# Pflichtenheft der SF GmbH
# SEDA-WI-Proj-B

## SF GmbH


| | |
|---|---|
| Denis Hamann | Matr.-Nr. 1684873 |
| Anna Kupfer | Matr.-Nr. 1491515 |
| Hannes Stadler | Matr.-Nr. 1692114 |
| Christian Hindelang | Matr.-Nr. 1685285 |
| Mario Serno | Matr.-Nr. 1687104 |

Otto-Friedrich-Universität Bamberg



Version: v0.5 - 4. November 2012

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# 1 Einleitung

Die Vorliegende Arbeit enthält die Gesamtheit an notwendigen Spezifikationen die uns von der Otto-Friedrich-Universität übermittelt wurden. In diesen Vorgaben, dem Lastenheft, wurden Ergebnisse aus eigens angeführten Ermittlungen zu den individuellen Anforderungen durch Fragebögen, Selbstaufschreibungen sowie Feldbeoachtungen [Balz09, S.303] zur aktuellen und gewünschten Einsatzsituation der neuen Software UniVis 2.0 zusammengestellt. Nach der ausführlichen Sichtung aller Unterlagen und einer ersten Vorstudie der Anforderungen konnte dieses Pflichtenheft erstellt werden. In diesem finden Sie jegliche fachliche Definitionen und Anforderungen, die im Zusammenhang mit der gewünschten Software UniVis 2.0 und den gewünschten Funktionen und Leistungen stehen.

Aus diesem Grund werden zunächst die unterschiedlichen Zielbestimmungen, der präzise Produkteinsatz sowie die gesamte Umgebung (Soft-, Hard-, Orgware und Produkt-Schnittstellen) erläutert. Dem folgen detaillierte Spezifikationen zu den Funktionen, Daten und Leistungen. Um erste Eindrücke zu sammeln und somit zu vermitteln werden Anforderungen an die Benutzeroberfläche sowie erste Prototyp-User-Interfaces vorgestellt. Das Pflichtenheft schließt mit qualitätsbezogenen Zielbestimmungen, globalen Testszeanrien und genauen Angaben zur Entwicklungsumgebung. Weiter Ergänzungen sowie ein Glossar dienen der Komplettierung und Vermittlung einer besseren Verständlichkeit des vorliegende Dokuments.

Die Anforderungensnummerierung setzt sich zusammen aus einem Buchstaben für den jeweiligen Spezifikationsbereich und eine abschnittsbezogene Nummerierung (Bsp.: Funktionen werde mit F abgekürzt und die erste Anforderung entspricht der Nummerierung /F10/). Bei abdingbaren Anforderungen wird ein W zwischen dem bezeichnenden Buchstaben und der Nummerierung eingeführt (Bsp.: /FW10/).

## 1.1 Definitionen

In diesem Abschnitt werden Abkürzungen und Begrifflichkeiten erläutert, die im Pflichtenheft verwendet werden.

/X0/ Eine derartige Kennzeichnung wird im Pflichtenheft für die Kennzeichnung von Software-Merkmalen verwendet. Anstelle des "Xßteht die Abkürzung des Merkmals. SZ"kennzeichnet Zielbestimmungen, "F"Funktionen, "D"Daten, "L"Leistungen, "B"die Benutzeroberfläche, "Q"qualitative Bestimmungen und "T"Testszenarien. Anstelle der "0ßteht die Nummer des jeweiligen Merkmals.

/XW0/ Diese Kennzeichnung entspricht der wie sie bei "/X0/"beschrieben ist, das zusätzliche "Wßignalisiert allerdings, dass es sich um ein wünschenswertes Merkmal handelt, dass je nach Entwicklungsaufwand und Dringlichkeit ggf. nicht in der ersten Version der Software implementiert sein wird.

# 2 Zielbestimmung

## 2.1 Musskriterien

/Z10/ Verwalten der Lehrveranstaltungen und Räume.

/Z20/ Koordinieren des Raumbedarfs der Lehrstühle und der Raumplanung durch die Hausverwaltung.

/Z30/ Vorhandensein einer Login-Funktion für Uni- und Hausverwaltung sowie Dozenten.

/Z40/ Erstellen von nutzerindividuellen Wochenplänen für den Universitätsstandort Erba (Ausgabe i. F. einer PDF).

/Z50/ Möglichkeit der Erweiterung des Systems um die restlichen Universitätsstandorte, Lehrveranstaltungen und Angehörige der Universität nach Evaluationszeitraum.

/Z60/ Erlangen eines Überblicks über aktuell laufende und demnächst startende Lehrveranstaltungen.

/Z70/ Erlangen von grundlegenden Informationen zu Räumen und Lehrveranstaltungen.

/Z80/ Eine intuitive Bedienbarkeit wird vorausgesetzt.

## 2.2 Wunschkriterien

(Wünsche an das Produkt, die nicht unabdingbar sind, deren Erfüllung aber so gut wie möglich angestrebt werden sollte.)

/ZW90/ Anzeige von organisatorischen Änderungen im LiveTicker.

/ZW100/ Vorhandensein eines personalisierten LiveTickers für Dozenten.

## 2.3 Abgrenzungskriterien

/Z110/ Zunächst keine personalisierter Login für Studenten.

/Z120/ Keine Speicherung der von Studenten erstellten Wochenpläne über die Dauer der Systemnutzung hinaus.

# 3 Produkteinsatz

Das Produkt dient der Verwaltung von Räumen und Lehrveranstaltungen am Standort ERBA der Otto-Friedrich-Universität Bamberg. Insbesondere soll der Raumbedarf der Lehrstühle und die Raumplanung durch die Hausverwaltung effektiv koordiniert werden. Des Weiteren soll den Studenten eine Möglichkeit geboten werden, sich einen Stundenplan zu erstellen.

## 3.1 Anwendungsbereiche

Das Produkt wird an der Otto-Friedrich-Universität Bamberg intern eingesetzt.

## 3.2 Zielgruppen

Zielgruppe des Produktes ist der Lehrstuhl für Systementwicklung und Datenbankanwendung der Otto-Friedrich-Universität Bamberg.

## 3.3 Betriebsbedingungen

Das Produkt wird auf Clients der Hausverwaltung, Lehrstühle und PC-Pools ausgeführt. Es ist so zu konzipieren, dass die einzelnen Anwendungsinstanzen über eine gemeinsame Datenbasis (PostgreSQL-Server) kommunizieren können. Die tägliche Betriebszeit des Produktes erstreckt werktags jeweils von 08:00 Uhr bis 20:00 Uhr. Eine Nutzung am Wochenende ist nicht vorgesehen. Zu Beginn jedes Semesters wird die Lauffähigkeit des Produktes durch einen Administrator über einen Zeitraum von drei Wochen beobachtet. Nach Ablauf dieses Zeitraums ist ein unbeaufsichtigter Betrieb vorgesehen.

# 4 Produktumgebung

Im folgenden wird die Umgebung beschrieben, in der die spätere Software eingesetzt werden wird.

## 4.1 Software

Die Software beschreibt die vorzuhaltende Software-Umgebung um eine problemlose Ausführung des Programms sicherzustellen.

Als Betriebssystem kommt Windows 7 (Professional) (x64 - 64bit) zum Einsatz. Dieses hat die neusten Updates vorzuweisen und eine gängige aktuelle Antiviren Lösung installiert zu haben.

Als Laufzeitumgebung wird auf die Java Virtual Machine Version 7 von Oracle gesetzt. Hierzu ist das entsprechende Java Runtime Environment in Version 7 vorzuhalten. Es muss sichergestellt werden, dass die entsprechenden Klassenpfade, sowie Systempfade gesetzt sind, sodass der Prozess java als auch javaw im CLI bekannt ist. Darüber hinaus müssen .jar-Dateien mit der JRE verknüpft sein um einen Start der Anwendung per Doppelklick zu gewährleisten. Entsprechende Einstellungen sind notfalls in der Registry vorzunehmen [vgl. Kapitel 12.1].

Die Datenbank welche in Verbindung mit der Software eingesetzt wird, stellt PostgreSQL in Version 9.2.1 dar. Es muss sichergestellt werden, dass die betreffenden PCs welche die neue Univis Software verwenden sollen für die Datenbank freigeschaltet sind (Freigabe der jeweiligen IPs).

Zusätzlich muss sichergestellt werden, dass entsprechende Netzwerk-Einstellungen (Firewalls, Router) eine ordnungsgemäße Verbindung zwischen Anwendungs-PC und Datenbank erlauben.

Das Programm wird auf die Swing-Komponenten von Java setzten, daher ist es notwendig sicher zus tellen, dass eine entsprechende Shell vorhanden ist. Als Oberfläche wird auf die Standard Fensteroberfläche von Windows 7 (shell: explorer.exe) gesetzt. Kiosk-Systeme sind als GUI nicht vorgesehen.

## 4.2 Hardware

Die Software soll auf IBM-Kompatiblen Computern der Intel x86 Architektur ausgeführt werden. Um sicher zu gehen, dass die Anwendung ausreichend schnell ausgeführt wird, werden nachfolgende Systemvoraussetzungen empfohlen:

CPU: 1Ghz
RAM: 1GB
HDD: >100MB
Peripherie: Maus & Tastatur

Zusätzlich muss sichergestellt werden, dass ein entsprechender Netzwerkanschluss für die Verbindung zum Datenbankserver vorhanden ist.

## 4.3 Orgware

Für die Entwicklung der Software wird vorausgesetzt, dass jeweils ein Mitarbeiter der Fachabteilungen zur Beantwortung von fachspezifischen Fragen zur Verfügung steht. Darüber hinaus ist für Arbeiten vor Ort ein entsprechender Platz zu stellen.

Benutzerhandbücher werden, sofern benötigt, in elektronischer Form (PDF) ausgeliefert. Sie beschreiben die grundlegenden Funktionen der Software.

Der Auftraggeber stellt sicher, dass dem Auftragsnehmer entsprechende Informationen zur Verfügung gestellt werden, um eine ordnungsgemäße Autorisierung und Authentifizierung der Benutzer sicherzustellen.

Zusätzlich wird seitens des Auftraggebers eine entsprechende Netzwerkinfrastruktur bereit gestellt, um eine Verbindung zwischen den Benutzer-PCs und der Datenbank herstellen zu können. Die Datenbank (PostgreSQL) muss auf einem entsprechenden Server installiert sein und als Dienst laufen. Es muss sicher gestellt werden, dass entsprechende Router und Firewalls konfiguriert sind und eine Verbindung in beide Richtungen möglich ist, um eine Persistierung der Daten zu ermöglichen. Es wird davon ausgegangen, dass die Hausverwaltung, sowie die Dozenten entsprechende Informationen zu Räumen und Lehrveranstaltungen korrekt und zeitnah in das System einpflegen um eine sinnvolle Nutzung zu ermöglichen.

## 4.4 Produkt-Schnittstellen

Das Produkt beinhaltet lediglich eine Schnittstelle zur Datenbank wie in Kapitel 4.1 beschrieben. Über diese findet die Persistierung der Daten, sowie die Abfrage komplexer Anfragen statt. Schnittstellen mit der zukünftig geplanten Weboberfläche, sowie Mobilen Anwendung erfolgt nicht über die Software direkt sondern über die gemeinsame Datenbank. Die dort vorliegende Datenstruktur ist allen weiteren Anwendungen bekannt und ermöglicht so eine reibungslose Interaktion zwischen den Systemen. Parallele Sitzungen der Software werden ebenfalls über die Datenbank synchron gehalten. D.h. trägt Benutzer1 an Workstation1 eine neue Lehrveranstaltung ein, so ist diese auch in Workstation2 bei Benutzer2 bei entsprechender Einstellung zu

sehen. Eine direkte Kommunikation zwischen den einzelnen Instanzen der Software findet nicht statt.

# 5 Produktfunktionen

## 5.1 Prozesse mit Dateneingabe

/F01/ **Prozess mit Dateneingabe:** Benutzer-Authentifizierung
**Akteur:** Alle
**Beschreibung:** Alle Benutzer des Systems haben die Möglichkeit sich mittels einer Login-Funktion am System anzumelden. Hierdurch werden Sie einer der möglichen Benutzergruppen zugeordnet. Nicht angemeldete Benutzer gelten als Mitglieder der Benutzergruppe SStudenten".
**Eingabevalidierung:** Nutzer müssen eine gültige Nutzerkennung mit dazu passendem Passwort eingeben. Diese werden von der Verwaltung festgelegt. Wünschenswert wäre, dass Studenten sich auch am System anmelden können (siehe /FW61/) und hierzu ihre normale Hochschulkennung und Passwort verwenden könnten. Im Falle von falschen Zugangsdaten erhält der Benutzer eine Meldung, die ihn über den Fehlversuch informiert. Es gibt nur eine Meldung, sodass für ihn nicht ersichtlich ist, ob das Passwort falsch war oder er schon eine ungültige Nutzerkennung eingegeben hat.

/F10/ **Prozess mit Dateneingabe:** Verwaltung von Raumdaten
**Akteur:** Verwaltung
**Beschreibung:** Mitglieder der Verwaltung können Räume erstellen, bearbeiten und löschen.
**Eingabevalidierung:** Es können beliebige Räume erstellt, gelöscht oder bearbeitet werden. Bei der Erstellung darf die Raumnummer (siehe Datenobjekt /D10/) nicht schon im System vorhanden sein (die systeminterne Raum ID ist für Nutzer nicht selbst wählbar, so wie das auch im Folgenden nie der Fall sein wird), anderenfalls wird der Nutzer mit einer Fehlermeldung darauf hingewiesen. Die übrigen Attribute eines Raums (/D10/) werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen. So kann eine Kapazität beispielsweise nur eine ganze positive Zahl sein.

/F20/ **Prozess mit Dateneingabe:** Verwalten von Lehrveranstaltungen
**Akteur:** Dozenten
**Beschreibung:** Dozenten können Lehrveranstaltungen, die sie oder andere Mitarbeiter ihres Lehrstuhls in einem Semester halten eintragen, bearbeiten und löschen. (Diese sind nur für angehörige ihres Lehrstuhls sichtbar und bearbeitbar. Für Studenten werden sie erst sichtbar und verwendbar, wenn für sie ein Raum gebucht wurde und dies bestätigt wurde.)
**Eingabevalidierung:** Es können beliebige Lehrveranstaltungen erstellt, gelöscht oder bearbeitet werden. Doppelerstellungen sind möglich, falls dies in speziellen Fällen von Dozenten benötigt wird (durch die systeminterne ID ist ein fehlerfreier Systembetrieb gewährleistet). Die übrigen Attribute einer Lehrveranstaltung (/D50/) werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

/FW21/ **Prozess mit Dateneingabe:** Verwalten von Live-Ticker Meldungen
**Akteur:** Dozenten und Verwaltung
**Beschreibung:** Dozenten und Mitglieder der Verwaltung können Meldungen erstellen, die Stundenten im Live-Ticker zu sehen bekommen. Es lässt sich ein Zeitraum eingeben, in dem die Meldung sichtbar ist. Wünschenswert wäre hierbei, dass Studenten ein Profil haben und eingeloggten Studenten nur relevante Meldungen (zu einer Lehrveranstaltung, einem Raum oder einem Lehrstuhl) angezeigt werden.
**Eingabevalidierung:** Es sollten beliebige Live-Ticker Meldungen erstellt, gelöscht oder bearbeitet werden können. Die übrigen Attribute einer Ticker-Meldung (/DW80/) werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

/F30/ **Prozess mit Dateneingabe:** Buchen von Räumen
**Akteur:** Dozenten
**Beschreibung:** Dozenten können ihre Lehrveranstaltung für ein Semester zu einer bestimmten Zeit an einem bestimmten Werktag in einem Raum eintragen. Eine Zeiteinheit beträgt dabei 2 Stunden, kann immer nur voll belegt werden und die Zählung der Einheiten geht von 8 Uhr morgens bis 20 Uhr abends (womit ein Tag aus 6 Einheiten besteht). Buchungen müssen seitens der Verwaltung bestätigt werden (siehe /F40/).
**Eingabevalidierung:** Es sind Buchungen für bereits eingetragene Lehrveranstaltungen möglich. Eine Buchung umfasst genau eine Zeiteinheit zu einem Raum an einem Tag. Es sind beliebig viele Buchungen für eine Lehrveranstaltung möglich. Eine Buchung darf sich nicht mit einer bereits von der Verwaltung bestätigten Buchung im gleichen Raum zur gleichen Zeit am gleichen Tag überschneiden, sonst wird der Vorgang abgebrochen und dem Nutzer ein entsprechender Fehler gezeigt. Die übrigen Attribute einer Buchung (/D60/) werden sofern sie vom Dozenten anzugeben sind und möglich sind nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Dozent darauf hingewiesen.

/F40/ **Prozess mit Dateneingabe:** Verwaltung von Raumbelegungen inklusive Konfliktlösung
**Akteur:** Verwaltung
**Beschreibung:** Mitglieder der Verwaltung können die Belegungswünsche zu Räumen seitens der Dozenten bestätigen, abweisen oder umlegen. Im Falle eines Konflikts, bzw. einer Ablehnung können sie dem Dozenten explizite Vorschläge machen (und hierzu Erläuterungen hinterlegen), die noch vor den vom System generierten automatischen Vorschlägen dem Dozenten angezeigt werden.
**Eingabevalidierung:** Es ist nur das zuweisen eines Status, wie bei /DW80/ spezifiziert, möglich. Dieser ist vom System vorgegeben so dass Falscheingaben nicht möglich sind. Im Falle einer Ablehnung kann zusätzlich eine Nachricht übermittelt werden, die bezüglich Größe validiert wird. Im Fehlerfall wird der Nutzer darauf hingewiesen und der Vorgang abgebrochen.

/F50/ **Prozess mit Dateneingabe:** Verwaltung von Nutzerdaten
**Akteur:** Verwaltung
**Beschreibung:** Mitglieder der Verwaltung können Benutzer (Dozenten und Verwaltungsmitglieder) in das System einpflegen, sie bearbeiten oder löschen.
**Eingabevalidierung:** Das Eintragen und Bearbeiten umfasst die in /D30/ spezifizierten Attribute (außer dem letzten Login und dem Salt, welche automatisch vom System generiert werden und wie gehabt der systeminternen ID). E-Mail und Benutzerkennung dürfen nicht schon im System vorhanden sein, anderenfalls erscheint eine entsprechende Fehlermeldung und der Vorgang wird abgebrochen. Das Passwort wird beim Erstellen und Bearbeiten direkt zur Laufzeit zusammen mit einem benutzerspezifischen zufälligen Wert (dem SSalt") in einen SHA512-Hash umgewandelt und nur als solcher persistent gespeichert. Es muss mindestens 8 Zeichen umfassen. Alle Attribute werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

/F51/ **Prozess mit Dateneingabe:** Verwaltung von Lehrstühlen
**Akteur:** Verwaltung
**Beschreibung:** Mitglieder der Verwaltung können Lehrstühle in das System einpflegen, sie bearbeiten oder löschen.
**Eingabevalidierung:** Das Eintragen und Bearbeiten umfasst die in /D20/ spezifizierten Attribute. Der Lehrstuhlname darf nicht bereits im System vorhanden sein, anderenfalls erscheint eine entsprechende Fehlermeldung und der Vorgang wird abgebrochen. Alle Attribute werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

/F60/ **Prozess mit Dateneingabe:** Belegung von Lehrveranstaltungen
**Akteur:** Studenten
**Beschreibung:** Studenten können sich für Lehrveranstaltungen einschreiben (diese Einschreibungen werden nicht persistent im System gespeichert). Überschneidungen (Belegung zur gleichen Zeit) ist möglich, das System generiert aber einen Warnhinweis.
**Eingabevalidierung:** Das System bietet nur die Möglichkeit eine öffentliche Veranstaltung als hinzuzufügen oder diese wieder zu entfernen. Eine vom Benutzer erstellte Eingabe und somit eine Validierung ist nicht von Nöten. Wünschenswert wäre die persistente Speicherung dieser Zuordnungen wie sie durch die Funktion /FW61/ möglich wäre und im Datenobjekt /DW70/ spezifiziert ist.

/FW61/ **Prozess mit Dateneingabe:** Profil-Verwaltung (inkl. Speichermöglichkeiten von Belegungen)
**Akteur:** Studenten
**Beschreibung:** Studenten sollten die Möglichkeit bekommen, ein Profil von sich anzulegen, welches persistent gespeichert wird und mit welchem ihre Belegungen dauerhaft verknüpft werden.
**Eingabevalidierung:** Die Profil-Bearbeitung umfasst die Attribute E-Mail, Passwort, Vor- und Nachname des Datenobjekts /D30/ für Studenten. Das Passwort wird dabei wie in /F50/ beschrieben gehasht und muss mindestens 8 Zeichen umfassen. Die E-Mail Adresse darf nicht bereits im System vorhanden sein, anderenfalls erscheint eine entsprechende Fehlermeldung und der Vorgang wird abgebrochen. Alle betreffenden Attribute werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

## 5.2 Listen

/F70/ **Liste:** Lehrveranstaltungen eines Lehrstuhls
**Akteur:** Dozenten
**Beschreibung:** Dozenten können sich im System eine Liste von allen Lehrveranstaltungen, die ihrem Lehrstuhl zugeordnet sind anzeigen lassen. Hierbei steht ein Filter zur Verfügung, der nur bestimmte Semester oder bestimmte Dozenten anzeigt.
**Enthaltene Daten:** Name der Veranstaltung, Dozent, Semester, Tag(e), Zeit(en), gebuchter Raum, Freigabestatus, *wünschenswert eingeschriebene Studenten*

/F80/ **Liste:** Buchungen und Buchungswünsche
**Akteur:** Verwaltung
**Beschreibung:** Mitglieder der Verwaltung können sich im System eine Liste mit allen gebuchten Räumen, sowie noch nicht freigegebenen oder abgelehnten gebuchten Räumen anzeigen lassen. Hierbei steht ein Filter zur Verfügung, der es erlaubt nach Semester, Freigabestatus, Tag, Raum oder Lehrstuhl zu filtern. Noch nicht bearbeitete Raumbuchungen (weder freigegeben noch abgelehnt) werden an erster Stelle angezeigt.
**Enthaltene Daten:** Name der Veranstaltung, Dozent, Lehrstuhl, Semester, Tag(e), Zeit(en), gebuchter Raum, Freigabestatus, ggf. Verweis auf überschneidende Buchungen, *wünschenswert eingeschriebene Studenten*

/F90/ **Liste:** Nutzer
**Akteur:** Verwaltung
**Beschreibung:** Mitglieder der Verwaltung können sich im System eine Liste mit allen Benutzern des Systems anzeigen lassen. Es steht ein Filter zur Verfügung, der es erlaubt nach Vor- oder Nachnamen sowie der Nutzerzugehörigkeit zu filtern.
**Enthaltene Daten:** Nutzerkennung, Vorname, Nachname, Nutzerzugehörigkeit (Lehrstuhl, Verwaltung oder ggf. auch Student), Zahl der Lehrveranstaltungen im aktuellen Semester, Datum des letzten Logins

/F100/ **Liste:** Lehrveranstaltungen
**Akteur:** Studenten (und andere)
**Beschreibung:** Studenten (aber auch anderen Nutzern) können sich im System eine Liste aller durch die Verwaltung freigegebenen Lehrveranstaltungen anzeigen lassen. Es steht ein Filter zur Verfügung, der es erlaubt nach Semester, Lehrstuhl, Dozent, Name der Veranstaltung, Tag, Zeit und Raum zu filtern.
**Enthaltene Daten:** Name der Veranstaltung, Dozent, Lehrstuhl, Semester, Tag(e), Zeit(en), gebuchter Raum, *wünschenswert eingeschriebene Studenten*

/F110/ **Liste:** Live-Ticker Meldungen
**Akteur:** Studenten (sowie wünschenswerterweise Verwaltung und Dozenten mit Bearbeitungsfunktionen)
**Beschreibung:** Studenten (aber auch anderen Nutzern) bekommen im System automatisch Live-Ticker Meldungen angezeigt, die die nächsten (falls möglich zu ihren Stundenplänen passenden) Lehrveranstaltungen anzeigen, sowie wünschenswerterweise auch von Dozenten oder Verwaltung zu diesen erstelle Meldungen.
**Enthaltene Daten:** Lehrveranstaltungsname, Raum, Zeit oder Meldungstexte

## 5.3 Berichte

/F120/ **Bericht:** Raumbelegung
**Akteur:** Verwaltung (und wünschenswerterweise andere Nutzer)
**Beschreibung:** Mitglieder der Verwaltung (und wünschenswerterweise auch andere Nutzer) können sich eine Übersicht zur Belegung eines bestimmten Raumes in einem Semester anzeigen lassen, welche als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar ist.

/F130/ **Bericht:** Semesterbelegung (Stundenplan)
**Akteur:** Studenten
**Beschreibung:** Studenten können sich eine Übersicht zu ihren Belegungen in einem Semester anzeigen lassen. Überschneidungen werden hierbei gesondert hervorgehoben. Die Übersicht (die einem Stundenplan gleich kommt) ist als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar.

/F140/ **Bericht:** Lehrstuhlveranstaltungen
**Akteur:** Dozenten
**Beschreibung:** Dozenten können sich eine Übersicht generieren, in der sie sehen, wann alle Lehrveranstaltungen ihres Lehrstuhls in einem bestimmten wählbaren Semester sind. Überschneidungen werden hervorgehoben. Wünschenswert wäre, wenn auch Überschneidungen mit Veranstaltungen anderer Lehrstühle markiert werden. Die Übersicht ist als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar.

/F150/ **Bericht:** Dozentenveranstaltungen
**Akteur:** Dozenten
**Beschreibung:** Dozenten können sich eine Übersicht generieren, in der sie sehen, wann alle ihre Lehrveranstaltungen in einem bestimmten wählbaren Semester sind. Überschneidungen werden hervorgehoben. Wünschenswert wäre, wenn auch Überschneidungen mit Veranstaltungen anderer Dozenten markiert werden. Die Übersicht ist als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar.

# 6 Produktdaten

/D10/ **Datentyp:** Räume
**Attribute:** Raum ID *(systemintern)*, Raumnummer, Gebäude, Stockwerk, Sitzplätze, PC-Plätze, Beamer, Visualizer, Overheads, Tafeln, Whiteboards

/D20/ **Datentyp:** Lehrstühle
**Attribute:** Lehrstuhl ID *(systemintern)*, Lehrstuhlname, Lehrstuhlinhaber, Fakultät, (Haupt-)Gebäude, (Haupt-)Stockwerk

/D30/ **Datentyp:** Benutzer
**Attribute:** Benutzer ID *(systemintern)*, Benutzerkennung, Passwort (Hash), Salt, E-Mail, Benutzerzugehörigkeit (Verwaltung, betreffender Lehrstuhl, wünschenswerterweise auch Student), Vorname, Nachname, letzter Login

/D50/ **Datentyp:** Lehrveranstaltungen
**Attribute:** Veranstaltungs ID *(systemintern)*, Benutzer ID (Dozent), Veranstaltungskurzbezeichnung, Veranstaltungsname, Semester, Benötigte SWS, Art (Vorlesung|Übung|Tutorium), Freigabe durch Dozent, Beschreibung (Tag, Zeiteinheiten, etc. wird über "Raumbelegung (/D60/)ërmittelt, wo für jede Zeiteinheit ein Eintrag erstellt wird und Veranstaltungen mehrere Einträge pro Semester buchen können.)

/D60/ **Datentyp:** Raumbelegungen (aller Freigabestatus-Arten)
**Attribute:** Belegungs ID *(systemintern)*, Veranstaltungs ID (Lehrveranstaltung), Raum ID, Semester, Tag, Zeiteinheit, Freigabestatus (unbearbeite|freigegeben|abgelehtn|gegenvorschlag), Freigabenachricht (Falls ein Vorschlag abgelehnt wurde und dies nun ein reservierter Vorschlag des Status "gegenvorschlagïst), Kommentar

/DW70/ **Datentyp:** Studentenbelegungen
**Attribute:** Studenten-Belegungs ID *(systemintern)*, Benutzer ID (Student), Belegungs ID (freigegebene Lehrveranstaltung)

/DW80/ **Datentyp:** Ticker-Nachricht
**Attribute:** Meldungs ID *(systemintern)*, Meldungstext, Start-Datum, End-Datum, exklusiv für Lehrstuhl ID ('null' wenn für alle), exklusiv für Veranstaltungs ID ('null' wenn für alle), exklusiv für Raum ID ('null' wenn für alle)

In Structured-Entity-Relationship-Modell (SERM) wird der Zusammenhang der oben spezifizierten persistent zu speichernden Daten verdeutlicht.

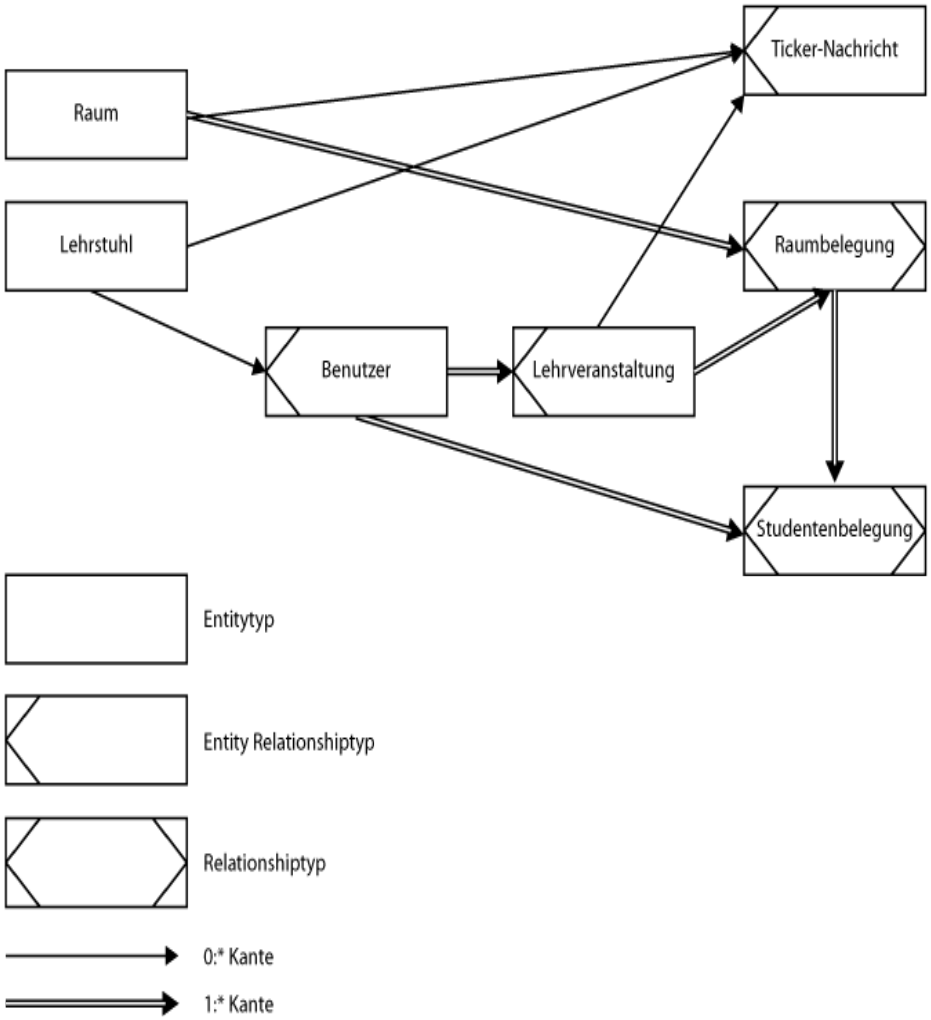


Abbildung 1: SERM der Daten-Architektur

Wie zu sehen ist, existieren sowohl Räume als auch Lehrstühle als eigenständige Einheiten (Entity). Benutzer können einem Lehrstuhl zugeordnet werden, womit sie der Klasse "Dozentëntsprechen. Sie können aber auch der Klasse "Verwaltungöder SStudentängehören und somit keinem Lehrstuhl zugeordnet werden. Lehrveranstaltungen müssen einem Benutzer zugeordnet werden, der (allein an der Grafik nicht erkennbar) ein der Klasse "Dozentängehören muss. Eine Raumbelegung ist keine Einheit sondern eine Zuordnung (Relationship) und muss einer Lehrveranstaltung und einem Raum zugeordnet werden können. Die Zuordnung Studentenbelegung muss einem Benutzer (der Klasse SStudent") und einer Raumbelegung zugeordnet werden können. Ticker-Nachrichten können einem Raum, einer Lehrveranstaltung oder einem Lehrstuhl zugeordnet werden, nichts davon ist aber zwingend.

# 7 Produktleistung

/L10/ Alle Daten, die im vorherigen Abschnitt aufgelistet sind, müssen, sofern sie realisiert werden, persistent mittels einer SQL-Datenbank (erreichbar und verwaltet durch einen PostgreSQL-Server) gespeichert werden.

/L20/ Bei allen Prozess mit Dateneingabe-Funktionen erhalten Nutzer aussagekräftige Fehlermeldungen, sollte ihr Prozess nicht ordnungsgemäß durchgeführt werden können.

/L30/ Die Realisierung erfolgt als Java-Anwendung, so dass ein möglicher Betrieb auf allen gängigen PCs der Universität sichergestellt ist.

/L40/ Die Programmarchitektur richtet sich nach einem Drei-Schichten-Modell, wobei in einer Schicht die Anwendungslogik (Ä-Schicht"), in einer weiteren die nötigen Funktionen und Schnittstellen zur Datenbank-Kommunikation ("D-Schicht") und in einer letzten die für den Nutzer sichtbaren Teile, mit denen er interagiert, ("K-Schicht") untergebracht sind.
Das Programm ist zudem in logische Bereiche (Module) untergliedert, die innerhalb der ADK-Struktur existieren und die sich bei der Implementierung eines GUI (Graphical User Interface) nach dem MVC-Konzept (was für Model, View und Controller steht) richten. So kann ein Modul Views (etwa Fenster, Tabs oder Eingabemasken) besitzen, die innerhalb der "K-Schicht"liegen, innerhalb der Ä-Schichtßind die den Views zugrundeliegenden Models, sowie deren Controller und ggf. zusätzlich nötige Anwendungslogik angesiedelt. Sofern Teile der Datenbank exklusiv bestimmten Modulen zuzuordnen sind, kann sich ein Modul auch bis in "D-Schichtërstrecken und hier eigene Funktionen und Schnittstellen bereitstellen. Diese Architektur erlaubt dabei einerseits maximale Flexibilität in Bezug auf zukünftige Erweiterungen und gewährleistet zudem durch viele klar voneinander abgegrenzte Bereiche ein effektives Arbeiten im Team bei der Implementierung.

Die erste Version der Anwendung soll aus folgenden Modulen bestehen:

**CoreGUI:** Hier wird die Start-Routine sowie der Startbildschirm der Anwendung realisiert. Die Funktion /F01/ und /F60/ werden von diesem Modul umgesetzt.

**Verwaltung:** Hier wird die Anzeige für Verwaltungsmitglieder umgesetzt, die die Funktion /F10/, wünschenswerterweise /FW21/, /F40/, /F50/, /F51/, /F80/ und /F90/ realisiert.

**Dozenten:** Hier wird die Anzeige für Dozenten umgesetzt, die die Funktion /F20/, wünschenswerterweise /FW21/, /F30/, /F140/, /F150/ und /F70/ realisiert.

**Stundenplan:** Hier wird die Anzeige für den Stundenplan von Studenten umgesetzt, die die Funktion /F130/ realisiert.
**Raumplan:** Hier wird die Anzeige für Mitglieder der Verwaltung (und ggf. andere) von Räumen und deren Belegung umgesetzt, die die Funktion /F120/ realisiert.

**Studentenprofil (wünschenswert):** Hier wird die Anzeige für Studenten umgesetzt, die die Funktion /FW61/ realisiert.

Alle Module sind von "CoreGUIäbhängig.

/L50/ Es wird eine Nutzerauthentifizierung realisiert, die es gewährleistet, dass nur berechtigte Nutzer Änderungen durchführen. Um Missbrauch ausschließen zu können, sollte allerdings eine Middleware (die großteils die Model- und Controller-Schicht abdeckt) zwischen Datenbank und Client (der haupsächlich die View-Schicht umsetzt) realisiert werden. In diesem Fall sollte die Kommunikation zwischen Client und Middleware durch SSL (mithilfe von javax.net.ssl.* und javax.rmi.ssl.*) abgesichert werden.

/L60/ Jede Interaktion sollte im Mittel zur Durchführung nicht länger als 3 Sekunden brauchen.

# 8 Benutzeroberfläche

Dieser Abschnitt beschäftigt sich mit der grundlegenden Gestaltung der Benutzeroberfläche für die unterschiedlichen Nutzer. Zunächst

/B10/ Fensterlayout, Dialogstruktur und Mausbedienung entsprechen dem Windows-Gestaltungs-Regelwerk.

/B11/ Sämtliche Daten sind passwortgeschützt und dürfen nur von autorisierten Mitarbeitern des Lehrstuhls bearbeitet werden.

## 8.1 Bildschirmlayout

/B20/ Übersichtliche Gestaltung der Funktionen und intuitive Nutzung durch ein angepasstes Bildschirmlayout.

/B30/ Standardmäßig startet das Programm mit einer Suchmaske. Weitere Funktionen sind via Tabs und einer Anmeldung erreichbar.

/BF40/ Individuelle Anpassungsfähigkeit des Bildschirmlayouts an die Fenstergröße sollte möglich sein.

## 8.2 Drucklayout

/B50/ Nicht Angemeldete Nutzer können über die Auswahl verschiedener Lehrveranstaltungen einen Stundenplan im PDF-Format in Din-A4 Größe erzeugen lassen.

/B60/ Die Hausverwaltungsnutzer können folgende Raumpläne im PFD-Format und Din-A4 Größe erzeugen lassen.

/B70/ Mitarbeiter der Lehrstühle können personen- oder lehrstuhlbezogene Wochenpläne in ein PDF-Format exportieren in Din-A4 Größe.

## 8.3 Tastaturbelegung

/B80/ Die Bedienoberfläche ist auf eine Bedienung mittels Tastatur und Maus auszulegen. Benutzer in der Studentenrolle können keine Eingaben mittels der Tastatur vornehmen. Dozenten und die Hausverwaltung brauchen die Tastatur um verschiedene Funktionen auszuführen.

/BF81/ Mögliche individuelle, nicht dem Windows Standard entsprechende, Tastaturbelegungen sind ein mögliches Wunschkriterium.

## 8.4 Dialogstruktur

/B90/ Zu Beachten ist: ISO 9241-10: 1996 bzgl. der ergonomischen Anforderungen für Bürotätigkeiten mit Bildschirmgeräten, Teil 10: Grundsätze der Dialoggestaltung.

/B91/ Folgende Rollen sind zu unterscheiden:

| Rolle | Rechte |
|---|---|
| Student | (F01), F60 , FW61, F130 |
| Dozent | F01, F20, FW21, F30, F70, F140, F150 |
| Verwaltungsangestellter | F01, F10, FW21, F40, F50, F51, F80, F90, |
| Alle | F01, F100, F110, F120 |

Nachfolgend wird die Dialogstruktur durch den GUI (General User Interface) Prototypen verdeutlicht um die Benutzeroberfläche vorzustellen und zu erläutern.

# 9 Qualitätszielbestimmung

Das zu entwerfende Programm (UnivIS 2.0) hat folgende Qualitätsziele, die nachfolgend erläutert werden.

/Q10/ Das Programm muss eine korrekte Funktionalität aufweisen, d. h. es muss richtige Ergebnisse bei Eingaben und Ausgaben liefern.

/Q20/ Es muss gewährleistet sein, dass die Daten sicher und vom Zugriff unberechtigter geschützt sind.

/Q30/ Die Software muss zuverlässig arbeiten. Es dürfen möglichst keine Fehler passieren.

/Q40/ Die Datenrückgewinnung muss gewährleistet sein, wenn der Server abstürzt.

/Q50/ Eine leichte und intuitive Bedienung von UniVis 2.0 ist von Nöten, da eine Vielzahl von potenziellen Nutzern auf das System zugreifen soll. Die Heterogenität der Studenten und des Personals macht es erforderlich, dass sich alle Beteiligten schnell an die Bedienung des Programms gewöhnen und keine langen Einarbeitungsphasen oder sogar Kurse notwendig werden.

/Q60/ Damit die Nutzer des Systems effizient arbeiten können, darf im Mittel die Ergebnisausgabe eines Stundenplans zwei Sekunden in Anspruch nehmen.

/Q60/ Die im Lastenheft gemachte Angabe, dass das System eventuell universitätsweit und auf mobilen Plattformen zum Einsatz kommen kann, macht es erforderlich, dass die Änderbarkeit und Anpassung des Programms schnell erfolgen kann. Maximal drei Monate sollten hierfür aufgewendet werden müssen.

/Q60/ Schlussendlich muss das Programm an andere Windowsversionen anpassbar sein. Dies ist gewährleistet, da eine JAVA-Datei verwendet wird, welche plattformunabhängig läuft.

# 10 Testszenarien

Damit festgestellt werden kann, ob die Software auch einwandfrei funktioniert, müssen verschiedene Testszenarien durchlaufen werden. Die Tests werden auf verschiedenen Ebenen ausgeführt. Unterschieden wird hierbei zwischen dem Modultest, Integrationstest, Systemtest und letztendlich dem Abnahmetest. Im Folgenden werden nun diese Testszenarien genauer beschrieben und dargelegt, wie sie am Programm durchgeführt werden. //

/T10/ Im Komponententest, welcher wie auc
Box-Tests fällt, beinhaltet das Testen
des Gesamtprogramms. Das Testen e
lungsumgebung Eclipse, wo der Quel
tionen und Unterprogramme werden i
worfen und auf Fehlerfreiheit geprüft
Ausgabe von Uhrzeiten. Auch SQL-A
tabfrage korrekt abgespeichert wurde

/T20/ Nachdem im Modultest jedes Modul einzeln überprüft wurde, wird nun im Integrationstest dazu übergegangen, auch die Interaktion einzelner Module mit anderen Modulen zu überprüfen. Hier soll u. a. getestet werden, ob der Live-Ticker auch alle Lehrveranstaltungen anzeigt, die in Kürze starten. Zudem soll geschaut werden, ob die Stundenplanausgabe die richtigen Inhalte ausgibt und diese anschließend in der PDF korrekt formatiert ausgegeben werden.

/T30/ Die dritte Teststufe beherbergt den Systemtest, bei dem das System gegen sämtliche Anforderungen aus dem Pflichtenheft getestet wird. Ab hier erfolgt das Testen nicht mehr in der Entwicklungsumgebung, sondern wird direkt am lauffähigem Programm ausgeführt. Dies wird auch als Black-Box Test bezeichnet. Hier werden nun 200 Lehrveranstaltungen zu Testzwecken angelegt. Diese werden vom Systemverwalter manipuliert und auch die Suche nach Alternativterminen für Veranstaltungen wird hierbei überprüft. Auch wird geschaut, ob die persistente Speicherung der Stammdaten (z. B. Name und Uhrzeit der Lehrveranstaltung) funktioniert. Hierfür wird ein Systemabsturz simuliert, bei dem aufeinanderfolgend erst dem Rechner, dann dem Server der Strom entzogen wird und beide somit neu booten müssen. Anschließend wird eine neue Datenbankabfrage ausgehend des Systems gemacht, um zu sehen, ob die Daten noch vorhanden sind.

/T40/ Schlussendlich erfolgt der Abnahmetest. Dieser wird allerdings nicht von unserem Softwareunternehmen ausgeführt, sondern erfolgt durch den Abnehmer selbst. Durch den tatsächlichen Gebrauch können hierbei noch Fehler entdeckt werden, die bislang nicht auftauchten, da unter Umständen eine andere Reihenfolge der Menüaufrufe, etc. stattfindet. Der Endabnehmer muss die Software akzeptieren, damit die Tests erfolgreich beendet werden können. Besteht das Programm diesen letzten Test nicht, muss es von der SF-GmbH nachgebessert werden.

# 11 Entwicklungsumgebung

Im folgenden wird die Umgebung beschrieben, in der die spätere Software entwickelt wird.

## 11.1 Software

Die Software beschreibt die eingesetzte Software-Umgebung um eine effiziente und effektive Erstellung des Programms sicherzustellen.

Als Betriebssystem kommt Windows 7 (Professional | Ultimate) (x64 - 64bit) zum Einsatz. Dieses hat die neusten Updates vorzuweisen und eine gängige aktuelle Antiviren Lösung installiert zu haben.

Als Laufzeitumgebung wird auf die Java Virtual Machine Version 7 von Oracle gesetzt. Hier kommt das Software Development Kit (SDK) in entsprechender Version zum Einsatz. Zum Testen des Programms ist das entsprechende Java Runtime Environment in Version 7 vorzuhalten. Es muss sichergestellt werden, dass die entsprechenden Klassenpfade, sowie Systempfade gesetzt sind, sodass der Prozess java als auch javaw im CLI bekannt ist. Darüber hinaus müssen .jar-Dateien mit der JRE verknüpft sein um einen Start der Anwendung per Doppelklick zu gewährleisten. Entsprechende Einstellungen sind notfalls in der Registry vorzunehmen [vgl. Kapitel 12.1]. Als IDE kommt die Eclipse Plattform mit den entsprechenden Java Entwicklungs-Komponenten zum Einsatz. Installierte Addons sind der Window Builder Pro [vgl. `http://www.eclipse.org/windowbuilder/download.php`] sowie optional das Subclipse Plugin für das Versionsverwaltungstool Subversion.
Für die Versionsverwaltung wird Subversion in der neusten Version (1.7.7) eingesetzt. Auf den einzelnen Entwicklern PCs kommt das Tool Tortoise SVN [`http://tortoisesvn.net/downloads.html`] Während der Entwicklung wird ein SVN-Server von Google verwendet [`http://code.google.com/p/seda-wip-sfgmbh/`]. Die finale Abgabe des Quellcodes erfolgt über das vom Auftraggeber zur Verfügung gestellte-SVN Repository.[1]

Für den Dateiaustausch innerhalb des Entwicklungsteams wird auf den Cloud-Dienst Dropbox gesetzt. Für die Kommunikation untereinander wird auf Skype gesetzt, da hier bereits entsprechende Konten vorhanden sind. Hangouts von Google+ hätten zusätzliche entsprechende Accounts vorausgesetzt und die Videoübertragungsfunktion wurde primär nicht benötigt.

Analog zur Produktumgebung wird die gleiche Datenbank verwendet. welche in Verbindung mit der Software eingesetzt wird, PostgreSQL in Version 9.2.1. Es muss sichergestellt werden, dass die betreffenden PCs welche die neue Univis Software verwenden sollen sowie die Entwickler-

[1] Ein SVN external Attribut wurde in diesem bereits eingerichtet.

PCs für die Datenbank freigeschaltet sind (Freigabe der jeweiligen IPs).

Zusätzlich muss sichergestellt werden, dass entsprechende Netzwerk-Einstellungen (Firewalls, Router) eine ordnungsgemäße Verbindung zwischen Anwendungs-PC, Entwicklungs-PCs und Datenbank erlauben.

Zur Entwicklung der in Kapitel 4.1beschriebenen Oberfläche des Programm wird wie ebreits angedeutet auf Swing gesetzt. Um die GUI schneller zu erstellen und testen wird das Plugin 'Windows Builder Pro' der Eclipse Foundation verwendet. Das Programm wird auf die Swing-Komponenten von Java setzten, daher ist es notwendig sicher zustellen, dass eine entsprechende Shell vorhanden ist. Als Oberfläche wird auf die Standard Fensteroberfläche von Windows 7 (shell: explorer.exe) gesetzt. Kiosk-Systeme sind als GUI nicht vorgesehen.

## 11.2 Hardware

Die Software soll auf IBM-Kompatiblen Computern der Intel x86 Architektur ausgeführt werden. Um sicher zu gehen, dass die Anwendung ausreichend schnell ausgeführt wird, werden nachfolgende Systemvoraussetzungen empfohlen:

CPU: 1Ghz
RAM: 1GB
HDD: >100MB
Peripherie: Maus & Tastatur

Zusätzlich muss sichergestellt werden, dass ein entsprechender Netzwerkanschluss für die Verbindung zum Datenbankserver vorhanden ist.

## 11.3 Orgware

Für die Entwicklung der Software wird vorausgesetzt, dass jeweils ein Mitarbeiter der Fachabteilungen zur Beantwortung von fachspezifischen Fragen zur Verfügung steht. Darüber hinaus ist für Arbeiten vor Ort ein entsprechender Platz zu stellen.

Benutzerhandbücher werden, sofern benötigt, in elektronischer Form (PDF) ausgeliefert. Sie beschreiben die grundlegenden Funktionen der Software.

Der Auftraggeber stellt sicher, dass dem Auftragsnehmer entsprechende Informationen zur Verfügung gestellt werden, um eine ordnungsgemäße Autorisierung und Authentifizierung der Benutzer sicherzustellen.

Zusätzlich wird seitens des Auftraggebers eine entsprechende Netzwerkinfrastruktur bereit gestellt, um eine Verbindung zwischen den Benutzer-PCs und der Datenbank herstellen zu können. Die Datenbank (PostgreSQL) muss auf einem entsprechenden Server installiert sein und als Dienst laufen. Es muss sicher gestellt werden, dass entsprechende Router und Firewalls konfiguriert sind und eine Verbindung in beide Richtungen möglich ist, um eine Persistierung der Daten zu ermöglichen. Es wird davon ausgegangen, dass die Hausverwaltung, sowie die Dozenten entsprechende Informationen zu Räumen und Lehrveranstaltungen korrekt und zeitnah in das System einpflegen um eine sinnvolle Nutzung zu ermöglichen.

## 11.4 Produkt-Schnittstellen

Das Produkt beinhaltet lediglich eine Schnittstelle zur Datenbank wie in Kapitel 4.1 beschrieben. Über diese findet die Persistierung der Daten, sowie die Abfrage komplexer Anfragen statt. Schnittstellen mit der zukünftig geplanten Weboberfläche, sowie Mobilen Anwendung erfolgt nicht über die Software direkt sondern über die gemeinsame Datenbank. Die dort vorliegende Datenstruktur ist allen weiteren Anwendungen bekannt und ermöglicht so eine reibungslose Interaktion zwischen den Systemen. Parallele Sitzungen der Software werden ebenfalls über die Datenbank synchron gehalten. D.h. trägt Benutzer1 an Workstation1 eine neue Lehrveranstaltung ein, so ist diese auch in Workstation2 bei Benutzer2 bei entsprechender Einstellung zu sehen. Eine direkte Kommunikation zwischen den einzelnen Instanzen der Software findet nicht statt.

# 12 Ergänzungen

## 12.1 Jar- Registry Extension

```
[HKEY_CLASSES_ROOT\.jar]
@="jarfile"

[HKEY_CLASSES_ROOT\jarfile]
@="Executable Jar File"

[HKEY_CLASSES_ROOT\jarfile\shell]

[HKEY_CLASSES_ROOT\jarfile\shell\open]

[HKEY_CLASSES_ROOT\jarfile\shell\open\command]
@="\"C:\\Program Files\\Java\\jre7\\bin\\javaw.exe\" -jar \"%1\" %*"
```