     echo "Jetzt - Startzeit<br>";
187     echo "SQL: " . $sql . "<br><br>\n";
188 }
189
190 $msg = "Beim der Ueberpruefung, ob die Startzeit in der Zukunft liegt, " .
191        "ist ein Fehler aufgetreten.<br>" .
192        "Wenden Sie sich an Ihren Administrator.";
193
194 $result = executeStatement($connection, $sql, $msg);
195
196 if ($debug)
197     echo "result: " . $result . "<br>";
198
199 $row = fetchArray($result, 0);
200 $alter = $row['alter'];
201
202 if ($debug)
203 {
204     echo "Alter: " . $alter . "<br>";
205     echo "Negatives Alter liegt in der Zukunft.<br>";
206 }
207
208 $pos = strpos($alter, '-');
209
210 if ($debug)
211     echo "pos: " . $pos . "<br>";
212
213 if (!is_numeric($pos))
214 {
215     $errors[] = "Zur gew&auml;hlten Startzeit ist " .
216               "keine &Uuml;bung mehr m&ouml;glich.";
217     if ($debug)
218         echo "Zur gew&auml;hlten Startzeit ist " .
219               "keine &Uuml;bung mehr m&ouml;glich.<br>";
220 }
221
222 // Es sind Fehler vorhanden
223 if (count($errors) > 0)
224 {
225     setSessionVars();
226
227     $header = "Location: " . $projectPath . "index_session.php?" .
228               "menu=" . $menu . "&content=buchen/index&PHPSESSID=" . session_id() .
229               "&day=" . $requestDay .
230               "&month=" . $requestMonth .
231               "&year=" . $requestYear;

```

```

232         if (isset($requestTime))
233             $header = $header . "&time=" . $requestTime;
234
235         if (!$debug)
236             header ($header);
237         else
238             echo ($header);
239         exit;
240     }
241     else
242     {
243         setSessionVars();
244
245         $header = "Location: " . $projectPath . "index_session.php?" .
246             "menu=" . $menu . "&content=buchen/buchen3&PHPSESSID=" . session_id() .
247             "&day=" . $requestDay .
248             "&month=" . $requestMonth .
249             "&year=" . $requestYear;
250
251         if (isset($requestTime))
252             $header = $header . "&time=" . $requestTime;
253
254         if (!$debug)
255             header ($header);
256         else
257             echo ($header);
258         exit;
259     }
260 }
261 }
262 ?>

```

Listing E.27: user/buchen/buchen3.php

```

<?php
2
3     /*
4     * user/buchen/buchen2.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // POST
11    // ----
12    // $HTTP_POST_VARS['PHPSESSID']
13    // $HTTP_POST_VARS['day']
14    // $HTTP_POST_VARS['month']
15    // $HTTP_POST_VARS['year']
16    // $HTTP_POST_VARS['time']
17    //
18    // GET
19    // ----
20    // $HTTP_GET_VARS['menu']

```

```

22 // $HTTP_GET_VARS['content']
23 //
24 // SESSION
25 // -----
26 // $HTTP_SESSION_VARS['projectPath']
27 // $HTTP_SESSION_VARS['absoluteProjectPath']
28 // $HTTP_SESSION_VARS['errors']
29 // $HTTP_SESSION_VARS['auth']
30 // $HTTP_SESSION_VARS['login']
31 // $HTTP_SESSION_VARS['menu']
32
33 $debug = false;
34
35 // hole Session-Variable
36 if (isset($HTTP_GET_VARS["PHPSESSID"]))
37     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
38 else
39     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
40
41 // hole vorhandene Session
42 session_start();
43 $PHPSESSID=session_id();
44
45 // falls Session verfuegbar, hole Session-Variablen
46 if (!empty($PHPSESSID))
47 {
48     if (isset($HTTP_SESSION_VARS["projectPath"]))
49         $projectPath = $HTTP_SESSION_VARS["projectPath"];
50
51     if (isset($HTTP_SESSION_VARS["absoluteProjectPath"]))
52         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
53
54     if (isset($HTTP_SESSION_VARS["errors"]))
55         $errors = $HTTP_SESSION_VARS["errors"];
56
57     if (isset($HTTP_SESSION_VARS["menu"]))
58         $menu = $HTTP_SESSION_VARS["menu"];
59 }
60
61 require_once $absoluteProjectPath . "functions.php";
62
63 // Fehlerbehandlung
64 ob_start('ob_gzhandler');
65 set_error_handler('handleErrors');
66 $scriptName = $menu . "/buchen/index.php";
67
68 if ($debug)
69 {
70     echo "Session: " . session_id() . "<br>";
71     showSessionVar();
72     echo "Post:<br>";
73     showPostVar();
74     echo "Get:<br>";
75     showGetVar();
76 }
77
78 function getErrors()
79 {

```

```

global $errors;
80  $msg = "";

82  if (isset($errors))
  {
84      $msg .= "<p class=\"error\">\n";

86      foreach ($errors as $value)
      {
88          $msg .= $value . "<br>\n";
      }

90      $msg .= "</p>\n";
92  }

94  return $msg;
  }

96  if (isset($_HTTP_GET_VARS['day']))
98      $requestDay = $_HTTP_GET_VARS['day'];
elseif (isset($_HTTP_POST_VARS['day']))
100     $requestDay = $_HTTP_POST_VARS['day'];

102  if (isset($_HTTP_GET_VARS['month']))
      $requestMonth = $_HTTP_GET_VARS['month'];
elseif (isset($_HTTP_POST_VARS['month']))
104     $requestMonth = $_HTTP_POST_VARS['month'];

106  if (isset($_HTTP_GET_VARS['year']))
108     $requestYear = $_HTTP_GET_VARS['year'];
elseif (isset($_HTTP_POST_VARS['year']))
110     $requestYear = $_HTTP_POST_VARS['year'];

112  if (isset($_HTTP_GET_VARS['time']))
      $requestTime = $_HTTP_GET_VARS['time'];
elseif (isset($_HTTP_POST_VARS['time']))
114     $requestTime = $_HTTP_POST_VARS['time'];

116  if ($debug)
118  {
      echo "day: " . $requestDay . "<br>";
120     echo "month: " . $requestMonth . "<br>";
      echo "year: " . $requestYear . "<br>";
122     echo "time: " . $requestTime . "<br>";
  }

124  function setSessionVars()
126  {
      global $errors;
128      global $_HTTP_SESSION_VARS;

130      $_HTTP_SESSION_VARS["errors"] = $errors;
  }

132  // Variablen in der Session registrieren
134  if (!session_is_registered("errors"))
      session_register("errors");
136

```

```

138 // Session-Variable
$errors = array();

140 ///////////////////////////////////////////////////////////////////
141 // Pruefen auf Fehler
142
143 if (!empty($HTTP_POST_VARS))
144 {
145     // pruefen auf leere Eingabe
146     if (empty($requestDay) || empty($requestMonth) ||
147         empty($requestYear) || empty($requestTime))
148     {
149         $errors[] = "Es wurde keine Startzeit gew&auml;hlt.";
150         if ($debug)
151             echo "Es wurde keine Startzeit gew&auml;hlt.<br>";
152     }
153
154     // Es sind Fehler vorhanden
155     if (count($errors) > 0)
156     {
157         setSessionVars();
158
159         $header = "Location: " . $projectPath . "index_session.php?" .
160             "menu=" . $menu . "&content=buchen/index&PHPSESSID=" . session_id() .
161             "&day=" . $requestDay .
162             "&month=" . $requestMonth .
163             "&year=" . $requestYear;
164
165         if (isset($requestTime))
166             $header = $header . "&time=" . $requestTime;
167
168         if (!$debug)
169             header ($header);
170         else
171             echo ($header);
172         exit;
173     }
174
175     ///////////////////////////////////////////////////////////////////
176     // pruefen, ob die gewaehlte Zeit in der Vergangenheit liegt
177
178     // SQL-Abfrage zur Startzeit - jetzt
179
180     $sql = "SELECT age( now(), '" . $requestMonth .
181         "/" . $requestDay . "/" . $requestYear . " " .
182         $requestTime . "'::timestamp - max(vorlauf)) as alter from uebungen";
183
184     if ($debug)
185     {
186         echo "Jetzt - Startzeit<br>";
187         echo "SQL: " . $sql . "<br><br>\n";
188     }
189
190     $msg = "Beim der Ueberpruefung, ob die Startzeit in der Zukunft liegt, " .
191         "ist ein Fehler aufgetreten.<br>" .
192         "Wenden Sie sich an Ihren Administrator.";
193
194     $result = executeStatement($connection, $sql, $msg);

```

```
196         if ($debug)
197             echo "result: " . $result . "<br>";
198
199         $row = fetchArray($result, 0);
200         $alter = $row['alter'];
201
202         if ($debug)
203         {
204             echo "Alter: " . $alter . "<br>";
205             echo "Negatives Alter liegt in der Zukunft.<br>";
206         }
207
208         $pos = strpos($alter, '-');
209
210         if ($debug)
211             echo "pos: " . $pos . "<br>";
212
213         if (!is_numeric($pos))
214         {
215             $errors[] = "Zur gew&auml;hlten Startzeit ist " .
216                 "keine &Uuml;bung mehr m&ouml;glich.";
217             if ($debug)
218                 echo "Zur gew&auml;hlten Startzeit ist " .
219                     "keine &Uuml;bung mehr m&ouml;glich.<br>";
220         }
221
222         // Es sind Fehler vorhanden
223         if (count($errors) > 0)
224         {
225             setSessionVars();
226
227             $header = "Location: " . $projectPath . "index_session.php?" .
228                 "menu=" . $menu . "&content=buchen/index&PHPSESSID=" . session_id() .
229                 "&day=" . $requestDay .
230                 "&month=" . $requestMonth .
231                 "&year=" . $requestYear;
232
233             if (isset($requestTime))
234                 $header = $header . "&time=" . $requestTime;
235
236             if (!$debug)
237                 header ($header);
238             else
239                 echo ($header);
240             exit;
241         }
242         else
243         {
244             setSessionVars();
245
246             $header = "Location: " . $projectPath . "index_session.php?" .
247                 "menu=" . $menu . "&content=buchen/buchen3&PHPSESSID=" . session_id() .
248                 "&day=" . $requestDay .
249                 "&month=" . $requestMonth .
250                 "&year=" . $requestYear;
251
252             if (isset($requestTime))
```

```

254         $header = $header . "&time=" . $requestTime;
255
256         if (!$debug)
257             header ($header);
258         else
259             echo ($header);
260         exit;
261     }
262 }
?>

```

Listing E.28: user/buchen/validate.php

```

<?php
2
3     /*
4     * user/buchen/validate.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ---
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['day']
15    // $HTTP_GET_VARS['month']
16    // $HTTP_GET_VARS['year']
17    // $HTTP_GET_VARS['time']
18    // $HTTP_GET_VARS['find']
19    // $HTTP_GET_VARS['sort']
20    // $HTTP_GET_VARS['startRecord']
21    // $HTTP_GET_VARS['PHPSESSID']
22    // POST
23    // ----
24    // $HTTP_POST_VARS['PHPSESSID']
25    // $HTTP_POST_VARS['day']
26    // $HTTP_POST_VARS['month']
27    // $HTTP_POST_VARS['year']
28    // $HTTP_POST_VARS['time']
29    //
30    // SESSION
31    // -----
32    // $HTTP_SESSION_VARS['projectPath']
33    // $HTTP_SESSION_VARS['absoluteProjectPath']
34    // $HTTP_SESSION_VARS['errors']
35    // $HTTP_SESSION_VARS['auth']
36    // $HTTP_SESSION_VARS['login']
37    // $HTTP_SESSION_VARS['menu']
38
39    $debug = false;
40
41    // hole Session-Variable

```



```
42  if (isset($_HTTP_GET_VARS["PHPSESSID"]))
        $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
44  else
        $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];
46
    // hole vorhandene Session
48  session_start();
    $PHPSESSID=session_id();
50
    // falls Session verfuegbar, hole Session-Variablen
52  if (!empty($PHPSESSID))
    {
54      if (isset($_HTTP_SESSION_VARS["projectPath"]))
          $projectPath = $_HTTP_SESSION_VARS["projectPath"];
56
58      if (isset($_HTTP_SESSION_VARS["projectPath"]))
          $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
60
62      if (isset($_HTTP_SESSION_VARS["menu"]))
          $menu = $_HTTP_SESSION_VARS["menu"];
64  }
66
    if ($debug)
    {
68        echo $projectPath . "<br>";
        echo $absoluteProjectPath . "<br>";
70
72        require_once $absoluteProjectPath . "functions.php";
        require_once $absoluteProjectPath . "config.php";
74
76        // Fehlerbehandlung
        ob_start('ob_gzhandler');
        set_error_handler('handleErrors');
        $scriptName = $menu . "/buchen/validate.php";
78
79        if ($debug)
80        {
81            echo "Session: " . session_id() . "<br>";
            showSessionVar();
82            echo "Post:<br>";
            showPostVar();
83            echo "Get:<br>";
            showGetVar();
84        }
86
87        function setSessionVars()
88        {
89            {
90                global $errors;
                global $_HTTP_SESSION_VARS;
92
93                $_HTTP_SESSION_VARS["errors"] = $errors;
94            }
96
97            // Variablen in der Session registrieren
98            if (!session_is_registered("errors"))
                session_register("errors");
```

```

100 // Auslesen der Formulardaten
    foreach($HTTP_POST_VARS as $varname => $value)
102     $formVars[$varname] = trim($value);

104 // Session-Variable
    $errors = array();
106

107 if (!empty($HTTP_POST_VARS))
108 {
    // zurueck-Button gewaehlt
110     if (isset($formVars['back']))
    {
112         setSessionVars();

114         $header = "Location: " . $projectPath . "index_session.php?" .
            "menu=" . $menu . "&content=buchen/index&PHPSESSID=" . session_id() .
116             "&day=" . $formVars['day'] .
            "&month=" . $formVars['month'] .
118             "&year=" . $formVars['year'] .
            "&time=" . $formVars['time'];

120
        if (!$debug)
122             header ($header);
        else
124             echo ($header);
        exit;
126     }
    // Uebung buchen - auf Fehler pruefen
128     elseif (empty($formVars['uebung']))
    {
130         $errors[] = "Es wurde keine &Uuml;bung gew&auml;hlt.";
        if ($debug)
132             echo "Es wurde keine &Uuml;bung gew&auml;hlt.<br>";
    }
134 // Es sind Fehler vorhanden
    if (count($errors) > 0)
136     {
        setSessionVars();
138
        $header = "Location: " . $projectPath . "index_session.php?" .
            "menu=" . $menu . "&content=buchen/buchen3&PHPSESSID=" . session_id() .
140             "&day=" . $formVars['day'] .
            "&month=" . $formVars['month'] .
142             "&year=" . $formVars['year'] .
            "&time=" . $formVars['time'];

144
        if (isset($formVars['find']))
            $header = $header . "&find=" . $formVars['find'];
146
148
        if (!$debug)
            header ($header);
        else
150             echo ($header);
        exit;
152
    }
154 else
156 {
    $header = "Location: " . $projectPath . "index_session.php?" .

```

```

158         "menu=" . $menu . "&content=buchen/zeigeBuchung&PHPSESSID=" .
            session_id() .
160         "&day=" . $formVars['day'] .
            "&month=" . $formVars['month'] .
162         "&year=" . $formVars['year'] .
            "&time=" . $formVars['time'] .
164         "&uebung=" . $formVars['uebung'];

166         if (isset($formVars['find']))
            $header = $header . "&find=" . $formVars['find'];
168
169         if (!$debug)
170             header ($header);
171         else
172             echo ($header);
173         exit;
174     }
175 }
176
?>

```

Listing E.29: user/buchen/zeigeBuchung.php

```

<?php
2
3     /*
4     * user/buchen/zeigeBuchung.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ---
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['day']
15    // $HTTP_GET_VARS['month']
16    // $HTTP_GET_VARS['year']
17    // $HTTP_GET_VARS['time']
18    // $HTTP_GET_VARS['PHPSESSID']
19    //
20    // SESSION
21    // -----
22    // $HTTP_SESSION_VARS['projectPath']
23    // $HTTP_SESSION_VARS['absoluteProjectPath']
24    // $HTTP_SESSION_VARS['errors']
25    // $HTTP_SESSION_VARS['auth']
26    // $HTTP_SESSION_VARS['login']
27    // $HTTP_SESSION_VARS['menu']
28
29    $debug = false;
30
31    // hole Session-Variable

```

```

32  if (isset($_HTTP_GET_VARS["PHPSESSID"]))
        $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
34  else
        $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];
36
    // hole vorhandene Session
38  session_start();
    $PHPSESSID=session_id();
40
    // falls Session verfuegbar, hole Session-Variablen
42  if (!empty($PHPSESSID))
    {
44      if (isset($_HTTP_SESSION_VARS["projectPath"]))
          $projectPath = $_HTTP_SESSION_VARS["projectPath"];
46
48      if (isset($_HTTP_SESSION_VARS["projectPath"]))
          $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
50
52      if (isset($_HTTP_SESSION_VARS["menu"]))
          $menu = $_HTTP_SESSION_VARS["menu"];
54  }
56
58  if ($debug)
    {
        echo $projectPath . "<br>";
        echo $absoluteProjectPath . "<br>";
    }
60
    require_once $absoluteProjectPath . "functions.php";
    require_once $absoluteProjectPath . "config.php";
62
    // Fehlerbehandlung
64  ob_start('ob_gzhandler');
    set_error_handler('handleErrors');
66  $scriptName = $menu . "/buchen/validate.php";
68
    if ($debug)
    {
69      echo "Session: " . session_id() . "<br>";
        showSessionVar();
72      echo "Post:<br>";
        showPostVar();
74      echo "Get:<br>";
        showGetVar();
76    }
78
    // Auslesen der Daten
    foreach($_HTTP_GET_VARS as $varname => $value)
80        $formVars[$varname] = trim($value);
82
    // baue Verbindung zur Datenbank auf
    $connection = getConnection();
84
    // SQL-Abfrage zusammensetzen
86  $sql = "SELECT * FROM uebungen WHERE uebung_id=' " .
        $formVars['uebung'] . " '";
88
    if ($debug)

```

```

90     echo "SQL: " . $sql . "<br>\n";

92     $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
           "Wenden Sie sich an Ihren Administrator.";

94

96     $result = executeStatement($connection, $sql, $msg);

98     if ($debug)
99         echo "Result: " . $result . "<br>";

100     //////////////////////////////////////
101     // Ausgabe
102
103     echo "<p class=\"head\">Buchung vornehmen</p>\n";
104
105     if (pg_numrows($result) == 1)
106     {
107         // hole Datensatz zur Uebung
108         $array = fetchArray($result, 0);
109
110         echo "<table class=\"vulab\" width=\"60%\">\n";
111         echo "<tr><td align=\"center\">\n";
112         echo "<p>&nbsp;</p>\n";
113
114         echo "<form method=\"post\" action=\"\" . $projectPath .
115              $menu . "/buchen/speichereBuchung.php\">\n";
116
117         if (!empty($PHPSESSID))
118             echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" .
119                  $PHPSESSID . "\">\n";
120         if (!empty($formVars['day']))
121             echo "<input type=\"hidden\" name=\"day\" value=\"\" .
122                  $formVars['day'] . "\">\n";
123         if (!empty($formVars['month']))
124             echo "<input type=\"hidden\" name=\"month\" value=\"\" .
125                  $formVars['month'] . "\">\n";
126         if (!empty($formVars['year']))
127             echo "<input type=\"hidden\" name=\"year\" value=\"\" .
128                  $formVars['year'] . "\">\n";
129         if (!empty($formVars['time']))
130             echo "<input type=\"hidden\" name=\"time\" value=\"\" .
131                  $formVars['time'] . "\">\n";
132         if (!empty($formVars['uebung']))
133             echo "<input type=\"hidden\" name=\"uebung\" value=\"\" .
134                  $formVars['uebung'] . "\">\n";
135         if (!empty($formVars['find']))
136             echo "<input type=\"hidden\" name=\"find\" value=\"\" .
137                  $formVars['find'] . "\">\n";
138
139         echo "<p class=\"text\">\n";
140         echo "<b>Folgende Buchung wird ausgew&uml;hlt:</b>\n";
141         echo "</p>\n";
142         echo "<p class=\"text\">\n";
143         echo $array['bezeichnung'] . "<br>(Dauer: " .
144              cutTime($array['dauer']) . " Minuten)<br>\n";
145         echo "am: " . $formVars['day'] . "." .
146              $formVars['month'] . "." . $formVars['year'] .
147              " beginnend um " . $formVars['time'] . " Uhr<br>\n";

```

```

148     echo "</p>\n";

150     echo "<table width=\"60%\">\n";
151     echo "<tr><td align=\"left\">\n";
152     echo "<input type=\"submit\" name=\"back\" value=\"zur&uuml;ck\"/>\n";
153     echo "</td><td align=\"right\">\n";
154     echo "<input type=\"submit\" name=\"buchen\" value=\"buchen !!\"/>\n";
155     echo "</td></tr>\n";
156     echo "</table>\n";

158     echo "</form>\n";

160     echo "</td></tr>\n";
161     echo "</table>\n";

162 }
163 else
164 {
165     echo "<table>\n";
166     echo "<tr><td><div align=\"center\">\n";
167     echo "<p class=\"error\">\n";
168     echo "Es ist ein Fehler aufgetreten\n";
169     echo "</p>\n";
170     echo "</div></td></tr>\n";
171     echo "</table>\n";
172 }
?>

```

Listing E.30: user/buchen/speichereBuchung.php

```

<?php
2  /*
   * user/buchen/speichereBuchung.php
4  */

6  // Verfuegbare Variablen
7  // =====
8  //
9  // GET
10 // ---
11 // $HTTP_GET_VARS['menu']
12 // $HTTP_GET_VARS['content']
13 // $HTTP_GET_VARS['day']
14 // $HTTP_GET_VARS['month']
15 // $HTTP_GET_VARS['year']
16 // $HTTP_GET_VARS['time']
17 // $HTTP_GET_VARS['uebung']
18 // $HTTP_GET_VARS['PHPSESSID']
19 //
20 // SESSION
21 // -----
22 // $HTTP_SESSION_VARS['projectPath']
23 // $HTTP_SESSION_VARS['absoluteProjectPath']
24 // $HTTP_SESSION_VARS['errors']
25 // $HTTP_SESSION_VARS['auth']

```

```
26 // $HTTP_SESSION_VARS['login']
27 // $HTTP_SESSION_VARS['menu']
28
29 $debug = false;
30
31 // hole Session-Variable
32 if (isset($HTTP_GET_VARS["PHPSESSID"]))
33     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34 else
35     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
36
37 // hole vorhandene Session
38 session_start();
39 $PHPSESSID=session_id();
40
41 // falls Session verfuegbar, hole Session-Variablen
42 if (!empty($PHPSESSID))
43 {
44     if (isset($HTTP_SESSION_VARS["projectPath"]))
45         $projectPath = $HTTP_SESSION_VARS["projectPath"];
46
47     if (isset($HTTP_SESSION_VARS["projectPath"]))
48         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
49
50     if (isset($HTTP_SESSION_VARS["menu"]))
51         $menu = $HTTP_SESSION_VARS["menu"];
52
53     unset($HTTP_SESSION_VARS['errors']);
54 }
55
56 if ($debug)
57 {
58     echo $projectPath . "<br>";
59     echo $absoluteProjectPath . "<br>";
60 }
61
62 require_once $absoluteProjectPath . "functions.php";
63 require_once $absoluteProjectPath . "config.php";
64
65 // Fehlerbehandlung
66 ob_start('ob_gzhandler');
67 set_error_handler('handleErrors');
68 $scriptName = $menu . "/buchen/speichereBuchung.php";
69
70 if ($debug)
71 {
72     echo "Session: " . session_id() . "<br>";
73     showSessionVar();
74     echo "Post:<br>";
75     showPostVar();
76     echo "Get:<br>";
77     showGetVar();
78 }
79
80 function setSessionVars()
81 {
82     global $errors;
83     global $HTTP_SESSION_VARS;
```

```

84     $HTTP_SESSION_VARS["errors"] = $errors;
85 }
86
87 function sendMail()
88 {
89     global $debug, $server, $adminName, $adminMail;
90     global $replyName, $replyMail, $Cc, $Bcc;
91     global $projectPath;
92     global $HTTP_SESSION_VARS, $formVars;
93
94     // baue Verbindung zur Datenbank auf
95     $connection = getConnection();
96
97     //////////////////////////////////////
98     // SQL-Abfrage: Auslesen des Namens
99
100     $sql = "SELECT * FROM benutzer " .
101         "WHERE login='" . $HTTP_SESSION_VARS['login'] . "'";
102
103     if ($debug)
104         echo $sql . "<br>\n";
105
106     $msg = "Bei Auslesen den Benutzers ist ein Fehler aufgetreten.<br>".
107         "Wenden Sie sich an Ihren Administrator.";
108
109     $result = executeStatement($connection, $sql, $msg);
110
111     $benutzer = fetchArray($result, 0);
112
113     //////////////////////////////////////
114     // SQL-Abfrage: Auslesen der Uebung
115
116     $sql = "SELECT * FROM uebungen " .
117         "WHERE uebung_id='" . $formVars['uebung'] . "'";
118
119     if ($debug)
120         echo $sql . "<br>\n";
121
122     $msg = "Bei Auslesen den Uebung ist ein Fehler aufgetreten.<br>".
123         "Wenden Sie sich an Ihren Administrator.";
124
125     $result = executeStatement($connection, $sql, $msg);
126
127     $uebung = fetchArray($result, 0);
128
129     //////////////////////////////////////
130     // SQL-Abfrage: Auslesen der Uebung
131
132     // Empfaenger
133     $to = $benutzer['vorname'] . " " . $benutzer['nachname'] .
134         "<" . $benutzer['email'] . ">";
135
136     // Betreff
137     $subject = "neue Buchung im virtuellen Unix Labor";
138
139     // Nachricht
140     $message = "Hallo " . $benutzer['vorname'] . " " .

```



```

142     $benutzer['nachname'] . " !!\n\n";
    $message .= "Soeben wurde folgende Uebung im " .
144         "'virtuellen Unix Labor' fuer Sie reserviert.\n\n";
    $message .= "Fuer " . $formVars['day'] . "." . $formVars['month'];
146     $message .= "." . $formVars['year'] . " um " . $formVars['time'];
    $message .= " Uhr:\n";
148     $message .= $uebung['bezeichnung'] . " (Dauer: " .
        cutTime($uebung['dauer']) . ")\n\n";
150     $message .= "Ihr " . $adminName . "\n\n";

    /* To send HTML mail, you can set the Content-type header. */
    $headers = "MIME-Version: 1.0\r\n";
154     $headers .= "Content-type: text/plain; charset=iso-8859-1\r\n";

    /* additional headers */
    $headers .= "From: virtuelles Unix Labor <" . $adminMail . ">\r\n";
158     $headers .= "Reply-to: " . $replyName . " <" . $replyMail . ">\r\n";
    $headers .= "Cc: " . $Cc . "\r\n";
160     $headers .= "Bcc: " . $Bcc . "\r\n";

    /* and now mail it */
    if (!$debug)
164         mail($to, $subject, $message, $headers);

    if ($debug)
166     {
        echo "<br><br>Mail : <br>\n";
        echo $to . "<br>\n";
170         echo $headers . "<br>\n";
        echo $subject . "<br>\n";
172         echo $message . "<br>\n";
    }
174 }

// Auslesen der Daten
foreach($HTTP_POST_VARS as $varname => $value)
178     $formVars[$varname] = trim($value);

if (isset($formVars['back']))
180 {
    $header = "Location: " . $projectPath . "index_session.php?" .
        "menu=" . $menu . "&content=buchen/buchen3&PHPSESSID=" . session_id() .
184         "&day=" . $formVars['day'] .
        "&month=" . $formVars['month'] .
186         "&year=" . $formVars['year'] .
        "&time=" . $formVars['time'] .
188         "&uebung=" . $formVars['uebung'];

    if (isset($formVars['find']))
        $header = $header . "&find=" . $formVars['find'];
192

    if (!$debug)
        header ($header);
194     else
        echo ($header);
196 }
elseif (isset($formVars['buchen']))
198 {

```

```

200 // baue Verbindung zur Datenbank auf
    $connection = getConnection();
202
    // SQL-Abfrage: die Buchung darf nur einmal geschehen
204 $sql = "SELECT buchungs_id FROM buchungen, benutzer " .
        "WHERE datum='" . $formVars['month'] . "/" .
206         $formVars['day'] . "/" . $formVars['year'] . "' " .
        "AND startzeit='" . $formVars['time'] . "' " .
208         "AND buchungen.user_id=benutzer.user_id " .
        "AND benutzer.login='" . $HTTP_SESSION_VARS['login'] . "'";
210
    if ($debug)
212         echo $sql . "<br>\n";
214
    $msg = "Bei Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
        "Wenden Sie sich an Ihren Administrator.";
216
    $result = executeStatement($connection, $sql, $msg);
218
    // diese Buchung darf es auch nicht geben
220 if (pg_numrows($result) > 0)
    {
222
        // Variablen in der Session registrieren
224 if (!session_is_registered("errors"))
            session_register("errors");
226
        $errors[] = "Eine Buchung f&uuml;r diesen Zeitpunkt wurde bereits vorgenommen.";
228
        if ($debug)
230             echo "Eine Buchung f&uuml;r diesen Zeitpunkt wurde bereits vorgenommen.";
232
        setSessionVars();
234
        $header = "Location: " . $projectPath . "index_session.php?" .
            "menu=" . $menu . "&content=buchen/index&PHPSESSID=" . session_id() .
236             "&day=" . $formVars['day'] .
            "&month=" . $formVars['month'] .
238             "&year=" . $formVars['year'] .
            "&time=" . $formVars['time'];
240
        if (!$debug)
242             header ($header);
        else
244             echo ($header);
        exit;
246     }
    else
248     {
        // SQL-Abfrage zusammensetzen
250 $sql = "INSERT INTO buchungen ";
        $sql .= "(user_id, uebung_id, datum, startzeit) ";
252 $sql .= "VALUES ";
        $sql .= "((SELECT user_id FROM benutzer WHERE login='" .
254             $HTTP_SESSION_VARS['login'] . "') , '" .
            $formVars['uebung'] . "', '" .
256             $formVars['month'] . "/" . $formVars['day'] . "/" .
            $formVars['year'] . "', '" .

```

```

258         $formVars['time'] . "'");
260
261         if ($debug)
262             echo $sql . "<br>\n";
263
264         $msg = "Bei Schreiben in die Datenbank ist ein Fehler aufgetreten.<br>".
265             "Wenden Sie sich an Ihren Administrator.";
266
267         $result = executeStatement($connection, $sql, $msg);
268
269         sendMail();
270
271         $header = "Location: " . $projectPath . "index_session.php?" .
272             "menu=" . $menu . "&content=buchungen/index&PHPSESSID=" . session_id();
273
274         //////////////////////////////////////
275
276         include $absoluteProjectPath . $menu . "/buchen/execute.php";
277
278         // Fehlerbehandlung
279         $scriptName = $menu . "/buchen/speichereBuchung.php";
280
281         //////////////////////////////////////
282
283         if (!$debug)
284             header ($header);
285         else
286             echo ($header);
287     }
288 }
?>

```

Listing E.31: user/buchen/execute.php

```

<?php
2
3     /*
4     * user/buchen/execute.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // SESSION
11    // -----
12    // $HTTP_SESSION_VARS['projectPath']
13    // $HTTP_SESSION_VARS['absoluteProjectPath']
14    // $HTTP_SESSION_VARS['errors']
15    // $HTTP_SESSION_VARS['auth']
16    // $HTTP_SESSION_VARS['login']
17    // $HTTP_SESSION_VARS['menu']
18    //
19    // formVars
20    // -----

```

```

22 // $formVars['menu']
23 // $formVars['content']
24 // $formVars['day']
25 // $formVars['month']
26 // $formVars['year']
27 // $formVars['time']
28 // $formVars['uebung']
29 // $formVars['PHPSESSID']

30 // Fehlerbehandlung
31 $scriptName = $menu . "/buchen/execute.php";

32
33 // baue Verbindung zur Datenbank auf
34 $connection = getConnection();

35
36 ///////////////////////////////////////////////////////////////////
37 // SQL-Abfrage zur Buchungs_Nr
38 $sql = "SELECT buchungs_id FROM buchungen, benutzer " .
39       "WHERE datum='" . $formVars['month'] . "/" .
40       $formVars['day'] . "/" . $formVars['year'] . "' " .
41       "AND startzeit='" . $formVars['time'] . "' " .
42       "AND buchungen.user_id=benutzer.user_id " .
43       "AND benutzer.login='" . $HTTP_SESSION_VARS['login'] . "'";
44
45 if ($debug)
46 {
47     echo "F&uuml;r Systembefehl - Buchungs_Nr: <br>";
48     echo "SQL: " . $sql . "<br><br>\n";
49 }

50 $msg = "Beim der Abfrage der Buchungsnummer ist ein Fehler aufgetreten.<br>".
51       "Wenden Sie sich an Ihren Administrator.";

52
53 $resultBuchung = executeStatement($connection, $sql, $msg);

54
55 if ($debug)
56     echo "ResultBuchung: " . $resultBuchung . "<br>";

57
58 $rowBuchung = fetchArray($resultBuchung, 0);
59 $buchungs_id = $rowBuchung['buchungs_id'];

60
61 ///////////////////////////////////////////////////////////////////
62 // SQL-Abfrage zur Startzeit - Vorlauf
63
64 $sql = "SELECT ('" . $formVars['month'] . "/" .
65       $formVars['day'] . "/" . $formVars['year'] . " " .
66       $formVars['time'] . "'::timestamp - vorlauf) " .
67       "AS startzeit FROM uebungen WHERE uebung_id='" .
68       $formVars['uebung'] . "'";
69
70
71 if ($debug)
72 {
73     echo "F&uuml;r Systembefehl - Startzeit: <br>";
74     echo "SQL: " . $sql . "<br><br>\n";
75 }

76
77 $msg = "Beim der Abfrage der Startzeit ist ein Fehler aufgetreten.<br>".
78       "Wenden Sie sich an Ihren Administrator.";

```

```

80     $resultStartZeit = executeStatement($connection, $sql, $msg);

82     if ($debug)
        echo "resultStartZeit: " . $resultStartZeit . "<br>";

84

86     $rowStartZeit = fetchArray($resultStartZeit, 0);
    $startzeit = $rowStartZeit['startzeit'];

88     //////////////////////////////////////
    // SQL-Abfrage zum Zerlegen der Startzeit

90

92     $sql = "SELECT date_part('hour', ' " . $startzeit . "'::timestamp) AS hour, " .
        "date_part('minute', ' " . $startzeit . "'::timestamp) AS minute, " .
        "date_part('day', ' " . $startzeit . "'::timestamp) AS day, " .
94     "date_part('month', ' " . $startzeit . "'::timestamp) AS month, " .
        "date_part('year', ' " . $startzeit . "'::timestamp) AS year";

96

98     if ($debug)
    {
        echo "F&uuml;r Systembefehl - zerlege Startzeit: <br>";
100     echo "SQL: " . $sql . "<br><br>\n";
    }

102

104     $msg = "Beim der Abfrage der Startzeit ist ein Fehler aufgetreten.<br>".
        "Wenden Sie sich an Ihren Administrator.";

106     $resultZerlegteZeit = executeStatement($connection, $sql, $msg);

108     if ($debug)
        echo "resultZerlegteZeit: " . $resultZerlegteZeit . "<br>";

110

112     $rowZerlegteZeit = fetchArray($resultZerlegteZeit, 0);

114     $hour = $rowZerlegteZeit['hour'];
    $minute = $rowZerlegteZeit['minute'];
    $day = $rowZerlegteZeit['day'];
116     $month = $rowZerlegteZeit['month'];
    $year = $rowZerlegteZeit['year'];

118

120     $time = $hour . ":" . $minute;
    $date = $month . "/" . $day . "/" . $year;

122     if ($debug)
        echo "Startzeit: " . $time . " " . $date . "<br>";

124

126     //////////////////////////////////////
    // Erstellen eines at-Jobs

128     if (!empty($buchungs_id) && !empty($startzeit))
    {
130         $command = "( echo cd /vulab; echo ./deploy " .
            $buchungs_id . " ) | " .
132         "at " . $time . " " . $date . " 2>&1";

134         // Systembefehl ausfuehren
        exec($command, $result);

136

```

```

138     $msg = "";
140     foreach ($result as $value)
141         $msg .= $value . "\n";
142
143     if ($debug)
144     {
145         echo $command . "<br>";
146         echo "Message: " . ($msg) . "<br>";
147     }
148
149     // Eintrag ins log_file
150     trigger_error("at-Job\n\t" . $command .
151         "\n\tMessage: " . $msg . "\n\t", E_USER_NOTICE);
152
153     // extrahiere at-Job-Nummer
154     // Job 13 will ... --> 13
155     $pos1 = strpos($msg, " ", 0);
156     $pos1 += 1;
157     $pos2 = strpos($msg, " ", $pos1);
158     $at_id = substr($msg, $pos1, $pos2 - $pos1);
159
160     // Eintrag ins log_file
161     trigger_error("at-Job " . $at_id .
162         " eingetragen: " . "\n\t", E_USER_NOTICE);
163
164     $sql = "UPDATE buchungen SET at_id='" . $at_id . "' " .
165         "WHERE datum='" .
166             $formVars['month'] . "/" .
167             $formVars['day'] . "/" .
168             $formVars['year'] . "' " .
169         "AND startzeit='" . $formVars['time'] . "' " .
170         "AND buchungen.user_id=benutzer.user_id " .
171         "AND benutzer.login='" . $HTTP_SESSION_VARS['login'] . "'";
172
173     if ($debug)
174     {
175         echo "F&uuml;r Systembefehl - at_id eintragen: <br>";
176         echo "SQL: " . $sql . "<br><br>\n";
177     }
178
179     $msg = "Beim eintragen der at_id ist ein Fehler aufgetreten.<br>".
180         "Wenden Sie sich an Ihren Administrator.";
181
182     $result = executeStatement($connection, $sql, $msg);
183
184     if ($debug)
185         echo "result: " . $result . "<br>";
186 }
?>

```

E.3.4 Buchungen einsehen

Listing E.32: user/buchungen/index.php

```
<?php
2
3   /*
4   * user/buchungen/index.php
5   */
6
7   // Verfuegbare Variablen
8   // =====
9   //
10  // GET
11  // ---
12  // $HTTP_GET_VARS['menu']
13  // $HTTP_GET_VARS['content']
14  // $HTTP_GET_VARS['day']
15  // $HTTP_GET_VARS['month']
16  // $HTTP_GET_VARS['year']
17  // $HTTP_GET_VARS['time']
18  // $HTTP_GET_VARS['PHPSESSID']
19  //
20  // SESSION
21  // -----
22  // $HTTP_SESSION_VARS['projectPath']
23  // $HTTP_SESSION_VARS['absoluteProjectPath']
24  // $HTTP_SESSION_VARS['errors']
25  // $HTTP_SESSION_VARS['auth']
26  // $HTTP_SESSION_VARS['login']
27  // $HTTP_SESSION_VARS['menu']
28
29  $debug = false;
30
31  // hole Session-Variable
32  if (isset($HTTP_GET_VARS["PHPSESSID"]))
33      $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34  else
35      $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
36
37  // hole vorhandene Session
38  session_start();
39  $PHPSESSID=session_id();
40
41  // falls Session verfuegbar, hole Session-Variablen
42  if (!empty($PHPSESSID))
43  {
44      if (isset($HTTP_SESSION_VARS["projectPath"]))
45          $projectPath = $HTTP_SESSION_VARS["projectPath"];
46
47      if (isset($HTTP_SESSION_VARS["projectPath"]))
48          $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
49
50      if (isset($HTTP_SESSION_VARS["menu"]))
51          $menu = $HTTP_SESSION_VARS["menu"];
52  }
```

```

54     if ($debug)
55     {
56         echo $projectPath . "<br>";
57         echo $absoluteProjectPath . "<br>";
58     }

60     require_once $absoluteProjectPath . "functions.php";
61     require_once $absoluteProjectPath . "config.php";
62
63     // Fehlerbehandlung
64     ob_start('ob_gzhandler');
65     set_error_handler('handleErrors');
66     $scriptName = $menu . "/buchungen/index.php";

68     if ($debug)
69     {
70         echo "Session: " . session_id() . "<br>";
71         showSessionVar();
72         echo "Post:<br>";
73         showPostVar();
74         echo "Get:<br>";
75         showGetVar();
76     }

78     // zerlegt Datum
79     // 2003-05-16 --> 16.05.2003
80     function makeDate($date)
81     {
82         $pos = (int) strrpos($date, '-');
83         $day = substr($date, $pos + 1);
84         $tmp = substr($date, 0, $pos);
85         $pos = (int) strrpos($tmp, '-');
86         $month = substr($tmp, $pos + 1);
87         $year = substr($tmp, 0, $pos);

88         return $day . "." . $month . "." . $year;
89     }

92     // baue Verbindung zur Datenbank auf
93     $connection = getConnection();
94
95     // Variable fuer Such-Abfragen
96     $find = getValue("GET", "find");
97     $finder = "";
98
99     $sqlFirst = "SELECT * FROM buchungen, uebungen, benutzer " .
100         "WHERE uebungen.uebung_id=buchungen.uebung_id AND " .
101         "buchungen.user_id=benutzer.user_id AND " .
102         "benutzer.login='" . $_HTTP_SESSION_VARS['login'] . "'";

104     // Such-Abfrage zusammensetzen
105     if (!empty($find))
106         $finder = "AND (upper(buchungen.uebung_id) LIKE upper('%" . $find . "%') OR " .
107             "upper(bezeichnung) LIKE upper('%" . $find . "%') OR " .
108             "upper(freigegeben) LIKE upper('%" . $find . "%') OR " .
109             "datum LIKE '%" . $find . "%' OR " .
110             "startzeit LIKE '%" . $find . "%' OR " .

```



```

112         "dauer LIKE '%" . $find . "%')";
113
114     // SQL-Abfrage zusammensetzen
115     if (isset($HTTP_GET_VARS['sort']) &&
116         ($HTTP_GET_VARS['sort'] != "dauer") &&
117         ($HTTP_GET_VARS['sort'] != "datum") &&
118         ($HTTP_GET_VARS['sort'] != "wiederholbar") &&
119         ($HTTP_GET_VARS['sort'] != "startzeit"))
120         $sql = $sqlFirst . $finder .
121             " ORDER BY LOWER(" . $HTTP_GET_VARS['sort'] . ") ";
122     elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "dauer")
123         $sql = $sqlFirst . $finder . " ORDER BY dauer ";
124     elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "datum")
125         $sql = $sqlFirst . $finder . " ORDER BY datum ";
126     elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "wiederholbar")
127         $sql = $sqlFirst . $finder . " ORDER BY wiederholbar ";
128     elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "startzeit")
129         $sql = $sqlFirst . $finder . " ORDER BY startzeit ";
130     else
131     {
132         $sql = $sqlFirst . $finder .
133             " ORDER BY datum, startzeit ";
134         $HTTP_GET_VARS['sort'] = "datum";
135     }
136
137     if ($debug)
138         echo "SQL: " . $sql . "<br>";
139
140     $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
141         "Wenden Sie sich an Ihren Administrator.";
142
143     $result = executeStatement($connection, $sql, $msg);
144
145     if ($debug)
146         echo "Result: " . $result . "<br>";
147
148     //////////////////////////////////////
149     // blaettern in den Datensatzen
150     //
151     // !! zu setzende Variablen siehe $absoluteProjectPath/config.php !!
152
153     // mit welchem Datensatz wird begonnen?
154     $startRecord = getValue("GET", "startRecord");
155
156     if (empty($startRecord))
157         $startRecord = 0;
158
159     // ermittle letzten Datensatz
160     $numrows = pg_numrows($result) - 1;
161
162     if ( ($numrows + 1 - $startRecord) < $number)
163         $stopRecord = $numrows + 1;
164     else
165         $stopRecord = $startRecord + $number;
166
167     // korrigiere $startRecord
168     if ($startRecord > $numrows)
169         $startRecord = $numrows;

```

```

170     if ( ($corr = $startRecord % $number) != 0 )
172         $startRecord = $startRecord - $corr;

174     if ( $startRecord < 0 )
176         $startRecord = 0;

178     // Teil-URL zum Sortieren
180     $sort = "?menu=" . $menu . "&content=buchungen/index";

182     if (!empty($find))
184         $sort = $sort . "&find=$find";

186     $sort = $sort . "&PHPSESSID=$PHPSESSID&sort=";
188     $sortKurzBezeichnung = $sort . "buchungen.uebung_id&startRecord=" . $startRecord;
190     $sortBezeichnung = $sort . "bezeichnung&startRecord=" . $startRecord;
192     $sortDatum = $sort . "datum&startRecord=" . $startRecord;
194     $sortStartzeit = $sort . "startzeit&startRecord=" . $startRecord;
196     $sortDauer = $sort . "dauer&startRecord=" . $startRecord;
198     $sortWiederholbar = $sort . "wiederholbar&startRecord=" . $startRecord;
200     $sortFreigegeben = $sort . "freigegeben&startRecord=" . $startRecord;

202     // Anzahl der Seiten
204     $pages = intval(($numrows + 1) / $number);

206     if ( (($numrows + 1) % $number) > 0 )
208         $pages = $pages + 1;

210     // aktuelle Seite
212     $currentPage = -1 ;

214     for ($i = 0; $i <= $startRecord; $i = $i + $number)
216         $currentPage = $currentPage + 1;

218     // erste Seite
220     $firstPage = $currentPage - intval($linkNumber / 2);

222     // letzte verfuegbare Seite
224     $tmp = $numrows % $number;
226     $tmp = $numrows - $tmp;
228     $lastPage = ($tmp / $number);

230     if ( ($lastPage - $currentPage) < intval($linkNumber / 2))
232         $firstPage = $lastPage - ($linkNumber - 1);

234     if ($firstPage < 0)
236         $firstPage = 0;

238     // letzte anzuzeigende Seite
240     if ($pages < $linkNumber)
242         $stopPage = $pages;
244     else
246         $stopPage = $linkNumber;

248     if ($debug)
250     {
252         echo "anzuzeitende Datensatze: $number <br>";
254         echo "Anzahl der Links: $linkNumber <br>";

```

```

228     echo "first Page: $firstPage <br>";
229     echo "current Page: $currentPage <br>";
230     echo "last Page: $lastPage <br>";
231     echo "stop Page: $stopPage <br>";
232 }
233
234 // liefert eine HTML-Zeile zum blaettern in den Datensatzen
235 function browse($count)
236 {
237     global $linkBack, $linkForward, $startRecord, $stopRecord;
238     global $number, $result, $sort, $HTTP_GET_VARS;
239     global $linkNumber, $currentPage, $firstPage;
240     global $numrows, $stopPage, $projectPath;
241
242     if ( $startRecord < $number)
243     {
244         $linkBack = "";
245     }
246     else
247     {
248         $linkBack = "<a href='" . $projectPath . 'index_session.php' .
249             $sort . $HTTP_GET_VARS['sort'] . "&startRecord=" .
250             ( $startRecord - $number ) . "' " .
251             "onmouseout=\"changeImage('imageBack$count', 'outBack');\" " .
252             "onmouseover=\"changeImage('imageBack$count', 'overBack');\" " .
253             "onmousedown=\"changeImage('imageBack$count', 'downBack');\">" .
254             "<img src='images/back1.gif' width='30' height='26' border='0' " .
255             "alt='zurueck' name='imageBack$count'></a>\n";
256     }
257
258     if ( $stopRecord >= ($numrows + 1))
259     {
260         $linkForward = "";
261     }
262     else
263     {
264         $linkForward = "<a href='" . $projectPath . 'index_session.php' .
265             $sort . $HTTP_GET_VARS['sort'] . "&startRecord=" .
266             ( $startRecord + $number ) . "' " .
267             "onmouseout=\"changeImage('imageForward$count', 'outForward');\" " .
268             "onmouseover=\"changeImage('imageForward$count', 'overForward');\" " .
269             "onmousedown=\"changeImage('imageForward$count', 'downForward');\">" .
270             "<img src='images/forward1.gif' width='30' height='26' border='0' " .
271             "alt='weiter' name='imageForward$count'></a>\n";
272     }
273
274     $links = "";
275
276     $i = $firstPage;
277
278     do {
279         if ( ($i * $number) == $startRecord)
280         {
281             $links = $links . "&nbsp;" . ($startRecord + 1). "-" .
282                 $stopRecord . "&nbsp;";
283         }
284         else
285         {
286             $links = $links . "&nbsp;" .
287                 "<a href='" . $projectPath . 'index_session.php' .
288                 $sort . $HTTP_GET_VARS['sort'] . "&startRecord=" .
289                 ($i * $number) . "'>" .
290                 (( $i * $number ) + 1). "-" . (( $i * $number ) + $number) .
291                 "</a>&nbsp;";
292         }
293     } while ( $i * $number < $startRecord );

```

```

286         $i++;
        } while ($i < $stopPage + $firstPage);

288     $links = $links . "&nbsp;";

290     // Ausgabe zusammensetzen
    $string = "<table border='0' cellpadding='2' >" .
292     "<tr>\n" .
    "<td align='right' width='40'>" .
294     $linkBack .
    "</td>\n" .
296     "<td align='center' width='300'><p class='text'>" .
    // "Datensatz " . $startRecord . " - " . $stopRecord .
298     $links .
    "</p></td>\n" .
300     "<td align='left' width='40'>" .
    $linkForward .
302     "</td>\n" .
    "</tr>\n" .
304     "</table>";

306     return $string;
    }
308     ?>

310 <script type="text/javascript">
var navName = navigator.appName;
312 var version = parseInt(navigator.appVersion);
var browserOk = (((navName == "Netscape") && (version >= 3)) ||
314     ((navName == "Microsoft Internet Explorer") && (version >= 4)));

316 function changeImage(imageName, newSrc) {
    // Changes images
318     if (browserOk) {
        document.images[imageName].src = eval(newSrc + ".src");
320     }
    }

322 if (browserOk) {
324     // Image Loading Start
    outDelete = new Image();
326     outEdit = new Image();
    outBack = new Image();
328     outForward = new Image();
    outDelete.src = "images/deletel.gif";
330     outEdit.src = "images/editl.gif";
    outBack.src = "images/backl.gif";
332     outForward.src = "images/forwardl.gif";

334     overDelete = new Image();
    overEdit = new Image();
336     overBack = new Image();
    overForward = new Image();
338     overDelete.src = "images/deletel.gif";
    overEdit.src = "images/editl.gif";
340     overBack.src = "images/backl.gif";
    overForward.src = "images/forwardl.gif";
342

```

```

344     downDelete = new Image();
345     downEdit    = new Image();
346     downBack    = new Image();
347     downForward = new Image();
348     downDelete.src = "images/delete2.gif";
349     downEdit.src   = "images/edit2.gif";
350     downBack.src   = "images/back2.gif";
351     downForward.src = "images/forward2.gif";
352 }
353 </script>
354 <div align="center">
355     <p class="head">Verwaltung gebuchter &Uuml;bungen</p>
356     <p></p>
357
358     <!-- Stichwort-Suche -->
359
360     <p class='border'>
361     <table>
362     <tr><td align="center" valign="middle" >
363     <form name="" action="<?php echo $projectPath . "index_session.php" ?>" method="get">
364         <p class='text'>&nbsp;Stichwort-Suche:&nbsp;
365         <input type="hidden" name="menu" value="<?php echo $menu ?>">
366         <input type="hidden" name="content" value="buchungen/index">
367     <?php
368         if (!empty($find))
369             echo "<input type='hidden' name='find' value='\" . $find . "\">\n";
370         if (!empty($PHPSESSID))
371             echo "<input type='hidden' name='\"PHPSESSID\" value='\" . $PHPSESSID . "\">\n";
372     ?>
373         <input type="text" name="find" size="20" value="<?php echo $find; ?>">&nbsp;
374         <input type="submit" name="" value="Suchen"> &nbsp;
375     </form></p>
376     </td></tr>
377     </table>
378     </p>
379
380     <!-- Such-Ergebnisse -->
381
382     <p></p>
383 <?php
384     echo "<p class='error'>";
385     echo "Achtung: Sollten Buchungen in roter Farbe auftreten, " .
386         "wenden sie sich bitte an Ihren Administrator !!";
387     echo "</p>";
388
389     echo "<p class='text'>";
390     echo "Vorhandene Buchungen: " . ($numrows + 1) . "<br>";
391     echo browse(1);
392     echo "</p>";
393
394     echo "<table border='0' cellpadding='2'>";
395
396     echo "<tr>\n";
397     echo "<th class='vulab'><a href='\" . $projectPath . 'index_session.php' .
398         $sortKurzBezeichnung . "\">Kurzbez.</a></th>\n";
399     echo "<th class='vulab'><a href='\" . $projectPath . 'index_session.php' .

```

```

    $sortBezeichnung . "\">Bezeichnung</a></th>\n";
402 echo "<th class='vulab' width=\\"80\\"><a href=\"\" . $projectPath . 'index_session.php' .
    $sortDatum . "\">Datum</a></th>\n";
404 echo "<th class='vulab' width=\\"70\\"><a href=\"\" . $projectPath . 'index_session.php' .
    $sortStartzeit . "\">Startzeit</a></th>\n";
406 echo "<th class='vulab' width=\\"40\\"><a href=\"\" . $projectPath . 'index_session.php' .
    $sortDauer . "\">Dauer</a></th>\n";
408 echo "<th class='vulab' width=\\"50\\"><a href=\"\" . $projectPath . 'index_session.php' .
    $sortWiederholbar . "\">wieder-holbar</a></th>\n";
410 echo "<th class='vulab'><a href=\"\" . $projectPath . 'index_session.php' .
    $sortFreigegeben . "\">freigegeben</a></th>\n";
412 echo "</tr>\n";

414 for ($i = $startRecord; $i < $stopRecord; $i++)
{
416     $array = fetchArray($result, $i);

418     if ( empty($array['at_id']) && $array['freigegeben'] == "nein")
        $class = "error";
420     else
        $class = "vulab";
422

    echo "<tr>\n";
424 echo "<td class='\" . $class . "\">\" . $array['uebung_id'] . "</td>\n";
    echo "<td class='\" . $class . "\">\" . $array['bezeichnung'] . "</td>\n";
426 echo "<td class='\" . $class . "\" align='right'>\" .
        makeDate($array['datum']) . "</td>\n";
428 echo "<td class='\" . $class . "\" align='right'>\" .
        cutTime($array['startzeit']) . " Uhr</td>\n";
430 echo "<td class='\" . $class . "\" align='right'>\" .
        cutTime($array['dauer']) . "</td>\n";
432 echo "<td class='\" . $class . "\" align='center'>\";
    if ($array['wiederholbar'] == 't')
434         echo "ja";
    else
436         echo "nein";
    echo "<td class='\" . $class . "\" align='center'>\" .
438         $array['freigegeben'] . "</td>\n";

440 if ($array['freigegeben'] == "nein")
{
442     echo "<td class='vulab'>\n";
    echo "<a href=\"\" . $projectPath . 'index_session.php' .
444         "?menu=" . $menu . "&content=buchungen/delete&at_id=" .
        $array['at_id'] . "&buchungs_id=" . $array['buchungs_id'] .
446         "&PHPSESSID=" . $PHPSESSID . "\"";

448     if (isset($_HTTP_GET_VARS['sort']))
        echo "&sort=" . $_HTTP_GET_VARS['sort'] ;
450     if (isset($_HTTP_GET_VARS['find']))
        echo "&find=" . $_HTTP_GET_VARS['find'] ;
452     if (isset($_HTTP_GET_VARS['startRecord']))
        echo "&startRecord=" . $_HTTP_GET_VARS['startRecord'] ;
454

    echo "\" onmouseout=\"changeImage('imageDelete$i', 'outDelete');\"\" .
456     "onmouseover=\"changeImage('imageDelete$i', 'overDelete');\"\" .
        "onmousedown=\"changeImage('imageDelete$i', 'downDelete');\">\" .
458     "<img src='images/delete1.gif' width='30' height='26' border='0' " .

```

```

        "alt='Buchung löschen' name='imageDelete$i'></a>\n";
460     echo "</td>\n";
        }
462     echo "</tr>\n";
464 }
466 ?>

</table></p>
</div>
470
<?php
472     echo "<p class='text'>";
        echo browse(2);
474     echo "</p>";
    ?>

```

Listing E.33: user/buchungen/delete.php

```

<?php
2
    /*
4     * user/buchungen/delete.php
    */
6
    // Verfügbare Variablen
    // =====
    //
10   // GET
    // ----
12   // $HTTP_GET_VARS['PHPSESSID']
    // $HTTP_GET_VARS['menu']
14   // $HTTP_GET_VARS['content']
    // $HTTP_GET_VARS['at_id']
16   // $HTTP_GET_VARS['sort']
    // $HTTP_GET_VARS['find']
18   // $HTTP_GET_VARS['startRecord']
    //
20   // SESSION
    // -----
22   // $HTTP_SESSION_VARS['projectPath']
    // $HTTP_SESSION_VARS['absoluteProjectPath']
24   // $HTTP_SESSION_VARS['errors']
    // $HTTP_SESSION_VARS['auth']
26   // $HTTP_SESSION_VARS['login']
    // $HTTP_SESSION_VARS['menu']
28
    $debug = false;
30
    // hole Session-Variable
32   if (isset($HTTP_GET_VARS["PHPSESSID"]))
        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34   else

```

```

36     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];

38     // hole vorhandene Session
39     session_start();
40     $PHPSESSID=session_id();

42     // falls Session verfuegbar, hole Session-Variablen
43     if (!empty($PHPSESSID))
44     {
45         if (isset($HTTP_SESSION_VARS["projectPath"]))
46             $projectPath = $HTTP_SESSION_VARS["projectPath"];

48         if (isset($HTTP_SESSION_VARS["projectPath"]))
49             $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];

50         if (isset($HTTP_SESSION_VARS["menu"]))
51             $menu = $HTTP_SESSION_VARS["menu"];
52     }

54     require_once $absoluteProjectPath . "functions.php";

56     // Fehlerbehandlung
57     ob_start('ob_gzhandler');
58     set_error_handler('handleErrors');
59     $scriptName = $menu . "/buchungen/delete.php";

60     if ($debug)
61     {
62         echo "Session: " . session_id() . "<br>";
63         showSessionVar();
64         echo "Post:<br>";
65         showPostVar();
66         echo "Get:<br>";
67         showGetVar();
68     }

70     // baue Verbindung zur Datenbank auf
71     $connection = getConnection();

72     //////////////////////////////////////
73     // hole passenden Datensatz zur MatrikelNr
74
75     if (isset($HTTP_GET_VARS['buchungs_id']))
76     {
77         $sql = "SELECT * FROM buchungen, uebungen, benutzer " .
78             "WHERE uebungen.uebung_id=buchungen.uebung_id AND " .
79             "buchungen.user_id=benutzer.user_id AND buchungs_id=" .
80             $HTTP_GET_VARS['buchungs_id'];
81     }
82     else
83     {
84         $sql = "";
85     }

86     if ($debug)
87         echo "sql: " . $sql . "<br>\n";

88     $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>" .
89         "Wenden Sie sich an Ihren Administrator.";

90     $result = executeStatement($connection, $sql, $msg);
91
92

```



```

$array = fetchArray($result, 0);
94
    if ($debug)
96        print_r ($array);
?>
98
<div align="center">
100    <p></p>
    <p class="head">Buchung l&ouml;uschen</p>
102
    <p class="text">
104        Die folgende Buchung wird unwiederruflich gel&ouml;scht !!
    </p>
106
    <form method="post" action="
108        <?php echo $projectPath .
            "index_session.php?menu=" . $menu . "&content=buchungen/delete2" ?>">
110    <p class="text">
    <table border='0' cellpadding='2'>
112
    <?php
114        if (!empty($PHPSESSID))
            echo "<input type='hidden' name='PHPSESSID' value='" . $PHPSESSID . "'>\n";
116        if (isset($_HTTP_GET_VARS['buchungs_id']))
            echo "<input type='hidden' name='buchungs_id' value='" .
118                $_HTTP_GET_VARS['buchungs_id'] . "'>\n";
        if (isset($_HTTP_GET_VARS['at_id']))
120            echo "<input type='hidden' name='at_id' value='" . $_HTTP_GET_VARS['at_id'] . "'>\n";
        if (isset($_HTTP_GET_VARS['sort']))
122            echo "<input type='hidden' name='sort' value='" . $_HTTP_GET_VARS['sort'] . "'>\n";
        if (isset($_HTTP_GET_VARS['find']))
124            echo "<input type='hidden' name='find' value='" . $_HTTP_GET_VARS['find'] . "'>\n";
        if (isset($_HTTP_GET_VARS['startRecord']))
126            echo "<input type='hidden' name='startRecord' value='" .
                $_HTTP_GET_VARS['startRecord'] . "'>\n";
128
    ?>
130
    <tr>
132        <th class="vulab">Buchungs_ID</th>
        <th class="vulab">Kurzbezeichnung</th>
134        <th class="vulab">Datum</th>
        <th class="vulab">Startzeit</th>
136        <th class="vulab">Dauer</th>
        <th class="vulab">freigegeben</th>
138    </tr>
    <tr>
140        <td class="vulab"><?php echo $array['buchungs_id'] ?> </td>
        <td class="vulab"><?php echo $array['uebung_id'] ?> </td>
142        <td class="vulab"><?php echo $array['datum'] ?> </td>
        <td class="vulab"><?php echo cutTime($array['startzeit']) ?> </td>
144        <td class="vulab"><?php echo cutTime($array['dauer']) ?> </td>
        <td class="vulab" align="center"><?php echo $array['freigegeben'] ?> </td>
146    </tr>
    </table></p>
148
    <p class="text">
150        Wollen Sie fortfahren ?

```

```
</p>
152
    <input type="submit" name="submitJa" value="ja">&nbsp;
154    <input type="submit" name="submitNein" value="nein">&nbsp;
156
    </form>
</div>
```

Listing E.34: user/buchungen/delete2.php

```
<?php
2
    /*
4    * user/buchungen/delete2.php
    */
6
    // Verfuegbare Variablen
8    // =====
    //
10   // GET
    // ----
12   // $HTTP_POST_VARS['PHPSESSID']
    // $HTTP_POST_VARS['menu']
14   // $HTTP_POST_VARS['content']
    // $HTTP_POST_VARS['at_id']
16   // $HTTP_POST_VARS['sort']
    // $HTTP_POST_VARS['find']
18   // $HTTP_POST_VARS['startRecord']
    //
20   // SESSION
    // -----
22   // $HTTP_SESSION_VARS['projectPath']
    // $HTTP_SESSION_VARS['absoluteProjectPath']
24   // $HTTP_SESSION_VARS['errors']
    // $HTTP_SESSION_VARS['auth']
26   // $HTTP_SESSION_VARS['login']
    // $HTTP_SESSION_VARS['menu']
28
    $debug = false;
30
    // hole Session-Variable
32   if (isset($HTTP_GET_VARS["PHPSESSID"]))
        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34   else
        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
36
    // hole vorhandene Session
38   session_start();
    $PHPSESSID=session_id();
40
    // falls Session verfuegbar, hole Session-Variablen
42   if (!empty($PHPSESSID))
    {
44       if (isset($HTTP_SESSION_VARS["projectPath"]))
```

```
46     $projectPath = $HTTP_SESSION_VARS["projectPath"];
47
48     if (isset($HTTP_SESSION_VARS["projectPath"]))
49         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
50
51     if (isset($HTTP_SESSION_VARS["menu"]))
52         $menu = $HTTP_SESSION_VARS["menu"];
53 }
54
55 require_once $absoluteProjectPath . "functions.php";
56
57 // Fehlerbehandlung
58 ob_start('ob_gzhandler');
59 set_error_handler('handleErrors');
60 $scriptName = $menu . "/buchen/delete2.php";
61
62 if ($debug)
63 {
64     echo "Session: " . session_id() . "<br>";
65     showSessionVar();
66     echo "Post:<br>";
67     showPostVar();
68     echo "Get:<br>";
69     showGetVar();
70 }
71
72 if (isset($HTTP_POST_VARS['submitJa']))
73 {
74     // loesche at-Job
75     if (isset($HTTP_POST_VARS['at_id']))
76     {
77         $command = "at -r " . $HTTP_POST_VARS['at_id'] . " 2>&1";
78         exec ($command, $result);
79
80         if ($debug)
81         {
82             echo "<b>Kommando: </b><br>\n" . $command . "<br><br>\n";
83
84             $msg = "";
85             foreach ($result as $value)
86                 $msg .= $value . "<br>\n";
87
88             echo "<b>Ausgabe: </b><br>\n" . $msg;
89         }
90     }
91
92     // loesche Datensatz
93     $connection = getConnection();
94
95     $sql = "DELETE FROM buchungen WHERE buchungs_id='" .
96         $HTTP_POST_VARS['buchungs_id'] . "'";
97
98     if ($debug)
99         echo "SQL: " . $sql . "<br>";
100
101     $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
102         "Wenden Sie sich an Ihren Administrator.";
```

```

104     $result = executeStatement($connection, $sql, $msg);
106     if ($debug)
107         echo "Result: " . $result . "<br>";
108
109     //////////////////////////////////////
110     $header = "Location: " . $projectPath . 'index_session.php' .
111         "?menu=" . $menu . "&content=buchungen/deleted&PHPSESSID=" .
112         session_id();
113
114     if (!$debug)
115         header ($header);
116
117     if ($debug)
118         echo "Buchung gel&ouml;scht.";
119
120     exit;
121 }
122 else
123 {
124     $header = "Location: " . $projectPath . 'index_session.php' .
125         "?menu=" . $menu . "&content=buchungen/index&PHPSESSID=" .
126         session_id();
127
128     if (isset($_HTTP_POST_VARS['sort']))
129         $header .= "&sort=" . $_HTTP_POST_VARS['sort'] ;
130     if (isset($_HTTP_POST_VARS['find']))
131         $header .= "&find=" . $_HTTP_POST_VARS['find'] ;
132     if (isset($_HTTP_POST_VARS['startRecord']))
133         $header .= "&startRecord=" . $_HTTP_POST_VARS['startRecord'] ;
134
135     if (!$debug)
136         header ($header);
137
138     if ($debug)
139         echo "Keine Aenderungen vorgenommen.";
140
141     exit;
142 }
143
144 ?>

```

Listing E.35: user/buchungen/deleted.php

```

<?php
2
/*
4  * user/buchungen/deleted.php
5  */
6
?>
8
<div align="center">

```

```

10      <p></p>
      <p class="head">Buchung l&ouml;schen</p>
12 </div>
<div align="center">
14     <br><br>
      <p class="border">
16     <table>
          <tr>
18         <td>
              <div align="center">
20                 <p class="text">
                    Die Buchung wurde erfolgreich gel&ouml;scht.
22                 </p>
              </div>
24         </td>
          </tr>
26     </table>
      </p>
28 </div>

```

E.3.5 Error

Listing E.36: user/error/index.php

```

<?php
2
    /*
4     * user/error/index.php
    */
6
    // Verfuegbare Variablen
8    // =====
    //
10   // GET
    // ----
12   // $HTTP_GET_VARS['menu']
    // $HTTP_GET_VARS['content']
14   // $HTTP_GET_VARS['msg']
    // $HTTP_GET_VARS['PHPSESSID']
16   //
    // POST
18   // ----
    // ???
20   //
    // SESSION
22   // -----
    // $HTTP_SESSION_VARS['projectPath']
24   // $HTTP_SESSION_VARS['absoluteProjectPath']
    // ???
26
    $debug = false;
28

```

```

30 // hole Session-Variable
31 if (isset($_HTTP_GET_VARS["PHPSESSID"]))
32     $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
33 else
34     $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];
35
36 // hole vorhandene Session
37 session_start();
38 $PHPSESSID=session_id();
39
40 // falls Session verfuegbar, hole Session-Variablen
41 if (!empty($PHPSESSID))
42 {
43     if (isset($_HTTP_SESSION_VARS["projectPath"]))
44         $projectPath = $_HTTP_SESSION_VARS["projectPath"];
45
46     if (isset($_HTTP_SESSION_VARS["projectPath"]))
47         $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
48
49     if (isset($_HTTP_SESSION_VARS["menu"]))
50         $menu = $_HTTP_SESSION_VARS["menu"];
51 }
52
53 require_once $absoluteProjectPath . "functions.php";
54
55 // Fehlerbehandlung
56 ob_start('ob_gzhandler');
57 set_error_handler('handleErrors');
58 $scriptName = $menu . "/error/index.php";
59
60 if ($debug)
61 {
62     echo "Session: " . session_id() . "<br>";
63     showSessionVar();
64     echo "Post:<br>";
65     showPostVar();
66     echo "Get:<br>";
67     showGetVar();
68 }
69
70 <div align="center">
71     <br>
72     <p></p>
73     <p class="head">Fehler im vuLab</p>
74 </div>
75 <div align="center">
76     <p class="error">
77         <table>
78             <tr>
79                 <td>
80                     <div align="center">
81
82 <?php
83     foreach($_HTTP_GET_VARS as $varname => $value)
84         $formVars[$varname] = trim($value);
85
86

```

```
88     echo "<p class=\"error\">";
      echo $formVars["msg"];
      echo "</p>";
90 ?>
      </div>
92     </td>
      </tr>
94     </table>
      </p>
96 </div>
```

E.4 Module im Administrationsbereich

Listing E.37: admin/config.php

```
<?php
2
/*
4  * admin/config.php
  */
6
// Menue-Eintraege in Reihenfolge ihres Auftretens
8 $items = array();
$items[1] = "home";
10 $items[2] = "Benutzerdaten";
$items[4] = "Buchungen";
12 $items[7] = "logout";

14 // Zuweisung der Verzeichnisse auf die Menue-Namen
// $access[Menue-Name] = Verzeichniss-Name
16 $access = array();
$access["error"] = "error";
18 $access["home"] = "home";
$access["Benutzerdaten"] = "userdata";
20 $access["Buchungen"] = "buchungen";

22 // keine Zuweisung fuer logout notwendig
// $access["logout"] = "logout";
24

// Zuweisung der Beschreibung auf die Menue-Namen
26 // $description[Menue-Name] = Beschreibung
$description = array();
28 $description["home"] = "Link auf diese Seite";
$description["Benutzerdaten"] = "Hier k&ouml;nnen Sie alle " .
30     "pers&ouml;nlichen Daten &auml;ndern";
$description["Buchungen"] = "Sehen Sie in die Liste alle gebuchten " .
32     " &Uuml;bungsaufgaben";
$description["logout"] = "Verlassen Sie ihr Profil";
34 ?>
```

E.4.1 Home

Der Quellcode des Moduls "Home" im Administrationsbereich ist identisch mit den Scripten des Moduls "Home" im Anwenderbereich (siehe Kapitel [E.3.1](#)).

E.4.2 Benutzerdaten

Listing E.38: admin/userdata/index.php

```
<?php
2
3     /*
4     * admin/userdata/index.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // beim ersten Aufruf:
13    // $HTTP_GET_VARS['menu']
14    // $HTTP_GET_VARS['content']
15    // $HTTP_GET_VARS['PHPSESSID']
16    // danach:
17    // $HTTP_GET_VARS['sort']
18    // $HTTP_GET_VARS['startRecord']
19    // $HTTP_GET_VARS['find']
20    //
21    // SESSION
22    // -----
23    // $HTTP_SESSION_VARS['projectPath']
24    // $HTTP_SESSION_VARS['absoluteProjectPath']
25    // $HTTP_SESSION_VARS['errors']
26    // $HTTP_SESSION_VARS['auth']
27    // $HTTP_SESSION_VARS['login']
28    // $HTTP_SESSION_VARS['menu']
29    //
30    // aus $projectPath/config.php
31    // -----
32    // $number
33    // $linkNumber
34
35    $debug = false;
36
37    // hole Session-Variable
38    if (isset($HTTP_GET_VARS["PHPSESSID"]))
39        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
40    else
41        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
42
43    // hole vorhandene Session
44    session_start();
45    $PHPSESSID=session_id();
46
47    // falls Session verfuegbar, hole Session-Variablen
48    if (!empty($PHPSESSID))
49    {
50        if (isset($HTTP_SESSION_VARS["projectPath"]))
51            $projectPath = $HTTP_SESSION_VARS["projectPath"];
52    }
```

```

        if (isset($_HTTP_SESSION_VARS["projectPath"]))
54         $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
    }

56
    require_once $absoluteProjectPath . "functions.php";
58     require_once $absoluteProjectPath . "config.php";

60     // Fehlerbehandlung
    ob_start('ob_gzhandler');
62     set_error_handler('handleErrors');
    $scriptName = "admin/userdata/index.php";

64
    if ($debug)
66     {
        echo "Session: " . session_id() . "<br>";
68         showSessionVar();
        echo "Post:<br>";
70         showPostVar();
        echo "Get:<br>";
72         showGetVar();
    }

74
    // baue Verbindung zur Datenbank auf
76     $connection = getConnection();

78     // Variable fuer Such-Abfragen
    $find = getValue("GET", "find");
80     $finder = "";

82     // Such-Abfrage zusammensetzen
    if (!empty($find))
84         $finder = "WHERE upper(vorname) LIKE upper('%" . $find . "%') OR " .
            "upper(nachname) LIKE upper('%" . $find . "%') OR " .
86             "upper(email) LIKE upper('%" . $find . "%') OR " .
            "upper(login) LIKE upper('%" . $find . "%') OR " .
88             "upper(typ) LIKE upper('%" . $find . "%') OR " .
            "int4(matrikel_nr) LIKE '%" . $find . "%'";

90
    // SQL-Abfrage zusammensetzen
92     if (isset($_HTTP_GET_VARS['sort']) && $_HTTP_GET_VARS['sort'] != "matrikel_nr")
        $sql = "SELECT * FROM benutzer " .
94             $finder . " ORDER BY LOWER(" . $_HTTP_GET_VARS['sort'] . ") ";
    elseif (isset($_HTTP_GET_VARS['sort']) && $_HTTP_GET_VARS['sort'] == "matrikel_nr")
96         $sql = "SELECT * FROM benutzer " . $finder . " ORDER BY matrikel_nr ";
    else
98     {
        $sql = "SELECT * FROM benutzer " .
100            $finder . " ORDER BY lower(nachname), lower(vorname) ";
        $_HTTP_GET_VARS['sort'] = "nachname";
102    }

104
    if ($debug)
        echo "SQL: " . $sql . "<br>";

106
    $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
108         "Wenden Sie sich an Ihren Administrator.";

110
    $result = executeStatement($connection, $sql, $msg);

```

```

112     if ($debug)
113         echo "Result: " . $result . "<br>";
114
115     //////////////////////////////////////
116     // blaettern in den Datensatzen
117     //
118     // !! zu setzende Variablen siehe $absoluteProjectPath/config.php !!
119
120     // mit welchem Datensatz wird begonnen?
121     $startRecord = getValue("GET", "startRecord");
122
123     if (empty($startRecord))
124     $startRecord = 0;
125
126     // ermittle letzten Datensatz
127     $numrows = pg_numrows($result) - 1;
128
129     if ( ($numrows + 1 - $startRecord) < $number)
130         $stopRecord = $numrows + 1;
131     else
132         $stopRecord = $startRecord + $number;
133
134     // korrigiere $startRecord
135     if ($startRecord > $numrows)
136         $startRecord = $numrows;
137
138     if ( ($corr = $startRecord % $number) != 0)
139         $startRecord = $startRecord - $corr;
140
141     if ( $startRecord < 0)
142         $startRecord = 0;
143
144     // Teil-URL zum Sortieren
145     $sort = "?menu=admin&content=userdata/index";
146
147     if (!empty($find))
148         $sort = $sort . "&find=$find";
149
150     $sort = $sort . "&PHPSESSID=$PHPSESSID&sort=";
151     $sortMatrikel = $sort . "matrikel_nr&startRecord=" . $startRecord;
152     $sortName = $sort . "nachname&startRecord=" . $startRecord;
153     $sortVorname = $sort . "vorname&startRecord=" . $startRecord;
154     $sortLogin = $sort . "login&startRecord=" . $startRecord;
155     $sortType = $sort . "typ&startRecord=" . $startRecord;
156
157     // Anzahl der Seiten
158     $pages = intval(($numrows + 1) / $number);
159
160     if ( (($numrows + 1) % $number) > 0 )
161         $pages = $pages + 1;
162
163     // aktuelle Seite
164     $currentPage = -1 ;
165
166     for ($i = 0; $i <= $startRecord; $i = $i + $number)
167         $currentPage = $currentPage + 1;
168

```

```

170 // erste Seite
171 $firstPage = $currentPage - intval($linkNumber / 2);

172 // letzte verfuegbare Seite
173 $tmp = $numrows % $number;
174 $tmp = $numrows - $tmp;
175 $lastPage = ($tmp / $number);

176
177 if ( ($lastPage - $currentPage) < intval($linkNumber / 2))
178     $firstPage = $lastPage - ($linkNumber - 1);

179
180 if ($firstPage < 0)
181     $firstPage = 0;

182
183 // letzte anzuzeigende Seite
184 if ($pages < $linkNumber)
185     $stopPage = $pages;
186 else
187     $stopPage = $linkNumber;

188
189 if ($debug)
190 {
191     echo "anzuzeitende Datensatze: $number <br>";
192     echo "Anzahl der Links: $linkNumber <br>";
193     echo "first Page: $firstPage <br>";
194     echo "current Page: $currentPage <br>";
195     echo "last Page: $lastPage <br>";
196     echo "stop Page: $stopPage <br>";
197 }

198
199 // liefert eine HTML-Zeile zum blaettern in den Datensatzen
200 function browse($count)
201 {
202     global $linkBack, $linkForward, $startRecord, $stopRecord;
203     global $number, $result, $sort, $HTTP_GET_VARS;
204     global $linkNumber, $currentPage, $firstPage;
205     global $numrows, $stopPage, $projectPath;

206
207     if ( $startRecord < $number)
208         $linkBack = "";
209     else
210     {
211         $linkBack = "<a href='\" . $projectPath . 'index_session.php' .
212             $sort . $HTTP_GET_VARS['sort'] . "&startRecord=" .
213             ( $startRecord - $number ) . "\" \" .
214             "onmouseout=\\\"changeImage('imageBack$count', 'outBack');\\\" \" .
215             "onmouseover=\\\"changeImage('imageBack$count', 'overBack');\\\" \" .
216             "onmousedown=\\\"changeImage('imageBack$count', 'downBack');\\\">\" .
217             "<img src='images/back1.gif' width='30' height='26' border='0' \" .
218             "alt='zurueck' name='imageBack$count'></a>\\n";
219     }

220
221     if ( $stopRecord >= ($numrows + 1))
222         $linkForward = "";
223     else
224     {
225         $linkForward = "<a href='\" . $projectPath . 'index_session.php' .
226             $sort . $HTTP_GET_VARS['sort'] . "&startRecord=" .

```

```

228         ( $startRecord + $number ) . " ' " .
        "onmouseout=\"changeImage('imageForward$count', 'outForward');\" " .
230        "onmouseover=\"changeImage('imageForward$count', 'overForward');\" " .
        "onmousedown=\"changeImage('imageForward$count', 'downForward');\">" .
232        "<img src='images/forward1.gif' width='30' height='26' border='0' " .
        "alt='weiter' name='imageForward$count'></a>\n";
    }
234
    $links = "";
236
    $i = $firstPage;
238
    do {
240        if ( ($i * $number) == $startRecord)
            $links = $links . "&nbsp;" . ($startRecord + 1). "-" .
242            $stopRecord . "&nbsp;";
        else
244            $links = $links . "&nbsp;" .
                "<a href='" . $projectPath . 'index_session.php' .
246                $sort . $_GET_VARS['sort'] . "&startRecord=" .
                    ($i * $number) . "'>" .
248                ((($i * $number) + 1). "-" . ((($i * $number) + $number) .
                    "</a>&nbsp;";
250
            $i++;
252    } while ($i < $stopPage + $firstPage);

    $links = $links . "&nbsp;";

256    // Ausgabe zusammensetzen
    $string = "<table border='0' cellpadding='2' >" .
258    "<tr>\n" .
        "<td align='right' width='40'>" .
260        $linkBack .
        "</td>\n" .
262        "<td align='center' width='300'><p class='text'>" .
        //"Datensatz " . $startRecord . " - " . $stopRecord .
264        $links .
        "</p></td>\n" .
266        "<td align='left' width='40'>" .
        $linkForward .
268        "</td>\n" .
        "</tr>\n" .
270        "</table>";

272    return $string;
    }
274    ?>

276    <script type="text/javascript">
    var navName = navigator.appName;
278    var version = parseInt(navigator.appVersion);
    var browserOk = (((navName == "Netscape") && (version >= 3)) ||
280        ((navName == "Microsoft Internet Explorer") && (version >= 4)));

282    function changeImage(imageName, newSrc) {
        // Changes images
284        if (browserOk) {

```

```

        document.images[imageName].src = eval(newSrc + ".src");
286     }
287 }
288
289 if (browserOk) {
290     // Image Loading Start
291     outDelete = new Image();
292     outEdit = new Image();
293     outBack = new Image();
294     outForward = new Image();
295     outDelete.src = "images/deletel.gif";
296     outEdit.src = "images/editl.gif";
297     outBack.src = "images/backl.gif";
298     outForward.src = "images/forwardl.gif";
299
300     overDelete = new Image();
301     overEdit = new Image();
302     overBack = new Image();
303     overForward = new Image();
304     overDelete.src = "images/deletel.gif";
305     overEdit.src = "images/editl.gif";
306     overBack.src = "images/backl.gif";
307     overForward.src = "images/forwardl.gif";
308
309     downDelete = new Image();
310     downEdit = new Image();
311     downBack = new Image();
312     downForward = new Image();
313     downDelete.src = "images/delete2.gif";
314     downEdit.src = "images/edit2.gif";
315     downBack.src = "images/back2.gif";
316     downForward.src = "images/forward2.gif";
317 }
318 </script>
319
320 <div align="center">
321     <p class="head">Benutzerdaten einsehen / &uuml;ndern</p>
322     <p></p>
323
324     <!-- Stichwort-Suche -->
325
326     <p class='border'>
327         <table> <!-- border='1' cellpadding='10' bgcolor=#D8D8F4-->
328         <tr><!-- bgcolor=#B7B7DE--><td align="center" valign="middle" >
329             <form name="" action="<?php echo $projectPath . "index_session.php" ?>" method="get">
330                 <p class='text'>&nbsp;Stichwort-Suche:&nbsp;
331                 <input type="hidden" name="menu" value="admin">
332                 <input type="hidden" name="content" value="userdata/index">
333             <?php
334                 if (!empty($find))
335                     echo "<input type='hidden' name='find' value='\" . $find . "\">\n";
336                 if (!empty($PHPSESSID))
337                     echo "<input type='hidden' name='\"PHPSESSID\" value='\" . $PHPSESSID . "\">\n";
338             ?>
339                 <input type="text" name="find" size="20" value="<?php echo $find; ?>">&nbsp;
340                 <input type="submit" name="" value="Suchen"> &nbsp;
341             </form></p>
342         </td></tr>

```

```

344     </table>
345     </p>
346
347     <!-- Such-Ergebnisse -->
348
349     <p></p>
350 <?php
351     echo "<p class='text'>";
352     echo "Vorhandene User: " . ($numrows + 1);
353     echo browse(1);
354     echo "<p></p>";
355
356     echo "<table border='0' cellpadding='2'> <!--bgcolor=#D8D8F4-->";
357
358     echo "<tr bgcolor=#B7B7DE>\n";
359     echo "<th class='vulab'><a href=\"\" .
360         $projectPath . 'index_session.php' . $sortMatrikel . \">MatrikelNr</a></th>\n";
361     echo "<th class='vulab'><a href=\"\" .
362         $projectPath . 'index_session.php' . $sortName . \">Name</a></th>\n";
363     echo "<th class='vulab'><a href=\"\" .
364         $projectPath . 'index_session.php' . $sortVorname . \">Vorname</a></th>\n";
365     echo "<th class='vulab'><a href=\"\" .
366         $projectPath . 'index_session.php' . $sortLogin . \">Login</a></th>\n";
367     echo "<th class='vulab'><a href=\"\" .
368         $projectPath . 'index_session.php' . $sortType . \">Type</a></th>\n";
369     echo "</tr>\n";
370
371     for ($i = $startRecord; $i < $stopRecord; $i++)
372     {
373         $array = fetchArray($result, $i);
374
375         echo "<tr>\n";
376         echo "<td class='vulab' align='right'>" . $array['matrikel_nr'] . "</td>\n";
377         echo "<td class='vulab'>" . $array['nachname'] . "</td>\n";
378         echo "<td class='vulab'>" . $array['vorname'] . "</td>\n";
379         echo "<td class='vulab'>" . $array['login'] . "</td>\n";
380         echo "<td class='vulab'>" . $array['typ'] . "</td>\n";
381         echo "<td class='vulab'>\n";
382         echo "<a href=\"\" . $projectPath . 'index_session.php' .
383             "?menu=admin&content=userdata/adminData2&matrikelNr=" .
384             $array['matrikel_nr'] . "&PHPSESSID=" . $PHPSESSID . "\"";
385
386         if (isset($_HTTP_GET_VARS['sort']))
387             echo "&sort=" . $_HTTP_GET_VARS['sort'] ;
388         if (isset($_HTTP_GET_VARS['find']))
389             echo "&find=" . $_HTTP_GET_VARS['find'] ;
390         if (isset($_HTTP_GET_VARS['startRecord']))
391             echo "&startRecord=" . $_HTTP_GET_VARS['startRecord'] ;
392
393         echo "\" onmouseout=\"changeImage('imageEdit$i', 'outEdit');\"\" .
394             "onmouseover=\"changeImage('imageEdit$i', 'overEdit');\"\" .
395             "onmousedown=\"changeImage('imageEdit$i', 'downEdit');\">" .
396             "<img src='images/edit1.gif' width='30' height='26' border='0' " .
397             "alt='Profil editieren' name='imageEdit$i'></a>\n";
398         echo "<a href=\"\" . $projectPath . 'index_session.php' .
399             "?menu=admin&content=userdata/delete&matrikelNr=" .
400             $array['matrikel_nr'] . "&PHPSESSID=" . $PHPSESSID . "\"";

```

```

402     if (isset($HTTP_GET_VARS['sort']))
        echo "&sort=" . $HTTP_GET_VARS['sort'] ;
404     if (isset($HTTP_GET_VARS['find']))
        echo "&find=" . $HTTP_GET_VARS['find'] ;
406     if (isset($HTTP_GET_VARS['startRecord']))
        echo "&startRecord=" . $HTTP_GET_VARS['startRecord'] ;

408     echo "\" onmouseout=\"changeImage('imageDelete$i', 'outDelete');\" " .
        "onmouseover=\"changeImage('imageDelete$i', 'overDelete');\" " .
410     "onmousedown=\"changeImage('imageDelete$i', 'downDelete');\">" .
        "<img src='images/deletel.gif' width='30' height='26' border='0' " .
412     "alt='Profil l&ouml;schen' name='imageDelete$i'></a>\n";
        echo "</td>\n";
414     echo "</tr>\n";
    }
416
?>
418
    </table></p>
420 </div>

422 <?php
    echo "<p class='text'>";
424    echo browse(2);
    echo "</p>";
426 ?>

```

Listing E.39: admin/userdata/adminData2.php

```

<?php
2
    /*
4    * admin/userdata/adminData2.php
    */
6
    // Verfuegbare Variablen
8    // =====
    //
10   // GET
    // ----
12   // $HTTP_GET_VARS['menu']
    // $HTTP_GET_VARS['content']
14   // $HTTP_GET_VARS['matrikelNr']
    // $HTTP_GET_VARS['PHPSESSID']
16   // $HTTP_GET_VARS['sort']
    // $HTTP_GET_VARS['find']
18   // $HTTP_GET_VARS['startRecord']
    //
20   // SESSION
    // -----
22   // $HTTP_SESSION_VARS['projectPath']
    // $HTTP_SESSION_VARS['absoluteProjectPath']
24   // $HTTP_SESSION_VARS['errors']
    // $HTTP_SESSION_VARS['auth']

```



```
26 // $HTTP_SESSION_VARS['login']
27 // $HTTP_SESSION_VARS['menu']
28
29 $debug = false;
30
31 // hole Session-Variable
32 if (isset($HTTP_GET_VARS["PHPSESSID"]))
33     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34 else
35     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
36
37 // hole vorhandene Session
38 session_start();
39 $PHPSESSID=session_id();
40
41 // falls Session verfuegbar, hole Session-Variablen
42 if (!empty($PHPSESSID))
43 {
44     if (isset($HTTP_SESSION_VARS["projectPath"]))
45         $projectPath = $HTTP_SESSION_VARS["projectPath"];
46
47     if (isset($HTTP_SESSION_VARS["projectPath"]))
48         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
49 }
50
51 require_once $absoluteProjectPath . "functions.php";
52
53 // Fehlerbehandlung
54 ob_start('ob_gzhandler');
55 set_error_handler('handleErrors');
56 $scriptName = "admin/userdata/adminData2.php";
57
58 if ($debug)
59 {
60     echo "Session: " . session_id() . "<br>";
61     showSessionVar();
62     echo "Post:<br>";
63     showPostVar();
64     echo "Get:<br>";
65     showGetVar();
66 }
67
68 function getFormVars($fieldName)
69 {
70     global $formVars;
71     $value = "";
72
73     if (isset($formVars[$fieldName]))
74         $value = $formVars[$fieldName];
75
76     return $value;
77 }
78
79 function getErrors()
80 {
81     global $errors;
82     $msg = "";
```

```

84     if (isset($errors))
85     {
86         $msg .= "<p class=\"error\">\n";
87
88         foreach ($errors as $value)
89         {
90             $msg .= $value . "<br>\n";
91         }
92
93         $msg .= "</p>\n";
94     }
95
96     return $msg;
97 }
98
99 // baue Verbindung zur Datenbank auf
100 $connection = getConnection();
101
102 $sql = "SELECT * FROM benutzer WHERE matrikel_nr='" . $HTTP_GET_VARS['matrikelNr'] . "'";
103
104 $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
105       "Wenden Sie sich an Ihren Administrator.";
106
107 $result = executeStatement($connection, $sql, $msg);
108
109 if ($debug)
110 {
111     echo "SQL: " . $sql . "<br>";
112     echo "Result: " . $result . "<br>";
113 }
114
115 // holt die erste Zeile des Resultats
116 $array = fetchArray($result, 0);
117
118 foreach($array as $varname => $value)
119     $formVars[$varname] = trim($value);
120
121 if (array_key_exists("passwort", $formVars))
122     unset($formVars['passwort']);
123
124 if (array_key_exists("secondPasswort", $formVars))
125     unset($formVars['secondPasswort']);
126
127 if ($debug)
128     foreach($formVars as $x => $y)
129         echo "$x -> $y <br>\n";
130
131 ?>
132
133 <div align="center">
134     <p class="head">Benutzerdaten</p>
135     <p class="text">
136         In diesem Bereich k&ouml;nnen Sie die gespeicherten
137         pers&ouml;nlichen Daten &auml;ndern.<br>
138     </p>
139
140     <?php echo getErrors(); ?>

```

```

142 <form name="" action="<?php
      echo $projectPath . 'admin/userdata/validateData.php'; ?>" method="post">
144 <div align="center">
      <p class="border">
146 <?php
      if (!empty($PHPSESSID))
148         echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" .
              getValue('GET', 'PHPSESSID') . "\">\n";
150 ?>

      <input type="hidden" name="matrikelNr" value="<?php
152         echo $formVars['matrikel_nr'] ?>">
      <input type="hidden" name="login" value="<?php
154         echo $formVars['login'] ?>">

156 <table border="0" cellpadding="3" cellspacing="3">
      <tr>
158         <td align="right">
              <p class="text">MatrikelNr</p>
160         </td>
              <td><input type="text" name="matrikelNrNeu" size="50"
162                 value="<?php echo getFormVars('matrikel_nr') ?>"></td>
      </tr>
164 <tr>
              <td align="right">
166                 <p class="text">Vorname</p>
              </td>
              <td><input type="text" name="vornameNeu" size="50"
168                 value="<?php echo getFormVars('vorname') ?>"></td>
      </tr>
170 <tr>
              <td align="right">
172                 <p class="text">Name</p>
              </td>
              <td><input type="text" name="nameNeu" size="50"
174                 value="<?php echo getFormVars('nachname') ?>"></td>
      </tr>
176 <tr>
              <td align="right">
178                 <p class="text">eMail-Adresse</p>
              </td>
              <td><input type="text" name="mailNeu" size="50"
180                 value="<?php echo getFormVars('email') ?>"></td>
      </tr>
182 <tr>
              <td align="right">
184                 <p class="text">Login</p>
              </td>
              <td><input type="text" name="loginNeu" size="50"
186                 value="<?php echo getFormVars('login') ?>"></td>
      </tr>
188 <tr>
              <td align="right">
190                 <p class="text">Benutzer-Typ ?</p>
              </td>
              <td>
192                 <p class="text">
194                     Admin<input type="radio" name="userTypeNeu"
196                     <?php if (getFormVars('typ') == "admin")
198

```

```

200         echo 'checked' ; ?> value="admin"> &nbsp;
        Dozent<input type="radio" name="userTypeNeu"
202         <?php if (getFormVars('typ') == "dozent")
            echo 'checked' ; ?> value="dozent"> &nbsp;
204        User<input type="radio" name="userTypeNeu"
            <?php if (getFormVars('typ') == "user")
206            echo 'checked' ; ?> value="user">
        </p>
208    </td>
</tr>
210<tr>
    <td align="right">
212        <p class="text">Anmeldung best&uuml;tigt ?</p>
    </td>
214    <td>
        <p class="text">
216        Ja<input type="radio" name="secretNeu"
            <?php if (getFormVars('freischalt_secret') == "bestaetigt")
218            echo 'checked' ; ?> value="ja"> &nbsp;
        Nein<input type="radio" name="secretNeu"
            <?php if (getFormVars('freischalt_secret') != "bestaetigt")
220            echo 'checked' ; ?> value="nein"> &nbsp;
        <input type="text" name="secret" size="15" value="<?php
222            echo getFormVars('freischalt_secret') ?>" readonly>
        </p>
224    </td>
</tr>
226<tr>
    <td colspan="2" align="right">
228        <p class="error">Passwortfeld nur ausf&uuml;llen,
        wenn neues Passwort gesetzt werden soll !</p>
    </td>
232</tr>
<tr>
234    <td align="right">
        <p class="text">neues Passwort</p>
236    </td>
    <td><input type="password" name="passwortNeu" size="50"
238        value="<?php echo getFormVars('passwort') ?>"></td>
</tr>
240<tr>
    <td align="right">
242        <p class="text">Passwort wiederholen</p>
    </td>
244    <td><input type="password" name="secondPasswortNeu" size="50"
        value="<?php echo getFormVars('secondPasswort') ?>"></td>
246</tr>
<tr>
248    <td colspan="2" align="center">
        <p class="text">
250            <input type="submit" name="" value="Werte &uuml;bernehmen">&nbsp;
            <input type="reset" value="Zur&uuml;cksetzen">
252        </p>
    </td>
254</tr>
</table>
256</p>
</div>

```

```
258     </form>
    </div>
```

Listing E.40: admin/userdata/adminData3.php

```
<?php
2
3     /*
4     * admin/userdata/adminData3.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['PHPSESSID']
15    //
16    // SESSION
17    // -----
18    // $HTTP_SESSION_VARS['projectPath']
19    // $HTTP_SESSION_VARS['absoluteProjectPath']
20    // $HTTP_SESSION_VARS['errors']
21    // $HTTP_SESSION_VARS['auth']
22    // $HTTP_SESSION_VARS['login']
23    // $HTTP_SESSION_VARS['menu']
24    // $HTTP_SESSION_VARS['formVars']
25
26    $debug = false;
27
28    // hole Session-Variable
29    if (isset($HTTP_GET_VARS["PHPSESSID"]))
30        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
31    else
32        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
33
34    // hole vorhandene Session
35    session_start();
36    $PHPSESSID=session_id();
37
38    // falls Session verfuegbar, hole Session-Variablen
39    if (!empty($PHPSESSID))
40    {
41        if (isset($HTTP_SESSION_VARS["projectPath"]))
42            $projectPath = $HTTP_SESSION_VARS["projectPath"];
43
44        if (isset($HTTP_SESSION_VARS["projectPath"]))
45            $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
46
47        if (isset($HTTP_SESSION_VARS["formVars"]))
48            $formVars = $HTTP_SESSION_VARS["formVars"];
```

```

50     if (isset($_HTTP_SESSION_VARS["errors"]))
51         $errors = $_HTTP_SESSION_VARS["errors"];
52     }

53
54     require_once $absoluteProjectPath . "functions.php";

55
56     // Fehlerbehandlung
57     ob_start('ob_gzhandler');
58     set_error_handler('handleErrors');
59     $scriptName = "admin/userdata/adminData3.php";
60
61     if ($debug)
62     {
63         echo "Session: " . session_id() . "<br>";
64         showSessionVar();
65         echo "Post:<br>";
66         showPostVar();
67         echo "Get:<br>";
68         showGetVar();
69     }
70
71     if (array_key_exists("password", $formVars))
72         unset($formVars['password']);
73
74     if (array_key_exists("secondPassword", $formVars))
75         unset($formVars['secondPassword']);
76
77     function getFormVars($fieldName)
78     {
79         global $formVars;
80         $value = "";
81
82         if (isset($formVars[$fieldName]))
83             $value = $formVars[$fieldName];
84
85         return $value;
86     }
87
88     function getErrors()
89     {
90         global $errors;
91         $msg = "";
92
93         if (isset($errors))
94         {
95             $msg .= "<p class=\"error\">\n";
96
97             foreach ($errors as $value)
98             {
99                 $msg .= $value . "<br>\n";
100             }
101
102             $msg .= "</p>\n";
103         }
104
105         return $msg;
106     }
107
108     ?>

```

```

108 <div align="center">
110   <p class="head">Benutzerdaten einsehen / &ouml;ndern</p>
112   <p class="text">
113     In diesem Bereich k&ouml;nnen Sie die gespeicherten
114     pers&ouml;nlichen Daten &ouml;ndern.<br>
115   </p>
116   <?php echo getErrors(); ?>
117
118   <form name="" action="<?php
119     echo $projectPath . 'admin/userdata/validateData.php'; ?>" method="post">
120     <div align="center">
121       <p class="border">
122 <?php
123     if (!empty($PHPSESSID))
124       echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" .
125         getValue('GET', 'PHPSESSID') . "\">\n";
126   ?>
127
128     <input type="hidden" name="matrikelNr" value="<?php
129       echo $formVars['matrikelNr'] ?>">
130     <input type="hidden" name="login" value="<?php
131       echo $formVars['login'] ?>">
132
133     <table border="0" cellpadding="3" cellspacing="3">
134       <tr>
135         <td align="right">
136           <p class="text">MatrikelNr</p>
137         </td>
138         <td><input type="text" name="matrikelNrNeu" size="50"
139           value="<?php echo getFormVars('matrikelNrNeu') ?>"></td>
140       </tr>
141       <tr>
142         <td align="right">
143           <p class="text">Vorname</p>
144         </td>
145         <td><input type="text" name="vornameNeu" size="50"
146           value="<?php echo getFormVars('vornameNeu') ?>"></td>
147       </tr>
148       <tr>
149         <td align="right">
150           <p class="text">Name</p>
151         </td>
152         <td><input type="text" name="nameNeu" size="50"
153           value="<?php echo getFormVars('nameNeu') ?>"></td>
154       </tr>
155       <tr>
156         <td align="right">
157           <p class="text">eMail-Adresse</p>
158         </td>
159         <td><input type="text" name="mailNeu" size="50"
160           value="<?php echo getFormVars('mailNeu') ?>"></td>
161       </tr>
162       <tr>
163         <td align="right">
164           <p class="text">Login</p>
165         </td>
166         <td><input type="text" name="loginNeu" size="50"

```

```

166         value="php echo getFormVars('loginNeu') ?&gt;"&gt;&lt;/td&gt;
167     &lt;/tr&gt;
168     &lt;tr&gt;
169         &lt;td align="right"&gt;
170             &lt;p class="text"&gt;Benutzer-Typ &lt;/p&gt;
171         &lt;/td&gt;
172         &lt;td&gt;
173             &lt;p class="text"&gt;
174                 Admin&lt;input type="radio" name="userTypeNeu"
175                     &lt;?php if (getFormVars('userTypeNeu') == "admin")
176                         echo 'checked' ; ?&gt; value="admin"&gt; &amp;nbsp;
177                 Dozent&lt;input type="radio" name="userTypeNeu"
178                     &lt;?php if (getFormVars('userTypeNeu') == "dozent")
179                         echo 'checked' ; ?&gt; value="dozent"&gt; &amp;nbsp;
180                 User&lt;input type="radio" name="userTypeNeu"
181                     &lt;?php if (getFormVars('userTypeNeu') == "user")
182                         echo 'checked' ; ?&gt; value="user"&gt;
183             &lt;/p&gt;
184         &lt;/td&gt;
185     &lt;/tr&gt;
186     &lt;tr&gt;
187         &lt;td align="right"&gt;
188             &lt;p class="text"&gt;Anmeldung best&amp;uuml;tigt&lt;/p&gt;
189         &lt;/td&gt;
190         &lt;td&gt;
191             &lt;p class="text"&gt;
192                 Ja&lt;input type="radio" name="secretNeu"
193                     &lt;?php if (getFormVars('secret') == "bestaetigt")
194                         echo 'checked' ; ?&gt; value="ja"&gt; &amp;nbsp;
195                 Nein&lt;input type="radio" name="secretNeu"
196                     &lt;?php if (getFormVars('secret') != "bestaetigt")
197                         echo 'checked' ; ?&gt; value="nein"&gt; &amp;nbsp;
198                 &lt;input type="text" name="secret" size="15"
199                     value="<?php echo getFormVars('secret') ?&gt;" readonly&gt;
200             &lt;/p&gt;
201         &lt;/td&gt;
202     &lt;/tr&gt;
203     &lt;tr&gt;
204         &lt;td colspan="2" align="right"&gt;
205             &lt;p class="error"&gt;Passwortfeld nur ausf&amp;uuml;llen,
206                 wenn neues Passwort gesetzt werden soll !&lt;/p&gt;
207         &lt;/td&gt;
208     &lt;/tr&gt;
209     &lt;tr&gt;
210         &lt;td align="right"&gt;
211             &lt;p class="text"&gt;neues Passwort&lt;/p&gt;
212         &lt;/td&gt;
213         &lt;td&gt;&lt;input type="password" name="passwortNeu" size="50"
214             value="<?php echo getFormVars('passwortNeu') ?&gt;"&gt;&lt;/td&gt;
215     &lt;/tr&gt;
216     &lt;tr&gt;
217         &lt;td align="right"&gt;
218             &lt;p class="text"&gt;Passwort wiederholen&lt;/p&gt;
219         &lt;/td&gt;
220         &lt;td&gt;&lt;input type="password" name="secondPasswortNeu" size="50"
221             value="<?php echo getFormVars('secondPasswortNeu') ?&gt;"&gt;&lt;/td&gt;
222     &lt;/tr&gt;
223 &lt;/tr&gt;
</pre

```



```

224         <td colspan="2" align="center">
225             <p class="text">
226                 <input type="submit" name="" value="Werte &uuml;bernehmen">&nbsp;
227                 <input type="reset" value="Zur&uuml;cksetzen">
228             </p>
229         </td>
230     </tr>
231 </table>
232 </p>
233 </div>
234 </form>
</div>

```

```

40 // hole Session-Variable
41 if (isset($_HTTP_GET_VARS["PHPSESSID"]))
42     $PHPSESSID = $_HTTP_GET_VARS["PHPSESSID"];
43 else
44     $PHPSESSID = $_HTTP_POST_VARS["PHPSESSID"];
45
46 // hole vorhandene Session
47 session_start();
48 $PHPSESSID=session_id();
49
50 // falls Session verfuegbar, hole Session-Variablen
51 if (!empty($PHPSESSID))
52 {
53     if (isset($_HTTP_SESSION_VARS["projectPath"]))
54         $projectPath = $_HTTP_SESSION_VARS["projectPath"];
55
56     if (isset($_HTTP_SESSION_VARS["projectPath"]))
57         $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
58 }
59
60 require_once $absoluteProjectPath . "functions.php";
61
62 // Fehlerbehandlung
63 ob_start('ob_gzhandler');
64 set_error_handler('handleErrors');
65 $scriptName = "admin/userdata/validateData.php";
66
67 if ($debug)
68 {
69     echo "Session: " . session_id() . "<br>";
70     showSessionVar();
71     echo "Post:<br>";
72     showPostVar();
73     echo "Get:<br>";
74     showGetVar();
75 }
76
77 function setSessionVars()
78 {
79     global $errors, $formVars, $login;
80     global $_HTTP_SESSION_VARS;
81
82     $_HTTP_SESSION_VARS["errors"] = $errors;
83     $_HTTP_SESSION_VARS["formVars"] = $formVars;
84     $_HTTP_SESSION_VARS["login"] = $login;
85 }
86
87 // Variablen in der Session registrieren
88 if (!session_is_registered("errors"))
89     session_register("errors");
90
91 if (!session_is_registered("formVars"))
92     session_register("formVars");
93
94 if (!session_is_registered("login"))
95     session_register("login");
96
97 // Session-Variablen

```

```

98  $errors = array();
    $login = $HTTP_SESSION_VARS['login'];
100
    foreach($HTTP_POST_VARS as $varname => $value)
102         $formVars[$varname] = trim($value);

104  $HTTP_SESSION_VARS['formVars'] = $formVars;

106  if ($debug)
    {
108         print_r ($formVars);
        echo "<br>";
110    }

112  setSessionVars();

114  $validEmailExpr =
        "[0-9a-z~!#$%&_-]([.]?[0-9a-z~!#$%&_-])*".
116        "@[0-9a-z~!#$%&_-]([.]?[0-9a-z~!#$%&_-])*".
        "\.[0-9a-z~!#$%&_-]([0-9a-z~!#$%&_-])*$";
118
    $validNumExpr = "[0-9]*$";
120
    $validNumLength = "[0-9]{1,10}$";
122
    // Kontrolliert, ob ein Eintrag die maximale
124    // Laenge ueberschreitet
    // $name: Bezeichnung des Feldes
126    // $value: Inhalt des Feldes
    // $length: maximale Feldlaenge
128    function checkFieldLength($name, $value, $length)
    {
130        global $errors, $debug;

132        if (strlen($value) > $length)
        {
134            $errors[] = "Das Feld ". $name . " kann maximal " .
                $length . " Zeichen aufnehmen.";
136
            if ($debug)
138                echo "Das Feld ". $name . " kann maximal " .
                    $length . " Zeichen aufnehmen.<br>\n";
140        }
    }

142
    // baue Verbindung zur Datenbank auf
144    $connection = getConnection();

146    if ($debug)
        echo "Connection: " . $connection . "<br>\n";
148

    //////////////////////////////////////
150    // Funktion zum Sicherstellen,
    // dass sich selbst keine Administratorrechte genommen werden
152

    function checkOwner()
154    {
        global $debug, $connection, $errors;

```

```

156     global $formVars, $HTTP_SESSION_VARS;

158     $sql = "SELECT * FROM benutzer WHERE matrikel_nr = ' " . $formVars['matrikelNr'] . "'";

160     $msg = "Bei der Ueberpruefung des AdminFlag ist ein Fehler aufgetreten.<br>".
           "Wenden Sie sich an Ihren Administrator.";

162

164     if ($debug)
           echo "sql: " . $sql . "<br>\n";

166     $result = executeStatement($connection, $sql, $msg);

168     if ($debug)
           echo "Result: " . $result . "<br>\n";

170

172     $array = fetchArray($result, 0);

174     if ($debug)
           {
176             print_r ($array);
178             echo "<br>\n";
178             echo "Type: " . $array['typ'] . "<br>\n";
178         }

180     // wenn AdminFlag geloescht wuerde
182     // und das zu veraendernde Profil Administratorrechte hat
182     // und das zu veraendernde Profil das Eigene ist
184     if ($formVars['userTypeNeu'] != 'admin' &&
184         $array['typ'] == 'admin' &&
184         $HTTP_SESSION_VARS['login'] == $formVars['login'])
186     {
188         $errors[] = "Das Admin-Flag kann nicht ver&auml;ndert werden. " .
188             "Dadurch w&uuml;rde Ihr eigenes Profil ung&uuml;ltig.";

190         if ($debug)
192             echo "Das Admin-Flag kann nicht ver&auml;ndert werden. <br>" .
192             "Dadurch w&uuml;rde Ihr eigenes Profil ung&uuml;ltig.<br>\n";
194     }

196     //////////////////////////////////////

198     if (!empty($HTTP_POST_VARS))
199     {
200         if ($debug)
201         {
202             showPostVar();
204             showGetVar();
204         }

206         // ueberpruefung auf leere Felder
208         if ( ( empty($formVars['matrikelNrNeu']) &&
208             !is_numeric($formVars['matrikelNrNeu']) ) ||
208             empty($formVars['nameNeu']) ||
210             empty($formVars['vornameNeu']) ||
210             empty($formVars['mailNeu']) ||
212             empty($formVars['loginNeu']) ||
212             empty($formVars['secretNeu'])

```

```

214     )
215     {
216         $errors[] = "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.";
217
218         if ($debug)
219             echo "Ein oder mehrere Felder wurden nicht ausgef&uuml;llt.<br>";
220     }
221
222     // ueberpruefung auf Nummern-Felder
223     if (!ereg($validNumExpr, $formVars['matrikelNrNeu']))
224     {
225         $errors[] = "Die Matrikel-Nummer darf keine Buchstaben " .
226             "oder andere Zeichen enthalten.";
227
228         if ($debug)
229             echo "Die Matrikel-Nummer darf keine Buchstaben oder " .
230                 "andere Zeichen enthalten.<br>";
231     }
232     // ueberpruefung auf maximal 10 Stellen
233     elseif (!ereg($validNumLength, $formVars['matrikelNrNeu']))
234     {
235         $errors[] = "Die MatrikelNr ist auf 10 Ziffern beschr&auml;nkt.";
236         if ($debug)
237             echo "Die MatrikelNr ist auf 10 Ziffern beschr&auml;nkt.";
238     }
239
240     // ueberpruefung auf korrekte email-Form
241     if (!ereg($validEmailExpr, $formVars['mailNeu']))
242     {
243         $errors[] = "Ihre eingegebene Email-Adresse hat keine g&uuml;ltige Form.";
244
245         if ($debug)
246             echo "Ihre eingegebene Email-Adresse hat keine g&uuml;ltige Form.<br>";
247     }
248
249     // ueberpruefung auf korrektes Passwort
250     if ($formVars['passwortNeu'] != $formVars['secondPasswortNeu'])
251     {
252         $errors[] = "Ihre beiden angegebenen Passw&ouml;rter " .
253             "stimmen nicht &uuml;berein.";
254
255         if ($debug)
256             echo "Ihre beiden angegebenen Passw&ouml;rter " .
257                 "stimmen nicht &uuml;berein.<br>";
258     }
259
260     checkFieldLength ("Vorname", $formVars["vornameNeu"], 50);
261     checkFieldLength ("Name", $formVars["nameNeu"], 50);
262     checkFieldLength ("eMail", $formVars["mailNeu"], 80);
263     checkFieldLength ("Login", $formVars["loginNeu"], 80);
264     checkFieldLength ("Passwort", $formVars["passwortNeu"], 50);
265     checkFieldLength ("Passwort wiederholen", $formVars["secondPasswortNeu"], 50);
266
267     //////////////////////////////////////
268     // Es sind Fehler vorhanden
269     if (count($errors) > 0)
270     {
271         setSessionVars();

```

```

272         if (!$debug)
273             header ("Location: " . $projectPath . 'index_session.php' .
274                 "?menu=admin&content=userdata/adminData3&PHPSESSID=" . session_id());
275         exit;
276     }
277     // alle Eingaben sind richtig erfolgt
278     else
279     {
280         checkOwner();
281
282         //////////////////////////////////////
283         // wird login bereits verwendet
284
285         $sql = "SELECT * FROM benutzer WHERE login='" . $formVars['loginNeu'] .
286             "' AND matrikel_nr!='" . $formVars['matrikelNr'] . "'";
287
288         $msg = "Bei der Ueberpruefung des Logins ist ein Fehler aufgetreten.<br>".
289             "Wenden Sie sich an Ihren Administrator.";
290
291         if ($debug)
292             echo "sql: " . $sql . "<br>\n";
293
294         $result = executeStatement($connection, $sql, $msg);
295
296         if ($debug)
297             echo "Result: " . $result . "<br>\n";
298
299         // es darf diesen login noch nicht geben
300         if (pg_numrows($result) > 0)
301         {
302             $errors[] = "Dieser login wird bereits von einer " .
303                 "anderen Person verwendet.";
304
305             if ($debug)
306                 echo "Dieser login wird bereits von einer " .
307                     "anderen Person verwendet.<br>";
308         }
309
310         //////////////////////////////////////
311         // wird matrikelNr bereits verwendet
312
313         $sql = "SELECT * FROM benutzer WHERE matrikel_nr='" .
314             $formVars['matrikelNrNeu'] . "' AND matrikel_nr!='" .
315             $formVars['matrikelNr'] . "'";
316
317         $msg = "Bei der Ueberpruefung der Matrikel-Nr ist ein Fehler aufgetreten.<br>".
318             "Wenden Sie sich an Ihren Administrator.";
319
320         if ($debug)
321             echo "sql: " . $sql . "<br>\n";
322
323         $result = executeStatement($connection, $sql, $msg);
324
325         if ($debug)
326             echo "Result: " . $result . "<br>\n";
327
328         // es darf diese matrikelNr noch nicht geben

```

```

330         if (pg_numrows($result) > 0)
331         {
332             $errors[] = "Diese MatrikelNr wird bereits von einer " .
333                 "anderen Person verwendet.";
334
335             if ($debug)
336                 echo "Diese MatrikelNr wird bereits von einer " .
337                     "anderen Person verwendet.<br>";
338         }
339
340         //////////////////////////////////////
341         // Es sind Fehler vorhanden
342         if (count($errors) > 0)
343         {
344             //$_SESSION_VARS['errors'] = $errors;
345             setSessionVars();
346
347             if (!$debug)
348                 header ("Location: " . $projectPath . 'index_session.php' .
349                     "?menu=admin&content=userdata/adminData3&PHPSESSID=" . session_id());
350             exit;
351         }
352
353         //////////////////////////////////////
354         // Eintag in die Datenbank
355         else
356         {
357             if ($debug)
358                 echo "Es sind alle Eingaben erfolgt.<br>\n";
359
360             $cryptPass = createPass($formVars['passwortNeu']);
361
362             $sql = "UPDATE benutzer SET ";
363             $sql .= "matrikel_nr='" . $formVars['matrikelNrNeu'] . "', ";
364             $sql .= "vorname='" . $formVars['vornameNeu'] . "', ";
365             $sql .= "nachname='" . $formVars['nameNeu'] . "', ";
366             $sql .= "email='" . $formVars['mailNeu'] . "', ";
367             $sql .= "login='" . $formVars['loginNeu'] . "' ";
368
369             if ($formVars['userTypeNeu'] == "admin")
370                 $sql .= ", typ='admin' ";
371             elseif ($formVars['userTypeNeu'] == "dozent")
372                 $sql .= ", typ='dozent' ";
373             else
374                 $sql .= ", typ='user' ";
375
376             if ($formVars['secretNeu'] == "ja")
377                 $sql .= ", freischalt_secret='bestaetigt' ";
378             else
379                 $sql .= ", freischalt_secret='" . $formVars['secret'] . "' ";
380
381             if (!empty($formVars['passwortNeu']))
382                 $sql .= ", passwort='" . $cryptPass . "' ";
383
384             $sql .= "WHERE matrikel_nr='" . $formVars['matrikelNr'] . "' ";
385
386             if ($debug)
387                 echo "SQL: " . $sql . "<br>\n";

```

```

388         setSessionVars();
390
391         $msg = "Bei Schreiben in die Datenbank ist ein Fehler aufgetreten.<br>".
392             "Wenden Sie sich an Ihren Administrator.";
394
395         if (!$debug)
396         {
397             $result = executeStatement($connection, $sql, $msg);
398
399             if ($HTTP_SESSION_VARS['login'] == $HTTP_POST_VARS['login'])
400                 $HTTP_SESSION_VARS['login'] = $formVars['loginNeu'];
402
403             header ("Location: " . $projectPath . 'index_session.php' .
404                 "?menu=admin&content=userdata/updated&PHPSESSID=" . session_id());
405         }
406     }
407 }
408 }
409 }
410 }
411 }
412 }
413 }
414 }
415 }
416 }
417 }
418 }
419 }
420 }
421 }
422 }
423 }
424 }
425 }
426 }
427 }
428 }
429 }
430 }
431 }
432 }
433 }
434 }
435 }
436 }
437 }
438 }
439 }
440 }
441 }
442 }
443 }
444 }
445 }
446 }
447 }
448 }
449 }
450 }
451 }
452 }
453 }
454 }
455 }
456 }
457 }
458 }
459 }
460 }
461 }
462 }
463 }
464 }
465 }
466 }
467 }
468 }
469 }
470 }
471 }
472 }
473 }
474 }
475 }
476 }
477 }
478 }
479 }
480 }
481 }
482 }
483 }
484 }
485 }
486 }
487 }
488 }
489 }
490 }
491 }
492 }
493 }
494 }
495 }
496 }
497 }
498 }
499 }
500 }
501 }
502 }
503 }
504 }
505 }
506 }
507 }
508 }
509 }
510 }
511 }
512 }
513 }
514 }
515 }
516 }
517 }
518 }
519 }
520 }
521 }
522 }
523 }
524 }
525 }
526 }
527 }
528 }
529 }
530 }
531 }
532 }
533 }
534 }
535 }
536 }
537 }
538 }
539 }
540 }
541 }
542 }
543 }
544 }
545 }
546 }
547 }
548 }
549 }
550 }
551 }
552 }
553 }
554 }
555 }
556 }
557 }
558 }
559 }
560 }
561 }
562 }
563 }
564 }
565 }
566 }
567 }
568 }
569 }
570 }
571 }
572 }
573 }
574 }
575 }
576 }
577 }
578 }
579 }
580 }
581 }
582 }
583 }
584 }
585 }
586 }
587 }
588 }
589 }
590 }
591 }
592 }
593 }
594 }
595 }
596 }
597 }
598 }
599 }
600 }
601 }
602 }
603 }
604 }
605 }
606 }
607 }
608 }
609 }
610 }
611 }
612 }
613 }
614 }
615 }
616 }
617 }
618 }
619 }
620 }
621 }
622 }
623 }
624 }
625 }
626 }
627 }
628 }
629 }
630 }
631 }
632 }
633 }
634 }
635 }
636 }
637 }
638 }
639 }
640 }
641 }
642 }
643 }
644 }
645 }
646 }
647 }
648 }
649 }
650 }
651 }
652 }
653 }
654 }
655 }
656 }
657 }
658 }
659 }
660 }
661 }
662 }
663 }
664 }
665 }
666 }
667 }
668 }
669 }
670 }
671 }
672 }
673 }
674 }
675 }
676 }
677 }
678 }
679 }
680 }
681 }
682 }
683 }
684 }
685 }
686 }
687 }
688 }
689 }
690 }
691 }
692 }
693 }
694 }
695 }
696 }
697 }
698 }
699 }
700 }
701 }
702 }
703 }
704 }
705 }
706 }
707 }
708 }
709 }
710 }
711 }
712 }
713 }
714 }
715 }
716 }
717 }
718 }
719 }
720 }
721 }
722 }
723 }
724 }
725 }
726 }
727 }
728 }
729 }
730 }
731 }
732 }
733 }
734 }
735 }
736 }
737 }
738 }
739 }
740 }
741 }
742 }
743 }
744 }
745 }
746 }
747 }
748 }
749 }
750 }
751 }
752 }
753 }
754 }
755 }
756 }
757 }
758 }
759 }
760 }
761 }
762 }
763 }
764 }
765 }
766 }
767 }
768 }
769 }
770 }
771 }
772 }
773 }
774 }
775 }
776 }
777 }
778 }
779 }
780 }
781 }
782 }
783 }
784 }
785 }
786 }
787 }
788 }
789 }
790 }
791 }
792 }
793 }
794 }
795 }
796 }
797 }
798 }
799 }
800 }
801 }
802 }
803 }
804 }
805 }
806 }
807 }
808 }
809 }
810 }
811 }
812 }
813 }
814 }
815 }
816 }
817 }
818 }
819 }
820 }
821 }
822 }
823 }
824 }
825 }
826 }
827 }
828 }
829 }
830 }
831 }
832 }
833 }
834 }
835 }
836 }
837 }
838 }
839 }
840 }
841 }
842 }
843 }
844 }
845 }
846 }
847 }
848 }
849 }
850 }
851 }
852 }
853 }
854 }
855 }
856 }
857 }
858 }
859 }
860 }
861 }
862 }
863 }
864 }
865 }
866 }
867 }
868 }
869 }
870 }
871 }
872 }
873 }
874 }
875 }
876 }
877 }
878 }
879 }
880 }
881 }
882 }
883 }
884 }
885 }
886 }
887 }
888 }
889 }
890 }
891 }
892 }
893 }
894 }
895 }
896 }
897 }
898 }
899 }
900 }
901 }
902 }
903 }
904 }
905 }
906 }
907 }
908 }
909 }
910 }
911 }
912 }
913 }
914 }
915 }
916 }
917 }
918 }
919 }
920 }
921 }
922 }
923 }
924 }
925 }
926 }
927 }
928 }
929 }
930 }
931 }
932 }
933 }
934 }
935 }
936 }
937 }
938 }
939 }
940 }
941 }
942 }
943 }
944 }
945 }
946 }
947 }
948 }
949 }
950 }
951 }
952 }
953 }
954 }
955 }
956 }
957 }
958 }
959 }
960 }
961 }
962 }
963 }
964 }
965 }
966 }
967 }
968 }
969 }
970 }
971 }
972 }
973 }
974 }
975 }
976 }
977 }
978 }
979 }
980 }
981 }
982 }
983 }
984 }
985 }
986 }
987 }
988 }
989 }
990 }
991 }
992 }
993 }
994 }
995 }
996 }
997 }
998 }
999 }
1000 }

```

Listing E.42: admin/userdata/updated.php

```

1  <?php
2
3      /*
4      * admin/userdata/updated.php
5      */
6
7      ?>
8
9
10     <div align="center">
11         <p></p>
12         <p class="head">Benutzerdaten</p>
13     </div>
14
15     <div align="center">
16         <br><br>
17         <p class="border">
18             <table>
19                 <tr>
20                     <td>
21                         <div align="center">
22                             <p class="text">
23                                 Ihre Daten wurden erfolgreich ge&auml;ndert.
24                             </p>
25                         </div>
26                     </td>
27                 </tr>
28             </table>
29         </p>
30     </div>

```


Listing E.43: admin/userdata/delete.php

```
<?php
2
3     /*
4     * admin/userdata/delete.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['matrikelNr']
15    // $HTTP_GET_VARS['PHPSESSID']
16    // $HTTP_GET_VARS['sort']
17    // $HTTP_GET_VARS['find']
18    // $HTTP_GET_VARS['startRecord']
19    //
20    // SESSION
21    // -----
22    // beim ersten Aufruf:
23    // $HTTP_SESSION_VARS['projectPath']
24    // $HTTP_SESSION_VARS['absoluteProjectPath']
25    // $HTTP_SESSION_VARS['errors']
26    // $HTTP_SESSION_VARS['auth']
27    // $HTTP_SESSION_VARS['login']
28    // $HTTP_SESSION_VARS['menu']
29
30    $debug = false;
31
32    // hole Session-Variable
33    if (isset($HTTP_GET_VARS["PHPSESSID"]))
34        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
35    else
36        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
37
38    // hole vorhandene Session
39    session_start();
40    $PHPSESSID=session_id();
41
42    // falls Session verfuegbar, hole Session-Variablen
43    if (!empty($PHPSESSID))
44    {
45        if (isset($HTTP_SESSION_VARS["projectPath"]))
46            $projectPath = $HTTP_SESSION_VARS["projectPath"];
47
48        if (isset($HTTP_SESSION_VARS["absoluteProjectPath"]))
49            $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
50    }
51
52    require_once $absoluteProjectPath . "functions.php";
53
54    // Fehlerbehandlung
55    ob_start('ob_gzhandler');
56    set_error_handler('handleErrors');
57    $scriptName = "admin/userdata/delete.php";
```

```

58     if ($debug)
60     {
        echo "Session: " . session_id() . "<br>";
62        showSessionVar();
        echo "Post:<br>";
64        showPostVar();
        echo "Get:<br>";
66        showGetVar();
    }

68    // baue Verbindung zur Datenbank auf
70    $connection = getConnection();

72    //////////////////////////////////////
74    // hole passenden Datensatz zur MatrikelNr

    if (isset($_HTTP_GET_VARS['matrikelNr']))
76        $sql = "SELECT * FROM benutzer WHERE matrikel_nr=" .
            $_HTTP_GET_VARS['matrikelNr'];
78    else
        $sql = "";
80
    if ($debug)
82        echo "sql: " . $sql . "<br>\n";

84    $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>" .
        "Wenden Sie sich an Ihren Administrator.";
86
88    $result = executeStatement($connection, $sql, $msg);

90    $array = fetchArray($result, 0);

92    if ($debug)
        print_r ($array);

94    //////////////////////////////////////
96    // Funktion zum Ueberpruefen,
    // ob der zu loeschende Datensatz der Eigene ist.

98    function checkOwner()
    {
100        global $array, $_HTTP_SESSION_VARS;
        global $projectPath, $menu;

102
        if ($array["login"] == $_HTTP_SESSION_VARS["login"])
104        {
            header ("Location: " . $projectPath .
106                "index_session.php?menu=admin&content=userdata/deleteMessage&PHPSESSID=" .
                session_id());
108        }
    }

110
112    ?>

114    <div align="center">
        <p></p>
        <p class="head">Datensatz löschen</p>

```

```

116     <?php checkOwner(); ?>
118
119     <p class="text">
120         Der folgende Datensatz wird unwiederruflich gel&ouml;scht !!
121     </p>
122
123     <form method="post" action="<?php echo $projectPath . 'admin/userdata/delete2.php' ?>">
124     <p class="text">
125     <table border="0" cellpadding='2'>
126
127 <?php
128     if (!empty($PHPSESSID))
129         echo "<input type='hidden' name='PHPSESSID' value='" . $PHPSESSID . "'>\n";
130
131     echo "<input type='hidden' name='matrikelNr' value='" . $array['matrikel_nr'] . "'>\n";
132
133     if (isset($_HTTP_GET_VARS['sort']))
134         echo "<input type='hidden' name='sort' value='" .
135             $_HTTP_GET_VARS['sort'] . "'>\n";
136     if (isset($_HTTP_GET_VARS['find']))
137         echo "<input type='hidden' name='find' value='" .
138             $_HTTP_GET_VARS['find'] . "'>\n";
139     if (isset($_HTTP_GET_VARS['startRecord']))
140         echo "<input type='hidden' name='startRecord' value='" .
141             $_HTTP_GET_VARS['startRecord'] . "'>\n";
142     ?>
143
144     <tr >
145         <th class='vulab'>MatrikelNr</th>
146         <th class='vulab'>Name</th>
147         <th class='vulab'>Vorname</th>
148         <th class='vulab'>login</th>
149     </tr>
150     <tr>
151         <td class='vulab' align="right"> <?php echo $array['matrikel_nr'] ?> </td>
152         <td class='vulab'> <?php echo $array['nachname'] ?> </td>
153         <td class='vulab'> <?php echo $array['vorname'] ?> </td>
154         <td class='vulab'> <?php echo $array['login'] ?> </td>
155     </tr>
156 </table></p>
157
158 <?php
159     if ($array['typ'] == 'admin')
160     {
161         echo "<p class=\"error\">\n";
162         echo "<b>Achtung: </b>Bei dem zu l&ouml;schenden Profil " .
163             "handelt es sich um einen Administrator !!\n";
164         echo "</p>\n";
165     }
166     ?>
167
168     <p class="text">
169         Wollen Sie fortfahren ?
170     </p>
171
172     <input type="submit" name="submitJa" value="ja">&nbsp;
173     <input type="submit" name="submitNein" value="nein">&nbsp;

```

```

174     </form>
175
176 </div>
177 <div align="left">
178     <p class="block">    </p>
179 </div>

```

Listing E.44: admin/userdata/delete2.php

```

<?php
2
3     /*
4     * admin/userdata/delete2.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // POST
11    // ----
12    // $HTTP_GET_VARS['matrikelNr']
13    // $HTTP_GET_VARS['PHPSESSID']
14    // $HTTP_GET_VARS['submitNein'] oder $HTTP_GET_VARS['submitJa']
15    // $HTTP_POST_VARS['sort']
16    // $HTTP_POST_VARS['find']
17    // $HTTP_POST_VARS['startRecord']
18    //
19    // SESSION
20    // -----
21    // beim ersten Aufruf:
22    // $HTTP_SESSION_VARS['projectPath']
23    // $HTTP_SESSION_VARS['absoluteProjectPath']
24    // $HTTP_SESSION_VARS['errors']
25    // $HTTP_SESSION_VARS['auth']
26    // $HTTP_SESSION_VARS['login']
27    // $HTTP_SESSION_VARS['menu']
28
29    $debug = false;
30
31    // hole Session-Variable
32    if (isset($HTTP_GET_VARS["PHPSESSID"]))
33        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34    else
35        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
36
37    // hole vorhandene Session
38    session_start();
39    $PHPSESSID=session_id();
40
41    // falls Session verfuegbar, hole Session-Variablen
42    if (!empty($PHPSESSID))
43    {
44        if (isset($HTTP_SESSION_VARS["projectPath"]))
45            $projectPath = $HTTP_SESSION_VARS["projectPath"];

```

```

46         if (isset($_HTTP_SESSION_VARS["projectPath"]))
47             $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
48     }
49
50     require_once $absoluteProjectPath . "functions.php";
51
52     // Fehlerbehandlung
53     ob_start('ob_gzhandler');
54     set_error_handler('handleErrors');
55     $scriptName = "admin/userdata/delete2.php";
56
57     if ($debug)
58     {
59         echo "Session: " . session_id() . "<br>";
60         showSessionVar();
61         echo "Post:<br>";
62         showPostVar();
63         echo "Get:<br>";
64         showGetVar();
65     }
66
67     if (isset($_HTTP_POST_VARS['submitJa']))
68     {
69         // baue Verbindung zur Datenbank auf
70         $connection = getConnection();
71
72         if (isset($_HTTP_POST_VARS['matrikelNr']))
73             $sql = "DELETE FROM benutzer WHERE matrikel_nr=" .
74                 $_HTTP_POST_VARS['matrikelNr'];
75         else
76             $sql = "";
77
78         if ($debug)
79             echo "sql: " . $sql . "<br>\n";
80
81         $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
82             "Wenden Sie sich an Ihren Administrator.";
83
84         if(!$debug)
85             $result = executeStatement($connection, $sql, $msg);
86
87         if (!$debug)
88             header ("Location: " . $projectPath . 'index_session.php' .
89                 "?menu=admin&content=userdata/deleted&PHPSESSID=" .
90                 session_id());
91     }
92     else
93     {
94         $header = "Location: " . $projectPath . 'index_session.php' .
95             "?menu=admin&content=userdata/index&PHPSESSID=" .
96             session_id();
97
98         if (isset($_HTTP_POST_VARS['sort']))
99             $header .= "&sort=" . $_HTTP_POST_VARS['sort'] ;
100         if (isset($_HTTP_POST_VARS['find']))
101             $header .= "&find=" . $_HTTP_POST_VARS['find'] ;
102         if (isset($_HTTP_POST_VARS['startRecord']))

```

```

104         $header .= "&startRecord=" . $_HTTP_POST_VARS['startRecord'] ;
106         if (!$debug)
            header ($header);
108
109         if ($debug)
110             echo "Keine Aenderungen vorgenommen.";
112
113         exit;
114     }
115 }
116
117 ?>

```

Listing E.45: admin/userdata/deleteMessage.php

```

<?php
2
3     /*
4     * admin/userdata/deleteMessage.php
5     */
6
7     ?>
8
9     <div align="center">
10         <p></p>
11         <p class="head">Datensatz l&ouml;schen</p>
12     </div>
13     <div align="center">
14         <br><br>
15         <table>
16             <tr>
17                 <td>
18                     <div align="center">
19                         <p class="error">
20                             <b>Achtung: </b>Bei dem zu l&ouml;schenden
21                             Profil handelt es sich um Ihr Eigenes !!<br>
22                             Dieser Datensatz kann nicht gel&ouml;scht werden.
23
24                         </p>
25                     </div>
26                 </td>
27             </tr>
28         </table>
29     </div>

```

Listing E.46: admin/userdata/deleted.php

```
<?php
2
3     /*
4     * admin/userdata/deleted.php
5     */
6
7     ?>
8
9     <div align="center">
10         <p></p>
11         <p class="head">Datensatz l&ouml;schen</p>
12     </div>
13     <div align="center">
14         <br><br>
15         <p class="border">
16             <table>
17                 <tr>
18                     <td>
19                         <div align="center">
20                             <p class="text">
21                                 Der Datensatz wurde erfolgreich gel&ouml;scht.
22                             </p>
23                         </div>
24                     </td>
25                 </tr>
26             </table>
27         </p>
28     </div>
```

E.4.3 Buchungen

Listing E.47: admin/buchungen/index.php

```
<?php
2
3     /*
4     * admin/buchungen/index.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ---
12    // $HTTP_GET_VARS['menu']
13    // $HTTP_GET_VARS['content']
14    // $HTTP_GET_VARS['PHPSESSID']
15    // $HTTP_GET_VARS['sort']
16    // $HTTP_GET_VARS['find']
17    // $HTTP_GET_VARS['startRecord']
```

```
18 //
19 // SESSION
20 // -----
21 // $HTTP_SESSION_VARS['projectPath']
22 // $HTTP_SESSION_VARS['absoluteProjectPath']
23 // $HTTP_SESSION_VARS['errors']
24 // $HTTP_SESSION_VARS['auth']
25 // $HTTP_SESSION_VARS['login']
26 // $HTTP_SESSION_VARS['menu']
27
28 $debug = false;
29
30 // hole Session-Variable
31 if (isset($HTTP_GET_VARS["PHPSESSID"]))
32     $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
33 else
34     $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
35
36 // hole vorhandene Session
37 session_start();
38 $PHPSESSID=session_id();
39
40 // falls Session verfuegbar, hole Session-Variablen
41 if (!empty($PHPSESSID))
42 {
43     if (isset($HTTP_SESSION_VARS["projectPath"]))
44         $projectPath = $HTTP_SESSION_VARS["projectPath"];
45
46     if (isset($HTTP_SESSION_VARS["absoluteProjectPath"]))
47         $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
48 }
49
50 if ($debug)
51 {
52     echo $projectPath . "<br>";
53     echo $absoluteProjectPath . "<br>";
54 }
55
56 require_once $absoluteProjectPath . "functions.php";
57 require_once $absoluteProjectPath . "config.php";
58
59 // Fehlerbehandlung
60 ob_start('ob_gzhandler');
61 set_error_handler('handleErrors');
62 $scriptName = "admin/buchungen/index.php";
63
64 if ($debug)
65 {
66     echo "Session: " . session_id() . "<br>";
67     showSessionVar();
68     echo "Post:<br>";
69     showPostVar();
70     echo "Get:<br>";
71     showGetVar();
72 }
73
74 // zerlegt Datum
75 // 2003-05-16 --> 16.05.2003
```



```

76 function makeDate($date)
77 {
78     $pos = (int) strrpos($date, '-');
79     $day = substr($date, $pos + 1);
80     $tmp = substr($date, 0, $pos);
81     $pos = (int) strrpos($tmp, '-');
82     $month = substr($tmp, $pos + 1);
83     $year = substr($tmp, 0, $pos);
84
85     return $day . "." . $month . "." . $year;
86 }
87
88 // baue Verbindung zur Datenbank auf
89 $connection = getConnection();
90
91 // Variable fuer Such-Abfragen
92 $find = getValue("GET", "find");
93 $finder = "";
94
95 $sqlFirst = "SELECT * FROM buchungen, uebungen, benutzer " .
96     "WHERE uebungen.uebung_id=buchungen.uebung_id AND " .
97     "buchungen.user_id=benutzer.user_id ";
98
99 // Such-Abfrage zusammensetzen
100 if (!empty($find))
101     $finder = "AND (upper(buchungen.uebung_id) LIKE upper('%" . $find . "%') OR " .
102         "upper(bezeichnung) LIKE upper('%" . $find . "%') OR " .
103         "upper(freigegeben) LIKE upper('%" . $find . "%') OR " .
104         "datum LIKE '%" . $find . "%' OR " .
105         "int4(buchungs_id) LIKE '%" . $find . "%' OR " .
106         "int4(at_id) LIKE '%" . $find . "%' OR " .
107         "startzeit LIKE '%" . $find . "%' OR " .
108         "dauer LIKE '%" . $find . "%')";
109
110 // SQL-Abfrage zusammensetzen
111 if (isset($HTTP_GET_VARS['sort']) &&
112     ($HTTP_GET_VARS['sort'] != "dauer") &&
113     ($HTTP_GET_VARS['sort'] != "datum") &&
114     ($HTTP_GET_VARS['sort'] != "at_id") &&
115     ($HTTP_GET_VARS['sort'] != "startzeit"))
116     $sql = $sqlFirst . $finder .
117         " ORDER BY LOWER(" . $HTTP_GET_VARS['sort'] . ") ";
118 elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "dauer")
119     $sql = $sqlFirst . $finder . " ORDER BY dauer ";
120 elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "datum")
121     $sql = $sqlFirst . $finder . " ORDER BY datum ";
122 elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "at_id")
123     $sql = $sqlFirst . $finder . " ORDER BY at_id ";
124 elseif (isset($HTTP_GET_VARS['sort']) && $HTTP_GET_VARS['sort'] == "startzeit")
125     $sql = $sqlFirst . $finder . " ORDER BY startzeit ";
126 else
127 {
128     $sql = $sqlFirst . $finder .
129         " ORDER BY datum, startzeit ";
130     $HTTP_GET_VARS['sort'] = "datum";
131 }
132
133 if ($debug)

```

```

134         echo "SQL: " . $sql . "<br>";

136     $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
137         "Wenden Sie sich an Ihren Administrator.";

138
139     $result = executeStatement($connection, $sql, $msg);

140
141     if ($debug)
142         echo "Result: " . $result . "<br>";

143
144     //////////////////////////////////////
145     // blaettern in den Datensatzen
146     //
147     // !! zu setzende Variablen siehe $absoluteProjectPath/config.php !!
148
149     // mit welchem Datensatz wird begonnen?
150     $startRecord = getValue("GET", "startRecord");

151
152     if (empty($startRecord))
153         $startRecord = 0;

154
155     // ermittle letzten Datensatz
156     $numrows = pg_numrows($result) - 1;

157
158     if ( ($numrows + 1 - $startRecord) < $number)
159         $stopRecord = $numrows + 1;
160     else
161         $stopRecord = $startRecord + $number;

162
163     // korrigiere $startRecord
164     if ($startRecord > $numrows)
165         $startRecord = $numrows;

166
167     if ( ($corr = $startRecord % $number) != 0)
168         $startRecord = $startRecord - $corr;

169
170     if ( $startRecord < 0)
171         $startRecord = 0;

172
173     // Teil-URL zum Sortieren
174     $sort = "?menu=admin&content=buchungen/index";

175
176     if (!empty($find))
177         $sort = $sort . "&find=$find";

178
179     $sort = $sort . "&PHPSESSID=$PHPSESSID&sort=";

180     $sortLogin = $sort . "benutzer.login&startRecord=" . $startRecord;
181     $sortKurzBezeichnung = $sort . "buchungen.uebung_id&startRecord=" . $startRecord;
182     $sortBezeichnung = $sort . "bezeichnung&startRecord=" . $startRecord;
183     $sortDatum = $sort . "datum&startRecord=" . $startRecord;
184     $sortStartzeit = $sort . "startzeit&startRecord=" . $startRecord;
185     $sortDauer = $sort . "dauer&startRecord=" . $startRecord;
186     $sortWiederholbar = $sort . "wiederholbar&startRecord=" . $startRecord;
187     $sortFreigegeben = $sort . "freigegeben&startRecord=" . $startRecord;
188     $sortAt_id = $sort . "at_id&startRecord=" . $startRecord;
189     $sortBuchungs_id = $sort . "buchungs_id&startRecord=" . $startRecord;

190
191     // Anzahl der Seiten

```

```

192     $pages = intval(($numrows + 1) / $number);
194     if ( (($numrows + 1) % $number) > 0 )
196         $pages = $pages + 1;
198     // aktuelle Seite
199     $currentPage = -1 ;
201     for ($i = 0; $i <= $startRecord; $i = $i + $number)
202         $currentPage = $currentPage + 1;
204     // erste Seite
205     $firstPage = $currentPage - intval($linkNumber / 2);
207     // letzte verfuegbare Seite
208     $tmp = $numrows % $number;
209     $tmp = $numrows - $tmp;
210     $lastPage = ($tmp / $number);
212     if ( ($lastPage - $currentPage) < intval($linkNumber / 2))
213         $firstPage = $lastPage - ($linkNumber - 1);
215     if ($firstPage < 0)
216         $firstPage = 0;
218     // letzte anzuzeigende Seite
219     if ($pages < $linkNumber)
220         $stopPage = $pages;
221     else
222         $stopPage = $linkNumber;
224     if ($debug)
225     {
226         echo "anzuzeitende Datensatze: $number <br>";
227         echo "Anzahl der Links: $linkNumber <br>";
228         echo "first Page: $firstPage <br>";
229         echo "current Page: $currentPage <br>";
230         echo "last Page: $lastPage <br>";
231         echo "stop Page: $stopPage <br>";
232     }
234     // liefert eine HTML-Zeile zum blaettern in den Datensatzen
235     function browse($count)
236     {
237         global $linkBack, $linkForward, $startRecord, $stopRecord;
238         global $number, $result, $sort, $HTTP_GET_VARS;
239         global $linkNumber, $currentPage, $firstPage;
240         global $numrows, $stopPage, $projectPath;
242         if ( $startRecord < $number)
243             $linkBack = "";
244         else
245         {
246             $linkBack = "<a href='" . $projectPath . 'index_session.php' .
247                 $sort . $HTTP_GET_VARS['sort'] . "&startRecord=" .
248                 ( $startRecord - $number ) . "' " .
249                 "onmouseout=\"changeImage('imageBack$count', 'outBack');\" " .
250                 "onmouseover=\"changeImage('imageBack$count', 'overBack');\" " .

```

```

250     "onmousedown=\"changeImage('imageBack$count', 'downBack');\">" .
251     "<img src='images/back1.gif' width='30' height='26' border='0' " .
252     "alt='zur&uuml;ck' name='imageBack$count'></a>\n";
253 }
254
255 if ( $stopRecord >= ($numrows + 1))
256     $linkForward = "";
257 else
258 {
259     $linkForward = "<a href='" . $projectPath . 'index_session.php' .
260     $sort . $_HTTP_GET_VARS['sort'] . "&startRecord=" .
261     ($startRecord + $number) . "' " .
262     "onmouseout=\"changeImage('imageForward$count', 'outForward');\" " .
263     "onmouseover=\"changeImage('imageForward$count', 'overForward');\" " .
264     "onmousedown=\"changeImage('imageForward$count', 'downForward');\">" .
265     "<img src='images/forward1.gif' width='30' height='26' border='0' " .
266     "alt='weiter' name='imageForward$count'></a>\n";
267 }
268
269 $links = "";
270
271 $i = $firstPage;
272
273 do {
274     if ( ($i * $number) == $startRecord)
275         $links = $links . "&nbsp;" . ($startRecord + 1). "-" .
276         $stopRecord . "&nbsp;";
277     else
278         $links = $links . "&nbsp;" .
279         "<a href='" . $projectPath . 'index_session.php' .
280         $sort . $_HTTP_GET_VARS['sort'] . "&startRecord=" .
281         ($i * $number) . "'>" .
282         (($i * $number) + 1). "-" . (($i * $number) + $number) .
283         "</a>&nbsp;";
284
285     $i++;
286 } while ($i < $stopPage + $firstPage);
287
288 $links = $links . "&nbsp;";
289
290 // Ausgabe zusammensetzen
291 $string = "<table border='0' cellpadding='2' >" .
292 "<tr>\n" .
293 "<td align='right' width='40'>" .
294 $linkBack .
295 "</td>\n" .
296 "<td align='center' width='300'><p class='text'>" .
297 // "Datensatz " . $startRecord . " - " . $stopRecord .
298 $links .
299 "</p></td>\n" .
300 "<td align='left' width='40'>" .
301 $linkForward .
302 "</td>\n" .
303 "</tr>\n" .
304 "</table>";
305
306 return $string;
307 }

```

```

308 ?>

310 <script type="text/javascript">
var navName = navigator.appName;
312 var version = parseInt(navigator.appVersion);
var browserOk = (((navName == "Netscape") && (version >= 3)) ||
314 ((navName == "Microsoft Internet Explorer") && (version >= 4)));

316 function changeImage(imageName, newSrc) {
    // Changes images
318     if (browserOk) {
        document.images[imageName].src = eval(newSrc + ".src");
320     }
    }
322
if (browserOk) {
324     // Image Loading Start
    outDelete = new Image();
326     outEdit = new Image();
    outBack = new Image();
328     outForward = new Image();
    outDelete.src = "images/deletel.gif";
330     outEdit.src = "images/editl.gif";
    outBack.src = "images/backl.gif";
332     outForward.src = "images/forwardl.gif";

334     overDelete = new Image();
    overEdit = new Image();
336     overBack = new Image();
    overForward = new Image();
338     overDelete.src = "images/deletel.gif";
    overEdit.src = "images/editl.gif";
340     overBack.src = "images/backl.gif";
    overForward.src = "images/forwardl.gif";
342

    downDelete = new Image();
344     downEdit = new Image();
    downBack = new Image();
346     downForward = new Image();
    downDelete.src = "images/delete2.gif";
348     downEdit.src = "images/edit2.gif";
    downBack.src = "images/back2.gif";
350     downForward.src = "images/forward2.gif";
    }
352 </script>

354 <div align="center">
    <p class="head">Verwaltung gebuchter &Uuml;bungen</p>
356 <p></p>

358 <!-- Stichwort-Suche -->

360 <p class='border'>
    <table>
362 <tr><td align="center" valign="middle" >
    <form name="" action="<?php echo $projectPath . "index_session.php" ?>" method="get">
364 <p class='text'>&nbsp;Stichwort-Suche:&nbsp;
    <input type="hidden" name="menu" value="admin">

```

```

366         <input type="hidden" name="content" value="buchungen/index">
<?php
368     if (!empty($find))
        echo "<input type=\"hidden\" name=\"find\" value=\"\" . $find . \">\n";
370     if (!empty($PHPSESSID))
        echo "<input type=\"hidden\" name=\"PHPSESSID\" value=\"\" . $PHPSESSID . \">\n";
372 ?>

        <input type="text" name="find" size="20" value="<?php echo $find; ?>">&nbsp;
374         <input type="submit" name="" value="Suchen"> &nbsp;
        </form></p>
376     </td></tr>
    </table>
378 </p>

380 <!-- Such-Ergebnisse -->

382 <p></p>
<?php
384
    echo "<p class='error'>";
386    echo "Achtung: Bei Buchungen in roter Farbe, " .
        "war das Erstellen des at-Jobs fehlerhaft !!";
388    echo "</p>";

390    echo "<p class='text'>";
    echo "Vorhandene Buchungen: " . ($numrows + 1) . "<br>";
392    echo browse(1);
    echo "</p>";
394

    echo "<table border='0' cellpadding='2'>";

396

    echo "<tr>\n";
    echo "<th class='vulab'><a href=\"\" .
        $projectPath . 'index_session.php' . $sortBuchungs_id . \">ID</a></th>\n";
400    echo "<th class='vulab'><a href=\"\" .
        $projectPath . 'index_session.php' . $sortLogin . \">Login</a></th>\n";
402    echo "<th class='vulab'><a href=\"\" .
        $projectPath . 'index_session.php' . $sortKurzBezeichnung . \">Kurzbez.</a></th>\n";
404    echo "<th class='vulab'><a href=\"\" .
        $projectPath . 'index_session.php' . $sortBezeichnung . \">Bezeichnung</a></th>\n";
406    echo "<th class='vulab' width=\"80\"><a href=\"\" .
        $projectPath . 'index_session.php' . $sortDatum . \">Datum</a></th>\n";
408    echo "<th class='vulab' width=\"70\"><a href=\"\" .
        $projectPath . 'index_session.php' . $sortStartzeit . \">Startzeit</a></th>\n";
410    echo "<th class='vulab' width=\"40\"><a href=\"\" .
        $projectPath . 'index_session.php' . $sortDauer . \">Dauer</a></th>\n";
412    echo "<th class='vulab'><a href=\"\" .
        $projectPath . 'index_session.php' . $sortFreigegeben . \">freigegeben</a></th>\n";
414    echo "<th class='vulab'><a href=\"\" .
        $projectPath . 'index_session.php' . $sortAt_id . \">at_id</a></th>\n";
416    echo "</tr>\n";

418    for ($i = $startRecord; $i < $stopRecord; $i++)
    {
420        $array = fetchArray($result, $i);

422        if ( empty($array['at_id']) && $array['freigegeben'] == "nein")
            $class = "error";

```

```

424     else
425         $class = "vulab";
426
427     echo "<tr>\n";
428     echo "<td class='\" . $class . "\">\" . $array['buchungs_id'] . "</td>\n";
429     echo "<td class='\" . $class . "\">\" . $array['login'] . "</td>\n";
430     echo "<td class='\" . $class . "\">\" . $array['uebung_id'] . "</td>\n";
431     echo "<td class='\" . $class . "\">\" . $array['bezeichnung'] . "</td>\n";
432     echo "<td class='\" . $class . "\" align='right'>\" .
433         makeDate($array['datum']) . "</td>\n";
434     echo "<td class='\" . $class . "\" align='right'>\" .
435         cutTime($array['startzeit']) . " Uhr</td>\n";
436     echo "<td class='\" . $class . "\" align='right'>\" .
437         cutTime($array['dauer']) . "</td>\n";
438     echo "<td class='\" . $class . "\" align='center'>\" .
439         $array['freigegeben'] . "</td>\n";
440     echo "<td class='\" . $class . "\" align='center'>\" .
441         $array['at_id'] . "</td>\n";
442
443     echo "<td class='vulab'>\n";
444     echo "<a href='\" . $projectPath . 'index_session.php' .
445         "?menu=admin&content=buchungen/show&at_id=" .
446         $array['at_id'] . "&PHPSESSID=" . $PHPSESSID . "\" ;
447
448     if (isset($HTTP_GET_VARS['sort']))
449         echo "&sort=" . $HTTP_GET_VARS['sort'] ;
450     if (isset($HTTP_GET_VARS['find']))
451         echo "&find=" . $HTTP_GET_VARS['find'] ;
452     if (isset($HTTP_GET_VARS['startRecord']))
453         echo "&startRecord=" . $HTTP_GET_VARS['startRecord'] ;
454
455     echo "\" onmouseout=\"changeImage('imageEdit$i', 'outEdit');\"\" .
456         " onmouseover=\"changeImage('imageEdit$i', 'overEdit');\"\" .
457         " onmousedown=\"changeImage('imageEdit$i', 'downEdit');\">\" .
458         "<img src='images/edit1.gif' width='30' height='26' border='0' \" .
459         "alt='at-Job einsehen' name='imageEdit$i'></a>\n";
460
461     echo "<a href='\" . $projectPath . 'index_session.php' .
462         "?menu=admin&content=buchungen/delete&at_id=" .
463         $array['at_id'] . "&buchungs_id=" . $array['buchungs_id'] .
464         "&PHPSESSID=" . $PHPSESSID . "\" ;
465
466     if (isset($HTTP_GET_VARS['sort']))
467         echo "&sort=" . $HTTP_GET_VARS['sort'] ;
468     if (isset($HTTP_GET_VARS['find']))
469         echo "&find=" . $HTTP_GET_VARS['find'] ;
470     if (isset($HTTP_GET_VARS['startRecord']))
471         echo "&startRecord=" . $HTTP_GET_VARS['startRecord'] ;
472
473     echo "\" onmouseout=\"changeImage('imageDelete$i', 'outDelete');\"\" .
474         " onmouseover=\"changeImage('imageDelete$i', 'overDelete');\"\" .
475         " onmousedown=\"changeImage('imageDelete$i', 'downDelete');\">\" .
476         "<img src='images/delete1.gif' width='30' height='26' border='0' \" .
477         "alt='Buchung löschen' name='imageDelete$i'></a>\n";
478     echo "</td>\n";
479     echo "</tr>\n";
480 }

```

```
482 ?>
484     </table></p>
486 </div>
486 <?php
488     echo "<p class='text'>";
488     echo browse(2);
490     echo "</p>";
490 ?>
```

Listing E.48: admin/buchungen/show.php

```
<?php
2
3     /*
4     * admin/buchungen/show.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS[ 'PHPSESSID' ]
13    // $HTTP_GET_VARS[ 'menu' ]
14    // $HTTP_GET_VARS[ 'content' ]
15    // $HTTP_GET_VARS[ 'at_id' ]
16    // $HTTP_GET_VARS[ 'sort' ]
17    // $HTTP_GET_VARS[ 'find' ]
18    // $HTTP_GET_VARS[ 'startRecord' ]
19    //
20    // SESSION
21    // -----
22    // $HTTP_SESSION_VARS[ 'projectPath' ]
23    // $HTTP_SESSION_VARS[ 'absoluteProjectPath' ]
24    // $HTTP_SESSION_VARS[ 'errors' ]
25    // $HTTP_SESSION_VARS[ 'auth' ]
26    // $HTTP_SESSION_VARS[ 'login' ]
27    // $HTTP_SESSION_VARS[ 'menu' ]
28
29    $debug = false;
30
31    // hole Session-Variable
32    if (isset($HTTP_GET_VARS[ "PHPSESSID" ]))
33        $PHPSESSID = $HTTP_GET_VARS[ "PHPSESSID" ];
34    else
35        $PHPSESSID = $HTTP_POST_VARS[ "PHPSESSID" ];
36
37    // hole vorhandene Session
38    session_start();
39    $PHPSESSID=session_id();
40
41    // falls Session verfuegbar, hole Session-Variablen
```



```

42  if (!empty($PHPSESSID))
43  {
44      if (isset($_HTTP_SESSION_VARS["projectPath"]))
45          $projectPath = $_HTTP_SESSION_VARS["projectPath"];
46
47      if (isset($_HTTP_SESSION_VARS["projectPath"]))
48          $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
49  }
50
51  require_once $absoluteProjectPath . "functions.php";
52
53  // Fehlerbehandlung
54  ob_start('ob_gzhandler');
55  set_error_handler('handleErrors');
56  $scriptName = "admin/buchen/show.php";
57
58  if ($debug)
59  {
60      echo "Session: " . session_id() . "<br>";
61      showSessionVar();
62      echo "Post:<br>";
63      showPostVar();
64      echo "Get:<br>";
65      showGetVar();
66  }
67
68  echo "<p class=\"head\">Verwaltung gebuchter &Uuml;bungen</p>\n";
69  echo "<p></p>\n";
70
71  echo "<div align=\"left\">\n";
72
73  echo "<table class='vulab' width='100%' border='0' cellpadding='10'>";
74  echo "<tr><td>";
75
76  $command = "at -c " . $_HTTP_GET_VARS['at_id'] . " 2>&1";
77  exec ($command, $result);
78  echo "<b>Kommando: </b><br>\n" . $command . "<br><br>\n";
79
80  $msg = "";
81  foreach ($result as $value)
82      $msg .= $value . "<br>\n";
83
84  echo "<b>Ausgabe: </b><br>\n" . $msg;
85
86  echo "<p>&nbsp;</p>\n";
87
88  echo "<a href=\" " . $projectPath . 'index_session.php' .
89      "?menu=admin&content=buchungen/index" .
90      "&PHPSESSID=" . $PHPSESSID . " \">";
91
92  if (isset($_HTTP_GET_VARS['sort']))
93      echo "&sort=" . $_HTTP_GET_VARS['sort'] ;
94  if (isset($_HTTP_GET_VARS['find']))
95      echo "&find=" . $_HTTP_GET_VARS['find'] ;
96  if (isset($_HTTP_GET_VARS['startRecord']))
97      echo "&startRecord=" . $_HTTP_GET_VARS['startRecord'] ;
98
99  echo ">zur&uuml;ck</a>\n";

```

```
100     echo "</td></tr>";
102     echo "</table>";
104     echo "</div>\n";
?>
```

Listing E.49: admin/buchungen/delete.php

```
<?php
2
3     /*
4     * admin/buchungen/delete.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_GET_VARS['PHPSESSID']
13    // $HTTP_GET_VARS['menu']
14    // $HTTP_GET_VARS['content']
15    // $HTTP_GET_VARS['at_id']
16    // $HTTP_GET_VARS['sort']
17    // $HTTP_GET_VARS['find']
18    // $HTTP_GET_VARS['startRecord']
19    //
20    // SESSION
21    // -----
22    // $HTTP_SESSION_VARS['projectPath']
23    // $HTTP_SESSION_VARS['absoluteProjectPath']
24    // $HTTP_SESSION_VARS['errors']
25    // $HTTP_SESSION_VARS['auth']
26    // $HTTP_SESSION_VARS['login']
27    // $HTTP_SESSION_VARS['menu']
28
29    $debug = false;
30
31    // hole Session-Variable
32    if (isset($HTTP_GET_VARS["PHPSESSID"]))
33        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34    else
35        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
36
37    // hole vorhandene Session
38    session_start();
39    $PHPSESSID=session_id();
40
41    // falls Session verfuegbar, hole Session-Variablen
42    if (!empty($PHPSESSID))
43    {
44        if (isset($HTTP_SESSION_VARS["projectPath"]))
45            $projectPath = $HTTP_SESSION_VARS["projectPath"];
```

```

46         if (isset($_HTTP_SESSION_VARS["projectPath"]))
47             $absoluteProjectPath = $_HTTP_SESSION_VARS["absoluteProjectPath"];
48     }
49
50     require_once $absoluteProjectPath . "functions.php";
51
52     // Fehlerbehandlung
53     ob_start('ob_gzhandler');
54     set_error_handler('handleErrors');
55     $scriptName = "admin/buchungen/delete.php";
56
57     if ($debug)
58     {
59         echo "Session: " . session_id() . "<br>";
60         showSessionVar();
61         echo "Post:<br>";
62         showPostVar();
63         echo "Get:<br>";
64         showGetVar();
65     }
66
67     // baue Verbindung zur Datenbank auf
68     $connection = getConnection();
69
70     //////////////////////////////////////
71     // hole passenden Datensatz zur MatrikelNr
72
73     if (isset($_HTTP_GET_VARS['buchungs_id']))
74         $sql = "SELECT * FROM buchungen, uebungen, benutzer " .
75             "WHERE uebungen.uebung_id=buchungen.uebung_id AND " .
76             "buchungen.user_id=benutzer.user_id AND buchungs_id=" .
77             $_HTTP_GET_VARS['buchungs_id'];
78     else
79         $sql = "";
80
81     if ($debug)
82         echo "sql: " . $sql . "<br>\n";
83
84     $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>" .
85         "Wenden Sie sich an Ihren Administrator.";
86
87     $result = executeStatement($connection, $sql, $msg);
88
89     $array = fetchArray($result, 0);
90
91     if ($debug)
92         print_r ($array);
93
94     ?>
95
96     <div align="center">
97         <p></p>
98         <p class="head">Buchung l&ouml;schen</p>
99
100         <p class="text">
101             Die folgende Buchung wird unwiederruflich gel&ouml;scht !!
102         </p>

```

```

104 <form method="post" action="<?php echo $projectPath .
      "index_session.php?menu=admin&content=buchungen/delete2" ?>">
106 <p class="text">
108 <table border='0' cellpadding='2'>
110 <?php
111     if (!empty($PHPSESSID))
112         echo "<input type='hidden' name='PHPSESSID' value='" . $PHPSESSID . "'>\n";
113     if (isset($_HTTP_GET_VARS['buchungs_id']))
114         echo "<input type='hidden' name='buchungs_id' value='" .
      $_HTTP_GET_VARS['buchungs_id'] . "'>\n";
115     if (isset($_HTTP_GET_VARS['at_id']))
116         echo "<input type='hidden' name='at_id' value='" .
      $_HTTP_GET_VARS['at_id'] . "'>\n";
117     if (isset($_HTTP_GET_VARS['sort']))
118         echo "<input type='hidden' name='sort' value='" .
      $_HTTP_GET_VARS['sort'] . "'>\n";
119     if (isset($_HTTP_GET_VARS['find']))
120         echo "<input type='hidden' name='find' value='" .
      $_HTTP_GET_VARS['find'] . "'>\n";
121     if (isset($_HTTP_GET_VARS['startRecord']))
122         echo "<input type='hidden' name='startRecord' value='" .
      $_HTTP_GET_VARS['startRecord'] . "'>\n";
123 ?>
124
125 <tr>
126     <th class="vulab">Buchungs_ID</th>
127     <th class="vulab">Login</th>
128     <th class="vulab">Kurzbezeichnung</th>
129     <th class="vulab">Datum</th>
130     <th class="vulab">Startzeit</th>
131     <th class="vulab">Dauer</th>
132     <th class="vulab">freigegeben</th>
133     <th class="vulab">at_id</th>
134 </tr>
135 <tr>
136     <td class="vulab"><?php echo $array['buchungs_id'] ?> </td>
137     <td class="vulab"><?php echo $array['login'] ?> </td>
138     <td class="vulab"><?php echo $array['uebung_id'] ?> </td>
139     <td class="vulab"><?php echo $array['datum'] ?> </td>
140     <td class="vulab"><?php echo $array['startzeit'] ?> </td>
141     <td class="vulab"><?php echo $array['dauer'] ?> </td>
142     <td class="vulab" align="center"><?php echo $array['freigegeben'] ?> </td>
143     <td class="vulab"><?php echo $array['at_id'] ?> </td>
144 </tr>
145 </table></p>
146
147 <p class="text">
148     Wollen Sie fortfahren ?
149 </p>
150
151 <input type="submit" name="submitJa" value="ja">&nbsp;
152 <input type="submit" name="submitNein" value="nein">&nbsp;
153
154 </form>
155 </div>

```

Listing E.50: admin/buchungen/delete2.php

```
<?php
2
3     /*
4     * admin/buchungen/delete2.php
5     */
6
7     // Verfuegbare Variablen
8     // =====
9     //
10    // GET
11    // ----
12    // $HTTP_POST_VARS['PHPSESSID']
13    // $HTTP_POST_VARS['menu']
14    // $HTTP_POST_VARS['content']
15    // $HTTP_POST_VARS['at_id']
16    // $HTTP_POST_VARS['sort']
17    // $HTTP_POST_VARS['find']
18    // $HTTP_POST_VARS['startRecord']
19    //
20    // SESSION
21    // -----
22    // $HTTP_SESSION_VARS['projectPath']
23    // $HTTP_SESSION_VARS['absoluteProjectPath']
24    // $HTTP_SESSION_VARS['errors']
25    // $HTTP_SESSION_VARS['auth']
26    // $HTTP_SESSION_VARS['login']
27    // $HTTP_SESSION_VARS['menu']
28
29    $debug = false;
30
31    // hole Session-Variable
32    if (isset($HTTP_GET_VARS["PHPSESSID"]))
33        $PHPSESSID = $HTTP_GET_VARS["PHPSESSID"];
34    else
35        $PHPSESSID = $HTTP_POST_VARS["PHPSESSID"];
36
37    // hole vorhandene Session
38    session_start();
39    $PHPSESSID=session_id();
40
41    // falls Session verfuegbar, hole Session-Variablen
42    if (!empty($PHPSESSID))
43    {
44        if (isset($HTTP_SESSION_VARS["projectPath"]))
45            $projectPath = $HTTP_SESSION_VARS["projectPath"];
46
47        if (isset($HTTP_SESSION_VARS["absoluteProjectPath"]))
48            $absoluteProjectPath = $HTTP_SESSION_VARS["absoluteProjectPath"];
49    }
50
51    require_once $absoluteProjectPath . "functions.php";
52
53    // Fehlerbehandlung
54    ob_start('ob_gzhandler');
55    set_error_handler('handleErrors');
56    $scriptName = "admin/buchen/delete2.php";
```

```

58     if ($debug)
59     {
60         echo "Session: " . session_id() . "<br>";
61         showSessionVar();
62         echo "Post:<br>";
63         showPostVar();
64         echo "Get:<br>";
65         showGetVar();
66     }

67     if (isset($_HTTP_POST_VARS['submitJa']))
68     {
69         // loesche at-Job
70         if (isset($_HTTP_POST_VARS['at_id']))
71         {
72             $command = "at -r " . $_HTTP_POST_VARS['at_id'] . " 2>&1";
73             exec ($command, $result);

74             if ($debug)
75             {
76                 echo "<b>Kommando: </b><br>\n" . $command . "<br><br>\n";

77                 $msg = "";
78                 foreach ($result as $value)
79                     $msg .= $value . "<br>\n";

80                 echo "<b>Ausgabe: </b><br>\n" . $msg;
81             }
82         }

83         // loesche Datensatz
84         $connection = getConnection();

85         $sql = "DELETE FROM buchungen WHERE buchungs_id='" .
86             $_HTTP_POST_VARS['buchungs_id'] . "'";

87         if ($debug)
88             echo "SQL: " . $sql . "<br>";

89         $msg = "Beim Zugriff auf die Datenbank ist ein Fehler aufgetreten.<br>".
90             "Wenden Sie sich an Ihren Administrator.";

91         $result = executeStatement($connection, $sql, $msg);

92         if ($debug)
93             echo "Result: " . $result . "<br>";

94         //////////////////////////////////////

95         $header = "Location: " . $projectPath . 'index_session.php' .
96             "?menu=admin&content=buchungen/deleted&PHPSESSID=" .
97             session_id();

98         if (!$debug)
99             header ($header);

100         if ($debug)
101             echo "Buchung geloescht.";

```

```

116         exit;
118     }
119     else
120     {
121         $header = "Location: " . $projectPath . 'index_session.php' .
122             "?menu=admin&content=buchungen/index&PHPSESSID=" .
123             session_id();
124
125         if (isset($_HTTP_POST_VARS['sort']))
126             $header .= "&sort=" . $_HTTP_POST_VARS['sort'] ;
127         if (isset($_HTTP_POST_VARS['find']))
128             $header .= "&find=" . $_HTTP_POST_VARS['find'] ;
129         if (isset($_HTTP_POST_VARS['startRecord']))
130             $header .= "&startRecord=" . $_HTTP_POST_VARS['startRecord'] ;
131
132         if (!$debug)
133             header ($header);
134
135         if ($debug)
136             echo "Keine Aenderungen vorgenommen.";
137
138         exit;
139     }
140     ?>

```

Listing E.51: admin/buchungen/deleted.php

```

<?php
2
3     /*
4     * admin/buchungen/deleted.php
5     */
6
7     ?>
8
9     <div align="center">
10         <p></p>
11         <p class="head">Buchung l&ouml;schen</p>
12     </div>
13     <div align="center">
14         <br><br>
15         <p class="border">
16             <table>
17                 <tr>
18                     <td>
19                         <div align="center">
20                             <p class="text">
21                                 Die Buchung wurde erfolgreich gel&ouml;scht.
22                             </p>
23                         </div>
24                     </td>
25                 </tr>
26             </table>

```

28

```
</p>  
</div>
```

E.4.4 Error

Der Quellcode des Moduls "Error" im Administrationsbereich ist identisch mit den Scripten des Moduls "Error" im Anwenderbereich (siehe Kapitel [E.3.5](#)).

E.5 Module im Dozentenbereich

Listing E.52: dozent/config.php

```
<?php
2
/*
4  * dozent/config.php
  */
6
// Menue-Eintraege in Reihenfolge ihres Auftretens
8 $items = array();
$items[1] = "home";
10 $items[2] = "Benutzerdaten";
$items[3] = "Buchungen";
12 $items[7] = "logout";

14 // Zuweisung der Verzeichnisse auf die Menue-Namen
// $access[Menue-Name] = Verzeichniss-Name
16 $access = array();
$access["error"] = "error";
18 $access["home"] = "home";
$access["Benutzerdaten"] = "userdata";
20 $access["Buchungen"] = "buchungen";
// keine Zuweisung fuer logout notwendig
22 // $access["logout"] = "logout";

24 // Zuweisung der Beschreibung auf die Menue-Namen
// $description[Menue-Name] = Beschreibung
26 $description = array();
$description["home"] = "Link auf diese Seite";
28 $description["Benutzerdaten"] = "Hier k&ouml;nnten Sie Ihre " .
    "pers&ouml;nlichen Daten &auml;ndern";
30 $description["Buchungen"] = "Sehen Sie in die Liste alle gebuchten " .
    "&Uuml;bungsaufgaben";
32 $description["logout"] = "Verlassen Sie ihr Profil";
34 ?>
```

E.5.1 Home

Der Quellcode des Moduls "Home" im Dozentenbereich ist identisch mit den Scripten des Moduls "Home" im Anwenderbereich (siehe Kapitel [E.3.1](#)).

E.5.2 Benutzerdaten

Der Quellcode des Moduls "Benutzerdaten" im Dozentenbereich ist identisch mit den Scripten des Moduls "Benutzerdaten" im Anwenderbereich (siehe Kapitel [E.3.2](#)).

E.5.3 Error

Der Quellcode des Moduls "Error" im Dozentenbereich ist identisch mit den Scripten des Moduls "Error" im Anwenderbereich (siehe Kapitel [E.3.5](#)).

Anhang F

Beigefügte CD

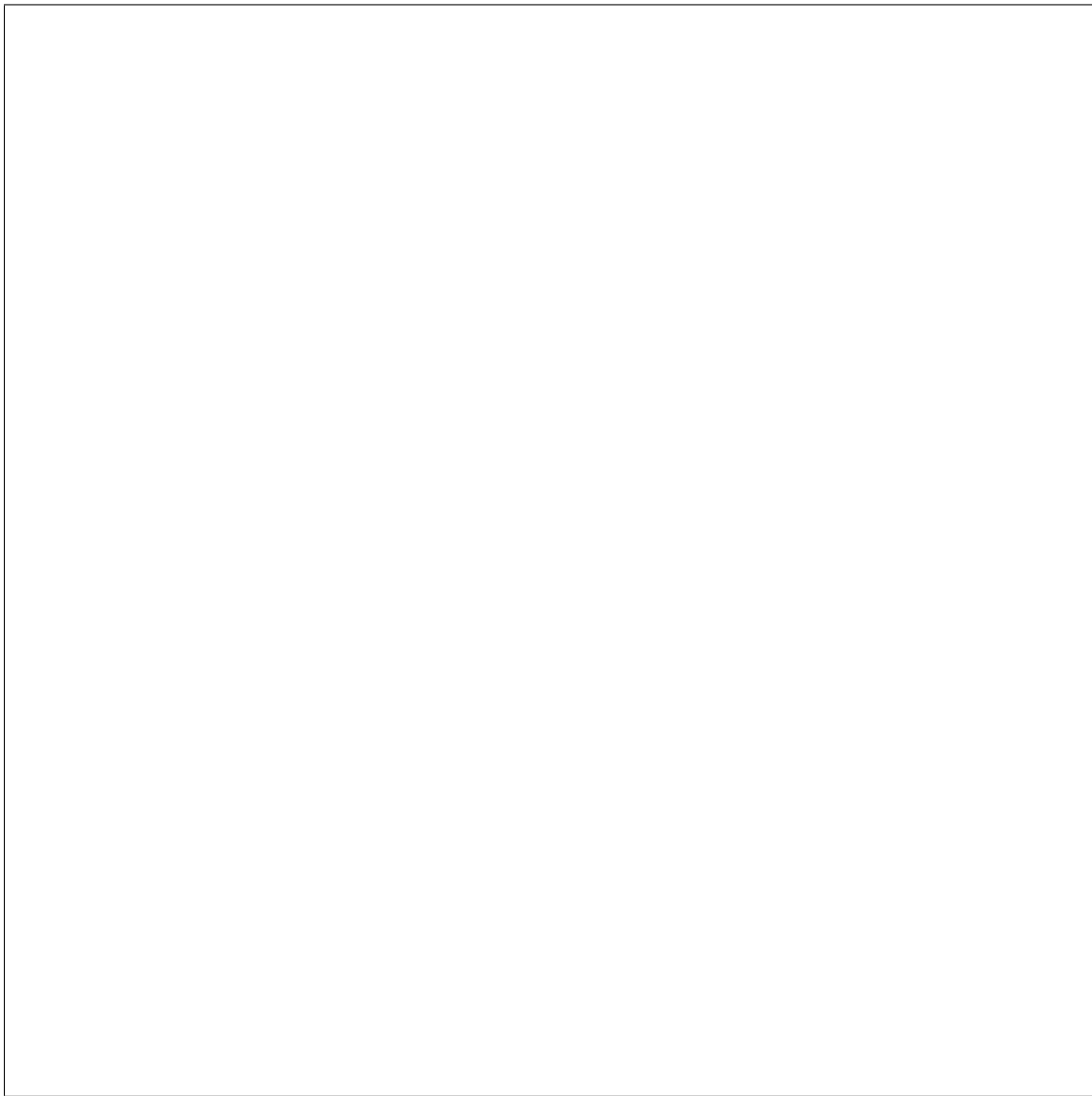


Abbildung F.1: beigefügte CD

Inhalt der beigefügten CD

Diplomarbeit

```
|
|--backup (Scripte zum Erzeugen und Sichern der Datenbank)
|
|--cron (Scripte, welche von 'cron' gesteuert werden)
|
+---doc
|   |
|   |--latex (Quelldateien zum Erzeugen der Diplomarbeit)
|   |
|   |--dipl.pdf
|   |
|   |--dipl.ps
|
+---public_html (vulab als PHP-Projekt)
    |
    |--admin
    |
    |--bin
    |
    |--default
    |
    |--dozent
    |
    |--images
    |
    |--log
    |
    |--user
```

Abkürzungsverzeichnis

API	<p>Application Programming Interface</p> <p>Dabei handelt es sich um den Satz an Funktionsaufrufen, die eine Programmiersprache dem Anwendungsprogrammierer zur Verfügung stellt.</p>
CGI	<p>Common Gateway Interface</p> <p>Ein CGI ist ein simples Protokoll, das man zum kommunizieren zwischen HTML-Formularen und einem Programm nutzen kann. Ein CGI-Skript kann in jeder Sprache geschrieben sein - solange sie das Lesen von STDIN, das schreiben auf STDOUT erlaubt und Environment-Variablen lesen kann. (C, Perl, AppleSkript...)</p>
DBMS	<p>DataBase Management System</p> <p>Ein DBMS verwaltet eine zusammenhängende Menge von Daten - die Datenbank. Mit einem DBMS wird die Sicherheit, Integrität und Konsistenz der Daten bei minimaler, kontrollierter Redundanz gewährleistet. Durch die Integration der Daten und die Kontrolle des Zugriffs auf die Daten durch ein DBMS können die Daten durch viele Benutzer gleichzeitig genutzt werden.</p>
DES	<p>Abkürzung für "Data Encryption Standard"</p> <p>(Datenverschlüsselungs-Standard).</p> <p>Von IBM entwickeltes Verschlüsselungssystem von 1974, das von 1977 bis 2000 von der US-Regierung als offizielles Datenchiffriersystem eingesetzt wurde. In einer symmetrischen Verschlüsselung werden Blöcke zu je 64 Bits mit einem 56-Bit-Schlüssel codiert [?].</p>
SAPI	<p>Server Application Programming Interface</p> <p>SAPI ermöglicht es Programmierern, Webbasierte Anwendungen zu entwickeln, die schneller laufen als konventionelle CGI's, da sie besser mit dem Webserver integriert sind.</p>

SQL Structured Query Language - eine standardisierte Abfragesprache für Datenbanken. Für SQL als solches existiert sowohl eine ANSI als auch eine ISO Norm, dennoch bieten die meisten Anbieter SQL-Datenbanken mit eigenen Erweiterungen an, welche über den normierten Funktionsumfang hinausgehen.

Three-Tier-Architektur

In einer 3-Tier-Architektur wird die Anwendung in drei Schichten aufgeteilt. Auf dem Frontend, dem Client-Rechner, liegt die Benutzerschnittstelle. Im Backend sind die Daten gespeichert. Dazwischen wird eine sogenannte "Middle-Tier" implementiert (häufig auch als Applikations-Server bezeichnet), die für Steuerung und Verarbeitung verantwortlich ist.

URL Abkürzung für URL "Uniform Resource Locator" – beispielsweise "http://www.glossar.de" oder "http://www.microsoft.de". Eine URL entspricht der Adresse eines INTERNET-Angebotes. Sie enthält

- die Bezeichnung des angesprochenen INTERNET-Dienstes bzw. des entsprechenden Übertragungsprotokolls - beispielsweise:
 - "http" steht für WWW-Seiten
 - "ftp" steht für File-Transfer
 - "news" steht für News bzw. Newsgroups
- die Serveradresse incl. der Domain (com, org oder edu) bzw. Landes-Kennung (de, ch, uk),
- optional den Port (z. B. "80")
- den Pfad auf dem Server
- und den Namen des Dokuments bzw. der Datei.

Tabellenverzeichnis

5.1	Zusammenfassung der Parameterarten	84
B.1	Benutzer	108
B.2	Rechner	108
B.3	Images	108
B.4	Buchungen	109
B.5	Übungen	109
B.6	Übungs_Checks	110
B.7	Ergebnis_Checks	110
B.8	Übungs_Setup	110

Abbildungsverzeichnis

1.1	Geplante Netzwerkstruktur	2
1.2	Three Tier Architektur	3
1.3	Datenbankstruktur	5
2.1	Ablaufplan der Neuanmeldung (Profil erstellen)	10
2.2	Ablaufplan des Logins	12
2.3	Screenshot des Login-Formulars	13
2.4	Ablaufplan zum Erstellen der Übungen	14
2.5	Ablaufplan zum Buchen der Übungen	15
2.6	Ablauf der Übung	18
3.1	HTTP-Dokument	22
3.2	Pinzipieller Seitenaufbau — <code>index.php</code>	24
3.3	Datei-Struktur des Projekts	26
3.4	Eingebettetes Menü des Moduls <code>home</code>	28
3.5	Blockdiagramm zum Ablauf der Funktion <code>getContentLink(\$content, \$userType)</code>	30
3.6	Verzeichnis-Struktur die das Beispiel verwendet	32
3.7	Formular zum Erstellen eines neuen Profils	39
3.8	Ablaufdiagramm zum Erstellen eines neuen Profils	40
3.9	Formular zum Einloggen eines Benutzers	41
3.10	Ablaufdiagramm zum Einlogg-Vorgang	42
3.11	Formular zum Verändern der Benutzerdaten	43
3.12	Bestätigung der veränderten Benutzerdaten	44
3.13	Ablaufdiagramm zum Verändern der Benutzerdaten	44
3.14	Verwaltung der Benutzer	46
3.15	Klick auf Schaltfläche 'Profil editieren'	46
3.16	Profil editieren	47
3.17	Klick auf Schaltfläche 'Profil löschen'	48
3.18	Das eigene Profil kann nicht gelöscht werden	48
3.19	Soll das Profil gelöscht werden ?	49

3.20 Ablaufdiagramm zum Verändern der Benutzerdaten als Administrator	50
3.21 Auswahl der verfügbaren Startzeiten des 26. Mai 2003	53
3.22 Auswahl der verfügbaren Übungen	54
3.23 Ausgabe aller gewählten Daten	55
3.24 Ablaufdiagramm zum Vorgang des Buchens	59
3.25 Verwaltung gebuchter Übungen	60
3.26 Buchung löschen	61
3.27 Ablaufdiagramm zum Verwalten der Buchungen im Anwenderbereich	63
3.28 Verwaltung gebuchter Übungen im Administrationsbereich	64
3.29 Klick auf Schaltfläche 'at-Job einsehen'	65
3.30 Informationen zum at-Job einsehen	66
3.31 Klick auf Schaltfläche 'Buchung löschen'	66
3.32 Soll die gebuchte Übung gelöscht werden ?	67
3.33 Ablaufdiagramm zum Verwalten der Buchungen im Administrationsbereich	68
4.1 Startseite des 'Virtuellen Unix Labors'	71
5.1 Hinzufügen eines neuen Menüpunkts im Anwenderbereich	77
5.2 Inhalt des neuen Moduls 'Beispiel'	78
A.1 Testseite des Apache Web Server	93
A.2 Informationsseite über PHP	97
B.1 Datenbankstruktur	104
F.1 beigefügte CD	297

Quellcodeverzeichnis

3.1	index.php	25
3.2	Konfigurationsdatei default/config.php	27
3.3	functions.php: getContentLink(\$content, \$userType)	29
3.4	index_session.php	35
3.5	logout.php	37
3.6	Auszug aus user/buchen/index.php	52
4.1	Auszug aus der Konfigurationsdatei config.php	70
4.2	SQL-Skript zum Säubern der Datenbank (cleanDB.sh)	73
5.1	user/config.php — Hinzufügen eines neuen Moduls	76
5.2	Code-Beispiel zum Session-Handling	79
5.3	Code-Beispiel zum Error-Handling	80
5.4	Beispiel eines Formulars mit versteckten Feldern	84
6.1	Auszug zur Berechtigungs-Prüfung	87
6.2	HTML-Code der fraglichen Ausgabe	88
6.3	Beispiel für schlechten PHP-Code	89
B.1	SQL-Skript zur Erzeugung der Datenbankstruktur	103
B.2	SQL-Skript zur Erzeugung der Datenbankstruktur (Fortsetzung)	104
C.1	SQL-Skript zum Löschen der Tabellen (dropdb.sql)	112
C.2	Makefile zur Sicherung der Daten	113
E.1	config.php	125
E.2	index.php	126
E.3	index_session.php	129
E.4	logout.php	134
E.5	noConfig.php	135
E.6	notFound.php	135
E.7	functions.php	136
E.8	vulab.css	146
E.9	default/config.php	151
E.10	default/home/index.php	151
E.11	default/login/index.php	153

E.12 default/login/validateLogin.php	156
E.13 default/firstLogin/index.php	160
E.14 default/firstLogin/validate.php	164
E.15 default/info/index.php	171
E.16 default/bestaetigen/index.php	173
E.17 default/error/index.php	175
E.18 user/config.php	177
E.19 user/home/index.php	178
E.20 user/home/index.php	180
E.21 user/home/userData2.php	184
E.22 user/home/validateUserData.php	188
E.23 user/home/updated.php	193
E.24 user/buchen/index.php	193
E.25 user/buchen/calender.php	199
E.26 user/buchen/buchen2.php	205
E.27 user/buchen/buchen3.php	209
E.28 user/buchen/validate.php	214
E.29 user/buchen/zeigeBuchung.php	217
E.30 user/buchen/speichereBuchung.php	220
E.31 user/buchen/execute.php	225
E.32 user/buchungen/index.php	229
E.33 user/buchungen/delete.php	237
E.34 user/buchungen/delete2.php	240
E.35 user/buchungen/deleted.php	242
E.36 user/error/index.php	243
E.37 admin/config.php	246
E.38 admin/userdata/index.php	247
E.39 admin/userdata/adminData2.php	254
E.40 admin/userdata/adminData3.php	259
E.41 admin/userdata/validateData.php	263
E.42 admin/userdata/updated.php	270
E.43 admin/userdata/delete.php	271
E.44 admin/userdata/delete2.php	274
E.45 admin/userdata/deleteMessage.php	276
E.46 admin/userdata/deleted.php	277
E.47 admin/buchungen/index.php	277
E.48 admin/buchungen/show.php	286
E.49 admin/buchungen/delete.php	288

E.50 admin/buchungen/delete2.php	291
E.51 admin/buchungen/deleted.php	293
E.52 dozent/config.php	295

Literaturverzeichnis

Thin oder thick client. *Computer Journal*, Mai 2001.

PHP Manual. <http://www.php.net/manual/en/>, November 2002.

NetBSD 1.6 Installieren - Konfigurieren - Administrieren. C&L Computer und Literatur, Zavelsteiner Straße 20, 71034 Böblingen, 2003.

Lars Eilebrecht. *Apache Web-Server*. MITP Verlag, Bonn, second edition, 1998.

Hubert Feyrer. *Virtuelles Unix Labor — Design*.

<http://smaug.fh-regensburg.de/~feyrer/design>, April 2003.

Helmut Kopka. *L^AT_EX Einführung*. Addison-Wesley Verlag, third edition, 2002.

Ben Laurie and Peter Laurie. *Apache - Das umfassende Referenzwerk*. O'Reilly, Balthasarstr. 81, 50670 Köln, zweite edition, 1999.

Rasmus Lerdorf and Kevin Tatroe. *Programming PHP*. O'Reilly, 1005 Gravenstein Highway North, Sebastopol, CA 95472, first edition, March 2002.

Hugh E. Williams and David Lane. *Web Database Application with PHP and MySQL*. O'Reilly, 1005 Gravenstein Highway North, Sebastopol, CA 95472, first edition, March 2002.

John C. Worsley and Joshua D. Drake. *Practical PostgreSQL*. O'Reilly, 1005 Gravenstein Highway North, Sebastopol, CA 95472, first edition, January 2002.

Stichwortverzeichnis

Übergabe von Parametern, [82](#)

Übung

vorbereiten, [57](#)

Übungen

Liste, [54](#)

wiederholbar, [54](#)

char, [86](#)

varchar, [86](#)

abgeschottetes Netz, [2](#)

Ablauf d. Übung, [16](#), [51](#)

admin, [12](#), [45](#)

Alter d. Buchung, [53](#)

Apache, [6](#), [91](#)

Installation, [91](#)

Konfiguration, [92](#)

starten, [92](#)

stoppen, [92](#)

Applikations-Logik, [4](#), [5](#)

Arbeitszeiten, [51](#)

Architektur des Projekts, [9](#)

at(1), [15](#), [16](#), [56–58](#), [61](#), [67](#)

einsehen, [65](#)

löschen, [62](#)

at-Job Nummer, [58](#), [65](#), [66](#)

Aufsetzen d. Images, [56](#)

Ausgabe umleiten, [58](#)

Ausgabe-Puffer, [21](#), [89](#)

abschalten, [23](#)

authentifiziert, [36](#)

Authentifizierung, [42](#)

Automatisierung, [9](#)

Benutzer-Berechtigung, [36](#)

Benutzerdaten

im Administrationsbereich, [45](#)

im Anwenderbereich, [42](#)

Benutzerkennung, [9](#)

Benutzerprofil anlegen, [38](#)

Benutzertyp, [28](#)

Benutzerverwaltung, [70](#)

Bestätigung d. Accounts, [72](#)

Bestätigungsnummer, [72](#)

buchen, [14](#)

buchen d. Übung, [51](#)

Buchung

abschließen, [55](#)

Alter der, [53](#)

einsehen, [60](#)

eintragen, [56](#)

entfernen, [66](#), [67](#)

löschen, [61](#), [66](#)

prüfen, [55](#)

rückgängig machen, [61](#)

Buchungen

kontrollieren, [64](#)

Liste der, [60](#)

verwalten als Administrator, [64](#)

verwalten als Anwender, [60](#)

zugreifen auf, [66](#)

Buchungsnummer, [56](#)

Buchungssystem, [51](#)

checkPass(\$plain, \$crypt), [115](#)

Client-Schicht, [4](#)

Connection-Objekt, [85](#)

createPass(\$plain), 116

cron, 11, 72

Crypt, 41

cutTime(\$time), 116

Datei-Stuktur, 26

Daten

 einsehen, 45

 gefälschte, 72

 in Session speichern, 38

 löschen, 49

 persönliche, 38, 42

 verändern, 42, 45

 verwerfen, 48

Datenbank

 überfluten, 72

 säubern, 72

Datenbank Management System, 7,
 97

Datenbankschicht, 5

DBMS, 97

deploy, 56, 58, 73

DES, 11

Design, 5

dozent, 12

Drei-Schichten-Modell, 4

Eingabemaske, 38, 41

Eingaben

 prüfen, 43, 46

Error-Handler, 81

Error-Handling, 80

executeStatement(...), 117

Fat Client, 3

Fehler

 analysieren d. Herkunft, 81

Fehler protokollieren, 70

Fehlerbehandlung, 80

Fehlerbenachrichtigung

 unterdrücken, 81

Fehlermeldungen, 38

feste Zeitfenster, 51

fetchArray(\$result, \$pos), 118

Firewall, 2, 16

functions.php, 115

 checkPass(\$plain, \$crypt), 115

 createPass(\$plain), 116

 cutTime(\$time), 116

 executeStatement(...), 117

 fetchArray(\$result, \$pos), 118

 getConnection(), 118

 getContentLink(...), 119

 getLocation(\$link), 120

 getSubdirs(\$dir), 120

 getValue(\$method, \$key), 121

 handleErrors(...), 121

 home(\$userType), 122

 makeSecret(), 122

 makeSeed(), 122

 menu(\$userType), 123

 showGetVar(), 123

 showPostVar(), 123

 showServerVar(), 124

 showSessionVar(), 124

gefälschte Daten, 72

GET, 20, 24

getConnection(), 118

getContentLink(...), 119

getLocation(\$link), 120

getSubdirs(\$dir), 120

getValue(\$method, \$key), 121

globale Variablen, 19, 88

Grundgerüst, 38

handleErrors(...), 121

Hash-Tabellen, 19

Header, 83, 89

- Herkunft d. Fehlers, [81](#)
- Hintergrundwissen, [75](#)
- home(\$userType), [122](#)
- HTTP-Body, [22](#)
- HTTP-Header, [22](#)

- Identifikation, [13](#)
- Image
 - aufsetzen, [56](#)
- Installation, [69](#)
- IP-Adresse, [16](#)

- Kalender, [51](#)
- Kompatibilität, [86](#)
- Konfiguration, [27](#)
- Konfigurationsdatei, [70](#), [75](#)
- Konzept des Projekts, [9](#)

- Leerzeichen, [88](#)
- Liste d. Übungen, [54](#)
- log-Datei, [81](#)
- Login, [42](#)
 - Vorgang, [12](#), [41](#)

- Makefile, [69](#)
- makeSecret(), [122](#)
- makeSeed(), [122](#)
- Manipulationen, [34](#)
- manipulieren der URL, [36](#)
- Menü, [24](#), [26](#)
- Menüpunkt
 - Beschreibung, [76](#)
 - Name d., [75](#)
 - neuer, [75](#)
- menu(\$userType), [123](#)
- Mittelschicht, [5](#)
- Modul, [27](#), [75](#), [82](#)
 - aufruf, [78](#)
 - Beschreibung, [76](#)
 - neues, [75](#)

- NetBSD, [91](#)
- Netzwerkstruktur, [2](#)
- Neuinstallation, [2](#)

- output-buffering, [21](#), [89](#)

- Parameter Übergabe, [82](#)
- Passwort, [39](#), [41](#)
- persönliche Daten, [38](#), [42](#)
- persönliches Profil, [11](#)
- PHP, [7](#), [85](#), [94](#)
 - Installation, [94](#)
 - Module, [94](#)
 - Scripte, [70](#)
- pkg_add, [91](#)
- Portierbarkeit, [19](#), [22](#)
- POST, [24](#)
- PostgreSQL, [7](#), [69](#), [85](#)
 - pg_hba.conf, [85](#)
 - Anlegen einer Datenbank, [100](#)
 - Anlegen eines Benutzers, [100](#)
 - Client Authentifizierung, [99](#)
 - fehlerhafte Konfiguration, [85](#)
 - Installation, [97](#)
 - pg_hba.conf, [99](#)
 - starten, [98](#)
 - stoppen, [98](#)
- Probleme mit char, [86](#)
- Profil
 - bestätigen, [11](#)
 - editieren, [45](#)
 - löschen, [48](#)
- Programmierung, [19](#)
 - Probleme, [19](#)
- Projekt-Architektur, [9](#)

- Rechte, [70](#)
- Reservierung d. Übung, [14](#)
- root, [14](#)
- root-Zugriff, [2](#)

Säubern d. Datenbank, 72

Schwierigkeiten, 85

Script

Aufruf, 82

via Formular, 84

via Link, 82

Seitenaufbau, 23

Seiteninhalt, 29

Session

übernehmen, 37

initialisieren, 78, 79

neue Session-ID, 34

Session-ID, 34

Sitzung übernehmen, 34

starten, 34

transiente Session-ID, 90

Variablen, 36

Session-Handling, 34, 78

enable-trans-sid, 90

Start-Datei, 34

showGetVar(), 123

showPostVar(), 123

showServerVar(), 124

showSessionVar(), 124

Sicherheit, 11, 37, 61, 67

Sicherheits-Überprüfung, 36

Sicherheits-Probleme, 36

Sicherheitsmechanismus, 48

Sicherheitsrisiken, 19

Sicherheitsrisiko, 88

Sitzung, *siehe* Session

Sonderfälle, 88

SQL, 6

SQL-Statement

via `pg_exec()`, 86

via `pg_query()`, 86

Startzeit, 57

übermitteln, 53

auswerten, 53

des Tages, 52

zerlegen, 57

Stichwörter, 60, 64

suchen, 60, 64

Systemaufruf, 58, 62, 67

Tabellen

anlegen, 69

Thick Client, 3

Thin Client, 4

Three-Tier Architektur, 3

Umgebungsvariable, 24

Umleiten d. Ausgabe, 58

unberechtigter Zugriff, 34, 36

user, 12

Validierer, 38, 41

variable Rechnerzeiten, 51

verschlüsseln, 11, 13, 39, 41

Verzeichnis `public_html`, 70

Vorlaufzeit, 57

`wall(1)`, 17

Warteschlange, 58, 66, 67

Web-Server, 6

Weiterentwicklung, 75

Wurzel-Verzeichnis, 26

Zeitfenster

feste, 51

Zeitraum, 51

Zeitstempel, 53

Zugang verschaffen, 72

Zugangs-Berechtigung, 38

Zugriff, 20, 37

unberechtigter, 34, 36