Bachelorarbeit am Institut für Informatik der Freien Universität Berlin

Human-Centered Computing (HCC)

Entwicklung einer Webanwendung für Sprechstundentermine

Timo Haffner

Matrikelnummer: 4666162

timo.haffner@fu-berlin.de

Betreuerin und Erstgutachterin: Prof. Dr. C. Müller-Birn

Zweitgutachterin: Prof. Dr. M. Esponda-Argüero

Berlin, 31.07.2017

Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

Berlin, den 31. Juli 2017

Timo Haffner

Zusammenfassung

Die Sprechstunde ist eines der wichtigsten Werkzeuge zur Kommunikation zwischen Professoren und Studierenden am Institut für Informatik an der FU Berlin [Com16]. Die Organisation der Sprechstunde erfolgt momentan durch den Einsatz mehrerer Software-Lösungen, wobei diese in Abhängigkeit von den jeweiligen Dozenten variieren. Durch die fehlende Standardisierung ist der Student gezwungen, sich mit verschiedenen Software-Lösungen zur Buchung eines persönlichen Termins auseinanderzusetzen. Jede der eingesetzten Lösungen zur Organisation einer Sprechstunde hat zudem Vor- und Nachteile. Es fehlt an einer einheitlichen Lösung, welche die Bedürfnisse und Anforderungen der Studierenden und Professoren in vollem Umfang berücksichtigt. Das Ziel dieser Bachelorarbeit besteht darin, eine einheitliche Software-Lösung zur Organisation der Sprechstunde unter Berücksichtigung der vielfältigen Anforderungen der Anwender zu entwickeln. Zur Zielerreichung sollen die Methoden des User-Centered Designs (UCD) eingesetzt werden.

Die im Rahmen dieser Bachelorarbeit implementierte Lösung basiert auf einer Webanwendung, die den Professoren, Studierenden und Gästen aus dem Internet zur Verfügung steht und die Anforderungen und Bedürfnisse der Anwender berücksichtigt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Mit dem folgenden Kapitel soll eine Einführung in Thema und Kontext der Arbeit gegeben werden. Zudem wird das Problem beschrieben, die Ausgangslage skizziert und das Ziel festgelegt. Am Ende dieses Kapitels wird auf den Aufbau der Arbeit eingegangen.

1.1 Thema und Kontext

Die Sprechstunde ist am Fachbereich Informatik an der FU Berlin ein zentrales Instrument zur Kommunikation zwischen Professoren und Studierenden [Com16]. Die Organisation bringt einen erheblichen Verwaltungsaufwand mit sich. Derzeit existieren verschiedene Lösungen, um Termine für Sprechstunden zu organisieren. Viele dieser Lösungen haben sowohl Vor- als auch Nachteile. Im Rahmen dieser Arbeit soll eine Anwendung entstehen, welche die Schwächen der bestehenden Lösungen ausbessert und deren Vorzüge beibehält. Das User Centered Design (UCD) stellt Software-Entwicklern Werkzeuge bereit, mit denen die Usability (Gebrauchstauglichkeit) erheblich gesteigert werden kann. Diese werden eingesetzt um Anforderungen zu erheben, Konzepte zu entwickeln und diese anschließend zu überprüfen und eignen sich daher ideal um das gesetzte Ziel zu erreichen. Das UCD kann sowohl als Philosophie als auch als eine Vielzahl an Methoden verstanden werden, in dessen Mittelpunkt die Einbeziehung der Nutzer steht [AMKP04]. Dieses Vorgehen soll die Usability der zu entwickelnden Software erhöhen, die durch die Effektivität, die Effizienz, sowie durch die Nutzerzufriedenheit definiert wird [Wik16b]. Letztendlich stellt eine gute Usability den entscheidenden Faktor für den Erfolg einer Software-Lösung dar.

1.2 Problembeschreibung und Ausgangslage

Viele der derzeit eingesetzten Lösungen um Sprechstunden zu verwalten haben neben ihren Vorteilen auch Nachteile. Einige erhöhen den Verwaltungsaufwand, andere haben hohe Wartezeiten zur Folge. Die meisten momentan existierenden Lösungen sind umständlich zu bedienen. Teilweise ist die Terminvereinbarung ausschließlich aus dem Universitätsnetzwerk möglich. Die Vielzahl der Lösungen erzeugt zudem eine Unübersichtlichkeit darüber, wie Sprechstunden von den Studierenden gebucht werden sollen. Dieser Sachverhalt wird in Kapitel 2.1 detailliert besprochen.

1.3 Zielsetzung der Arbeit

Das Ziel der Arbeit besteht in der Entwicklung und Implementierung einer einheitlichen Software-Lösung zur Organisation von Sprechstunden am Beispiel der FU Berlin. Mit Hilfe des UCD sollen vor allem die Anforderungen der verschiedenen Benutzergruppen, sowie die an der FU Berlin bereits bestehende Infrastruktur berücksichtigt werden. Zudem sollen bestehende Vorteile der verschiedenen Lösungen vereinigt werden um dadurch eine möglichst komfortable Terminvereinbarung zwischen Professoren und Studierenden zu ermöglichen.

1.4 Vorgehen bei der Umsetzung

Die entstehende Software soll in einem iterativen Prozess entwickelt werden. Während in der ersten Iteration ein Prototyp mit grundlegenden Funktionen entsteht, wird dieser in den nachfolgenden Iterationen um weitere Funktionen ergänzt und weiter an die Anforderungen der Nutzer angepasst und verbessert. Während des Entstehungsprozess werden verschiedene Methoden des UCD genutzt. Daher wird der Nutzer in den Design-Prozess einbezogen, um so Erkenntnisse über Anforderungen zu erlangen und eine effektive und effiziente Lösung zu entwerfen, die den Nutzer bei der Terminvereinbarung, sowie bei den anfallenden administrativen Aufgaben unterstützt.

Der typische UCD-Prozess beginnt zunächst mit der Analysephase, welche dem Sammeln von relevanten Informationen dient. Darunter fallen Aufgaben, Ziele und Kontext der Nutzer [Wik17b]. Aus diesen werden Anforderungen definiert. Nach der Analyse folgt die Designphase, in der die Informationen in einen ersten Prototypen überführt werden. Anschließend wird ein solcher Prototyp getestet und evaluiert. Die Testergebnisse stellen dann die Basis für die nächste Analyse dar und der typische Kreislauf des UCD beginnt erneut.

1.4.1 Analyse

Für die Recherche werden vom UCD Methoden wie User-Interviews, User-Observations und Contextual Inquiries vorgeschlagen. Zudem können Fragebögen entworfen und Umfragen durchgeführt werden. Alternativ zur Befragung einzelner Personen können Focus Groups und Nutzer-Vertreter herangezogen werden [AMKP04, MB, Wik17b]. Bereits bestehende Lösungen, die gleiche oder ähnliche Probleme lösen, können ebenfalls analysiert werden (Review-Analysis) und bieten zugleich einen idealen Einstiegspunkt und die Möglichkeit sich in das Thema einzuarbeiten. Neben harten Fakten, wie Zahlen und Daten, sind insbesondere die Eigenschaften, Fähigkeiten und Angewohnheiten der Nutzer relevant. Auch die kulturellen Unterschiede innerhalb der Zielgruppe haben Einfluss auf die fertige Lösung.

Aus erlangten Informationen werden Nutzergruppen mithilfe von Affinitydiagrammen modelliert und einzelne typische Nutzer (Personas) definiert. Zusätzlich werden Aufgaben und Szenarien (Tasks, Scenarios, Use Cases) erstellt, um anschließend die gesammelten Daten in ein Design zu überführen. Mithilfe einer Primary Noun Architecture werden die wichtigsten Begriffe zusammen mit der Art und Häufigkeit ihres Auftretens aufgedeckt. Aufgabenabläufe können mithilfe von Ablaufdiagrammen visualisiert werden.

1.4.2 Design

Der Übergang von der Analyse- in die Designphase ist fließend. Daher fallen in die Designphase auch weitere Analysen der Daten. Als Beispiele lassen sich das Card Sorting, User Interface Flow Diagrams und Storyboards nennen. Durch das Aufbereiten der Daten sollten sich die grundlegenden Anforderungen und Arbeitsabläufe klar herausstellen. Anhand dieser werden anschließend Protoypen entworfen. In frühen Stadien werden meist Low Fidelity Prototypen entwickelt, wie zum Beispiel Pen and Paper Prototypen. In nachfolgenden Stadien wird ein solcher in einen funktionierenden Prototypen überführt aus dem sich letztendlich die fertige Software entwickelt.

1.4.3 Evaluation

Um das Design zu validieren stellt das UCD weitere Methoden bereit. Zunächst wird es noch einmal mit den Informationen abgeglichen, die in den vorigen Schritten erfasst wurden. Die Begutachtung von einem Experten durch einen Cognitive Walkthrough kann bereits erste Probleme aufdecken [Mö10]. Eine weitere, häufig eingesetzte Methode ist die Evaluation anhand von Heuristiken, wie zum Beispiel der von Nielsen [Mö10]. Ein anderer wichtiger Bestandteil der Evaluation ist der Usability Test. Die einzelnen Schritte dieses Tests können sehr variabel sein. Den Nutzer laut denken zu lassen, ihn in die Technologie einzuweisen, Tests aufzuzeichnen (Video, Ton, Daten, Protokollieren), den Nutzern bestimmte Aufgaben zu stellen oder ganze Szenarien zu testen sind nur einige Möglichkeiten um ein Design zu validieren [CQ08]. Die Schritte können zudem je nach Art der Evaluation variieren. So lässt eine summative Evaluation bei Usability Tests etwaige Hilfestellungen durch einen Moderator und das 'laute denken' des Nutzers entfallen, während es in formativen Evaluationen darum geht, Probleme durch diese Methoden aufzudecken [CQ08].

1.5 Aufbau der Arbeit

In den Kapiteln 2. Analyse, 3. Konzeption und 5. Evaluation wird genauer auf die Verwendung der Methoden des UCD eingegangen, um die jeweiligen Problemstellungen zu lösen. Sie umfassen dabei eine Auswahl der Erkenntnisse, die im gesamten Verlauf der Entwicklung erlangt worden sind. In den einzelnen Kapiteln wird zudem abgebildet, wie sich das Projekt im Laufe der Zeit

und durch die Verwendung der verschiedenen Methoden entwickelt hat. Im Kapitel 4. Implementierung werden zunächst eingesetzte Technologien besprochen. Anschließend sind exemplarisch Abschnitte des Quelltextes abgebildet, anhand der die Funktionsweise des zur Umsetzung der Webanwendung eingesetzten Frameworks leicht verständlich ist. Wichtige Teile der Anwendung werden beschrieben. Die Reihenfolge der Kapitel entspricht der gleichen Reihenfolge, die eine Iteration in einem typischen UCD-Prozess hat: Analyse, Konzeption, Implementierung und Evaluation. Eine schematische Übersicht über einen Iterationsschritt ist in Abbildung 1.1 dargestellt.

Um die Lesbarkeit dieser Bachelorarbeit zu gewährleisten wird für personenbezogene Ausdrücke die männliche Schreibweise gewählt. Auf die Übersetzung englischer Fachbegriffe wird verzichtet, um die Bedeutung dieser nicht zu verfälschen und dem Leser die Arbeit mit englischer Literatur zu vereinfachen.

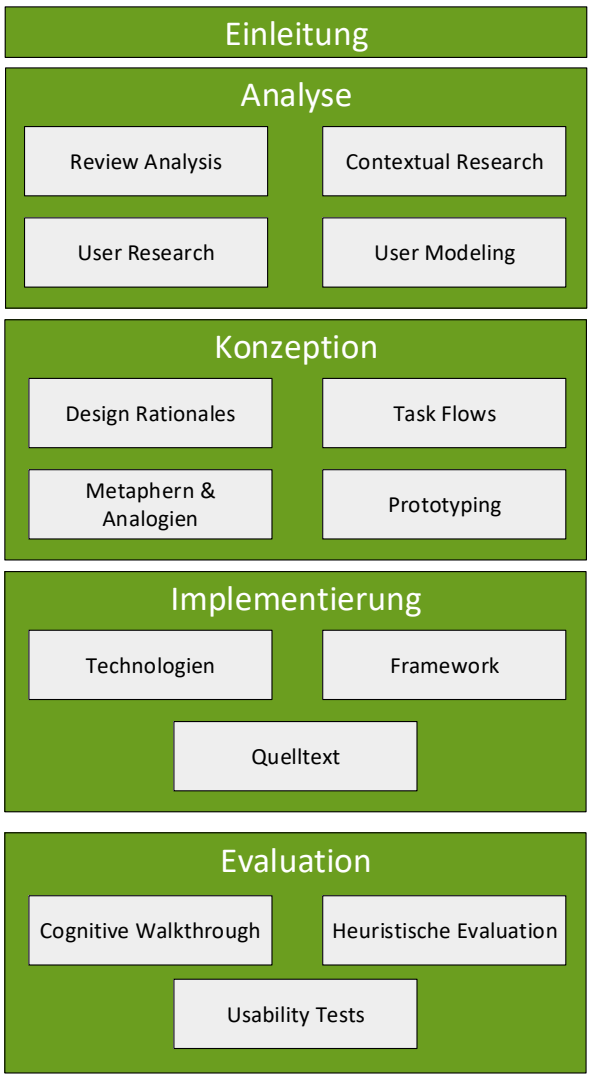

Abbildung 1.1: Schematische Darstellung einer Iteration des User Centered Design mit den Methoden, die in der vorliegenden Bachelorarbeit eingesetzt worden sind.

2 Analyse

Zu Beginn des Entwicklungsprozess muss zunächst in Erfahrung gebracht werden, was die zukünftigen Nutzer der entstehenden Anwendung benötigen. Das Requirements Engineering, unter das die Anforderungsanalyse fällt, ist der Prozess, der sich damit beschäftigt Anforderungen zu definieren, zu dokumentieren und zu pflegen [Wik16c]. In diesem Kapitel wird zunächst die Review Analysis verwendet, um bestehende Lösungen zu analysieren. Anschließend wird Contextual- und User-Research betrieben um mehr über die Anforderungen der Nutzer und deren Umfelder zu erfahren. Mithilfe der gewonnenen Informationen werden User Groups festgelegt und Personas definiert, um eine bessere Vorstellung über die zukünftigen Nutzer der Anwendung zu erlangen. Mithilfe von Scenarios wird anschaulich dargestellt, wie die Anwendung in Zukunft verwendet werden könnte. Mit Tasks werden die einzelnen Ziele der Nutzer genauer formuliert.

2.1 Review Analysis

Die Review Analysis ist eine der wichtigsten Quellen um erste Informationen zu sammeln und bezieht alle Hilfsmittel und Aktionen ein, die versuchen das Problem zu vereinfachen oder zu lösen. Dabei können sowohl technische als auch nicht-technische Hilfsmittel betrachtet werden. Eine E-Mail kann genauso wie eine Konversation nach einer Vorlesung als Informationsquelle genutzt werden. Im Folgenden werden daher die wichtigsten Ansätze genauer betrachtet.

Keine Vergabe von Sprechstundenterminen

Wenn Professoren keine Termine vergeben, stehen die Studierenden vor dem Büro und warten darauf ihr Anliegen vorbringen zu können. Diese Lösung hat den Vorteil eines minimalen Aufwands. Nach dem Minimalprinzip erreicht der Professor mit einem gegebenen Erfolg (der zustande gekommenen Sprechstunde) den Aufwand (die Organisation) zu minimieren. Für die Studierenden ergeben sich dadurch Nachteile, wie z. B. lange Wartezeiten und die Ungewissheit ob alle Sachverhalte in der zur Verfügung stehenden Zeit geklärt werden können.

Mündliche Absprachen

Die einfachste Art Termine zu machen, ist diese persönlich zu vereinbaren. Der Initiator, der die Konversation aktiv beginnt, hat bereits die Person gewählt,

bei der er einen Termin benötigt. Datum und Zeit werden meist explizit abgesprochen. In der Regel wird der Dozent eine Vorauswahl treffen, je nachdem wann er für die Studierenden Zeit hat. Die Studierenden können dann aus der Vorauswahl einen Termin wählen. Der Ort, der in den meisten Fällen durch das Büro des Dozenten bestimmt ist, kann als Gegebenheit vereinbart werden. Sind sich beide Teilnehmer einig wird der Termin meist vermerkt. Dazu eignet sich ein Kalender oder eine einfache Notiz.

Terminvergabe per E-Mail

Die Terminvergabe per E-Mail erfolgt in der Regel in mehreren Schritten. Zunächst bittet der Initiator, also derjenige der den Kontakt aufnimmt, um einen Termin. Hier kann bereits ein erster Vorschlag für Datum und Zeit erfolgen. Anschließend wird der Gesprächspartner den Termin annehmen oder einen Gegenvorschlag unterbreiten. Sollte zu diesem Zeitpunkt noch kein Termin gefunden worden sein, werden sich diese Schritte wiederholen, bis eine Partei den Vorschlag der anderen annimmt. Der Vorteil dieser Lösung ist eine zusätzliche Flexibilität durch den direkten Kontakt mit dem Dozenten. Es können beispielsweise Sprechstunden außerhalb der vorgegebenen Zeiten vereinbart oder andere gesonderte Absprachen getroffen werden. Ein weiterer Vorteil im Vergleich zu den beiden vorigen Lösungen ist die Ortsunabhängigkeit durch die Nutzung der E-Mail. Nachteilig ist die Notwendigkeit der Kontaktaufnahme und der damit verbundene, meist längere Austausch zwischen den Parteien. Dieser hat einen erhöhten Verwaltungsaufwand zur Folge.

Terminvergabe per Wiki

Ein Wiki ist nach [Wik17c] ein System für Webseiten, deren Inhalte von dessen Nutzern sowohl gelesen als auch verändert werden können. Ein solches System wird am Fachbereich Informatik der FU Berlin für die Vergabe von Terminen genutzt. Die typische Vorgehensweise besteht darin, dass die Dozenten in einem Text beschreiben, wie die Termine einzutragen sind und wann die Sprechstunden stattfinden. Dabei wird angegeben wie lange ein Termin dauern sollte, wie dieser formatiert werden soll, wie mit vergangenen Terminen umgegangen werden soll und welche Informationen bei der Buchung hinterlegt werden sollen. Das sind meist Uhrzeit an dem gewählten Datum, Name, E-Mail-Adresse und Thema. Die Terminvergabe mit Hilfe von Wiki-Seiten ist problematisch, wenn zum Beispiel eine Seite von mehreren Nutzern gleichzeitig bearbeitet wird, eingetragene Termine von anderen Nutzern versehentlich bearbeitet werden oder wenn eine Sprechstunde ausfällt und sich die Studenten trotzdem für die Sprechstunde eintragen können. Zudem sind Namen und Themen für alle anderen Nutzer sichtbar hinterlegt. Besonders Vorteilhaft bei dieser Lösung ist die schnelle Bestätigung des Termins, denn beim Speichern der Änderungen ist der Termin eingetragen. Weitere Vorteile sind der geringe

Aufwand des Dozenten im Vergleich zu anderen Lösungen und die Zugriffsmöglichkeit durch Assistenten. Um den Nachteilen dieser Lösung zu entgehen, wird als Anforderung das Anlegen von privaten Terminen vermerkt, das heißt, die Termininformationen werden nicht öffentlich gespeichert. Zudem könnte eine Anforderung das Anlegen regelmäßig wiederkehrender Sprechstundentermine sein, die momentan dadurch realisiert wird, dass die Studierenden die Liste der Termine fortführen. Zuletzt hat ein Dozent durch die Liste eine gute Übersicht über die Termine. Eine solche Übersicht wird ebenfalls als mögliche Anforderung aufgenommen. Diese sollte die Informationen Datum, Uhrzeit, Name, E-Mail-Adresse und Thema beinhalten. Einen kleineren Vorteil könnten die öffentlichen Informationen allerdings auch bieten: Die Studierenden können sich in dringenden Fällen gegenseitig kontaktieren um zum Beispiel Termine zu tauschen.

Terminvergabe per Website

Eine Professorin hat die Problematik der Terminbuchung mithilfe einer Website gelöst. Hier tragen sich Studierende in vordefinierte Slots ein. Für längere Termine werden mehrere Slots gebucht. Durch das Bestätigen des Buttons auf der Seite wird der Termin sofort reserviert. Die Darstellung in Slots kommuniziert klar, welche Zeiten noch offen sind. Vergebene Slots sind entsprechend gekennzeichnet. Die Nachteile dieser Lösung sind, dass eingetragene Termine von den Studierenden nicht angesehen oder bearbeitet werden können und dass es keine Buchungsbestätigung gibt. Zur Absage eines Termins muss eine E-Mail an die Professorin geschickt werden.

Terminvergabe per Sakai

Das Sakai-System ist eine Open Source Plattform, die die Organisation an Universitäten und Hochschulen vereinfachen soll [Wik16d]. An der FU Berlin wird es derzeit beispielsweise für die Organisation von Lehrveranstaltungen genutzt. Die Dozenten, die das Sakai-System nutzen, verlinken es in der Regel auf ihrer Steckbriefseite. Von dort aus weitergeleitet müssen sich die Studierenden zunächst einloggen. Anschließend wird Datum, Ort und Zeitraum der nächsten Sprechstunde angezeigt. Weitere Termine können mit einem Klick auf ein kleines Plus-Symbol neben dem ersten Termin angezeigt werden. Wird die Sprechstunde ausgewählt, werden die einzelnen Termine in Slots angezeigt. Jeder Slot kann durch einen einfachen Mausklick gebucht werden. Ebenso können gebuchte Termine durch dieses Verfahren wieder abgesagt werden.

Zusammenfassung der durch Analyse bestehender Lösungen gewonnenen Erkenntnisse

In der Tabelle 2.1 werden die derzeit bestehenden Lösungen und deren Funktionen miteinander verglichen. Dazu sind die Funktionen und Vorteile aller Lö-

sungen in der Spalte ganz links aufgelistet. Jede weitere Spalte ist einer Lösung zugeordnet. Unterstützte Funktionen sind mit einem Häkchen markiert. Nicht unterstützte Funktionen mit einem Kreuz. In der Tabelle 2.2 werden offensichtliche und mögliche Anforderungen aus den bestehenden Lösungen abgeleitet. Diese sind jeweils der Nutzergruppe Dozenten oder Studierende zugeordnet.

Funktion / Vorteil	*Keine*	*Mündlich*	*E-Mail*	*Wiki*	*Website*	*Sakai*
Termin vereinbaren	✗	✓	✓	✓	✓	✓
Termin ändern	✗	✓	✓	✓	✗	✓
Länge variabel	✓	✓	✓	✓	✗*	✗*
Beschränkung der Termindauer	✗	✓	✓	✓	?	✓
Buchungsdeadline	✗	✓	✓	✗	?	✓
Terminbestätigung	✗	✗	✓	✗	✗	✓
Automatische Mitteilung bei Terminänderung	✗	✗	✗	✗	✗	✓
Automatische Mitteilung bei Terminabsage	✗	✗	✗	✗	✗	✓
Online-Verfügbarkeit	✗	✗	✓	✓	✓	✓
Zugang ohne VPN	-	-	✓	✗	✓	✓
Warteliste mit Nachrückfunktion	✗	✗	✗	✗	✗	✓
Zugriff für Assistenten	✗	✗	✗	✓	?	?
Terminübersicht für Dozenten	✗	✗	✗	✓	✓	✓
Termine privat	✗	✓	✓	✗	✓	✓
Direktes Feedback / Keine Wartezeit	✗	✓	✗	✓	✓	✓
Hinweise zur Vorbereitung auf die Sprechstunde	✗	✓	✓	✓	✓	✗

Tabelle 2.1: Vergleich der eingesetzten Lösungen. *Eine ansatzweise variable Länge kann durch die Buchung mehrerer Zeitslots erreicht werden.

Anforderung	Dozenten	Studenten
Termine vereinbaren	X	X
Termine verschieben	X	
Termine absagen	X	X
Terminpartner auswählen		X
Datum und Zeit vorgeben	X	
Datum und Zeit wählen		X
Ort vorgeben	X	
Bestätigung über erfolgreich gebuchten Termin	X	X
Ortsunabhängigkeit / Online-Verfügbarkeit	X	X
Angabe der wichtigsten Informationen: E-Mail, Name, Thema, Datum und Zeit	X	X
Termine privat speichern	X	X
Anlegen von Serienterminen für Sprechstunden	X	
Termin-Übersicht für Dozenten	X	
Übersicht über vereinbarte Termine für Studierende		X
Dozenten suchen (Name, Fachbereich, Zuständigkeiten)		X
Terminerinnerung		X
Automatische Nachricht über geänderten Termin	X	

Tabelle 2.2: Mögliche Anforderungen nach Nutzern, die durch die Analyse bestehender Lösungen ermittelt wurden und noch validiert werden müssen.

2.2 Contextual and User Research

Dieser Abschnitt befasst sich mit dem Sammeln von Informationen über die Nutzer, ihre Ziele und Absichten und ihr Umfeld.

Interviews mit Professoren

Zur Vorbereitung auf die Interviews wird ein Fragebogen als Hilfsmittel erstellt. Ziel ist es, die möglichen Anforderungen, die aus den bestehenden Lösungen gefolgert wurden, zu validieren und neue, noch nicht berücksichtigte Anforderungen aufzufinden. Eine besondere Schwierigkeit liegt in der Gestaltung der Fragen. Durch diese dürfen die Nutzer nicht beeinflusst werden [MB]. Die Reihenfolge der Fragen spielt hierbei eine wichtige Rolle. In diesem Fall wurden zunächst allgemeine Fragen gestellt, um den Nutzer möglichst viel von sich aus erzählen zu lassen. Im Nachfolgenden wurden die Fragen konkretisiert und zielten darauf ab getroffene Annahmen zu validieren. Im Interview selbst führt diese Vorgehensweise zu der Schwierigkeit die Übersicht darüber zu behalten, welche Fragen im allgemeineren Teil bereits beantwortet wurden.

Nach jedem Interview wurde der Fragebogen etwas erweitert. Im Folgenden eine Aufstellung der gestellten Fragen. Die Anmerkungen in Klammern sind als Hinweise zu verstehen, die verdeutlichen, welchen Hintergrund die jeweilige Frage hat.

- Allgemeine Fragen
  - Wie organisieren Sie Ihre Sprechstunde?
  - Welche technischen Hilfsmittel benutzen Sie dazu?
  - Was sind die Vorteile des technischen Hilfsmittels, das sie benutzen?
  - Sehen Sie Nachteile in diesem technischen Hilfsmittel?
  - Wie bereiten Sie sich auf ein einzelnes Gespräch vor?

- Speziellere Fragen, die gestellt werden, sofern noch nicht geklärt
  - Wie merken Sie sich vereinbarte Termine? (Werden technische Hilfsmittel wie z.B. Kalender benutzt? Wenn ja welche? (Schnittstelle))
  - Benutzen Sie eine Übersicht über die Termine einer Sprechstunde? Welche Informationen sind dort enthalten?
  - Wie gehen Sie vor, falls eine Sprechstunde ausfallen muss?
  - Welche Informationen benötigen Sie von den Studierenden im Vorhinein?
  - Wenn Ihnen ein Student eine E-Mail mit Informationen schickt, die für die Sprechstunde relevant sind, welche Informationen sollten im Betreff stehen?
  - Welche Informationen möchten Sie den Studenten zusätzlich zu Ort (Raum, Stockwerk, Straße, Hausnummer) und Sprechstundenzeiten zur Verfügung stellen? (Zum Beispiel Checklisten mit zu erbringenden Unterlagen zu bestimmten Terminen)
  - Gibt es Themen die in der Sprechstunde erfragt wurden, aber in den Bereich eines Kollegen fallen oder gar nicht behandelt werden? Falls ja, welche?
  - Wie kann man verhindern, dass Sprechstundentermine mit diesen Themen „blockiert“ werden?
  - Wie lange sollte ein Termin dauern?
  - Wovon ist die Dauer abhängig?

  - Greifen Sie von anderen Orten als der FU Berlin auf ihre Termine zu? (VPN-Verbindung)
  - Wie gehen Sie mit Studenten vor, die versuchen einen Termin über einen anderen Weg, als über das von Ihnen vorgesehene Hilfsmittel zu vereinbaren?
  - Wie gehen Sie mit Studenten vor, die spontan zu einem Sprechstundentermin kommen?

- Quantitative Fragen
  - Dauer der Sprechstunde?
  - Wie viele Studenten pro Sprechstunde?
  - Wie viele Sprechstunden pro Woche / Monat?
  - Immer regelmäßig (Terminmuster) alle x Tage?
  - Wann werden Termine geändert? Semesterwechsel? (Immer?)

- Welche weiteren Informationen könnten relevant sein, die ich nicht erfragt habe?

Das Interview wurde mit vier Dozenten geführt. Dabei wurden drei verschiedene Systeme zur Sprechstundenorganisation genutzt. Die Länge der Interviews lag zwischen 15 und 30 Minuten. Die Motivationen der Professoren für die Wahl des verwendeten technischen Hilfsmittels zur Terminvereinbarung sind unterschiedlich. Bei den Wiki-Lösungen steht die Arbeitsersparnis im Vordergrund. Bei E-Mail-Lösungen wird eine gewisse Verbindlichkeit als Vorteil gesehen. Zudem werden E-Mail-Postfächer gerne als Liste mit unerledigten Aufgaben genutzt. Dieser Sachverhalt spiegelt sich auch bei Professoren wider, die eine andere Lösung als die Vereinbarung der Termine per E-Mail gewählt haben. Die Wahl der unterschiedlichen Lösungen ist zudem stark davon abhängig, wie oft die Sprechstunden in Anspruch genommen werden. Ein weiterer Vorteil, der explizit erwähnt wurde, ist, dass die Anmeldung auf der entsprechenden Wiki-Seite nur sehr selten abläuft. Die Session ist also sehr lange aktiv und hat den Vorteil, dass die Anmeldung nicht wiederholt werden muss. Die Öffentlichkeit der Einträge, die auf einer Wiki-Seite gemacht werden, dient als Vorlage für andere Studenten und wird somit als Vorteil gesehen. Als Alternative für Studierende, die ihre Informationen nicht öffentlich teilen möchten, wird die Terminvereinbarung per E-Mail angeboten.

Einige Nachteile wurden ebenfalls genannt. Bei Wiki-Lösungen werden die Regeln, die auf dem Kopf der Seite stehen, von den Studierenden häufig nicht eingehalten. Diese Regeln sollen dafür sorgen, dass Lücken zwischen Terminen gesondert gekennzeichnet werden, alte Termine von den Studierenden gelöscht werden und das Format beibehalten wird. Zudem ist es umständlich bereits vereinbarte Termine zu verschieben oder abzusagen, da alle Teilnehmer manuell benachrichtigt werden müssen. Dafür werden die Studierenden teilweise in neue Termine verrückt und per E-Mail informiert.

Termine, die während einer Sprechstunde abgehalten werden, werden meist nicht gesondert von den Professoren festgehalten. Stattdessen wird der gesamte Block für diesen Zweck freigehalten. Teilweise werden technische Hilfsmittel eingesetzt. Darunter Kalender von verschiedenen Herstellern. Eine Dozentin würde sich eine Lösung wünschen, die ihren digitalen Kalender unterstützt und einen Abgleich der Termine, sowohl vom Kalender zur Anwendung als auch umgekehrt, zulässt. Zudem sollten Termine von den Dozenten und deren Assistenten nachgetragen, verschoben und abgesagt werden können.

Eine Übersicht über Termine einer Sprechstunde wird gerne genutzt. Dazu eignet sich die Wiki-Lösung gut, denn die Termine sind in der richtigen Reihenfolge unter Angabe von Datum, Uhrzeit, Namen und Thema aufgelistet. Die meisten anderen Dozenten verwenden ähnliche Übersichten, wenn sie regelmäßig Sprechstunden geben.

Studierende, die versuchen einen Termin über einen anderen Weg als über das vorgesehene Hilfsmittel zu machen, werden in aller Regel auf die entsprechende Lösung verwiesen. Studierende, die spontan zur Sprechstunde erscheinen werden in der Regel auf die Terminvereinbarung aufmerksam gemacht.

Einige Professoren würden den Studierenden gerne Hinweise zu Themen geben, die dabei helfen in der Sprechstunde effizient mit der Zeit umzugehen. Dazu zählen zum Beispiel Hinweise und Checklisten, mit denen sich Studierende vorbereiten können. Ein beliebter Anwendungsfall ist der Hinweis auf auszufüllende Formulare, die zur Sprechstunde mitgebracht werden sollen. Zudem sollten die Studierenden darauf hingewiesen werden, dass Sprechstunden wieder abgesagt werden sollen, wenn diese nicht in Anspruch genommen werden möchten.

Ein Termin sollte je nach Inhalt maximal eine halbe Stunde dauern. Dabei wird nach Themenbereichen unterschieden. Eine Bachelorarbeit zu besprechen sollte in der Regel weniger Zeit als die Besprechung einer Masterarbeit in Anspruch nehmen. Fünfzehn Minuten sind ein beliebter Richtwert. Projekte

werden meist außerhalb der regulären Sprechzeiten besprochen. Diese dauern häufig auch länger.

Zusätzlich hat sich herausgestellt, dass eine Anmeldung über ein bereits vorhandenes System erfolgen sollte. An der FU Berlin gibt es derzeit zwei wichtige Systeme, die am Fachbereich Informatik verwendet werden. Das ist zum einen das Sakai-System und zum anderen das Shibboleth-System. Die Anwendung sollte die Anmeldung über eines der beiden Systeme unterstützen um den Nutzern die Notwendigkeit zusätzlicher Passwörter zu ersparen.

Professoren, die weniger frequentierte Sprechstunden anbieten, wollen unter Umständen ihre alten Lösungen beibehalten.

Rein quantitativ sind bei regelmäßig abgehaltenen Sprechstunden im Schnitt etwa 4-6 Studierende in einer Sprechstunde von 90-120 Minuten untergebracht. Diese Sprechstunden werden in der Regel einmal in der Woche, mit einem Intervall von 7 Tagen abgehalten. In der vorlesungsfreien Zeit werden Sprechstunden generell unregelmäßiger angeboten. Zudem ändern sich die Sprechstundenzeiten oft am Ende eines Semesters.

Die Tabellen 2.3, 2.4 und 2.5 enthalten die bisher ermittelten Anforderungen unter Angabe der Art, wie sie ermittelt worden sind und für wen diese Anforderungen bestehen.

Anforderung	**Für**	**Durch**	**Validiert durch**
Termine vereinbaren	Studierende	AbL	IV
Termine verschieben	Professoren	IV	IV
Termine absagen	Beide	IV	IV
Termine für andere Teilnehmer eintragen	Professoren	IV	IV
Dozenten auswählen	Studierende	AbL	IV / CI
Datum und Zeit vorgeben	Professoren	AbL	IV
Datum und Zeit auswählen	Studierende	AbL	-
Ort vorgeben	Professoren	AbL	IV
Termine für Softwarekalender bereitstellen	Beide	IV	IV
Angabe der wichtigsten Informationen: E-Mail, Name, Thema, Datum und Zeit	Beide	AbL	IV
Termine nicht-öffentlich speichern	Beide	AbL	IV
Anlegen regelmäßig wiederkehrender Sprechstunden	Professoren	AbL	IV
Dozenten suchen (Name, Fachbereich, Zuständigkeiten)	Studierende	CI	CI
Zeitvorgaben für Termine sinnvoll vorschlagen	Professoren	IV	IV
Andere Lösungen als das Buchungssystem ermöglichen	Professoren	IV	IV
Terminerinnerung	Studierende	IV	IV
Mehrere Sprechstunden an einem Tag anbieten	Professoren	IV	IV
Warteliste mit automatischer Nachrückfunktion für Sprechstunde	Beide	IV	IV
Berechtigungen Dozent / Student mit Server abgleichen	System	IV	IV
Zugang für Assistenten	Professoren	IV	IV
Kalender-API-Unterstützung, Abgleich der Termine in beide Richtungen	Professoren	IV	IV
Teilnehmer über geänderten Termin informieren	Professoren	IV	IV

Tabelle 2.3: Anforderungen an Features die durch die Analyse bestehender Lösungen (AbL), Interviews (IV) und die Contextual Inquiry (CI) mit potentiellen Nutzern ermittelt wurden.

Anforderung	**Für**	**Durch**	**Validiert durch**
Bildschirmnachricht über erfolgreich gebuchten Termin	Beide	AbL	IV / CI
Terminübersicht	Professoren	AbL	IV
Hinweise und Checklisten anzeigen	Professoren	IV	IV
Anwendung in englischer und deutscher Sprache	Beide	IV	IV
Formatierungsoptionen für Hinweise und Checklisten	Professoren	AbL	IV

Tabelle 2.4: Anforderungen an das Interface. Ermittelt durch die Analyse bestehender Lösungen (AbL), Interviews (IV) und die Contextual Inquiry (CI) mit potentiellen Nutzern.

Anforderung	**Für**	**Durch**	**Validiert durch**
Ortsunabhängigkeit	Beide	AbL	IV
Lang anhaltende Sessions	Professoren	IV	IV
Anmeldung über ein vorhandenes System lösen	Professoren	IV	IV

Tabelle 2.5: Anforderungen an das Interaktions-Design. Ermittelt durch die Analyse bestehender Lösungen (AbL), Interviews (IV) und die Contextual Inquiry (CI) mit potentiellen Nutzern.

Contextual Inquiry mit Studierenden

Um die Anforderungen der Studierenden zu erfahren wurden diese mit den bestehenden Lösungen im Sinne einer Contextual Inquiry konfrontiert. Die Contextual Inquiry ist eine Methode, bei der ein Analyst die Benutzer bei der Lösung ihrer Probleme beobachtet und sie dazu befragt. Der Analyst kann dabei als Lehrling betrachtet werden [RF10]. Die Contextual Inquiry bezieht das Umfeld bzw. den Kontext des Nutzers mit ein. Dadurch können Handlungsstrategien, Vorgehensweisen, Rolleneinteilung, Kommunikation, kulturelle und soziale Einflüsse und das physische Umfeld genauer untersucht werden[MR13]. Zusätzlich können Informationen über andere Artefakte gewonnen werden. Darunter fallen genutzte Dokumente, der Aufbau und Gehalt von Informationen, der Verwendungszweck sowie Vorteile und Probleme bei der Arbeit mit bestehenden Lösungen [RF10]. Den befragten Studierenden wurde die vermeintlich simple Aufgabe gestellt einen Termin bei einem bestimmten Professor zu vereinbaren.

Während der Contextual Inquiry wurden verschiedene Dinge erkannt:

- Es wurde von den Studierenden als Vorteil empfunden:
  - wenn die Sprechstunden auf den Steckbriefseiten der Dozenten verlinkt sind
  - wenn Termine nach dem Buchungsvorgang als gebucht angezeigt werden
  - wenn explizit angegeben ist, wie die Termine vereinbart werden sollen bzw. welche Lösung von den Dozenten gewählt wurde
- Es wurde von den Studierenden als Nachteil empfunden:
  - dass die Liste der Dozenten auf der Fachbereichsseite unvollständig ist
- In Bezug auf die Terminvereinbarung per E-Mail:
  - Einige Dozenten weisen darauf hin, dass ein Termin erforderlich ist. Von einigen Dozenten, wird allerdings nicht darauf hingewiesen, wie der Termin zu vereinbaren ist. Meistens ist dann die E-Mail das Mittel der Wahl. Gerade in Anbetracht der Vielzahl an verschiedenen Lösungen sind einige Studierende länger auf der Suche nach einem solchen Hinweis gewesen.
- In Bezug auf die Terminvereinbarung per Wiki:
  - Keiner der befragten Studierenden hatte bei der Frage nach Nachteilen den Datenschutz bemängelt. Erst als konkret nach den Daten gefragt wurde, wurde Kritik geäußert.
  - Ein Studierender hatte zunächst Schwierigkeiten den Edit-Link zu finden.
  - Die Anleitung zur Benutzung der Wiki-Seite wurde oft als zu lang empfunden.
  - Studierende, die mit dieser Lösung bereits gearbeitet haben, fanden sich in der Regel deutlich schneller zurecht.
- In Bezug auf die Terminvereinbarung per Sakai:
  - Zum Zeitpunkt der Durchführung erfolgt eine Weiterleitung auf die Login-Seite, sofern der Nutzer nicht bereits im Sakai-System angemeldet gewesen ist. Nach erfolgtem Login wurde dieser allerdings nicht wieder zurück auf die angeforderte Seite geführt.
  - Auf der Sakai-Seite wird zunächst nur der erste Termin angezeigt. Neben dem Termin ist ein kleines Plus-Symbol um weitere Termine anzuzeigen. In einem Durchlauf hat der Nutzer dieses Plus-Symbol übersehen und aufgrund des belegten ersten Termins gefolgert, dass insgesamt keine Termine mehr verfügbar seien.

- In Bezug auf die Vorgehensweise:
  - Die meisten Studierenden nutzen eine Suchmaschine um nach dem Namen des Dozenten zu suchen. Auf diese Weise erreichten sie zunächst die Steckbriefseite. Die Steckbriefseiten sind Teil des Internetauftritts des Fachbereichs Mathematik und Informatik und fassen die wichtigsten Informationen der Dozenten zusammen. Darunter sind oft Name, Adresse, Telefon, Fax, E-Mail und Homepage. Oft werden auch die verwendeten Lösungen um die Sprechstunden zu organisieren verlinkt. Andere Studierende wählten als Anlaufstelle die Fachbereichsseite. Von dort aus wurde die Liste der Professoren des Fachbereichs gefunden. In dieser Liste sind die einzelnen Steckbriefseiten verlinkt.
  - wenn keine Angabe zur Art der Terminvereinbarung gemacht wurde, wird auf das Schreiben einer E-Mail zurückgegriffen

Die befragten Studierenden nutzten die Sprechstunde im vergangenen Semester zwischen ein und fünf Mal. Es wurden verschiedene Themen behandelt. Darunter waren Erasmus, Abschlussarbeiten und das Anrechnen von Leistungen von einer anderen Universität.

2.3 User Modeling

Der Einsatz des User Modeling und der Analyse des Kontextes sollen Entwickler und Designer in die Lage versetzen aus Sicht der Nutzer zu „denken“ und sich mögliche Konzeptmodelle, die ein Nutzer im Kopf hat zu verdeutlichen. Don Norman definiert das Konzeptmodell als eine Erklärung, die auf eine vereinfachte Weise darstellt, wie etwas funktioniert [Nor16]. Es handelt sich also um ein Modell, das der Nutzer im Kopf hat, um sich die Anwendung zu erklären.

User Groups

Insgesamt lassen sich die Nutzer anhand der Daten in drei Gruppen aufteilen. Studierende, die Sprechstunden in Anspruch nehmen wollen um ihre Anliegen zu besprechen; Professoren und Assistenten, die Sprechstunden anbieten und organisieren möchten und Administratoren, die das System verwalten, warten und initial einrichten. Vernachlässigt man die letzte Gruppe, da diese aufgrund von technischem Verständnis und Erfahrung auch mit weniger optimierten Systemen arbeiten kann, bleiben also zwei Hauptgruppen, nämlich Studierende und Professoren.

Personas

Personas sind Modelle von Nutzern. Sie beschreiben Eigenschaften, Denk- und Verhaltensweisen, die Art der Kommunikation und die Ziele der Nutzer. Die Nutzung von Personas hat den Vorteil, komplexes Wissen abstrakt darstellen zu können und die Kommunikation über die Nutzer zu erleichtern [CRC07]. Sie fassen die Erkenntnisse aus der Analysephase zusammen. Im Folgenden wurden einige Personas definiert.

Prof. Dr. Maria Lange
59 Jahre alt
Erfahrene Professorin an der FU Berlin
Gibt regelmäßig Sprechstunden

Frau Lange ist verantwortlich für die Anrechnung von Leistungen für Masterstudenten und organisiert zusätzlich noch das Erasmus-Programm. Um ihre Aufgaben zu bewältigen bittet sie regelmäßig ihren Assistenten Herrn Müller um Hilfe. Dieser trägt bei Bedarf neue Termine ein oder verschiebt diese, auch wenn letzteres in den letzten Jahren nur ein einziges Mal notwendig gewesen ist.

Prof. Dr. Henry Ford
45 Jahre alt, geboren in den USA
Neuer Professor an der FU Berlin
Gibt Sprechstunden nach Bedarf

Herr Ford spricht zwar gut Deutsch, aber noch nicht ganz so sicher, wie er gerne würde. Im Allgemeinen nutzt er sein E-Mail-Postfach als Liste mit Dingen, die noch zu erledigen sind. Er sieht die Verbindlichkeit der Termine, die per E-Mail vereinbart worden sind, als Vorteil. Er möchte seine Lösung beibehalten, da er Termine ohnehin nur auf Anfrage vergibt.

Matthias Gallo
21 Jahre alt
Austauschstudent aus Italien an der FU Berlin

Spricht italienisch und englisch. Nimmt hin und wieder Sprechstunden in Anspruch um seine laufende Abschlussarbeit zu besprechen. Die Termine bucht er meistens in der Uni mit seinem Laptop.

Scenarios

Mithilfe von Scenarios werden die Interaktionen der Nutzer mit dem System narrativ und informell beschrieben [Car00]. Es wird dabei versucht die Lücke zwischen den erlangten Informationen und dem Design zu überbrücken, indem die wichtigsten Anforderungen an das neue System ausgearbeitet werden [CRC07]. Die entwickelten Personas geben einen guten Einstiegspunkt, um Szenarien zu formulieren.

Szenario 1
Zu Beginn des neuen Semesters stellt Frau Lange fest, dass sie zu den üblichen Sprechstundenzeiten eine Vorlesung halten wird. In ihrem Büro in der Universität ruft sie das Sakai-System auf und folgt von dort dem Link zur Web-Applikation zur Verwaltung der Sprechstunden. Hier wählt sie die vorhandene Sprechstunde aus und löscht die restlichen Termine. Beim Anlegen der neuen Sprechstunde, gibt sie die neuen Sprechzeiten entsprechend an.

Nach einigen Wochen stellt Frau Lange morgens fest, dass sie erkältet ist. Da nach ihrem Besuch beim Arzt feststeht, dass sie erst nächste Woche wieder fit sein wird, muss sie die Sprechstunde morgen ausfallen lassen. Bei sich zu Hause nimmt sie ihr Mobiltelefon zur Hand. In der Anwendung stellt sie einen Ersatztermin zur Verfügung und sagt den betroffenen Termin ab. Die Termine der Studierenden werden automatisch auf die nächsten Sprechstunden verteilt und über die Änderung informiert.

Szenario 2
Herr Ford gibt seine Sprechstunden nach Bedarf. Er möchte die Termine weiterhin per E-Mail vereinbaren. Die Idee einer zentralen Anlaufstelle für die Studierenden befürwortet er. Daher gibt er seinen Namen und die Adresse seines Büros in seinem Profil an. Zudem ändert er in den Einstellungen die Art der Terminvereinbarung auf E-Mail, so dass Studierende bei Interesse einen Termin per E-Mail vereinbaren können. Die entsprechenden Informationen sind in der Anwendung hinterlegt.

Szenario 3
Matthias braucht einen Termin um den Fortschritt an seiner Bachelorarbeit zu besprechen. Da er gerade in der Uni ist, setzt er sich an einen Rechner aus den Pool-Räumen und meldet sich im Sakai-System an. Von hier ruft er die Webanwendung auf und sucht zunächst den Betreuer seiner Arbeit heraus. Anschließend wählt er aus den übrigen freien Terminen einen aus. Damit der Professor weiß worum es geht, gibt er das Thema an. Nachdem er den Termin vereinbart hat, sieht er in der Zusammenfassung noch einmal wann sein Termin ist und wo dieser stattfindet.

Am Tag seines Termins hat Matthias die Adresse längst vergessen. In der Anwendung, die er über den Browser auf seinem Smartphone geöffnet hat, wählt er in seiner Terminübersicht den heutigen Termin aus, der automatisch ganz oben in der Liste angezeigt wird. Anschließend werden ihm die gesuchten Informationen übersichtlich angezeigt.

Tasks

Aus den Szenarien lassen sich die einzelnen Tasks ableiten, die der Nutzer erledigen möchte. Nachfolgend eine Aufstellung über die gefundenen Tasks.

- Als Dozent (oder Vertreter eines Professors) möchte ich:
  - den Studierenden Informationen, wie Name, Adresse und Raum zur Verfügung stellen
  - den Studierenden Informationen zu häufigen Anliegen bereitstellen, damit sich diese besser vorbereiten können.
  - Sprechstunden an bestimmten Tagen zu festgelegten Zeiten bereitstellen.
  - regelmäßig wiederkehrende Sprechstunden bereitstellen
  - die maximale Dauer für einen Termin festlegen.
  - eine Übersicht über die Termine kommender Sprechstunden haben
  - Sprechstundentermine mit meinem Kalender synchronisieren
  - Sprechstunden absagen
  - Sprechstunden verschieben
  - bei der Änderung (Absagen, Verschieben) einer Sprechstunde alle Teilnehmer benachrichtigen
  - beim Absagen von Sprechstunden bereits vereinbarte Termine automatisch verschieben lassen
  - eine Frist festlegen, bis zu welchem Zeitpunkt Studierende sich für eine Sprechstunde eintragen können (z.B. bis 24 Stunden vor dem Termin)
  - Gästen die Anmeldung zu Sprechstunden ermöglichen.
  - meinen Assistenten die Anwendung für mich verwalten lassen.

- Als Studierender möchte ich:
  - einen bestimmten Dozenten finden.

- aus den freien Terminen eines Professors einen auswählen und einige Informationen hinterlassen.
- meine gemachten Termine ansehen können.
- meine gemachten Termine absagen können.
- die wichtigsten Informationen zu meinen Terminen einsehen (Dozent, Zeit, Adresse, Telefon, E-Mail).
- meine gemachten Termine in meinen Kalender übertragen.

2.4 Zusammenfassung der Analyse

In diesem Kapitel wurden die Anforderungen an die neue Software erfasst. Die Review Analysis wurde verwendet um bestehende Lösungen zu betrachten. Sie hat sich als ein effektives Mittel erwiesen, um eine Übersicht über die wichtigsten Anforderungen zu erlangen und erleichtert damit den Einstieg in das neue Thema. Sie bereitet gut auf die im Anschluss durchgeführten Nutzerbefragungen vor, die aus Interviews und Contextual Inquiries bestehen. Die Interviews wurden mit den Dozenten durchgeführt. Die vorbereiteten Fragen helfen dabei eine grundlegende Struktur zu schaffen. Mithilfe der Interviews wurden viele Anforderungen auf Seite der Dozenten aufgedeckt. Mit den befragten Studierenden wurde eine Contextual Inquiry durchgeführt. Im Vergleich zum Interview ist insbesondere die Reihenfolge der Vorgehensweise aufgedeckt worden. Als Ergebnis der drei Methoden sind die Tabellen 2.3, 2.4 und 2.5 entstanden. Sie geben einen Überblick über gewünschte Funktionalitäten.

Im Anschluss wurden diese Anforderungen in Modellen zusammengefasst. Zunächst wurden typische User Groups festgelegt. Dabei wurden drei Gruppen identifiziert, nämlich Professoren, Studierende und Administratoren. Für den Entwicklungsprozess wurde ein besonderer Fokus auf Professoren und Studierende gelegt. Mit Hilfe von Personas wurden wichtige Eigenschaften der typischen Nutzer abstrahiert dargestellt. Die Personas haben sich im Verlauf der Entwicklung der Anwendung als weniger nützlich herausgestellt. Sie werden in größeren Teams neben der Zusammenfassung von Wissen über die Nutzer zur Erleichterung der Kommunikation verwendet. Dieser Vorteil konnte während der Arbeit nicht überprüft werden. Scenarios werden verwendet, um die Nutzung der Software in einen Kontext mit dem Nutzer zu setzen. Durch die knappe Darstellung der wichtigsten Abläufe, helfen sie dabei während der Entwicklung auf diese Abläufe fokussiert zu bleiben. Aus diesen Darstellungen konnten zudem die Tasks und Task Flows generiert werden.

3 Konzeption

Im letzten Kapitel wurde erläutert, wie die Anforderungen der Nutzer erfasst worden sind. In diesem Kapitel geht es darum, aus den Anforderungen Ziele für die entstehende Anwendung festzulegen und grundlegende Entscheidungen zu treffen, die den wesentlichen Charakter der Lösung bestimmen. Zudem sollen Strategien und Maßnahmen für die Umsetzung dieser Ziele herausgestellt werden. Für Entwurf und Design des Konzepts werden wieder einige UCD-Methoden verwendet. Dazu gehören Task Flows, die visualisieren wie bestimmte Ziele erreicht werden und die Primary Noun Architecture, die Häufigkeiten, Aktionen und Ansichten für bestimmte Begriffe genauer betrachtet. Mithilfe von Design Rationales werden in diesem Kapitel Entscheidungen anhand bestimmter Kriterien getroffen. Am Ende des Kapitels wird das erarbeitete Design in Prototypen umgesetzt.

3.1 Art der Anwendung

Viele der bestehenden Lösungen haben gemeinsam, dass sie aus dem Internet erreichbar sind. Jeder Studierende an der FU Berlin hat wenigstens über die Pool-Rechner Zugang zum Internet. Dieser Vorteil soll auch in der entstehenden Anwendung beibehalten werden. Daher bietet sich eine Webapplikation als Lösung an. Die Anwendung könnte auf verschiedene Art und Weise in vorhandene Systeme integriert werden. Abbildung 3.1 zeigt Vor- und Nachteile der verschiedenen Möglichkeiten anhand eines Design Rationales in QOC-Notation.

Ein Design Rationale ist die Auflistung getroffener Entscheidungen unter Angabe der Gründe und Kriterien anhand derer entschieden worden ist [JLI92]. Die QOC-Notation ist eine grafische Aufbereitung von Design Rationales. Sie verbindet die Design-Fragen (**Q**uestions) mit den verschiedenen Optionen (**O**ptions) und Kriterien (**C**riteria).

Die Möglichkeiten, die Anwendung in das Sakai-System zu integrieren oder anzubinden, sind besonders interessant, da sich die Nutzer unter Verwendung des Nutzerkontos für das Sakai-System in der Anwendung anmelden können. Im Vergleich zu den anderen vorgeschlagenen Optionen in der Grafik kommen diese zudem ohne VPN-Verbindung aus und sind die vergleichsweise bequemsten Lösungen. Der letzte wichtige Faktor ist die Wiederverwendbarkeit der Lösung, die durch eine externe Lösung mit Anbindung besser gegeben ist als durch eine Integration als Plugin.

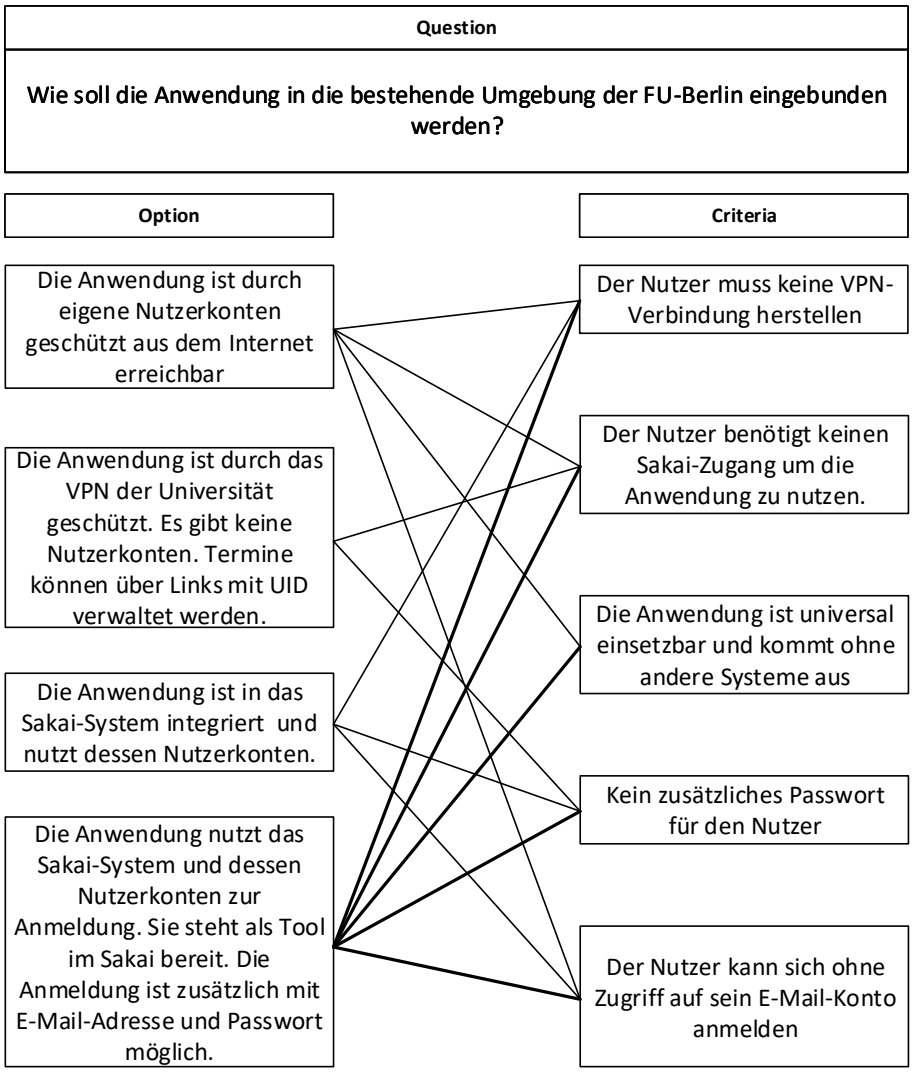

Abbildung 3.1: Design Rationale in QOC-Notation um die Frage der Einbindung der Anwendung in die bestehende Umgebung besser bewerten zu können. Die Frage ist für bessere Lesbarkeit nach oben gerückt.

3.2 User Task Flows

Eine Möglichkeit zur Überführung der Daten in ein Design ist das Erstellen von User Task Flows. In Abbildung 3.2 und 3.3 sind die Task Flows für die Erstellung einer neuen Sprechstunde und das Buchen eines Termins zu sehen. Insbesondere der Buchungsvorgang konnte durch die durchgeführte Contextual Inquiry mit den Studierenden dem typischen Ablauf angepasst werden. Die Task Flows haben zusätzlich dazu beigetragen eine Priorisierung der einzelnen Funktionen und Anforderungen aufzustellen, anhand derer der Entwicklungsprozess besser geplant werden kann.

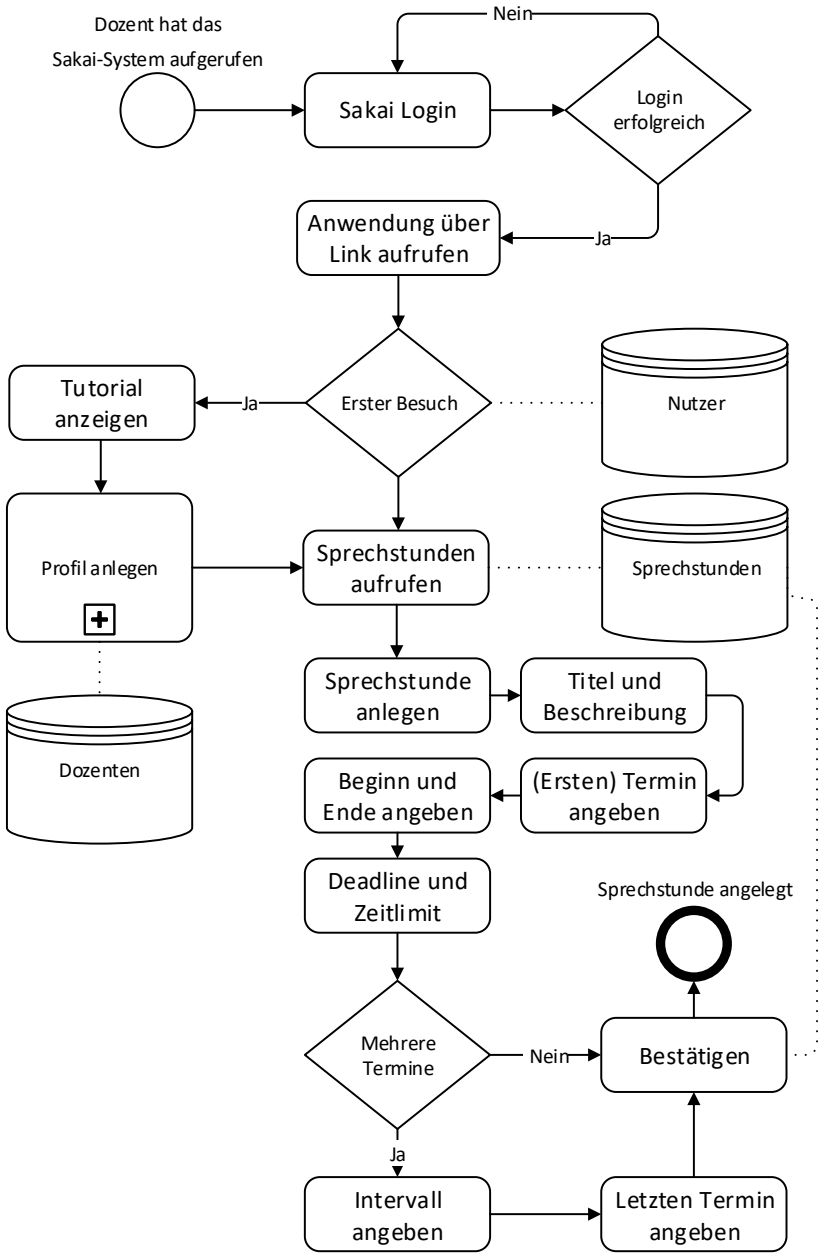

Abbildung 3.2: Task Flow - Wie ein Dozent eine Sprechstunde erstellt: Zunächst meldet sich der Nutzer über das Sakai-System an. Von dort folgt dieser einem Link zur Anwendung. Bei der ersten Anmeldung, wird ihm eine kurze Einführung angezeigt. Im nächsten Schritt wählt der Nutzer den Eintrag „Sprechstunden“, der angezeigt wird, da er als Dozent oder Assistent angemeldet ist. Nach Angabe der Details wie Titel, Beschreibung, Datum, Beginn und Ende bestätigt der Nutzer seine Eingabe. Die Sprechstunde ist angelegt und verfügbar.

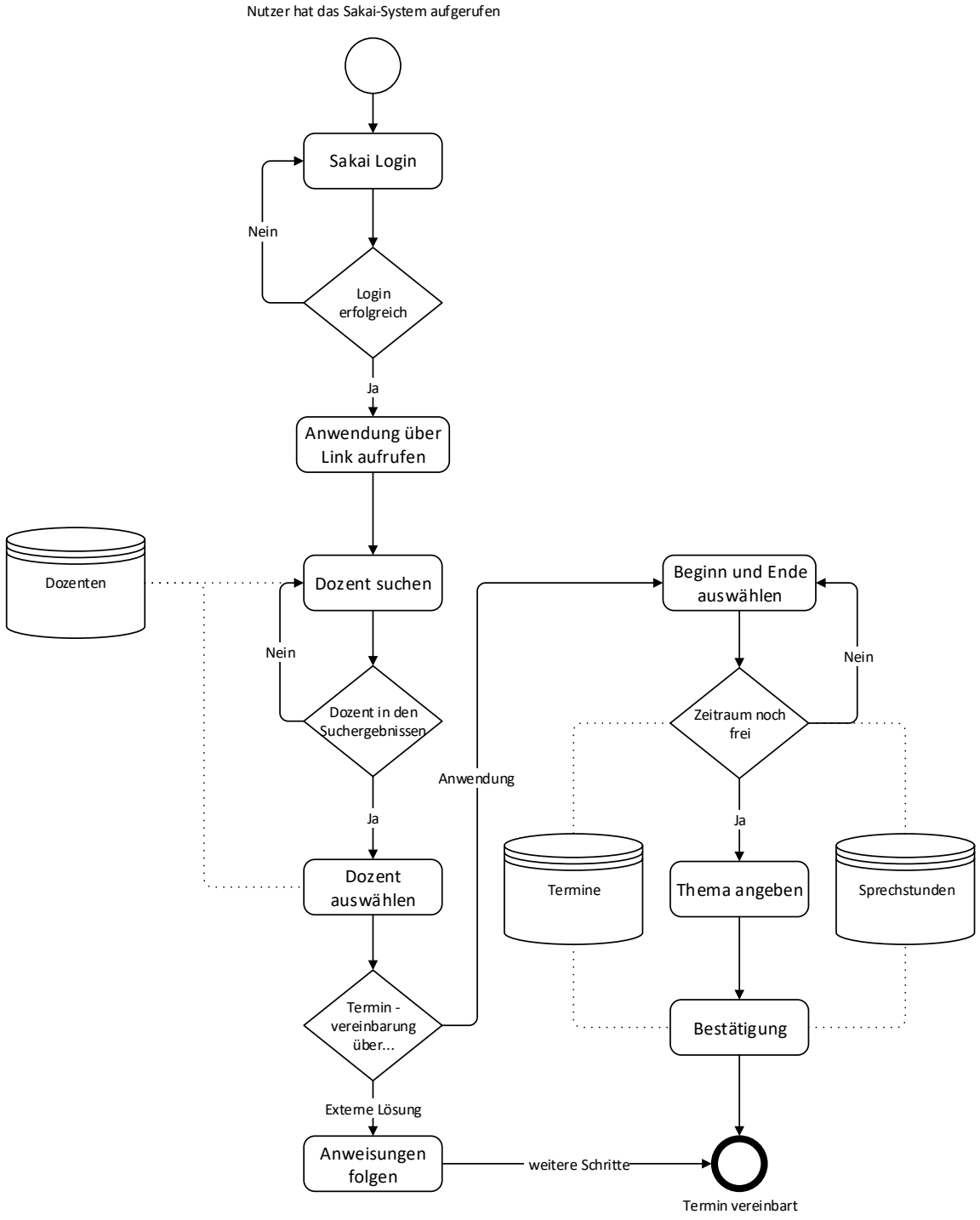

Abbildung 3.3: Task Flow - Wie ein Nutzer einen Termin vereinbart: Nach Anmeldung über das Sakai-System wird der Nutzer zur Auswahl des Dozenten aufgefordert. Anschließend wird diesem die nächste Sprechstunde angezeigt, auf der er einen Termin und dessen Dauer wählen kann. Anschließend kann dieser ein Thema angeben. Nach Bestätigung ist der Termin gebucht.

3.3 Primary Noun Architecture

Um aus den Task Flows ein Design zu erstellen, bieten sich mehrere Methoden an. Eine davon ist die Erstellung einer Primary Noun Architecture wie in den Tabellen 3.1 und 3.2 gezeigt. Mit Hilfe der Primary Noun Value Matrix (NVM) lassen sich die wichtigsten Begriffe, mit denen der Nutzer während seiner Tasks umgeht, darstellen. Die Primary Noun User Group Matrix (UVM) stellt dar, welche Nutzer mit welchen Teilen der Anwendung in Berührung kommen. Die Primary Nouns Dozent und Profil überschneiden sich in ihren Attributen. Während ein Dozent seine Daten in seinem Profil anpasst, wählen die Studierenden aus der Liste von Dozenten und nicht aus der Liste von Profilen. Eine andere Besonderheit ist der Assistent. Diesem soll Zugang zu den Sprechstunden und Terminen eines anderen Dozenten gewährt werden, wenn dieser als Vertretung eingesetzt wird. Der Zugriff auf die Einstellungen des Dozenten soll allerdings nicht möglich sein, während der Assistent auf seine eigenen Einstellungen zugreifen kann.

3.4 Metaphern und Analogien

Metaphern und Analogien zeichnen sich meist durch grafische Elemente wie Icons aus. Die Nutzung solcher Elemente unterstützt die Nutzer dabei, die Verhaltensweise des Systems zu verstehen und ein Konzeptmodell zu bilden [Nor16]. Durch die geschickte Vorgabe eines Konzeptmodells mithilfe von Analogien und Metaphern kann versucht werden die verschiedenen Konzeptmodelle der Nutzer anzugleichen.

Zeitstrahl

In der späteren Entwicklung der Anwendung hat sich herausgestellt, dass es wünschenswert ist, die Auswahl der Dauer eines Termins zu visualisieren. Dafür wurde ein Zeitstrahl gewählt, der die verfügbaren Zeiten aufzeigt und den Nutzer in zwei Schritten zunächst den Beginn seines Termins wählen lässt und anschließend das Ende. Durch die farbig hervorgehobene Verbindung zwischen den beiden Punkten auf dem Zeitstrahl wird deutlich, dass ein Zeitraum ausgewählt ist.

Material Design Icons

In der Anwendung werden die Material Design Icons verwendet um bestimmte Aktionen zu repräsentieren. Die Nutzung eines Icons, dass einen Mülleimer zeigt, lässt Nutzer aus verschiedenen Gründen wissen, dass die repräsentierte Aktion das Löschen eines Elements ist. Ein Grund ist die Verwendung eines Mülleimers im täglichen Gebrauch zum Entsorgen nicht benötigter Dinge. Ein anderer ist die weitläufige Verwendung dieser Abbildung in anderen Anwendungen. Der Mülleimer ist nur ein Beispiel. Weitere eingesetzte Icons sind

Primary Nouns	**Anzahl**	**Attribute**	**Actions**	**Views**
Dozent Dozent	10-200	Titel Vorname Nachname Adresse Raum Etage Fachbereiche	Ansehen	Liste Details
Termine Appointments	1 - 10 pro Sprechstunde und Dozent	Start Ende Thema	Vereinbaren Absagen Verschieben	Liste Vereinbaren Ändern
Sprechstunde Office Hour	1 bis 2 pro Woche	Start Ende Dauer	Anlegen Absagen Ändern	Liste Details Ändern
Profil Profile	1	Titel Vorname Nachname Adresse Raum	Ändern	Ändern
Einstellungen Settings	1	Profil Vertretung Gastnutzer	Ändern	Ändern
System-einstellungen System-Settings	1	Schnittstellen E-Mail		Ändern

Tabelle 3.1: Primary Noun Value Matrix (NVM)

beispielsweise ein Stift zum Bearbeiten, eine Lupe zum Suchen, Pfeile zum Ein- und Ausklappen von Elementen und viele mehr.

3.5 Wahl des Frontend Frameworks

Da die Anwendung primär für die Freie Universität Berlin entsteht, gilt es einige Design-Entscheidungen zu treffen, die das Wesen des User-Interfaces betreffen. Die Freie Universität Berlin bestimmt mit ihrem Corporate Design das einheitliche Erscheinungsbild ihrer Auftritte nach außen [FUB15]. Mangels Verfügbarkeit und aufgrund gestalterischer Freiheit wurde auf die strikte Einhaltung dieser Regeln verzichtet. Dennoch sind Elemente aus diesem Design in der entstandenen Anwendung enthalten um vertrauten Nutzern die Zugehörigkeit zur Freien Universität Berlin zu verdeutlichen. Um den Entwicklungsprozess zu beschleunigen und einige grafische Designfragen zu vermeiden wurde

Primary Nouns	Studierende	Dozenten	Assistenten	Admins
Dozent Professor	X			
Termine Appointments	X	X	X	
Sprechstunde Office Hour	X	X	X	
Profil Profile		X		
Einstellungen Settings		X	(X)	
Systemeinstellungen System-Settings				X

Tabelle 3.2: Primary Noun User Group Matrix (UVM) - Übersicht darüber, welcher Nutzer mit welchen Teilen der Anwendung in Berührung kommt.

weiterhin das Material Design Lite ([Goob]) verwendet. Hierbei handelt es sich um ein Framework für das Frontend, dass unter Verwendung von HTML, CSS und JavaScript versucht die Vorgaben des Material Design umzusetzen. Das Material Design wurde von Google entwickelt und ist eine Designsprache, die viele grafische Elemente vorgibt [Gooa]. Die Nutzung des Material Design Lite hilft zudem dabei bestimmte Design Patterns umzusetzen. So sind zum Beispiel Dropdown-Menüs, Navigation-Tabs und einige andere Konzepte vordefiniert. Durch das einheitliche Design lassen sich Heuristiken wie Consistency (siehe Abschnitt 5.1.2) einfacher handhaben. Anhand verschiedener Design Rationales wurden Entscheidungen getroffen, die bestimmen, wie sich beide Konzepte am sinnvollsten vereinigen lassen.

3.6 Hauptmenü

Stellvertretend für viele weitere Designentscheidungen, die das User Interface direkt betreffen, lässt sich der Einsatz eines weiteren Design Rationales an der Konzeption des Hauptmenüs zeigen. In Abbildung 3.4 wird entschieden, wie das Hauptmenü auf mobilen Endgeräten aussehen soll. Dabei werden die verschiedenen Möglichkeiten, die in den Abbildungen 6.1 und 6.2 gezeigt sind, gegeneinander aufgewogen.

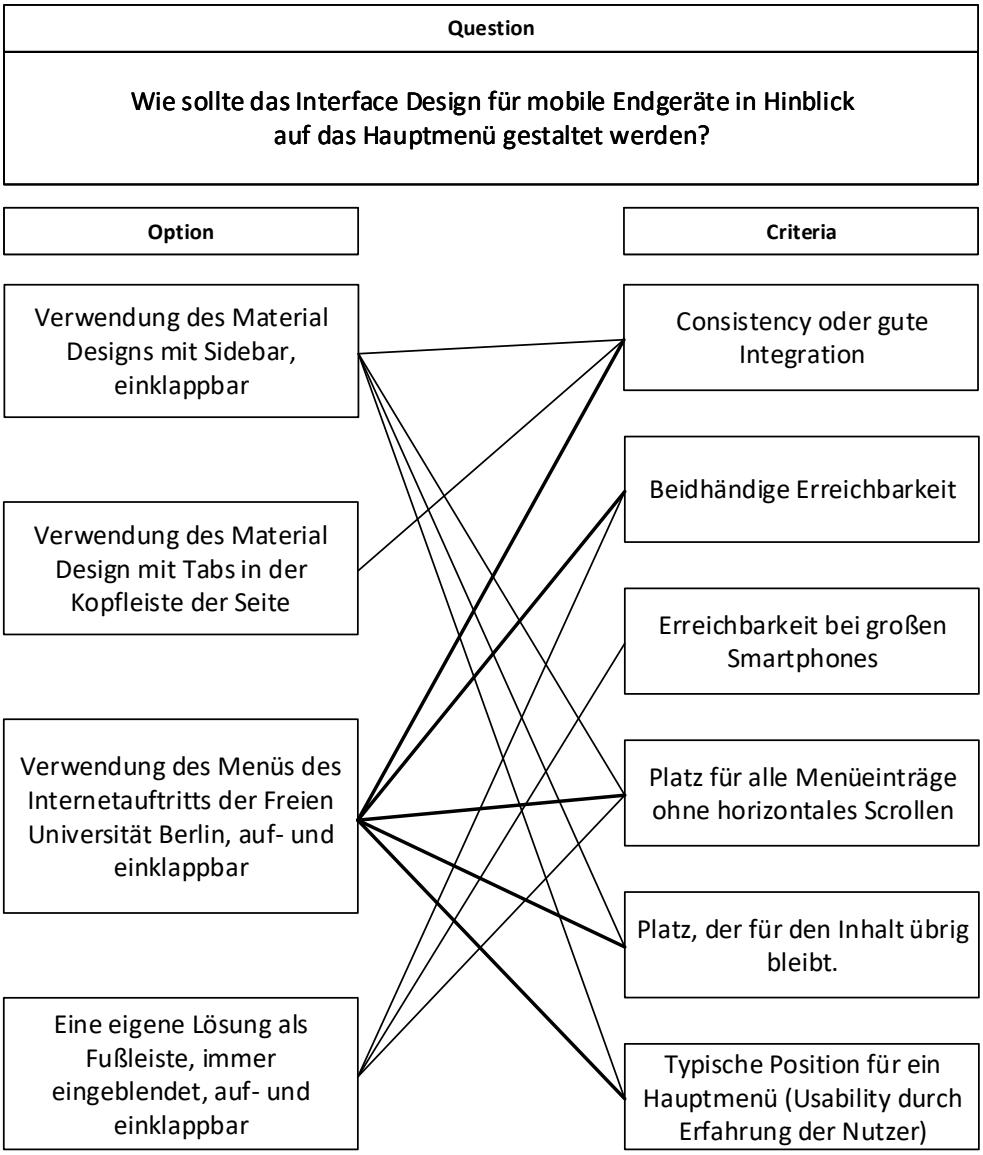

Abbildung 3.4: Design Rationale in QOC-Notation. Alle Kriterien sind positiv formuliert, so dass jede Verbindungslinie ein Vorteil darstellt. Drei der Optionen sind in den Abbildungen 6.1 und 6.2 dargestellt.

3.7 Zeitslots oder freie Terminwahl

Auch Entscheidungen, die die Interaktion betreffen, können mithilfe von Design Rationales entschieden werden. Ein Beispiel dafür ist in Abbildung 3.5 zu sehen. Es muss entschieden werden, ob die Terminbuchung durch die Wahl

von Zeitslots oder frei erfolgen soll. In diesem Fall ist die Wahl aus den in der Grafik genannten Gründen auf die freie Terminwahl in 5-Minuten-Schritten gefallen.

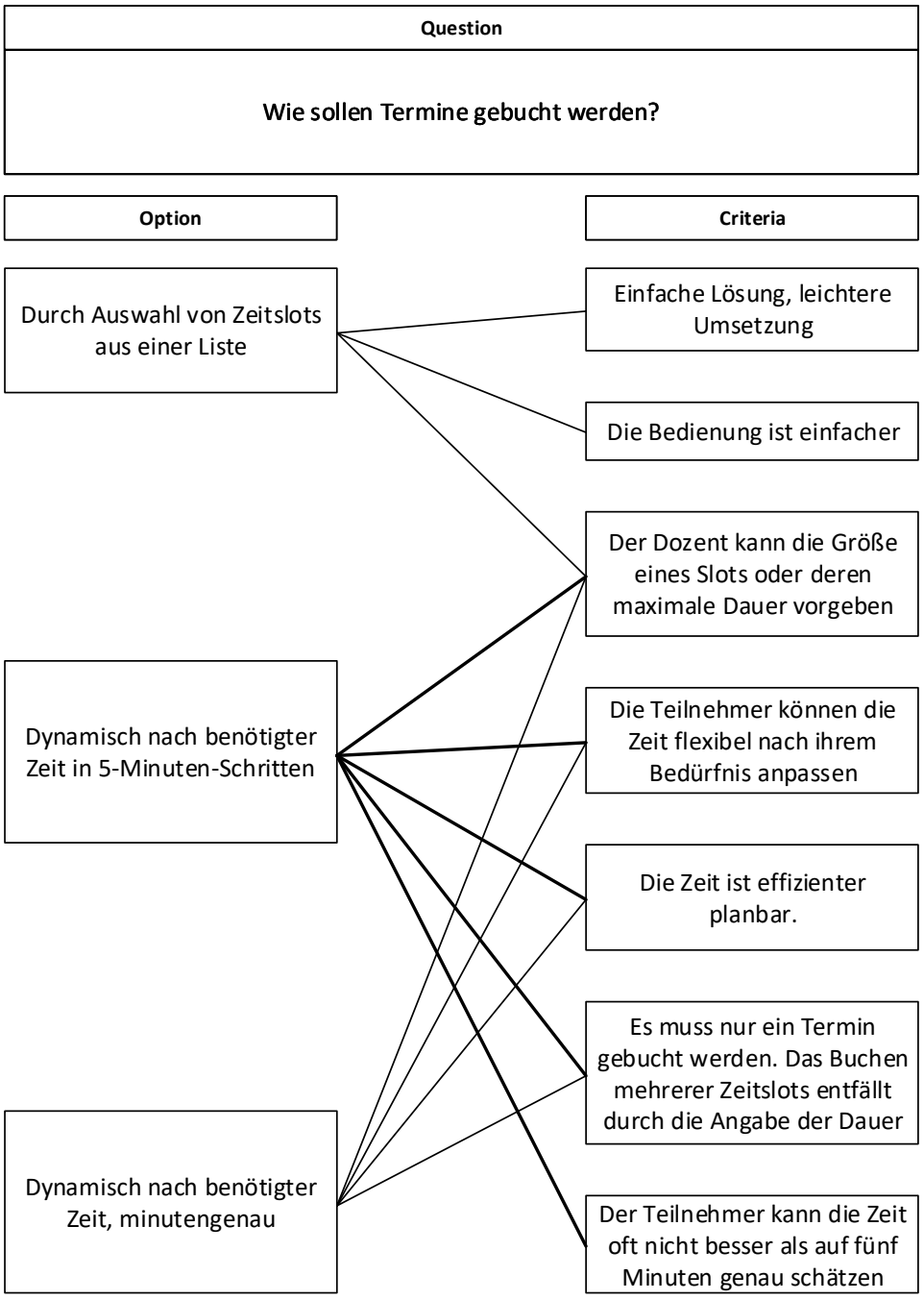

Abbildung 3.5: Entscheidung zwischen dynamischer und fester Terminlänge mithilfe der QOC-Notation.

3.8 Umsetzung

Nachdem die grundlegenden Entscheidungen getroffen worden sind wird in diesem Abschnitt mit der Umsetzung erster Ideen begonnen. Dazu werden die wichtigsten Anforderungen nach Priorität sortiert und anschließend erste Prototypen umgesetzt.

3.8.1 Priorisierung der Funktionen

Während der Anforderungsanalyse ist eine Liste der gewünschten Anforderungen der Nutzer entstanden. Um herauszufinden, welche dieser Anforderungen in einem ersten Prototypen umgesetzt werden müssen, wurden diese zunächst in zwei Task Flows überführt, die das Anlegen von Sprechstunden 3.2 und das Buchen eines Termins 3.3 visualisieren. Diese wurden deshalb gewählt, da sie die elementaren Ziele der Anwendung widerspiegeln. Wichtige Anforderungen sind jene, die zur grundlegenden Funktionalität beitragen und auf die weitere Funktionen aufbauen. Diese stehen in der Tabelle 3.3 weiter oben und sollten im Verlauf der Entwicklung früher implementiert werden, damit diese in der verfügbaren Zeit mit Sicherheit umgesetzt werden.

3.8.2 Prototyping

Das Prototyping teilt sich in zwei Teile auf. Zum einen werden Low-Fidelity-Prototypen entworfen. Diese sind meist unvollständig und mit Hilfe von Stift und Papier umgesetzt [RF10]. High-Fidelity-Prototypen sind dagegen oftmals Teile funktionierender Software.

Entwürfe auf Papier

Im Regelfall geht es, bevor eine erste prototypische Anwendung entsteht, zunächst an die Konzeption und das Design eines Pen-and-Paper-Prototypen. Auch in diesem Fall wurden zunächst einige Layouts für die verschiedenen Ansichten erarbeitet. Abbildung 3.6 zeigt einen solchen Prototyp der Startseite. Diese soll dem Anwender direkt vermitteln wo er sich befindet, welchen Zweck die Anwendung erfüllt und welche Optionen er hat. Da zeitgleich die Möglichkeit der Anmeldung über das Sakai-System geprüft wurde und daher bereits ein Teil der Anwendung existierte, wurde auf Evaluationen für diesen Low-Fidelity-Prototypen verzichtet und stattdessen auf dem High-Fidelity-Prototypen durchgeführt.

High-Fidelity-Prototyping

Im späteren Verlauf muss entschieden werden wie erste technische Prototypen erstellt werden. Dazu gibt es im Wesentlichen drei Varianten. Zum einen kann vertikales Prototyping genutzt werden, bei dem eine Auswahl der Anforderungen bis ins Detail implementiert wird. Ein anderer Ansatz ist das horizontale

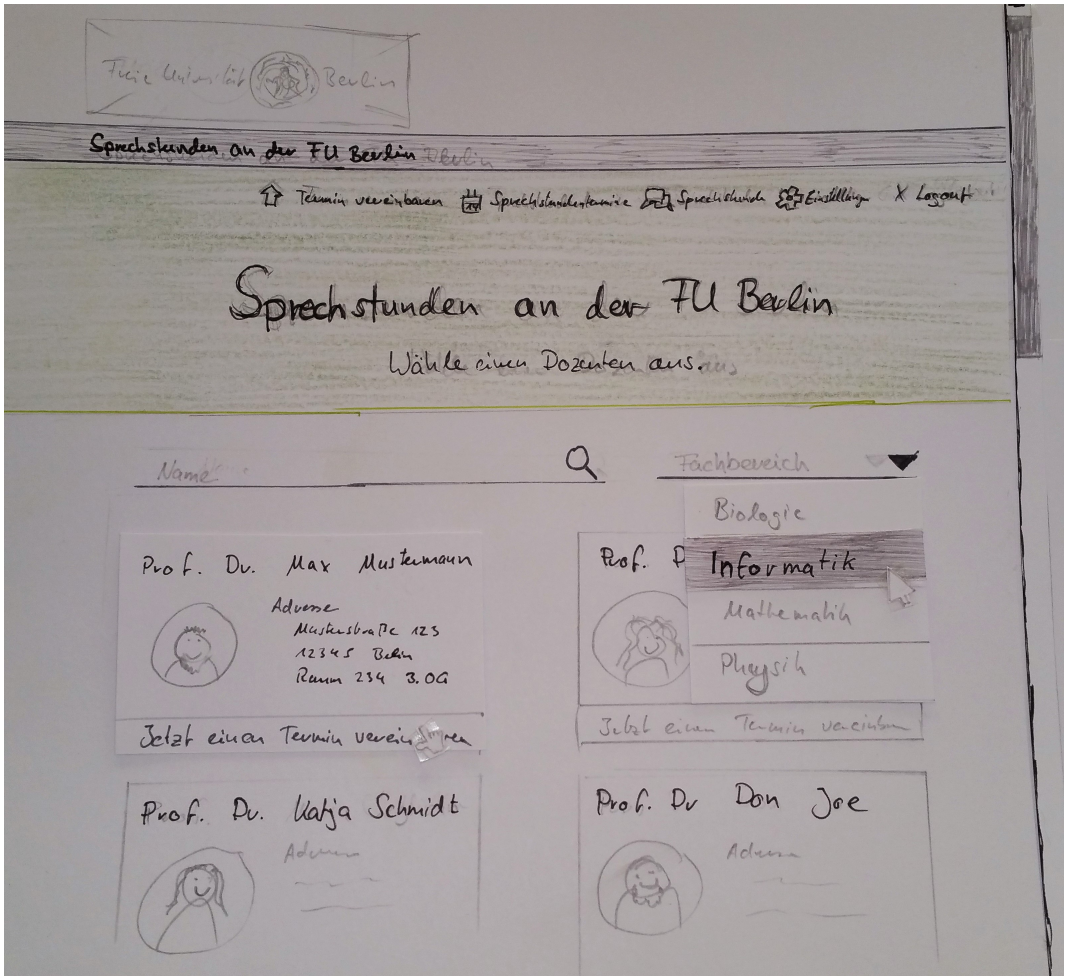

Abbildung 3.6: Low Fidelity Prototyp der Startseite der Anwendung mit Dropdown, Scrollbar und austauschbaren Suchergebnissen. Die unterschiedlichen Cursor sind beispielhaft an zwei verschiedenen Stellen dargestellt.

Prototyping, bei dem versucht wird einen möglichst großen Teil der Anforderungen oberflächlich umzusetzen. Als dritte Alternative existiert noch das Szenario-basierte Prototyping, bei dem die Anforderungen für bestimmte Szenarien bis ins Detail implementiert werden [Mö10]. Für diese Anwendung wird vertikales Prototyping verwendet. Dafür gibt es zwei wichtige Gründe. Der erste Grund ist, dass durch die Prüfung der technischen Umsetzbarkeit des Login-Systems bereits relativ früh ein wesentlicher Teil realisiert wurde. Als zweiter Grund ist der Aufbau der Anforderungen zu nennen. Es gibt einige elementare Anforderungen, auf die viele weitere aufbauen. Als Beispiel sind grundlegende Funktionen wie das Anlegen einer neuen Sprechstunde oder die Terminbuchung zu nennen. Funktionalitäten, wie der Zugriff von Assistenten oder die zeitliche Beschränkung der Möglichkeit Termine zu buchen, bauen darauf auf.

Anforderung	**Professoren**	**Studierende**
Anmeldung über ein vorhandenes System lösen	X	X
Datum und Zeit vorgeben	X	
Datum und Zeit auswählen		X
Termine vereinbaren		X
Termine absagen	X	X
Dozenten auswählen		X
Termine verschieben	X	
Teilnehmer über geänderten Termin informieren / Informiert werden	X	X
Termine für andere Teilnehmer eintragen	X	
Angabe der wichtigsten Informationen: E-Mail, Name, Thema, Datum und Zeit	X	X
Ort vorgeben	X	
Anlegen regelmäßig wiederkehrender Sprechstunden	X	
Professoren suchen (nach Name, Fachbereich, Zuständigkeiten)		X
Terminübersicht einer Sprechstunde	X	
Bildschirmnachricht über erfolgreich gebuchten Termin (Bestätigung)	X	X
Hinweise und Checklisten anzeigen	X	
Anwendung in englischer und deutscher Sprache	X	X
Andere Lösungen als das Buchungssystem ermöglichen	X	
Termine nicht-öffentlich speichern		X
Terminerinnerungen		X
Zugang für Assistenten	X	
Termine für Softwarekalender bereitstellen	X	X
Kalender-API-Unterstützung Abgleich der Termine in beide Richtungen	X	
Zeitvorgaben für Termine sinnvoll vorschlagen	X	

Tabelle 3.3: Anforderungen sortiert nach Priorität

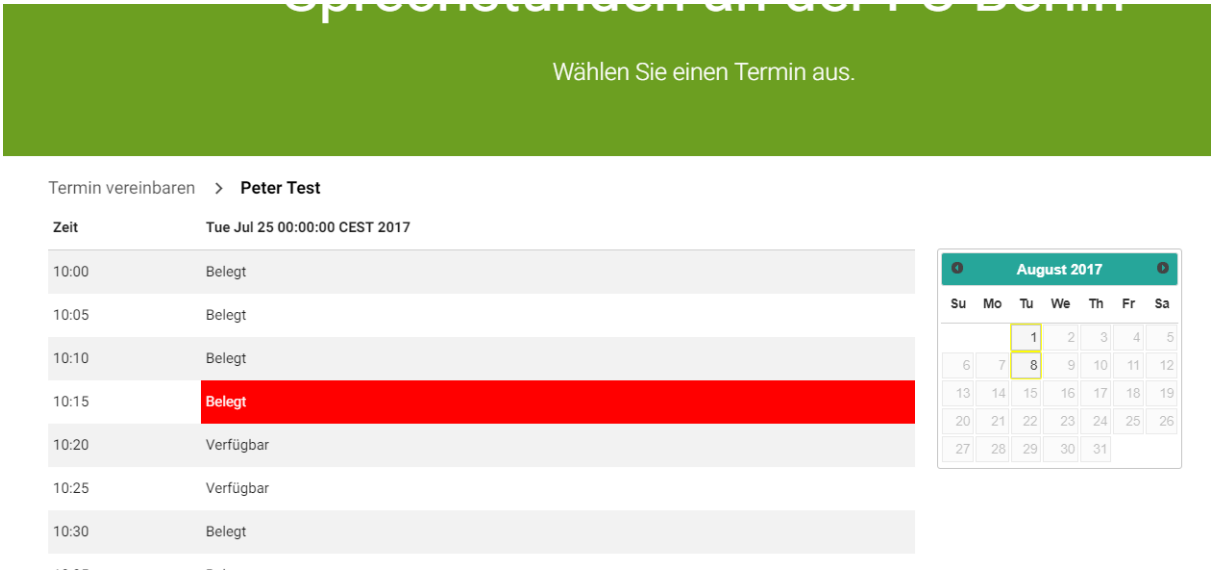

Abbildung 3.7: High Fidelity Prototyp der Terminauswahl. Im späteren Entwicklungsprozess wird diese Ansicht durch einen Zeitstrahl ersetzt, mithilfe dessen zusätzlich auch die Dauer des Termins verändert werden kann.

3.9 Zusammenfassung der Konzeption

In diesem Kapitel wurden wichtige Schritte zur Konzeption der Softwarelösung vorgenommen. Viele der Entscheidungen wurden mit Hilfe von Design Rationales in QOC-Notation getroffen. Die grafische Darstellung mit Auflistung aller Kriterien sind hilfreich um Entscheidungen unvoreingenommen zu treffen. Wie bei vielen anderen Methoden des UCD haben die Design Rationales in QOC-Notation weitere Vorteile für die Kommunikation im Team [CRC07]. Diese Vorteile konnten nicht überprüft werden. Die Task Flows sind in diesem Abschnitt besonders hilfreich gewesen. Sie verbinden die Analyse mit der Konzeption und beschreiben den Prozess der Nutzer anschaulich. Die Primary Noun Architecture ist für das Design der Datenbank hilfreich. Insbesondere durch die Angabe der Anzahlen, helfen sie zudem dabei Entscheidungen für das User-Interface zu treffen. Beispielsweise ist eine Suchfunktion erst ab einer gewissen Anzahl an Objekten notwendig.

Der zweite Teil dieses Kapitels verbindet die Konzeption mit der Implementierung durch die Anfertigung eines Prototypen. Das Low-Fidelity-Prototyping hilft dabei frühzeitig Probleme an der Anwendung zu erkennen. Während dieser Phase sind einige unterschiedliche Skizzen und Entwürfe für User-Interfaces entstanden. Durch die klaren Anforderungen aus der Analysephase und den Beispielen aus den bestehenden Lösungen sind diese bereits sehr ähnlich gewesen. Die Implementierung mit Hilfe solcher Entwürfe geht deutlich schneller

voran, als ohne. Der Grund dafür ist, dass gezeichnete Konzepte einmal umgesetzt werden, während bei einer freien Implementierung auch fertige Teile der Software immer wieder verbessert und ersetzt werden. Für die Evaluation eignen sich High-Fidelity-Prototypen allerdings deutlich besser, insbesondere, wenn diese von einer Person durchgeführt werden. Das liegt daran, dass ein klassischer Usability Test mit einem Low-Fidelity-Prototypen mehrere Rollen umfasst [RF10]. Dabei wird die Software simuliert, ein Protokoll angefertigt und moderiert. Mit einem High-Fidelity-Prototyp entfällt die Software-Simulation und das Protokoll kann durch technische Hilfsmittel ersetzt werden.

4 Implementierung

In diesem Kapitel wird auf die Wahl der Technologie und wichtige Stellen im Quelltext der fertigen Lösung eingegangen.

4.1 Technologien

Für die Entwicklung der Anwendung wurde Groovy in Verbindung mit dem Webframework Grails verwendet. Groovy ist eine Programmiersprache, die sowohl statische als auch dynamische Typisierung unterstützt [Wik17a]. Ihr übersetzter Bytecode wird auf der Java Virtual Machine (JVM) ausgeführt. Grails baut unter Nutzung von Groovy, auf Frameworks wie Spring und Hibernate auf. Mit Grails erstellte Webanwendungen können in ein Web Application Archive (WAR) exportiert und zum Beispiel mit einem Apache Tomcat betrieben werden. Neben dem Einsatz von Groovy kann auch Java als Programmiersprache zum Einsatz kommen. Das Grails Framework gibt durch seine Struktur für die Anwendung eine Aufteilung in Model, View und Controller (MVC) vor. Zudem können in sogenannte Services weiterer Code ausgelagert werden. Die Domain-Klassen repräsentieren das Model. Über diese wird abgebildet welche Tabellen und Spalten später in der Datenbank vorhanden sind und welche Beziehungen die einzelnen Domains haben. Die einzelnen Abhängigkeiten der Domain-Klassen können Abbildung 4.1 entnommen werden. Views sind durch Groovy Server Pages (GSP) abgebildet. Diese dienen der einfachen und dynamischen Erzeugung von HTML-Seiten. Controller verarbeiten die Anfragen an den Server und überprüfen diese auf Vollständigkeit und Richtigkeit.

Als Datenbank ist MySQL eingesetzt worden. JavaScript und jQuery wurden für dynamische Inhalte der Webanwendung verwendet. Für die grundlegende Strukturierung der Inhalte wurde das Framework „Material Design Lite“ verwendet [Goob].

4.2 Überblick

Die fertige Anwendung besteht aus mehreren wichtigen Bereichen, die sich anhand der Menüstruktur gut ablesen lassen:

- **Terminbuchung**
- Meine Termine
- **Sprechstunde**

- Einstellungen
  - Profil
  - Art der Terminvereinbarung
  - Benachrichtigungen
  - Nutzer registrieren
  - Assistenten hinzufügen
  - Systemeinstellungen
  - Fachbereiche
- **Login-System**

Im Folgenden wird auf die hervorgehobenen Bereiche genauer eingegangen. Dabei werden grundlegende Teile der Anwendung zuerst behandelt.

4.3 Login-System

Dem Nutzer das Anmelden zu ermöglichen ist eine der grundlegenden Anforderungen an diese Sprechstunden-Anwendung. Ohne Anmeldung ist eine dem Nutzer überlassene Verwaltung seiner gemachten Termine nur über Umwege möglich. Insbesondere die Anforderung der Nutzer, dass die Anwendung ohne weitere Passwörter auskommen sollte, stand dem persönlichen Ziel, von anderen Systemen unabhängig zu bleiben entgegen und ist ein besonderer Schwerpunkt gewesen. Letztlich wurde die Anmeldung sowohl über das Sakai-System als auch mit E-Mail und Passwort für Gastnutzer ermöglicht. Für die Authentisierung und Authentifizierung der Nutzer wurde das auf dem Spring Security Framework basierende Grails-Plugin benutzt. Dieses Framework stellt bereits eine User- und eine Role-Domain bereit um einen Anmeldevorgang unter Berücksichtigung bestimmter Rechte zu ermöglichen. Daher musste das System so erweitert werden, dass sich Sakai-Nutzer ebenfalls anmelden können.

Die verwendete Schnittstelle des Sakai-Systems ist dabei „Learning Tools Interoperability“ (LTI). LTI ist eine Spezifikation in der sogenannte Tool-Provider an einen oder mehrere Tool-Consumer angebunden sind [Inc]. In diesem Fall handelt es sich bei dem Tool-Provider um die Sprechstundenverwaltung und der Tool-Consumer ist das Sakai-System. Das Sakai-System ist in der Lage solche Tool-Provider in das eigene System einzubetten. Wenn der Nutzer das Tool aufruft wird diesem die Sakai-User-ID mitgeteilt. Anhand der User-ID können Nutzer später identifiziert und Zugriff auf ihre Daten gewährt werden. Mithilfe von OAuth wird dabei sicher gestellt, dass der Toolaufruf wirklich von dem Tool-Consumer vorgenommen wurde. Dazu ist es bei der Einrichtung der Systeme notwendig, dass der Tool-Consumer einen eindeutigen Key zugeordnet bekommt und beide Systeme über ein geteiltes Geheimnis verfügen. Dieses

Geheimnis kann als Administrator in den Systemeinstellungen zusammen mit dem Key und der Domain des Tool-Consumers eingegeben werden.

Wenn ein Nutzer die Webapplikation aufruft, wird zunächst ein Request an die Anwendung gesendet. Dieser Request wird im LtiController verarbeitet. Zunächst wird geprüft, ob die verwendeten LTI-Versionen des Tool-Consumers und des Tool-Providers übereinstimmen. Der Key wird bei jedem Request vom Tool-Consumer übermittelt. Anhand dieses Keys kann das bei der Einrichtung gespeicherte Geheimnis aus der Datenbank geholt werden. Das Geheimnis wird verwendet, um mit LTI und OAuth den Request zu prüfen. Da das Geheimnis beiden Parteien vorliegt, kann die übermittelte Signatur vom Tool-Provider überprüpft werden. Dadurch kann verifiziert werden, dass der Request tatsächlich von einem bestimmten Tool-Consumer stammt und die Inhalte nicht verändert worden sind.

Wenn der Nutzer, der den Vorgang ausgelöst hat, die Anwendung das erste Mal aufruft, wird für ihn ein neuer Eintrag in der Datenbank angelegt. Hier werden zum Beispiel Name und seine Termine gespeichert. Bei seinem nächsten Besuch können diese Daten aus der Datenbank geladen werden und der Nutzer kann diese verändern. Die Klassen LTIAuthenticationProvider und LTIAuthenticationToken werden dazu verwendet den Nutzer direkt beim Aufruf über das Sakai in der Anwendung anzumelden.

4.4 Sprechstunden

Eine Sprechstunde besteht aus dem Beginn, dem Ende, einem optionalen Titel und einer optionalen Beschreibung. Zudem können Professoren die maximale Dauer eines einzelnen Termins festlegen und beschränken, bis zu welchem Zeitpunkt Termine gebucht werden können (Deadline). Zu jeder Sprechstunde gehören zudem die gebuchten Termine, die durch eine “”hasMany“-Beziehung verbunden sind und der Dozent. Wenn mehrere Sprechstunden gleichzeitig angelegt werden, sind diese in der gleichen OfficeHourGroup. Die Domain-Klasse, die diese Daten in der Datenbank abbildet ist sehr anschaulich und im Listing dargestellt. Die Validatoren überprüfen jeweils, ob der Beginn einer Sprechstunde auch wirklich vor dessen Ende liegt. Die Richtung dieser Abhängigkeiten können anhand der Abbildung 4.1 abgelesen werden. So hat ein Lecturer viele OfficeHours (1..*) und zu einer OfficeHour gehören viele Appointments.

```
1    Date start
2    Date end
3
4    String publicTitle
5    String publicDescription
6
7    Date deadline
8    Integer maxDurationInMinutes
```

```
9
10      static hasMany = [appointments:Appointment]
11
12      static belongsTo = [lecturer:Lecturer, officeHourGroup:
            OfficeHourGroup]
13
14      static constraints = {
15          start nullable: false, validator: {val, obj -> obj.start && obj.
                end && obj.start < obj.end}
16          end nullable: false, validator: {val, obj -> obj.start && obj.end
                 && obj.start < obj.end}
17          publicTitle maxSize: 255, nullable: true
18          publicDescription maxSize: 1024, nullable: true
19      }
```

Listing 4.1: OfficeHour.groovy

4.5 Termine und Buchung

Zusammen mit den Sprechstunden bilden die Termine den Kern der Anwendung. Auch für den Termin gibt es eine Domain-Klasse, einen Controller und einen Service. Jede Sprechstunde ist in Zeitabschnitte (Slots) von fünf Minuten unterteilt. Bei der Buchung eines freien Termins kann der Nutzer nach Vorgabe der Dozenten wählen, wie viel Zeit er benötigt, um sein Anliegen zu besprechen. Um diese Slots in der Anwendung anzuzeigen werden die Daten aus der Datenbank zunächst aufbereitet. Damit bestimmte Teile des Quelltext nicht mehrfach implementiert werden müssen, können Abschnitte des Codes in Services ausgelagert werden. Beispielsweise wurde die Funktion zur Aufbereitung dieser Daten für die Darstellung „getOfficeHourSlots“ in den Service implementiert. Von hier aus ist sie von allen Controllern aufrufbar. Nachdem diese Daten aufbereitet aus dem Service an den Controller zurückgereicht wurden folgt der Aufruf von ‘”render“ in der letzten Zeile der Funktion.

```
1  def index() {
2      // Check permissions and get user
3      [..]
4
5      def officeHours
6      if (params.date) {
7          Date d = new Date(params.long("date"))
8          officeHours = appointmentService.getOfficeHourSlots(lecturer, d)
9      } else {
10         officeHours = appointmentService.getOfficeHourSlots(lecturer,
               null)
11     }
12
13     def dates = appointmentService.getOfficeHourDates(lecturer)
```

```
14      def model = [officeHours:officeHours, dates: dates, lecturer:
            lecturer, user: user]
15      render (view: 'index', model: model)
16  }
```

Listing 4.2: OfficeHour.groovy

An dieser Stelle werden die HTML-Seiten erzeugt, die an den Client zurückgesendet werden. Dazu gibt es im Grails-Framework sogenannte Groovy Server Pages (GSP). Innerhalb dieser kann auf die Daten der Anwendung zugegriffen werden, um die Inhalte entsprechend der Anfrage des Clients anzupassen. Mithilfe von Closures beginnt ein Codeblock innerhalb einer GSP-Datei. So wird im Folgenden Beispiel der Name des Dozenten in die Überschrift gerendert.

```
1  [..]
2  <h4>Einen Termin bei \${lecturer?.title} \${lecturer?.firstName} \${
       lecturer?.lastName} vereinbaren</h4>
3  [..]
```

Listing 4.3: appointment/index.gsp

Einige Domain-Objekte können von verschiedenen Nutzern verändert werden. Dazu zählt zum Beispiel die Domain „Appointment“, die einen Termin repräsentiert. Ein Termin kann sowohl von dem Nutzer abgesagt werden, der ihn vereinbart hat, als auch von dem Dozenten, in dessen Sprechstunde der Termin vereinbart wurde.

Das Spring Security Plugin stellt die @Secured-Annotation bereit, um den Zugriff auf einzelne Funktionen oder ganze Controller zu limitieren. So wird mit der folgenden Annotation der Zugriff allen angemeldeten Nutzern gewährt.

```
1  @Secured("hasAnyRole('ROLE_USER', 'ROLE_LECTURER', 'ROLE_ADMIN')")
2  class AppointmentController {
3      [..]
```

Listing 4.4: @Secured-Annotation

Um zu verhindern, dass ein Nutzer den Termin eines anderen Nutzers bearbeiten oder absagen kann, müssen zusätzlich seine Berechtigungen geprüft werden. In Controllern, die Requests für solche Domain-Klassen behandeln, gibt es eine ähnliche Struktur. Mit Hilfe der Annotation wird zunächst überprüft, ob der Nutzer angemeldet ist. Anschließend wird überprüft, ob der Nutzer die spezielle Berechtigung für das bestimmte Objekt, wie zum Beispiel den Termin, hat. In den meisten Fällen werden dann die übermittelten Parameter überprüft. Weitere Anweisungen folgen je nach Anwendungsfall. Am Ende

wird das Ergebnis ausgegeben. Dafür kann zum Beispiel der Befehl „render“ oder „respond“ genutzt werden.

```
def show(Appointment appointment) {
    User user = springSecurityService.getCurrentUser()
    if (user == appointment.user || appointment.officeHour.lecturer ==
        user) {
        respond appointment
        return
    }

    response.status = 401
    render (controller: 'error', action:'unauthorized')
}
```

Listing 4.5: @Secured-Annotation

Eine weitere Besonderheit beim Buchen der Termine ist, dass mehrere Benutzer versuchen könnten einen Termin gleichzeitig zu buchen. Die entsprechende Funktion zum Buchen eines Termins wurde in den Appointment-Service ausgelagert. Da Services im Grails-Framework als Singletons implementiert sind, kann es nur eine Instanz eines Services geben. Durch die Verwendung des Schlüsselworts „synchronized“ bei der Deklaration einer Funktion, kann der Code also nicht mehr gleichzeitig von verschiedenen Threads bearbeitet werden. Dadurch wird sichergestellt, dass Termine nicht doppelt belegt werden.

```
/**
* Creates a new appointment if there are no problems
* @param appointment
* @returns true if appointment was created
*/
def synchronized checkBlockedAndCreate(Appointment appointment) {
    if (!checkIfFree(appointment)){
        return false
    }

    appointment.save()
    return !appointment.hasErrors()
}
```

Listing 4.6: Terminbuchung

4.6 Zusammenfassung der Implementierung

In diesem Kapitel sind zunächst einige Grundlagen in der Verwendung des Grails-Frameworks vermittelt worden. Anschließend wurden Teile der Sakai-Schnittstelle und der Terminbuchung besprochen. Zudem wurde darauf eingegangen wie der Zugriff auf Termine geregelt ist.

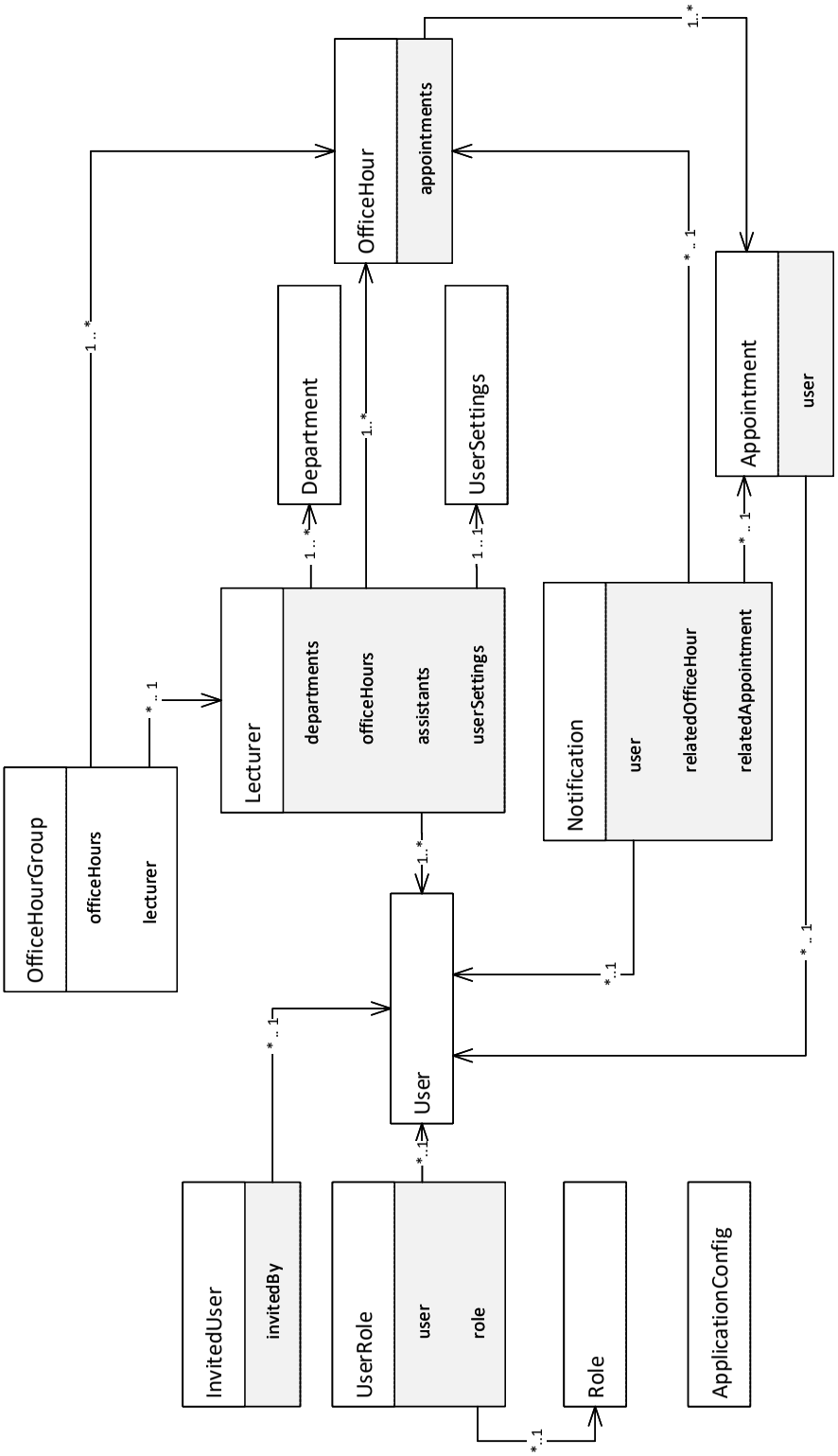

Abbildung 4.1: Abhängigkeiten der Domain-Klassen. Die Beziehungen der einzelnen Klassen sind unter Angabe der Kardinalitäten dargestellt.

5 Evaluation

Die Evaluation befasst sich mit der Bewertung der entstandenen Prototypen und der fertigen Anwendung. Mithilfe des Cognitive Walkthroughs und der heuristischen Evaluation können Entwickler erste Usability-Probleme erkennen. Anschließend wurden mehrere Usability Tests durchgeführt um weitere Schwierigkeiten mit der Webapplikation zu erkennen. Abschließend wird mithilfe der System Usability Scale beurteilt, ob die Anwender mit dem System zufrieden sind. Anhand des Erfolgs kann Effektivität gemessen werden und anhand der Dauer, die für eine Aufgabe benötigt wurde kann ein Einblick in die Effizienz erlangt werden.

5.1 Formative Evaluation

Die formative Evaluation befasst sich mit der Bewertung und Verbesserung der Software vor ihrer Fertigstellung oder der Fertigstellung einzelner Komponenten [Wik16a]. Sie begleitet den Entwicklungsprozess.

5.1.1 Cognitive Walkthrough

Im Rahmen des Cognitive Walkthrough sollen sich ein oder mehrere Usability-Experten in die Lage eines Nutzers versetzen. Die Experten werden dabei vor die gleiche Aufgabe gestellt, die ein Nutzer typischerweise erledigen muss. Dabei stellen sie sich für jeden Schritt selbst die Frage, wie ein Nutzer vorgehen würde [WRLP94]. Als Hilfestellung können an dieser Stelle sämtliche Informationen aus dem User Modeling genutzt werden. Das mentale Modell des Nutzers spielt für diese Methode eine besonders wichtige Rolle. Je besser sich die Experten in den Nutzer hinein versetzen können, desto besser funktioniert diese Methode. Das Cognitive Walkthrough zielt mit seiner Evaluation auf die einfache Erlernbarkeit einer Software ab [WRLP94].

Vorbereitung

Um einen Cognitive Walkthrough vorzubereiten wird bestimmt wer die Nutzer der Anwendung sind, welche Aufgaben analysiert werden sollen und was der korrekte Lösungsablauf ist. Zur Vorbereitung auf die Usability Tests wurden die Aufgaben in Abschnitt 5.1.3 definiert. Die typische Nutzergruppe für die gewählten Aufgaben sind Studierende. Die Evaluation wird am Prototypen durchgeführt.

Durchführung

Die Durchführung besteht im Wesentlichen aus der Begutachtung jeder Aktion und dem Versuchen eine Begründung zu finden, warum der Nutzer diese Aktion durchführt (Prüfung der Plausibilität) [WRLP94]. Bei jedem Schritt sollten sich die Experten folgende Fragen stellen, die der Quelle [WRLP94] entnommen und frei übersetzt worden sind:

- Wird der Nutzer versuchen den richtigen Effekt zu erzielen?
- Wird der Nutzer bemerken, das die richtige Aktion verfügbar ist?
- Wird der Nutzer die richtige Aktion mit dem richtigen Effekt assoziieren?
- Wenn der Nutzer die richtige Aktion durchführt, wird er dann erkennen, dass er einen Fortschritt in Richtung der Aufgabenlösung gemacht hat?

Task 1 - Einen Termin anlegen

Nachfolgend wird der Cognitive Walkthrough exemplarisch an einem späten Prototyp gezeigt. Ausgewählte Ergebnisse dieses und voriger Durchgänge werden im Anschluss gezeigt.

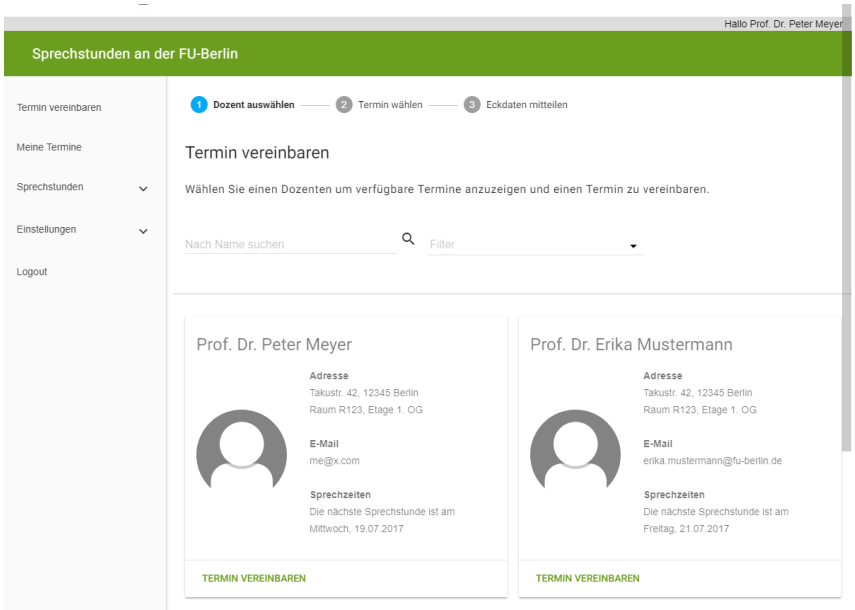

Abbildung 5.1: Schritt 1 - Dozent suchen und / oder auswählen.

Schritt 1: Der Nutzer sucht den Dozenten durch Eingabe seines Namens in der Suchleiste.
Prüfung der Plausibilität:

- Auf der Startseite werden dem Nutzer eine Liste von Dozenten präsentiert. Daher wird er versuchen seinen gewünschten Gesprächspartner in der Liste zu finden.

- Der Nutzer liest „Nach Namen suchen“ und erkennt das Lupen-Icon als regelmäßig eingesetztes Element zur Kennzeichnung von Suchfeldern wieder. Die Möglichkeit nach dem Fachbereich zu suchen verbirgt sich hinter „Filter“. Dieser Hinweis sollte ersetzt werden, da Nutzer ansonsten nur explorativ auf diese Möglichkeit stoßen werden.
- Bei der ersten Eingabe in die Suchleiste symbolisiert der angezeigte Spinner den Ladevorgang. Dieser ist ein typisches Symbol für einen Ladevorgang.
- Während der Eingabe des Namens werden nach und nach weniger Suchergebnisse gezeigt, daher weiß der Nutzer, dass er auf dem richtigen Weg ist.

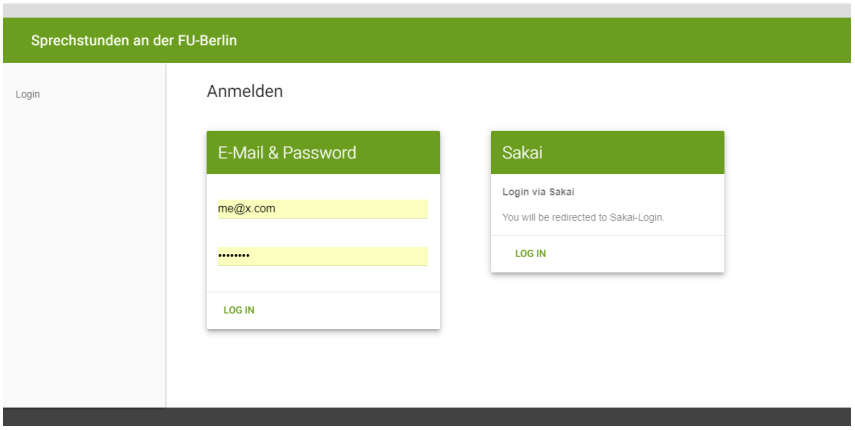

Abbildung 5.2: Schritt 2 - Anmelden

Schritt 2: Anmelden.
Prüfung der Plausibilität:

- Die Anwendung leitet den Nutzer automatisch auf die Anmeldeseite weiter. Nutzer die regelmäßig im Internet sind, wissen aus ihrer Erfahrung, dass sie sich jetzt anmelden müssen. Die Eingabe von E-Mail-Adresse und Passwort wird gefordert. Das verleitet den Nutzer dazu die gewünschten Informationen einzugeben. Nutzer die das Sakai-System nutzen, kennen das System. Hier könnten unter Umständen zwei Probleme auftreten. Das erste Problem könnte darin liegen, dass Nutzername und Passwort des Sakai-Systems im Gäste-Login ausprobiert werden. Das zweite Problem könnte der Begriff „Sakai“ sein. Viele Studierende kennen dieses System unter dem Namen KVV.
- Der Nutzer sollte durch die Überschrift „Anmelden“ und die beiden vorhandenen Karten „E-Mail & Passwort“ und „Sakai“ erkennen, dass eine Anmeldung erforderlich ist und dass zwei Optionen zur Verfügung stehen.

- Beide Karten sind mit dem Button Log-In versehen. Die Funktion der Karten sollte erkennbar sein.
- Nach der Anmeldung mit den richtigen Nutzerdaten wird die Anwendung den Nutzer automatisch zum ausgewählten Dozenten weiterleiten. Der Nutzer wird an der grau hinterlegten Zeile oben rechts erkennen, dass er angemeldet ist, da dort seine E-Mail-Adresse oder sein Name steht. Die notwendigen Schritte zur Terminvereinbarung kann er an den Steppern oben im Bild 5.2 ablesen.

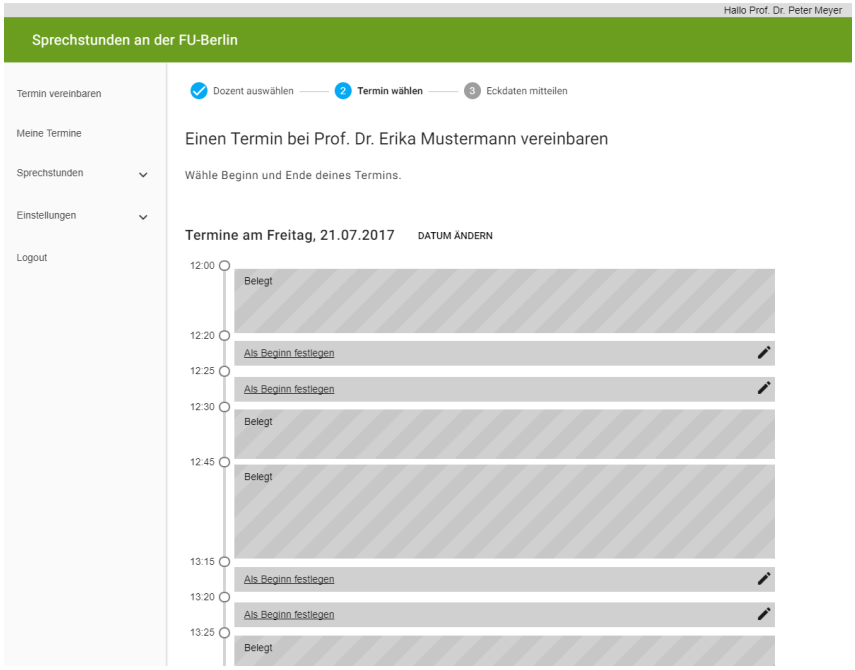

Abbildung 5.3: Schritt 4 - Erkennen, dass keine Termine mit einer Dauer von 20 Minuten verfügbar sind.

Schritt 4: Termine sind vergeben
Prüfung der Plausibilität:

- Der Nutzer erkennt die angezeigten Termine. Bei genauer Betrachtung wird ihm auffallen, dass ausschließlich Zeiträume verfügbar sind, die kürzer als 20 Minuten sind. Daher sollte der Nutzer versuchen, sich die nächsten Sprechstunden anzusehen.
- Die belegten Termine zu erkennen sollte dem Nutzer durch die grafische Hervorhebung und den Hinweis “”Belegt“ leicht fallen. Die Lücken und deren Länge lassen sich ebenfalls leicht durch die angegebenen Zeiten identifizieren. Der Nutzer könnte Schwierigkeiten damit haben, den “”Datum ändern“-Button zu finden, da er nicht aus dem Interface hervorsticht.

- Den Wunsch die nächste Sprechstunde zu sehen sollte der Nutzer mit dem Datum in Verbindung bringen können.
- Nach dem Klick auf den '"Datum ändern"-Button öffnet sich eine Übersicht mit verfügbaren Daten. Der Nutzer erhält direktes Feedback.

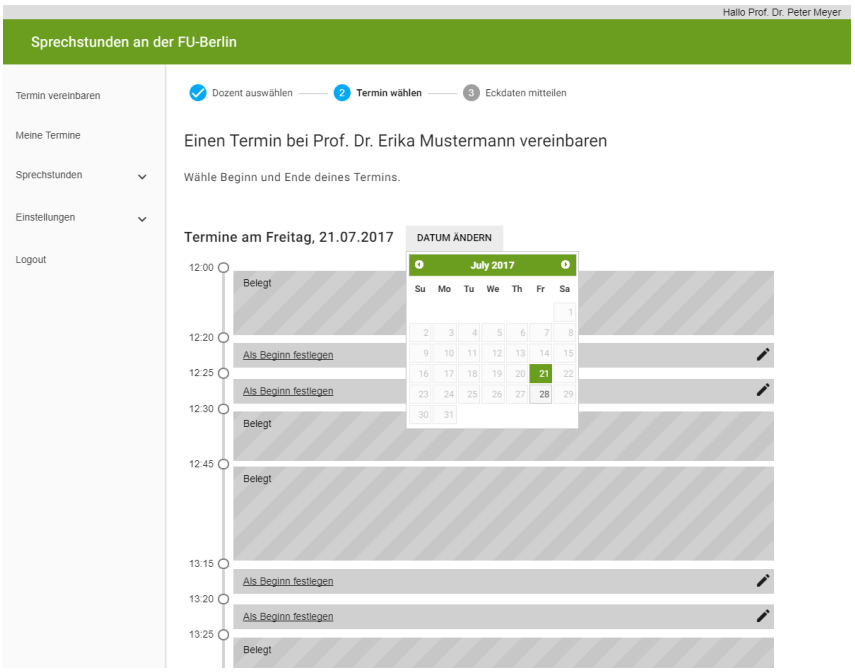

Abbildung 5.4: Schritt 5 - Andere Sprechstunde wählen.

Schritt 5: Verfügbares Datum wählen.
Prüfung der Plausibilität:

- Dem Nutzer wird eine Monatsübersicht mit verfügbaren Terminen angezeigt. Aufgrund dieser Ansicht und seiner Absicht das Datum zu ändern sollte er versuchen ein anderes Datum aus dieser Übersicht zu wählen.
- Es sind nur Daten hervorgehoben, an denen mindestens eine Sprechstunde abgehalten wird. Bewegt der Nutzer die Maus über ein verfügbares Datum, nimmt der Cursor das Pointer-Icon (Hand) an, wodurch dem Nutzer symbolisiert wird, dass ein Klick eine Aktion auslöst.
- Da der Nutzer bereits die Absicht hat, sich die Sprechstunden eines anderen Tages anzeigen zu lassen und aufgrund der Beschriftung des Buttons, sollte dieser herausfinden können, dass der Klick auf ein Datum die zugehörige Sprechstunde von diesem Tag anzeigen lässt. Weitere Gründe die für den Erfolg des Nutzers sprechen sind die optisch hervorgehobenen Daten und der sich verändernde Cursor, der das Pointer-Icon annimmt (Hand). Aus Erfahrung wissen die Nutzer zudem, dass Sprechstunden in der Regel wöchentlich stattfinden.

- Nach dem Klick auf ein anderes verfügbares Datum, wird der Nutzer zur entsprechenden Sprechstunde weitergeleitet. Zusätzlich ist der Fortschritt anhand der Leiste oben im Bild 5.4 zu erkennen.

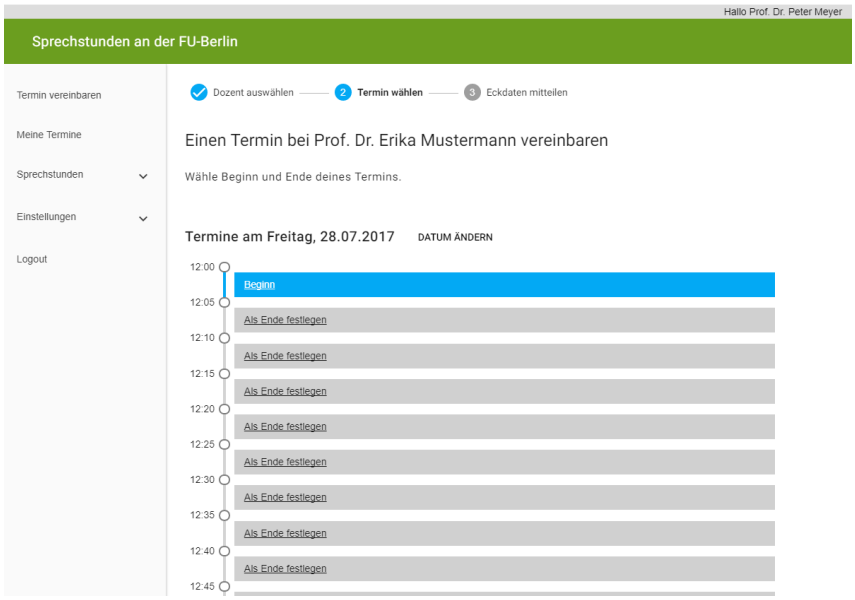

Abbildung 5.5: Schritt 6 - Beginn wählen

Schritt 6: Beginn des Termins wählen.
Prüfung der Plausibilität:

- Dem Nutzer wird eine Übersicht über die Termine in Form einer vertikalen Zeitleiste angezeigt. Der Nutzer möchte per Aufgabendefinition einen Termin mit einer Dauer von 20 Minuten auswählen.

- Verfügbare Slots weisen im ersten Schritt darauf hin, dass diese als Beginn ausgewählt werden können.

- Durch den Hinweistext und den sich ändernden Cursor, wenn dieser über den Zeitslot bewegt wird, wird der Nutzer erkennen, dass eine Aktion auf dem Zeitslot verfügbar ist. Durch den Hinweistext ist der Nutzer in der Lage zu erkennen, dass ein Klick auf den Eintrag diesen als Beginn des Termins markiert.

- Nach dem Klick auf einen verfügbaren Zeitslot erkennt der Nutzer, dass dieser als Beginn markiert ist. Das wird zusätzlich durch seine neue Beschriftung, der blauen Hinterlegung und an dem Hinweistext für andere Zeitslots deutlich, da diese nun als Ende ausgewählt werden können.

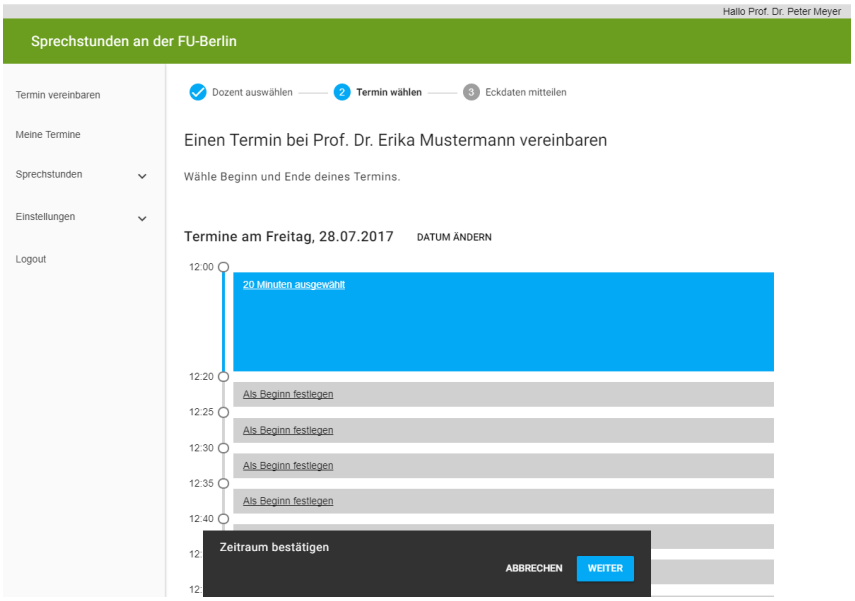

Abbildung 5.6: Schritt 7 - Ende des Termins wählen.

Schritt 7: Ende des Termins wählen.
Prüfung der Plausibilität:

- Dadurch, dass der Beginn des Termins bereits gewählt ist, sollte der Nutzer das Ende wählen wollen. Spätestens durch den Hinweistext „Als Ende markieren“ sollte dem Nutzer klar werden, dass diese Aktion notwendig ist.

- Durch den Hinweistext und seine Erfahrung aus dem letzten Schritt, sollte der Nutzer erkennen, dass es möglich ist einen Slot anzuklicken und dass diese Aktion das Ende des Termins festlegt.

- Nachdem der Nutzer Beginn und Ende ausgewählt hat, wird er gefragt, ob er mit seiner Auswahl fortfahren möchte. Daran erkennt er seinen Fortschritt. Durch die Bestätigung des Weiter-Buttons wird er zu Schritt 8 weitergeleitet. Die blaue Farbgebung lässt den Nutzer den Zusammenhang zwischen seinem gewählten Termin und dem Weiter-Button erkennen.

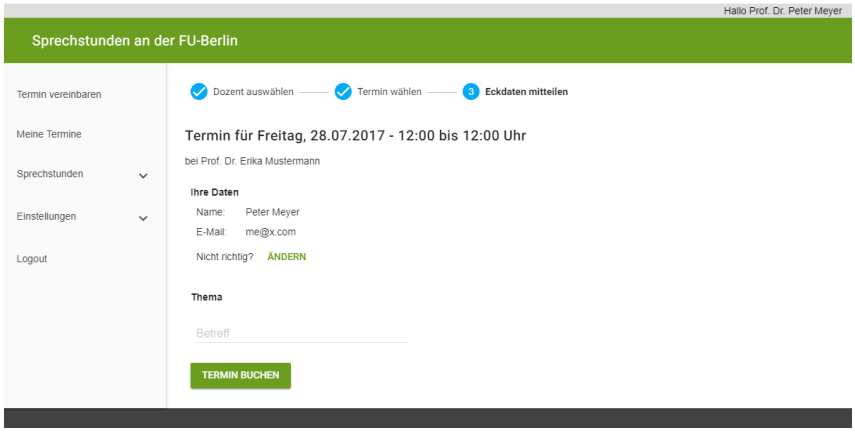

Abbildung 5.7: Schritt 8 - Eingabe des Themas, das besprochen werden soll.

Schritt 8: Betreff / Thema angeben.
Prüfung der Plausibilität:

- Der Nutzer sieht anhand der leeren Eingabezeile, dass ein Thema gefordert ist. Er sollte dem Dozenten mitteilen, warum er einen Termin für die Sprechstunde reserviert.
- Die leere Eingabezeile mit dem Hinweis Betreff und dem hervorgehobenen Text oberhalb sollten den Nutzer auf die Möglichkeit der Eingabe eines Themas hinweisen. Der Nutzer sollte deshalb auch verstehen, wofür dieses Feld gedacht ist. Der Anschließende Klick auf Termin buchen ist ebenfalls verständlich.
- Nach dem Klick auf Termin buchen, wird dem Nutzer eine neue Seite mit dem Hinweis „Terminreservierung erfolgreich“ angezeigt. Daran sollte der Nutzer erkennen, dass seine Terminreservierung erfolgreich durchgeführt worden ist.

Ergebnisse des Cognitive Walktrough

In der exemplarischen Untersuchung von Aufgabe 1 sind einige Schwachstellen aufgefallen. Im ersten Schritt wurde festgestellt, dass das Sakai-System von den Studierenden regelmäßig als KVV bezeichnet wird. Das System ist auf offiziellen Seiten der Universität ebenfalls so benannt und die Abkürzung ist sogar in der URL enthalten. Aus diesen Gründen sollte hier nachgebessert werden. In einer späteren Version wird dort „KVV / Sakai“ stehen. Auf der Seite, auf der nach den Dozenten gesucht werden kann ist aufgefallen, dass der Filter nicht sinnvoll benannt ist. Hier wurde eine Änderung in „Fachbereich“ vorgenommen. Der Button um das Datum zu ändern, ist in der Ansicht des darauffolgenden Schrittes untergegangen. Dieser wird durch eine graue Hintergrundfarbe in Zukunft deutlich besser zu sehen sein.

In vorangegangenen Walkthroughs sind durchaus auch größere Änderungen vorgenommen worden. In Abbildung 5.8 sind zwei verschiedene Designs dargestellt. Das untere stammt aus einem sehr frühen Prototyp. Das festgestellte Problem ist das fehleranfällige Ablesen der Zeit. Wird der belegte Zeitraum genauer betrachtet, könnte fälschlicherweise angenommen werden, dass es sich um einen Termin mit einer Dauer von 10 Minuten handelt, da der markierte Bereich um 10:30 Uhr beginnt und um 10:40 Uhr aufhört. Korrekt ist allerdings eine Dauer von 15 Minuten, da jede Zeile die Dauer von fünf Minuten repräsentiert. Um dieses Problem zu beheben wurde das Design oben in der Abbildung eingeführt. Die Zeiten sind in dieser Ansicht zwischen den Zeitslots dargestellt. Ein Ablesen ist in diesem Fall viel leichter möglich. Bei der Implementierung dieses Interfaces ergaben sich allerdings Probleme bei der Interaktion mit den Elementen. Durch unsichtbare Überlagerungen kam es gelegentlich zu Fehleingaben. Zudem ist für dieses Design keine Auswahl eines Zeitraums angedacht gewesen, sondern lediglich die Angabe des Beginns. Erst im nächsten Schritt wurde die Angabe der Dauer des Termins verlangt, was zur Folge hatte, dass die Nutzer nicht mehr wussten, wie viel Zeit maximal zur Verfügung gestanden hätte.

Wenn der Cognitive Walkthrough weniger formal und dafür häufiger während der gesamten Entwicklung angewandt wird, lassen sich grobe Fehler leicht vermeiden. Gerade vor einem Usability Test ermöglicht diese Methode weitere Fehler zu beseitigen. Dadurch können in dem nachfolgenden Test mit echten Nutzern andere Fehler aufgedeckt werden, die sich durch den Cognitive Walkthrough weniger leicht finden lassen.

5.1.2 Heuristische Evaluation

Die heuristische Evaluation ist eine informelle Methode der Usability-Analyse, in der Experten zu ihrer Meinung zu einem Interface befragt werden [NM90]. In der Regel evaluieren diese Experten das gegebene Interface anhand bestimmter Heuristiken, wie die neun von Molich und Nielsen [MN90]:

- Simple and Natural Language
- Speak the User's Language
- Minimize the User's Memory Load
- Be Consistent
- Provide Feedback
- Provide Clearly Marked Exits
- Provide Shortcuts

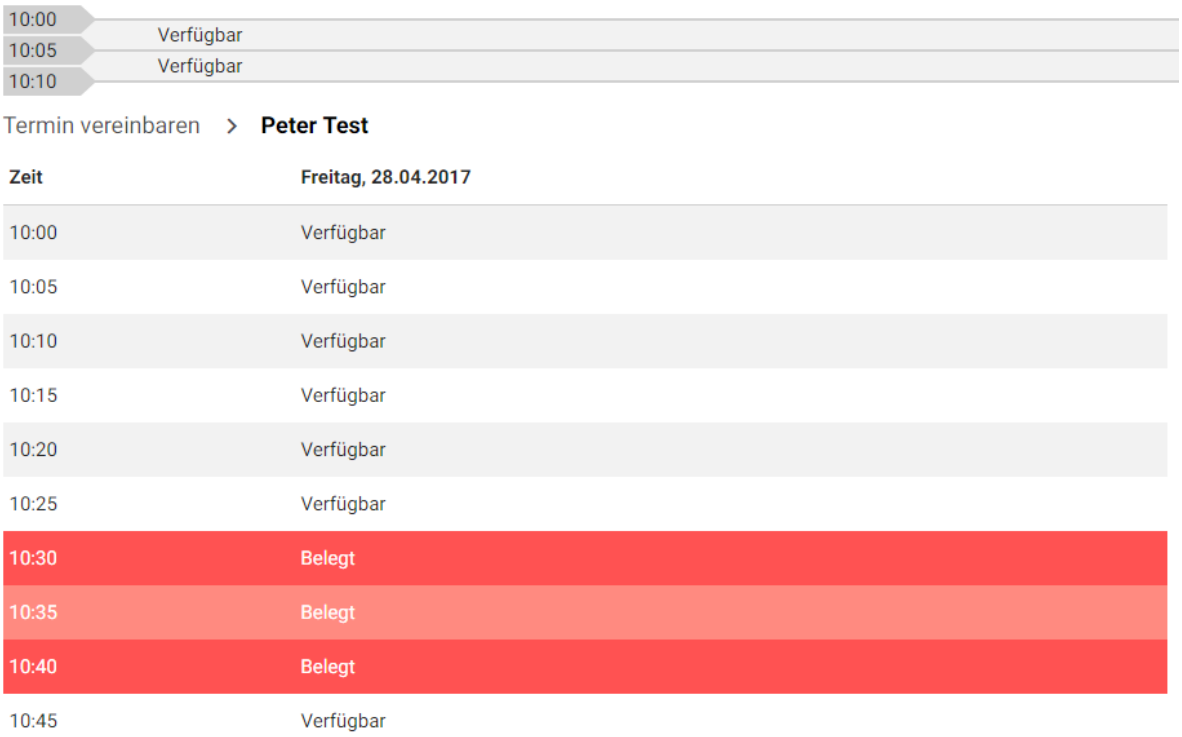

Abbildung 5.8: Abwägung verschiedener Möglichkeiten der Terminauswahl beim High-Fidelity Prototyping. Während einer der ersten Tests wurde festgestellt, dass in der unteren Ansicht die Dauer eines Termins für die Nutzer nicht sofort offensichtlich ist. Der Nutzer ist im ersten Moment davon ausgegangen, dass der Termin im unteren Bereich der Grafik zehn Minuten beträgt und die Zeit von 10:30 Uhr bis 10:40 Uhr belegt. Tatsächlich ist der Termin bis 10:45 Uhr gebucht. In der oberen Ansicht sind die Uhrzeiten daher zwischen den Zeitabschnitten um diese optisch voneinander abzuheben.

- Provide Good Error Messages
- Error Prevention

Die verletzten Heuristiken werden im Folgenden mit den Hinweisen der Experten dargestellt. Sie sind nach verletzter Heuristik gruppiert.

- Simple and Natural Language

  – Belegte Termine könnten besser hervorgehoben werden (2)

    * Gesonderte farbliche Markierung belegter Termine.

  - Beim Ändern des Datums sind verfügbare Tage nicht klar genug herausgestellt (1)
    * Gesonderte Hervorhebung.

  - Beim Auswählen des Zeitraums eines Termins wird nicht klar kommuniziert, dass ein Dozent eine zeitliche Beschränkung für die Dauer eines Termins vorgegeben hat. (2)
    * Die maximale Dauer eines Termins sollte angegeben werden.

  - Die ausgewählten, blau markierten Zeitslots sollten auch abgewählt werden können wenn erst einer markiert ist. (2)
    * Dem Nutzer könnte die Möglichkeit durch ein Kreuz innerhalb des blauen Kastens geboten werden.

- Speak the user's language

  - Die Texte wechseln in der Sprache häufig zwischen Englisch und Deutsch. (3)
    * Englische Inhalte auf Deutsch übersetzen. Zusätzlich eine komplette englische Version anbieten.

  - In der Terminübersicht gibt es eine Spalte mit dem Titel „bei“. Das könnte etwas zu umgangsprachlich sein. (1)
    * Alternativ könnte dort Dozent, Professor oder Name stehen. Konsistenz beachten.

- Minimize the User's Memory Load

  - Der Button „Termin vereinbaren“ ist auch verfügbar, wenn der Dozent keine Termine anbietet. (2)
    * Dieser könnte deaktiviert werden, wenn keine Termine verfügbar sind.

  - Beim Anlegen von Sprechstunden sollte das Format für die Zeit in den Feldern Beginn und Ende vorgegeben werden. Eine Eingabe von Buchstaben sollte nicht möglich sein. (3)
    * Ein Time-Picker wäre eine sinnvolle Ergänzung.

- Be Consistent
  - An einer Stelle wird für die Buttons eine andere Farbe verwendet. (1)
    - Die Farbe entsprechend angleichen.
  - Unter Verwendung anderer Browser verschieben sich einige Icons. (1)
    - Die Seite sollte auch für die Verwendung mit den entsprechenden Browsern optimiert werden.
- Provide Clearly Marked Exits
  - Bei der Eingabe eines Themas und der Buchung des Termins fehlt eine Abbruchmöglichkeit. (2)
    - Dem Nutzer sollte ein Abbrechen-Button zur Verfügung gestellt werden.
- Provide Good Error Messages
  - Es fehlen Fehlermeldungen, wenn bei der Buchung des Termins ein Feld leer gelassen wird oder sie sind unverständlich. (2 bis 4)
    - Dem Nutzer sollte ein Abbrechen-Button zur Verfügung gestellt werden.

5.1.3 Usability Test

Während der Entwicklung wurde das Feedback von Nutzern in regelmäßigen Abständen informell eingeholt. Zusätzlich wurden zwei Usability-Tests geplant und durchgeführt. Für diese Tests wurden Ziel, Teilnehmer, eingesetzte Methoden, die Aufgaben, die Testumgebung und die zu erfassenden Daten vorher festgelegt.

Ziel

Ziel des Tests ist es herauszufinden, ob Nutzer der Anwendung dazu in der Lage sind Termine zu buchen, zu ändern und abzusagen. Weiterhin sollten potentielle Schwachstellen herausgearbeitet werden. Benchmarking ist nicht Teil des Tests, da sich die Nutzer mehr Zeit lassen sollen und für Rückfragen nach dem Test zur Verfügung stehen sollen.

Teilnehmer

Die Hauptzielgruppe für die reguläre Buchung von Terminen sind die Studierenden. Aus diesen wurden Kandidaten nach bestimmten Kriterien gewählt. Die Probanden sollten verschiedenen Alters und Geschlechts sein. Der Test wird mit jedem Nutzer einzeln durchgeführt und teilt sich in zwei Durchläufe mit jeweils drei Kandidaten. Nach dem ersten Durchgang wurden die gefundenen Fehler behoben.

Eingesetzte Methoden

Zu Beginn des Tests werden die Nutzer dazu aufgefordert laut zu denken. Dadurch sollen auch kleinere Probleme erkannt werden, über die der Nutzer „stolpert“. Bildschirminhalte und Ton werden aufgezeichnet, um anschließend eine genaue Analyse vornehmen und dokumentieren zu können.

Aufgaben

Alle Probanden sollen die selben Aufgaben mit den gleichen Voraussetzungen durchführen. Um dieses Ziel zu erreichen wird eine Liste mit Aufgaben erstellt, die die Ziele, Erwartungen, einzelnen Schritte, die geschätzte benötigte Zeit und Tipps an den Nutzer zusammenfassen. Dadurch wird sichergestellt, dass alle Nutzer die gleichen Tipps erhalten, falls nötig. Fragen zu bestimmten Entscheidungen die ein Proband getroffen hat werden nach dem Test gestellt. Solche Fragen betreffen etwaige Abweichungen vom angedachten Weg und werden nach dem Test spontan formuliert.

Test Umgebung

Die Nutzer führen den Test an einem mobilen Laptop aus. Zur Vermeidung von Eingabefehlern über das Touchpad ist eine Maus angeschlossen. Die Probanden können selbst entscheiden welche Eingabegeräte verwendet werden sollen. Zudem ist eine Software installiert, die das Aufzeichnen des Bildschirminhalts ermöglicht (Screencast). Die Anwendung ist lokal installiert. Die URL ist für diese Tests immer gleich.

Tests, die über das Internet durchgeführt werden, nutzen Software die den Bildschirminhalt und Ton übertragen. Sie werden ebenfalls aufgezeichnet. Eine besondere Voraussetzung ist, dass die Software auf dem Gerät des Probanden installiert werden kann oder bereits installiert ist.

Für beide Fälle werden Passwort und Nutzername gestellt. Nach einem Test wird das System zurückgesetzt und damit sichergestellt, dass die Anwendung für alle Nutzer gleich aussieht. Insbesondere sind die selben Termine belegt und frei, die gleiche Anzahl an Terminen verfügbar und die gleiche Menge an Dozenten im System zu finden.

Task 1: Vereinbare einen Termin bei Dozent Prof. Dr. Peter Meyer mit einer Länge von 20 Minuten.	
Ziel	Der Nutzer hat einen Termin zur angegebenen Zeit beim angegebenen Dozenten reserviert.
Erwartungen / Annahmen	Diese Aufgabe sollte dem Nutzer leicht fallen. Probleme könnten bei der Schreibweise des Namens auftreten. Zudem könnte der „Anderes Datum“ Button schwer zu finden sein.
Schritte:	1. Dozent auswählen oder Klick auf Login 2. Eingabe von Nutzername und Passwort 3. Dozent auswählen wenn noch nicht in Schritt 1 4. Da alle Termine belegt sind auf Datum ändern klicken 5. Verfügbares Datum auswählen 6. Beginn auswählen 7. Ende auswählen 8. Zeitraum bestätigen (Weiter-Button) 9. Thema eingeben 10. Termin buchen - Button
Geschätzte Dauer:	Ca. 1 ½ Minuten.
Tipps:	Der Nutzer bekommt Nutzernamen, Passwort und URL gestellt. Keine weiteren Tipps.

Tabelle 5.1: Aufgabe 1 - Terminvereinbarung - Tabelle in Anlehnung an [MB]

Task 2: Ändere das von dir angegebene Thema für deinen gemachten Termin.	
Ziel	Der Nutzer hat den Titel der zuvor vereinbarten Sprechstunde geändert.
Erwartungen / Annahmen	Diese Aufgabe sollte dem Nutzer leicht fallen.
Schritte:	1. Klick auf Login 2. Eingabe von Nutzername und Passwort 3. Klick auf „Meine Termine“ 4. Stift-Button anklicken 5. Neuen Titel eingeben 6. Ändern-Button anklicken
Geschätzte Dauer:	Ca. 1 ½ Minuten.
Tipps:	Der Nutzer bekommt Nutzernamen, Passwort und URL gestellt. Keine weiteren Tipps.

Tabelle 5.2: Aufgabe 2 - Titel bearbeiten - Tabelle in Anlehnung an [MB]

Task 3: Sage den von dir gemachten Termin ab.	
Ziel	Der Nutzer hat seinen Termin abgesagt.
Erwartungen / Annahmen	Diese Aufgabe sollte dem Nutzer leicht fallen.
Schritte:	1. Klick auf Login 2. Eingabe von Nutzername und Passwort 3. Klick auf „Meine Termine“ 4. Mülleimer-Button anklicken
Geschätzte Dauer:	Ca. 1 ½ Minuten.
Tipps:	Der Nutzer bekommt Nutzernamen, Passwort und URL gestellt. Keine weiteren Tipps.

Tabelle 5.3: Aufgabe 3 - Termin absagen - Tabelle in Anlehnung an [MB]

Zu erfassende Daten und Maßstäbe

Probleme, die der Nutzer während des Tests hatte, werden dokumentiert. Dazu zählen auch Abweichungen vom vorgesehenen Weg. In solchen Fällen sind die Nutzer zu befragen warum sie abgewichen sind. Die Art der Abweichung ist im Screencast dokumentiert und wird anschließend in Schriftform zusammengefasst.

Report und Präsentation

Die folgenden Reports umfassen stichpunktartig die erkannten Probleme. Die Stellungnahme des Nutzers ist in diesen Stichpunkten enthalten. Verbesserungsvorschläge wurden ebenfalls notiert.

Früher High-Fidelity-Prototyp

Der erste Test ergab einige Probleme. Teilweise sind diese bereits durch das frühe Stadium abzusehen gewesen. Die folgenden Stichpunkte geben eine kurze Zusammenfassung:

- Die belegten Termine sind optisch zu schwach von freien Terminen abgegrenzt
  - Diese Ansicht wurde in einer späteren Version komplett überarbeitet, da sich herausgestellt hat, dass es noch weitere Probleme damit gab.
- Bei der Eingabe von Thema und Dauer, die *nach* der Terminübersicht angezeigt wird, ergibt sich das Problem, dass sich mehrere Nutzer über die Limitierung der Dauer wundern. In der vorigen Ansicht wäre zu erkennen gewesen, dass zu einer bestimmten Zeit ein weiterer Termin gebucht war.

  - Diese Ansicht wurde in einer späteren Version komplett überarbeitet, da sich herausgestellt hat, dass es noch weitere Probleme damit gab.

- Bei der Angabe der Dauer fehlt die Zeiteinheit Minuten
  - Die explizite Angabe der Länge des Termins wird in der späteren Lösung durch die Auswahl auf einem Zeitstrahl ersetzt.

Später High-Fidelity-Prototyp
Die Usability-Probleme aus dem vorangegangenen Test wurden in diesem Prototyp weitestgehend behoben. Insbesondere wurde die Übersicht über gebuchte und freie Zeiträume und die Auswahl des Zeitraums deutlich verbessert. Die in dem neuen Test gefundenen Probleme gehen tiefer ins Detail und sind auch für diesen Test nachfolgend in Stichpunkten aufgeführt:

- Der Ändern-Dialog von Terminen sollte auch die Möglichkeit zum Löschen geben.
  - Ein entsprechender Button wird nachgebessert.

- Ein Nutzer hat zunächst versucht die Punkte der Zeitleiste zu verschieben, um die Auswahl zu vergrößern.
  - Das es diese Möglichkeit nicht gibt hat der Nutzer innerhalb von Sekunden verstanden und die richtige Aktion vorgenommen.

- Ein Nutzer hat im Ändern-Dialog zunächst die Möglichkeit übersehen, dass das Thema veränderbar ist.
  - Diese Problematik erfordert weitere Beobachtung. Eine offensichtliche Lösung für dieses Problem gibt es nicht.

- Ein Nutzer findet den Pop-Up-Button am unteren Ende der Seite verwirrend. Ein anderer Nutzer hätte statt „Weiter“ zunächst auf „20 Minuten ausgewählt“ geklickt.
  - Eine mögliche Lösung ist es, den Button in die Leiste zu integrieren. Durch die Doppelbelegung der Leiste mit zwei Funktionen wird von dieser Lösung allerdings ein negativer Effekt erwartet. Stattdessen wird versucht das Pop-Up markanter zu gestalten. Ein weiterer Test wird zeigen, ob das den gewünschten Effekt erzielt.

- Ein Nutzer hatte Probleme mit der Schreibweise von Meyer.
  - Eine mögliche Lösung für dieses Problem könnte die phonetische Suche sein, die ähnlich klingende Wörter in die Suchergebnisse aufnimmt.

- Einem Nutzer fehlt Feedback beim Löschen seines Termins.

  - Eine entsprechende Meldung wird nachgebessert.

- Ein Nutzer möchte gerne vor dem Eintragen eines Termins wissen, ob dieser danach noch bearbeitet oder abgesagt werden kann. Ein entsprechender Hinweis würde sein Gefühl der Sicherheit bei der Nutzung der Anwendung erhöhen.
  - Ein entsprechender Hinweis wird eingesetzt. Dazu muss zunächst die richtige Stelle gefunden werden.

- Ein Nutzer hätte beim Anklicken des Logos erwartet, dass dieses auf die Startseite der Anwendung verweist anstatt auf die der Universität.
  - Der Link wird entsprechend geändert.

5.2 Summative Evaluation

In diesem Abschnitt geht es um die Bewertung der Anwendung in Hinsicht auf die Usability kurz vor Abgabe dieser Arbeit. Die Usability setzt sich nach DIN EN ISO 9241-11 aus drei wichtigen Teilen zusammen: Effektivität, Effizienz Nutzerzufriedenheit [Wik16b]. Für die summative Evaluation wurde ein Usability Test mit 16 Teilnehmern durchgeführt. Dabei galt es sechs verschiedene Aufgaben zu lösen:

- Vereinbare einen Termin bei Prof. Dr. Meyer mit einer Länge von mindestens 20 Minuten um deine Bachelorarbeit zu besprechen.
- Du hast dich vertan und wolltest eigentlich deine Masterarbeit besprechen. Ändere den Grund des Termins.
- Leider ist etwas dazwischengekommen. Verschiebe deinen Termin auf einen beliebigen anderen Zeitpunkt.
- Du kannst für unbestimmte Zeit leider keine Termine wahrnehmen. Sage deinen Termin ab.
- Lege eine Sprechstunde immer dienstags von 10-12 Uhr im gesamten August an.
- Sage die Sprechstunde am zweiten Dienstag des Augusts wieder ab (08.08.2017).

Effektivität

Die Effektivität gibt an, ob die Nutzer ihre Aufgaben erfolgreich durchgeführt haben. In allen durchgeführten Tests sind die Nutzer in der Lage gewesen alle ihre Aufgaben erfolgreich und ohne Hilfe durchzuführen. Es gab keinen Test in dem ein Nutzer das Ziel nicht erreicht hat. Der Quotient aus erfolgreichen Versuchen und der Gesamtzahl der Versuche ergibt eine Completion Rate von 100%.

Effizienz

Die Effizienz wird meist anhand der benötigten Zeit im Verhältnis zum Erfolg bewertet. In diesem Fall wurde aufgrund fehlender Vergleichswerte darauf verzichtet. Die Terminvereinbarung wurde im Durchschnitt in rund 2 Minuten und 36 Sekunden erfolgreich durchgeführt. Während der Evaluation hat sich gezeigt, dass im Hinblick auf die Effizienz noch nachgebessert werden kann. So haben zwei der 16 Teilnehmer 5,0 und 5,2 Minuten benötigt, während die restlichen Teilnehmer einen Durchschnittswert von unter 2 Minuten erreichten. Der Grund dafür ist, dass die Schaltfläche zum Ändern des Datums nicht gefunden oder der nachfolgende Monat nicht betrachtet wurde. Daher sollte zusätzlich durch Pfeile, jeweils links und rechts vom Datum, symbolisiert werden, dass es weitere Sprechstunden gibt und ein schnellerer Wechsel zwischen diesen ermöglicht werden.

Nutzerzufriedenheit

Der dritte Bestandteil ist die Nutzerzufriedenheit, die mit Hilfe der System Usability Scale (SUS) bestimmt wird. SUS ist ein Fragebogen, der zehn standardisierte Fragen umfasst. Unter Zuhilfenahme von Likert-Skalen wird für jede Aussage von den Probanden bestimmt, inwiefern sie zustimmen. Das Ergebnis der SUS ist eine Punktzahl die von 0 bis 100 Punkten reichen kann [Bro86]. Ab etwa 72 Punkten kann von einer guten Nutzerzufriedenheit gesprochen werden [BKM09].

Die Teilnehmer wurden nach Durchführung des Usability Tests gebeten, den Fragebogen auszufüllen. Die Abbildung 5.9 zeigt die erreichten Punktzahlen für jede einzelne Befragung. Insgesamt ergibt sich ein Durchschnittswert in Höhe von 88,75 Punkten.

5.3 Zusammenfassung der Evaluation

In diesem Kapitel wurden drei verschieden Formen der Evaluation gezeigt. Während der formativen Evaluation wurde ein Cognitive Walkthrough durchgeführt. Dieser hat sich als schwierig erwiesen. Während einige Probleme erkannt wurden sind viele andere unerkannt geblieben. Gerade aus Sicht der Entwickler einer Software fallen Probleme mit dem Interface weniger auf, da ihre Benutzung für diese sehr klar ist.

Die heuristische Evaluation gibt klare Punkte vor, anhand derer die Usability überprüft werden soll. Das hilft im Vergleich zum Cognitive Walkthrough und ist daher sehr effektiv gewesen.

Der Usability Test stellt sich als wichtigstes Instrument zur Evaluation heraus. Zusammen mit der summativen Evaluation wurden insgesamt 24 Tests

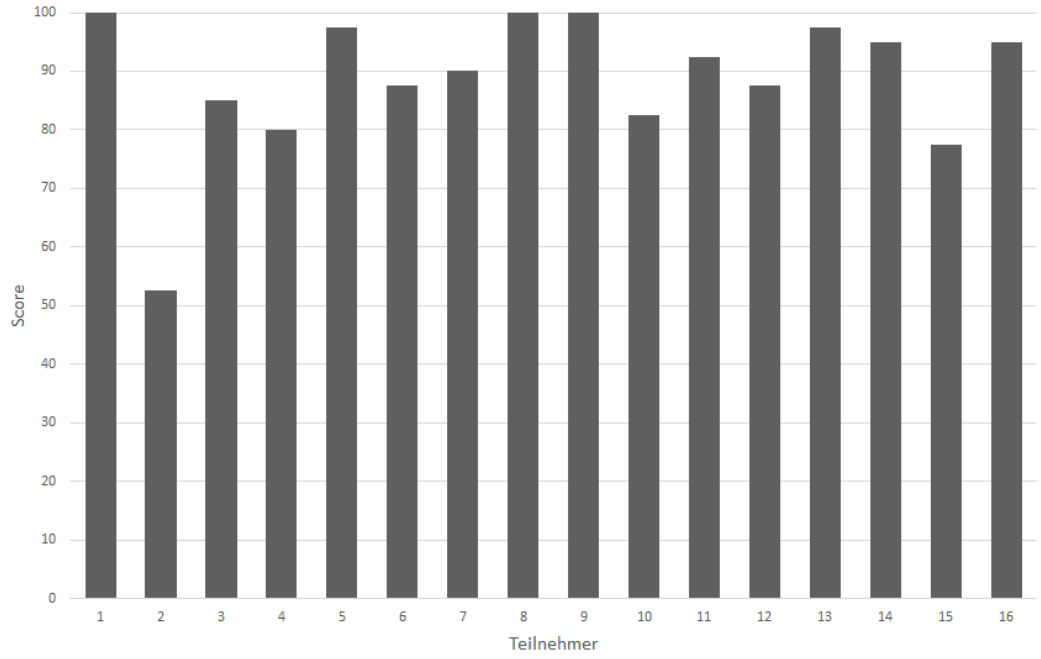

Abbildung 5.9: Nutzerzufriedenheit anhand einer SUS-Umfrage. Punkte von jedem Teilnehmer. Im Durchschnitt wurde ein Wert von 88,75 Punkten erreicht.

durchgeführt. Viele der Tests haben neue Möglichkeiten zur Verbesserung der Software gebracht. Im Vergleich zur heuristischen Evaluation wurden unterschiedliche Arten von Problemen entdeckt. Während Nutzer eher Elemente vermissten oder vorschlugen, wurden bei der heuristischen Evaluation vorhandene Elemente bemängelt. Durch die Vorgabe von Heuristiken, wie Consistency sind beispielsweise unterschiedliche Farben von Buttons bemerkt worden, die die Nutzer anscheinend nicht gestört haben.

6 Zusammenfassung und Ausblick

Abschließend werden die Ergebnisse der Arbeit noch einmal zusammengefasst und mit den gesetzten Zielen verglichen. Zudem wird auf Probleme bei der Umsetzung und mögliche weitere Funktionen für die Anwendung eingegangen.

6.1 Fazit

Die konsequente Verwendung der Methoden des User Centered Design hilft dabei die Usability zu steigern. Maßnahmen, die unter das User Modeling fallen, sind gerade im Hinblick auf die Kommunikation im Team noch einmal wirkungsvoller. Sie erfüllen ihren Zweck allerdings auch, wenn dem eigenen Gedächtnis auf die Sprünge geholfen werden soll. Durch das frühzeitige Feedback von Nutzern können größere Fehler vermieden und die gewünschte Effektivität erreicht werden. Die Ergebnisse der später folgenden Evaluationen sorgen durch vergleichbare Werte dafür, dass die Nutzerzufriedenheit und Effizienz im späteren Verlauf der Entwicklung verbessert werden können. Da der Aufwand, der mit der Durchführung dieser Techniken verbunden ist, nicht unerheblich ist, sollten Entwickler und Designer für sich selbst herausfinden, welche Kombination der Methoden für sie den größten Vorteil bringt.

In dem Entwicklungsprozess hat sich die Review Analysis als sehr effektiv erwiesen. Insbesondere auch die Überlegung, wie Nutzer ohne technische Hilfsmittel vorgehen würden und die Contextual Inquiry, hatten großen Einfluss auf die Abfolge der notwendigen Interaktionen. Die Schritte, zunächst einen Dozenten auszuwählen, dann einen Zeitraum festzulegen und das Thema anzugeben und zum Schluss eine Bestätigung angezeigt zu bekommen, stimmen mit denen einer Terminvereinbarung ohne technisches Hilfsmittel überein. Die Contextual Inquiry hat ergeben, dass die Studierenden auf die gleiche Weise vorgehen, wenn sie Termine über momentan eingesetzte Hilfsmittel vereinbaren.

Der Cognitive Walkthrough hat sich als geeignete Methode erwiesen um einen Prototyp auf einen Usability Test vorzubereiten. Gerade wenn die Entwickler diesen Walkthrough selbst vornehmen ersetzt dieser den Usability Test nicht. Während dieser Tests haben sich Probleme ergeben, die für einen Entwickler nicht erkennbar gewesen wären. Die Möglichkeit, die Nutzer nach dem Test zu ihren Gedanken und Problemen zu befragen, hat für ein tieferes Verständnis gesorgt. Darauf sollte nicht verzichtet werden, wenn ein solcher Test durchgeführt wird. Die Nutzer laut denken zu lassen, ist die Grundlage sol-

che Fragen gezielt zu stellen. Einigen Testkandidaten ist es schwerer gefallen, das laute Denken durchzuführen, so dass sie regelmäßig darauf aufmerksam gemacht werden mussten.

Die System Usability Scale gibt ein praktisches Feedback über die Meinung der Nutzer. Allerdings hat sich durch Nutzerfeedback ergeben, dass es hin und wieder zu Fehleingaben kam, da sich die Fragen in positiven und negativen Skalen abwechseln.

In diesem Entwicklungsprozess wurden Personas eingesetzt. Diese hatten keinen größeren Vorteil für den Prozess. Sie werden oft dazu eingesetzt die Kommunikation im Team zu verbessern und anderen Teammitgliedern die Anforderungen und Nutzereigenschaften näher zu bringen. Aufgrund des fehlenden Teams konnten diese Vorteile nicht gewinnbringend zur Geltung gebracht werden. Dennoch eigneten sie sich als Gedächtnisstütze und zur Repräsentation der Nutzer in dieser Arbeit.

6.1.1 Zielerreichung

Die Menge der Anforderungen und die Funktionalitäten, die die eingesetzten Lösungen bereits boten wurden zu Beginn der Aufnahme der Arbeit deutlich unterschätzt. Das Ziel, alle Vorteile zu vereinen wurde nicht erreicht. Dennoch ist eine Anwendung entstanden, die einige nützliche Funktionalitäten aufweist und andere bereits in Grundzügen implementiert.

Die Bereitstellung von Sprechstunden unter Angabe eines Titels und einer Beschreibung ist funktionsfähig. Die Beschreibung kann genutzt werden um den Studierenden Hinweise zu geben. Auch regelmäßige Sprechstunden können geplant werden. Zudem wird eine Funktion zur Limitierung der maximal buchbaren Termindauer bereitgestellt. Auch eine Deadline für die Buchung von Terminen ist implementiert. Assistenten kann der Zugriff auf die eigenen Sprechstunden gewährt werden um diese zu verwalten. Termine können von den Assistenten ebenfalls eingetragen werden. Dozenten können über ihr Profil Informationen wie die Adresse ihres Büros und ihren Fachbereich teilen. Gastnutzer können über das System eingeladen werden. So können beispielsweise Projektteams von außerhalb der FU Berlin selbstständig Termine buchen.

Für die Studierenden ist die Buchung und Verwaltung von Terminen möglich. Ein weiterer Vorteil ist die zentrale Anlaufstelle, selbst wenn einige Dozenten ihre alte Lösung behalten wollen. Diese können sie in der Anwendung verlinken. Auch die variable Länge kann zur effizienten Nutzung der vorhandenen Zeit einer Sprechstunde für beide Parteien ein Vorteil sein. Die mögliche Anbindung an das Sakai-System vermeidet die Notwendigkeit zusätzlicher Passwörter.

Die durchgeführte Evaluation hat gezeigt, dass die entstandene Software gebrauchstauglich, aber noch nicht perfekt ist. Gerade im Hinblick auf die Effizienz kann an manchen Stellen noch nachgebessert werden.

6.1.2 Vergleich mit anderen Lösungen

Um die Funktionen der implementierten Software zu vergleichen, wird die Tabelle 2.1 um die neue Lösung und ihre Funktionen ergänzt. In dieser (Tabelle 6.1) ist zu erkennen, dass die neue Software alle wichtigen Funktionen abdeckt. Die E-Mail-Notifications sind noch nicht vollständig implementiert. Eine Warteliste mit Nachrückfunktion ist derzeit nicht geplant.

Funktion / Vorteil	Keine	Mündlich	E-Mail	Wiki	Cgi	Sakai	Neu
Termin vereinbaren	✗	✓	✓	✓	✓	✓	✓
Termin ändern	✗	✓	✓	✓	✗	✓	✓
Länge variabel	✓	✓	✓	✓	✗*	✗*	✓
Beschränkung der Termindauer	✗	✓	✓	✓	?	✓	✓
Buchungsdeadline	✗	✓	✓	✗	?	✓	✓
Terminbestätigung	✗	✗	✓	✗	✗	✓	✗[1]
Automatische Mitteilung bei Terminänderung	✗	✗	✗	✗	✗	✓	✗[1]
Automatische Mitteilung bei Terminabsage	✗	✗	✗	✗	✗	✓	✗[1]
Automatisches Verschieben der Termine abgesagter Sprechstunden	✗	✗	✗	✗	✗	✗	✓
Online-Verfügbarkeit	✗	✗	✓	✓	✓	✓	✓
Zugang ohne VPN	-	-	✓	✗	✓	✓	✓
Warteliste mit Nachrückfunktion	✗	✗	✗	✗	✗	✓	✗
Zugriff für Assistenten	✗	✗	✗	✓	?	?	✓
Terminübersicht für Dozenten	✗	✗	✗	✓	✓	✓	✓
Terminübersicht für Studierende	✗	✗	✗	✗	✗	✗	✓
Termine privat	✗	✓	✓	✗	✓	✓	✓
Direktes Feedback / Keine Wartezeit	✗	✓	✗	✓	✓	✓	✓
Hinweise zur Vorbereitung auf die Sprechstunde	✗	✓	✓	✓	✓	✗	✓

Tabelle 6.1: Vergleich der bisherigen Lösungen mit der neuen Software. *Eine ansatzweise variable Länge kann durch die Buchung mehrerer Zeitslots erreicht werden. [1]Diese Teile sind noch nicht vollständig implementiert.

6.2 Nächste Schritte / Ausblick

Bevor die entstandene Anwendung produktiv eingesetzt werden kann, müssen noch einige offene Punkte bearbeitet werden. Gerade um die Usability für Studenten aus dem Ausland zu erhöhen, sollte die Anwendung zusätzlich auch in englischer Sprache zur Verfügung gestellt werden. Einzelne Seiten sind noch nicht für den Einsatz auf mobilen Endgeräten optimiert.

Auf der technischen Ebene sind ebenfalls Punkte offen geblieben. Zur vollständigen Integration in das bestehende System an dem Fachbereich Mathematik und Informatik, muss eine Schnittstelle geschaffen werden, über die sich die Rechte der Nutzer abgleichen lassen. Es existiert bereits eine Datenbank, mit der sich diese Informationen abgleichen lassen könnten. Das Sakai-System stellt diese der Anwendung zum Zeitpunkt der Abgabe nicht bereit. Um die Anwendung vor Fehlern abzusichern sollten zudem vor dem produktiven Einsatz Softwaretests geschrieben werden.

Für die weitere Optimierung in Hinsicht auf die Usability, könnten weitere Ideen umgesetzt werden:

- Ein Wizard, der den Studierenden bei der Auswahl des richtigen Ansprechpartners hilft. Je nach Thema, das der Studierende besprechen möchte, könnte ihm ein Dozent vorgeschlagen werden. Das könnte auch für die Dozenten von Vorteil sein, die sich ein Amt anhand der Anfangsbuchstaben der Nachnamen der Studierenden teilen. Zum Beispiel könnte Dozent A für die Anrechnung von Studienleistungen für alle Studierenden, deren Nachnamen mit A-M beginnen, zuständig sein.
- Einigen Studierenden ist es unangenehm andere Gespräche zu unterbrechen. Durch eine Rückmeldung in der Anwendung könnte dem Dozenten mitgeteilt werden, dass sein nächster Termin eingetroffen und bereit ist.
- Alternativ könnte eine Benachrichtigung zum Ende des Termins angezeigt werden, unabhängig davon, ob der nächste Teilnehmer bereits anwesend ist.

Literaturverzeichnis

[AMKP04] Chadia Abras, Diane Maloney-Krichmar, and Jenny Preece. User-centered design. *Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications*, 37(4):445–456, 2004.

[BKM09] Aaron Bangor, Philip Kortum, and James Miller. Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *J. Usability Studies*, 4(3):114–123, May 2009.

[Bro86] John Brooke. SUS - A quick and dirty usability scale. *Usability Evaluation in Industry*, 1986.

[Car00] J.M Carroll. Introduction to this special issue on "scenario-based system development". *Interacting with Computers*, 13(1):41–42, 2000.

[Com16] Fachbereich Human Centered Computing. Entwicklung einer Webanwendung für Sprechstundentermine. `https://www.mi.fu-berlin.de/inf/groups/hcc/theses/running/Webanwendung-Sprechstundentermine/index.html`, 2016. [Online; Stand 03. Dezember 2017].

[CQ08] Torkil Clemmensen and Shi Qingxin. What is part of a usability test? In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, pages 3603–3608, New York, NY, USA, 2008. ACM.

[CRC07] Alan Cooper, Robert Reimann, and Dave Cronin. *About Face 3 The Essentials of Interaction Design*. Wiley Publishing, Inc., third edition edition, 2007.

[FUB15] Center für Digitale Systeme (CeDiS) Freie Universität Berlin. Corporate-Design-Handbuch Band II: Webdesign. `http://www.fu-berlin.de/sites/cd/downloads_container/cd_v2-4_band_II_handbuch.pdf`, 06 2015. [Online; Stand 18. Juli 2017, Nur aus dem Netzwerk der FU Berlin erreichbar].

[Gooa] Google Inc. Material Design Guidelines. `https://material.io/guidelines/`. [Online; Stand 18. Juli 2017].

[Goob] Google Inc. Material Design Lite. `https://getmdl.io/started/index.html#license`. [Online; Stand 05. Juli 2017].

[Inc] IMS Global Learning Consortium Inc. Learning tools interoperability. https://www.imsglobal.org/activity/learning-tools-interoperability. [Online; Stand 05. Juli 2017].

[JLI92] Alex P. J. Jarczyk, Peter Löffler, and Frank M. Shipman III. Design rationale for software engineering: A survey. In *Proceedings of the 25th Hawaii International Conference on System Sciences*, 1992.

[Mö10] Sebastian Möller. *Quality Engineering*. Springer, 2010.

[MB] Prof. Dr. Claudia Müller-Birn. User-centered software development. Vorlesungsskript.

[MN90] Rolf Molich and Jakob Nielsen. Improving a human-computer dialogue. *Commun. ACM*, 33(3):338–348, March 1990.

[MR13] Markus D. Flückiger Michael Richter. *Usability Engineering kompakt*. Springer Berlin Heidelberg, 1st edition, 2013.

[NM90] Jakob Nielsen and Rolf Molich. Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, pages 249–256, New York, NY, USA, 1990. ACM.

[Nor16] Don Norman. *The Design of Everyday Things*. Vahlen Franz GmbH, 2016.

[RF10] Michael Richter and Markus D. Flückiger. *Usability Engineering kompakt*. Spektrum Akademischer Verlag, 2010.

[Wik16a] Wikipedia. Formative evaluation — wikipedia, die freie enzyklopädie. https://de.wikipedia.org/w/index.php?title=Formative_Evaluation&oldid=152587563, 2016. [Online; Stand 31. Juli 2017].

[Wik16b] Wikipedia. Gebrauchstauglichkeit (Produkt) — Wikipedia, Die freie Enzyklopädie. https://de.wikipedia.org/w/index.php?title=Gebrauchstauglichkeit_(Produkt)&oldid=159056626, 2016. [Online; Stand 24. Juli 2017].

[Wik16c] Wikipedia. Requirements engineering — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Requirements_engineering&oldid=747258607, 2016. [Online; Stand 08. Februar 2017].

[Wik16d] Wikipedia. Sakai (Software) — Wikipedia, Die freie Enzyklopädie. https://de.wikipedia.org/w/index.php?title=Sakai_(Software)&oldid=158596911, 2016. [Online; Stand 30. Juli 2017].

[Wik17a] Wikipedia. Groovy — Wikipedia, Die freie Enzyklopädie. https://de.wikipedia.org/w/index.php?title=Groovy&oldid=166688253, 2017. [Online; Stand 04. Juli 2017].

[Wik17b] Wikipedia. Nutzerorientierte Gestaltung — Wikipedia, Die freie Enzyklopädie. https://de.wikipedia.org/w/index.php?title=Nutzerorientierte_Gestaltung&oldid=162004057, 2017. [Online; Stand 08. Februar 2017].

[Wik17c] Wikipedia. Wiki — Wikipedia, Die freie Enzyklopädie. https://de.wikipedia.org/w/index.php?title=Wiki&oldid=162230282, 2017. [Online; Stand 08. Februar 2017].

[WRLP94] Cathleen Wharton, John Rieman, Clayton Lewis, and Peter Polson. The cognitive walkthrough method: A practitioner's guide. In R. Mack J. Nielsen, editor, *Usability Inspection Methods*. John Wiley & Sons, 1994.

[Wro14] Luke Wroblewski. Designing for large screen smartphones. https://www.lukew.com/ff/entry.asp?1927, 2014. [Online; Stand 31. Juli 2017].

Appendix

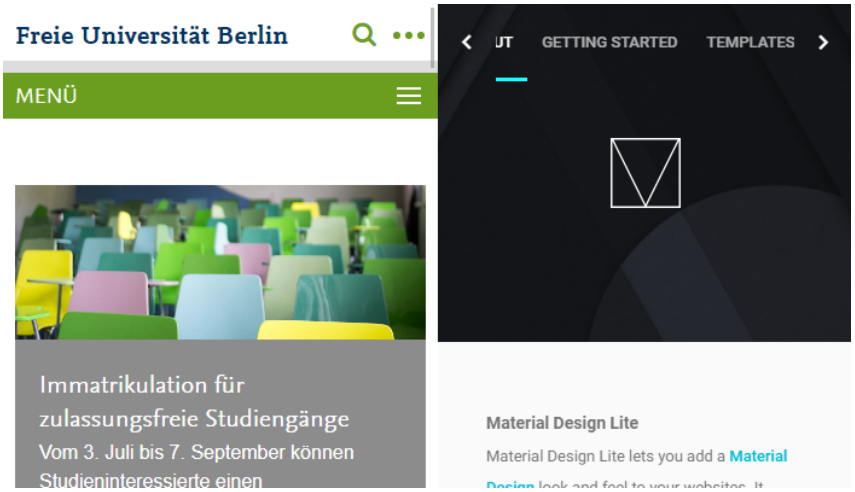

Abbildung 6.1: Vergleich verschiedener mobiler Hauptmenüs - Links ein Screenshot der Lösung auf der Homepage der FU Berlin. Hier lässt sich das Hauptmenü auf- und zuklappen. Symbolisiert wird die Möglichkeit durch das Wort Menü und den so genannten „Burger-Button“ Rechts ein Screenshot der Seite http://getmdl.io mit horizontaler Scrollbar. Die Pfeile symbolisieren die Möglichkeit des Scrollens nach links und rechts (Signifier). Beide Screenshots sind am 18.07.2017 erstellt worden.

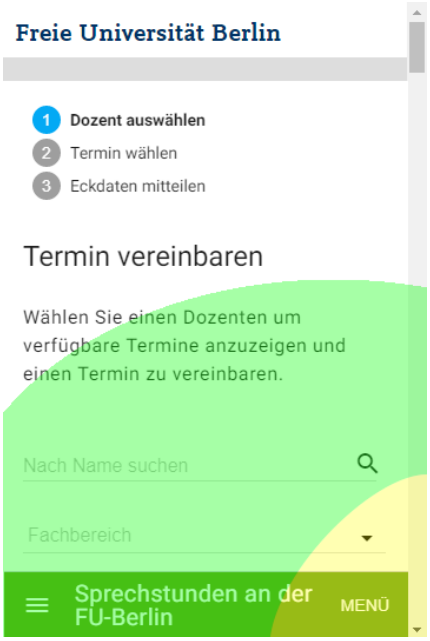

Abbildung 6.2: Entwurf eines Menüs am unteren Seitenrand. Das Menü würde nach oben ausklappen. Damit ist der Vorteil verbunden, dass Smartphonenutzer dieses Menü ohne Probleme erreichen können, auch wenn es sich um ein Gerät mit großem Bildschirm handelt. Die grüne markierte Zone stellt den Bereich dar, der sich auf einem großen Smartphone mit dem Daumen leicht erreichen lässt. Die gelbe Zone ist weniger leicht erreichbar. Die nicht hinterlegte Zone ist ohne Umgreifen nicht erreichbar. Bei kleineren Telefonen oder der Benutzung mit beiden Händen tritt dieses Problem weniger stark oder gar nicht auf. Grafik in Anlehnung an [Wro14]