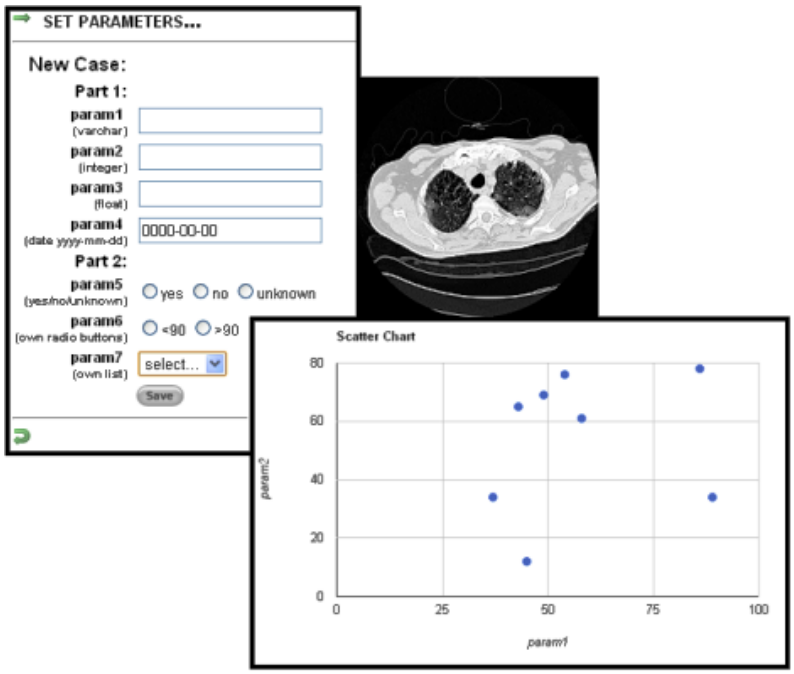
Hes·so VALAIS WALLIS
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences Western Switzerland

Bachelorarbeit 2011

Studiengang Wirtschaftsinformatik

Multidimensional clinical data acquisition



Student : Daniel Imhof

Dozent : Henning Müller

www.hevs.ch

ZUSAMMENFASSUNG

1. AUFGABENSTELLUNG

Ich habe die Projektarbeit *Multidimensional clinical data acquisition* zu meinen Favoriten gewählt, da ich gerne Webanwendungen programmiere, vor allem mit den Technologien PHP und MySQL.

2. PFLICHTENHEFT

Für die ersten rund zwei Wochen hatte ich Zeit ein detailliertes Pflichtenheft zu erfassen. Bei dieser Projektarbeit war es auch wichtig, dass ich das Pflichtenheft selber schreibe, um so mehr über die Funktionalitäten und die Arbeit zu erfahren. Anfangs war mir noch nicht so klar, was genau gemacht werden muss, da die Aufgabenstellung sehr oberflächlich beschrieben war.

Das Pflichtenheft habe ich dann in die Bereiche *Analyse der Technologien*, *Datenbank*, *Webinterface* sowie *Visualisierung der Daten und Statistiken* eingeteilt. Für einige Bereiche wurden zusätzliche zu den obligatorischen auch noch optionale Funktionalitäten definiert.

3. ARBEITSPLAN

Sobald das Pflichtenheft mit allen zu erfüllenden Funktionalitäten fest stand, erstellte ich mir einen Arbeitsplan, um das Projekt auch rechtzeitig und ohne Zeitprobleme zu realisieren.

Für die gesamte Projektarbeit habe ich 340 Stunden eingeplant.

4. ARBEITSJOURNAL

Für jede Woche habe ich alle Tätigkeiten sowie deren Aufwand schriftlich festgehalten, um zum Schluss den Arbeitsplan mit dem Arbeitsjournal zu vergleichen.

Die Erstellung und Programmierung des Webinterfaces habe ich etwas unterschätzt und somit zu wenig Zeit hierfür eingeplant. Da ich jedoch für die Datenbank und die Visualisierung der Daten und Statistiken sehr wenig Zeit beansprucht habe, hat sich dies glücklicherweise wieder ausgeglichen.

Die ersten Wochen habe ich eher zu viel gearbeitet und war dadurch ziemlich im Vorsprung. Dennoch realisierte ich alle geplanten Tätigkeiten schön der Reihe nach. Durch diesen Vorsprung konnte ich die Projektarbeit am Ende ohne Zeitprobleme fertigstellen und pünktlich abgeben.

Für die Realisierung dieser Projektarbeit benötigte ich ungefähr 330 Stunden Arbeit.

5. Sitzungen

Damit ich bei der Realisierung des Projektes immer gut vorankam, hatten wir jede Woche eine Sitzung. In jeder Sitzung haben wir zuerst über die zuletzt durchgeführten Arbeiten gesprochen. In einem zweiten Teil wurde ich immer gefragt, ob ich Probleme oder Fragen habe. Zum Schluss haben wir noch die nächsten durchzuführenden Aufgaben definiert.

Während der Woche blieben wir ständig in E-Mail-Kontakt. Dies war sehr nützlich bei Fragen, Problemen oder Bemerkungen der Dozenten.

Für jede Sitzung schrieb ich ausserdem noch jeweils immer ein Sitzungsprotokoll, in welchem ich die behandelten Themen erwähnte sowie die nächsten zu erledigenden Aufgaben.

6. Datenbank

Für die Datenbank musste ich mir klar überlegen, welche Tabellen ich verwenden werde, so dass das Webinterface am einfachsten zu realisieren ist. Ich musste hierfür immer wieder Änderungen in der Struktur und in den Tabellen selber vornehmen, da beim Programmieren immer wieder kleine Probleme auftauchten, die ich vorhin nicht durchdacht hatte.

7. Webinterface

Beim Erstellen des Webinterfaces musste ich auch immer wieder Änderungen vornehmen, bis die verantwortlichen Dozenten zufrieden waren. Immer wieder mussten Kleinigkeiten verändert werden, was sehr viel Zeit beanspruchte.

Zusätzlich zum normalen Webinterface habe ich noch ein mobiles Webinterface erstellt, um auch mit mobilen Geräten Daten zu erfassen und Statistiken zu erstellen.

Für die Entwicklung des Webinterfaces habe ich mir XAMPP mit integriertem Apache-Server (Webserver) und einer MySQL-Datenbank installiert. Für die Erstellung der einzelnen Internetseiten verwendete ich Dreamweaver.

Damit die verantwortlichen Dozenten auch immer auf dem Laufenden betreffend Webinterface waren, habe ich mir bei Funpic ein Konto erstellt, um die Internetdateien auf deren Webserver hochzuladen. Auf diese Weise konnten die Dozenten jederzeit auf der entsprechenden Seite nachschauen, wie das Webinterface aussieht. Sie haben mir oft Bemerkungen zum Realisierten via E-Mail zugeschickt. So kam ich auch zwischen den Sitzungen gut voran und hatte nie Leerlaufzeiten.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 EINLEITUNG

In der biomedizinischen Forschung werden häufig Daten für statistische Zwecke gesammelt. Solche Datensammlungen werden häufig ad-hoc und mit wechselnden Benutzerschnittstellen erstellt [1].

Eine entsprechende Datensammlung und Analyse dieser Daten inklusive Bilddaten [2] ist integraler Bestandteil der meisten biomedizinischen Forschungsprojekte.

Um den Aufwand für die Erstellung solcher Datensammlungen zu vermindern, kann eine generische Schnittstelle zur Erstellung von Datensammlungen benutzt werden. Ziel dieser Arbeit ist eine solche Schnittstelle.

Über ein Webinterface sollten die Forscher nun ganz einfach eigene Datensammlungen erstellen können und diesen die benötigten Parameter zuweisen.

Ausserdem können für jede Datensammlung Patienten gespeichert werden, da jede Datensammlung ihre eigenen Patienten aufweist.

Bei der Eingabe von Parametern kann dann schliesslich jeder Parameter nach oben und unten verschoben werden, so dass man seine eigene Reihenfolge bestimmen kann.

Falls man jetzt einen neuen Fall (Episode of care) hinzufügen möchte, müssen als erstes die Datensammlung sowie ein Patient, welcher zu dieser Datensammlung gehört, ausgewählt werden. Nun ist es den Forschern möglich, für alle vorher in der Datensammlung definierten Parametern Werte zuzuweisen, die dann zu diesem Fall (Episode of care) gespeichert werden. Diese Werte können später natürlich auch angeschaut und bearbeitet werden.

Wenn jetzt einer der Forscher etwas in der Datensammlung ändert, z.B. Änderung der Reihenfolge der Parameter oder Hinzufügen/Bearbeiten/Löschen eines Parameters, wird dies nicht nur in der Datensammlung selber geändert, sondern auch in allen Fällen (Episodes of care), welche zu dieser Datensammlung gehören. So weisen alle Fälle (Episodes of care), welche zur selben Datensammlung gehören, auch einheitlich die gleichen Parameter auf.

Eine weitere nützliche Funktion ist das Erstellen von Statistiken (Scatter Charts). Hierzu muss erstmals die Datensammlung ausgewählt werden, danach kann je ein Parameter für die X- und Y-Achse gewählt werden und schon wird die Statistik generiert.

Damit nicht jeder beliebige Internet-Benutzer auf diese Seite gelangen kann, muss man sich zuerst einloggen und sich somit Zugang zum Webinterface verschaffen. Für dieses Webinterface existieren verschiedene Benutzer mit unterschiedlichen Rechten (Administrator, Editor, User).

1.1 MOTIVATION

Im Folgenden stehen ein paar Sätze über die Motivation des Studenten und der Forscher, die dieses Projekt in Zukunft nützen möchten.

1.2 MOTIVATION DES STUDENTEN

Das Projekt *Multidimensional clinical data acquisition* galt für mich deshalb zu den Favoriten, weil ich gerne Webanwendungen erstelle, programmiere und mit Datenbanken arbeite.

Es ist auch motivierend, ein so grosses Projekt vom Anfang bis zum Schluss alleine durchzuziehen und Verantwortung dafür zu übernehmen.

Eine zusätzliche Motivation ist natürlich die Tatsache, dass dieses Projekt später auch produktiv verwendet werden soll.

1.3 MOTIVATION DER FORSCHER

Für die Forscher ist ein flexibles Webinterface für die Verwaltung von klinischen Daten sehr hilfreich. Zudem können sie aus bestimmten Daten auch Statistiken erstellen.

2 METHODEN

Dieses Kapitel ist in zwei Teile eingeteilt. Zum einen in ein Unterkapitel *Wahl der Technologien*, wo ein paar Informationen zu den einzelnen Technologien stehen und schliesslich eine kurze Begründung für die getroffene Wahl, zum anderen in weitere Methoden und Programme, die für die Realisierung dieses Projektes benötigt wurden.

2.1 WAHL DER TECHNOLOGIEN

Bevor mit der Implementierung begonnen werden kann, müssen natürlich Technologien verglichen werden, um dann schliesslich eine gute Wahl treffen zu können.

Für zwei zentrale Technologien musste eine Entscheidung getroffen werden. Einerseits für eine Web-Programmiersprache und andererseits für eine relationale Datenbank zur Speicherung der Daten.

Zum Schluss war noch eine Methode nötig, um Statistiken aus den Daten der Datenbank einfach erstellen zu können.

2.1.1 WEB-PROGRAMMIERSPRACHE

Im Folgenden werden von den beiden Web-Programmiersprachen *ASP.NET* und *PHP* jeweils die Vor- und Nachteile aufgelistet.

Anschliessend wird die Entscheidung kurz begründet.

2.1.1.1 ASP.NET (C#, VB.NET)[1]

2.1.1.1.1 Vorteile

- ASP.NET bietet eine klare Trennung von Design und Programmcode
- Geschwindigkeitsgewinn nach dem Prinzip "Kompilieren statt interpretieren"
- Einfache Datenbankintegration, Lokalisierung und Fehlerbehandlung

2.1.1.1.2 Nachteile

- Eine durchaus steilere Lernkurve als bei PHP
- Voller Funktionsumfang ist bisher an die Implementierung von Microsoft gebunden (die entsprechenden Alternativen für Linux und Mac OS X verfügen jedoch bereits über die wichtigsten Funktionen)

2.1.1.2 PHP[2]

2.1.1.2.1 Vorteile

- PHP ist leicht zu lernen – Umsteiger werden auf den ersten Blick die Ähnlichkeiten mit Java oder C++ erkennen und schon nach wenigen Minuten sollten sich die Meisten wie zu Hause fühlen. Neulinge können schnell in die Programmierung einsteigen und bald erste Ergebnisse im Browser betrachten.
- PHP ist frei verfügbar – Jeder kann den Quellcode von PHP einsehen und nach seinen Vorstellungen verbessern oder weiterentwickeln. Wer einfach nur PHP-Skripte entwickeln will, kann sich jederzeit kostenlos die Software herunterladen und auf seinem Rechner installieren.

---

[1] *http://de.wikibooks.org/wiki/Webentwicklung_mit_ASP.NET:_Grundlagen_und_Programmiersprachen* (Stand: 01.06.2011)

[2] *http://de.wikibooks.org/wiki/Websiteentwicklung:_PHP* (Stand: 01.06.2011)

- PHP ist vielseitig – Es sind PHP-Versionen für Unix, Windows, Mac OS und einige andere Plattformen verfügbar.PHP hat standardmäßig viele Schnittstellen zu anderen Programmiersprachen und Datenbanken.
- PHP ist populär – Mittlerweile verwenden viele Millionen Internetseiten PHP. In unterschiedlichsten Einsatzgebieten und Webseiten, die von der einfachen Homepage bis zum Mega-Webportal reichen, hat sich PHP bereits bewährt.
- PHP ist gut dokumentiert – Die einfache Erlernbarkeit von PHP ist wohl vor allem der ausgesprochen guten Homepage zu verdanken. Allerdings gibt es auch endlos viele Bücher, Foren und Newsgruppen zum Thema PHP.

2.1.1.2.2 Nachteile

- PHP hat bislang keinen Anwendungs-Server – d.h., alle Variablen müssen bei jedem Aufruf wieder neu geladen bzw. berechnet werden.
- PHP ist populär – Was oft ein Vorteil ist kann durchaus auch ein Nachteil sein. Schwachstellen in PHP oder PHP-Skripten können schneller und wirksamer ausgenutzt werden, weil viele potentielle Opfer verfügbar sind.
- PHP ist leicht zu lernen – Auch das kann ein Nachteil sein, weil deshalb viele PHP-Projekte ohne das nötige Hintergrundwissen verwirklicht werden. Die Folge davon ist, dass viele Websites gravierende Sicherheitsmängel aufweisen.
- PHP ist eine rein interpretierte Sprache und die erzeugten Programme damit vergleichsweise langsam in der Ausführung. Dies kann jedoch durch Verwendung von Op-Code-Caches ausgeglichen werden.
- PHP hat im Gegensatz zu anderen Skriptsprachen keine Möglichkeiten der Programmierung von Threads.

2.1.1.3 ENTSCHEIDUNG

Ich habe mich für PHP entschieden, da ich schon viel Erfahrung damit habe, auch in Zusammenhang mit der Datenbankanbindung. Ich würde also viel Zeit sparen und könnte mich umso mehr auf die Funktionalitäten konzentrieren, da ich mich nicht noch zuerst in die Technologie einarbeiten muss.

Zudem sind im Internet viele Informationen über PHP zu finden. Dies wird sehr nützlich sein, falls zusätzliche Informationen nötig sind im Verlauf des Projektes.

2.1.2 DATENBANK[3]

Im Folgenden werden von den beiden Datenbanken *PostgreSQL* und *MySQL* jeweils die Vor- und Nachteile aufgelistet.

Anschliessend wird die Entscheidung kurz begründet.

2.1.2.1 POSTGRESQL

2.1.2.1.1 Vorteile

- PostgreSQL steht frei zur Verfügung und bietet eine umfangreiche Anzahl an unterstützten Features wie z.B. Transaktionen, referentielle Integrität, Views, Triggers,…
- Ebenfalls wird die Erstellung eigener Datentypen, Operatoren, Aggregatfunktionen und Funktionen unterstützt.
- Eine weitere Besonderheit von PostgreSQL ist, dass eigene Datenbankfunktionen in verschiedenen Sprachen geschrieben werden können. Unterstützte Sprachen sind unter anderen PG/SQL, C, Perl, Python, TCL, PHP,...
- Für das RDMS steht ein ausgezeichnetes Handbuch bereit, welches allerdings nur in Englisch zur Verfügung steht. In Frankreich und anderen Ländern gibt es jedoch Gruppen, die das Handbuch in ihre jeweilige Landessprache überführen.
- Zur einfachen Administration von PostgreSQL können die beiden hervorragenden Programme phpPgAdmin oder pgAdmin verwendet werden.

2.1.2.1.2 Nachteile

- Die vielfältigen und umfangreichen Möglichkeiten PostgreSQLs wirken gerade auf Anfänger abschreckend, doch wird dies durch das Handbuch und die Community schnell relativiert. Schnell wird klar, dass diese Unterstützungen grundlegend für eine professionelle Entwicklung von Datenbankanwendungen sind. Über viele Jahre hinweg hat dieser Umfang aber die Geschwindigkeit von PostgreSQL stark negativ beeinflusst. In den letzten Jahren ist die Performance von PostgreSQL jedoch stark gestiegen.

[3] *http://www.php-faq.de/q-db-vergleich.html*
(Stand: 01.06.2011)

2.1.2.2 MySQL

2.1.2.2.1 Vorteile

- MySQL ist das wohl bekannteste und am weitesten verbreitete freie relationale Datenbankmanagementsystem. Es zeichnet sich durch eine hohe Geschwindigkeit und geringen Speicherverbrauch aus.
- Mit Version 5.0 wurde die Unterstützung für eine Reihe weiterer Features eingeführt, wobei die bedeutendsten Trigger, Views und Prozeduren darstellen.
- Das Datenbanksystem verfügt über ein sehr gutes und umfangreiches Handbuch in mehreren Sprachen. Durch die weite Verbreitung von MySQL gibt es eine breite Community von Nutzern, die auf vielfältige Weise Hilfe anbieten – von einfachen Tipps in Foren bis hin zur professionellen Betreuung und Schulungen zum RDMS.
- Um eine einfache Administration zu ermöglichen, gibt es für MySQL eine Vielzahl an Lösungen, allen voran das bekannte phpMyAdmin.

2.1.2.2.2 Nachteile

- Die hohe Geschwindigkeit der Weiterentwicklung von MySQL ist zugleich auch ein Nachteil von MySQL. Viele der angebotenen Features lassen sich nicht kombinieren.

2.1.2.3 Entscheidung

Ich habe mich für MySQL entschieden, da ich auch in diesem Gebiet, genauso wie im Bereich *Web-Programmiersprache*, viel Erfahrung mit dem Umgang dieser Technologie mitbringe und somit keine Zeit für die Einarbeitung verschwenden werde.

Zudem ist es mit einer MySQL-Datenbank recht einfach, diese in PHP anzubinden. Für die Verwaltung der Daten ist auch eine sehr einfach zu bedienende Oberfläche verfügbar (phpMyAdmin).

2.1.3 Statistik

Für Bereich *Statistik* war im Internet ausser von *Google Chart API* kaum etwas Sinnvolles zu finden.

Ausserdem ist die Programmierschnittstelle (API) von Google sehr einfach zu benützen und bietet viele Features zur Gestaltung der jeweiligen Statistik.

Die Entscheidung für den Bereich *Statistik* war somit klar.

2.2 TECHNOLOGIEN

Folgend aufgelistete Methoden und Programme wurden für die Realisierung dieses Projektes benötigt.

2.2.1 PHP[4]

Die kostenlose Skriptsprache PHP kann verwendet werden, um dynamische Webanwendungen zu erstellen [4]. PHP zeichnet sich auch durch die breite Datenbankunterstützung aus [3].

PHP wird auf ca. 75% aller Webseiten als serverseitige Programmiersprache eingesetzt und ist damit die am häufigsten verwendete Sprache zur Erstellung von Webseiten.

2.2.2 MYSQL

MySQL ist ein relationales Datenbankverwaltungssystem, das einfach in PHP integriert sowie mit Hilfe des nützlichen und einfach zu bedienenden Tools *phpMyAdmin* verwaltet werden kann [5].

2.2.3 DREAMWEAVER

Dreamweaver ist ein gutes Programm, um Internetseiten zu erstellen. Hier kann zudem zwischen Code- und Design-Ansicht gewählt werden, was das Erstellen der Seiten vereinfacht. Die Design-Ansicht ist beispielsweise für das Gestalten von Formularen sehr hilfreich.

Wichtig ist auch, dass der Code je nach Typ anders farblich dargestellt wird, so dass es einfacher ist, die Übersicht über den Programm-Code zu behalten.

---

[4] *http://de.wikipedia.org/wiki/PHP*
(Stand: 09.08.2011)

2.2.4 XAMPP

Das Programm *XAMPP* bietet neben einem Apache-Server (Web-Server), der u.a. PHP-Code interpretieren kann, auch noch eine MySQL-Datenbank.

XAMPP bedeutet folgendes:

- X plattformunabhängig
- A Apache-Server
- M MySQL
- P Perl
- P PHP

2.2.5 GOOGLE CHART API[5]

Google Chart API ist eine Programmierschnittstelle (API) von Google, die folgende Statistiken zur Verfügung stellt:

- Area Charts
- Bar Charts
- Candlestick Charts
- Column Charts
- Combo Charts
- Gauge Charts
- Geo Charts
- Line Charts
- Pie Charts
- Scatter Charts
- Table Charts
- Tree Map Charts

In diesem Projekt wurde *Scatter Charts* verwendet.

Im folgenden Beispiel ist der Code-Teil, in welchem die eigentlichen Daten angegeben werden, mit einem roten Viereck umrahmt.

Diese Statistik besteht aus 6 Punkten im Koordinatensystem. Ein Datensatz wie beispielsweise *data.setValue(0, 0, 8);* steht für Folgendes:

- 0: Nummer des Punktes
- 0: X (1 wäre Y)
- 8: X-Wert

[5] *http://code.google.com/intl/de-DE/apis/chart/interactive/docs/gallery.html* (Stand: 05.08.2011)

Dieses Beispiel beinhaltet jedoch nicht alle möglichen Features.

```
<html>
  <head>
    <script type="text/javascript" src="https://www.google.com/jsapi"></script>
    <script type="text/javascript">
      google.load("visualization", "1", {packages:["corechart"]});
      google.setOnLoadCallback(drawChart);
      function drawChart() {
        var data = new google.visualization.DataTable();
        data.addColumn('number', 'Age');
        data.addColumn('number', 'Weight');
        data.addRows(6);
        data.setValue(0, 0, 8);
        data.setValue(0, 1, 12);
        data.setValue(1, 0, 4);
        data.setValue(1, 1, 5.5);
        data.setValue(2, 0, 11);
        data.setValue(2, 1, 14);
        data.setValue(3, 0, 4);
        data.setValue(3, 1, 4.5);
        data.setValue(4, 0, 3);
        data.setValue(4, 1, 3.5);
        data.setValue(5, 0, 6.5);
        data.setValue(5, 1, 7);

        var chart = new google.visualization.ScatterChart(document.getElementById('chart_div'));
        chart.draw(data, {width: 400, height: 240,
                          title: 'Age vs. Weight comparison',
                          hAxis: {title: 'Age', minValue: 0, maxValue: 15},
                          vAxis: {title: 'Weight', minValue: 0, maxValue: 15},
                          legend: 'none'
                         });
      }
    </script>
  </head>

  <body>
    <div id="chart_div"></div>
  </body>
</html>
```

Abb. 1 Google Chart API Beispiel-Code

Die Statistik (Scatter Chart) des obigen Code-Beispiels würde wie folgt aussehen:

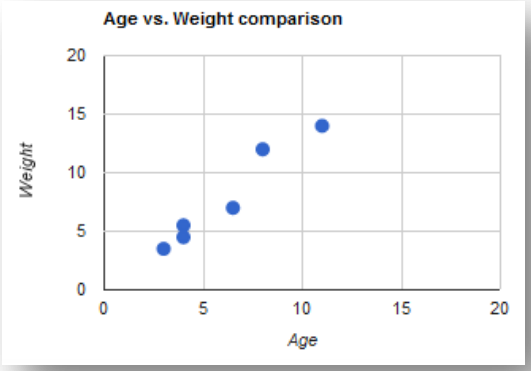


Abb. 2 Google Chart API Beispiel-Vorschau

2.2.6 NOTEPAD++

Diese Dokumentation enthält einige Code-Ausschnitte aus den PHP-Files, wofür zur Darstellung des Codes das Programm *Notepad++* verwendet wurde.

3 RESULTAT

Dieses Kapitel ist in die drei Unterkapitel *Installation*, *Webinterface* und *Datenbank* unterteilt und gibt Ihnen die wichtigsten Informationen, um den Aufbau des Webinterfaces zu verstehen.

3.1 INSTALLATION

Um das Webinterface zu entwickeln wurde lokal eine Entwicklungsumgebung eingerichtet. Damit das fortlaufend Entwickelte den Experten auch immer ganz einfach gezeigt werden konnte, wurde zudem auch eine Online-Entwicklungsumgebung eingerichtet.

3.1.1 LOKAL

Nach der Installation von XAMPP sollten die Dienste Apache (Webserver) und MySql (relationale Datenbank) gestartet werden.

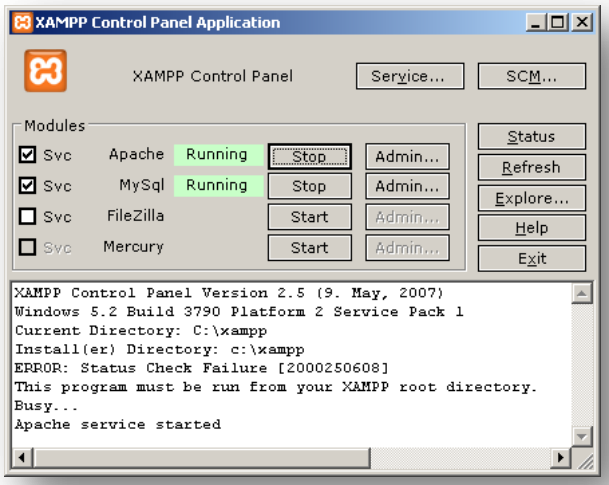


Abb. 3 XAMPP starten

Nach der Installation von Dreamweaver wurde eine neue Site in Dreamweaver eingerichtet, um die Files des Webinterfaces dort verwalten zu können.

Unter dem Pfad *C://xampp/htdocs/ba/* konnten nun alle Files abgelegt werden, die dann vom Apache-Server (Webserver) von XAMPP gelesen werden konnten. Dieser Apache-Server war auch fähig, PHP-Code zu interpretieren.

Beim Aufruf von *http://localhost/ba/* wird nun die Datei *index.html* aufgerufen, die natürlich zuvor erstellt werden muss. Durch diese Adresse war nun das Webinterface zugänglich, welches lokal getestet und später online gestellt werden konnte.

Unter der Adresse *http://localhost/phpmyadmin/* war es möglich, die lokale Datenbank mit einem einfachen GUI zu verwalten.

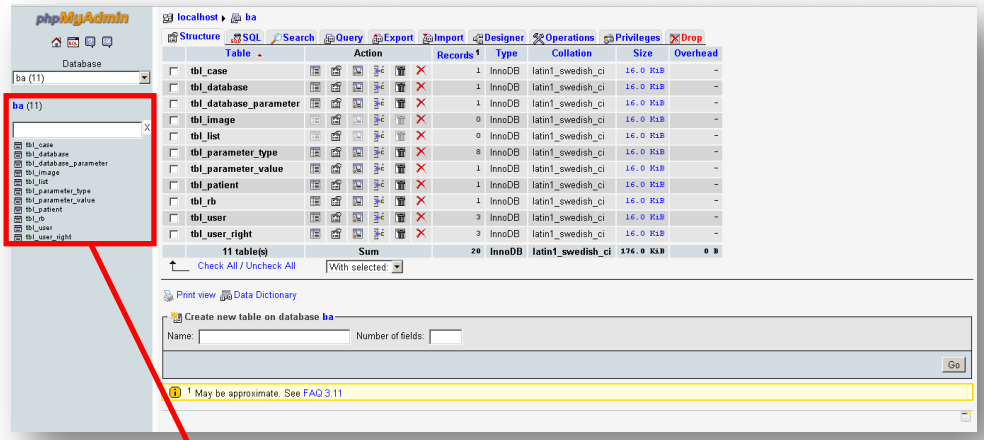


Abb. 4 phpMyAdmin Übersicht

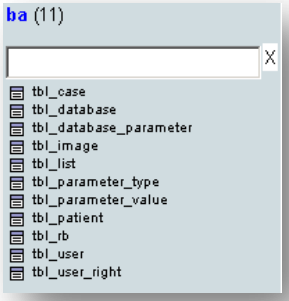


Abb. 5 phpMyAdmin Datenbank-Tabellen

In diesem Fall ist *ba* die Datenbank und darunter sind alle 11 Tabellen dieser Datenbank aufgelistet.

3.1.2 ONLINE

Um das erstellte Webinterface auch den verantwortlichen Dozenten auf einfache Art und Weise zeigen zu können, wurde bei Funpic[6] ein Konto eingerichtet.

Hierfür mussten nur alle Files des Webinterfaces auf den Funpic-Server hochgeladen werden. Dazu wurde ein FTP-Dienst[7] verwendet.

Die Online-Datenbank konnte hier genauso wie lokal mit phpmyadmin verwaltet werden. Dafür war nur ein Login bei Funpic, ein Klick im User-Center auf *Webhosting MySQL* und zum Schluss ein weiterer Klick im dann erscheinenden Fenster auf *phpmyadmin* nötig.

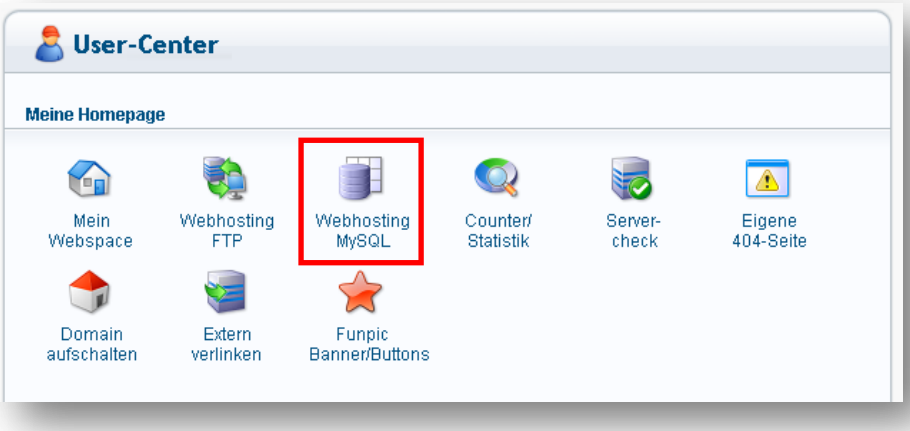


Abb. 6 Funpic User-Center

6 *http://www.funpic.de* (Stand: 08.08.2011)

7 *http://www2ftp.de* (Stand: 08.08.2011)

3.2 WEBINTERFACE

Dieses Kapitel ist in drei Bereiche eingeteilt. Zuerst geht es darum, ob beim Betreten der Seite das nichtmobile oder das mobile Webinterface gestartet wird.

In einem zweiten Teil wird der allgemeine Aufbau der Seite anhand eines Beispiels erklärt.

Zum Schluss werden noch zusätzliche wichtige PHP-Files erwähnt sowie deren Funktion beschrieben.

3.2.1 NICHT-MOBILES/MOBILES WEBINTERFACE

Wenn die Seite *index.html* (Startscreen) aufgerufen wird, wird zuerst geprüft, ob es sich um ein mobiles Gerät handelt oder nicht. Je nachdem wird auf die entsprechende Seite weitergeleitet.

Dies wird mit folgendem Javascript-Code überprüft:

```
<script type="text/javascript">
<!--
if (screen.width <=600) {
document.location = "mobile.index.html";
}
//-->
</script>
```

Abb. 7 Javascript-Code nichtmobiles/mobiles Webinterface

Falls es kein mobiles Gerät ist, bleibt der Benutzer auf der Seite *index.html*, ansonsten wird er auf die Seite *mobile.index.html* geleitet.

Jede einzelne Seite des nicht-mobilen Webinterfaces existiert auch für das mobile Webinterface. Nur das beim mobilen Webinterface vor jeder Datei *mobile.* steht.

Der einzige Unterschied der Dateien des mobilen Webinterfaces ist der DOCTYPE zuoberst in der HTML- oder PHP-Datei:

Nicht-mobile Datei:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

Mobile Datei:

<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN" "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">

3.2.2 ZUSAMMENSPIEL DER PHP-FILES

Beim Klicken auf auf dem Startscreen wird eine bestimmte PHP-Datei aufgerufen, die überprüft, ob die Login-Daten korrekt sind. Falls die Login-Daten korrekt sind, wird man auf das eigentliche Webinterface weitergeleitet. Je nach Benutzertyp (Administrator, Editor, User) hat man andere Rechte.

Im Folgenden wird das Zusammenspiel der benötigten PHP-Seiten für die Rubrik *User management* des Webinterfaces aus Sicht des Benutzertyps Administrator gezeigt. Alle anderen Rubriken und Unter-Rubriken sind im gleichen Stil aufgebaut.

Beim Klicken im Webinterface auf *User management* erscheint folgender Screen:

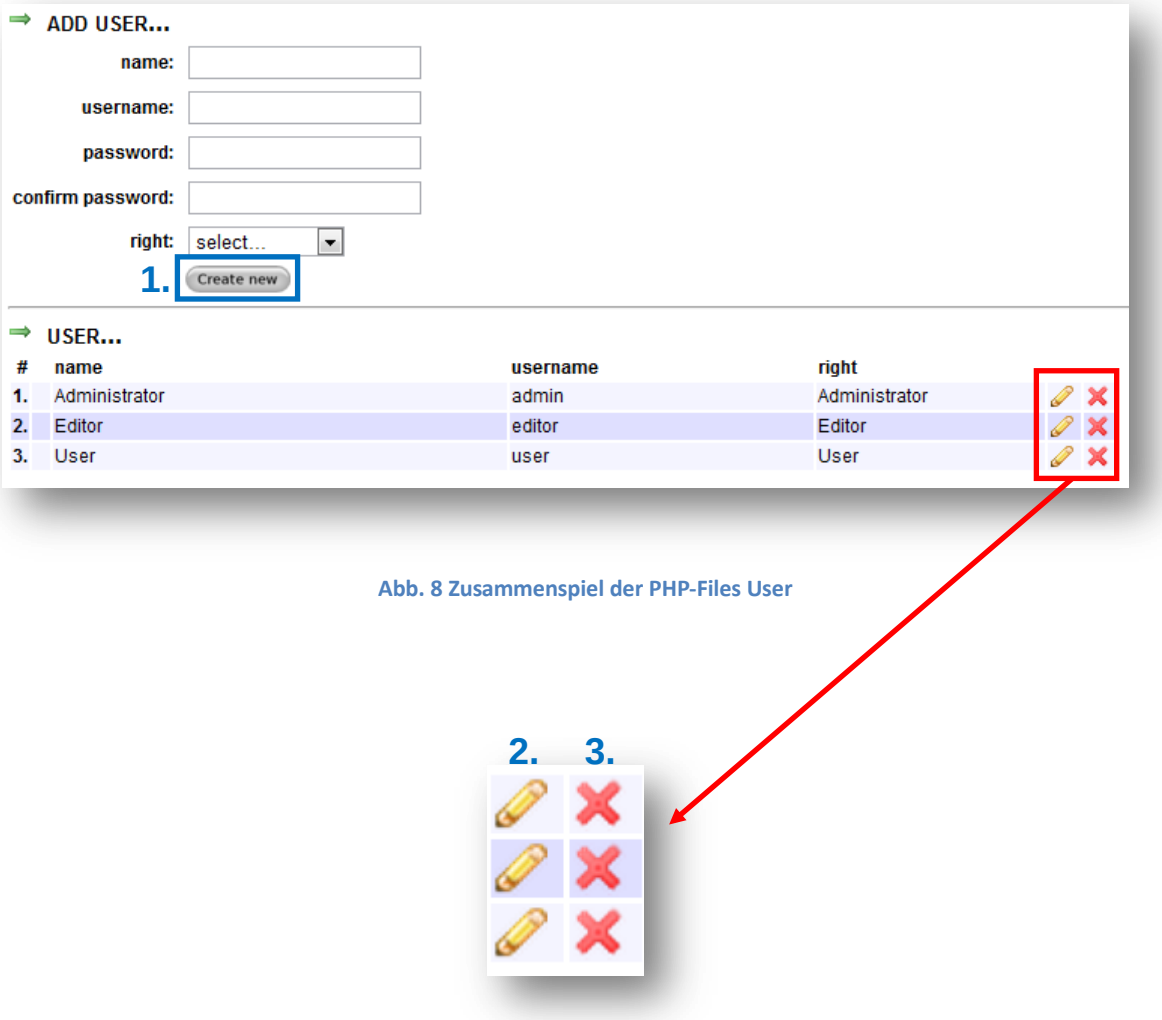


Abb. 8 Zusammenspiel der PHP-Files User

1. Oben unter *add user*... können neue Benutzer hinzugefügt werden. Beim Klicken auf den Button *Create new* wird ein entsprechendes PHP-File für das Hinzufügen eines Benutzers aufgerufen. Bei korrekter Dateneingabe wird dann der Benutzer in die Datenbank hinzugefügt und es erscheint folgende Meldung:

   User successfully added.

   Falls nicht alle Felder korrekt ausgefüllt wurden, erscheint folgende Fehlermeldung:

   Error: Please fill out all fields correctly.

   Falls ein Benutzer bereits existiert, erscheint folgende Fehlermeldung:

   Error: Username already exist.

   Letztere Fehlermeldung erscheint natürlich auch, wenn beim Bearbeiten eines existierenden Benutzers ein Benutzername gewählt wird, der schon existiert. In diesem Fall erscheint die obige Fehlermeldung im Edit-Fenster.

Unter *user*... werden alle Benutzer der Datenbank angezeigt und können hier einfach verwaltet werden.

2. Beim Klicken auf beispielsweise des Benutzers *Editor* erscheint folgendes Fenster:

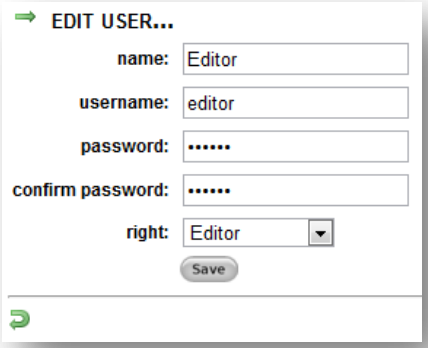


Abb. 9 Zusammenspiel der PHP-Files User bearbeiten

Der Benutzer *editor* kann hier nun bearbeitet werden. Mit einem Klick auf *Save* wird erneut ein entsprechendes PHP-File für das Speichern des bearbeiteten Benutzers aufgerufen. Bei korrekter Dateneingabe gelangt man zurück auf das Hauptfenster der Rubrik *User management* und es erscheint folgende Meldung:

User successfully saved.

Falls nicht alle Felder korrekt ausgefüllt wurden, gelangt man zurück auf das Edit-Fenster und es erscheint folgende Fehlermeldung:

Error: Please fill out all fields correctly.

3. Wenn ein Benutzer gelöscht werden soll, kann auf das Symbol ✖ beispielsweise des Benutzers *Editor* geklickt werden. Aus Sicherheitsgründen muss das Löschen eines Benutzers bestätigt werden.

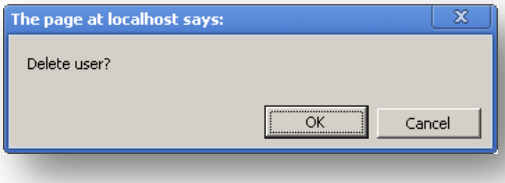


Abb. 10 Zusammenspiel der PHP-Files User löschen

Beim Klicken auf *OK* wird das entsprechende PHP-File für das Löschen eines Benutzers aufgerufen und der Benutzer *editor* wird gelöscht. Nach dem Löschvorgang gelangt man wieder auf die Hauptseite der Rubrik *User management* und es erscheint folgende Meldung:

User successfully deleted.

Beim Klicken auf *Cancel* wird der Benutzer nicht gelöscht.

Für die Rubrik *User management* braucht es also 5 PHP-Files:

1. Hauptfenster (oben Maske *Hinzufügen*, unten Benutzer *Anzeigen*)
2. User Add
3. User Edit (Maske *Bearbeiten*)
4. User Save (Bearbeiteten Benutzer speichern)
5. User Delete

Hinweis: Detaillierte Informationen über die Handhabung des Webinterfaces finden Sie übrigens im Anhang unter dem Kapitel Benutzerhandbuch.

3.2.3 ZUSÄTZLICHE PHP-FILES

3.2.3.1 LOGIN_CHECK.PHP

Diese Klasse ist für das Login zuständig. Sie sucht in der Datenbank-Tabelle *tbl_user* nach dem eingegebenen Benutzernamen. Dann vergleicht sie das dazugehörige Kennwort mit dem eingegebenen Kennwort. Falls diese übereinstimmen, gelangt man auf das Webinterface, ansonsten wird man zurück auf den Startscreen (Login) geleitet und das Login ist somit fehlgeschlagen.

Zudem werden der *username* sowie die *right_id* (Administrator, Editor, User) jeweils in eine Session-Variable gespeichert.

```
$query = new Query("SELECT
                    tbl_user_username,
                    tbl_user_password,
                    tbl_user_fk_tbl_user_right_id
                    FROM tbl_user
                    WHERE tbl_user_username = '" . $tbl_user_username . "' LIMIT 1");

$row = mysql_fetch_array($query->getResult());

//check login data
if(($tbl_user_password != '') && ($tbl_user_password == $row['tbl_user_password'])){
    $_SESSION['tbl_user_username'] = $tbl_user_username;
    $_SESSION['right'] = $row['tbl_user_fk_tbl_user_right_id'];
    header('location: webinterface.php');
}
else{
    header('location: index.html');
}
```

Abb. 11 login_check.php

3.2.3.2 LOGOUT.PHP

Diese Klasse ist für das Logout zuständig. Hier werden alle Sessions mit dem Befehl *session_destroy()* gelöscht und man wird auf den Startscreen (Login) geleitet.

```
session_start();

//delete all session variables
session_destroy();
header('location: index.html');
```

Abb. 12 logout.php

3.2.3.3 CLASS_DBCONN.PHP

Diese Klasse stellt eine Verbindung zur Datenbank her. Sie dient auch noch dazu, eine Datenbank-Verbindung zu schliessen.

```
function connect(){
    $this->conn = mysql_connect($this->server,
                                $this->user,
                                $this->password)
                                or die ('database connection failed.');
    mysql_select_db($this->db);
}

function disconnect(){
    if(isset($_SESSION['tbl_user_username'])){
        mysql_close ($this->conn);
    }
    else{
        header('location: empty.php');
    }
}
```

Abb. 13 class_DbConn.php

3.2.3.4 CLASS_QUERY.PHP

Dieser Klasse gibt man eine Abfrage mit, diese wird ausgeführt und man kann sich mit der entsprechenden Funktion *getResult()* das Resultat holen, falls dies nötig ist.

```
function getData(){
    //save result
    $this->result = mysql_query($this->query);
}

function getResult(){
    //get result
    return $this->result;
}
```

Abb. 14 class_Query.php

3.2.3.5 CLASS_FREE.PHP

Diese Klasse überprüft, ob beispielsweise ein eingegebener Benutzername bereits existiert oder nicht. Wobei dieser Klasse die Resultate einer Abfrage mitgegeben werden müssen. Als Ergebnis erhält man *true* oder *false*.

```
function isFree(){
    //check if it is free
    if(mysql_num_rows($this->result) == 0){
        $this->isFree = 'true';
    }
    else{
        $this->isFree = 'false';
    }
}

function getIsFree(){
    //get result
    return $this->isFree;
}
```

Abb. 15 class_Free.php

3.2.3.6 CLASS_UPANDDOWN.PHP

Diese Klasse ist dafür zuständig, zwei Datenbank-Parameter miteinander zu verstauschen, falls man auf die entsprechenden Pfeile drückt.

Diese Klasse sorgt dann auch noch dafür, dass auch bei allen Fällen (Episodes of care) dieser Datenbank die Parameter korrekt vertauscht werden, so dass die Reihenfolge überall gleich ist.

3.2.3.7 CLASS_EXPORT.PHP

Diese Klasse holt, nachdem eine Datenbank ausgewählt wurde, alle Daten aus den entsprechenden Tabellen, die zu dieser ausgewählten Datenbank gehören, und schreibt die jeweiligen SQL-Befehle in eine SQL-Datei. Diese Datei kann dann heruntergeladen[8] werden.

Es werden die Daten folgender Tabellen in die SQL-Datei geschrieben:

- tbl_database
- tbl_database_parameter
- tbl_rb
- tbl_list
- tbl_case
- tbl_parameter_value
- tbl_patient

[8] *http://www.php-dummies.de/script/Tutorials/Praxis/File_Download.html* (Stand: 09.08.2011)

3.2.3.8 CLASS_CHART.PHP

Diese Klasse holt aus der Datenbank die entsprechenden Informationen der ausgewählten Parameter für die X- und Y-Achse, speichert die X-Werte in ein Array *paramX* und die Y-Werte in ein Array *paramY* und schreibt den nötigen Code in die Datei *chart.html*. Diese Datei wird dann direkt angezeigt. Eine Statistik (Scatter Chart) mit den beiden ausgewählten Parametern für die X- und Y-Achse wird dann schliesslich angezeigt.

Der Code-Ausschnitt fürs Schreiben der X- und Y-Werte in die Datei *chart.html* sieht wie folgt aus:

```
$numberOfRows = count($paramX);
fWrite($handler , "data.addRows($numberOfRows);");
//write values
for($i=0; $i<(count($paramX)); $i++){
    fWrite($handler , "data.setValue($i, 0, $paramX[$i]);");
    fWrite($handler , "data.setValue($i, 1, $paramY[$i]);");
}
```

Abb. 16 class_Chart.php

3.2.3.9 CASE_IMAGE_ADD_DICOM.PHP

Diese Klasse ermittelt zuerst den Datentyp, der hochgeladen werden soll. Falls es ein einzelnes File ist, wird es in die Datenbank eingetragen und hochgeladen. Ist es jedoch ein ZIP-File, wird es hochgeladen, entpackt[9] und schliesslich noch in die Datenbank eingetragen.

Vor dem Upload wird im Ordner *./upload/* des Webservers zusätzlich noch ein Ordner erstellt, dessen Name die ID des Falls (Episode of care) ist. Alle Uploads dieses Falls (Episode of care) werden dann in diesen erstellten Ordner gespeichert.

[9] *http://php.net/manual/de/function.ziparchive-extractto.php* (Stand: 07.08.2011)

3.3 DATENBANK

Nach dem Unterkapitel *ERM* finden Sie noch ausführliche Informationen über die einzelnen Tabellen-Felder und deren Funktion.

3.3.1 ENTITY-RELATIONSHIP-MODEL

In der untenstehenden Abbildung sehen Sie die Verbindungen zwischen den einzelnen Datenbank-Tabellen (Entity-Relationship-Model, kurz ERM).

Die einzelnen Felder der jeweiligen Tabellen wurden in dieser Abbildung ausgeblendet.

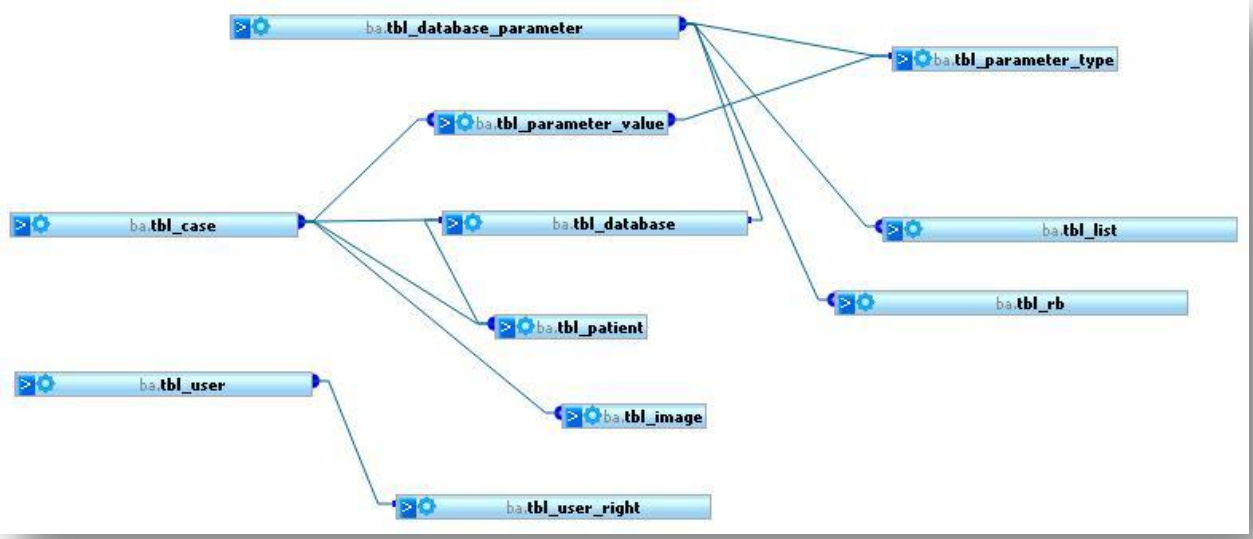


Abb. 17 Datenbank ERM

3.3.2 DATENBANK-TABELLEN

3.3.2.1 TBL_USER

Felder: id, name, username, password, fk_tbl_user_right_id

Hier werden die User gespeichert. Mit *fk_tbl_user_right_id* wird dem Benutzer ein Typ zugeordnet.

3.3.2.2 TBL_USER_RIGHT

Felder: id, description

Hier werden die Typen (Administrator, Editor, User) gespeichert.

3.3.2.3 TBL_DATABASE

Felder: id, description

Hier werden die Datenbanken gespeichert.

3.3.2.4 TBL_DATABASE_PARAMETER

Felder: id, fk_tbl_database_id, description, unit, fk_tbl_parameter_type_id, title_size

Hier werden die erforderlichen Parameter einer Datenbank gespeichert.

Mit *fk_tbl_database_id* sind die Parameter mit der entsprechenden Datenbank verknüpft.

Mit *fk_tbl_parameter_type_id* wird dem Parameter ein Typ zugeordnet.

Für den Typ *title* kann noch eine Schriftgrösse angegeben werden. Diese wird dann im Feld *title_size* gespeichert.

3.3.2.5 TBL_RB

Felder: id, value, tbl_database_parameter_id

Hier werden die Einträge der Radio Buttons gespeichert.

Mit *tbl_database_parameter_id* sind diese Einträge mit dem entsprechenden Datenbank-Parameter verknüpft.

3.3.2.6 TBL_LIST

Felder: id, value, tbl_database_parameter_id

Hier werden die Einträge der Listen gespeichert.

Mit *tbl_database_parameter_id* sind diese Einträge mit dem entsprechenden Datenbank-Parameter verknüpft.

3.3.2.7 TBL_PARAMETER_VALUE

Felder: id, description, fk_tbl_parameter_type_id, valueVarchar, valueInteger, valueFloat, valueDate, unit, fk_tbl_case_id, database_parameter_id

Hier werden die Parameter-Werte von einem Fall (Episode of care) gespeichert. Hierfür werden die notwendigen Parameter der ausgewählten Datenbank aus der Tabelle *tbl_database_parameter* geholt und beim Speichern in die Tabelle *tbl_parameter_value* gespeichert.

Mit *fk_tbl_parameter_type_id* wird dem Parameter ein Typ zugeordnet.

Da nicht alle Parameter denselben Typ haben, wurden hier entsprechende Felder (valueVarchar, valueInteger, valueFloat, valueDate) hinzugefügt.

Mit *fk_tbl_case_id* ist der Parameter mit einem Fall (Episode of care) verbunden.

Das Feld *database_parameter_id* ist für die Parameter des Typs *Radio Button* und *List* notwendig. Falls ein solcher Parameter unter der Rubrik *Episodes of care* bearbeitet werden soll, müssen alle Einträge (Radio Buttons oder Listen) dieses Parameters aufgelistet werden, um daraus einen Wert auszuwählen. Da diese Einträge in der Tabelle *tbl_rb* bzw. *tbl_list* gespeichert sind, diese Einträge wiederum mit der Tabelle *tbl_database_parameter* verknüpft sind, ist die Parameter-ID (*database_parameter_id*) notwendig.

Zusätzlich ist das Feld *database_parameter_id* dafür notwendig, um alle Fälle (Episodes of care) einer bearbeiteten Datenbank zu synchronisieren. Wenn in einer Datenbank unter der Rubrik *Databases*

Parameter hinzugefügt, bearbeitet, gelöscht oder verschoben werden, müssen die Parameter aller Fälle (Episodes of care) einer Datenbank auch entsprechend erneuert werden.

3.3.2.8 TBL_PARAMETER_TYPE

Felder: id, type

Hier werden alle Typen (varchar, integer, float, date,...) gespeichert.

3.3.2.9 TBL_PATIENT

Felder: id, lastName, firstName, birth_date, initial, fk_tbl_database_id

Hier werden alle Patienten gespeichert. Da eine Datenbank seine eigene Patienten-Sammlung besitzt, wird ein Patient mit dem Feld *fk_tbl_database_id* mit der jeweiligen Datenbank verbunden.

3.3.2.10 TBL_CASE

Felder: id, fk_tbl_database_id, fk_tbl_patient_id, description, date

Hier werden alle Fälle (Episodes of care) gespeichert.

Mit *fk_tbl_database_id* und *fk_tbl_patient_id* ist ein Fall (Episode of care) mit einer Datenbank bzw. mit einem Patienten verbunden.

3.3.2.11 TBL_IMAGE

Felder: id, path, fk_tbl_case_id, name

Hier werden alle hochgeladenen Files gespeichert.

Mit *fk_tbl_case_id* ist ein Bild mit einem Fall (Episode of care) verknüpft.

3.4 TESTS

Die Tests 1-4 müssen sowohl für das nicht-mobile als auch für das mobile Webinterface getestet werden.

Test 5 und 6 werden nur im nicht-mobilen Webinterface getestet, da im mobilen Webinterface keine Files hochgeladen und Datenbanken exportiert werden können.

3.4.1 TEST 1

Durchführung	Jeder existierende Benutzer muss sich einloggen können und dann die jeweiligen Rechte (Administrator, Editor, User) besitzen. Danach muss er sich wieder ausloggen können.
Testergebnis	OK
Testperson	Daniel Imhof
Unterschrift	

3.4.2 TEST 2

Durchführung	Die Funktionen Hinzufügen, Bearbeiten, Löschen müssen unter jeder Rubrik funktionieren.
Testergebnis	OK
Testperson	Daniel Imhof
Unterschrift	

3.4.3 TEST 3

Durchführung	Alle erforderlichen Parameter einer Datenbank werden beim Hinzufügen eines Falles (Episode of care) korrekt aufgelistet und gespeichert.
Testergebnis	OK
Testperson	Daniel Imhof
Unterschrift	

3.4.4 TEST 4

Durchführung	Beim Hinzufügen, Bearbeiten, Löschen und Verschieben eines Datenbank-Parameters müssen die Parameter sowohl für die Datenbank als auch für alle Fälle (Episodes of care) dieser Datenbank korrekt übernommen werden.
Testergebnis	OK
Testperson	Daniel Imhof
Unterschrift	

3.4.5 TEST 5

Durchführung	Bei einem Fall (Episode of care) müssen ein einzelnes File sowie ein ZIP-File korrekt hochgeladen werden können.
Testergebnis	OK
Testperson	Daniel Imhof
Unterschrift	

3.4.6 TEST 6

Durchführung	Eine komplette Datenbank mit allen zugehörigen Daten muss in ein SQL-File exportiert werden können.
Testergebnis	OK
Testperson	Daniel Imhof
Unterschrift	

4 SCHLUSSFOLGERUNG

In diesem Kapitel geht es darum, das entwickelte Projekt zu analysieren, um zu erfahren, ob alles planmässig und vollständig realisiert wurde.

Zudem gibt es noch ein Unterkapitel für die Weiterentwicklung des Projektes sowie ein Unterkapitel, in welchem der Student seine persönlichen Erfahrungen beschreibt.

4.1 ANALYSE

Im ersten Teil dieses Kapitels wird der Ablauf der Projektarbeit und im zweiten Teil die Zielsetzung analysiert.

4.1.1 PLANUNG

Hinweis: Die Soll-Planung, welche am Anfang des Projekts erstellt wurde, sowie der tatsächliche Ablauf des Projektes (Ist-Planung) finden Sie übrigens im Anhang unter den entsprechenden Kapiteln.

4.1.1.1 DIFFERENZEN ZWISCHEN SOLL- UND IST-PLANUNG

Ich versuchte mich durchgehend an den Arbeitsplan zu halten. Dies habe ich auch mehr oder weniger geschafft. Jedoch habe ich die ersten Wochen sehr viel am Projekt gearbeitet, mehr als eigentlich vorgesehen. Ich war also die ersten Wochen ziemlich im Vorsprung. Trotzdem habe ich die Arbeiten immer der Reihe nach gemäss Arbeitsplan durchgeführt.

Den Zeitaufwand für das Webinterface habe ich unterschätzt. Ich musste fast nur Zeit in die Erstellung des Webinterfaces investieren. Die Datenbank sowie die Visualisierung der Daten und Statistiken nahmen kaum den geplanten Zeitaufwand in Anspruch.

Auch für die Dokumentation habe ich pro Woche zu viel Zeit eingeplant. An der Dokumentation habe ich anfangs für die Struktur und die letzten zwei Wochen intensiv gearbeitet. Ansonsten arbeitete ich fast ausschliesslich an der Programmierung des Webinterfaces.

4.1.2 ZIELE

Im Folgenden sind die Bereiche genauso unterteilt wie im Pflichtenheft (Siehe *Pflichtenheft* im Anhang).

Hinweis: Die kursiv geschriebenen sowie heller eingefärbten Punkte wurden jeweils als optionale Ziele im Pflichtenheft definiert.

4.1.2.1 ANALYSE DER TECHNOLOGIEN

Für die 3 Teilbereiche (Datenbank, Webinterface, Visualisierung der Daten und Statistiken) die aktuellen Technologien vergleichen und sich für jeweils eine mit einer detaillierten Begründung entscheiden.	OK

Beide zentralen Bereiche (Datenbank und Webinterface) wurden analysiert. Für den Bereich *Statistik* war kaum etwas Nützliches im Internet zu finden. Somit brauchte es hierfür keine Entscheidung.

4.1.2.2 DATENBANK

Alle erforderlichen Parameter sowie Bilder müssen in eine Datenbank abgespeichert werden können.	OK
In der Datenbank werden zusätzlich noch die Benutzer und deren Rechte verwaltet.	OK
Die Datenbank sollte angepasst werden können, d.h., neue Datentypen können hinzugefügt werden, sodass mehrere unterschiedliche Datensätze gleichzeitig in einer einheitlichen Oberfläche verwaltet werden.	OK

Alle definierten Funktionen der Datenbank wurden realisiert.

4.1.2.3 WEBINTERFACE

Um die oben erwähnten Parameter in die Datenbank speichern zu können, wird ein Webinterface für die Verwaltung der Datenbank erstellt.	OK
Die Benutzer müssen sich anfangs registrieren, der Administrator kann dann verschiedene Rechte zuweisen, um dann Daten zu verwalten.	OK
Interaktivität sollte gewährleistet werden, damit es benutzerfreundlicher ist.	OK
Anzeige von Thumbnails der Bilder oder einfacher 3D-Ansichten.	
Mobile-Interface für die Dateneingabe und Visualisierung.	OK
Einige Daten können z.B. auch durch einen Klick auf eine bestimmte Position eines Bildes in die Datenbank eingetragen werden, z.B. durch ein anatomisches Interface.	

Das Webinterface entspricht den obligatorischen Vorgaben des Pflichtenhefts. Zusätzlich wurde das als optional definierte mobile Webinterface für die Datenerfassung und Visualisierung realisiert.

Ausserdem wurde noch die sehr nützliche Export-Funktion programmiert, obwohl diese Funktionalität anfangs im Pflichtenheft nicht aufgeführt wurde.

4.1.2.4 VISUALISIERUNG DER DATEN UND STATISTIKEN

Aus den gewünschten Daten der Datenbank müssen einfache Statistiken generiert werden können.	OK
Zusätzliche zu den einfachen Statistiken können komplexe Statistiken sowie einfache grafische Darstellungen generiert werden.	OK
3D-Darstellung der Bilder in den Statistiken.	
Bei der Eingabe eines neuen Falls werden ähnliche Fälle aus der Vergangenheit angezeigt.	

Einfache Darstellungen der Parameter einer Datenbank können gegenübergestellt in einer Statistik dargestellt werden.

4.2 WEITERENTWICKLUNG

Folgende Funktionalitäten könnten noch entwickelt werden:

- Thumbnail und Grossansicht für DICOM-Bilder
- File-Upload mit Prozentanzeige des Upload-Verlaufs
- Zusätzliche Statistiken

4.3 PERSÖNLICHE STELLUNGNAHME

Dieses Projekt hat mir grosse Freude bereitet, da ich gerne Webanwendungen programmiere, vor allem mit den Technologien PHP und MySQL, mit denen ich schon einige Projekte realisiert und dadurch viel Erfahrung habe.

Es war auch eine grosse Motivation und Verantwortung, ein so grosses Projekt von Planung zur Realisierung alleine zu entwickeln. Die bisherigen Projekte waren teils Gruppenarbeiten und nicht annähernd in der Grössenordnung dieses Projektes.

Die Zusammenarbeit mit den beiden zuständigen Dozenten war sehr unkompliziert. Wöchentlich hatten wir ein Meeting (Siehe *Sitzungsprotokolle* im Anhang) wo wir die erledigten und nächsten Aufgaben besprachen. Zusätzlich wurde ich immer gefragt, ob es Probleme gibt. Ansonsten konnten wir die Woche durch E-Mails miteinander austauschen, falls es Bemerkungen, Fragen oder Probleme gab.

Mit dem Resultat bin ich ganz zufrieden. Dadurch, dass ich anfangs mehr Zeit investierte und dadurch im Vorsprung war, hatte ich gegen Ende keine Zeitprobleme und konnte mich der Dokumentation widmen. Diese stellte für mich auch kein Problem dar, da ich mir die Dokumentation vom letzten Jahr anschauen konnte. Zudem haben mir meine zuständigen Dozenten immer wieder Tipps gegeben und mir somit sehr geholfen.

Das Projekt konnte somit ohne grössere Schwierigkeiten im geplanten Zeitraum realisiert werden.

5 LITERATURVERZEICHNIS

[1.] Depeursinge Adrien, Vargas Alejandro, Platon Alexandra, Geissbuhler Antoine, Poletti Pierre-Alexandre and Müller Henning, *Building a Reference Multimedia Database for Interstitial Lung Diseases, Computerized Medical Imaging and Graphics*, page 1-35, 2011.

[2.] Depeursinge Adrien, Zrimec Tatjana, Busayarat Sata and Müller Henning, *3D Lung Image Retrieval Using Localized Features, Medical Imaging 2001: Computer-Aided Diagnosis*, page 1-14, 2011.

[3.] Hugh E. Williams and David Lan, *O'REILLY, Webdatenbank-Applikationen mit PHP und MySQL*, page 1-856, 2004.

[4.] Rasmus Lerdorf, Kevin Tatroe and Peter MacIntyre, *O'REILLY, Programmieren mit PHP*, page 1-553, 2007.

[5.] George Reese, Randy Jay Yarger and Tim King, *O'REILLY: MySQL Einsatz und Programmierung*, page 3-446, 2003.

6 ANHANG

6.1 AUFGABENSTELLUNG

HES-SO Valais

EE	IG	EST
X	X	

Données du travail de bachelor
Daten der Bachelorarbeit

FO.2.2.02.18.BB
stt/07/06/2010

Filière / *Studiengang* : Informatique de gestion

Confidentiel / *Vertraulich* ☐

Etudiant / *Student* Daniel Imhof	**Année / *Jahr*** 2011
Proposé par / *vorgeschlagen von* Henning Müller	**Lieu d'exécution / *Ausführungsort*** Sierre

Titre / *Titel*:

Multidimensional clinical data acquisition

Description / *Beschreibung*:

This bacheor thesis is situated in the context of the MANY (Medical image retrieval in mANY dimensions), funded by the Swiss national science foundation. Several data sets including 3D and 4D data qill be acquire in this projet. For each data set , image data, textual and other clinical parameteres need to be acquired and need to be stored in a database. Based on the data set, the database needs to be configurable to add or remove data types.

The bachelor thesis includes the creation of a configurable database that can be used for acquiring several clinical datasets including images. The databased need to be easy to modify and configure for varying data types. The database includes structured data, free text and images and needs to have a web-based interface for data acquisition. Potentially a 3D annotation tool can be integrated. A simple interface is also necessary to visualize data sets, and to have an automatic quality control. The tools should also include simple statistical analyses of the acquired data.

Expected results:

- web interface for the acquisition of medical data including images and clinical parametres;
- adaptation of the interface for several existing and to be created data sets;
- data export functions based on a variety of parametres;
- automatic functions for quality control of the data..

Objectifs / *Ziele*:

— Creation of a configurable database to store varying data types including images and clinical data, among others using the DICOM format;
— Creation of a simple web interface to allow simple data additions;
— Simple visualizations of the data.

Signature ou visa / *Unterschrift oder Visum*	**Délais / *Termine***
Resp. de la filière Informatique de gestion ..... Professeur/*Dozent*: Henning Müller ..... Etudiant/*Student*:	Attribution du thème / Ausgabe des Auftrags: Remise du rapport / Abgabe des Schlussberichts: Exposition publique /Ausstellung Bachelorarbeiten

Un travail de bachelor, non confidentiel, ayant obtenu une note supérieure ou égale à cinq [Evaluation C] est publié dans son intégralité (sauf annexes pour la filière économie d'entreprise et informatique de gestion) à la médiathèque de la HES-SO Valais [RERO DOC].

Nicht vertrauliche Bachelorarbeiten, die mindestens die Note 5.0 erzielt haben (Note C), werden vollständig (ohne Anhänge für die Studiengänge Betriebsökonomie und Wirtschaftsinformatik) in der Mediathek der HES-SO Wallis (RERO DOC) veröffentlicht.

Rapport reçu le / *Schlussbericht erhalten am* ..... Visa du secrétariat / *Visum des Sekretariats:* .....

6.2 PFLICHTENHEFT

Hes·so VALAIS WALLIS
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences Western Switzerland

Bachelorarbeit

Student : Daniel Imhof
Dozent : Henning Müller

Multidimensional clinical data acquisition

Pflichtenheft

EINLEITUNG

Das Ziel dieser Projektarbeit ist es, ein flexibles Webinterface zur Verwaltung von klinischen Daten zu erstellen, diese Daten grafisch darzustellen und daraus verschiedene Statistiken zu generieren.

FUNKTIONALITÄTEN

Die Funktionalitäten können in folgende 4 Bereiche eingeteilt werden:

1. Analyse der Technologien

Obligatoriche Funktionalitäten :

- Für die 3 Teilbereiche (Datenbank, Webinterface, Visualisierung der Daten und Statistiken) die aktuellen Technologien vergleichen und sich für jeweils eine mit einer detaillierten Begründung entscheiden.

2. Datenbank

Obligatorische Funktionalitäten :

- Alle erforderlichen Parameter sowie Bilder müssen in eine Datenbank abgespeichert werden können.
- In der Datenbank werden zusätzlich noch die Benutzer und deren Rechte verwaltet.
- Die Datenbank sollte angepasst werden können, d.h., neue Datentypen können hinzugefügt werden, sodass mehrere unterschiedliche Datensätze gleichzeitig in einer einheitlichen Oberfläche verwaltet werden.

3. Webinterface

Obligatorische Funktionalitäten :

- Um die oben erwähnten Parameter in die Datenbank speichern zu können, wird ein Webinterface für die Verwaltung der Datenbank erstellt.
- Die Benutzer müssen sich anfangs registrieren, der Administrator kann dann verschiedene Rechte zuweisen, um dann Daten zu verwalten.
- Interaktivität sollte gewährleistet werden, damit es benutzerfreundlicher ist.

Optionale Funktionalitäten :

- Anzeige von Thumbnails der Bilder oder einfacher 3D-Ansichten.
- Mobile-Interface für die Dateneingabe und Visualisierung.
- Einige Daten können z.B. auch durch einen Klick auf eine bestimmte Position eines Bildes in die Datenbank eingetragen werden, z.B. durch ein anatomisches Interface.

4. Visualisierung der Daten und Statistiken

Obligatorische Funktionalitäten :

- Aus den gewünschten Daten der Datenbank müssen einfache Statistiken generiert werden können.

Optionale Funktionalitäten :

- Zusätzliche zu den einfachen Statistiken können komplexe Statistiken sowie einfache grafische Darstellungen generiert werden.
- 3D-Darstellung der Bilder in den Statistiken.
- Bei der Eingabe eines neuen Falls werden ähnliche Fälle aus der Vergangenheit angezeigt.

Daniel Imhof, 30.05.2011

Unterschriften:

Dozent :	Student :
(Henning Müller)	(Daniel Imhof)

6.3 SOLL-PLANUNG

	Von	Bis	Aufgabe	Zeit [h]		
1	16.05.2011	22.05.2011	Vorlage für die Dokumentation erstellen	10		
			Informationen suchen; Einlesen ins Projekt	4		
			Arbeitsumgebung einrichten (Programme installieren,...)	6	20	
2	23.05.2011	29.05.2011	Arbeitsumgebung einrichten (Programme installieren,...)	4		
			Pflichtenheft erstellen	4		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	20	
3	30.05.2011	05.06.2011	Arbeitsplan erstellen	4		
			Analyse der Technologien	4		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	20	
4	06.06.2011	12.06.2011	Analyse der Technologien	4		
			Datenbank erstellen	4		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	20	
5	13.06.2011	19.06.2011	Arbeiten am Webinterface	8		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	20	
6	04.07.2011	10.07.2011	Arbeiten am Webinterface	30		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	42	
7	11.07.2011	17.07.2011	Arbeiten am Webinterface	30		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	42	
8	18.07.2011	24.07.2011	Arbeiten am Webinterface	30		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	42	
9	25.07.2011	31.07.2011	Arbeiten an der Visualisierung der Daten und Statistiken	30		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	42	
10	01.08.2011	07.08.2011	Arbeiten an der Visualisierung der Daten und Statistiken	30		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	42	
11	08.08.2011	14.08.2011	Arbeiten an der Visualisierung der Daten und Statistiken	10		
			Tests durchführen	4		
			Schlusskontrolle	4		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	8		
			Arbeitsjournal	1	30	340

6.4 IST-PLANUNG

	Von	Bis	Aufgabe	Zeit [h]		
1	16.05.2011	22.05.2011	Vorlage für die Dokumentation erstellt	10		
			Informationen suchen; Einlesen ins Projekt	2		
			Arbeitsumgebung eingerichtet (Programme installiert,...)	6	18	
2	23.05.2011	29.05.2011	Arbeitsumgebung eingerichtet (Programme installiert,...)	4		
			Pflichtenheft erstellt	2		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	2		
			Arbeitsjournal	1	12	
3	30.05.2011	05.06.2011	Pflichtenheft fertiggestellt	2		
			Arbeitsplan erstellt	2		
			Technologien (Web Programmiersprache, Datenbank) analysiert	4		
			Datenbank erstellt	4		
			Am Webinterface gearbeitet	35		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	2		
			Arbeitsjournal	1	53	
4	06.06.2011	12.06.2011	Am Webinterface gearbeitet	15		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	2		
			Arbeitsjournal	1	21	
5	13.06.2011	19.06.2011	Am Webinterface gearbeitet	35		
			Am mobilen Webinterface gearbeitet	5		
			Sitzung	4		
			Sitzungsprotokoll	2		
			Dokumentation	5		
			Arbeitsjournal	1	52	
6	04.07.2011	10.07.2011	Am Webinterface gearbeitet	25		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	2		
			Arbeitsjournal	1	31	
7	11.07.2011	17.07.2011	Am Webinterface gearbeitet	5		
			Am mobilen Webinterface gearbeitet	5		
			Dokumentation	4		
			Arbeitsjournal	1	15	
8	18.07.2011	24.07.2011	Am Webinterface gearbeitet	5		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	4		
			Arbeitsjournal	1	13	
9	25.07.2011	31.07.2011	Am Webinterface gearbeitet	20		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	10		
			Arbeitsjournal	1	34	
10	01.08.2011	07.08.2011	Am Webinterface gearbeitet	20		
			An der Visualisierung der Daten und Statistiken gearbeitet	5		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	17		
			Arbeitsjournal	1	46	
11	08.08.2011	14.08.2011	Am Webinterface gearbeitet	5		
			Tests durchgeführt	2		
			Schlusskontrolle durchgeführt	4		
			Sitzung	2		
			Sitzungsprotokoll	1		
			Dokumentation	20		
			Arbeitsjournal	1	35	330

6.5 SITZUNGSPROTOKOLLE

Die folgenden Sitzungsprotokolle sind nach Datum sortiert. Bei jeder Sitzung wird dann unter *Besprochene Themen* kurz der Inhalt beschrieben sowie die nächsten Schritte unter dem Punkt *Nächste Aufgaben* aufgelistet.

6.5.1 PROTOKOLL VOM 25.05.2011

6.5.1.1 BESPROCHENE THEMEN

- Allgemeine Informationen zur Bachelorarbeit (Ablauf, Sitzungen, Dokumentation, Projekt,...)
- Möglichst einmal pro Woche eine Sitzung
- Bei jeder Sitzung (Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben)

6.5.1.2 NÄCHSTE AUFGABEN

- Pflichtenheft erstellen
- Evtl. Dokumentation vom letzten Jahr durchschauen
- Evtl. PDF-File über DICOM anschauen

6.5.2 PROTOKOLL VOM 30.05.2011

6.5.2.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Pflichtenheft durchgegangen
- Zusätzliche Informationen zur Projektarbeit

6.5.2.2 NÄCHSTE AUFGABEN

- Pflichtenheft fertigstellen
- Arbeitsplan erstellen
- Analyse der Technologien
- Evtl. Datenbankdesign erstellen

6.5.3 PROTOKOLL VOM 06.06.2011

6.5.3.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Funktionalitäten des Webinterfaces

6.5.3.2 NÄCHSTE AUFGABEN

- Google docs einrichten + nötige Dokumente hochladen
- Arbeiten am Webinterface

6.5.4 PROTOKOLL VOM 16.06.2011

6.5.4.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Funktionalitäten des Webinterfaces

6.5.4.2 NÄCHSTE AUFGABEN

- Pflichtenheft dem Studiengangsleiter schicken
- Webinterface überarbeiten
- Struktur der Dokumentation überarbeiten

6.5.5 PROTOKOLL VOM 28.06.2011

6.5.5.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Funktionalitäten des Webinterfaces

6.5.5.2 NÄCHSTE AUFGABEN

- Dokumentation überarbeiten und auf Google docs hochladen
- Listen erstellen (z.B. true/false/unknown,...)
- Daten übers Webinterfache in die Datenbank eintragen
- Image-Upload

6.5.6 PROTOKOLL VOM 04.07.2011

6.5.6.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Funktionalitäten des Webinterfaces

6.5.6.2 NÄCHSTE AUFGABEN

- Webinterface überarbeiten
- Einige Symbole in Buttons umändern
- Radio Buttons erstellen
- Daten übers Webinterface in die Datenbank eintragen
- Image-Upload

6.5.7 PROTOKOLL VOM 11.07.2011

6.5.7.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Funktionalitäten des Webinterfaces

6.5.7.2 NÄCHSTE AUFGABEN

- Symbol „Patients“ in Button „Patients“ umändern
- Arbeiten an der Dokumentation

6.5.8 PROTOKOLL VOM 22.07.2011

6.5.8.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Verbesserungen des Webinterfaces

6.5.8.2 NÄCHSTE AUFGABEN

- Webinterface verbessern

6.5.9 PROTOKOLL VOM 29.07.2011

6.5.9.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben
- Zusammen mit dem Dozenten Testdaten eingegeben
- Verbesserungen des Webinterfaces

6.5.9.2 NÄCHSTE AUFGABEN

- Webinterface verbessern
- Statistiken erstellen

6.5.10 PROTOKOLL VOM 03.08.2011

6.5.10.1 BESPROCHENE THEMEN

- Erledigte Aufgaben; Fragen, Probleme,...; Nächste Aufgaben

6.5.10.2 NÄCHSTE AUFGABEN

- Arbeiten an der Dokumentation
- Image-Files im Browser anzeigen (falls noch Zeit übrig)
- Export-Funktion (falls noch Zeit übrig)

6.5.11 PROTOKOLL VOM 10.08.2011

6.5.11.1 BESPROCHENE THEMEN

- Erledigte Aufgaben.
- Informationen über abschliessende Arbeiten sowie über die Präsentation des Projektes

Info: Am 17.08.2011 wird diese Projektarbeit auf einen Server der HES-SO Wallis kopiert und dort eingesetzt. Zusätzlich muss ich einem Verantwortlichen den PHP-Code erklären, damit er das Webinterface später auch warten kann.

6.5.11.2 NÄCHSTE AUFGABEN

- Dokumentation fertigstellen + Dossier bereitstellen

6.6 BENUTZERHANDBUCH

Das Benutzerhandbuch ist in die Bereich *Nicht-mobiles Webinterface* und *Mobiles Webinterface* eingeteilt, wobei beim mobilen Webinterface nur ein Beispiel gezeigt wird, da alles andere dem nicht-mobilen Webinterface ziemlich ähnelt.

6.6.1 NICHT-MOBILES WEBINTERFACE

Beim Betreten der Webseite erscheint als erstes der Startscreen, wo man sich zuerst einloggen muss, um auf das Webinterface zu gelangen.

6.6.1.1.1 Startscreen

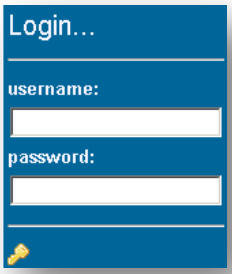


Abb. 18 Benutzerhandbuch Startscreen

Folgende Benutzer-Typen sind möglich:

1. **Administrator:**
   Alle Rechte
2. **Editor:**
   Alle Rechte ausser die Löschfunktion
3. **User:**
   Kann nur Daten anschauen

Bei erfolgreicher Eingabe von *username* und *password* erscheint das eigentliche Webinterface.

6.6.1.1.2 Webinterface

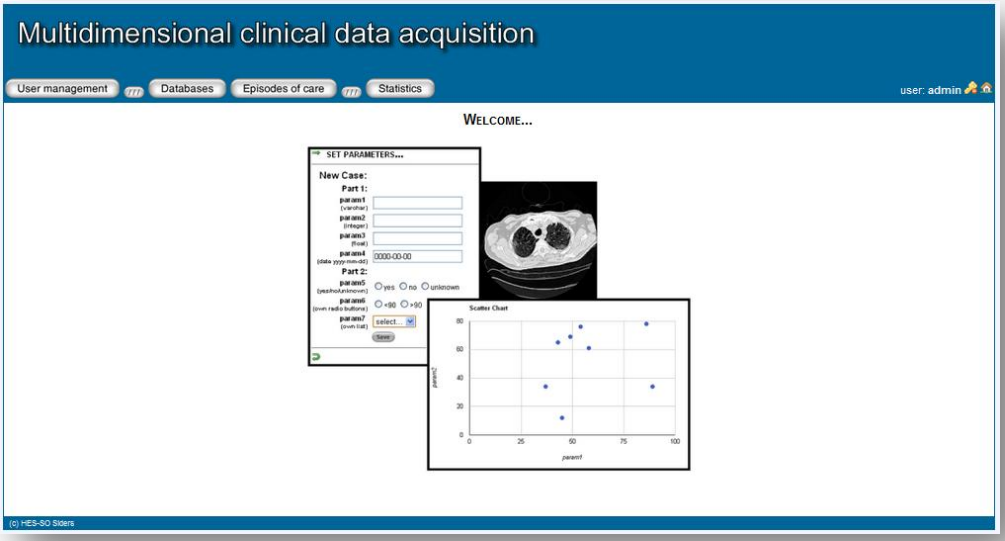


Abb. 19 Benutzerhandbuch Webinterface

Wenn man zu lange untätig auf dem Webinterface ist, dann läuft die Session-ID ab und es kommt folgende Meldung beim Weitersurfen auf dem Webinterface:

failed…

Die Lösung hierfür ist ein erneutes Logout und Login.

6.6.1.1.2.1 Menu

Abb. 20 Benutzerhandbuch Menu

6.6.1.1.3 Benutzer

6.6.1.1.3.1 Hinzufügen, Anzeigen

Im oberen Teil des Fensters kann ein neuer Benutzer hinzugefügt werden, im unteren Teil des Fensters werden alle bestehenden Benutzer aufgelistet.

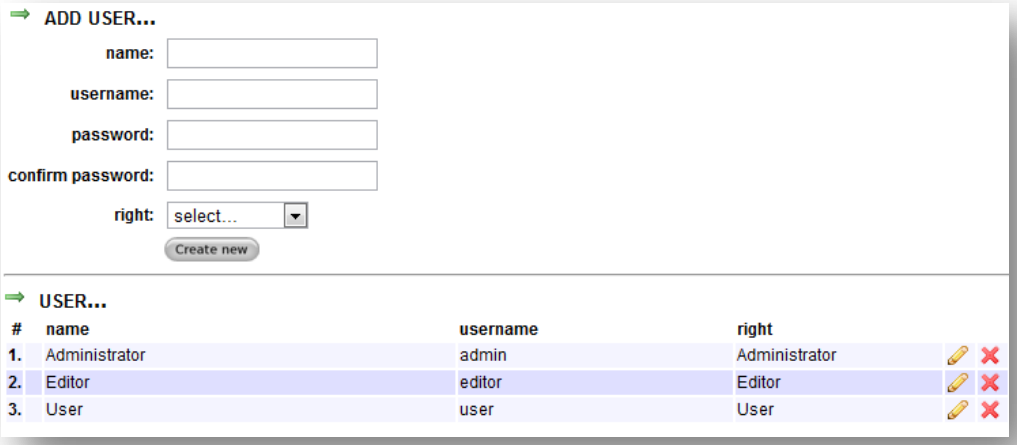


Abb. 21 Benutzerhandbuch User

6.6.1.1.3.2 Bearbeiten

In diesem Fenster kann ein bestehender Benutzer bearbeitet werden.

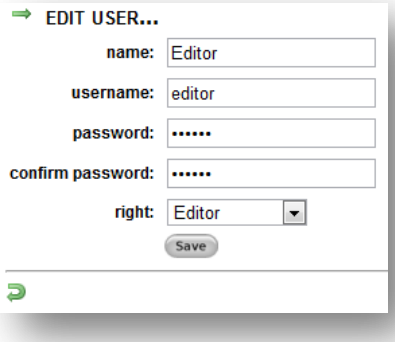


Abb. 22 Benutzerhandbuch bearbeiten

6.6.1.1.4 Datenbanken

6.6.1.1.4.1 Hinzufügen, Anzeigen

Im oberen Teil des Fensters kann eine neue Datenbank hinzugefügt werden, im unteren Teil des Fensters werden alle bestehenden Datenbanken aufgelistet.

Abb. 23 Benutzerhandbuch Datenbanken

6.6.1.1.4.2 Bearbeiten

In diesem Fenster kann eine bestehende Datenbank bearbeitet werden.

Abb. 24 Benutzerhandbuch Datenbank bearbeiten

6.6.1.1.5 Patienten

6.6.1.1.5.1 Hinzufügen, Anzeigen

Zu diesem Fenster gelangt man, falls man bei einer Datenbank auf *Manage patients* klickt.

Im oberen Teil des Fensters kann ein neuer Patient hinzugefügt werden, im unteren Teil des Fensters werden alle bestehenden Patienten aufgelistet.

Abb. 25 Benutzerhandbuch Patienten

6.6.1.1.5.2 Bearbeiten

In diesem Fenster kann ein bestehender Patient bearbeitet werden.

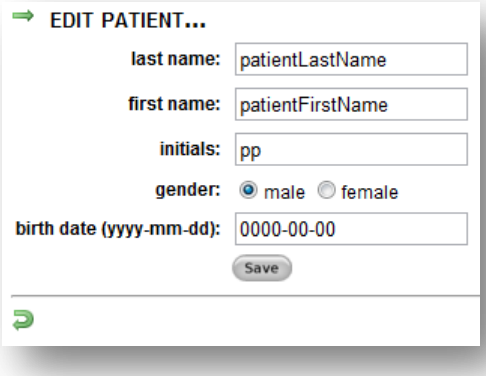


Abb. 26 Benutzerhandbuch Patient bearbeiten

6.6.1.1.6 Datenbank-Parameter

6.6.1.1.6.1 Hinzufügen, Anzeigen

Zu diesem Fenster gelangt man, falls man bei einer Datenbank auf klickt.

Im oberen Teil des Fensters kann ein neuer Parameter hinzugefügt werden, im unteren Teil des Fensters werden alle bestehenden Parameter aufgelistet.

Mit und kann ein Datensatz nach oben bzw. nach unten verschoben werden. Auf diese Weise kann man sich eine beliebige Reihenfolge erstellen. Bei allen Fällen (Episodes of care), die zu dieser Datenbank gehören, werden die Parameter auch nach oben bzw. nach unten verschoben. Dasselbe gilt, falls man einen Datenbank-Parameter hinzufügt, bearbeitet oder löscht. So existieren bei allen Fällen (Episodes of care) der gleichen Datenbank einheitlich auch dieselben Parameter.

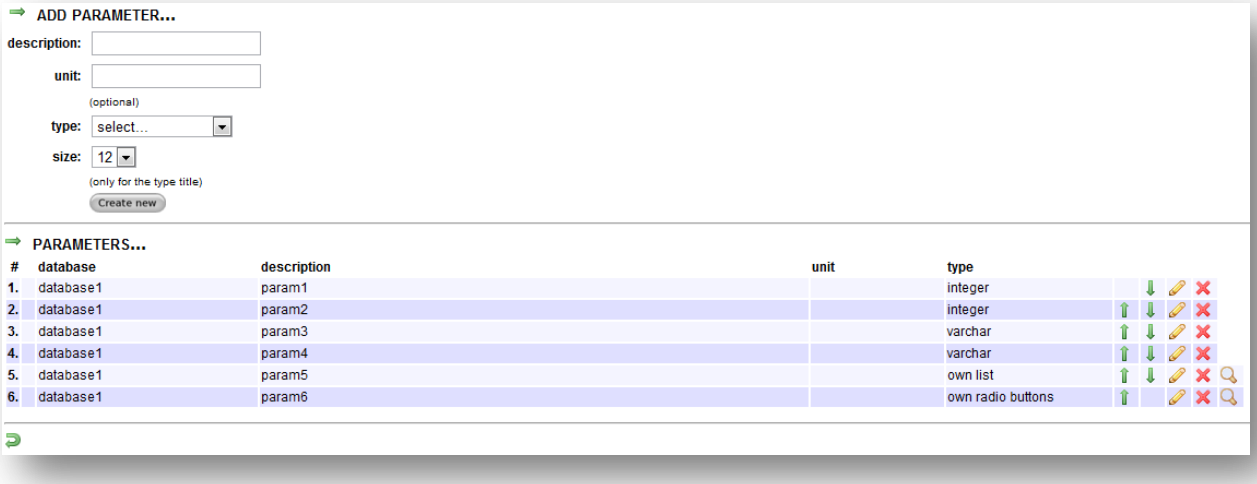


Abb. 27 Benutzerhandbuch Datenbank-Parameter

Folgende Parameter-Typen können hinzugefügt werden:

- varchar
- integer
- float
- date yyyy-mm-dd
- title
- yes/no/unknown
- own radio buttons
- own list
- text

Bei den Parameter-Typen *own radio buttons* und *own lilst* erscheint ganz rechts das Symbol . Hier können alle Werte für Radio Buttons bzw. Listen verwaltet werden.

6.6.1.1.6.2 Bearbeiten

In diesem Fenster kann ein bestehender Datenbank-Parameter bearbeitet werden.

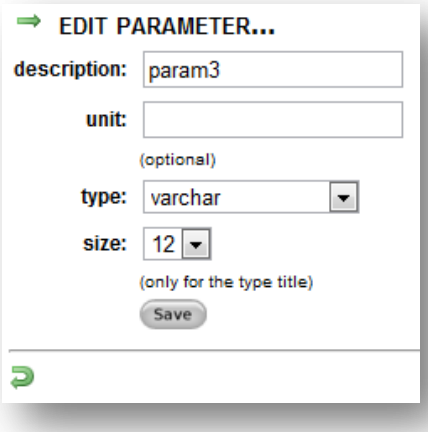


Abb. 28 Benutzerhandbuch Datenbank-Parameter bearbeiten

6.6.1.1.7 Werte für Radio Buttons und Listen

6.6.1.1.7.1 Hinzufügen, Anzeigen

Zu diesem Fenster gelangt man, falls man bei einem Datenbank-Parameter des Typs *own radio buttons* oder *own list* auf klickt.

Im oberen Teil des Fensters kann ein neuer Wert hinzugefügt werden, im unteren Teil des Fensters werden alle bestehenden Werte aufgelistet.

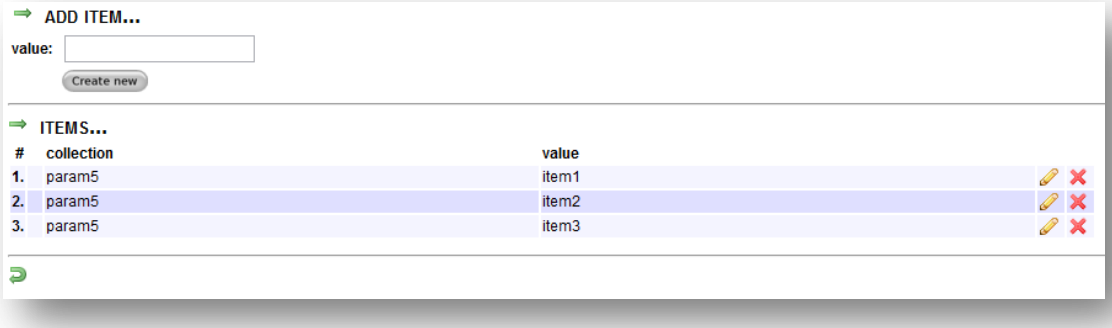


Abb. 29 Benutzerhandbuch Werte für Radio Buttons/Listen

6.6.1.1.7.2 Bearbeiten

In diesem Fenster kann ein bestehender Wert bearbeitet werden.

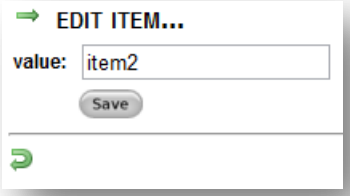


Abb. 30 Benutzerhandbuch Wert für Radio Button/Liste bearbeiten

6.6.1.1.8 Export-Funktion

Falls Sie alle Daten einer Datenbank sichern möchten, können sie bei der gewünschten Datenbank auf klicken. Es wird nun eine SQL-Datei mit allen nötigen Daten geschrieben und zum Download zur Verfügung gestellt. Die SQL-Datei wird dann genauso heissen wie der Name der Datenbank (z.B. db1.sql).

Natürlich werden aus diesen Tabellen nur jene Daten gesichert, die auch zur ausgewählten Datenbank gehören.

6.6.1.1.9 Episodes of care

6.6.1.1.9.1 Hinzufügen, Anzeigen

Im oberen Teil des Fensters kann ein neuer Fall (Episode of care) hinzugefügt werden, im unteren Teil des Fensters werden alle bestehenden Fälle (Episodes of care) aufgelistet.

Um einen neuen Fall (Episode of care) hinzufügen zu können, muss als erstes die Datenbank ausgewählt werden. Anschliessend muss noch ein Patient ausgewählt und eine Beschreibung des Falls (Episode of care) angegeben werden.

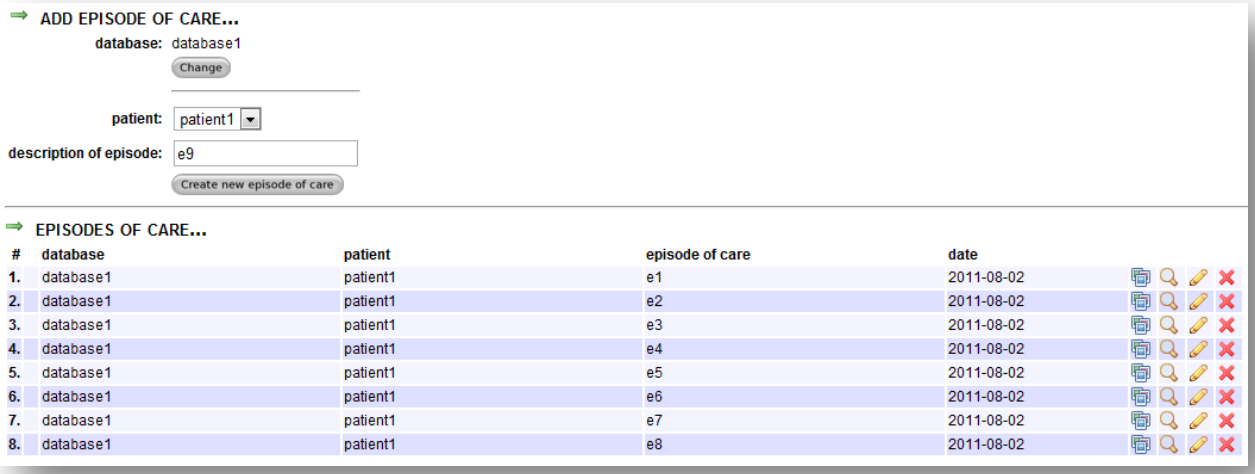


Abb. 31 Benutzerhandbuch Fälle (Episodes of care)

6.6.1.1.9.2 Hinzufügen

Bevor ein Fall (Episode of care) erstellt wird, können noch die erforderlichen Parameter-Werte eingegeben werden. Dies könnte wie folgt aussehen.

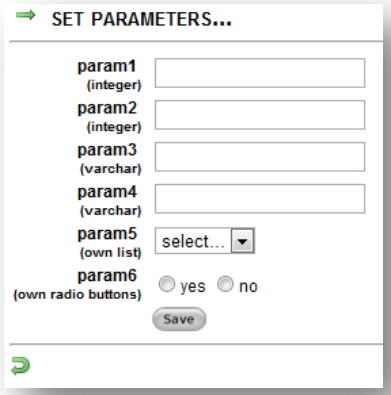


Abb. 32 Benutzerhandbuch Fall-Werte setzen

6.6.1.1.9.3 Bearbeiten

In diesem Fenster kann ein bestehender Fall (Episode of care) bearbeitet werden.

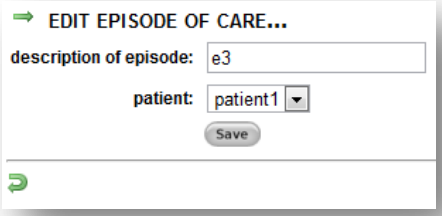


Abb. 33 Benutzerhandbuch Fall (Episode of care) bearbeiten

6.6.1.1.10 Parameter eines Falls (Episode of care)

6.6.1.1.10.1 Anzeigen

In diesem Fenster werden alle bestehenden Parameter eines Falls (Episode of care) aufgelistet.

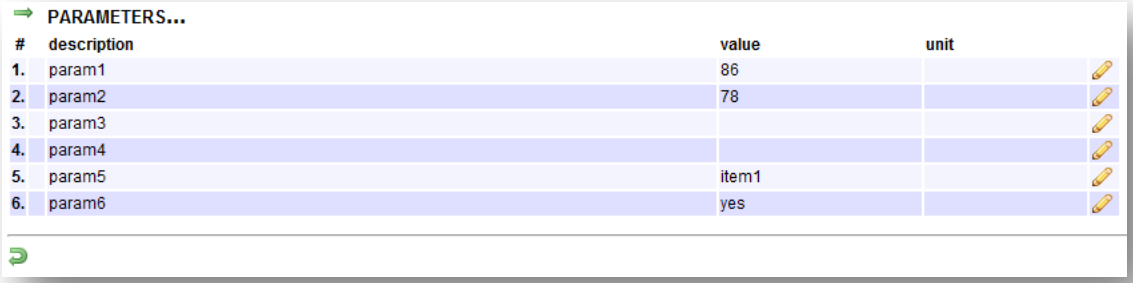


Abb. 34 Benutzerhandbuch Fall-Werte

6.6.1.1.10.2 Bearbeiten

In diesem Fenster kann der Wert eines bestehenden Parameters eines Falls (Episode of care) bearbeitet werden.

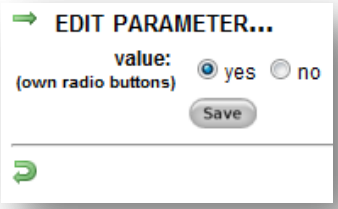


Abb. 35 Benutzerhandbuch Fall-Wert bearbeiten

6.6.1.1.11 File-Upload

6.6.1.1.11.1 Hinzufügen, Anzeigen

Zu diesem Fenster gelangt man, falls man bei einem Fall (Episode of Care) auf das Symbol klickt.

Dieses Fenster ist in zwei Bereiche eingeteilt. Im oberen Teil sind die DICOM-Files und im unteren Teil die Segmentation-Files.

Beide Bereiche sind wiederum in zwei weitere Teile eingeteilt, nämlich in den Bereich zum Hinzufügen von neuen Files und in den Bereich zum Anzeigen der bereits hochgeladenen Files.

Im Bereich der DICOM-Files können sie neben einem Einzel-Upload auch eine ZIP-Datei hochladen, in welcher mehrere DICOM-Bilder enthalten sind. Dadurch müssen Sie nicht jedes Bild einzeln hochladen und sparen Zeit.

Diese hochgeladene ZIP-Datei wird dann nach dem Upload auf dem Webserver entpackt. Bei Funpic ist dies jedoch aufgrund nicht genügender Rechte nicht möglich, ZIP-Dateien auf dem Webserver zu entpacken.

Für den Bereich der DICOM-Files sind nur Dateien des Typs *.dcm* und *.zip* gestattet. Für den Bereich der Segmentation-Files sind alle Typen erlaubt, da diese variieren.

Wenn Sie auf das Symbol eines bestimmten Files klicken, wird es in einem neuen Fenster angezeigt, falls der Browser diesen Datei-Typ unterstützt, ansonsten wird die Datei heruntergeladen.

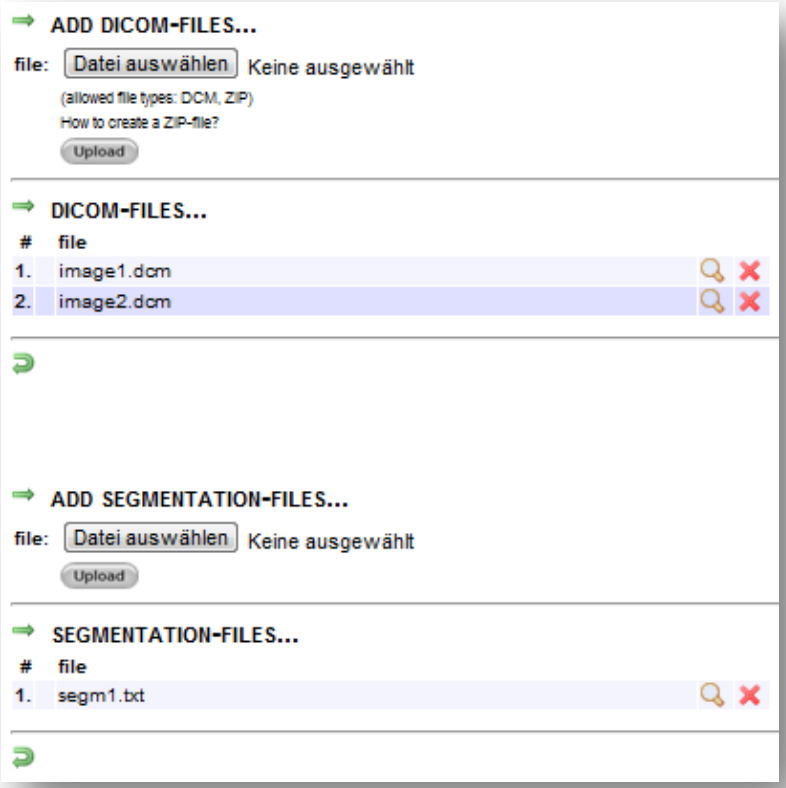


Abb. 36 Benutzerhandbuch Hochgeladene Dateien

6.6.1.1.11.2 ZIP-Datei erstellen

Falls Sie nicht wissen wie eine ZIP-Datei erstellt wird, können sie im Bereich der DICOM-Files auf *How to create a ZIP-File?* klicken und es öffnet sich ein neues Fenster mit einer kurzen Anleitung.

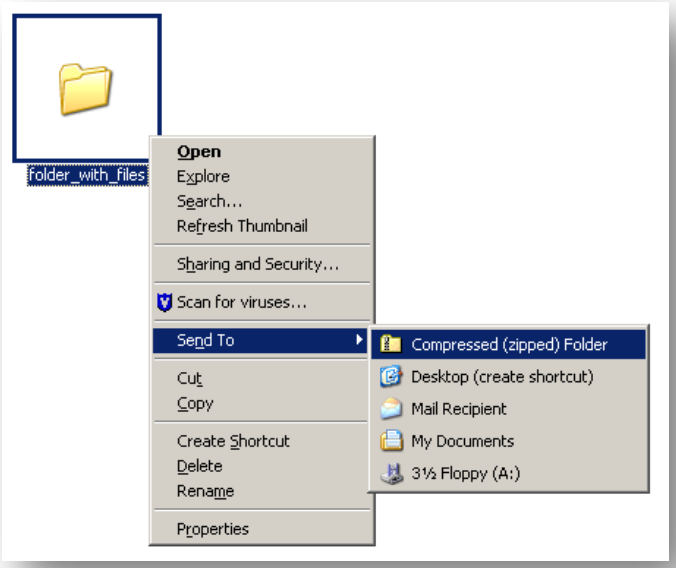


Abb. 37 Benutzerhandbuch ZIP-Datei erstellen

6.6.1.1.12 Statistics

Bevor eine Statistik generiert werden kann, muss eine Datenbank ausgewählt werden mit einem Klick auf das Symbol .

Abb. 38 Benutzerhandbuch Statistik Datenbank auswählen

Anschliessend können alle Datenbank-Parameter der ausgewählten Datenbank des Typs *integer*, *float* und *yes/no/unknown* ausgewählt werden. Es muss immer jeweils ein Parameter für die X- und Y-Achse angegeben werden.

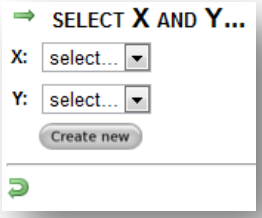


Abb. 39 Benutzerhandbuch Statistik X-Achse und Y-Achse wählen

Eine Statistik könnte dann beispielsweise wie folgt aussehen.

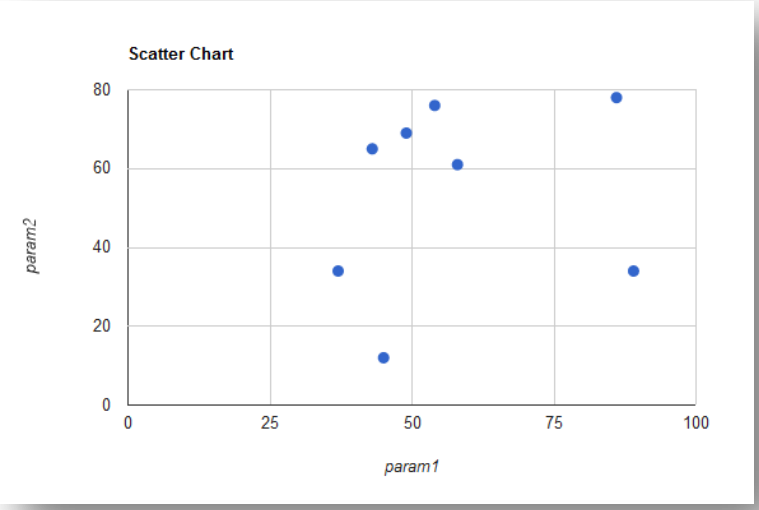


Abb. 40 Benutzerhandbuch Statistik anzeigen

6.6.1.1.13 Symbole

- Login
- Logout
- Startseite
- Zeile nach oben verschieben
- Zeile nach unten verschieben
- Export-Funktion
- File-Upload
- Statistik erstellen
- Details
- Zeile bearbeiten
- Zeile löschen
- Zurück

6.6.2 MOBILES WEBINTERFACE

Beim mobilen Webinterfache erscheint nach erfolgreichem Login auf dem Startscreen folgendes Fenster.

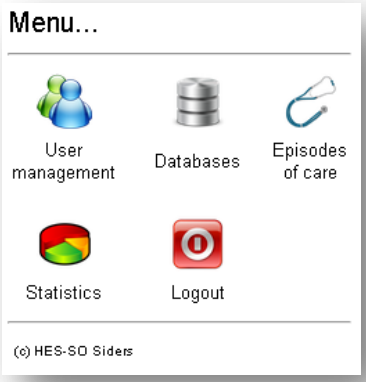


Abb. 41 Benutzerhandbuch Mobiles Webinterface

Alle anderen Fenster sehen ähnlich aus wie beim nicht-mobilen Webinterface, nur dass beim Auflisten der Einträge aus Platzmangel weniger Informationen angezeigt werden.

Zudem können mit dem mobilen Webinterface keine Files hochgeladen sowie Datenbanken exportiert werden.

Die Rubrik *User management* sieht beispielsweise wie folgt aus.

Abb. 42 Benutzerhandbuch Mobiles Webinterface User

Bei einem Klick auf oben links gelangt man zum Menü zurück.

7 INDEX