

SIS - School Information Service

Florian BUCHBERGER

Marco HANDLE

Matthias KLOTZ

Mathias WEILAND

HTL Anichstraße

15. April 2014

made with L^AT_EX

Inhaltsverzeichnis

| | |
|---|-----------|
| Gesamtdokumentation | vi |
| 0.1 Selbstständigkeitserklärung | vi |
| 1 Zusammenfassung | 1 |
| 1.1 Kurzbeschreibung | 1 |
| 1.2 Projektergebnis | 1 |
| 2 Pflichtenheft | 2 |
| 2.1 Funktionale Anforderungen | 2 |
| 2.1.1 Definitionen | 2 |
| 2.1.2 Supplierungssystem | 2 |
| 2.1.3 News | 2 |
| 2.1.4 Monitorsystem | 2 |
| 2.1.5 Authentifizierung | 3 |
| 2.1.6 Eingabe | 3 |
| 2.2 Schnittstellen | 6 |
| 2.3 Abnahmekriterien | 6 |
| 2.4 Dokumentationsanforderungen | 6 |
| 2.5 Qualitätsstandards | 6 |
| 2.6 Prozessmodell | 6 |
| 3 Systemdokumentation | 7 |
| 3.1 Technologien | 7 |
| 3.1.1 HTTP (Weiland) | 7 |
| 3.1.1.1 Verbindungsvorgang | 7 |
| 3.1.1.2 Statuscodes | 7 |
| 3.1.1.3 Versionen | 8 |
| 3.1.2 HTTPS (Weiland) | 8 |
| 3.1.3 Serverseitige Technologien | 8 |
| 3.1.3.1 PHP (Handle) | 8 |
| 3.1.3.2 MySQL (Handle) | 13 |
| 3.1.3.3 Datenbank-Design (Buchberger) | 22 |
| 3.1.3.4 LDAP (Buchberger) | 23 |
| 3.1.4 Clientseitige Technologien | 27 |

| | | |
|---------|--------------------------------|----|
| 3.1.4.1 | HTML (Weiland) | 28 |
| 3.1.4.2 | Javascript (Klotz) | 29 |
| 3.1.4.3 | AJAX (Buchberger) | 32 |
| 3.1.5 | Mobil App (Klotz) | 33 |
| 3.1.5.1 | PhoneGap | 33 |
| 3.1.5.2 | iOS | 34 |
| 3.1.5.3 | Android | 34 |
| 3.1.5.4 | Windows Phone | 35 |
| 3.1.5.5 | Sonstiges | 35 |
| 3.2 | Lösungswege | 35 |
| 3.2.1 | PHP | 35 |
| 3.2.1.1 | Gewählte Lösung | 35 |
| 3.2.1.2 | Alternative Lösung | 35 |
| 3.2.2 | MySQL | 35 |
| 3.2.2.1 | Gewählte Lösung | 35 |
| 3.2.2.2 | Alternative Lösung | 35 |
| 3.2.3 | Datenbankdesign | 35 |
| 3.2.3.1 | Gewählte Lösung | 35 |
| 3.2.3.2 | Alternative Lösungen | 37 |
| 3.2.4 | HTTPS | 37 |
| 3.2.4.1 | Gewählte Lösung | 37 |
| 3.2.4.2 | Alternative Lösungen | 37 |
| 3.2.5 | Mobile App | 37 |
| 3.2.5.1 | Gewählte Lösung | 37 |
| 3.2.5.2 | Alternative Lösung | 37 |
| 3.2.6 | Prinzip des Monitor-Systems | 37 |
| 3.2.6.1 | Laden der Daten | 38 |
| 3.2.7 | Authentifizierung der Monitore | 38 |
| 3.2.7.1 | Gewählte Lösung | 38 |
| 3.2.7.2 | Alternative Lösungen | 39 |
| 3.2.8 | Benutzermanagement | 39 |
| 3.2.8.1 | eDirectory-Gruppen | 40 |
| 3.2.8.2 | Verwendung | 40 |
| 3.3 | Grobentwurf | 40 |
| 3.3.1 | Dateibaum | 40 |
| 3.3.2 | Datenbankdesign | 42 |
| 3.3.2.1 | Monitorsystem | 42 |
| 3.3.3 | Monitorsystem | 43 |
| 3.3.3.1 | Grundsätzlicher Seiten-Aufbau | 43 |
| 3.3.3.2 | Initialisierung | 43 |
| 3.3.3.3 | Verringerung des Traffics | 43 |
| 3.3.3.4 | Multi-Modi | 44 |
| 3.4 | Feinentwurf | 44 |
| 3.5 | Implementierung | 44 |

| | | |
|----------|--------------------------------------|-----------|
| 3.5.1 | Sourcecode | 44 |
| 3.5.2 | Test- Und Messergebnisse | 44 |
| A | Anleitungen: Benutzer | 45 |
| A.1 | Einleitung | 45 |
| A.2 | Stundenplan | 45 |
| A.3 | Supplierplan | 45 |
| A.4 | News | 45 |
| A.5 | App | 45 |
| B | Anleitungen: Administrator | 46 |
| B.1 | Einleitung | 46 |
| B.2 | News | 46 |
| B.2.1 | News bearbeiten/hinzufügen | 46 |
| B.2.2 | News löschen | 47 |
| B.3 | Monitore | 47 |
| B.3.1 | Hinzufügen | 48 |
| B.3.2 | Entfernen | 49 |
| B.3.3 | Text verändern | 49 |
| B.3.4 | Raum verändern | 49 |
| B.3.5 | Abteilung verändern | 49 |
| B.3.6 | Modus verändern | 49 |
| B.3.7 | Datei verändern | 50 |
| B.3.8 | Display-Modus verändern | 50 |
| B.3.8.1 | Einschalt-Zeit | 51 |
| B.3.8.2 | Ausschalt-Zeit | 51 |
| B.4 | Fehlende | 51 |
| B.4.1 | Fehlende Lehrer | 52 |
| B.4.2 | Fehlende Schüler | 53 |
| B.5 | Supplierungen | 53 |
| B.6 | Administrative Eingaben | 53 |
| C | Begleitendes FTKL-Projekt | 54 |
| C.1 | Aufgabenstellung | 54 |
| C.2 | Hardware | 54 |
| C.2.1 | Überlegungen | 54 |
| C.2.2 | Probleme | 57 |
| C.3 | Software | 58 |
| C.3.1 | Überlegungen | 58 |
| C.3.2 | Lösung | 59 |
| C.3.2.1 | Überlegungen | 59 |
| C.3.2.2 | Darstellung | 59 |
| C.3.2.3 | GPIO | 60 |
| C.3.2.4 | Verteilung | 61 |
| C.3.3 | Probleme | 61 |

| | | |
|----------|-------------------------------|-----------|
| C.4 | Fazit | 61 |
| C.5 | Anhänge | 61 |
| C.5.1 | Codes | 61 |
| D | Buchberger | 64 |
| D.1 | Zusammenfassung | 64 |
| D.2 | Projekterfahrung | 64 |
| D.3 | Verfahrensanleitung | 64 |
| E | Handle | 65 |
| E.1 | Zusammenfassung | 65 |
| E.2 | Projekterfahrung | 65 |
| E.3 | Verfahrensanleitung | 65 |
| F | Klotz | 66 |
| F.1 | Zusammenfassung | 66 |
| F.2 | Projekterfahrung | 66 |
| F.3 | Verfahrensanleitung | 66 |
| G | Weiland | 67 |
| G.1 | Zusammenfassung | 67 |
| G.2 | Projekterfahrung | 67 |
| G.3 | Verfahrensanleitung | 67 |
| H | Anhang: Projektplanung | 68 |
| H.1 | Terminplanung | 68 |
| H.2 | Personalplanung | 68 |
| H.2.1 | Kostenplanung | 68 |
| H.2.2 | Arbeitsnachweis | 68 |

Gesamtdokumentation und begleitende Dokumente

0.1 Selbstständigkeitserklärung

1 || Zusammenfassung des Projektergebnisses

1.1 Kurzbeschreibung

Im Zuge dieser Diplomarbeit wurde SIS (School Information Service) entwickelt. Dieses System stellt über ein Webinterface den Schülern und Lehrer den Stundenplan und Supplierplan, sowie aktuelle News zur Verfügung. Zusätzlich wird ein Stundenplan generiert, in den bereits die supplieden Stunden hervorgehoben werden. Für diese Funktionalität wird auch eine App für die Mobilbetriebssysteme iOS, Android und Windows Phone zur Verfügung gestellt. Darüber hinaus werden auf den Monitoren, die vor den Werkstätten und manchen Klassenräumen positioniert sind, je nach Einstellung, die aktuellen schulrelaventen Neuigkeiten, die Supplierpläne der Abteilung, der Stundenplan des nächst-gelegenen Raumes, oder benutzerdefinierte Bilder angezeigt. Die News können von den Administratoren der Abteilungen eingetragen, sowie von den News-Beauftragten der Klassen vorgeschlagen werden.

Das System wurde für alle 4 Abteilungen ausgelegt, aber die Nutzung nur in der Elektronik-Abteilung forsiert.

1.2 Projektergebnis

Alle Schüler verwenden unser System und sich rundum glücklich und zufrieden... bis auf den Schüler, der sich aus Verzweiflung, weil er mit dem Interface der App nicht klarkam, aus einem Fenster des Elektronik-Labors im vierten Stock des Gebäudeteils Anichstraße geworfen hat...

... TODO

2 || Pflichtenheft

2.1 Funktionale Anforderungen

2.1.1 Definitionen

angepasster Stundenplan

Stundenplan mit eingearbeiteten Supplierungen

tabellarischer Supplierplan

Auflistung aller Supplierungen, Ausfälle, etc

Relevanz bei Ersatzlehrern

Ist der Lehrer an diesem Tag nicht in der Schule, so ist er als Supplierlehrer nicht erste Wahl (kursiv oder grau hinterlegt darstellen). Ist er an diesem Tag in der Schule, hat jedoch Unterricht, so ist er nicht als Supplierlehrer einsetzbar, ist er jedoch als Zweitlehrer im selben Unterricht mit dem Absenzlehrer, dann kann er als "Klasse alleineeingeteilt werden, etc.

2.1.2 Supplierungssystem

Es soll ein System entwickelt werden, dass die Stundenpläne und Supplierungen in digitaler Form speichert. Dazu soll eine Ein und Ausgabe der Daten über eine Website und eine App zur Verfügung gestellt werden (siehe Punkt Ausgabe). Weiters wird ein Formular generiert (PDF), das ausgedruckt werden kann.

Eingabe über die App nur eingeschränkt und wenn zeitlich möglich. Die Supplierungen und Stundenpläne werden vom Administrator (AV, WL, ...) eingegeben (siehe Punkt Eingabe).

2.1.3 News

News sollen vom Administrator (AV, WL, ...) eingegeben werden (abrufbar über die Website bzw. App).

2.1.4 Monitorsystem

ThinClients (z.B.: Raspberry Pi) mit Monitoren sollen mit Daten versorgt werden. Dazu soll nur ein HTML5 kompatibler Browser benötigt werden. Der dementsprechende

HTMLCode soll möglichst Auflösungskompatibel sein. Es soll möglich sein, das auf den ThinClients Dargestellte individuell über die Website zu konfigurieren.

Folgende Möglichkeiten:

- angepasster Stundenplan des nächstgelegenen Raumes
- tabellarischer Supplierplan der Abteilung (mit Informationen bzgl: Magazin und den News)
- Bild als JPG, PNG oder GIF (Upload über die Website)
- Video im MP4-Container (Upload über die Website)
- Uhr

2.1.5 Authentifizierung

Authentifizierung erfolgt für die Schüler und Lehrer via LDAP, gilt auch für Monitore (diese müssen sich als Monitore identifizieren. Ohne erfolgreichen Login sind keine Informationen abrufbar.

2.1.6 Eingabe

Administratoren und AVs dürfen Eingaben tätigen. Damit einfache Eingaben auch delegiert werden können muss ein Berechtigungssystem hinterlegt werden.

Lehrer

Name, Initialen, Abteilung
Buttons zum Hinzufügen, Editieren und Löschen (LDAP)

Klassen

Name, KV (als DropdownMenü), Abteilung (als DropdownMenü), Raumbelegung

Räume

Bezeichnung, Abteilung

Fächer

Bezeichnung (Kürzel und Langname)

Stunden(-pläne)

Fach, Lehrer (Dropdown-Menü; weitere Felder erscheinen bei der Auswahl), Dauer, Raum (Dropdown-Menü)

Auswahl der Klasse über ein Menü. Stundenplan aus "KlassenSicht". Liste der Wochentage und Buttons zum Hinzufügen, Platzieren, Editieren und Löschen von Stunden im Stundenplan.

Supplierungen

Drei Eingaben:

fehlende Lehrer

Lehrer (Dropdown-Menü), von-bis, Grund

fehlende Klassen

Klasse (Dropdown-Menü), von-bis, Grund

Supplierungen

Stunde (Dropdown-Menü), Klasse (Dropdown-Menü), ausblenden (CheckBox; wenn gesetzt, wird diese Stunde in den angepassten Stundenplänen nicht angezeigt), Supplierlehrer (Dropdown-Menü; zeigt die Lehrer sortiert und markiert nach Relevanz), Kommentar (Hier wird eingetragen zb: Mitbetreuung", SStillbeschäftigung", entfällt", etc), bestätigen (CheckBox; Eintrag ist erst wirksam, wenn gesetzt)

Ein Supplierlehrer muss bei Mitbetreuung nicht angegeben werden, da alle anderen Lehrkräfte dieser Stunde, sowieso mit dieser verknüpft sind.

Verschobene Stunden werden als 2 Einträge (einmal äusgefallenmit dem äusblendenButton) und einmal neu eingefügt"(gekennzeichnet über Kommentar) eingegeben.

(ev. falls noch Zeit: Wenn ein fehlender Lehrer eingetragen wurde, so werden automatisch alle "Kollisionenängezeigt.)

News

Name, Beschreibung, vonbis, Abteilung (Dropdown-Menü; auch mit Auswahl für die ganze Schule), die News werden nach Ablauf (BisDatum) nicht mehr angezeigt, aber nicht gelöscht.

Monitore

Modus (Auswahlliste, siehe Punkt Monitorsystem), falls benötigt: Datei (Upload für Bild, Video)

Die Monitore melden sich selbst in der DB an, so ist kein Hinzufügen von Monitoren nötig.

Allerdings: Möglichkeit zum Sperren von Einträgen, sollte sich ein Monitor verändern.

Über CheckBoxen wählt man alle oder einzelne Monitore aus, bei denen man die Konfiguration ändern will. Buttons für alle, keinen und einzelne auswählen.

Ausgabe

Hier gibt es 2 verschiedene Möglichkeiten:

Benutzer-Website/App

Nach Login:

Für Schüler und Lehrer wird ein Klassen/Lehrerspezifischer angepasster Stundenplan generiert. Über einen Button auf der Startseite kann die AnzeigeArt verändert werden.

Monitore

siehe Punkt Monitorsystem

App

Es soll eine App für Android, Windows Phone und iOS erstellt werden, die die gleichen Funktionen bietet, wie die StandardBenutzerWebsite (keine Administrativen Funktionen).

Zusätzlich soll die BenutzerWebsite (aufgrund der kompatibilität zu anderen Mobil-Betriebssystemen) auch als mobile Website implementiert werden.

Formular

Das Formular für die Übertragung der Supplierungen in das Abrechnungssystem, wird nach derzeitiger Vorlage generiert. Ein weiteres Formular wäre sinnvoll: Die Auflistung nach fehlendem Lehrer, damit man einen Überblick erhält:

Bsp:

YH fehlend:

| | | |
|-------------------|-------|----|
| 1.6. 1. Std. TKHF | 1aHEL | Nz |
| 2. Std. TKHF | 2aHEL | MT |
| 2.6. 3. Std. LA1 | 4aHEL | XY |

...

Layout

Die Eingabeseite/Eingabenmasken, sollen übersichtlich und einfach zu bedienen sein. Das Layout wird der neuen HTL Homepage angepasst (Corporate Design) - als Grundlage dient das FTKL Projekt (Machac, Handle, Wucherer).

Stundenplandesign

Als Vorgabe dienen die derzeitigen Raumbeschriftungen der Werkstätten – das Layout wird wieder an das neue Corporate Design angepasst.

AppDesign

siehe Corporate Design

Stundenplaneingabe

Am Schuljahresanfang wird der Stundeplan der Abteilung händisch ins SIS übertragen. Die Grundlage für die Eingabe ist der Klassenstundenplan. Es gibt Lehrer, die in anderen Abteilungen eingeteilt sind, es muss für den jeweiligen

Administrator möglich sein, auch diese Stunden einzugeben. Die Eingabemaske soll dem Wochenstundenplan angepasst sein (Stunde (116) Fach, Klasse, Raum).

Dokumentation

Die Dokumentation wird lt. Vorlage (Mail von Prof. Stecher) ausgeführt. Es sind Bedienungs und Serviceanleitungen zu erstellen. Mit diesen Unterlagen muss eine Weiterentwicklung (für andere Diplomanten) und eine Servicesierung durch das Lehrpersonal gewährleistet sein. Der Sourcecode ist sauber zu dokumentieren. Eine Hilfe im Programm im HTML Format ist zu erstellen.

Ein Projekttagebuch ist zu führen (Beginn des Tasks/Sprints; Zeit und Task; Unterbrechungen; Status)

Code im Code dokumentieren: doxygen/javadoc

Uhranzeige

Auf jedem Monitor ist eine Zeitanzeige zu sehen und diese wird dem Design der Anzeigeseite angepasst (Corporate Design).

2.2 Schnittstellen

Es wurde zwar eine Software-Schnittstelle zur verwendeten Schul-Management-Software Untis angedacht, diese Idee wurde aber verworfen, da die Sinnhaftigkeit aufgrund des kommenden Umstiegs der Schule auf eine neue Version in Frage gestellt wird.

2.3 Abnahmekriterien

// TODO

2.4 Dokumentationsanforderungen

// TODO, hint: Javadoc

2.5 Qualitätsstandards

// TODO

2.6 Prozessmodell

// TODO

3 || Systemdokumentation

3.1 Verwendete Technologien und Entwicklungswerkzeuge

3.1.1 HTTP (Weiland)

Das “Hypertext Transfer Protocol“ ist ein Protokoll des Application Layers des OSI-Layer-Modells.

HTTP ist ein zustandsloses Protokoll. Anfragen werden stets getrennt behandelt, auch wenn sie vom selben Client stammen. Dies kann durch eine Session geändert werden.

3.1.1.1 Verbindungsvorgang

Zu Beginn wird die Zieladresse in eine IP-Adresse umgewandelt. Anschließend wird eine TCP-Verbindung mit dem Server aufgebaut. Dann wird eine Anfrage an den Port 80 des Server gesendet.

[GET /index.html HTTP/1.1](#)

In diesem Fall würde die Datei “index.html“ mittels HTTP1.1 angefordert werden.

Im Header der Anfrage können außerdem Informationen, wie zum Beispiel der verwendete Webbrowser, enthalten sein.

[User-Agent: Mozilla/5.0 \(Windows NT 6.1; WOW64; rv:27.0\) Gecko/20100101 Firefox/27.0,](#)

Wenn die Anfrage erfolgreich bearbeitet wurde, wird an den Client eine Bestätigung gesendet.

[HTTP/1.1 200 OK](#)

3.1.1.2 Statuscodes

Es gibt für HTTP sechs verschiedene Arten von Statuscodes

3.1.1.2.1 1xx Diese Statuscodes werden während der Bearbeitung der Anfrage verwendet und dienen der Information.

3.1.1.2.2 2xx Diese Statuscodes werden benutzt , wenn die Anfrage erfolgreich bewältigt wurde.

3.1.1.2.3 3xx Diese Codes dienen dazu eine Umleitung ersichtlich zu machen. Bei solch einem Code wird eine Aktion des Clients gefordert, was meist automatisch geschieht.

3.1.1.2.4 4xx Hiermit werden Fehler gekennzeichnet. Zum Beispiel 404: Not Found".

3.1.1.2.5 5xx Diese Codes sollen Server-Fehler kennzeichnen.

3.1.1.3 Versionen

Aktuell sind zwei verschiedene Versionen von HTTP im Einsatz, HTTP/1.0 und HTTP/1.1.

Diese unterscheiden sich insofern, dass bei HTTP/1.0 für jede Anfrage eine neue Verbindung zum Server aufgebaut wird. Dies wirkt sich nachteilig auf die Geschwindigkeit aus, da z.B. auf einer Website mit vielen Bildern, für jedes Bild eine neue Verbindung hergestellt werden muss und diese Verbindungen durch die Eigenschaften von TCP-Verbindungen (z.B. Slow-start) entsprechend langsam sind.

3.1.2 HTTPS (Weiland)

3.1.3 Serverseitige Technologien

3.1.3.1 PHP (Handle)

Die Abkürzung PHP steht für "PHP Hypertext Preprocessor". Es handelt sich hierbei um eine serverseitige Programmiersprache, die vor allem in der Webentwicklung zum Einsatz kommt. Die Syntax ist an Perl und C angelehnt.

Als PHP-Module wurden nur php5-mysql sowie php5-ldap verwendet.

PHP ist seit Version 5 vollständig Objekt-orientiert, wurde aber imperativ/funktional verwendet.

Ein PHP Programm kann im Gegensatz zu anderen serverseitigen Programmiersprachen direkt in den HTML-Quelltext der Website eingebunden werden. Gekennzeichnet werden diese eingebetteten Programme mit den PHP-Tags (siehe Programm-Code 3.1).

```
1 <?php
2      /* Programm-Code */
3 ?>
```

Code 3.1: PHP-Tags

Befindet sich der PHP Code eingebettet in HTML-Quelltext, so ignoriert der Interpreter alles, das außerhalb der PHP-Tags steht.

Eines der großen Vorteile an PHP ist, dass es vollständig serverseitig verarbeitet wird, das heißt am Client wird keine Rechenleistung für das Ausführen des PHP-Codes benötigt.

3.1.3.1.1 Funktionsweise Der Client fragt am Webserver eine Datei mit der Endung .php an. Anschließend lädt der Webserver die Datei und übergibt diese dem PHP Interpreter. Dieser generiert in den meisten Fällen eine HTML Datei, welche anschließend dem Webserver übergeben wird. Dieser sendet die fertige Webseite an den Client. (siehe Abb. 3.1) Der PHP Interpreter ist nicht nur auf HTML Dateien begrenzt, es können auch andere Dateitypen, wie Bilder oder PDF Dateien generiert werden. Diese Funktionsweise hat das Problem, dass die Seite bei jedem neuen Aufruf erneut generiert werden muss, dies führt zu einer höheren Auslastung am Webserver. Um dieses Problem zu vermindern gibt es die Caches am Webserver, die häufig verwendete Programmteile zwischenspeichern, um sie bei erneutem Aufruf nicht erneut Interpretieren zu müssen.

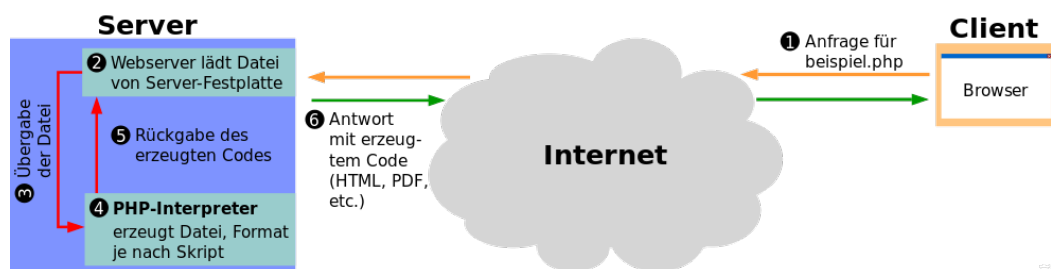


Abbildung 3.1: PHP Funktionsweise
 Quelle: <http://de.wikipedia.org/wiki/PHP>

3.1.3.1.2 PHP Sessions Mit PHP Sessions kann man einen Besucher einer Webseite über mehrere Aufrufe der Webseite hinweg genau identifizieren. Jedem Besucher auf der Webseite wird eine eindeutige Session-ID zugewiesen, diese wird in einem Cookie auf dem Server abgelegt. Eine PHP Seite, die die Session verwenden soll, muss als erstes des PHP-Dokuments diese Zeile stehen. (siehe Programm-Code 3.2)

```

1 <?php
2     session_start();
3 ?>
  
```

Code 3.2: Session

Damit wird dem Webserver vermittelt, dass diese Seite mit einer Session arbeitet. Die session-ID ist eine 128 Bit lange Zahl die zufällig generiert wird. Diese muss ab nun bei jeder Antwort vom Server an den Client mitgeliefert werden und auch umgedreht. Damit können personenbezogene Informationen einer bestimmten Session zugeordnet werden, wie ein Warenkorb bei Online-Shopping, Status von einer Anmeldung usw. Wir verwenden Sessions um den User über die Webseite hinweg identifizieren zu können.

3.1.3.1.3 Kommunikation mit Server PHP hat auch eine Möglichkeit, um diverse Daten dem Server zu senden, diese können von der zu empfangenden Webseite verarbeitet werden.

Diese Funktion wird vor allem bei Formularen eingesetzt, um die User-Interaktion mit

der Seite zu steuern. Es wird auch dazu verwendet um auf einer Seite zu navigieren oder ein Login bereitzustellen. Jedoch muss man in diesem Bereich im wesentlichen zwischen 2 Teile unterscheiden:

- POST
- GET

POST Bei POST werden die mitgegebenen Daten nicht an die URL angefügt, wie bei GET, aber dazu später, sondern werden im Body Teil der HTTP/S Anfrage an den Server. (siehe Programm-Code 3.3) Dies hat den großen Vorteil, dass die mitgegebenen Daten praktisch eine unbegrenzte Größe haben können. Dadurch können auch lange Texte, Bilder, Dateien usw. an den Server gesendet werden. Ein weiterer Vorteil liegt darin, dass der User die eingegebenen Daten nicht sieht, d.h. gibt er ein Passwort ein so ist es für ihn nicht einsehbar. Dies schützt jedoch nicht vor Angreifer, die die HTTP Anfragen mitschneiden, deshalb verwendet man beim Senden von Passwörtern immer HTTPS. Im Beispiel Programm-Code 3.4 ist ein kleine Eingabe von einem Text möglich. Bei drücken des Buttons Speichern wird an den Server eine Anfrage gestellt, dass die gleiche Seite nochmals aufgerufen werden sollte, jedoch werden POST Variablen mitgegeben. Diese wertet die Seite mit dem PHP Teil aus und gibt den eingegebenen Text aus. Da dieser Text beliebig lang sein kann, muss POST verwendet werden. Warm siehe Punkt GET.

```

1 POST /login/index.php      //Angefragte Seite
2 HOST sis.htlinn.ac.at     //Host
3      //Weitere Informationen
4      //Weitere Informationen
5      //Weitere Informationen
6      //leere Zeile - Trennt Header von Body ab
7 user=20091234&password=password&send=1  //POST-Parameter

```

Code 3.3: Ausschnitt HTTP POST Request

```

1 <html>
2 <head>
3 </head>
4 <body>
5 <?php
6
7 if(!empty($_POST['text']))      //Wenn die Variable text nicht leer
8     echo $_POST['text'];        //Soll der Text ausgegeben werden
9 else                            //Wenn sie leer ist
10    echo "nichts_eingegeben";   //Soll eine Warnung
11                                //ausgegeben werden
12 ?>
13 <form method="post">          //method="post" --> Parameter werden mit POST
14                                //mitgegeben

```



```

14         <textarea name="text"></textarea>
15         <button type="submit" name="save" value="Speichern">
16     </form>
17 </body>
18 </html>

```

Code 3.4: Beispiel POST

GET Bei GET werden die mitgegebenen Parameter direkt an die URL angehängt, um dies trennen zu können, wird die Parameterliste mit einem ? eingeleitet. Einzelne Parameter werden mit einem & getrennt. Diese Methode ist die Standardmethode wenn man ein HTML-Form erstellt. Dies wird bei kleinen Daten verwendet. Der Vorteil liegt darin, dass der User die Seite neu laden kann und die Parameter werden übernommen. Auch kann der User die URL samt den Parametern als Lesezeichen abspeichern, um beispielsweise bei einer Webseite, die die Parameter zur Navigation auf der Seite verwendet, genau die gewünschte Seite zu erhalten. Eine Beispiel-URL sieht wie folgt aus: *sis.htlinn.ac.at/index.php?text=hallo&send=Speichern*. Dies wäre die URL, die bei absenden des Beispiel Programm-Codes 3.5 entstehen würde. Sofern die Datei des Programm-Codes index.php heißt. Im Prinzip läuft die Auswertung der mitgelieferten Parametern gleich ab, wie bei POST Parametern.

Bei GET besteht die Grenze der Datenlänge darin, dass eine URL nur ca. 2000 lang sein darf, dies ist Browser und Server abhängig, aber 2000 Zeichen funktioniert bei allen Server-Client Kombinationen.

```

1 <html>
2 <head>
3 </head>
4 <body>
5 <?php
6
7 if(!empty($_GET['text']))           //Wenn die Variable text nicht leer
    ist
8     echo $_GET['text'];             //Soll der Text ausgegeben werden
9 else                                //Wenn sie leer ist
10    echo "nichts_eingegeben";       //Soll eine Warnung
    ausgegeben werden
11 ?>
12
13 <form method="get">                //method="get" --> Parameter werden mit GET
    mitgegeben
14     <input type="text" name="text"></textarea>
15     <button type="submit" name="save" value="Speichern">
16 </form>
17 </body>
18 </html>

```

Code 3.5: Beispiel GET

URL-Encoding Die POST und GET Daten die an den Webserver gesendet werden, werden nicht exakt mit den Zeichen übertragen, die man zum Beispiel in ein Textfeld eingibt. Dies hat den Hintergrund, dass einige Zeichen reserviert sind und deshalb nicht in den GET oder POST Daten vorkommen dürfen. Um dies zu verhindern werden diese Zeichen mit anderen ausgetauscht. Dazu wird die Darstellung mit einem % verwendet. Dabei wird das Zeichen mit der im entsprechenden Hexadezimalen ASCII Code und einem % ausgetauscht.

Also wird die # durch %23 ausgetauscht, da die # die Hexadezimale 23 im ASCII-Code darstellt.

Als reserviert gelten die Zeichen ! # \$ % & ' () * + , / : ; = ? @ [| . Jedes Leerzeichen wird durch ein + ausgetauscht, welches dann jedoch nicht URL-Codiert wird.

3.1.3.1.4 Probleme

Typisierung Die Typisierung in PHP ist sehr flexible (dynamisch), so kann einer Variable, die zum Beispiel eine Zahl enthält, eine Zeichenkette, oder ein Array neu zugewiesen werden.

Manche Standard-Funktionen in PHP haben numerische Rückgabewerte und geben den bool'schen Wert false zurück, wenn ein Fehler auftritt. Da alle Werte, die nicht 0 sind, laut Definition gleich dem bool'schen true sind, kann es zu Fehlinterpretation des Rückgabewertes kommen. Um solche Situationen so vermeiden, sollte statt auf Wertegleichheit (==) auf Äquivalenz (===), das bedeutet in diesem Zusammenhang Werte- und Typgleichheit (Bool != Integer, trotz dynamischer Typisierung), geprüft werden (Beispiel: siehe Programm-Code 3.6).

```
1 <?php
2     $string = "Hallo_Welt";
3     $position = strpos("H", $string);
4     // H liegt an Position 0
5
6     // falsch:
7     if ($position == false) {
8         echo "Abfrage_1\n";
9     }
10    // richtig:
11    if ($position === false) {
12        echo "Abfrage_2\n";
13    }
14 ?>
```

Code 3.6: false

Serverseitige Realisierung Ein Problem und gleichzeitig auch ein Vorteil besteht darin, dass der Webserver die PHP-Datei compiliert und eine fertige HTML-Datei an den Client sendet. Dies muss in dieser Vorgehensweise geschehen, da der Webbrowser am Client nur HTML Code versteht. Der Vorteil liegt darin, dass keine Rechenleistung,

beziehungsweise nur soviel, damit die Webseite gerendert werden kann, am Client nötig ist. Ein weiterer Vorteil liegt darin, dass Daten, die in der PHP Datei verarbeitet werden, nicht unbedingt in dieser Form an den Client gesendet werden. Jedoch bedeutet dies auch, dass ohne neue Anfrage zum Server und daraus resultierendem Neuladen der ganzen Seite kann sich nichts am Seitenaufbau ändern. Soll sich z.B. bei Auswählen einer Check Box etwas am Aufbau der Seite ändern, so muss entweder die Seite neu geladen werden oder man greift auf andere Technologien zurück. (siehe 3.1.4.2 oder 3.1.4.3)

3.1.3.2 MySQL (Handle)

MySQL ist eine der bekanntesten und weit verbreiteten relationalen Datenbankverwaltungssysteme auf der Welt. Ursprünglich wurde es von MySQL AB entwickelt, wurde jedoch von Sun Microsystems übernommen und ist jetzt in Händen von Oracle. Es ist als Open-Source Software oder als kommerzielle Enterprise Version verfügbar.

3.1.3.2.1 SQL Dieses Datenbankmanagementsystem verwendet als Sprache für den Zugriff auf die Datenbank SQL.

SQL steht für **Structured Query Language** wurde dafür entwickelt eine einheitliche und leicht lesbare Programmierschnittstelle zur Verfügung zu stellen, um Datenbanken zu bearbeiten.

SQL ist eine Datenbanksprache für relationale Datenbanken. Sie kann dazu verwendet werden Datenbanken und Tabellen zu erstellen und bearbeiten und Datensätze zu Bearbeiten, damit ist löschen, ändern oder einfügen von Datensätzen in Tabellen gemeint. Außerdem können auch Datensätze abgefragt werden und können auch gefiltert, sortiert und vieles mehr werden. Die Voraussetzung ist, dass das Datenbanksystem auf dieser Datenbanksprache basiert.

Hier die wichtigsten Befehle die wir verwendet haben.

Beispiele für Befehle

- SELECT

- Mit SELECT können Datensätze aus Tabellen abgerufen werden. Zu diesem Befehl gibt es noch einige Zusätze, um zum Beispiel die Daten zu filtern, zu sortieren oder Tabellen miteinander zu Verknüpfen.

- * *SELECT * FROM classes;*

- Damit werden alle Datensätze aus der Tabelle classes ausgelesen.

- * *SELECT short FROM teachers;*

- Damit wird nur die Spalte der Lehrerkürzel ausgelesen, jedoch von allen Datensätzen.

- * *SELECT * FROM classes WHERE name LIKE '2 %'*

- Damit werden alle Datensätze von Klassen ausgelesen, bei denen der Name mit 2 beginnt. So erhält man alle Klassen des 2. Jahrgangs.

- * *SELECT * FROM substitutes ORDER BY time ASC*

Damit werden alle Lieferungen in aufsteigender Reihenfolge(ASC) ausgelesen. Sollen die Lieferungen absteigend ausgelesen werden verwendet man statt ASC DESC.

- * *SELECT teachers.name FROM classes LEFT JOIN teachers ON teachers.ID = classes.teacherFK*

Mit dieser Abfrage können alle KV's der Klassen abgefragt werden. Da wir in unserem Datenbankdesign auf Redundanzvermeidung geachtet haben, verweisen wir in der classes Tabelle auf die teachers Tabelle. Dort stehen die Namen der KV's. In der classes Tabelle steht nur die eindeutige ID des Lehrers.

SELECT teachers.name FROM classes LEFT JOIN teachers ON teachers.ID = classes.teacherFK

Gibt alle Einträge der Tabelle classes zurück, jedoch nur die Einträge von der teachers Tabelle, die die ON Bedingung erfüllen.

SELECT teachers.name FROM classes INNER JOIN teachers ON teachers.ID = classes.teacherFK

Gibt nur diejenigen Datensätze zurück, die die ON Bedingung erfüllen.

SELECT teachers.name FROM classes RIGHT JOIN teachers ON teachers.ID = classes.teacherFK

Gibt alle Einträge der Tabelle teachers zurück und nur die Einträge der Tabelle classes die die ON Bedingung erfüllen.

SELECT teachers.name FROM classes OUTER JOIN teachers ON teachers.ID = classes.teacherFK

Gibt alle Einträge aus beiden Tabellen zurück. Die Einträge, die die ON Bedingung erfüllen werden zusammengefügt und die anderen werden jeweils als einzelner Datensatz angezeigt.

- * *SELECT name as Klassenname FROM classes*

Mit as können die Spaltennamen der Tabelle in besser lesbaren oder besser identifizierbaren Spaltennamen umbenannt werden.

- INSERT

- Mit dem INSERT Befehl können neue Datensätze in eine Tabelle eingefügt werden.

- * *INSERT INTO sections (name,short,teacherFK) VALUES ('TestAbteilung','T','2')*

Damit wird eine neue Abteilung in der Tabelle sections erstellt. Dabei werden die angegebenen Spalten auf die in der Klammer stehenden Werte gesetzt. Hier muss die Spalte ID nicht angegeben werden, da sie auf *auto increment* gesetzt ist.

- UPDATE

- Mit dem UPDATE Befehl können Datensätze in einer Tabelle verändert werden.

- * *UPDATE sections SET name = 'TestAbteilung2' WHERE name = 'TestAbteilung'*

- Mit diesem UPDATE Befehl wird jeder Datensatz der in der Spalte name TestAbteilung stehen hat abgeändert. Dabei wird TestAbteilung in TestAbteilung2 geändert.

- * *UPDATE sections SET name = 'TestAbteilung123',short = 'T123' WHERE name = 'TestAbteilung2'*

- Dies funktioniert im Prinzip gleich wie der Befehl zuvor. Mit dem Unterschied, dass dabei 2 Spalten geändert werden.

- DELETE

- Mit DELETE können einzelne Datensätze oder auch mehrere Datensätze auf einmal gelöscht werden.

- * *DELETE FROM sections WHERE name = 'TestAbteilung123'*

- Damit werden alle Datensätze mit name = TestAbteilung123 gelöscht. Ohne dem WHERE Parameter werden alle Datensätze aus der Tabelle sections gelöscht.

- TRUNCATE

- Mit TRUNCATE können alle Datensätze aus einer Tabelle gelöscht werden. Im Unterschied zu DELETE wird hier auch der Index zurückgesetzt. Bei DELETE werden die neuen Datensätze fortlaufend weiter mit dem letzten Index nummeriert. Bei TRUNCATE beginnt die Nummerierung von vorne.

- * *TRUNCATE FROM classes*

- Dabei wird die ganze Tabelle classes zurückgesetzt und damit auch alle Datensätze gelöscht.

3.1.3.2.2 Aufbau von MySQL Bei MySQL ist es im Normalfall so, dass die Datenbank/en auf einem MySQL Server liegen und die MySQL-Client/s Anfragen auf diesen Server senden.

Ein Server kann mehrere Datenbanken beinhalten. Eine solche Datenbank kann jeweils mehrere Tabellen beinhalten. Jede Tabelle hat mehrere Spalten.

Jede Spalte hat einen bestimmten Datentyp. Hier einige wichtige Datentypen, die wir benutzen:

- TINYINT

- Wird für die Speicherung von BOOL Werten verwendet, da es den wenigsten Speicherplatz benötigt

- **INT**
Wird für die Speicherung von Ganzzahlen verwendet.
- **DATE**
Wird für die Speicherung von Datumsangaben verwendet.
- **TEXT**
Wird für die Speicherung von Texten verwendet.

3.1.3.2.3 Verarbeitung von Anfragen Eine Anfrage, die an einen MySQL-Server gesendet werden, durchläuft wird grundsätzlich mit diesen Schritten abgearbeitet. Zuerst wird im Query-Cache nachgeschaut ob ein gleicher Query schon einmal gestellt wurde und ob sich die Tabelle seither nicht mehr geändert hat. Ist es ein neuer Query, dann wird der Query an den Parser gesendet und anschließend wird der Query optimiert und das Ergebnis wird zurückgeliefert.

Query-Cache Wird ein Query an den MySQL-Server gesendet wird zuerst im Query-Cache nachgeschaut, ob der selber Query schon einmal gestellt wurde. Ist dies der Fall, so muss noch kontrolliert werden ob sich die Daten in der Datenbank seither geändert wurden. Hat sich nichts verändert, so sendet der MySQL-Server die gespeicherten Ergebnisse zurück. Dies hat den Vorteil, dass der MySQL-Server nicht wieder den Query ausführen muss und ist somit Ressourcensparender.

Parsing Beim Parsing wird der gestellte Query in seine einzelnen Befehle zerlegt. Außerdem werden alle Informationen über den Query gesammelt. Wie zum Beispiel wird die Art(SELECT,DELETE,UPDATE,...) des Querys bestimmt. Ist das Parsing abgeschlossen so ist ein so genannter Parse-Baum erstellt worden. Dieser wird anschließend Optimiert.

Optimierung In diesem Schritt wird der geparsete Code optimiert. Dabei wird der Query analysiert, so werden die besten Abfragereihenfolgen ausarbeitet. Es wird bestimmt, welche Tabellen wie und wann gejoined werden. Außerdem wird kontrolliert ob alle Tabellen die im Query vorkommen auch wirklich benötigt werden. Ist die optimale Abfragereihenfolge gefunden wird der Query ausgeführt und das Ergebnis wird zurückgegeben.

3.1.3.2.4 Administration MySQL liefert Standardmäßig einen Kommandozeilen-Client mit. Dieser kann in der Konsole mit *mysql* aufgerufen werden. Damit können Datenbanken, Tabellen erstellt und bearbeitet werden, Datensätze können bearbeitet, verändert und gelöscht werden. Es können noch viele weitere Funktionen damit genutzt werden.

Grafische Tools Es gibt noch zahlreiche grafische Tools MySQL Server zu administrieren. Eines der am weit Verbreitetsten grafischen Tools ist phpMyAdmin. Dies ist ein Tool, das auf einen Webserver kopiert wird und es erlaubt über ein Webinterface alle Einstellungen und Datenbankbefehle grafisch ausführen zu können. Dies erleichtert und vereinfacht das Erstellen, Bearbeiten und Verändern von Datenbanken und Tabellen. Damit können auch ungeübte Personen MySQL Datenbanken verwenden.

3.1.3.2.5 MySQL ⇒ PHP Um über PHP auf einen MySQL Server zugreifen zu können benötigt man das PHP-Modul php5-mysql, welches einige MySQL Befehle zur Verfügung stellt um mit einem MySQL-Server zu kommunizieren.

Verbindung aufbauen Um mit einem MySQL-Server eine Verbindung aufzubauen müssen diese PHP Befehle verwendet werden(siehe Programm-Code 3.7).

```
1 <?php
2     mysql_connect($host, $user, $passwd);
3     mysql_select_db($db);
4 ?>
```

Code 3.7: MySQL Connect

Wobei in diesem Beispiel entspricht *\$host* der Adresse des MySQL-Server, liegt der Webserver, von wo aus die Anfrage gesendet wird und er MySQL Server am selben Server so ist dies *localhost*. *\$user* ist der Usernamen zum anmelden auf dem MySQL Server und *\$pass* ist das Passwort für die Anmeldung am MySQL Server. Mit *\$db* wird die Datenbank angegeben mit der gearbeitet werden soll. Die Funktion *mysql_connect()* gibt im Fehlerfall FALSE zurück und im Erfolgsfall einen Ressource Wert. *mysql_select_db()* gibt bei Erfolg TRUE zurück, sonst FALSE.

Anfragen Um Anfragen an einen MySQL Server senden zu können muss man diesen Syntax verwenden(siehe Programm-Code 3.8)

```
1 <?php
2     $sql="SELECT * FROM classes";
3     mysql_query($sql);
4 ?>
```

Code 3.8: MySQL Querys

In der unter 3.8 gezeigten Programmsequenz ist eine Anfrage an den MySQL Server gezeigt. Wichtig ist, dass zuvor die Zeilen vom Code 3.7 stehen, damit eine Anfrage an den Server gesendet werden kann. *mysql_query* gibt bei Datenbankabfragen(SELECT, SHOW, EXPLAIN, ...) das Abfrageergebnis im Erfolgsfall zurück, bei Fehlerfall FALSE. Wird ein anderer Datenbankbefehl(INSERT, DELETE, UPDATE, ...) ausgeführt, so gibt die Funktion bei Erfolg TRUE und bei Misserfolg FALSE zurück. Das Ergebnis ist in Zeilen aufgebaut, d.h. will man jede Zeile abarbeiten, muss man jede

Zeile zum Beispiel mit einer do-while-Schleife durchlaufen. Ein interner Positionszeiger verweist auf die momentane Stelle. Die Funktionen zum Abfragen dieser Zeilen stellt den Zeiger immer eine Stelle weiter.

Das Ergebnis bei einer Abfrage ist jedoch nicht in einem Format mit dem man arbeiten kann, deshalb muss dieses Ergebnis noch an weitere Funktionen weitergegeben werden. Hier die von uns am häufigsten verwendeten Funktionen:

- `mysql_fetch_array()`
 - Mit dieser Funktion wird eine Zeile des Ergebnisses gelesen und in ein Array geschrieben. Zusätzlich wird der Positionszeiger eine Stelle weiter gestellt. Die Indizes des Arrays werden numerisch nummeriert und sind nochmals mit den Spaltennamen nummeriert. Das heißt, die Ergebnisse sind in diesem Array doppelt vorhanden
- `mysql_fetch_object()`
 - Mit dieser Funktion wird die aktuelle Zeile des Ergebnisses ausgelesen und als Objekt zurückgegeben. Iwederum wird der Positionszeiger eine Stelle weiter gestellt.
- `mysql_fetch_row()`
 - Diese Funktion gibt eine Zeile des Ergebnisses als Array zurück das numerisch nummerierte Indizes hat. Das heißt hier sind aus dem Array die Spaltennamen nicht ersichtlich. Auch hier wird der Datenzeiger um eine Stelle weiter gestellt.

Im Programm Code 3.9 ist ein Ausschnitt einer Abarbeitung eines MySQL Results zu sehen. Hier wird zuerst ein Query an den Server gesendet und danach werden alle Ergebniszeilen durchlaufen und in ein Array gespeichert. Dazu wird `mysql_fetch_array()` verwendet. Dies ist die Funktion die ich persönlich bevorzuge.

```

1 <?php
2     $sql = "SELECT * FROM hours LIMIT 0,5"; //Mit LIMIT wird die
3         Abfrage auf die ersten 5 Ergebnisse begrenzt
4     $result = mysql_query($sql); //Ergebnis wird in der Variable
5         $result gespeichert
6
7     while($row = mysql_fetch_array($result)){           //Jede
8         Ergebniszeile wird durchlaufen
9         $ergebnis[] = $row;           //Hier wird die Zeile in
10            einen Eintrag des Arrays geschrieben
11     }
12
13     print_r($ergebnis);           //Relationale Ausgabe des Arrays mit
14         allen Zeilen
15
16 ?>

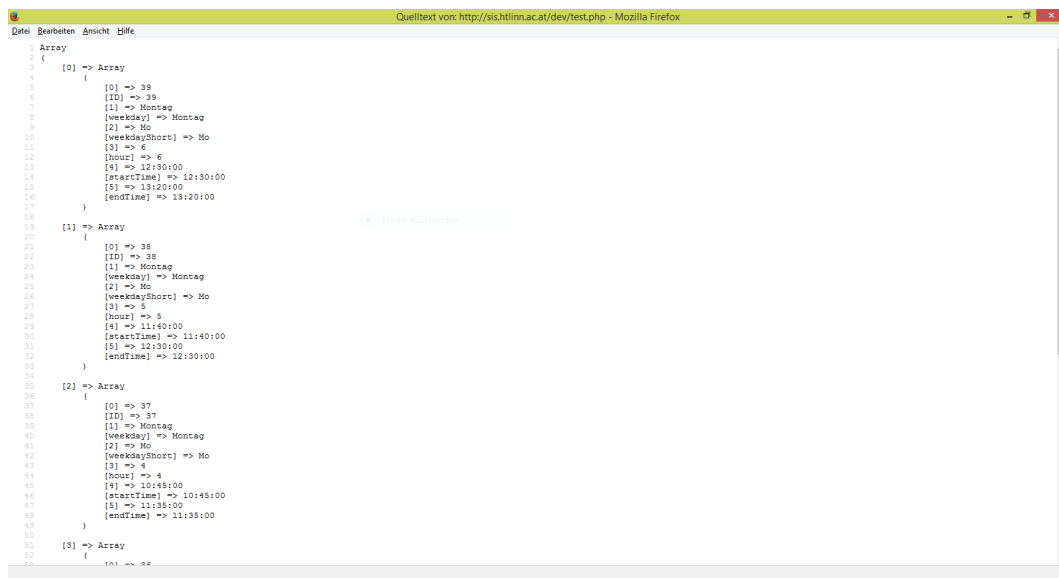
```

Code 3.9: MySQL Query weiterverarbeiten

Öffnet man den Quelltext der oberen Seite, wiederum muss zuvor natürlich der Programm-Code 3.7 in dem PHP Code eingefügt werden, um überhaupt eine Verbindung mit dem MySQL Server herstellen zu können, dann wird das Array *\$ergebnis* schön dargestellt. Das Ergebnis ist in der Abb. 3.2 zu sehen.

Hier ist gut zu sehen, dass bei *mysql_fetch_array()* die Spalten numerisch und laut Spaltenname indiziert werden. Dies kann durch dem zweiten Parameter in der Funktion *mysql_fetch_array()* geändert werden. Außerdem ist hier zu sehen, dass jede Ergebniszeile einen eigenen Index (numerisch Nummeriert) in dem Array *\$ergebnis* hat, daraus ergibt sich ein Mehrdimensionales Array.

Auf die erste Zeile des Ergebnisses kann also mit *\$ergebnis[0]* zugegriffen werden. Das Ergebnis dieses Codes ergibt Abb. 3.3. Um dann auf die Start-Uhrzeit der Stunde von der ersten Zeile zugreifen zu können, kann man entweder den numerischen Index oder die Spaltenbezeichnung verwenden. Also entweder *\$ergebnis[0][4]* oder *\$ergebnis[0]['startTime']*. Das Ergebnis dieser Zeilen ist dasselbe, dieses ist in Abb. 3.4 zu sehen.



```
Array
(
    [0] => Array
        (
            [0] => 39
            [ID] => 39
            [1] => Montag
            [weekday] => Montag
            [2] => Mo
            [weekdayShort] => Mo
            [3] => 6
            [hour] => 6
            [4] => 12:30:00
            [startTime] => 12:30:00
            [5] => 13:20:00
            [endTime] => 13:20:00
        )
    [1] => Array
        (
            [0] => 38
            [ID] => 38
            [1] => Montag
            [weekday] => Montag
            [2] => Mo
            [weekdayShort] => Mo
            [3] => 5
            [hour] => 5
            [4] => 11:40:00
            [startTime] => 11:40:00
            [5] => 12:30:00
            [endTime] => 12:30:00
        )
    [2] => Array
        (
            [0] => 37
            [ID] => 37
            [1] => Montag
            [weekday] => Montag
            [2] => Mo
            [weekdayShort] => Mo
            [3] => 4
            [hour] => 4
            [4] => 10:45:00
            [startTime] => 10:45:00
            [5] => 11:35:00
            [endTime] => 11:35:00
        )
    [3] => Array
        (
            [0] => 36
            [ID] => 36
            [1] => Montag
            [weekday] => Montag
            [2] => Mo
            [weekdayShort] => Mo
            [3] => 3
            [hour] => 3
            [4] => 10:00:00
            [startTime] => 10:00:00
            [5] => 10:45:00
            [endTime] => 10:45:00
        )
)
```

Abbildung 3.2: MySQL Fetch Array

```

Array
(
    [0] => 39
    [ID] => 39
    [1] => Montag
    [weekday] => Montag
    [2] => Mo
    [weekdayShort] => Mo
    [3] => 6
    [hour] => 6
    [4] => 12:30:00
    [startTime] => 12:30:00
    [5] => 13:20:00
    [endTime] => 13:20:00
)

```

Abbildung 3.3: MySQL Zeile

```

1 12:30:00

```

Abbildung 3.4: MySQL Spalte

3.1.3.2.6 SQL Injection Mit SQL Injections versucht ein Angreifer Daten aus der Datenbank auszulesen, Daten zu manipulieren und noch einige weitere Dinge, die nicht im Auge des Programmierers liegen, zu machen. Diese Injections sollen natürlich so gut es geht vom Programmierer verhindert werden, dazu stellt PHP einen Befehl zur Verfügung. Dieser lautet: `mysql_real_escape_string()`.

Beispiele Bei dem Programm-Code 3.10 ist gezeigt wie eine Injection erfolgreich durchgeführt werden kann. Dabei wird bei einer einfachen Anmeldung die Anmeldung ausgetrickst und man kann sich ohne Passwort anmelden.

```

1 <?php
2     $sql = "SELECT * FROM users WHERE name='$_POST['username
3         ']' AND password='$_POST['password']'"; //Die
4         mitgegebenen Werte werden in den Query übernommen
5     mysql_query($sql); //Query wird ausgeführt
6 ?>

```

Code 3.10: MySQL Injection: Falsch

Sofern der User wirklich einen regulären Benutzernamen und ein reguläres Passwort eingibt macht diese Vorgangsweise keine Probleme.

Nehmen wir jetzt an, dass der User ein Angreifer ist und sich ohne ein Passwort zu kennen Zugriff zum Passwortgeschützten Bereich machen will, dann macht der Programm-Code

3.10 Probleme.

Beispiel:

- `$_POST['username'] = 'asdf'`
Dieser Benutzername muss nicht existieren in der Datenbank
- `$_POST['password'] = " OR ""`
Damit wird ein Ausdruck angefügt der immer 1 ergibt

→ *SELECT * FROM users WHERE name='asdf' AND password="" OR ""*

Mit dieser Eingabe eines Angreifers bekommt man bei diesem Select Query immer ein Ergebnis zurück. Welches, kommt drauf an wie es ausgewertet wird, zum erfolgreichen einloggen führen kann. Dies kann mit dem oben erwähnten Befehl verhindert werden. Siehe Programm-Code 3.11

```

1 <?php
2     $user = mysql_real_escape_string($_POST['username']);
3     $passwd = mysql_real_escape_string($_POST['password']);
4     $sql = "SELECT * FROM users WHERE name='$_user' AND password='
           $passwd'"; //Die kontrollierten Werte werden in den Query
           übernommen
5     mysql_query($sql); //Query wird ausgeführt
6
7 ?>

```

Code 3.11: MySQL Injection: Richtig

Geht man davon aus, dass die selben Werte vom Angreifer eingegeben werden, so kommt es jetzt nicht zu einem Ergebnis, da die Funktion alle nicht erlaubten Zeichen 'eliminiert'. Sie werden nicht direkt eliminiert, sondern die nicht erlaubten Zeichen werden nicht gelöscht, sondern mit einem \ maskiert. Also aus einem ' wird ein \', aus einem " wird ein \" usw. Die Zeichen, die mit einem Backslash maskiert werden, sind ",',\n,\r,\,\x00,\x1a. Es können nicht nur wie oben gezeigt SELECT-Befehle verändert werden, sondern es können ganze Befehle in Befehle eingefügt werden. Dadurch kann ein Angreifer zum Beispiel einen neuen Datensatz anlegen, Datensätze löschen und so weiter.

```

1 <?php
2     $ID = $_GET['ID'];
3
4     $sql = "SELECT * FROM users WHERE ID={$ID}";
5     mysql_query($sql); //Query wird ausgeführt
6
7     //Mitgegebener GET Wert = 2
8     //Query : SELECT * FROM users WHERE ID=2
9
10    //Mitgegebener GET WERT = 2; UPDATE users SET type="admin"
11    WHERE ID=23
12    //Query : SELECT * FROM users WHERE ID=2; UPDATE users SET
13    type="admin" WHERE ID=23

```

```
12      //Damit kann zum Beispiel einem User eine andere Berechtigung
13      ?>      gegeben werden
```

Code 3.12: MySQL Injection: Beispiel Query in Qeury

Im Code 3.12 ist gezeigt wie ein ANgreifer zum Beispiel einem beliebigen User die Berechtigungen admin geben kann. Was natürlich eine sehr große Sicherheitslücke darstellt. Dies kann wiederum mit der Funktion `mysql_real_escape_string()` verhindert werden.

3.1.3.3 Datenbank-Design (Buchberger)

Bei relationellen Datenbanken, wie das für das Projekt verwendete MySQL, gelten grundsätzliche Richtlinien, um die Konsistenz der Daten zu gewährleisten:

Entitätsintegrität

Die Kennung eines Datensatzes (Primärschlüssel) muss auf jeden Fall eindeutig sein. Dies kann durch die MySQL-Eigenschaft `auto increment` erreicht werden.

Referenzielle Integrität

Verweise auf andere Tabellen (Fremdschlüssel) müssen auf Datensätze zeigen, die existieren. Alternativ kann NULL zugewiesen werden.

Beispiel: Eine Schulstunde hat einen Lehrer auf den mit einem Fremdschlüssel verwiesen wird. Der Lehrer verlässt die Schule und sein Datensatz wird gelöscht.

Die Stunde hat weiterhin den gleichen Lehrer, welcher aber nicht mehr existiert.

Bereichsintegrität

Die Werte eines Datenfeldes müssen in einem definierten Bereich liegen.

Weitere wichtige Grundsätze sind:

Redundanzen sind zu vermeiden

Anders formuliert: Jede Information soll möglichst nur einmal in der Datenbank vorhanden sein.

Redundanzen sorgen zwangsweise für Probleme, wenn Datensätze aktualisiert werden sollen.

Beispiel: Bei einem großen Geschäft wohnen mehrere Kunden aus dem selben Ort. Ändert sich nun die Postleitzahl des Ortes, so müssen alle Datensätze aktualisiert werden.

Lösung: Auslagern des Ortes und der Postleitzahl in eine weitere Tabelle und via Fremdschlüssel verknüpfen.

So muss bei Änderung der Postleitzahl nur an einer Stelle geändert werden.

Es stellt sich allerdings die Frage, wie weit die Datenstruktur aufgefächert werden soll.

Gerechnet wird von der Datenbank

Die Sprache, mit der die Abfrage an die Datenbank gestellt wird, soll die Daten nicht mehr nachbearbeiten müssen.

Gründe dafür sind, dass die Datenbank für die Verarbeitung von Daten optimiert ist, und dass es beim sequenziellen Abfragen von Tabellen zu Fällen kommen kann, in denen eine Tabelle bereits aktualisiert ist, die verknüpfte Tabelle aber nicht.

3.1.3.4 LDAP (Buchberger)

LDAP (Lightweight Directory Access Protocol) ist ein Protokoll für den Zugriff auf Verzeichnis-Datenbanken, wie Microsoft Active Directory, Apple Open Directory, openLDAP oder Novell eDirectory.

Verzeichnis-Dienste werden dazu verwendet, um Informationen über Personen oder Rechner-Konfigurationen zu speichern und abzufragen.

3.1.3.4.1 LDAP-Datenstruktur Eine LDAP-Datenstruktur ist Baum-förmig (DIT - Directory Information Tree) aufgebaut.

Dies hat den Grund, dass man so geografische und/oder organisatorische Gegebenheiten besser in der Datenbank darstellen kann.

Man unterscheidet zwischen Container-Objekten (enthalten ihrerseits weitere Objekte) und Blatt-Objekten (Sie stellen Enden des Baumes dar).

LDAP ist objektorientiert aufgebaut, das heißt, es gibt Objekt-Klassen (wird durch das Schema definiert), welche Eigenschaften der Objekte vorgeben. Ein LDAP-Objekt besitzt mindestens eine Klasse.

Alle Objekte sind durch ihren DN (Distinguished Name) vollständig identifiziert. Der DN besteht aus dem Attribut, das das Objekt innerhalb des darüberliegenden Container-Objektes eindeutig macht, zusätzlich zum DN des Container-Objektes.

Beispiel:

uid=mueller,ou=beratung,ou=filiale1,c=at,o=firma

Die uid=mueller identifiziert so ein Objekt (vermutlich ein Benutzer) in der ou=beratung. Diese ist wiederum einzigartig in der ou=filiale1, u.s.w.

Das lässt sich nun so fortsetzen. Das oberste Container-Objekt wird als Root-Objekt bezeichnet. In den meisten Fällen wird dies mit dem Attribut o (für Organisation) gekennzeichnet.

Die Eigenschaften der Objekt sind oft abgekürzt.

Häufige Abkürzungen:

- o** organisationName (Organisation)
- st** stateOrProvinceName (Staat/Provinz)
- c** country (Land)
- ou** organizational unit (Organisations Einheit)
- cn** common name (Allgemeiner Name)
- mail**
 e-mail-address (E-Mail-Adresse)

Die Objekt-Klassen definieren die Eigenschaften der Objekt.

Beispiel: Attribute der Objekt-Klasse inetOrgPerson (Definition des Standard-Schemas initOrgPerson von openLDAP in der Version 1.4.2.6)

- erbt von**
 organizationalPerson
- erbt von**
 person
- erbt von**
 top (abstrakte Klasse; keine Attribute)
- !sn**
 sur name (Nachname)
- !cn**
 common name (Allgemeiner Name)
- userPassword**
 Benutzerpasswort
- telephoneNumber**
 Telefon Nummer
- seeAlso**
 Verweist (als dn)
- description**
 Beschreibung
- title**
 Titel
- x121Address**
 wurde für das x.121-Protokoll verwendet

registeredAddress

Registrierungs Adresse

destinationIndicator

wurde für Telegram-Services verwendet

preferredDeliveryMethod

Bevorzugte Liefer Methode

telexNumber

wurde für Telex verwendet

teletexTerminalIdentifier

wurde für Teletex verwendet

telephoneNumber

Telefon Nummer

internationalISDNNumber

ISDN Nummer

facsimileTelephoneNumber

Fax Nummer

street

Straßen Adresse

postOfficeBox

Postfach

postalCode

Postleitzahl

postalAddress

Post Adresse

physicalDeliveryOfficeName

Postbüro Name

ou organisation unit (Organisations Einheit)**st** stateOrProvinceName (Staat/Provinz)**l** localityName (Ort/Ortschaft)**audio**

Audio

businessCategory

Geschäfts-Kategorie

carLicense

Fahrzeug-Lizenz

departmentNumber

Abteilung innerhalb eines Unternehmens

displayName

bevorzugter Anzeigename

employeeNumber

Angestellten Nummer

employeeType

Anstellungsart

givenName

Vorname(n)

homePhone

Heim-Telefon-Nummer

homePostalAddress

Heim-Adresse

initials

Initialen

jpegPhoto

JPEG Foto

labeledURI

Uniforme Ressourcen ID mit optimaler Bezeichnung

mail

e-mail-address (E-Mail-Adresse)

manager

dn des Managers

mobile

mobile Telefon-Nummer

o organisationName (Organisation)**pager**

Telefon-Nummer des Pagers

photo

Foto

roomNumber

Raum Nummer

secretary

dn des Sekretärs

uid Benutzer Kennung

userCertificate
Benutzer Zertifikat

x500uniqueIdentifier
ID für das X.500 Protokoll

preferredLanguage
bevorzugte Sprache

userSMIMECertificate
SignedData für S/MIME

userPKCS12
PKCS #12 PFX PDU für den Austausch von persönlichen Identitäts-Informationen

3.1.3.4.2 LDAP-Suche Für die Suche in einer LDAP-Datenstruktur wird als erstes ein baseDN festgelegt, dieser entspricht dem Objekt, ab dem im Baum gesucht werden soll. Weiters wird der Scope festgelegt, hierbei gibt es 3 verschiedene Möglichkeiten:

base
Hierbei wird nur das Objekt mit dem baseDN durchsucht.

one Eine Ebene unterhalb des baseDN wird gesucht.

sub Alle darunter liegende Objekte sowie das, durch den baseDN referenzierte, werden durchsucht.

Die Suchkriterien werden in der Polnischen Notation (Präfixnotation) geschrieben und dürfen Wildcards enthalten.

Beispiel:

baseDN: ou=STUDENTS,o=HTLinn

scope: sub

(&(l=5*H*)(mail=Andreas*)(cn=2009*))

Diese Suchanfrage würde alle Objecte liefern, die

- unterhalb von ou=STUDENTS,o=HTLinn liegen
- und die die Eigenschaften mail, l und cn enthalten
- und bei denen der cn mit 2009 beginnt
- und deren E-Mail-Adresse mit Andreas beginnt
- und deren Ort-Attribut mit 5 beginnt und ein H enthält.

3.1.4 Clientseitige Technologien

3.1.4.1 HTML (Weiland)

HTML steht für “Hypertext Markup Language“ und bezeichnet eine Auszeichnungssprache. HTML-Dateien werden hauptsächlich für Websites verwendet.

3.1.4.1.1 Aufbau Eine HTML-Datei ist grundsätzlich immer gleich aufgebaut. Sie besteht aus einem Header in dem unter Anderem der Titel und die Meta-Daten bestimmt werden. Im Body steht der Inhalt der Datei, welcher angezeigt werden soll.

```
1 <!DOCTYPE html>
2 <html>
3     <head>
4         /* Datei-Kopf */
5     </head>
6     <body>
7         /* Inhalt der Datei */
8     </body>
9 </html>
```

Code 3.13: HTML-Tags

3.1.4.1.2 HTML5 HTML5 ist die aktuellste Version von HTML. Die Entwicklung begann am 29. April 2009 und soll im Jahr 2014 fertiggestellt werden.

Im Gegensatz zu früheren Versionen von HTML wird die Wiedergabe von Video- und Audiodateien unterstützt. Jedoch sind aktuell noch nicht alle Browser fähig diese Funktionen zu verwenden.

Die unterstützten Formate sind für Videodateien Ogg Theora, MP4(H.264) und WebM(VP8) und für Audiodateien Ogg Vorbis, MP3 und Wav.

3.1.4.2 Javascript (Klotz)

JavaScript ist eine Skriptsprache welche für dynamische Erweiterungen im Webbrowser gedacht war. Es sollte helfen interaktive Websites, einfacher zu erstellen. Daher ist JavaScript auch nicht als alleinstehende Programmiersprache geeignet, und kommt fast immer in Kombination mit anderen Programmen oder Programmiersprachen zum Vorschein.

JavaScript wird hauptsächlich Clientseitig, das heißt im Webbrowser des Nutzers, genutzt um für den Nutzer Extras einzubauen. Dadurch wird auch der Server entlastet, da die Rechenleistung des Anwender-PCs genutzt wird und nicht die des Servers.

Es gibt aber auch Serverseitiges-JavaScript(SSJS). Das kann dann, mit einem entsprechenden Interpreter (z.B.: Node.js), gleich wie andere Serverseitige Programmiersprachen verwendet werden.

Da es sich bei JavaScript um eine Skriptsprache handelt ist sie sehr einfach gehalten und Variablendeklarationen und ähnliches fallen weg (Variablentypen können auch während Laufzeit einfach verändert werden).

JavaScript ist eine Interpreter-Programmiersprache, das heißt, dass der Code erst im Webbrowser durch den integrierten Interpreter umgesetzt wird.

Obwohl es sich bei JavaScript nur um eine Skriptsprache handelt sind trotzdem alle wichtigen Strukturelemente, wie Schleifen, If-Klauseln, etc. welche auch in anderen Programmiersprachen wie zum Beispiel C oder Java existieren, enthalten.

Weiters gibt es bei JavaScript die Möglichkeit Bibliotheken einzubinden. Dadurch kann man bereits vorgefertigte Funktionen einfach einbinden und anwenden. Ein Beispiel dafür ist JQuery.

Anwendung

Um JavaScript in einer Webseite zu nutzen gibt es zwei Möglichkeiten, entweder der Code wird direkt in das HTML-Dokument geschrieben (Variante 1) oder man schreibt den JavaScript-Code in ein eigenes Dokument und bindet dieses dann ein (Variante 2).

Variante 1

Wenn man den Code direkt in die HTML-Datei schreiben möchte, muss dieser Code dementsprechend markiert werden. Dazu verwendet man den Script-Tag (siehe Codebeispiel), dieser kann im Head oder im Body stehen, es ist jedoch üblich Funktionen in den Head zu schreiben.

```
1 <html>
2   <head>
3     <script>
4       //Here comes JS
5     </script>
6   </head>
7
8   <body>
9   </body>
10 </html>
```

Variante 2

Im Gegensatz zur ersten Variante kommt in diesem Fall nur die Verlinkung zur JavaScript-Datei und nicht der ganze Code in das HTML-Dokument. Das eingebundene JavaScript-Dokument muss nicht auf dem gleichen Server oder Rechner gespeichert sein wie die Webseite, der Code kann sogar aus dem Internet geladen werden. Zum Einbinden einer JavaScript-Datei wird wieder ein Script-Tag verwendet, diesmal wird aber die URL der JavaScript-Datei als Attribut („src =“) mitgegeben.

```
1 <script src="js/jquery.js" type="text/javascript"></script>
```

Funktionen

In JavaScript kann man auch Funktionen schreiben, welche dann bei Eintreten bestimmter Ereignisse ausgeführt werden. Der Code für eine JavaScript-Funktion sieht wie folgt aus:

```
1 function functionname()  
2 {  
3   some code to be executed  
4 }
```

Zwischen den geschweiften Klammern wird der Code, der ausgeführt werden soll, geschrieben.

Um eine Funktion aufzurufen schreibt man den Funktionsnamen mit Klammern dahinter. Der Funktionsaufruf kann zwischen zwei Script-Tags geschrieben werden oder als bestimmtes Attribut (z.B. „onClick“) in manchen anderen Tags.

Bei einem Funktionsaufruf kann man der Funktion auch Werte mitgeben, dazu muss man in der Klammer hinter dem Funktionsnamen die Werte/Variablen eintragen. Die Mitgabe von Parametern muss natürlich in der Funktion vorgesehen werden, ansonsten werden die Werte einfach ignoriert.

Variablen

Bei JavaScript gibt es, wie auch bei anderen Programmiersprachen, Variablen. Diesen muss aber im Gegensatz zu Programmiersprachen wie C, Java oder ähnlichen, bei der Deklaration, kein eindeutiger Variablentyp zugewiesen werden. In JavaScript nur zwei Unterscheidungen bei den Variablentypen, nämlich in Zahlen und in Zeichen bzw. Zeichenketten. Um in JavaScript eine Variable zu deklarieren, schreibt man var und dann den Variablennamen. Um ihr dann noch einen Wert zuzuweisen muss man nur den Variablennamen und ein = Symbol schreiben, danach wird der gewünschte Wert hingeschrieben, falls es sich um eine Zeichenkette handelt muss man den Wert zwischen Anführungszeichen setzen:

```
1 var name = "Peter";  
2 var Anzahl;  
3 Anzahl = 3;  
4 name = "Hans";
```

Weiters muss man zwischen globalen und lokalen Variablen unterscheiden, während globale Variablen im gesamten Dokument definiert sind, sind lokale Variablen nur in der

Funktion in der sie deklariert werden nutzbar. Um eine Variabel global zu definieren muss sie außerhalb jeglicher Funktionen definiert werden, wenn man eine Variabel aber innerhalb einer Funktion definiert handelt es sich um eine lokale Variable.

Arrays

In JavaScript gibt es auch Felder(Arrays). In diesen Feldern können sowohl Zahlen als auch Zeichen bzw. Zeichenketten gespeichert werden, es ist sogar möglich Zahlen und Zeichenketten in dasselbe Array zu speichern.

Einen Wert in einem Array zu speichern funktioniert gleich, wie einer Variable einen Wert zuzuweisen, aber bei einem Array muss man zusätzlich zum Arraynamen noch die angeben welchen Arrayeintrag man verändern möchte.

```
1 var cars = new Array();  
2 cars[0] = "Audi";  
3 cars[1] = "BMW";  
4 cars[2] = "Mercedes";
```

JSON

JSON (JavaScript Object Notation) ist ein Datenaustauschformat welches so gestaltet ist, dass Menschen es leicht lesen können und der Computer es einfach parsen kann. Es basiert auf JavaScript.

JSON wird sehr häufig in Verbindung mit JavaScript verwendet, kann aber auch mit anderen Programmiersprachen verwendet werden.

Beispiel für ein JSON-Objekt:

```
1 {  
2   "äGert": "Auto",  
3   "Marke": "Audi",  
4   "Farbe": "rot",  
5   "Seriennummer": 02345032,  
6 }
```

Die Daten werden in Form von Daten-Wert Paaren gespeichert. Aus diesem Objekt kann man nur relativ einfach Daten auslesen. Der JavaScript-Code um die Marke auszu- lesen würde zum Beispiel wie folgt aussehen: var Marke = object.Marke;

3.1.4.3 AJAX (Buchberger)

3.1.5 Mobil App (Klotz)

3.1.5.1 PhoneGap

PhoneGap ist ein Framework, von Adobe Systems, um mobile Apps zu erstellen. Diese Apps sind jedoch weder Web-Apps noch native Apps. Hierbei handelt es sich um Hybrid-Apps. Das heißt die App verwendet nicht die nativen Userinterface Frameworks um das Layout zu gestalten, sondern Web-Technologien, aber die App arbeitet trotzdem vollkommen lokal auf dem Gerät.

Das funktioniert indem die App praktisch im Webbrowser des eigenen Gerätes ausgeführt wird, aber alle browsertypischen Eigenschaften, wie zum Beispiel der Rahmen, die URL-Leiste oder die Einstellungen deaktiviert oder ausgeblendet werden.

Im Gegensatz zu herkömmlichen Web-Applikationen, kann man mit PhoneGap-Apps auch auf Funktionen wie zum Beispiel den Beschleunigungssensor oder die Kamera zu nutzen. Das wird durch die PhoneGap API ermöglicht. Dabei handelt es sich um Java-Dateien die bei der Installation der App heruntergeladen werden. Mit JavaScript kann man dann über diese Java-Dateien auf die Geräteinternen Sensoren zugreifen. Es handelt sich hierbei also um einen Kompromiss aus Web-Entwicklung und nativer App-Entwicklung.

Um PhoneGap zu nutzen muss man sich für alle Systeme, auf denen die App nach der Entwicklung betrieben werden soll, ein SDK installieren, zum Beispiel für Android Eclipse, für iOS XCode oder für WindowsPhone das Microsoft SDK. In dieses SDK muss nun PhoneGap als Plugin geladen werden und mit diesem Plugin kann man die App auch mit Webtechnologien (HTML, CSS, JS) anstatt gerätespezifischer Programmiersprachen (Java, ObjectiveC, VisualC) entwickeln.

Unterstützte mobile Betriebssysteme bis Version(2.9):

Android

iOS

Windows Phone 7 & 8

Blackberry

WebOS(HP)

Tizen

Symbian

Bada

Ab PhoneGap-Version 3.0 werden nur noch Android, iOS und Windows Phone unterstützt.

PhoneGap basiert auf dem Open-Source-Projekt Apache Cordova. Daher darf PhoneGap auch vollkommen kostenlos genutzt werden.

PhoneGapBuild

Für dieses Projekt wurde PhoneGapBuild verwendet.

Bei PhoneGapBuild handelt es sich um eine Online-Variante von PhoneGap. Diese wird von AdobeSystems kostenlos zur Verfügung gestellt. Der Vorteil dieser Variante ist, dass nicht für jedes Betriebssystem, für das die App entwickelt werden soll, eine eigene SDK installiert werden muss, da die Applikation direkt online kompiliert wird.

Den Code kann man entweder in Form einzelner HTML-, CSS-, und JS-Dateien verpackt

als ZIP-Datei hochladen, oder ein GitHub Projekt angeben in dem sich der Code befindet. Nach dem Hochladen des Codes wird die App online sofort kompiliert und die Installationsdateien (APK, XPA, etc.) werden als Download zur Verfügung gestellt.

Für die App-Entwicklung mit PhoneGapBuild benötigt man ausschließlich einen Editor und einen Webbrowser, da es sich ja eigentlich um Webentwicklung handelt. Die App besteht nur aus einer(oder mehreren) Webseite(n), welche mit CSS gestaltet wird. Um die Applikation interaktiv zu gestalten, kann man mit JavaScript-Skripts arbeiten und diese auch integrieren. Im Gegensatz zur Webentwicklung gibt es bei der App zusätzlich noch eine Datei mit dem Namen config.xml. In dieser Datei stehen alle Informationen, wie zum Beispiel die Versionsnummer oder der Name, zu der App und anhand dieser Datei können Berechtigungen für den Zugriff auf das Gerät vergeben werden.

3.1.5.2 iOS

3.1.5.3 Android

Android ist ein Betriebssystem für Smartphones, welches von Google entwickelt wird. Es basiert auf einem Linux-Kernel.

Bei Android handelt es sich um das am weitesten verbreitete mobile Betriebssystem, es hat einen Marktanteil von ca. 80%. Es wird als freie Software gehandelt und wird als Open-Source entwickelt.

Architektur

Android baut wie bereits erwähnt auf einem Linux-Kernel auf. Dieser stellt eine Schnittstelle zwischen der Hardware und der höher gelegenen Software dar.

In der nächsthöheren Systemschicht sind Android-Klassenbibliotheken, durch welche man die Funktionen des Kernels nutzen kann. Zusätzlich befindet sich auf dieser Ebene eine Dalvik-Virtual-Machine, dabei handelt es sich um eine virtuelle Maschine in der die Applikationen ausgeführt werden.

Diese virtuelle Maschine wurde von Google entwickelt und ähnelt in ihrer Funktionalität sehr der Java-VM. Die virtuellen Maschinen führen den Bytecode der Applikationen aus, aber die Dalvik Maschine arbeitet als Registermaschine, weshalb normaler Java-Bytecode auf Android nicht funktioniert.

Android startet für jede gestartete Applikation eine eigene virtuelle Maschine, dadurch kann keine App direkt auf das System zugreifen sondern nur über die virtuelle Maschine und des weiteren können sich die Apps nicht gegenseitig stören oder beeinflussen, dadurch dass sie in verschiedenen virtuellen Maschinen betrieben werden.

Store Im Google Play Store sind viele Apps für Android verfügbar. Dieser Store ist komplett kostenlos nutzbar, jedoch gibt es kostenpflichtige Applikationen. Um Apps aus dem Play-Store zu installieren muss man aber einen Google-Account besitzen welcher auch komplett kostenlos ist.

Bei Android kann man den Store relativ einfach umgehen, denn bei Android kann man

alle APK-Files(Android-Installationsdateien) einfach ausführen. Dazu muss man nur in den Einstellungen im Menü Anwendungen, den Punkt Unbekannte Quellen aktivieren. Damit erlaubt man das Installieren von Apps die nicht aus dem Play-Store heruntergeladen werden.

Ist das erledigt muss man nur noch das gewünschte APK-File auf dem Smartphone speichern (downloaden oder über USB auf dem Smartphone speichern) und dieses dann öffnen. Dann wird die Applikation automatisch installiert und ist danach wie jede andere Applikation auf dem Gerät installiert.

Um Apps in den Play-Store hochzuladen muss man als Entwickler registriert sein. Eine Registrierung als Entwickler kostet einmalig(Registrierungsgebühren) 25\$. Nach dieser Registrierung kann man kostenlos so viele Apps hochladen wie man will, bei kostenpflichtigen Apps verlangt Google jedoch ca 30% der Einnahmen.

3.1.5.4 Windows Phone

3.1.5.5 Sonstiges

3.2 Lösungswege

3.2.1 PHP

3.2.1.1 Gewählte Lösung

3.2.1.2 Alternative Lösung

3.2.2 MySQL

3.2.2.1 Gewählte Lösung

3.2.2.2 Alternative Lösung

3.2.3 Datenbankdesign

Bei Design der Datenbank stellten sich einige grundlegenden Fragen:

- Wie werden die Schulstunden gespeichert?
- Wie werden die Supplierungen gespeichert?

3.2.3.1 Gewählte Lösung

Die Gruppe entschied sich für folgende Lösung:

Es gibt Tabellen für:

- Abteilungen

- Klassen
- Fächer
- Lehrer
- Uhrzeiten
- Die Basis-Stunden
- Die Stunden
- fehlende Lehrer
- fehlende Klassen
- Die Supplierung

Die Basis-Stunde verknüpft die Klasse der Stunde mit der Uhrzeit. Die Stunde verknüpft die Basis-Stunde mit dem Lehrer, dem Fach und dem Raum.

Die Basis-Stunde wird benötigt, damit die Information, welche Klasse wann Unterricht hat, nicht mehrfach gespeichert werden muss.

Die Tabellen für die fehlenden Lehrer und die fehlenden Klassen haben Spalten für eine Start- und End-Zeit.

Die Supplierungs-Tabelle hat Felder für die Stunde, den Supplierlehrer, den neuen Klassenraum, die neue Start- und Endstunde, sowie Flags zum Ausblenden der Stunde auf dem angepassten Stundenplan.

Es ist möglich, mehrere Supplierungen für die gleiche Stunde einzutragen. Dieses Verhalten ist für manche Ausnahmefälle notwendig (siehe Beispiel).

The tex file displaying your dot graph could not be created. This usually means one of two things:

- dot2tex is not in your path
- write18 is not enabled

Abbildung 3.5: Supplierungs-Datenbank

Für das folgende Beispiel wird angenommen, dass eine Schulstunde 60 Minuten entspricht.

Beispiel: Klasse K hat am Montag um 8:00 eine Doppelstunde des Faches F mit Lehrer L in Raum R. Da der Lehrer zu spät kommen wird (ist bereits im vorhinein bekannt), soll nun die erste Stunde auf 16:00 in Raum P verschoben werden, die zweite Stunde soll aber bleiben.

Lösung: Die Doppel-Stunde wird 2 mal supplied. Einmal mit dem selben Lehrer im selben Raum mit Start-Stunde um 9:00 und End-Stunde um 10:00. Und das zweitemal mit dem selben Lehrer im Raum P mit Start-Stunde um 16:00 und End-Stunde um 17:00.

3.2.3.2 Alternative Lösungen

Eine alternative Lösung, die angedacht wurde, ist folgende:

Ansatt zwei Tabellen für die Basis-Stunde und die Stunde gibt es nur eine Stunden-Tabelle. Diese verknüpft die Klasse mit der Uhrzeit, der Länge der Stunde, einem kombinierten Lehrer-Feld, einem kombinierten Fächer-Feld, einem kombinierten Raum-Feld.

Die kombinierten Felden sollten den ersten Wert enthalten. Für jeden weiteren Wert wird der aktuelle Feld-Wert verodert mit dem nächsten Wert, welcher zuvor um $(x \cdot (\text{Anzahl der bisherigen Einträge}))$ Bits nach links verschoben wurde.

Hierbei wurde x noch nicht bestimmt, es müsste so gewählt werden, dass der Eintrag mit der höchsten ID nicht in den dahinterliegenden Eintrag überläuft (Für die Lehrer wäre 2^8 (=256) ideal, da es knapp unter 200 Lehrer an der HTL gibt).

Diese Lösungsmöglichkeit wurde allerdings verworfen, da es einiges an Aufwand ist, immer die richtigen Lehrer herauszulesen und da das Holen der Lehrer-Daten (Name, etc) nicht mehr im selben Query ausgeführt werden kann, als das Holen der Stundendaten, was bei mehr als 1 Zugriffspunkt zu Problemen mit der Daten-Konsistenz führen kann.

3.2.4 HTTPS

3.2.4.1 Gewählte Lösung

3.2.4.2 Alternative Lösungen

3.2.5 Mobile App

3.2.5.1 Gewählte Lösung

3.2.5.2 Alternative Lösung

3.2.6 Prinzip des Monitor-Systems

Als Thin-Client für die Monitore werden Raspberry Pis verwendet.

Die Kommunikation zum Server und die Art des Anzeigens der abgerufenen Informationen wird mithilfe eines Webbrowsers gelöst. Es wurde der Webbrowser Chromium ausgewählt,

da dieser die neuesten HTML5 Standards implementiert hat und auf den Raspberry Pis laufen kann.

3.2.6.1 Laden der Daten

Für das Laden der Daten vom Webserver wird Javascript in Verbindung mit AJAX verwendet.

Dies hat gegenüber dem Neuladen der Seiten den Vorteil, dass, im Falle dessen, dass der Webserver kurzzeitig nicht erreichbar ist, das System trotzdem ohne Probleme weiter laufen kann - sieht man davon ab, dass mit der Zeit, die angezeigten Daten veralten.

Der einzige Zeitpunkt, an dem der Server auf alle Fälle erreichbar sein muss, ist der Start der Raspberry Pis, da hier die Website vom Server geladen werden wird. Auch diese Bedingung hätte entfernt werden können, indem man die Seite lokal abgelegt hätte. Dies hat allerdings den Nachteil, dass die Seite über ein weiteres Script hätte immer auf dem aktuellen Stand gebracht werden.

3.2.7 Authentifizierung der Monitore

Es stellt sich die Frage, die die Monitore am Server registriert werden. Da alle Monitore über das Webinterface zentral administriert werden können, ist es äußerst kontraproduktiv, wenn eventuell Schüler - oder gar Außenstehende - die Datenbank-Tabelle für die registrierten Monitore fluten können.

Es wurde von den Betreuern vorgegeben, dass für sämtliche Verbindungen zum Server HTTPS verwendet werden sollte (siehe Lösungswege/HTTPS). Dies führt bei der Registrierung der Monitore zu Problemen (siehe folgendes).

3.2.7.1 Gewählte Lösung

Da zu anfangs nicht klar war, welches IP-Netz das W-LAN der Monitore bekommen sollte, konnte keine IP-Filterung verwendet werden.

Das Problem wurde gelöst, indem in der Datenbank für die Monitore eine Spalte für die IP-Adresse vorgesehen wurde. Die Idee bestand darin, dass jede IP-Adresse nur maximal einen Monitor registrieren darf (Vgl. Absatz 3.2.7.2.1).

Aufgrund dessen, dass für die verschlüsselte Verbindung die Pakete nach außen geNATet werden und wieder durch die Firewall zurück kommen, haben alle Monitore die öffentliche IP-Adresse der Schule.

Um dieses Problem zu umgehen wurde mit Betreuer Lassnig vereinbart, dass die Verbindung der Monitore unverschlüsselt sein darf, da keine sicherheitsrelevanten Daten übertragen werden, und da das W-LAN selbst schon verschlüsselt.

Als Ziel-Adresse wird also anstatt `https://sis.htlinn.ac.at` `http://sis.clients.htlinn.ac.at` eingetragen.

3.2.7.2 Alternative Lösungen

3.2.7.2.1 Registrierung Die Registrierung der Monitor hätte manuell erfolgen können. Hier würden mehrere Möglichkeiten in Frage kommen:

Registrierung nur vom Administrator-Interface aus

Dies hat den Nachteil, dass es zu frustrierenden Situationen kommen kann, wenn der tatsächliche Name des Monitors (eingestellt im Raspberry Pi) mit dem in der Datenbank nicht identisch ist.

Registrierung mittels Passwort

Ein Nachteil dieser Variante ist, dass das Passwort direkt an den Raspberry Pi eingegeben werden muss. Was den administrativen Aufwand vergrößert. Weiters muss dieses Passwort für Konfiguration mehr oder weniger öffentlich zugänglich sein muss, was den Zusatzaufwand nichtig macht.

Registrierung mittels Cookie-Hash

Diese Variante hat ähnliche Nachteile, wie die Registrierung mittels Passwort.

3.2.7.2.2 Verschlüsselung Als Alternative zur fehlenden Verschlüsselung hätte ein, von der HTL ausgestelltes SSL-Zertifikat für die Wild-Card-Domain *.clients.htlinn.ac.at ausgestellt werden können, den Raspberry Pi hätte man dann müssen das HTL-Root-Zertifikat importieren müssen. Dies ist allerdings mit recht viel administrativen Aufwand verbunden, daher wurde diese Idee verworfen.

Als eine weitere Variante hätte man den Raspberry Pi den Server, auf dem die Website abgelegt wurde, als SSL-Ausnahme eintragen können.

3.2.8 Benutzermanagement

In der Schule sind im Wesentlichen 4 Berechtigungsstufen vorhanden.

- Schüler
- Lehrer
- Abteilungs-Vorstände
- Administratoren

Diese Gliederung wird 1:1 in SIS übertragen, allerdings wird zusätzlich eine Gruppe für die „Newsbeauftragten“, welche gegenüber den Schülern/Lehrern zusätzlich die Berechtigung haben, den Abteilungsvorständen News vorzuschlagen, vorgesehen.

Da die Benutzer-Authentifizierung über den LDAP-Server läuft, liegt der Gedanke nahe, dass auch die Autorisierung über LDAP laufen könnte.

Da alle Lehrer automatisch im eDirectory-Container ou=LEHRER,o=HTLinn und die Schüler automatisch im eDirectory-Container ou=STUDENTS,o=HTLinn liegen, lässt

sich anhand des DN des Benutzers ausgelesen werden, ob es sich um einen Lehrer oder um einen Schüler handelt.

3.2.8.1 eDirectory-Gruppen

Für alle weiteren logischen Gruppen werden vom eDirectory-Administrator auch dementsprechende eDirectory-Gruppen erstellt:

- cn=SIS-SuperUser,ou=SIS,o=HTL1
- cn=SIS-News,ou=SIS,o=HTL1
- cn=SIS-Admin-E,ou=SIS,o=HTL1
- cn=SIS-Admin-M,ou=SIS,o=HTL1
- cn=SIS-Admin-N,ou=SIS,o=HTL1
- cn=SIS-Admin-W,ou=SIS,o=HTL1

Es ist prinzipiell möglich, dass ein Benutzer Mitglied mehrerer eDirectory-Gruppen ist.

3.2.8.2 Verwendung

Meldet sich ein Benutzer an, so werden seine Berechtigungen (sprich Gruppen-Zugehörigkeiten) am LDAP-Server abgefragt und in der PHP-Session gespeichert. Auch, ob es sich um einen Lehrer handelt, wird in der Session gespeichert. Dies hat den Vorteil, dass nur einmal die Berechtigungen abgefragt werden müssen.

Jedes PHP-Script kann die Berechtigungen über die PHP-Session abfragen und mehr oder weniger autonom entscheiden, welche Berechtigungen der Benutzer erhält.

3.3 Grobentwurf

3.3.1 Dateibaum

Der Aufbau der Ordnerstruktur für das Projekt wird folgendermaßen festgelegt.

/backend/

Hier befindet sich das Menü für die Eingaben.

Dieses Menü ist für die Benutzergruppen „SIS-Admin-E“, „SIS-Admin-N“, „SIS-Admin-M“, „SIS-Admin-W“, „SIS-News“ und „SIS-SuperUser“ verfügbar.

./absentees/

Hier befindet sich das Menü für das Eintragen der Fehlenden.

Dieses Menü ist für die Benutzergruppen „SIS-Admin-E“, „SIS-Admin-N“, „SIS-Admin-M“, „SIS-Admin-W“ und „SIS-SuperUser“ verfügbar.

./classes/

Hier können fehlende Klassen eingetragen werden.

Dieses Formular ist für die Benutzergruppen „SIS-Admin-E“, „SIS-Admin-N“, „SIS-Admin-M“, „SIS-Admin-W“ und „SIS-SuperUser“ verfügbar.

./teachers/

Hier können fehlende Lehrer eingetragen werden.

Dieses Formular ist für die Benutzergruppen „SIS-Admin-E“, „SIS-Admin-N“, „SIS-Admin-M“, „SIS-Admin-W“ und „SIS-SuperUser“ verfügbar.

./administration/

Hier befindet sich das Administratoren-Menü.

Dieses Menü sowie alle Unterpunkte sind nur für die Benutzergruppe „SIS-SuperUser“ verfügbar.

./classes/

In diesem Formular können die Klassen modifiziert werden.

./hours/

Hier können die zeitlichen Unterrichtsstunden verändert werden.

./lessons/

In diesem Formular können die Stundenpläne verändert werden.

./rooms/

Hier können die Räume eingetragen werden.

./sections/

Hier können die Abteilungen modifiziert werden.

./subjects/

Fächer können hier hinzugefügt werden.

./teachers/

In diesem Formular können die Lehrer modifiziert werden.

./monitors/

Alle Einstellungen für die Monitore sind hier zu finden.

Dieses Formular ist für die Benutzergruppen „SIS-Admin-E“, „SIS-Admin-N“, „SIS-Admin-M“, „SIS-Admin-W“ und „SIS-SuperUser“ verfügbar.

./news/

In diesem Punkt können News eingetragen werden.

Dieses Menü ist für die Benutzergruppen „SIS-Admin-E“, „SIS-Admin-N“, „SIS-Admin-M“, „SIS-Admin-W“, „SIS-News“ und „SIS-SuperUser“ verfügbar.

./substitudes/

Hier befindet sich das Menü für die Auswahl der Abteilung bei den Lieferungen.

Dieses Menü ist nur für die Benutzergruppe „SIS-SuperUser“ verfügbar.

./form/

Hier können die Lieferungen eingetragen werden.

Dieses Formular ist nur für die Benutzergruppen „SIS-Admin-E“, „SIS-Admin-N“, „SIS-Admin-M“, „SIS-Admin-W“ und „SIS-SuperUser“ verfügbar.

/cookies/

Hier müssen vor dem Betreten der Seite die Cookies akzeptiert werden (siehe

/data/ ./fonts/

Alle Schriftart-Dateien liegen hier.

./images/

Hier befinden sich sämtliche Bilder.

Dateien, welche sich logisch gruppieren lassen, besitzen eigene Ordner.

./scripts/

Alle Javascripts, welche in Dateien extrahiert wurden, sind hier zu finden.

./styles/

Hier sind sämtliche Stylesheets.

/impressum/

Wie der Name schon sagt, ist hier das Impressum zu finden.

/login/

Hier befindet sich der Login.

/logout/

Und hier ist der Logout.

/logs/

Hier sind spezielle Log-Dateien zu finden.

/mobile/

Die Mobil-Seite befindet sich hier.

./api/

3.3.2 Datenbankdesign

Das Konzept des Datenbankdesigns wird bereits unter Punkt „Lösungswege“ behandelt. Hinzu kommt noch das das Monitorsystem.

3.3.2.1 Monitorsystem

Damit die Monitore einen Supplierplan anzeigen können, müssen jeder Monitor einer Abteilung zugeordnet sein.

Auch müssen die Monitore damit sie die Raumstundenpläne anzeigen können, mit einem Raum verknüpft sein.

Zusätzlich dazu sollen die Monitore wissen, was sie anzeigen sollen (Stundenplan, Supplierplan, Bilder, etc). Hierzu wird eine Tabelle für den Typ des Monitors erstellt und diese verknüpft.

Für die Erweiterung des Monitorsystem um das Display-Steuer-System muss eine weitere Tabelle für die Display-Typ (permanent ein, permanent aus, automatisch) vorgesehen und verknüpft werden (für Details, siehe Anhang C - Begleitendes FTKL-Projekt (Seite 54)).

Es ergibt sich folgendes Datenbank-Layout für die Monitore:

The tex file displaying your dot graph could not be created. This usually means one of two things:

- dot2tex is not in your path
- write18 is not enabled

Abbildung 3.6: Lieferungs-Datenbank

3.3.3 Monitorsystem

3.3.3.1 Grundsätzlicher Seiten-Aufbau

Der Aufbau der, auf den Monitoren angezeigten, Seite wird folgendermaßen festgelegt. Rechts unten wird Platz für eine Uhrzeit- und Datumsanzeige vorgesehen. Links daneben werden später die raumspezifischen Beschriftungen angezeigt.

Rechts oben soll der aktive Modus des Monitors am Hintergrund angezeigt werden. Horizontal und vertikal zentriert wird im Hintergrund das SIS-Logo platziert.

Der Platz oberhalb der Uhr steht Inhalten zur Verfügung, auch wenn dieser das SIS-Logo oder den Text rechts oben verdecken sollte.

3.3.3.2 Initialisierung

Wird die Seite geöffnet, so wird die Seite grundsätzliche aufgebaut.

Nun werden zwei Timer gestartet, von denen einer die Uhr aktualisiert (jede Sekunde) und der andere den Inhalt aktualisiert (10 Sekunden).

3.3.3.3 Verringerung des Traffics

Damit der Durchschnittliche Traffic möglichst gering gehalten werden kann, wird beim Anfragen des Inhalts am Server ein Hash generiert, welcher eindeutig den aktuellen Inhalt identifiziert. Dieser Hash wird mitgesendet und am Client gespeichert. Nun wird mit jeder weiteren Anfrage der Hash des Clients mitgesendet, sollten der Client-Hash und der neu generierte Hash am Server übereinstimmen, so wird ein Flag gesendet, dass sich der Inhalt nicht geändert habe. Der eigentliche Inhalt aber wird nicht gesendet.

Da sich die Inhalte relativ zum Aktualisierungszyklus des Clients sehr selten ändern, kann der Traffic so um weit über 50 % reduziert werden.

3.3.3.4 Multi-Modi

Für den Monitor-Modus „Supplierplan & News“ wird am Server ein Sonderfall vorgesehen. Es wird eine zusätzliche Variable mit gesendet, diese identifiziert den „Submode“, also quasi ob Supplierplan oder die News gesendet werden sollen. Wann sich der Submode ändert wird durch eine weitere Variable vorgegeben. Diese enthält den Timestamp der nächsten Änderung. Diese beiden Variablen werden vom Server generiert und vom Client zwar gespeichert, aber nicht mehr verändert, bis der Server neue Werte mitgibt.

3.4 Feinentwurf

3.5 Implementierung

3.5.1 Sourcecode

Der vollständige Sourcecode (Stand:) // TODO ist auf der angehängten CD zu finden.

3.5.2 Test- Und Messergebnisse

A || Betriebsanleitung Benutzer

A.1 Einleitung

A.2 Stundenplan

A.3 Supplierplan

A.4 News

A.5 App

B || Betriebsanleitung Administrator

B.1 Einleitung

B.2 News

Um die News zu bearbeiten, klicken Sie im „Data-Input-Menü“ auf den Punkt „News hinzufügen“

Nach dem Öffnen der Seite werden die eingetragenen News angezeigt.

B.2.1 News bearbeiten/hinzufügen

Titel Titel der angezeigt werden soll.

Text Text der angezeigt werden soll.

Abt. Abteilung für welche die News angezeigt werden soll. Wenn dieses Feld leer ist, wird die News in allen Abteilungen angezeigt.

Anzeigebeginn-Datum Datum ab welchem die News angezeigt werden soll.

Anzeigend-Datum Datum bis zu welchem die News angezeigt werden soll.

Anzeigen Zeigt an ob die News angezeigt wird. Wenn ein Newsbeauftragter eine News hinzufügt, ist dieser Punkt nicht gewählt und muss vom Abteilungsvorstand ausgewählt werden, um die News freizugeben.

Ersteller In dieses Feld kann nichts eingegeben werden. Es dient dazu, dass der Abteilungsleiter weiß, wer die News erstellt hat.

Nur Website Wenn dieser Punkt ausgewählt ist, wird die News nur auf der Website oder in der App angezeigt und nicht auf den Monitoren.

B.2.2 News löschen

B.3 Monitore

Um in die Einstellungen für die Monitore zu gelangen, klicken Sie im „Data-Input-Menü“ auf den Punkt „Monitore verwalten“ (siehe Abbildung B.1).



Abbildung B.1: Data-Input-Menü

Nach dem Öffnen der Seite werden oben im Fenster die aktiven Monitore aufgelistet. Darunter sind die Einstellungen zu finden.

| Name | Text | Raum | Abteilung | Modus | Datei | erstellt von | am | Monitor-Steuerung |
|---------|-------------------------|-------|-----------|--------------|-------------------------|-----------------|--------------|-----------------------------------|
| ■ I202A | | I202A | N | News | def.mp4 | 10.10.54.122 | 03. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I202 | | I202 | N | News | def.mp4 | 10.10.120.13 | 28. 01. 2014 | ein zwischen 7:00:00 und 17:00:00 |
| ■ I310 | Klassenzimmer der 5Yoda | I310 | N | News | 1391085381-DSC00064.JPG | 10.10.120.20 | 30. 01. 2014 | ein zwischen 7:00:00 und 17:00:00 |
| ■ I210 | | I210 | N | News | def.mp4 | 10.10.54.122 | 03. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I207 | | I207 | N | News | def.mp4 | 10.10.120.28 | 28. 01. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I201 | | I201 | N | News | def.mp4 | 10.10.120.5 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I151b | SIGE-Plan beachten | I151B | N | News | def.mp4 | 10.10.120.11 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I213 | | I207 | N | News | def.mp4 | 10.10.120.19 | 28. 01. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I151a | SIGE-Plan beachten | I151A | N | News | def.mp4 | 10.10.120.27 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I251 | | I251 | N | News | def.mp4 | 10.10.120.24 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I208 | | I208 | N | News | def.mp4 | 10.10.120.21 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I212 | | I212 | N | News | def.mp4 | 10.10.120.6 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I211 | | I211 | N | News | def.mp4 | 10.10.120.23 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I209 | | I209 | N | News | def.mp4 | 10.10.120.26 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ I116 | | I116 | N | News | def.mp4 | 10.10.120.22 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ TV-I2 | | AK02 | N | fallback | def.mp4 | 10.10.120.14 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ TV-I1 | | AK02 | N | fallback | def.mp4 | 10.10.120.4 | 04. 02. 2014 | ein zwischen 7:00:00 und 18:00:00 |
| ■ test | | I402 | E | Supplierplan | def.mp4 | 194.166.179.171 | 04. 02. 2014 | immer ein |

Text: (leer für unverändert)

Raum: (leer für unverändert)

Abteilung: (leer für unverändert)

Modus: (leer für unverändert)

Datei: (leer für unverändert)

Display-Modus: (leer für unverändert)

Einschalt-Zeit (leer für unverändert)

Ausschalt-Zeit (leer für unverändert)

Löschen? ☐

Abbildung B.2: Monitor-Einstellungen

Die Monitore, deren Einstellungen verändert werden sollen, müssen mit einem Klick auf die Checkbox links neben der Anzeige ausgewählt werden. Es können mehrere ausgewählt werden. Mit dem Button „Auswahl invertieren“ werden alle gesetzten Checkbox rückgesetzt, und umgekehrt.

B.3.1 Hinzufügen

Für folgendes wird vorausgesetzt, dass das von uns zur Verfügung gestellte Image verwendet wird. Dieses kann unter folgender Adresse heruntergeladen werden:

<https://sis.htlinn.ac.at/sis.img>

Das Image ist 4 GB groß, wir empfehlen deshalb, das Image ohne Proxy-Einstellung mit der Adresse <http://sis.clients.htlinn.ac.at/sis.img> im Schulnetz zu laden. Das Image sollte auf eine SD-Karte mit mindestens 3965190144 Bytes Speicherplatz gespielt werden (Getestet wurde mit „SanDisk Extreme 4GB“).

Der vordefinierte Benutzer hat die Kennung „pi“ und das Passwort „sis“.

Ein neuer Monitor kann nicht über das Web-Interface hinzugefügt werden, die Monitore registrieren sich selbstständig am Server. Nach der Registrierung stehen sie zur Verwaltung zur Verfügung.

Damit sich die Monitore registrieren können, müssen sie ihre eigene Identität kennen. Diese wird in der Datei `/etc/sis.conf` definiert.

```
1 name=test-monitor
```

Code B.1: Beispiel `/etc/sis.conf`

Die Form des Namenseintrags ist im Beispiel Code B.1 ersichtlich.

Wenn der Name des Monitors geändert wird, sollte aus Gründen der Übersichtlichkeit und Nachvollziehbarkeit darauf geachtet werden, dass der Hostname des Raspberry Pis mit dem Namen übereinstimmt. Dazu wird die Datei `/etc/hostname` bearbeitet. Damit der Host nicht neu gestartet werden muss, kann mit dem Befehl „hostname“ der Hostname zusätzlich auch direkt geändert werden.

Beispiel: `hostname I386`

Achtung: Da es zu Problemen mit manchen Programmen (wie zum Beispiel `sudo`) kommen kann, sollte in der Datei `/etc/hosts` ein Eintrag für den Hostnamen angelegt werden, welcher auf die lokale Adresse zeigt (üblicherweise die Loopback-Adresse 127.0.1.1).

Achtung: Die Software lässt es zu, dass mehrere Monitore den selben Namen tragen. Dies sollte allerdings vermieden werden.

B.3.2 Entfernen

Um einen Monitor aus der Liste der aktiven Monitore zu entfernen, wählen Sie den jeweiligen Monitor über die Checkbox aus, setzen Sie die „Löschen“-Checkbox und klicken Sie auf „Änderungen anwenden“.

Achtung: Sollte der Monitor nach dem Entfernen noch eingeschalten sein, so wird sich dieser, sofern die Standard-Konfiguration eingestellt ist, wieder neu registrieren.

B.3.3 Text verändern

Der Text, der am linken unterem Eck des Monitors kann über das Eingabefeld „Text“ geändert werden. Wird das Feld leer gelassen, so wird der alte Wert beibehalten.

Achtung: Leerzeichen werden nicht als leer erkannt.

B.3.4 Raum verändern

Für die Darstellung des Stundenplanes des jeweiligen Raumes wird jedem Monitor ein Raum zugeordnet. Diese Einstellung kann über das Eingabefeld „Raum“ angepasst werden. Durch Doppelklick auf das Eingabefeld öffnet sich - sofern der verwendete Browser diese Funktion unterstützt - ein Menü, in dem durch Eingeben gesucht werden kann. Wird das Feld leer gelassen, so wird der alte Wert beibehalten.

B.3.5 Abteilung verändern

Die News können Abteilungen zugeordnet werden. Ebenso ist der Supplierplan für jede Abteilung anders. Als Folge daraus gibt es eine Einstellungsmöglichkeit für die Abteilung. Durch Doppelklick auf das Eingabefeld öffnet sich - sofern der verwendete Browser diese Funktion unterstützt - ein Menü, in dem durch Eingeben gesucht werden kann. Wird das Feld leer gelassen, so wird der alte Wert beibehalten.

B.3.6 Modus verändern

Um die ausgewählten Monitore auf Supplierplan, Stundenplan, News etc. umzuschalten, wird das Eingabefeld „Modus“ verwendet. In diesem Feld sind folgende Werte zulässig:

- News
- Stundenplan
- Supplierplan
- Supplierplan & News
- Bild

- (Video)

Durch Doppelklick auf das Eingabefeld öffnet sich - sofern der verwendete Browser diese Funktion unterstützt - ein Menü mit den möglichen Werten. Wird das Feld leer gelassen, so wird der alte Wert beibehalten.

Der Modus „Video“ ist zwar implementiert, allerdings ist der Raspberry Pi nicht in der Lage, die Videos wiederzugeben.

B.3.7 Datei verändern

Für die Modi „Bild“ und „Video“ muss eine Datei hochgeladen werden, welche dann angezeigt wird.

Unterstützte Dateitypen sind:

- JPG, JPEG
- PNG
- GIF
- MP4, MPEG

Die maximale Dateigröße ist limitiert durch:

- das Upload-Script auf 800 MB
- den Upload-HTML-Tag auf 800 MB
- die PHP-Konfiguration am Server (Stand: 2014-02-28) auf 128 MB

Wird das Feld leer gelassen, so wird der alte Wert beibehalten.

B.3.8 Display-Modus verändern

Die Raspberry Pis verfügen über einen Mechanismus, um den angeschlossenen Monitor ein- bzw auszuschalten.

Es sind folgende Modi möglich:

- permanent Ein
- permanent Aus
- automatisch

Im automatischen Modus werden die Einstellungen von B.3.8.1 und B.3.8.2 verwendet.

Wird das Feld leer gelassen, so wird der alte Wert beibehalten.

B.3.8.1 Einschalt-Zeit

Bestimmt die Zeit, an dem sich der angeschlossene Monitor einschalten soll.

Beispiele:

| | |
|----------|---------------------|
| 7 | 7 Uhr früh |
| 6:35 | 6:35 früh |
| 17:05:11 | 5:05:11 nachmittags |

Wird das Feld leer gelassen, so wird der alte Wert beibehalten.

B.3.8.2 Ausschalt-Zeit

Analog zu B.3.8.1.

B.4 Fehlende

Im Punkt Fehlende im Menü "Data-Input" können fehlende Lehrer und fehlende Klassen eingegeben werden. Klickt man nun im Menü "Data-Input" (siehe Abbildung B.3) auf Fehlende, dann gelangt man in ein weiteres Menü. (siehe Abbildung B.4) In welchem man zwischen Klassen und Lehrer wählen kann. Wie der Name schon sagt, bei fehlende Lehrer werden die fehlenden Lehrer eingetragen und bei fehlenden Klassen werden die fehlenden Klassen eingetragen.



Abbildung B.3: Data-Input-Menü

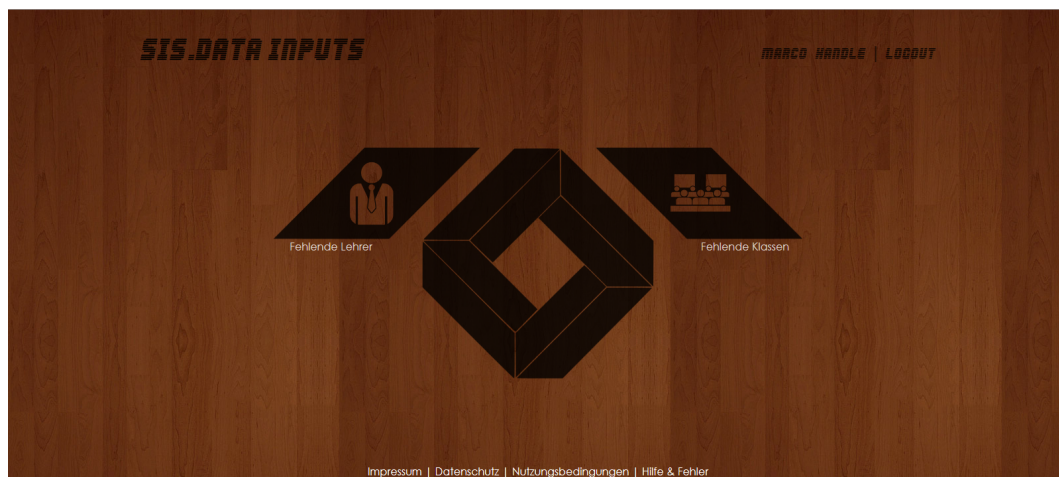


Abbildung B.4: Fehlende Auswahl

B.4.1 Fehlende Lehrer

Hat man nun den Punkt fehlende Lehrer Ausgewählt, so gelangt man zur Eingabemaske, die wie in Abbildung B.5 aufgebaut ist.

B.4.1.0.1 Hinzufügen Unter der Überschrift ist eine Auswahlmöglichkeit für das Datum, dieses Datum spiegelt den ersten Tag des Fehlens eines Lehrers wieder. Will man nun einen Lehrer als fehlend eintragen, so muss man im ersten Schritt zum richtigen Start-Datum gehen. Dies kann entweder durch drücken der Pfeiltaste geschehen oder durch Eingabe des Datums in dem Textfeld zwischen den Pfeilen. Wird diese Möglichkeit genutzt muss darauf geachtet werden, dass das Datum im Format yyyy-mm-dd eingegeben wird. Mit Enter muss diese Eingabe bestätigt werden.

Hat man nun das gewünschte Datum erreicht, so kann mit der Eingabe begonnen werden. Im ersten Feld, das mit Lehrer Beschriftet ist, muss das Lehrer-Kürzel eingetragen werden. Wird ein Buchstabe eingegeben so erscheint eine Liste von Lehrern die diesen Buchstaben enthalten. Nachdem der Lehrer eingegeben wurde, muss die Start-Stunde eingegeben werden. Das Feld Starttag kann nicht verändert werden, dies entspricht immer dem oben eingestellten Datum. Als Start-Stunde sind die Werte 1-16 erlaubt, es sei denn es wurden in der Eingabe der Stunden etwas anderes definiert. (Eingabe Stunde siehe Punkt ??? TODO) Für den Endtag ist Standardmäßig auch das Start-Datum eingetragen, dieses kann jedoch in das gewünschte Datum geändert werden. Voraussetzung dafür ist, dass es größer oder gleich dem Startdatum entspricht und es muss im Format yyyy-mm-dd eingegeben werden. Außerdem muss beachtet werden, dass dieses Datum kein Wochenende sein darf. Also es muss ein Datum eingegeben werden welches einem Wochentag zwischen Montag und Freitag entspricht. Im letzten Feld kann ein Grund für das Fehlen eingetragen werden. Ist alles wie gewünscht ausgefüllt (Bis auf den Grund müssen alle Felder ausgefüllt werden) so kann auf Übernehmen geklickt werden.

Wird ein Lehrer oder eine Stunde nicht gefunden, so wird ein Pop-Up angezeigt, auf dem der Name der falschen Zeit steht, der beispielsweise wie in Abbildung B.6 zu sehen ausse-

hen kann. Dieser muss mit OK bestätigt werden. Anschließend muss die Eingabe erneut erfolgen.

B.4.1.0.2 Löschen Ist ein Eintrag fehlerhaft oder muss er entfernt werden, so muss ein Hacken bei Löschen, in der Zeile des zu löschenden Eintrages, gesetzt werden und anschließend mit übernehmen bestätigen. (siehe Abbildung B.7) Nun ist dieser Eintrag **unwiderruflich** gelöscht.

Abbildung B.5: Fehlende Lehrer

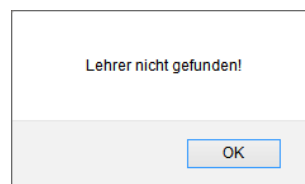


Abbildung B.6: Falsche Eingabe

Abbildung B.7: Fehlenden Lehrer löschen

B.4.2 Fehlende Schüler

Die Eingabe der Fehlenden Schüler läuft exakt nach dem gleichen Verfahren und Regeln ab wie die Eingabe der Fehlenden Lehrer. Deshalb werde ich hier nichts mehr zur Eingabe der fehlenden Klassen schreiben.

B.5 Supplierungen

B.6 Administrative Eingaben

C || Begleitendes FTKL-Projekt

C.1 Aufgabenstellung

Es sollte eine Möglichkeit geschaffen werden, die Monitore, welche an vielen Stellen in der Schule positioniert sind, zentral stromfrei zu schalten.

Die Umsetzung soll möglichst zeit- und geldsparend sein.

Das Benutzer-Interface für diese Funktion soll dabei in die Diplomarbeit (SIS) der Gruppe inkludiert werden.

C.2 Hardware

C.2.1 Überlegungen

Für die Umsetzung wurden handelsübliche Funksteckdosen verwendet, da diese relativ günstig in den meisten Geschäften zu haben sind. Diese erfüllen jedoch Standardmäßig einen anderen Zweck, diese werden über eine Fernbedienung ein bzw. aus geschaltet. Da es uns nicht möglich ist, Steckdosen mit dem Raspberry über Funk zu steuern, musste eine andere Möglichkeit her.

Uns kam die Idee, dass in diesen Steckdosen eigentlich nur ein Relais vorhanden sein muss, welches die 230V schaltet. Diese Tatsache machten wir uns zu nutze und suchten auf der Platine nach diesem Schalterpunkt und entwickelten eine passende Schaltung, welche diese Relais schaltet.

Um widerrechtliches steuern der Steckdosen durch gefinkelte Schüler zu verhindern und da das Funkmodul nur auf die Platine aufgelötet war, bauten wir diese Platine aus. Somit ist es nicht mehr möglich die Steckdosen mit der passenden Fernbedienung zu steuern.

Die Schaltung der Steckdose ist in keinsten Weise dokumentiert, auch die auf der Schaltung vorhandenen IC's sind kaum bis gar nicht dokumentiert, weshalb es uns nicht möglich war die Funktionsweise der einzelnen Teile fest zu stellen. Durch Analysen der Leiterbahnen konnten wir uns eine grobe Übersicht über die Schaltung verschaffen. Dieser Überblick reichte jedoch nicht aus, um den Schalterpunkt des Relais zu finden. Der eigentliche Schalterpin am Relais konnte identifiziert werden, jedoch schaltet dieses Relais erst bei 48V, welche nicht angelegt werden konnte, deshalb musste eine andere Möglichkeit gefunden werden. Die Art des Suchens war nicht sehr professionell, da wir versuchten, durch Anlegen einer

Steuerspannung an bestimmte Pins der ICs, den Pin zu finden, durch welchen das Relais geschaltet wird. Nach einigen Versuchen konnte der Punkt gefunden werden, dieser war doch nicht ideal, da dessen Innenwiderstand sehr klein war, was sich dahingehend zeigte, dass der Strom, der benötigt wurde, um die Steuerspannung zu halten, sehr hoch war. Wir fanden allerdings einen besseren Punkt, welcher wesentlich weniger Strom zog. Durch das Anlegen von 5V schaltete das Relais, dies entspricht der Versorgungsspannung des Funkmoduls. Es liegt nahe, diese zu verwenden.

Für eine leichtere Montage wurden die Vorhandenen Kontakte des Funkmoduls verwendet. Über die Kontakte wurde die Platine versorgt, diese Funktionsweise verwendeten auch wir, weshalb wir von dem gefundenen Punkt eine Drahtbrücke zum Kontakt machten. Die 5V waren sowieso schon auf einem der Kontakte vorhanden. Um die ursprüngliche Funktion des Kontaktes ab zu schalten trennten wir durch Entfernung eines Widerstandes die Leiterbahn auf. Nun musste nur mehr die Platine in die Kontaktlöcher gesteckt werden. Die nötigen Signale sind bei den Kontakten angelegt und werden über die Verbindung auf der Platine bereitgestellt.

Nun musste noch eine Lösung gefunden werden, wie mit dem Raspberry diese 2 Punkte verbunden werden konnten. Dies wurde mit einer Transistorschaltung realisiert. Wenn der Raspberry an einem der GPIO-Pins digital 1 (analog 3,3V DC) anlegt, so schaltet der Transistor und verbindet somit die zwei Punkte.

Dies war jedoch nicht genug, da noch eine Potentialtrennung zwischen Raspberry und Steckdose notwendig war, wie ein Test ohne Potentialtrennung zeigt. (Siehe Punkt???). Diese lösten wir mit einem Optokoppler, welcher den Raspberry von der Steckdose trennen soll. Dies erfolgte mit der Schaltung ordnungsgemäß. Jedoch packten wir die Potentialtrennung auf den Header, welcher auf den Raspberry aufgesteckt wird, dies stellte sich als fataler Fehler heraus. (siehe Punkt ????)

Der finale Aufbau war dann wie folgt:

Auf die GPIO-Pins (siehe Abbildung C.1) wird eine Platine(Header) gesteckt, welche die Transistorschaltung beinhaltet. Die Transistorschaltung schaltet die 5V Versorgung über einen Vorwiderstand den Optokoppler, welcher auf einer Platine in der Steckdose liegt. Die Verbindung wurde mit einem 2 poligen Draht, welcher auf beiden Seiten eine 3,5mm Mono Stecker montiert hat. Auf der Platine in der Steckdose ist nur der Optokoppler, der das Ein- bzw. Ausschalten der Steckdose vornimmt.

Für die Kommunikation mit dem Raspberry wird der GPIO Pin 4 (Header Pin 7) verwendet. Die 5V Spannung und die Masse für das Schalten des Optokopplers wird der Pin 2(5V) und der Pin 6(0V) des Rasperrys verwendet. Wichtig: Hier ist anzumerken, dass die Rasperrys Revision 1 und Revision 2 verschiedene Anordnung haben. Unsere Schaltung ist für Revision 2 entwickelt, es ist also nicht zu empfehlen, die Platine auf einem Revision 1 Raspberry zu verwenden, da er Schaden nehmen könnte.

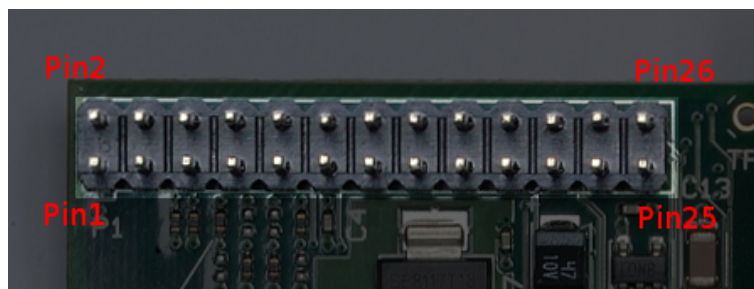


Abbildung C.1: GPIO-Pins

Quelle: http://elinux.org/Rpi_Low-level_peripherals

Raspberry gemacht werden, damit keine hohen Spannungen auf den Raspberry übertragen werden und dadurch ein hoher Strom resultiert. Dies hat die Folge, dass der Raspberry zerstört wird. Die Lösung dieses Problem ist die Trennung mit Hilfe eines Optokopplers. Die Ausführung siehe Punkt ???

Spannung auf berührbaren Teilen Dieses Problem hat eigentlich wieder mit der Potentialtrennung zu tun. Wir lösten die Potentialtrennung auf dem Raspberry Header, das heißt das nicht Potential getrennte Signal ist von der Steckdose über das Verbindungskabel zum Raspberry geführt worden. Dies ist jedoch nicht so dramatisch. Jedoch erzeugt die Steckdose ihr Potential für die Gleichspannungsversorgung immer fix. Das heißt die Steckdose nimmt immer, egal wie die Steckdose in der Steckdose steckt, zum Beispiel den rechten Pin der Steckdose.

Zu besseren Erklärung wird dies am folgendem Beispiel geschildert:

- Nehmen wir an die Steckdose steckt mit der Position A in der Steckdose, somit hat die Schaltung als Masse den Neutralleiter, was dann, bezogen auf die Erdung, ergibt, dass die Masse gleich Erdung und die 5V gleich $5V + \text{Erdungspotential}$ ist. Somit ist alles in Ordnung.
- Nun nehmen wir an, dass die Steckdose mit Position B in der Steckdose steckt, was soviel wie eine Drehung um 180° bedeutet. Nun erzeugt die Schaltung wieder auf dem selben Prinzip die Versorgungsspannung. Nur mit dem Unterschied, dass wo zuvor der Neutralleiter war nun die Phase ist und umgekehrt. Dies hat zur Folge, dass die Schaltung die Versorgungsspannung auf Basis der Phase macht, also Masse gleich Phase gleich 230V und 5V gleich $\text{Phase} + 5V \text{ DC}$.

Somit lässt sich nun sagen, dass wir in unserem Fall, wenn die Steckdose falsch in der Steckdose gesteckt ist", $5V + 230V$ über das Verbindungskabel auf den Raspberry Header übertragen haben und somit bei Berührung des Headers einen 230V Stromschlag bekommen haben.

Die Lösung dafür war, dass wir die Potentialtrennung in die Steckdose verlegten und somit nur ein Potentialfreies Signal über die Verbindungsleitung übertragen haben.

C.3 Software

C.3.1 Überlegungen

Die GPIO-Pins lassen sich über virtuelle Dateien des Linux-Kernels ansteuern. Somit liegt hier kein Problem vor.

Das größere Problem ist das Verteilen der Befehle auf alle Raspberry Pis. Da das Benutzerinterface sowieso in SIS integriert werden soll, liegt der Gedanke nahe, auch die Verteilung der Befehle über SIS zu managen.

Das Team entschied sich dafür, in einem vorbestimmten Zeitintervall am Server anzufragen, welchen Zustand der Monitor einnehmen soll (eingeschaltet oder ausgeschaltet). Sollte der Server nicht erreichbar sein, so soll der aktuelle Zustand beibehalten werden.

Zwar gehört die Darstellungsfrage und Konfigurationsfrage von der Aufgabenstellung her nicht zu diesem Projekt. Allerdings passt es thematisch hierher und soll ebenfalls hier behandelt werden.

C.3.2 Lösung

C.3.2.1 Überlegungen

Die Software zum regelmäßigen Anfragen am Server muss bereits beim Systemstart mitgestartet werden. Es bietet sich an, die Scripts über die Datei `/etc/rc.local` zu starten, da diese Datei relativ unkompliziert verwendet werden kann (??, Seite ??).

C.3.2.2 Darstellung

Damit kein Tool für den automatischen Login installiert werden muss, wird der Default-Runlevel auf gesetzt (Dies kann durch Modifikation des Schlüssels „initdefault“ in der Datei `/etc/inittab` erreicht werden. Siehe Code C.4, Seite 61.) und der X-Server manuell gestartet.

Das Starten des X-Servers wird durch das Script `/usr/bin/sis-x` initialisiert.

```
1  #!/bin/bash
2
3  if test "`whoami`" = "root"; then
4      echo "we_are_root-_restarting"
5      exec /bin/su - pi -c "$0"
6      exit 1;
7  fi
8
9  echo "changing_to_/home/pi"
10
11 cd /home/pi
12
13 echo "executing_x11"
14
15 startx
```

Code C.1: `/usr/bin/sis-x`

Wie in Code C.1 ersichtlich ist, wird überprüft, ob das Script als root gestartet wurde. Wenn dies der Fall sein sollte, dann startet das Script sich selbst mit Benutzerkennung „pi“ neu, wechselt in das Heimverzeichnis des Benutzers (statisch eingetragen als `/home/pi`) und initialisiert mit dem Befehl „startx“ einen neuen X-Server.

X11 führt beim Starten die Datei `.xinitrc` im Heimverzeichnis des Benutzers (also `/home/pi/.xinitrc`) aus. Siehe dazu Code C.5, Seite 62.

In dieser passiert folgendes:

- Es wird verhindert, dass der Bildschirm abgedunkelt wird.
- Es wird ein Hintergrundbild gesetzt.
- Das Programm `unclutter` wird gestartet, welches den Mauszeiger automatisch ausblendet.
- Der Name des Monitors wird aus der Datei `/etc/sis.conf` ausgelesen.
- Nun wird die Session des Webbrowsers initialisiert (Chromium im App-Modus mit der Startseite `http://sis.clients.htlinn.ac.at/monitors/?[Inhalt der /etc/sis.conf]`).
- Eine Session des Display-Managers „blackbox“ wird gestartet.
- Nach einer Verzögerungszeit von einer Sekunde wird das Hintergrundbild von Blackbox gesetzt.
- Nach weiteren 19 Sekunden wird mit dem Tool „xdotool“ ein Tastendruck auf F11 simuliert, wodurch der Webbrowser in den Vollbild-Modus geschaltet wird.

C.3.2.3 GPIO

Für die Ansteuerung der GPIO Pins werden zwei Shell-Scripts verwendet.

```

1  #!/bin/bash
2
3  if test "`whoami`" != "root"; then
4      echo "You are not root!"
5      exit 1;
6  fi
7
8  echo 4 > /sys/class/gpio/export 2>&1
9  echo out > /sys/class/gpio/gpio4/direction
10 echo 1 > /sys/class/gpio/gpio4/value

```

Code C.2: `/usr/bin/monitorOn`

```

1  #!/bin/bash
2
3  if test "`whoami`" != "root"; then
4      echo "You are not root!"
5      exit 1;
6  fi
7
8  echo 4 > /sys/class/gpio/export 2>&1
9  echo out > /sys/class/gpio/gpio4/direction

```

```
10| echo 0 > /sys/class/gpio/gpio4/value
```

Code C.3: /usr/bin/monitorOff

Zuerst wird geprüft, ob derausführende Benutzer root ist. Dies hat den Grund, dass nur root auf das GPIO-System schreiben darf.

Danach wird mit der Zeile 8 der GPIO-Pin 4 aktiviert. Der Pipe-Operator „2>&1“ dient dazu, dass etwaige Fehlermeldungen nicht ausgegeben werden (diese entstehen, wenn der Pin bereits als GPIO-Pin verwendet wird).

Nun (Zeile 9) wird der GPIO-Pin als Ausgang konfiguriert.

Also letztes wird nur noch der jeweilige Wert auf die Datei „value“ geschrieben. Hierbei symbolisiert 1 das logische *true*, und damit das Verhandensein der Steuerspannung (3.3 V) am Pin, und 0 das logische *false*, also 0 V.

C.3.2.4 Verteilung

Für die Verteilung der Befehle wird ein Script erstellt, das beim Systemstart mitgestartet wird (über /etc/rc.local) und danach regelmäßig Anfragen an den Server schickt. Dieses Script wird unter /usr/bin/sis-monitors gespeichert.

Für den Inhalt siehe Code C.6, Seite 62.

The tex file displaying your dot graph could not be created. This usually means one of two things:

- dot2tex is not in your path
- write18 is not enabled

Abbildung C.3: Programm-Ablauf sis-monitors

C.3.3 Probleme

C.4 Fazit

C.5 Anhänge

C.5.1 Codes

Code C.4: /etc/inittab

```

1  #!/bin/bash
2
3  #x-terminal-emulator
4
5  xset s off
6  xset -dpms
7  xset s noblank
8
9  bsetbg -center Downloads/background.png
10
11 unclutter &
12
13 name='cat /etc/sis.conf '
14 chromium-browser --app="http://sis.clients.htlinn.ac.at/monitors/?
   $name" &
15
16 i () {
17     sleep 1s
18     bsetbg -center Downloads/background.png
19 }
20
21 h () {
22     sleep 20s
23     xdotool key F11
24 }
25
26 h &
27 i &
28 blackbox

```

Code C.5: /home/pi/.xinitrc

```

1  #!/bin/bash
2
3  name=( $(cat /etc/sis.conf ) )
4
5  while true
6  do
7      wget -O /run/sis.www.tmp http://sis.clients.htlinn.ac.at/
        monitors/api/display.php?$name >> /dev/null 2>&1
8      monitor=( $(cat /run/sis.www.tmp ) )
9      if test "$monitor" = "none"
10     then
11         echo "SIS_Monitors- no_settings"
12     elif test "$monitor" = "true"
13     then
14         echo "SIS_Monitors- monitor_on"
15         monitorOn
16     else
17         echo "SIS_Monitors- monitor_off"
18         monitorOff
19     fi

```

```
20
21     wget -O /run/sis.www.tmp http://sis.clients.htlinn.ac.at/
22         monitors/api/restart.php >> /dev/null 2>&1
23     reboot=( $(cat /run/sis.www.tmp) )
24     if test "$reboot" = "kill"
25     then
26         echo "SIS: restart x"
27         killall xinit
28         sis-x &
29         sleep 20s
30     elif test "$reboot" = "false"
31     then
32         echo "SIS: do not restart x"
33     else
34         echo "SIS: and now for something completely different"
35         "
36     fi
37     sleep 10s
done
```

Code C.6: /usr/bin/sis-monitors

D || Vertiefende Aufgabenstellung: Buchberger Florian

D.1 Zusammenfassung

D.2 Projekterfahrung

D.3 Verfahrensanweisung

E || Vertiefende Aufgabenstellung: Handle W. Marco

E.1 Zusammenfassung

E.2 Projekterfahrung

E.3 Verfahrensanweisung

F || Vertiefende Aufgabenstellung: Klotz Matthias

F.1 Zusammenfassung

F.2 Projekterfahrung

F.3 Verfahrensanweisung

G || Vertiefende Aufgabenstellung: Weiland Mathias

G.1 Zusammenfassung

G.2 Projekterfahrung

G.3 Verfahrensanweisung

H || Projektplanung

H.1 Terminplanung

H.2 Personalplanung

H.2.1 Kostenplanung

H.2.2 Arbeitsnachweis

H.2.2.0.1 Buchberger Florian

H.2.2.0.2 Handle W. Marco

H.2.2.0.3 Klotz Matthias

H.2.2.0.4 Weiland Mathias

Abbildungsverzeichnis

| | | |
|-----|--|----|
| 3.1 | PHP Funktionsweise | 9 |
| 3.2 | MySQL Fetch Array | 19 |
| 3.3 | MySQL Zeile | 20 |
| 3.4 | MySQL Spalte | 20 |
| 3.5 | Supplierungs-Datenbank | 36 |
| 3.6 | Supplierungs-Datenbank | 43 |
| | | |
| B.1 | Data-Input-Menü | 47 |
| B.2 | Monitor-Einstellungen | 47 |
| B.3 | Data-Input-Menü | 51 |
| B.4 | Fehlende Auswahl | 52 |
| B.5 | Fehlende Lehrer | 53 |
| B.6 | Falsche Eingabe | 53 |
| B.7 | Fehlenden Lehrer löschen | 53 |
| | | |
| C.1 | GPIO-Pins | 56 |
| C.2 | GPIO-Pinbelegung | 57 |
| C.3 | Programm-Ablauf sis-monitors | 61 |

Codes

| | | |
|------|--|----|
| 3.1 | PHP-Tags | 8 |
| 3.2 | Session | 9 |
| 3.3 | Ausschnitt HTTP POST Request | 10 |
| 3.4 | Beispiel POST | 10 |
| 3.5 | Beispiel GET | 11 |
| 3.6 | false | 12 |
| 3.7 | MySQL Connect | 17 |
| 3.8 | MySQL Querys | 17 |
| 3.9 | MySQL Query weiterverarbeiten | 18 |
| 3.10 | MySQL Injection: Falsch | 20 |
| 3.11 | MySQL Injection: Richtig | 21 |
| 3.12 | MySQL Injection: Beispiel Query in Query | 21 |
| 3.13 | HTML-Tags | 28 |
| B.1 | Beispiel /etc/sis.conf | 48 |
| C.1 | /usr/bin/sis-x | 59 |
| C.2 | /usr/bin/monitorOn | 60 |
| C.3 | /usr/bin/monitorOff | 60 |
| C.4 | /etc/inittab | 61 |
| C.5 | /home/pi/.xinitrc | 62 |
| C.6 | /usr/bin/sis-monitors | 62 |

Tabellenverzeichnis