

DIPLOMARBEIT

Therapeutenverwaltung des medizinisch-technischen Dienstes

Ausgeführt im Schuljahr 2016/17 von:

Design- und Frontend-Funktionalität Lukas GANSTER (LG)	5AHIF-3
iOS- und Android-App Julia KARNER (JK)	5AHIF-12
Datenbank, API und Backend-Funktionalität Patrick TARNOK (PT)	5AHIF-24

Betreuer / Betreuerin:

Mag. Gabriele Haas

Wiener Neustadt, am 3. April 2017

Abgabevermerk:

Übernommen von:

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.

Wiener Neustadt, am 3. April 2017

Verfasser / Verfasserinnen:



Lukas GANSTER (LG)



Julia KARNER (JK)



Patrick TARNOK (PT)

Inhaltsverzeichnis

Eidesstattliche Erklärung	i
Vorwort	vii
Danksagung	viii
Diplomarbeit Dokumentation	ix
Diploma Thesis Documentation	xi
Kurzfassung	xiii
Abstract	xiv
1 Design und Usability (LG)	1
1.1 Motivation	1
1.2 Logo	1
1.2.1 Bildmarken	2
1.2.2 Wortmarken	2
1.2.3 Wort-Bildmarken	3
1.2.4 Fazit	4
1.3 Usability	4
1.3.1 Definition und Abgrenzung	4
1.3.2 Icons	5
2 Frontend (LG)	7
2.1 Motivation	7
2.2 Funktionen der Webapp	7
2.2.1 Unterteilung der Rollen	7
2.2.2 Dienstplan	8
2.2.3 Raumplan	10
2.3 Evaluierung von AngularJS	10
2.3.1 Chancen und Möglichkeiten	11
2.3.2 Vergleichbare Technologien	11
2.3.3 Fazit	13
2.4 Architektur	13
2.4.1 Multiple Page Application (MPA)	14
2.4.2 Single Page Application (SPA)	14
2.4.3 Fazit	16
2.5 Trennung der Komponenten	16

2.5.1	Model-View-Controller	17
2.5.2	Model	17
2.5.3	View	18
2.5.4	Controller	19
2.6	Technologien zur Datenbeschaffung	20
2.6.1	JSON	20
2.6.2	XML	21
2.6.3	YAML	21
2.6.4	Fazit	22
3	Backend (PT)	23
3.1	Einleitung	23
3.2	Evaluierung von PHP	23
3.3	Definition einer Webanwendung	23
3.4	Sicherheitsprobleme von Webanwendungen	24
3.4.1	Technische Ursachen	24
3.4.2	Know-How-bedingte Ursachen	26
3.5	Sicherheitsmechanismen für Webanwendungen	28
3.5.1	Datenbanksicherheit durch die Art der Einbindung	28
3.5.2	Das PDO Objekt	29
3.5.3	Schutz vor Injections	30
3.6	Funktionalität	32
3.6.1	Kommunikation mit dem Frontend	32
3.6.2	Dienstplan	33
4	App (JK)	35
4.1	Einführung	35
4.2	Evaluierung von Ionic	36
4.3	Verwendete Entwicklungsumgebung	36
4.4	Einrichten von Ionic	37
4.4.1	Projektstruktur	37
4.5	Verwendung / Testen	39
4.5.1	Server	40
4.6	Funktionen der App	41
4.6.1	Datenanzeige / Datenbearbeitung	42
4.6.2	Arbeitszeiten-, Abwesenheiten- und Blockadenanzeige	43
4.6.3	Dienstplananzeige	44
4.6.4	Vertretungsverwaltung	45
4.7	Webapplikation zu Ionic-App	46
4.7.1	Kommunikation mit der API	46
4.7.2	Vergleich localStorage zu Cookies	47
4.7.3	UI-Elemente	48
4.7.4	Probleme und Lösungen	49
4.7.5	Vor- und Nachteile dieser Umsetzung	51
4.8	Push Benachrichtungen	51
4.8.1	Allgemeines zu Push Benachrichtungen / Alternativen	51
4.8.2	Anwendung	52
4.8.3	Implementierung	52
5	Datenbank (PT)	57

5.1	Einleitung	57
5.2	Evaluiierung von MySQL	57
5.3	Aufbau der Datenbank	58
5.3.1	Ausgangslage	58
5.3.2	Datenbankstruktur	58
5.3.3	Die „Employees“-Tabelle	58
5.3.4	Die „Groups“-Tabelle	59
5.3.5	Erweiterungen zur „Employees“-Tabelle	59
5.3.6	Erweiterungen zur „Groups“-Tabelle	60
5.4	Stored Procedures (SP)	61
5.5	User-Defined Functions (UDF)	63
5.6	Events	63
5.7	Trigger	63
6	Fazit	65
	Glossary	67
	Literatur	68

Abbildungsverzeichnis

1.1	medtec-Logo als Bildmarke	2
1.2	medtec-Logo als Wortmarke	3
1.3	medtec-Logo als Wort-Bildmarke	3
1.4	Schriftmarke als Logo für Anwendung	4
1.5	Tooltip bei Button	6
2.1	Vergleich von Navigationsleisten der verschiedenen Rollen	8
2.2	Dienstplan-Anzeige	9
2.3	Erstellung des Dienstplans im Frontend	9
2.4	Raumplan-Anzeige	10
2.5	Statistik zur Verwendung von Programmiersprachen auf Client-Seite	11
2.6	Architektur einer MPA	14
2.7	Architektur einer SPA	15
2.8	Model-View-Controller Pattern	17
2.9	View beim Bearbeiten eines Therapeuten	19
3.1	3-Tier-Architektur ¹	24
3.2	Netzwerk- vs Anwendungslayer ²	25
3.3	Ein gewöhnliches Login um sich in einen Account einzuloggen.	31
3.4	SQL-Injection	31
3.5	Syntaxfehler mit Abfrageausschnitt	32
4.1	Übersicht der Technologien und Abhängigkeiten	36
4.2	Projektstruktur eines Ionic-Projekts	37
4.3	App beim Start und Funktionen der App	41
4.4	Datenanzeige und Datenbearbeitung	42
4.5	Abwesenheiten und Arbeitszeiten	43
4.6	Dienstplananzeige	44
4.7	Vertretungsverwaltungsanzeige	45
4.8	Benachrichtigungsübersicht und Anzeige der offenen Benachrichtigungen	54
5.1	Employees & Groups Tabelle	59
5.2	Employees Tabelle & Erweiterungen	60
5.3	Groups Tabelle & Erweiterungen	61

¹Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 2.

²Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 4.

Listings

2.1	Model beim Bearbeiten eines Therapeuten	18
2.2	Vereinfachter Controller beim Bearbeiten eines Therapeuten	19
2.3	Therapeut in JSON	20
2.4	Therapeut in XML	21
2.5	Therapeut in YAML	22
3.1	HTTP-Request	25
3.2	HTTP-Response	26
3.3	Datenbank Verbindungsaufbau	29
3.4	Datenbank Datenabruf	30
3.5	SQL-Abfrage für das vorherige Login	31
3.6	SQL-Abfrage mit Injection	31
3.7	Adminspezifische „GET“-Requests	33
4.1	Implementierung der Unterscheidung der Abmeldungen	46
4.2	Der GET-Request für die Abwesenheiten an den AjaxController	47
4.3	Der Body-Tabellenaufbau für die Abwesenheiten eines Therapeuten	49
4.4	Der JavaScript Code für die Logik des Akkordeons	49
4.5	Zugriffe von außen erlauben	49
4.6	Zuweisung der Email Variable im AjaxController	50
4.7	hideDrawer Funktion	50
4.8	Benötigter Code für das Verhindern der Überlappung der Statusbar auf iPhones	50
4.9	Registration beim Push-Service	53
4.10	Benötigte Variablen zum Senden einer Push-Benachrichtigung	55
4.11	Eigentliches Senden der Benachrichtigung mit cURL	56
5.1	Abfrage aller Therapeuten	61
5.2	Abfrage auf Hauptverantwortlichkeit	62
5.3	Abfrage des Dienstplanes	62
5.4	Funktion zur Bestimmt der Anwesenheit eines Therapeuten	63
5.5	Event zum leeren der „AbsentToday“-Tabelle	63
5.6	Check Zeile innerhalb einer Tabelle zur Überprüfung des Berechtigungslevels	64
5.7	Trigger zur Überprüfung des Berechtigungslevels	64

Vorwort

Die Verwaltung einer medizin-technischen Abteilung in einem Krankenhaus ist eine aufwendige und zeitraubende Angelegenheit, welche durch technische Unterstützung stark vereinfacht werden könnte.

Die medtec Webapplikation mit ihren dazugehörigen Apps ermöglicht dies nun. Eine intuitive und leicht zu bedienende Oberfläche wurde in enger Zusammenarbeit mit leitenden Therapeuten einer solchen Abteilung entwickelt und angepasst. Unsere Webapplikation soll dadurch in der Lage sein, selbst mit geringem technischen Wissen, alle anfallenden Verwaltungsaufgaben der Abteilung abzuwickeln. Mit der App kommt eine leicht zugängliche Schnittstelle dazu, welche für alle angestellten Therapeuten verfügbar ist. Diese ermöglicht dem Personal den Abruf ihrer aktuellen Dienstzeiten und Aufgaben, sowie die Möglichkeit Vertretungen über die eigenen Abwesenheiten zu informieren.

Danksagung

An dieser Stelle möchten wir uns bei all denjenigen bedanken, die uns während des gesamten Verlaufs der Diplomarbeit unterstützt haben.

Danken möchten wir in erster Linie unserer Betreuerin Frau Professor Mag. Gabriele Haas für ihre ausgiebige Unterstützung und die gute Zusammenarbeit während dieser Diplomarbeit. Durch stetig kritisches Hinterfragen und konstruktive Kritik verhalf sie uns zu einem sauberen Ergebnis.

Außerdem gilt unser Dank unserem Abteilungsvorstand Herrn Professor Dipl.-Ing. Felix Schwab und der Auftraggeberin Klaudia Tarnok, die uns während unserer Diplomarbeit immer zur Seite standen.

Nicht zuletzt gebührt unseren Eltern großer Dank, denn sie haben uns nicht nur durch alle Höhen und Tiefen dieses Prozesses unterstützt, sondern uns auch die Jahre davor auf unserem Bildungsweg begleitet.

Diplomarbeit Dokumentation

Namen der Verfasser/innen	Ganster Lukas Karner Julia Tarnok Patrick
Jahrgang Schuljahr	5AHIF 2016 / 17
Thema der Diplomarbeit	Therapeutenverwaltung des medizinisch-technischen Dienstes
Kooperationspartner	

Aufgabenstellung	Realisierung einer Webanwendung, sowie iOS- und Android App zur Verwaltung der medizin-technischen Abteilung des Landeskrankums Wiener Neustadt.
------------------	--

Realisierung	Es wurde eine Webanwendung inklusive Server mit Datenbankbindung realisiert. Sowie eine iOS- und Android App basierend auf den Framework Ionic. Die Webanwendung dient der Verwaltung des gesamten Systems, während die Apps der Verwaltung des einzelnen Therapeuten dienen.
--------------	---

Ergebnisse	Funktionsfähige Webapplikation Relationale und konsistente Datenbank Funktionsfähige iOS- und Android-App Dokumentationen und Benutzerhandbuch
------------	---

Typische Grafik, Foto
etc. (mit Erläuterung)

Logo der App

medtec
MEDIZIN-TECHNISCHE ADMINISTRATION

Teilnahme an
Wettbewerben,
Auszeichnungen

Möglichkeiten der
Einsichtnahme in die
Arbeit

HTBLuVA Wiener Neustadt
Dr.-Eckener-Gasse 2
A 2700 Wiener Neustadt

Approbation


Prüfer

Abteilungsvorstand

(Datum, Unterschrift)

Mag. Gabriele Haas

AV Dipl.-Ing. Felix Schwab

	COLLEGE OF ENGINEERING WIENER NEUSTADT
	Department: Informatik

Diploma Thesis Documentation

Authors	Ganster Lukas Karner Julia Tarnok Patrick
Form	5AHIF
Academic Year	2016 / 17
Topic	Therapistmanagement of the medical-technical department
Co-operation partners	

Assignment of tasks	Realization of a web application and an iOS- and Android app to manage the medical-technical department of the state hospital Wiener Neustadt.
---------------------	--

Realization	A web application including a server with a database connection was implemented. As well as an iOS and Android app based on the Ionic Framework. The web application is used to manage the entire system, while the apps serve the administration of the individual therapist.
-------------	--

Results	Functional web application Relational and consistent database Functional iOS and Android app Documentation and user manual
---------	---

Kurzfassung

Die manuelle Verwaltung einer Abteilung mit ca. 60 Mitarbeitern ist ein sehr aufwendiges Unterfangen. Um der medizinisch-technischen Abteilung des Landesklinikums Wiener Neustadt diese Arbeit wesentlich zu vereinfachen, wurde die medtec-Webanwendung ins Leben gerufen.

Mit der Entwicklung einer Webanwendung zusammen mit Apps für iOS und Android wurde eine möglichst intuitive und leicht verfügbare Plattform zur Unterstützung der Abteilung geschaffen.

Die Schwerpunkte der Webanwendung sind unter anderem die Verwaltung der angestellten Therapeuten hinsichtlich Diensterteilung und Aufgabenzuordnung. Diese Aufgabenzuordnung ist aufgrund der Größe der Abteilung und der Vielfältigkeit des Therapieangebotes sehr zeitintensiv. Die komplexen Zusammenhänge zwischen den persönlichen Vorlieben der Therapeuten, den Parametern der einzelnen Therapien (Zeitpunkt der Therapie, Dauer, Ort, ...) und den vorgegebenen Verfügbarkeiten der Therapeuten erschweren die manuelle Planung erheblich. Die Webanwendung generiert in kürzester Zeit Dienstpläne automatisch unter Berücksichtigung all dieser Parameter und lässt zusätzlich auch noch Freiraum für manuelle Änderungen und Dienstzuteilungen.

Die Webanwendung bietet unterschiedliche Optionen für die Team-Therapeuten und die Therapeutin in der Leitungsfunktion. Der Leitung wird die Verwaltungsebene freigeschaltet, welche die Verwaltung aller Therapeuten, Arbeitszeiten, Gruppen, Blockaden und Abwesenheiten ermöglicht. Außerdem können über diese Oberfläche der Dienstplan und die dazugehörigen Raumpläne automatisch generiert werden, welche die aktuellen Abwesenheiten und Blockaden berücksichtigen. Die erstellten Pläne können als PDF exportiert und ausgedruckt werden. Die Team-Therapeuten sehen ihren momentanen Dienstplan, sowie ihre aktuellen Arbeitszeiten, eingetragenen Urlaube und Verpflichtungen über die Webanwendung. Weiters können sie diese Informationen über die App abrufen.

Ein weiteres sehr unterstützendes und effizienzsteigerndes Feature ist das Vertretungsmanagement, welches für alle Therapeuten verfügbar ist. Dieses Tool ermöglicht den Therapeuten das Abmelden für den aktuellen Tag durch Betätigen eines einzigen Buttons. Automatisch werden die definierten Vertretungen über die Abwesenheit per App-Nachricht informiert und die Aufgaben des abwesenden Mitarbeiters umverteilt.

Die medtec-Webanwendung und ihre Apps ersetzen unzählige aufwendige Telefongespräche innerhalb des Teams, die bisher notwendig waren, um einen reibungslosen Ablauf in der Abteilung zu gewährleisten.

Abstract

Managing a complete department by hand can be an extremely time-consuming task. The web application medtec reduces this necessary workload of the medical-technical department of the state hospital Wiener Neustadt.

A web application, iOS and Android app were developed to create an intuitive and easily accessible platform for the department.

This application offers different options for the leading and the common therapist. The management has access to the administration layer, allowing them to manage all therapists, their worktimes, groups, and absences.

Furthermore, this layer offers the functionality of generating a service plan, including the related room timetables. These generated plans take all current absences and blockades into account and can be exported to PDF or printed out.

Common therapists can check on their momentary service plan, worktimes, registered absences and obligations from the web application or App.

Another feature is the replacement management, which is available for all therapists. With this system therapists can sign out, should they drop out for this day, and all needed deputies are automatically informed.

Kapitel 1

Design und Usability (LG)

1.1 Motivation

In den folgenden Abschnitten wird auf verschiedene designtechnische Aspekte, die während der Entwicklung der Therapeutenverwaltung medtec beachtet wurden, eingegangen. Das umfasst unter anderem Festlegungen bezüglich Farbgestaltung, Logodesign oder Typografie. Desweiteren wird geklärt, wie sich Design und Usability unterscheiden und wie diese Bereiche speziell in Software-Projekten helfen können, um die Mensch-Computer-Interaktion zu verbessern.

1.2 Logo

Ein Logo ist die visuelle Darstellung die z.B: eine Marke, ein Produkt, ein Unternehmen oder eine Person repräsentiert und einen wesentlichen Teil des Erscheinungsbilds ausmacht. Es gilt als Identifikationssymbol und ist in seiner Gestaltung, welche auf verschiedene Arten erfolgen kann, abhängig von Sinn und Einsatzzweck.

Ein gutes Logo sollte mindestens folgenden vier Grundsätzen ¹ folgen, um erfolgreich zu sein:

1. **Verständlichkeit**
Eine verständliche Kommunikation der Kernbotschaft.
2. **Reproduzierbarkeit**
Gewährleistung der Vervielfältigung auf verschiedenen Formaten und Größen, wie zB.: App-Icon, Stempelaufdruck und Plakat.
3. **Einprägsamkeit**
Einprägende Wirkung, um langfristig im Gedächtnis zu bleiben. Wesentliche Aussage soll auf den Punkt gebracht werden.
4. **Unverwechselbarkeit**
Einmaligkeit in der Gestaltung. Es dürfen keine offensichtliche Verbindungen zu anderen Logos bestehen.

Logos kann man in drei verschiedene Kategorien unterteilen: Wortmarken, Bildmarken und Wort-Bildmarken. In den nachfolgenden Abschnitten wird kurz darauf eingegangen.

¹Vgl. <https://www.seo-analyse.com/seo-lexikon/1/logo/>, besucht am 03.04.2017

1.2.1 Bildmarken

Eine Bildmarke ist grundsätzlich eine Abbildung, die eindeutig genug ist, um ein Unternehmen oder ein Produkt zu identifizieren. In der Regel enthält es keinen Text und besteht aus markanten Formen oder Elementen, die gut einpräglich sind.

Bei der Gestaltung einer möglichen Bildmarke für die Therapeutenverwaltung wäre die Kombination eines Personensymbols mit Datenblatt passend, die Therapeuten mit gewissen Aufgaben symbolisiert. In Abbildung 1.1 sieht man das Ergebnis dieser Darstellung.

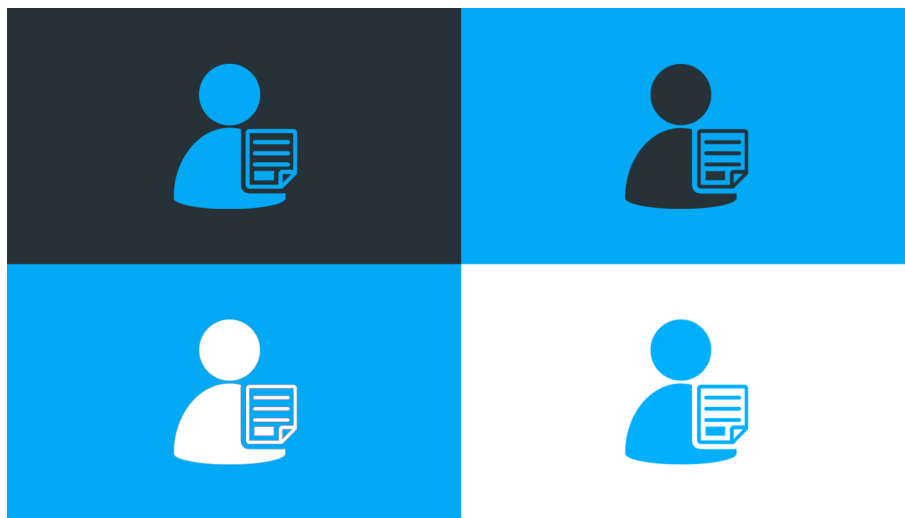


Abbildung 1.1: medtec-Logo als Bildmarke

1.2.2 Wortmarken

Eine Wortmarke ist ein aus Schriftzeichen bestehender Text, der so gestaltet ist, dass ein Unternehmen oder Produkt eindeutig damit in Verbindung gebracht werden kann. Hierfür können alle Groß- und Kleinbuchstaben, Ziffern und Sonderzeichen verwendet werden. Im Gegensatz zu der Bildmarke kann es hierbei oft an Alleinstellungsmerkmalen fehlen.

Bei der Wortmarke für das medtec-Logo wurde eine serifenlose Schrift in den Schriftsätzen *bold* und *regular* gewählt. In Abbildung 1.2 sieht man die visuelle Darstellung vom diesem Logo.



Abbildung 1.2: medtec-Logo als Wortmarke

1.2.3 Wort-Bildmarken

Eine Wort-Bildmarke ist die dauerhafte Kombination aus textlichen und grafischen Elementen² zur eindeutigen Identifizierung eines Unternehmens oder eines Produkts. Dabei werden die wesentlichen Gesichtspunkte einer Wort- und Bildmarke miteinander verbunden. Dadurch sollen die Vorteile der anderen Varianten kombiniert werden.

Das Logo von medtec als Wort-Bildmarke ist in Abbildung 1.3 zu sehen. Da es in unserem Fall quadratisch ist, eignet es sich besonders gut für die Darstellung als App-Icon.

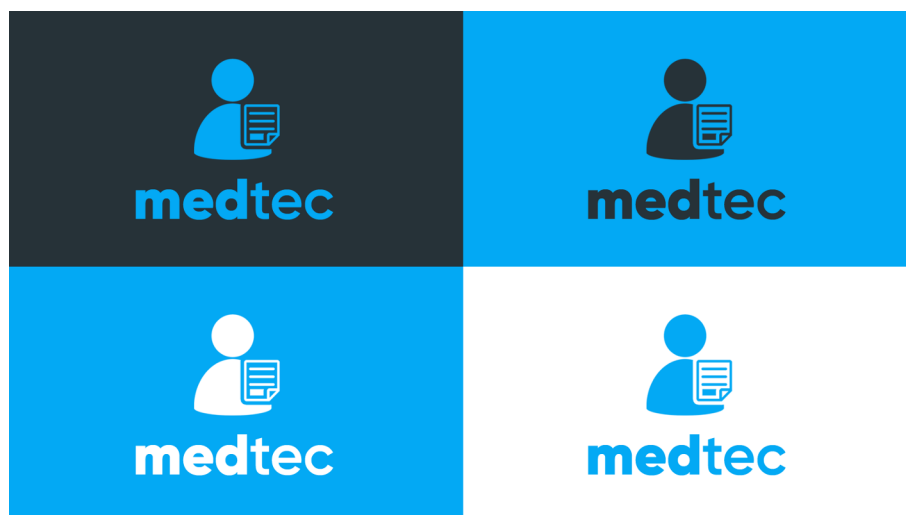


Abbildung 1.3: medtec-Logo als Wort-Bildmarke

²Vgl. <https://de.wikipedia.org/wiki/Wort-Bild-Marke>, besucht am 03.04.2017

1.2.4 Fazit

Wichtig zu erwähnen ist, dass im menschlichen Gehirn, Formen oder Logos nicht isoliert eingeordnet, sondern oft mit anderen Dingen, wie zum Beispiel Melodien, Emotionen, Gerüchen, Namen oder persönlichen Erfahrungen, verknüpft werden, so dass sich bei einer ständigen visuellen Wiedergabe diese Bilder verfestigen³. Die unterschiedlichen Gestaltungskategorien haben verschiedenste Vor- und Nachteile.

Für die Gestaltung von medtec war es wichtig, die Kernbotschaft klar und deutlich zu übermitteln. Der Name „medtec“ steht ganz einfach für med - Medizin und tec - technisch. Die Therapeutenverwaltung der medizinisch technischen Abteilung, ist weder ein Produkt, welches verkauft wird und sich am Markt von anderen abheben muss, noch bei einem Wettbewerb bezüglich Logodesign groß aufspielen muss. Es sollte per Definition des Auftraggebers dezent und sauber gehalten sein, sowie sich von negativen Gefühlen wie Aggressivität stark abgrenzen, weswegen auf rote Farbtöne⁴ im Logo und der Oberfläche verzichtet wurde. Auf die Farbgestaltung in der Gestaltung wird zu einem späteren Zeitpunkt nochmal ausführlich eingegangen. Die Entscheidung über die Art des Logos fiel auf eine horizontal-ausgerichtete *Schriftmarke*, weil sie in der Navigationsleiste für Apps im Browser, oder am Handy, wie in Abbildung 1.4 zu sehen, optisch am besten passt.

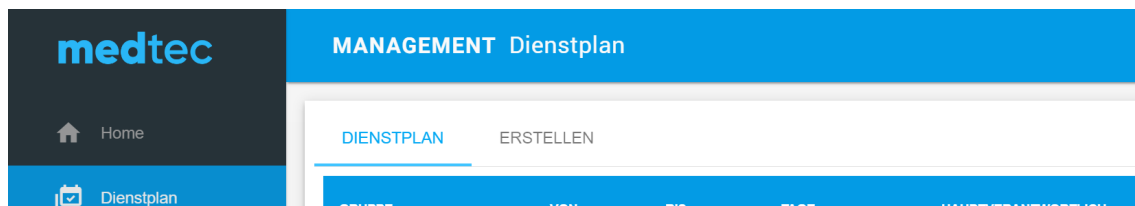


Abbildung 1.4: Schriftmarke als Logo für Anwendung

1.3 Usability

Geht es um die Software-Entwicklung wird oft das Modewort „Usability“ erwähnt. Was Usability bedeutet und welche Ziele damit verfolgt werden, wird im folgenden Abschnitt näher erläutert.

1.3.1 Definition und Abgrenzung

Auf Deutsch übersetzt wird schnell klar, was mit Usability gemeint ist, es verbindet die Werte der Worte „Gebrauchstauglichkeit“, „einfache Bedienung“ und „Benutzerfreundlichkeit“. Usability selbst ist noch nicht wertend, sondern sollte als messbarer Wert, wie etwa „Qualität“ oder „Wetter“ angesehen werden. Jeder Tag hat ein Wetter, der eine ein gutes, der andere ein schlechtes. Jedes Produkt hat eine Qualität, das eine Produkt eine gute, das andere eine schlechte⁵.

³Vgl. Mattschek, *Marken, Logos und Symbole - Einfluss und Bedeutung*.

⁴Vgl. <https://www.lichtkreis.at/wissenswelten/welt-der-farben/die-farbe-rot/>, besucht am 03.04.2017

⁵Vgl. Florin, *User - Interface - Design*, S. 16-17.

Eine gute Usability stellt den Benutzer in das Zentrum und kann folgende positive Auswirkungen haben⁶:

- **Weniger Schulungsaufwand**
Aufwand zum Erlernen der Software sinkt. Damit verbunden sinken auch die Kosten für eventuelle Einschulungen für Nutzer in ein Programm.
- **Bessere Performance**
Die Benutzer können schneller mit dem Programm arbeiten und erreichen dadurch mehr in kürzerer Zeit.
- **Höhere Nutzerzufriedenheit**
Anwender hat weniger Frust mit der Software und dadurch steigt auch die Empfehlungswahrscheinlichkeit. Die Risiken der falschen Benutzung und deren Konsequenzen werden minimiert.

Oft werden die Begriffe Design und Usability gleichbedeutend verwendet. Während sich Design mit den sichtbaren Aspekten und dem konkreten Aussehen beschäftigt, geht es bei der Usability eher um die Bedienbarkeit. Ein optisch ansprechendes Design ist quasi der sichtbarste Aspekt der Usability und kann die verschiedenen Bedürfnisse des Benutzers zufriedenstellen.

Im nächsten Kapitel wird anhand des Beispiels von Icons erklärt, wie gute Usability erreicht werden kann.

1.3.2 Icons

Ein Bild sagt ja bekanntlich mehr als tausend Worte. Icons, Piktogramme, aber auch Emojis gewinnen im User Interface Design immer mehr an Bedeutung. Sie prägen die moderne Informationsdarstellung und sind aus heutigen Websites und Applikationen kaum mehr wegzudenken, weswegen sie sorgfältig und vor allem gezielt eingesetzt werden sollten.

Ein gutes Bildzeichen gibt Orientierung und muss deshalb vor allem eines sein: **eindeutig**. Gute Icons helfen, Informationen oder Funktionsweisen schnell zu erfassen und wiederzuerkennen. Sie sparen Platz, sehen gut aus und funktionieren sprachübergreifend intuitiv. Doch leider können solche bildlichen Darstellungen häufig missverstanden werden, weil sie kontextabhängig und mehrdeutig interpretierbar sind.

Die selben Icons haben in verschiedenen Umgebungen oft andere Bedeutungen. Eine Lupe kann für „Suchen“ oder „Vergrößern“ stehen. Oft gebrauchte Icons erschließen sich erst im Zusammenspiel mit Text oder einem anderen Element. Wenn die Lupe an ein Eingabefeld gebunden ist, bedeutet sie „Suchen“, auf einer Werkzeugleiste hingegen „Vergrößern“. Kurze Textlabels wie beispielsweise ein Tooltip können den Sinn und Zweck unterstützen. Tooltips sind kleine Textmeldungen die zusätzliche Informationen bereitstellen und erst beim Mausover sichtbar sind.

⁶Vgl. Florin, *User - Interface - Design*, S. 3.

Bereits geläufige und bekannte Symbole sind sehr viel effektiver in ihrer Funktionsweise als außergewöhnliche und künstlerische Zeichen⁷. Letztere sind womöglich zwar schön anzusehen, sorgen aber beim Benutzer in der Regel für mehr Verwirrung als Klarheit. Das Ziel eines UI-Designers sollte es hier also sein, zwischen den verschiedenen Möglichkeiten mit Bedacht zu unterscheiden, um für den Anwendungszweck die möglichst passende Lösung zu finden.



Abbildung 1.5: Tooltip bei Button

Bei der grafischen Oberfläche von medtec wurden diverse Icons zur Erhöhung der Usability verwendet. Beispielsweise erkennt man in Abbildung 1.5 einen Plusbutton, welcher auch missverstanden werden kann. Möglicherweise stellt sich der Benutzer hier die Frage, was hinzugefügt wird. Durch den Tooltip wird hier signalisiert, dass es um das Hinzufügen von Therapeuten geht. Bei Menüelementen, mit denen der Benutzer interagiert, ist es gebräuchlich, dass das Symbol des Cursors beim Darüberfahren vom Mauszeiger zum Symbol der Hand wird⁸.

⁷Vgl. Florin, *User - Interface - Design*, S. 243.

⁸Vgl. Ippen, *Web Fatale: Wie Du Webseiten und Web-Apps gestaltest, denen niemand widerstehen kann: Usability, User Experience und Interaktion*, S. 49-50.

Kapitel 2

Frontend (LG)

2.1 Motivation

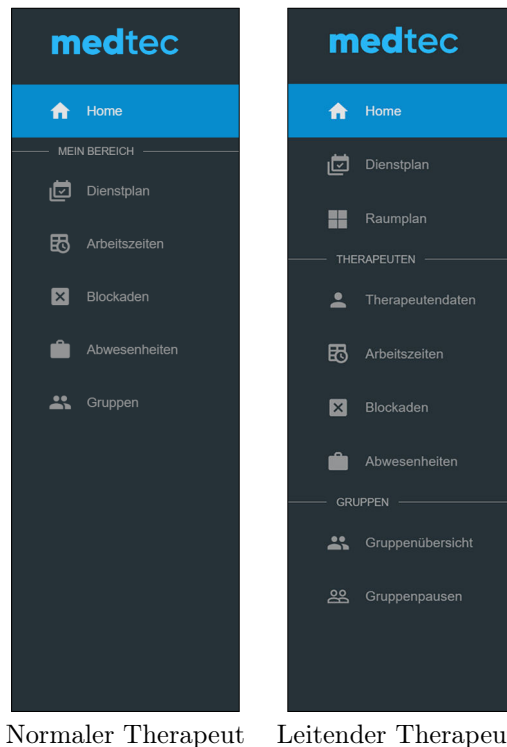
Dieses Kapitel befasst sich mit allen technischen Aspekten, die primär das Frontend der Webapplikation **medtec** betreffen. Das Web und die damit verbundenen Ansprüche des Benutzers an die Anwendungen entwickeln sich ständig weiter. Es wird geklärt, wie gute Webanwendungen aufgebaut sind und welche Methoden und Praktiken für Entwicklung von medtec verwendet wurden. Weiters werden die Kernfunktionen und wesentlichen Aufgaben der Therapeutenverwaltung vorgestellt.

2.2 Funktionen der Webapp

Derzeit erfolgt die Zuteilung der Therapeuten zu den einzelnen Gruppen der medizintechnischen Abteilung im LK Wiener Neustadt per Hand. Dieser Vorgang gestaltet sich bei etwa 60 Therapeuten sehr zeitaufwändig und kann mit entsprechender Software unterstützt bzw. digitalisiert werden. Im folgenden Abschnitt wird auf die allgemeinen Aufgaben und Kernfunktionen der Webanwendung eingegangen.

2.2.1 Unterteilung der Rollen

Die Unterteilung der Nutzer für die Webanwendung fällt prinzipiell in zwei Kategorien: *Leitende Therapeuten* und *Normale Therapeuten*. Während ein *Normaler Therapeut*, wie in der nächsten Abbildung ersichtlich, nur Zugriff auf seine Daten, wie etwa Arbeitszeiten hat, bietet die Therapeutenverwaltung den *Leitenden Therapeuten* wesentlich mehr Möglichkeiten. Dieser kann nicht nur in alle im System gespeicherten Daten einsehen, sondern diese auch verändern oder löschen. Da das Hauptaugenmerk während der Entwicklung auf die Perspektive des *Leitenden Therapeuten* gelegt wurde, werden die nachfolgenden Schritte aus dessen Sicht erklärt.



Normaler Therapeut Leitender Therapeut

Abbildung 2.1: Vergleich von Navigationsleisten der verschiedenen Rollen

2.2.2 Dienstplan

Die digitale Darstellung des Dienstplans und dessen automatische Generierung war für den Auftraggeber einer der primären Einsatzzwecke der Therapeutenverwaltung medtec. Im Nachfolgenden wird kurz auf die Anzeige der Daten und auf die automatische Erstellung eines neuen Dienstplans aus Sicht eines leitenden Therapeuten eingegangen.

Dienstplan-Anzeige

In der Anzeige-Funktion wird übersichtlich der Dienstplan, der aktuellen Kalenderwoche mit allen relevanten Informationen dargestellt. Er klärt, welche Gruppen an welchen Wochentagen, zu welchen Tageszeiten stattfinden, und führt an, wie in Abbildung 2.2 erkennbar, welche Therapeuten für die jeweiligen Gruppen eingeteilt sind. Pro Gruppe gibt es immer einen Hauptverantwortlichen und zwei Stellvertreter, sollte ersterer verhindert sein. Zusätzlich dazu werden noch mögliche Abwesenheiten der einzelnen Therapeuten in der Woche markiert. Das macht sich durch die Rotfärbung des Wochentages ersichtlich.

Um den Dienstplan der aktuellen Woche auch physisch zur Verfügung zu stellen, wurde zusätzlich eine Funktion zum Ausdrucken implementiert. Das Exportieren in ein druckbares PDF-Format wird mit der frei verfügbaren JavaScript-Library PDF-Make¹ durchgeführt. Durch die Verlagerung dieser Aufgabe auf die Client-Seite wird beim Server an kostbaren Ressourcen gespart.

¹Vgl. <https://github.com/bpampuch/pdfmake>, besucht am 03.04.2017

GRUPPE	VON	BIS	TAGE	HAUPTVERANTWORTLICH	ANWESEND	VERTRETUNG 1	ANWESEND	VERTRETUNG 2	ANWESEND
TT UE	07:00 Uhr	08:00 Uhr	Mo Mi Fr	Ofenböck Cornelia	Mo Mi Fr	Kostak Eva	Mo Mi Fr	Ernst Marie-Theres	Mo Mi Fr
TT WS 2	07:00 Uhr	08:00 Uhr	Di Do	Binder Petra	Di Do	Ernst Marie-Theres	Di Do	Trofer Julia	Di Do
Knie 1 belastet	07:00 Uhr	07:30 Uhr	Mo Mi Fr	Ernst Marie-Theres	Mo Mi Fr	Zachar Remy	Mo Mi Fr	Stockreiter Kerstin	Mo Mi Fr
WS-Analyse	07:00 Uhr	07:30 Uhr	Do	Müller Christine	Do	Kornfeld Johanna	Do	Tache Claudia	Do
WS-Analyse	07:00 Uhr	08:00 Uhr	Di	Müller Christine	Di	Kornfeld Johanna	Di	Tache Claudia	Di
Unterwasser	07:00 Uhr	09:00 Uhr	Di Do	Haumer Sabine	Di Do	Pohl Angelika	Di Do	Kostak Eva	Di Do
Wochenbetgymnastik	07:00 Uhr	08:00 Uhr	Mo Di Mi Do Fr	Schuller Elke	Mo Di Mi Do Fr	Kornhäusl Alexandra	Mo Di Mi Do Fr	Knotzer Katharina	Mo Di Mi Do Fr
Knie 1 entlastet	07:30 Uhr	08:00 Uhr	Mo Mi Fr	Knotzer Katharina	Mo Mi Fr	Kornfeld Johanna	Mo Mi Fr	Tarnok Klaudia	Mo Mi Fr
Unterwasser	07:30 Uhr	09:00 Uhr	Mo Mi Fr	Haumer Sabine	Mo Mi Fr	Pohl Angelika	Mo Mi Fr	Kostak Eva	Mo Mi Fr
Sprunggelenk belastet	08:00 Uhr	08:30 Uhr	Mo Mi Fr	Kalwitz Bettina	Mo Mi Fr	Bauer Manuela	Mo Mi Fr	Haumer Sabine	Mo Mi Fr
Sprunggelenk entlastet	08:00 Uhr	08:30 Uhr	Di Do	Ganster Lukas	Di Do	Knotzer Katharina	Di Do	Gmoser Lukas	Di Do
Knie 2	08:00 Uhr	09:00 Uhr	Mo Mi Fr	Tache Claudia	Mo Mi Fr	Wieder Christoph	Mo Mi Fr	Kornhäusl Alexandra	Mo Mi Fr
Hüfte 2	08:30 Uhr	09:00 Uhr	Mo Mi Fr	Reinisch Rafael	Mo Mi Fr	Knotzer Katharina	Mo Mi Fr	Gmoser Lukas	Mo Mi Fr

Abbildung 2.2: Dienstplan-Anzeige

Erstellung des Dienstplans

Die Planung des Dienstplans gestaltet sich nicht so trivial, wie sie zunächst scheint. Normale Therapeuten können bestimmte Gruppen bevorzugen, die sie besonders gerne leiten wollen, indem sie diese dem Leitenden Therapeuten schriftlich kommunizieren. Dieser kann dann, wie in Abbildung 2.3 erkennbar, die Auswahl der bevorzugten Therapeuten zu den jeweiligen Gruppen bei der Erstellung einplanen. Bei den *Verfügbaren Therapeuten* in der Abbildung wird nicht zwischen *Leitenden Therapeuten* oder *Normalen Therapeuten* unterschieden. Anhand dieser Informationen kann dann in weiterer Folge, mit einem einfachen Klick auf den zuständigen Button, ein Dienstplan generiert werden. Damit erfolgt die Generierung also vollkommen autonom, man kann jedoch die Zuteilung geringfügig mit verschiedenen Präferenzen beeinflussen. Durch die automatische Erstellung des Dienstplans sollen der zuständigen Person im LK Wiener Neustadt einige Tage an Planung in nur wenigen Sekunden abgenommen werden.

Gruppen

Suche Gruppe:

- Discus ☐ Turnsaal Tür 5
- Ellbogen ☐ Turnsaal Tür 5
- Entspannung ☒ Med. Trainingst
- Gilchlist - HG ☐ Turnsaal Tür 5
- Hand/Finger ☐ Turnsaal Tür 5
- HWS ☐ Turnsaal Tür 5
- Hüfte 1 ☐ Med. Trainingst
- Hüfte 2 ☐ Turnsaal Tür 5
- Klangreisen ☐ Med. Trainingst

Verfügbare Therapeuten

Suche Therapeuten:

- Jung Alexander
- Kalwitz Bettina
- Knotzer Katharina
- Kornfeld Johanna
- Kornhäusl Alexandra
- Kostak Eva
- Lechner Martina
- Müller Christine
- Ofenböck Cornelia
- Pohl Angelika

Bevorzugte Therapeuten

aktuelle Gruppe: Entspannung Anzahl: 5

- Bauer Manuela
- Ernst Marie-Theres
- Haidvogel Manuela
- Kornfeld Johanna
- Müller Christine

Abbildung 2.3: Erstellung des Dienstplans im Frontend

2.2.3 Raumplan

Zusätzlich zum Dienstplan war es dem Auftraggeber noch wichtig, über eine einfache Darstellung der Räume und den damit verbundenen Informationen zu verfügen. Der Raumplan gilt immer für die aktuelle Kalenderwoche und gibt pro Raum an, an welchen Wochentagen, welche Gruppen stattfinden. Zusätzlich zu den Gruppen werden noch die Beginn- bzw. Endzeit und der zuständige Therapeut angezeigt. Der Name des zuständigen Therapeuten wird dann abhängig von der Anwesenheit in grün(anwesend) oder rot(abwesend) gehalten.

Ähnlich wie beim Dienstplan kann auch der Raumplan als druckbare PDF-Datei, mit einem einfachen Klick auf den zuständigen Button exportiert werden.

MANAGEMENT Raumplan

RAUMPLAN RÄUME

TURNUSAL TUR 5

MONTAG	DIENTAG	MITTWOCH	DONNERSTAG	FREITAG
07:00 Uhr 07:30 Uhr Ernst Marie-Theres Knie 1 belastet	07:00 Uhr 08:00 Uhr Müller Christine WS-Analyse	07:00 Uhr 07:30 Uhr Ernst Marie-Theres Knie 1 belastet	07:00 Uhr 07:30 Uhr Müller Christine WS-Analyse	07:00 Uhr 07:30 Uhr Ernst Marie-Theres Knie 1 belastet
07:30 Uhr 08:00 Uhr Knotzer Katharina Knie 1 entlastet	08:00 Uhr 08:30 Uhr Ganster Lukas Sprunggelenk entlastet	07:30 Uhr 08:00 Uhr Knotzer Katharina Knie 1 entlastet	08:00 Uhr 08:30 Uhr Ganster Lukas Sprunggelenk entlastet	07:30 Uhr 08:00 Uhr Knotzer Katharina Knie 1 entlastet
08:00 Uhr 08:30 Uhr Kalwitz Bettina Sprunggelenk belastet	09:00 Uhr 10:00 Uhr Pohl Angelika Schulter 2	08:00 Uhr 08:30 Uhr Kalwitz Bettina Sprunggelenk belastet	09:00 Uhr 10:00 Uhr Pohl Angelika Schulter 2	08:00 Uhr 08:30 Uhr Kalwitz Bettina Sprunggelenk belastet
08:30 Uhr 09:00 Uhr Reinisch Raffael Hüfte 2	10:00 Uhr 10:30 Uhr Tache Claudia LWS	08:30 Uhr 09:00 Uhr Reinisch Raffael Hüfte 2	10:00 Uhr 10:30 Uhr Tache Claudia LWS	08:30 Uhr 09:00 Uhr Reinisch Raffael Hüfte 2
09:00 Uhr 10:00 Uhr Gmoser Lukas Schulter 1	10:30 Uhr 11:00 Uhr Kalwitz Bettina Gleichricht - HG	09:00 Uhr 10:00 Uhr Gmoser Lukas Schulter 1	10:30 Uhr 11:30 Uhr Trofer Julia Rückenschule Arzt	09:00 Uhr 10:00 Uhr Gmoser Lukas Schulter 1
11:00 Uhr 11:30 Uhr Müller Christine Hand/Finger	12:30 Uhr 13:00 Uhr Kornhäusl Alexan... Discus	10:00 Uhr 11:00 Uhr Suhaj Zsuzsanna Schwangerenturnen	10:30 Uhr 11:30 Uhr Aigner Gerl Rückenschule PT	11:00 Uhr 11:30 Uhr Müller Christine Hand/Finger
11:30 Uhr 12:00 Uhr Zachar Remy Ellbogen	13:00 Uhr 13:30 Uhr Kostak Eva HWS	11:00 Uhr 11:30 Uhr Müller Christine Hand/Finger	10:30 Uhr 11:30 Uhr Knotzer Katharina Rückenschule ET	11:30 Uhr 12:00 Uhr Zachar Remy Ellbogen
13:00 Uhr 13:30 Uhr Kornhäusl Alexan... TEP UE		11:30 Uhr 12:00 Uhr Zachar Remy Ellbogen	12:30 Uhr 13:00 Uhr Kornhäusl Alexan... Discus	12:30 Uhr 13:00 Uhr Kornhäusl Alexan... Discus
		13:00 Uhr 13:30 Uhr Kornhäusl Alexan... TEP UE	13:00 Uhr 13:30 Uhr Kostak Eva HWS	13:00 Uhr 13:30 Uhr Kornhäusl Alexan... TEP UE

Abbildung 2.4: Raumplan-Anzeige

2.3 Evaluierung von AngularJS

In diesem Abschnitt geht es um die technischen Grundsätze guter Entwicklung, welche Möglichkeiten sich durch die Verwendung von Entwurfsmustern bei der Programmierung bieten, und warum AngularJS für die Entwicklung unserer Anwendung eingesetzt wurde.

Während vor vielen Jahren noch galt, dass Webanwendungen ihre Logik, soweit es geht, auf dem Server lagern, hat sich das Blatt in den letzten Jahren gewendet. Der Trend geht von statischen Webauftritten mehr in Richtung der dynamischen Webanwendungen. Immer mehr an Bedeutung gewinnen clientseitige Techniken, wie etwa **JavaScript**².

Laut einer Statistik von *W3Techs*³, wie in Abbildung 2.5 erkennbar, verwenden knapp 95 Prozent aller Websites diese Programmiersprache.

²Vgl. Steyer, *Angular JS - Moderne Webanwendungen und SPAs mit JavaScript*, S. 7.

³Vgl. https://w3techs.com/technologies/overview/client_side_language/all, besucht am 03.04.2017

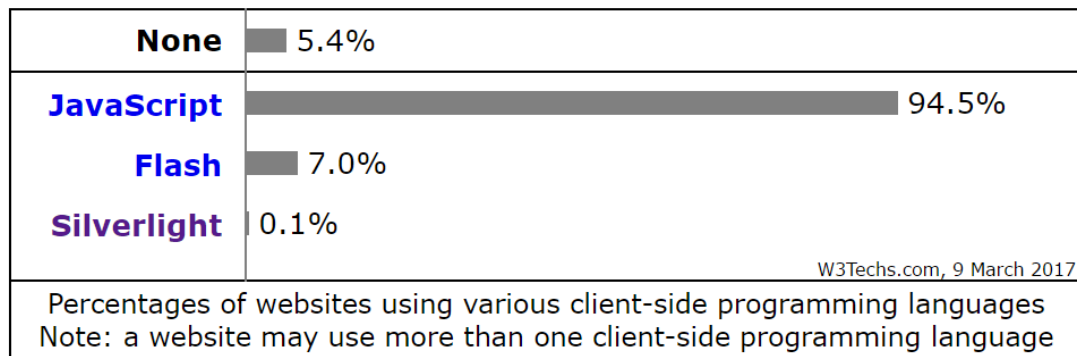


Abbildung 2.5: Statistik zur Verwendung von Programmiersprachen auf Client-Seite

2.3.1 Chancen und Möglichkeiten

Die Programmierung einer modernen Webanwendung auf Basis von JavaScript gestaltet sich sehr anspruchsvoll. Der Verantwortliche muss sich um das Verwenden von diversen Services, um das Binden von verschiedenen Daten sowie das Kontrollieren von Eingaben kümmern. Der Programmcode, der dabei entsteht, soll darüber hinaus übersichtlich, klar-strukturiert und wartbar sein. Außerdem sollten zukünftige Änderungen bzw. Erweiterungen ohne große Umstände möglich sein. All das ist zwar mit JavaScript zu bewältigen, erfordert aber, seitens des Entwicklers, eine Menge Disziplin und geht mit der Implementierung analoger Codeabschnitte einher⁴.

Genau für diesen Zweck gibt es eine Fülle an Frontend-Frameworks, die das Entwickeln stark vereinfachen. Das populäre Open-Source-Framework **AngularJS**, welches von Google im Jahr 2009⁵ veröffentlicht wurde, ermöglicht die Trennung der einzelnen Komponenten und vereinfacht den Ablauf der Softwareentwicklung.

2.3.2 Vergleichbare Technologien

Um die verschiedenen Besonderheiten von AngularJS hervorzuheben, werden vergleichbare Frameworks und Bibliotheken, die ähnliche Funktionalität bieten, wie etwa JQuery, Ember.js oder Backbone.js im Nachfolgenden kurz vorgestellt und verglichen.

JQuery

JQuery ist eine sehr weit verbreitete und frei zur Verfügung stehende JavaScript-Bibliothek, die das Selektieren, Navigieren oder Manipulieren im **Document Object Model**(DOM) sowie das Verwenden von Methoden für AJAX-Requests ermöglicht⁶.

⁴Vgl. Steyer, *Angular JS - Moderne Webanwendungen und SPAs mit JavaScript*, S. 7.

⁵Vgl. <https://de.wikipedia.org/wiki/AngularJS>, besucht am 02.04.2017

⁶Vgl. <https://jquery.com/>, besucht am 02.04.2017

Folgende Möglichkeiten bieten JQuery und AngularJS im direkten Vergleich⁷:

Tabelle 2.1: Gegenüberstellung der Möglichkeiten von jQuery mit AngularJS

	<i>jQuery</i>	<i>AngularJS</i>
Abstraktion des DOM	ja	ja
Asynchrone Methodenaufrufe	ja	ja
Unterstützung von Animationen	ja	ja
AJAX bzw. JSONP	ja	ja
Unterstützung des MVC-Pattern	nein	ja
Templating	nein	ja
Datenbindung (Two-way)	nein	ja
Form-Validierung	nein	ja
Lokalisierung	nein	ja

Werden die beiden JavaScript-Erweiterungen in der minimierten Version hinsichtlich ihrer Dateigröße verglichen, so wären es bei jQuery 3.1.1 etwa **85kb** und bei AngularJS 1.6 **164kb**, was der umfangreicheren Funktionalität geschuldet ist. Wie in der Vergleichstabelle 2.1 zu sehen, bietet AngularJS alle Möglichkeiten, die jQuery auch hat - und noch mehr. Wer also moderne Webanwendungen entwickeln will, sollte in diesem Fall eher zu Googles Framework als zu jQuery tendieren.

Ember.js

„A framework for creating ambitious web applications“⁸

Ember.js ist ein Client-Framework für JavaScript, welches speziell die Erstellung von komplexen Single Page Applications (darauf wird in einem späteren Abschnitt ausführlich eingegangen) vereinfacht. Es ist in der minimierten Version mit **199kb** wesentlich größer als AngularJS und bietet gleichwertige Möglichkeiten. Eine kleinere Dateigröße ermöglicht logischerweise auch schnellere Ladezeiten.

Der direkte Vergleich von Ember.js mit AngularJS fällt wie folgt aus⁹:

Tabelle 2.2: Gegenüberstellung der Möglichkeiten von Ember.js mit AngularJS

	<i>Ember.js</i>	<i>AngularJS</i>
Modelle sind einfache JS-Objekte	nein	ja
Asynchrone Methodenaufrufe	ja	ja
Reicht über HTML hinaus	nein	ja
Dependency Injection	ja	ja
Unterstützung des MVC-Pattern	ja	ja
Beinhaltet jQuery Lite	nein	ja
Datenbindung (Two-way)	ja	ja
Form-Validierung	nein	ja
Serverseitiges vorrendern von Views	ja	nein
Unterstützung von Routing/Deep-Linking	ja	ja

⁷Vgl. Lamb, *jQuery vs. AngularJS: A Comparison and Migration Walkthrough*, übersetzt vom Autor.

⁸Vgl. <http://emberjs.com/>, besucht am 03.04.2017

⁹Vgl. <https://versus.com/de/angularjs-vs-ember-js>, besucht am 03.04.2017, übersetzt vom Autor

Man sieht, dass beide Frameworks ihre Vor- und Nachteile haben. Wird die Entwicklung einer benutzgetriebenen Anwendung fokussiert, bei der Benutzereingabe überprüft werden muss, kann hier AngularJS mit der eingebauten Eingabe-Validierung den Vergleich gewinnen. Wenn aber serverseitiges Vorrendern von Templates gewünscht ist, was die Performance vor allem bei schwachen Rechnern erhöht, gewinnt Ember.js den Vergleich.

Backbone.js

Backbone.js ist ein leichtgewichtiges clientseitiges-JavaScript Framework, welches die Programmierung von Single Page Applications unterstützt¹⁰. Es hat im Vergleich zu AngularJS, in der minimierten Variante, eine wesentlich kleinere Dateigröße von nur **19kb**.

Der direkte Vergleich von Backbone.js mit AngularJS¹¹:

Tabelle 2.3: Gegenüberstellung der Möglichkeiten von Backbone.js mit AngularJS

	<i>Backbone.js</i>	<i>AngularJS</i>
Reicht über HTML hinaus	nein	ja
Kann auf dem Server gerendert werden	ja	nein
Form-Validierung	nein	ja
Unterstützung des MVC-Pattern	ja	ja
Beinhaltet jQuery Lite	nein	ja
Unterstützung von Datenbindungen	nein	ja
Unterstützung von Routing/Deep-Linking	ja	ja

2.3.3 Fazit

Wie bereits erwähnt, kann die Dateigröße bei einem Framework oder einer Bibliothek schnell zur tragenden Rolle werden, weil sie die Ladezeit der Webanwendung beeinflusst. Ein wichtiges Kriterium bei der Entscheidung des richtigen Framework für ein Projekt sollte auch die Community dahinter sein. Eine größere Community und eine stetig wachsenden Anzahl an Entwicklern bedeutet deutlich mehr Antworten auf gestellte Fragen und mehr Inhalte auf Plattformen wie Youtube oder Ähnliches.

Aufgrund der riesigen Auswahl an Anwendungsmöglichkeiten, der großartigen Dokumentation und Google als Hersteller fiel die Wahl bei der Entwicklung der Therapeutenverwaltung medtec eindeutig auf AngularJS.

2.4 Architektur

Grundsätzlich kann man bei der Entwicklung einer Webanwendung, auf zwei verschiedene Verfahrensweisen bezüglich Web-Architektur zurückgreifen. Ganz konventionell, wie es lange Zeit gehandhabt wurde, kann man sich für eine **Multiple Page Application** (MPA) entscheiden. Die Alternative wäre, dass man sich für eine modernere Variante, wie die **Single Page Applications** (SPA), bei denen die Webanwendung im Prinzip aus einer einzigen Seite besteht, entscheidet. Nachfolgend wird auf beide Architekturen kurz eingegangen, die Funktionsweise erklärt und ein Resümee gezogen.

¹⁰Vgl. <http://backbonejs.org/>, besucht am 03.04.2017

¹¹Vgl. <https://versus.com/de/angularjs-vs-backbone-js>, besucht am 02.04.2017

2.4.1 Multiple Page Application (MPA)

Webanwendungen die nach der MPA-Architektur entwickelt werden, funktionieren ganz „traditionell“. Sie enthält mehrere statische HTML-Seiten, die untereinander mit Hyperlinks in Beziehung stehen. Bei jeder Interaktion des Benutzers mit der App, wie zum Beispiel beim Versenden von Formulardaten, wird die Webanwendung vom Server neu gerendert und anschließend zum Client transportiert.

Technisch gesehen wird also, wie in Abbildung 2.6 erkennbar¹², ein HTTP Request an den Webserver geschickt und mit einer neuen HTML-Datei geantwortet¹³. Der Client-Browser verwirft dann die alte HTML-Datei und lädt die Neue. Das führt zu einer Störung des Benutzererlebnis und kostet in weiterer Folge auch Zeit und Ressourcen.

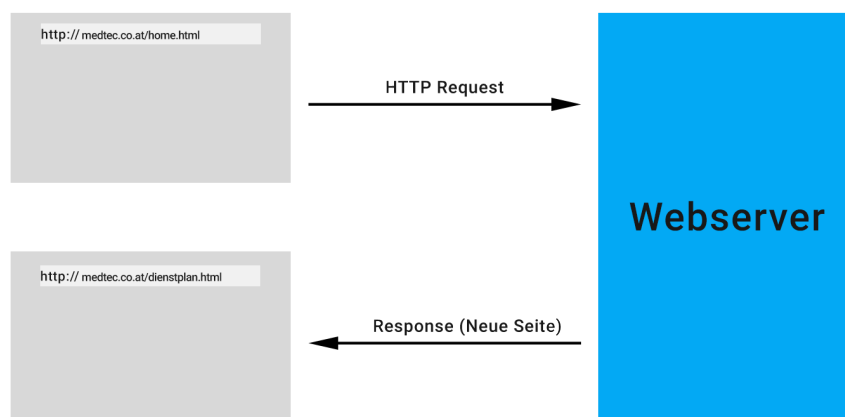


Abbildung 2.6: Architektur einer MPA

Die wesentlichen Fakten zu MPA¹⁴:

- Sehr gut für **Suchmaschinen-Optimierung** (SEO) und das damit verbundene Ranking bei Diensten wie Google oder Yahoo.
- Front- und Backend-Entwicklung liegen sehr nah zusammen.
- Die Entwicklung kann gegebenenfalls, sehr an Komplexität zunehmen, wie zum Beispiel mit diversen Frameworks für Front- und Backend, was längere Entwicklungszeiten verursacht.
- Ganze Seiten werden mittels HTTP Request geladen.

2.4.2 Single Page Application (SPA)

Die Architektur der statischen Webumgebung ist im direkten Vergleich zu den Single Page Applications anders aufgebaut. Eine SPA ist eine im Browser ausgeführte, plattformunabhängige, Web-Applikation, die während ihrer Verwendung kein einziges Mal neu geladen werden muss. Alle notwendigen Ressourcen werden einmalig zu Beginn heruntergeladen

¹²Abbildung ähnlich: <https://blog.4psa.com/an-intro-into-single-page-applications-spa/>, besucht am 01.04.2017

¹³Vgl. Pechhacker, „Single Page Webapplikationsentwicklung anhand des JavaScript Frameworks Ember.js“, S. 9.

¹⁴Vgl. Skolski, *Single Page Application vs. Multiple Page Applications*, übersetzt vom Autor.

und im Client-Browser angezeigt. Werden während der Laufzeit zusätzliche Daten benötigt, lädt man diese separat herunter, ohne die komplette Seite neu zu laden¹⁵. Erkennbar¹⁶ ist das gut in Abbildung 2.7. Diese Technik führt zu einer Erhöhung der User Experience und zur Schonung von kostbaren Ressourcen, wie Bandbreite oder Zeit. Kurz gesagt wird dadurch wesentlich weniger „Overhead“ erzeugt.

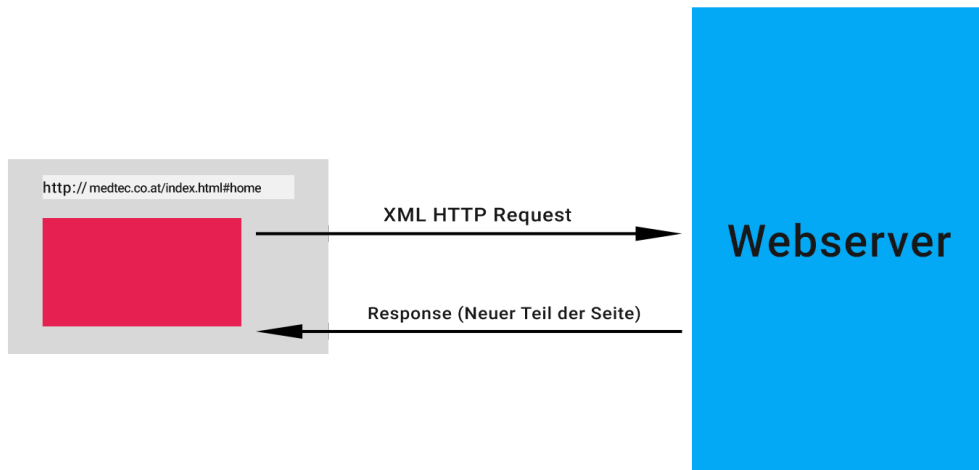


Abbildung 2.7: Architektur einer SPA

Ein gutes Beispiel der Architektur einer SPA ist die Webapp **Google Maps**¹⁷. Die Karte lässt sich einfach per Drag and Drop bewegen. Weitere Bereiche, die umliegend sind, werden bei Bedarf dynamisch nachgeladen. Durch diese Funktionalität erhält der Nutzer das Gefühl, als wäre er in einem normalen Programm. Genau das ist der große Vorteil einer Single Page Application.

Werden also bei einer speziellen Benutzerinteraktion Inhalte benötigt, können diese separat nachgeladen werden. Es wird mittels JavaScript ein Skript ausgeführt, welches den Browser dazu bringt, einen **XmlHttpRequest**, oder kurz XHR an den Server zu schicken. Der Server verarbeitet den Request und antwortet mit einem Fragment, welches die geforderten Daten, die dem Client bis zu diesem Zeitpunkt nicht zu Verfügung standen, beinhaltet¹⁸. Die Geschäftslogik wird bei dieser Architektur also primär vom Server auf den Client ausgelagert. Der Server wird dann nur für das Nachladen von Daten mittels AJAX benötigt.

¹⁵Vgl. Pechhacker, „Single Page Webapplikationsentwicklung anhand des JavaScript Frameworks Ember.js“, S. 10.

¹⁶Abbildung ähnlich: <https://blog.4psa.com/an-intro-into-single-page-applications-spa/>, besucht am 01.04.2017

¹⁷Vgl. <https://www.google.at/maps>, besucht am 03.04.2017

¹⁸Vgl. Pechhacker, „Single Page Webapplikationsentwicklung anhand des JavaScript Frameworks Ember.js“, S. 11.

AJAX steht „Asynchronus JavaScript and XML“ und ist die moderne Technologie, die das nachträgliche Laden von Inhalten, ohne zusätzliche Plugins wie etwa bei Microsoft Silverlight, ermöglicht¹⁹. Die neu geladenen Daten können dann ganz einfach mittels JavaScript an der passenden Stelle im Document Object Model dynamisch eingefügt werden.

Die wesentlichen Fakten zu SPAs sind²⁰:

- Ein Großteil der Daten werden während des Lebenszyklus nur einmalig geladen.
- Eine gute Suchmaschinenoptimierung ist für SPAs nur schwer durchzusetzen, da es ja im Grunde nur eine Seite ist. Dadurch können sie im Vergleich zu klassischen Websites weniger gut von Diensten wie Google oder Yahoo indiziert werden.
- SPAs können sehr einfach mit Tools wie etwa dem Google Chrome Browser debuggt werden.
- Durch das Auslagern der Geschäftslogik können sie teilweise auch Offline funktionieren.
- Funktioniert nur, wenn die Ausführung von JavaScript im Browser aktiviert ist.

2.4.3 Fazit

Für die Therapeutenverwaltung medtec wurde die Architektur der Single Page Applications gewählt, weil sie das Potenzial haben, einige Nachteile von statisch übertragenen Webanwendungen aufzuheben, zum Beispiel das schlechtere Interaktionsverhalten. Das wiederholte Neuendern und Übertragen der Seite hat einen negativen Einfluss auf die User-Experience, da die Latenzen bei der Übertragung nicht vermieden werden können. Das kann dazu führen, dass der Benutzer die Webanwendung wegen zu langen Ladezeiten verlässt. Jedoch haben die SPA im Vergleich zu statischen Webanwendungen nicht nur Vorteile. Da die Logik auf den Client ausgelagert wird, ist zwingend die Aktivierung von JavaScript notwendig. Ohne aktiviertes JavaScript können eine große Anzahl von Inhalten der Anwendung nicht angezeigt werden.

2.5 Trennung der Komponenten

Bei der Entwicklung von Software empfiehlt es sich, die Anwendung auf verschiedene Komponenten aufzuteilen. Über die Jahre haben sich viele Methoden zur Trennung bewährt. Eine davon ist die **Drei-Schichten Architektur**. Hier empfiehlt es sich zum Beispiel die Benutzeroberfläche, Geschäftslogik sowie Datenhaltung voneinander zu trennen. Dadurch ergeben sich drei verschiedene Schichten, die hierarchisch nach ihrem Abstraktionsniveau aufgebaut sind²¹:

- **Schicht der Präsentation**
Enthält alle Elemente, die benötigt werden, damit der Nutzer mit dem System interagieren kann.
- **Schicht der Geschäftslogik**
Schnittstelle zwischen Präsentation- und Datenhaltungsschicht.
- **Schicht der Datenhaltung**
Wird meist mit Datenbank gelöst. Speichert Informationen dauerhaft.

¹⁹Vgl. <http://adaptivepath.org/ideas/ajax-new-approach-web-applications>, besucht am 02.04.2017, übersetzt vom Autor.

²⁰Vgl. Skolski, *Single Page Application vs. Multiple Page Applications*, übersetzt vom Autor.

²¹Vgl. Balzert, *Java: Objektorientiert programmieren*, S. 220-221.

Welche Vor- und Nachteile diese modulare Trennung mit sich bringt und wie es konkret bei der Therapeutenverwaltung mit AngularJS und dem MVC-Pattern umgesetzt wurde, wird im nachfolgenden Abschnitt erklärt.

2.5.1 Model-View-Controller

Wie viele andere Frameworks und Bibliotheken basiert auch AngularJS auf dem Prinzip des Model-View-Pattern (MVC) Entwurfsmusters.

Dieses Muster sieht es vor, die Software, wie in Abbildung 2.8 ersichtlich, in drei Komponenten zu unterteilen: Datenmodell(*engl. Model*), Präsentation(*engl. View*) und Programmsteuerung(*engl. Controller*)²².

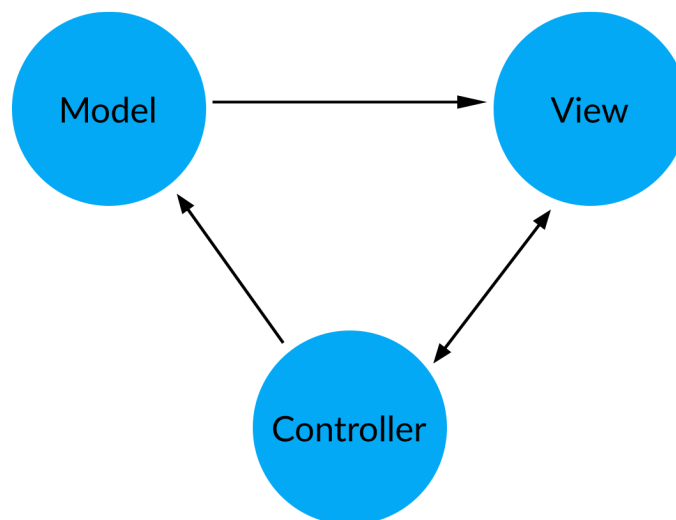


Abbildung 2.8: Model-View-Controller Pattern

Durch die Aufteilung der Anwendung in die Teilbereiche der Logik und Darstellung ergeben sich viele Vorteile. Die einzelnen Komponenten werden übersichtlicher und können dadurch besser ausgetauscht oder gewartet werden. Zusätzlich dazu ist es einfacher bestehende Systeme und Anwendungen mit neuen MVC-Komponenten zu erweitern²³.

2.5.2 Model

Das Model enthält alle notwendigen Daten betreffend der Zustands- und Anwendungslogik. Es ist die Schnittstelle zwischen View und Controller, weiß allerdings nicht von den beiden anderen Komponenten. Bei Änderungen bezüglich des Zustands wird die View benachrichtigt und gegebenenfalls aktualisiert²⁴.

²²Vgl. Steyer, *Angular JS - Moderne Webanwendungen und SPAs mit JavaScript*, S. 10.

²³Vgl. Steyer, *Angular JS - Moderne Webanwendungen und SPAs mit JavaScript*, S. 34-35.

²⁴Vgl. Eric Freeman, *Entwurfsmuster von Kopf bis Fuß*, S.535-537.

Beim Listing 2.1 wird das *Model* anhand des Bearbeiten eines Therapeuten exemplarisch gezeigt. Das JavaScript-Objekt *current* repräsentiert den ausgewählten Therapeuten, der bearbeitet wird. Mit der speziellen Auszeichnung von *ng-model* wird dem Eingabefeld ein Wert des Objektes zugewiesen.

Listing 2.1: Model beim Bearbeiten eines Therapeuten

```

1 <div class="mdl-textfield mdl-js-textfield inputHalf ">
2   <label class="namingInput ">Vorname:</label>
3   <input type="text" id="currentFirstname" ng-model="current.firstname" maxlength=
    ="20 ">
4   <label class="mdl-textfield__label "></label>
5 </div>
6 <div class="mdl-textfield mdl-js-textfield inputHalf ">
7   <label class="namingInput ">Nachname:</label>
8   <input type="text" id="currentLastname" ng-model="current.lastname" maxlength="
    20 ">
9   <label class="mdl-textfield__label "></label>
10 </div>
11 <div class="mdl-textfield mdl-js-textfield inputHalf ">
12   <label class="namingInput ">Telefonnummer:</label>
13   <input type="text" id="currentTelnr" ng-model="current.telnr" maxlength="15"
    pattern="\d{1,15}">
14   <label class="mdl-textfield__label "></label>
15 </div>
16 <div class="mdl-textfield mdl-js-textfield inputQuarter ">
17   <md-checkbox ng-model="current.isActive" >
18     Aktiver Therapeut
19   </md-checkbox>
20 </div>
21 <div class="mdl-textfield mdl-js-textfield inputQuarter ">
22   <md-checkbox ng-model="current.isAdmin" >
23     Leitender Therapeut
24   </md-checkbox>
25 </div>

```

2.5.3 View

Die View ist, ähnlich der Präsentationsschicht bei der Drei-Schichten-Architektur und stellt die sichtbare Schnittstelle zum Benutzer dar. Sie erhält vom Model die benötigten Informationen und liefert eine Präsentation der Daten. Bei Interaktionen, wie zum Beispiel dem Drücken eines Buttons, teilt die View das dem Controller mit.

In der Abbildung 2.9 wird gezeigt, wie die Darstellung der Daten in der View aussieht. Die Texteingabefelder und Checkboxes sind direkt an die Attribute des JavaScript-Objektes gebunden und werden bei Änderungen sofort aktualisiert. Auf das konkrete Aussehen eines Therapeuten-Objektes wird später nochmals eingegangen.

The screenshot shows a web application interface with a modal dialog titled "Bearbeiten von Alexander Jung". The dialog is overlaid on a background that appears to be a list of therapists. The dialog contains the following fields and controls:

- Email:** alexander.jung@wienerneustadt.lknoe.at
- Bereich:** Physiotherapie (selected from a dropdown menu)
- Vorname:** Alexander
- Nachname:** Jung
- Telefonnummer:** 0644987654321
- Aktiver Therapeut:** ☒ (checked)
- Leitender Therapeut:** ☐ (unchecked)
- Buttons:** PASSWORT ZURÜCKSETZEN, SPEICHERN, ABBRECHEN

Abbildung 2.9: View beim Bearbeiten eines Therapeuten

2.5.4 Controller

Der Controller kümmert sich um Eingaben und Anfragen des Benutzers und stellt das Bindeglied zwischen Model und View dar. Er sollte sowohl von Model, als auch von View unabhängig sein. In der Regel fordert er das Model auf, den Zustand zu ändern, kann aber auch direkt den Zustand der View verwalten.

Im Listing 2.2 wird vereinfacht dargestellt, wie das Bearbeiten auf Controller-Seite erfolgt. Durch die Datenbindung werden Veränderungen der Werte in der View beim Therapeuten-Objekt im Controller automatisch aktualisiert. Wenn der Benutzer jetzt auf *Speichern* klickt, wird im *TherapeutenCtrl* ein asynchroner Funktionsaufruf mit dem Daten des Therapeuten zum Aktualisieren an den Server gesendet. Bei erfolgter Durchführung wird der Dialog zum Bearbeiten geschlossen, eine Meldung ausgegeben und das *current-Objekt* des Therapeuten auf *null* gesetzt.

Listing 2.2: Vereinfachter Controller beim Bearbeiten eines Therapeuten

```

1 function TherapeutenCtrl($scope, $http, EmployeeService, $log) {
2
3     $scope.saveDialogEditTherap = function () {
4         EmployeeService.updateEmployee($scope.current).then(function (result) {
5             $log.log("Therapeut wurde bearbeitet.");
6             $scope.current = null;
7             $scope.closeDialogEditTherap();
8         });
9     }
10 }
11 ctrl.controller("TherapeutenCtrl", TherapeutenCtrl);

```

2.6 Technologien zur Datenbeschaffung

Dieses Unterkapitel befasst sich damit, wie im Frontend Daten bereitgestellt werden. Immer mehr Webanwendungen beziehen heutzutage ihre Daten über eine Web-API. Die Kommunikation zwischen einzelnen Anwendungen oder Client und Server kann über viele verschiedene Arten geschehen. Mit Abstand am bekanntesten in der modernen Webentwicklung sind die „eXtensible Markup Language“ (**XML**), „JavaScript Object Notation“ (**JSON**) und „Yet Another Markup Language“ (**YAML**). Jedes Format hat seine Besonderheiten, Stärken sowie Schwächen und eignet sich für ein bestimmtes Anwendungsgebiet. Im Nachfolgenden werden die drei Formate kurz vorgestellt und die Vorgehensweise während der Entwicklung von medtec beschrieben. Für das selbe Therapeut-Objekt finden Sie in den nächsten Abschnitten entsprechende JSON, XML und YAML-Dokumente, um die Eigenheiten besser zu veranschaulichen.

2.6.1 JSON

Die **JavaScript Object Notation**, kurz JSON ist ein Format zum Austausch von Daten, welches ursprünglich von Douglas Crockford spezifiziert wurde. Es folgt zwei verschiedenen Standards, RFC 7159 von Crockford und ECMA-404.²⁵ Primär wird es für die Serialisierung von Objekten zur sprachenunabhängigen Kommunikation verwendet. Allerdings können auch Arrays, Records, Structs, Dictioanrys, Hash Tables oder Assoziative Arrays damit serialisiert werden. Die Syntax dieser kompakten Notation läuft auf Basis der JavaScript Programmiersprache und strukturiert Daten auf einfachste Weise. Jedes korrekte JSON-Dokument ist also ein direkt ausführbarer JavaScript-Code²⁶.

Listing 2.3: Therapeut in JSON

```
1 {  
2   "email": "max@mustermann.com",  
3   "firstname": "Max",  
4   "lastname": "Mustermann",  
5   "occupation": "Verwaltung",  
6   "telnr": 0043699123456,  
7   "entryDate": "2016-12-31",  
8   "permission": 1,  
9   "password": "$2y$10$KHmWDbCgrqjU10$KHmWDb10$KHmWDb"  
10 }
```

Das obenstehende Listing 2.3 zeigt, dass JSON für den Menschen gut lesbar und für Maschinen sehr einfach verarbeitbar ist. In JSON gibt es Elemente wie Objekte, Arrays, Strings, Zahlen, Boolesche Werte, sowie den Spezialwert „null“²⁷. Ein Objekt in JSON mit geschwungenen Klammern ausgezeichnet und stellt eine unsortierte Sammlung von *Key/Value-Paaren*, die diverse Eigenschaften mit jeweiligen Werten repräsentieren dar. Die einzelnen Key/Value-Paare werden mit Beistrichen getrennt.

²⁵Vgl. Niemann, *JSON (JavaScript object notation)*.

²⁶Vgl. Niemann, *JSON (JavaScript object notation)*.

²⁷Vgl. Niemann, *JSON (JavaScript object notation)*.

2.6.2 XML

Die **eXtensible Markup Language**, kurz XML ist eine Auszeichnungssprache, deren Inhalte hierarchisch angeordnet sind und zur Datenübertragung von einem System in ein anderes System verwendet wird. Als universelles Konzept zur Datenspeicherung von W3C, dem World Wide Web Consortium wurde es erstmals am 10. Februar 1998 in der Version 1.0 spezifiziert und besteht aus Textdateien, standardmäßig mit UTF-8-Kodierung. XML-Dokumente bestehen immer aus einer Instanz mit dem tatsächlichen Inhalt und können zusätzlich noch eine Dokumenttypdefinition (DTD oder XML-Schema), welche die Struktur und Grammatik beschreibt beinhalten. Die Informationen werden als Inhalt zwischen einem `<Starttag>` und einem `<Endtag>` oder als Attribute eines Tags ausgeschrieben²⁸.

Listing 2.4: Therapeut in XML

```

1 <employee enr="1">
2   <email>max@mustermann.com</email>
3   <name>
4     <firstname>Max</firstname>
5     <lastname>Mustermann</lastname>
6   </name>
7   <occupation>Verwaltung</occupation>
8   <telnr>00436991234567</telnr>
9   <password>$2y$10$KHmWDbCgrqjU10$KHmWDb10$KHmWDb</password>
10  <entryDate>2016-12-31</entryDate>
11  <permission>1</permission>
12 </employee>

```

Im Listing 2.4 sieht man, wie ein Therapeuten-Objekt in XML dargestellt werden könnte. Es ist erkennbar, dass der Code in diesem Format für den Menschen, dank aussagekräftiger Beschreibung der Elemente sehr gut lesbar ist. Die Werte in den Tags repräsentieren die einzelnen Eigenschaften des Objektes. Dieser Therapeut heißt im Nachnamen **Mustermann** und arbeitet im Bereich **Verwaltung**. Spezielle Informationen können auch als Attribute beschrieben werden. So hat unser Beispielstherapeut die eindeutige ENR, also Employee-Nummer **1**. Besonders beim Therapeutenamen erkennt man, dass XML eine Baumstruktur aufweist, welches die Verarbeitung für Maschinen besonders vereinfacht. Zusätzlich dazu hat man die Möglichkeit mit der XML-Sprache **XPath** den Gesamtbaum oder Teilbäume zu analysieren, verarbeiten oder extrahieren. Es eignet sich also zu viel mehr, als nur zur Serialisierung von Objekten. Es gibt sehr viele Markup-Sprachen, die auf XML basieren wie z.B: XHTML, XAML, MathML (**Mathematische Formelsprache**) oder auch SVG (**Scalable Vector Graphics**)²⁹.

2.6.3 YAML

Die „**Yet Another Markup Language**“, kurz YAML ist eine Auszeichnungssprache mit der Daten serialisiert werden. Spezifiziert wurde es 11. Mai 2001 von Clark Evans, Brian Ingerson und Oren Ben-Kiki³⁰. Es ist einfach zu verwenden, einfach zu lesen und wurde in Anlehnung an XML zum Datenaustausch zwischen verschiedenen Programmiersprachen entwickelt. Besonders häufig wird es in der Programmiersprache „Ruby on Rails“ und bei Konfigurationsfiles oder Logfiles verwendet.

²⁸Vgl. Hillebrand, *Datenbanken und Informationssysteme*, S. 123-129.

²⁹Vgl. Schonmann, *Einführung in XML*.

³⁰Vgl. <http://www.yaml.org/spec/1.2/spec.html>, besucht am 03.04.2017, übersetzt vom Autor

Listing 2.5: Therapeut in YAML

```
1 ---
2 email: max@mustermann.com
3 firstname: Max
4 lastname: Mustermann
5 occupation: Verwaltung
6 telnr: 0043699123456
7 entryDate: '2016-12-31'
8 permission: 1
9 password: $2y$10$KHmWDbCgrqjU10$KHmWDb10$KHmWDb
10 ...
```

Gekennzeichnet werden YAML-Sequenzen mit drei Bindestrichen am Anfang und drei Punkten am Ende. Ähnlich wie in JSON erkennt man im oberen Listing, dass YAML die Daten mit *Key/Value-Paaren* strukturiert. Hierbei werden die einzelnen Paare aber nicht durch ein Komma getrennt, sondern über einen Zeilenumbruch. Außerdem ist es hierbei möglich Kommentare im Code zu verfassen, was speziell für Dokumente nützlich ist, bei denen der Mensch bewusst Änderungen durchführt. YAML ist eine Übermenge von JSON, was zur Folge hat, dass jedes gültige JSON-Dokument auch ein gültiges YAML-Dokument darstellt. Weitere mächtige Features bei dieser Sprache sind die Unterstützung von selbst-erstellten Datentypen und Referenzierungen zu anderen Key/Value-Paaren innerhalb des YAML-Code³¹.

2.6.4 Fazit

Lange Zeit galt XML als Nummer eins Datenaustauschformat zwischen Server und Client. Doch inzwischen hat das kompakte JSON-Format das ältere XML im Bereich der sprachunabhängigen Kommunikation weitgehend verdrängt. Sowohl die serverseitigen Programmiersprachen PHP und node.js als auch die clientseitige Programmiersprache JavaScript stellen Funktionen von Haus aus zur Verfügung, welche das Arbeiten mit JSON-Dokumenten vereinfacht.

Für die Markup Language YAML ist das erst mit zusätzlichen Erweiterungen möglich. Verwendet man es im Bereich des Datenaustauschs, so gehen einige mächtige Funktionalitäten, wie die eigenen Datentypen oder die Möglichkeit zur Referenzierung innerhalb des eigenen YAML-Dokument verloren.

Vergleicht man das vorherige Therapeuten-Objekt von JSON mit jenem von XML, erkennt man, dass es wesentlich kürzer ist, da die auszeichnenden „Tags“ wegfallen. Während in XML mit den Start- und Endtags ein gewisser Overhead (Daten die nicht direkt nützlich sind) entsteht, werden in JSON die Key/Value-Paare mit einfachen Kommas getrennt, was die Lesbarkeit für Mensch und die Verarbeitbarkeit für Maschinen um ein Vielfaches vereinfacht. Man kann also die gleiche Menge an Informationen wesentlich kompakter transportieren, was kürzere Ladezeiten mit sich trägt. Hinsichtlich der Performance einer Webanwendung, vorallem bei größeren Objekten ist also in diesem Bereich JSON zu bevorzugen. Außerdem werden die Dokumente dieses Formates, wie schon zuvor erwähnt, direkt als Javascript-Code geparkt, was ein weiteres wichtiges Kriterium in der Verarbeitbarkeit darstellt.

³¹Vgl. Fender, *YAML vs. JSON*.

Kapitel 3

Backend (PT)

3.1 Einleitung

Dieses Kapitel befasst sich mit den technischen Aspekten und Überlegungen, die sich primär auf das Backend der Webapplikation beziehen. Die Ansprüche an eine moderne Webanwendung steigen immer weiter, dazu gehören auch die Sicherheitsanforderungen einer Anwendung. Da die medtec-Anwendung vor allem mit vertraulichen Daten umgeht, liegt ein besonderer Fokus auf der Absicherung vor Angriffen auf die Webanwendung.

3.2 Evaluierung von PHP

PHP ist eine serverseitige Skriptsprache welche seit 1995 in dynamischen Webseiten und Webanwendungen verwendet wird. Durch diese lange und stetige Präsenz hat sich eine große Community um PHP gebildet, was eine große Auswahl an Frameworks, Tutorials und Hilfen zur Folge hat. Dadurch ist es sehr einfach die Sprache zu erlernen oder anfallende Probleme zu beheben.

Ein weiteres Argument für PHP ist die weite Verbreitung bei Webpace Providern sowie die einfache Aufsetzung eines PHP fähigen Servers wie zum Beispiel des „Apache HTTP Servers“.

Mit der aktuellsten Version, PHP 7, wurde zusätzlich die Performance deutlich verbessert. Dies führt dazu, dass PHP 7 im Vergleich zu seinen Konkurrenten, keinen Performance-nachteil im Bereich der klein- bis mittelgroßen Webanwendungen mit sich bringt.

PHP ist außerdem eine Open-Source Skriptsprache, dies bringt den Vorteil der ständigen Weiterentwicklung durch die Community mit sich. Jedoch schadet dies ebenso der Sicherheit, da durch den offenen Zugang Schwachstellen im Code leichter gefunden und ausgenutzt werden können. Dies erhöht den Aufwand für ein sicheres System. Der dadurch entstehende Aufwand ist im Rahmen dieses Projektes jedoch minimal und mit wenigen Arbeitsstunden abgedeckt.

3.3 Definition einer Webanwendung

Eine Webanwendung muss nicht über was „World Wide Web“ erreichbar sein, sie kann auch nur in einem internen Netzwerk existieren. Da der Name Webanwendung vom Einsatz von Webtechnologien kommt, kann folgende Definition eine Webanwendung beschreiben:

Eine Webanwendung ist eine Client-Server-Anwendung, die auf Webtechnologien (HTTP, HTML etc.) aufsetzt¹.

Die Architektur der Webanwendung lässt sich hierbei in einer einfachen 3-Tier-Architektur (Drei-Schichten-Architektur) darstellen (siehe Abb. 3.1)

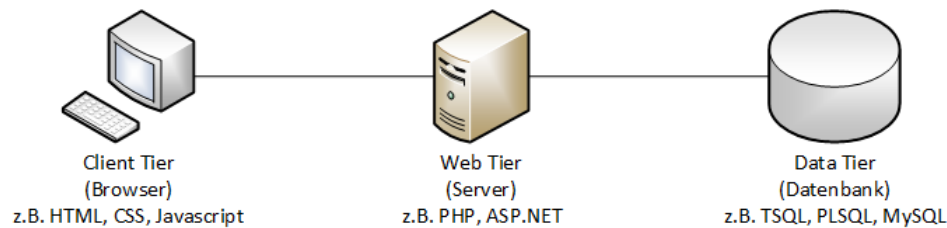


Abbildung 3.1: 3-Tier-Architektur²

3.4 Sicherheitsprobleme von Webanwendungen

Die Gründe für Sicherheitsmängel sind meist nicht ausschließlich technischen Ursprungs, bilden aber den größeren Anteil der möglichen Bedrohungen. Da die Anwendung mit vertraulichen Daten arbeitet, muss eine möglichst hohe Sicherheitsstufe gewährleistet werden.

3.4.1 Technische Ursachen

- **Netzwerklayer und Anwendungslayer:**

IT-Sicherheitsmaßnahmen und Angriffe unterscheidet man grundsätzlich auf 2 Ebenen. Jene, die auf dem Netzwerklayer wirken, und jene, die auf dem Anwendungslayer wirken.

1. **Netzwerklayer:**

Unter dem Netzwerklayer versteht man alle Sicherheitsmaßnahmen die über Infrastrukturkomponenten umgesetzt werden, wie zum Beispiel Netzwerkfirewalls oder IDS³ / IPS⁴ Systeme.

Das Problem besteht darin, dass diese Systeme hauptsächlich auf der Ebene des TCP/IP-Protokolls agieren und somit keinen Schutz vor Angriffen über die höher liegenden Protokolle des Anwendungslayers bieten.

2. **Anwendungslayer:**

Unter dem Anwendungslayer versteht man alle Protokolle, die zur Kommunikation zwischen der Anwendung und den weiter unten liegenden Schichten verwendet werden. Zum Beispiel das HTTP Protokoll, also die höchste Schicht die mit dem Server kommuniziert.

Webanwendungen sind dabei in den überwiegenden Fällen Angriffen über den Anwendungslayer ausgesetzt.

Bereits im Jahr 2003 ging die NIST davon aus, dass 75% der Angriffe über den Anwendungslayer durchgeführt werden, vergleichsweise machten Angriffe über den

¹Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 1.

³IDS = Intrusion Detection System

⁴IPS = Intrusion Prevention System

Anwendungslayer im Jahr 2010 bereits 86% aller Angriffe aus⁵.

Die dargestellte Abbildung (Siehe Abb. 3.2) verdeutlicht die Kommunikation zwischen einem Client und einem Server, sowie den beteiligten Protokollen.

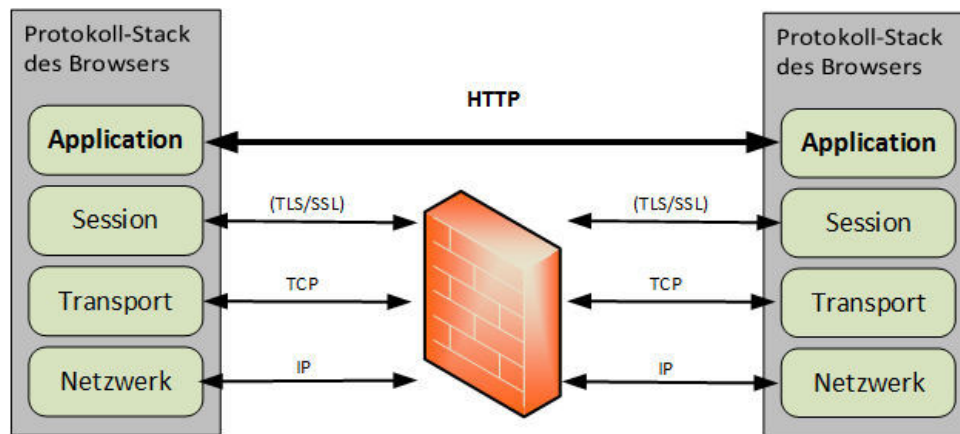


Abbildung 3.2: Netzwerk- vs Anwendungslayer⁶

- **Das HTTP Protokoll:**

HTTP ist das zugrundeliegende Übertragungsprotokoll für den Großteil aller Webtechnologien die in modernen Webanwendungen verwendet werden. Die Spezifikation erfolgte bereits in 1981⁷. Die damaligen Designziele waren vor allem **Robustheit** und **Interoperabilität**, ähnlich dem TCP Protokoll.

Ein Zitat von Jon Postel, der auch „God of the Internet“ zu seiner Lebzeit genannt wurde:

*"An implementation should be conservative in its sending behavior and liberal in its receiving behavior."*⁸

Zu Deutsch: „Sei konservativ in dem, was du sendest, und liberal in dem, was du von anderen akzeptierst.“

Daraus ist klar erkennbar, dass der Punkt Sicherheit nicht zu den Prioritäten der damaligen Entwickler des HTTP Protokolls galt. Im Laufe der Jahre wurde HTTP von Version 0.9, über 1.0 auf den aktuellen Stand 1.1 gebracht. An den Sicherheitsaspekten wurde jedoch nur vereinzelt gearbeitet.

Zur Verdeutlichung der geringen Sicherheitsansprüche, zeigt Listing 3.1 einen „HTTP Request“ unter 1.1, der mit nur 2 Zeilen eine Ressource aus dem Web abrufen.

Listing 3.1: HTTP-Request

```
1 GET /login.php HTTP/1.1
2 Host: www.medtec.co.at
```

Die erste Zeile definiert, um welche Ressource es geht und was mit dieser geschehen soll, während die zweite den Host definiert, von welchem die Ressource angefordert

⁵Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 4.

⁷Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 4.

⁸Vgl. *Request for Comments*, p. 21.

werden soll.

Auf solch einen Request sendet der Server eine einfache Antwort in der Form einer „HTTP Response“.

Listing 3.2: HTTP-Response

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html
3 Content-Length: 3177
4 Date: Tue, 02 Mar 2017 20:06:10 GMT
5
6 <html ng-app="medtec">
7
8 <head>
9   <title>Login | medtec</title>
10  <meta charset="UTF-8">
11  <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  <link rel="stylesheet" href="css/style.css" />
13  <link rel="shortcut icon" type="image/x-icon" href="favicon.ico">
14 </head>
15 <body class="login">
16   [HTML-Markup for Login Page]
17 </body>
18
19 </html>
```

Dieser HTTP Request besteht aus dem Header und dem Body.

Im Header wird ein Status mit Statustext angemerkt, welcher die Antwort der angeforderten Aktion beschreibt. Weiters wird ein Content-Type spezifiziert, der den Typ der eigentlichen Nachricht spezifiziert. Danach wird noch die Größe der Ressource in Bytes und der Zeitpunkt der letzten Änderung mitgeschickt.

HTTP ist unverschlüsselt. Das ist der Grund warum es das Protokoll HTTPS seit 1994 gibt. HTTP wird dabei durch einen zusätzlichen SSL/TLS-Layer erweitert (Siehe Abb. 3.2).

Dieser Layer sorgt für eine verschlüsselte Übertragung, sowie eine Authentifizierung beider Seiten durch SSL-Zertifikate. Vor allem Banken, Webshops und ähnliche Webservices über welche Geldtransaktionen stattfinden, verwenden HTTPS. Leider ist HTTPS keine zwingende Vorschrift und wird daher im Großteil der heutigen Webkommunikationen nicht verwendet, was wiederum bedeutet, dass all diese Kommunikationen **unverschlüsselt** stattfinden.

Angreifer haben dadurch ein leichtes Spiel potenziell sensible Daten abzufangen und zu manipulieren. Weitere große Probleme sind der fehlende Integritätsschutz sowie die unsicheren Authentifizierungsverfahren (HTTP Basic & Digest).

Um die Webanwendung vor Angriffen über das HTTP-Protokoll zu schützen, verwendet der Server die HTTPS-Erweiterung sowie Überprüfung der Session vor jeder administrativen Aktion.

3.4.2 Know-How-bedingte Ursachen

- **Offenlegung serverseitiger Funktionen:**

Eine große Anzahl an Webanwendungen basiert auf sogenannten „Web-Content-Management-Systemen“ wie zum Beispiel Typo3 oder Wordpress. Solche Systeme haben für gewöhnlich webbasierte, administrative Schnittstellen die auf einen bereits vorgefertigten

Code zugreifen.

Diese Art der Schnittstelle, ist jedoch nicht nur für den Administrator der Webanwendung erreichbar. Hacker könnten versuchen, über nicht ausreichend geschützte Funktionen, welche für die Funktionalität der Webanwendung nicht notwendig sind, sich Zugang zum Backend zu verschaffen.

Durch die zunehmende Anforderung an Webanwendungen die gleiche Funktionalität wie Desktopanwendungen zu liefern, werden immer mehr Funktionen clientseitig abgebildet. Oft ohne dabei die entstehende Bedrohung wahrzunehmen. Problematisch dabei sind nicht nur Schnittstellen die serverseitige Funktionen offenlegen, sondern auch „versteckte“ Felder die zur Parametrisierung der Anwendung dienen. Alle Daten die auf dem Client erfasst und verarbeitet werden, können manipuliert werden. Dazu zählen auch jene Daten, die an den Server weitergegeben werden.

„In der Praxis wird ein Großteil der Angriffe gegen Webanwendungen über die Manipulation von Anwendungsparametern durchgeführt“⁹.

In der medtec-Webanwendung wird diese Situation vermieden durch den Verzicht auf versteckte Felder, ein selbstimplementiertes Content-Management-System sowie serverseitiger Verarbeitung und Validierung aller sensiblen Daten.

- **Unzureichende Absicherung der Backendsysteme:**

Es ist keine Seltenheit, dass die Absicherung einer Webanwendung rein im Frontend durch Eingabefilter erfolgt. Die weitere Kommunikation zu dahinterliegenden Systemen lässt sich damit jedoch nicht ausreichend validieren. Überwindet ein Angreifer diese Filter, ist das gesamte System ungeschützt.

Weiters bringt solch ein „Single Point of Security“ die Gefahr, dass der Angreifer diesen umgeht und ein Test- oder „Legacy“-System, welches ebenfalls von außen erreichbar ist angreift, da diese Systeme selten im Sicherheitsfokus des Betreibers liegen¹⁰. Die medtec-Webanwendung ist deswegen auf *3 Ebenen* abgesichert.

1. Auf der *Client-Seite* durch Eingabefilter.
2. Auf der *Server-Seite* durch eine Validierung der Daten, bevor diese an die Datenbank weitergeben werden.
3. In der *Datenbank* um den Eintrag „unmöglicher“ Daten zu verhindern.

- **Die Bedrohungslage wird unterschätzt:**

Viele Angriffe auf Webanwendungen hätten, durch das bloße Bewusstsein einer Gefahr bereits verhindert werden können. Trotzdem versuchen viele Unternehmen Kosten zu sparen und möglichst günstig ein ausreichendes Sicherheitslevel zu erreichen. Oft wird die Gefahr dabei so sehr unterschätzt, dass komplett auf professionelle Absicherungen verzichtet wird.

Angriffe werden dadurch begünstigt, dass Sicherheit in diesem Bereich asymmetrischer Natur ist¹¹. Dies bedeutet, dass ein Angreifer kaum mehr als einen Internetanschluss und einige kostenlose Tools benötigt, um Angriffe durchzuführen. Im Vergleich dazu braucht ein Unternehmen eventuell ganze Abteilungen, aufwendige Vorausplanung und teure Software um Angriffe zu kontern.

Vergleichsweise steigt die Cyberkriminalität immer weiter und wird immer organisierter.

⁹Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 14.

¹⁰Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 20.

¹¹Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 24.

„Denn längst haben Kriminelle, bis hin zum organisierten Verbrechen, die Möglichkeit von Cyberangriffen erkannt. Insgesamt lassen sich heute drei aus 4 Vorfällen im Bezug auf den Diebstahl personenbezogener Daten auf jene Tätergruppen zurückführen [...]“¹².

*“Studies estimate that the Internet economy annually generates between \$2 trillion and \$3 trillion, [...].
If our estimates are right, cybercrime extracts between 15% and 20% of the value created by the Internet.”¹³.*

Zu Deutsch:

*„Studien schätzen, dass von den 2 - 3 Trillionen US Dollar die die Internetwirtschaft jährlich generiert, [...].
Sollten unsere Schätzungen stimmen, werden zwischen 15% und 20%, des Wertes den das Internet erzeugt, von Cyberkriminalität gestohlen“¹⁴.*

Diese Zahlen zeigen auf wie groß die Bedrohung durch Angreifer wirklich ist und warum diese leicht unterschätzt wird.

3.5 Sicherheitsmechanismen für Webanwendungen

Es gibt eine Vielzahl an Sicherheitsmechanismen (Security Controls) um eine Webapplikation vor Übergriffen zu schützen. Dabei unterscheidet man drei Varianten von Sicherheitsmaßnahmen¹⁵:

- Primäre Maßnahmen, dienen der ursächlichen Behebung einer Schwachstelle.
- Sekundäre Maßnahmen, beheben nicht das ursächliche Problem, können jedoch die Auswirkung eines Angriffes abschwächen.
- Additive Maßnahmen, können ergänzend aber nicht als Alternative zu entsprechenden primären und sekundären Maßnahmen agieren.

Der Fokus sollte dabei auf der Umsetzung primärer und damit präventiver Maßnahmen liegen. In der medtec-Webanwendung sind ausschließlich primäre Maßnahmen umgesetzt.

3.5.1 Datenbanksicherheit durch die Art der Einbindung

Bei der Einbindung der Datenbank in die Webanwendung gibt es grundsätzlich 2 wichtige Unterschiede:

1. Persistente Verbindungen:

Bei persistenten oder auch „keep-alive“ Verbindungen wird die Verbindung zur Datenbank für die gesamte Länge der Sitzung offen gehalten, auch nach Abschluss der letzten Anweisung. Diese Art der Verständigung bringt keine zusätzliche Funktionalität mit sich, das bedeutet, es kann jede persistente Verbindung durch eine nicht persistente ersetzt werden, ohne den Verlust von technischen Möglichkeiten.

Eine persistente Verbindung hat jedoch gewisse Vorteile:

¹²Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 24-25.

¹³Vgl. Net Losses: *Estimating the Global Cost of Cybercrime*, p. 7.

¹⁴Vgl. Net Losses: *Estimating the Global Cost of Cybercrime*, S. 7, Übersetzt vom Autor.

¹⁵Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 114-115.

- Kürzere Antwortzeiten durch das Wegfallen des erneuten Verbindungsaufbaus
- Verringerung des „**Overhead**“ bei Abfragen

Ein großer Nachteil dieser Art der Kommunikation ist die einmalige Authentifizierung für die gesamte Session.

2. Nicht-persistente Verbindungen:

Bei nicht-persistenten Verbindungen wird die Verbindung vor jeder Abfrage aufgebaut und danach wieder abgebaut. Hier ist der Aufwand der Rechenleistung im Vergleich zur persistenten Verbindung erhöht. Jedoch haben nicht persistente Verbindungen zwei wichtige Vorteile:

- Sie ermöglichen die Verbindung einer größeren Anzahl von Clients trotz Limits auf die maximalen Verbindungen zur Datenbank.
- Es erfolgt eine erneute Authentifizierung vor jeder Transaktion.

In der medtec-Webanwendung werden nicht persistente Verbindungen verwendet, da das Limit der Datenbank bei maximal 20 gleichzeitigen Verbindungen zur Datenbank liegt. Dieses Limit ist vom Webspaceprovider vorgesehen und kann nur gegen einen Aufpreis erhöht werden. Durch die Bedingung der Auftraggeberin sollten diese Kosten möglichst gering gehalten werden, weswegen eine nicht persistente Verbindung umgesetzt wurde. Als positiven Nebeneffekt wird die Datenbank durch die zusätzlichen Authentifizierungen vor jedem Datenbankzugriff besser abgesichert.

In PHP ist eine persistente bzw. nicht persistente Verbindung sehr leicht umzusetzen, veranschaulicht durch folgendes Beispiel:

Listing 3.3: Datenbank Verbindungsaufbau

```

1 function dboconnect(){
2     $dbConnection = new PDO('mysql:dbname=medtec; host=127.0.0.1;
3                             charset=utf8', 'root', 'pw');
4
5     $dbConnection->setAttribute(PDO::ATTR_EMULATE_PREPARES, true);
6     // no persistent connections!
7     $dbConnection->setAttribute(PDO::ATTR_PERSISTENT, false);
8     // ----- //
9     $dbConnection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
10
11     return $dbConnection;}

```

3.5.2 Das PDO Objekt

*"The PHP Data Objects (PDO) extension defines a lightweight, consistent interface for accessing databases in PHP."*¹⁶

zu Deutsch:

„Die PHP Data Objects(PDO) Erweiterung definiert eine performante und konsistente Schnittstelle zum Zugriff auf Datenbanken in PHP".¹⁷

¹⁶Vgl. *PDO Object*.

¹⁷Vgl. *PDO Object*, Übersetzt vom Autor.

Der große Vorteil des PDO Objektes ist seine Sicherheit gegenüber SQL Injections. Der Aufruf ist dabei sehr einfach gestaltet, wie im folgenden Beispiel zu erkennen ist:

Listing 3.4: Datenbank Datenabruf

```

1 function getEmployeesController() {
2     $employees = array();
3     $pdo = dbconnect();
4     //Statement gets prepared to be sent to DB
5     $getUser = $pdo->prepare("call sp_getEmployees()");
6     // ----- //
7     $getUser->execute();
8     foreach ($getUser as $row) {
9         $employee = parseEmployee($row);
10        array_push($employees, $employee);
11    }
12    $pdo = null; //reset of db connection
13    return $employees;}

```

Erst wird die Verbindung zur Datenbank aufgebaut durch die Funktion „dbconnect()“ (siehe Abb. 3.3).

Danach folgt die „prepare“ Anweisung in der der String auf jegliche Arten von SQL Injections überprüft wird, um diese zu verhindern. Dem folgt die Ausführung der Abfrage in Form eines Aufrufs einer Stored Procedure, die alle angestellten Therapeuten liefert. Die Daten werden verarbeitet und die Verbindung wird mit „\$pdo = null;“ geschlossen.

3.5.3 Schutz vor Injections

Interpreter Injections

Eine der gefährlichsten Arten von Bedrohungen bilden so genannte „Interpreter Injections“.

"Interpreter injection attacks send malicious code that is intended for execution by one of the interpreters that web applications use, both on the server side and in the browser"¹⁸.

zu Deutsch:

„Interpreter-Injection-Angriffe senden böswilligen Code, der von einem der Interpreter, den die Webanwendung verwendet, sowohl Server- als auch Browserseitig, ausgeführt werden soll“.¹⁹

Ein Interpreter ist für die Verarbeitung von Anweisungen zuständig, zum Beispiel²⁰:

- **SQL Interpreter:** verarbeitet SQL-Anweisungen und erzeugt daraus die eigentlichen Datenbankabfragen.
- **XML Interpreter:** parst, angepasst an das jeweilige Schema, XML-Daten in eine interne Datenrepräsentation.
- **HTML Interpreter:** wandelt den vom Server übertragenen HTML-Text in die Darstellung einer Webseite um.

Diese Interpreter werden alle dadurch charakterisiert, dass die verarbeiteten Zeichenketten Kommandos und Steuerzeichen als auch Nutzdaten enthalten. Dies ermöglicht einem

¹⁸Vgl. *Interpreter Injection*.

¹⁹Vgl. *Interpreter Injection*, Übersetzt vom Autor.

²⁰Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 60.

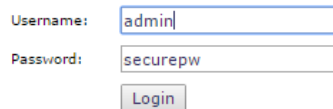
Angreifer, bei fehlendem Schutz, eigene Kommandos einzuschleusen und eventuell sensible Daten auszulesen.

SQL Injections

Die vermutlich bekannteste und kritischste Form einer solchen Injection ist die „*SQL Injection*“. Dabei wird eine SQL-Datenbank, die über einen SQL-Interpreter in der Webanwendung angesprochen wird und im Backend agiert, angegriffen.

Ein primitives Beispiel dazu könnte folgendermaßen aussehen ²¹:

Online Banking Login



Username:

Password:

Abbildung 3.3: Ein gewöhnliches Login um sich in einen Account einzuloggen.

Die im Hintergrund ausgeführte SQL-Abfrage würde normalerweise so aussehen:

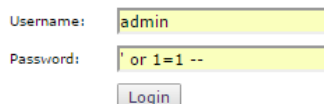
Listing 3.5: SQL-Abfrage für das vorherige Login

```
1 SELECT count(name)
2 FROM users
3 WHERE name = 'admin' AND password = 'securepw';
```

Mit einer simplen Überprüfung ob das zurückgelieferte Ergebnis größer als Null ist, wird überprüft ob die Daten korrekt sind.

Ändert man jetzt das Passwort ab, auf eine Eingabe wie diese:

Online Banking Login



Username:

Password:

Abbildung 3.4: SQL-Injection

Verändert sich damit die Abfrage im Backend auf diese:

Listing 3.6: SQL-Abfrage mit Injection

```
1 SELECT count(name)
2 FROM users
3 WHERE name = 'admin' AND password = '' or 1=1 -- ';
```

Durch das neu eingefügte OR wird die Abfrage immer den Benutzer finden und dadurch einen Wert größer als 1 liefern. Die beiden Bindestriche symbolisieren dabei ein Kommentar in SQL und führen dazu, dass der Interpreter alle folgenden Anweisungen in dieser Zeile ignoriert.

²¹Zur Darstellung des Beispiels wurde <http://testfire.net/bank/login.aspx> verwendet.

Dies führt zu einer Authentifizierung obwohl kein korrektes Passwort eingegeben wurde und ermöglicht dem Angreifer Zutritt in den geschützten Bereich. Diese Art der SQL Injection nennt man auch „Blind SQL Injection“, da der Angreifer versucht blind auf die Datenbank zuzugreifen²².

Oft ist es auch möglich Teile der Abfrage über eine absichtlich ausgelöste Fehlermeldung auszulesen, etwa durch die Eingabe eines ungültigen Ausdrucks.

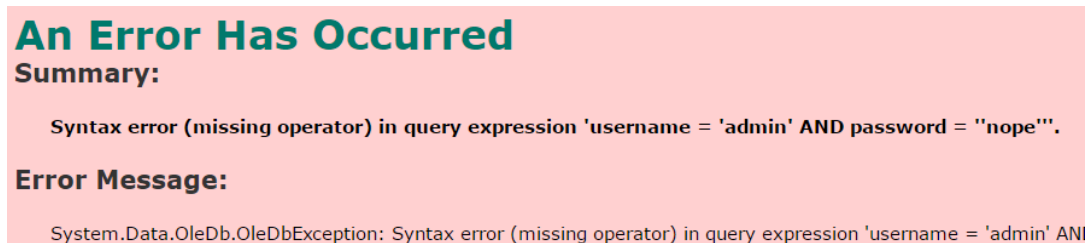


Abbildung 3.5: Syntaxfehler mit Abfrageausschnitt

Solche Injections könnten zum Beispiel dazu verwendet werden neue Benutzer anzulegen, bestehende Daten zu löschen oder die Datenbank herunterzufahren.

In der medtec-Webanwendung werden solche Angriffe durch die Verwendung des PHP Data Objects (siehe 3.5.2) verhindert, sowie durch die Verwendung von Stored Procedures und der Blockierung direkter Abfragen in der Datenbank.

3.6 Funktionalität

Neben der Brückenbildung zwischen dem Frontend und der Datenbank (illustriert in Abb. 3.1), hat das Backend zwei primäre Aufgaben. Die Generierung des Dienstplanes und damit gleichzeitig des Raumplanes, sowie das Informieren aller betroffenen Personen im Falle eines ausfallenden Therapeuten. Die *Vertretungsverwaltung* wird genauer im Kapitel 4.6.4 behandelt.

3.6.1 Kommunikation mit dem Frontend

Die Kommunikation mit dem Frontend erfolgt über sogenannte „AJAX-Calls“. AJAX steht dabei für „Asynchronous JavaScript And XML“, statt XML kann jedoch auch JSON verwendet werden, so wie es in der medtec-Webanwendung auch der Fall ist. Dabei sendet der Client einen vordefinierten Request an den Server, welcher dann über den Namen des Requests die gewünschte Aktion durchführt.

In der Webapplikation sind 82 solcher Requests definiert. Dabei gibt es zwei unterschiedliche Typen an Requests, die „GET“- und „POST“-Requests. Wie auch bei HTTP (siehe 3.4.1) liefert ein „GET“-Request Daten und ein „POST“-Request wird dazu verwendet Daten an den Server zu senden. Im sogenannten „AjaxController“ der Anwendung gibt es ein zentrales Requestmanagement, welches überprüft, ob die Anfrage von einem dazu berechtigten Benutzer kommt.

Das Beispiel 3.7 zeigt Adminspezifische Anfragen, die Daten über die angestellten Therapeuten liefern. Sollte der Anfragende die nötigen Rechte haben, wird der Inhalt der „GET“-Variable „req“ geprüft. Wird dabei eine Übereinstimmung mit einem der Requests

²²Vgl. Rohr, *Sicherheit von Webanwendungen in der Praxis*, S. 62.

festgestellt, wird die darin definierte Funktion ausgeführt. Diese antwortet dem Client mit den entsprechenden Daten, geparkt als JSON.

Listing 3.7: Adminspezifische „GET“-Requests

```

1  ///// Admin Requests ///// start
2  if($_SESSION["admin"]){ //Prüfung ob der Benutzer berechtigt ist
3  ///// GET Requests ///// start
4  //Employee Requests start
5  if($_GET["req"] == "getTherapists"){
6      getTherapists(); //wird ausgeführt wenn der Requestname übereinstimmt
7  }elseif($_GET["req"] == "getTherapistsLight"){
8      getTherapistsLight();
9  }elseif($_GET["req"] == "getTherapistsInactive"){
10     getTherapistsInactive();
11  }elseif($_GET["req"] == "getTherapistsLightInactive"){
12     getTherapistsLightInactive();
13  //end

```

Um diese serverseitige Rechteüberprüfung zu ermöglichen, wird beim Login die Berechtigungsstufe aus der Datenbank abgefragt und in die serverseitige `Session` gespeichert. Diese ist, solange die Session offen ist, jederzeit abrufbar. Weiters ist diese Variable für den Benutzer nicht veränderbar, da sie serverseitig abgespeichert wird und in regelmäßigen Abständen auf ihre Richtigkeit überprüft wird.

3.6.2 Dienstplan

Die Dienstplangenerierung ist die wichtigste Funktion der medtec-Webanwendung. Sie spart dem leitenden Personal beim monatlichen Festlegen des Dienstplanes mehrere Stunden Arbeitszeit. Weiters ist durch die dynamische Anzeige des Dienstplans sofort erkenntlich welcher Therapeut einen anderen vertritt.

Zusätzlich bietet der Dienstplangenerator die Möglichkeit Vorzüge miteinzubeziehen. Dies bedeutet dass die leitende Therapeutin in der Lage ist, Therapeuten für das System zu markieren, die bevorzugt für die ausgewählte Gruppe verantwortlich gemacht werden sollen.

Der genaue Ablauf sieht dabei wie folgt aus:

1. Parsen der vom Frontend empfangenen Daten

Das empfangene JSON enthält alle Gruppen mit den ausgewählte Therapeuten. Jene Gruppen, ohne bevorzugte Therapeuten werden ans Ende gereiht.

2. Iterieren aller Gruppen

Dabei werden folgende Eigenschaften der Gruppe überprüft:

1. **Prüfung ob die Gruppe priorisierte Therapeuten zugeteilt hat.**
Ist dies nicht der Fall, werden alle Therapeuten eingesetzt. Eingesetzte Therapeuten sind nach der Gesamtanzahl ihrer Verantwortlichkeiten sortiert.
2. **Prüfung ob bereits alle Positionen innerhalb der Gruppe besetzt sind.**
Ist dies der Fall, ist die Gruppe abgeschlossen und die nächste Gruppe wird überprüft.

3. Überprüfen aller angegebenen Therapeuten auf ihre Zulässigkeit.

Dabei wird überprüft ob der Therapeut zu den Zeiten, an denen die Gruppe stattfindet, überhaupt arbeitet. Danach wird auf ausschließende Blockaden kontrolliert, denn Therapeuten die an einem Tag zum Beispiel eine Einzeltherapie abhalten, dürfen für keine Gruppe verantwortlich sein. Die nächste Validierung betrifft gewöhnliche Blockaden, also Zeiträume in denen der Therapeut verhindert ist, weil dieser in einer anderen Abteilung gebraucht wird.

4. Einfügen des Therapeuten in die Listen „möglicher“ Therapeuten

Hat der Therapeut all die vorhergehenden Prüfungen bestanden, wird kontrolliert ob dieser bereits Hauptverantwortlicher, erster Stellvertreter oder zweiter Stellvertreter für eine andere Gruppe ist. Ist dies nicht der Fall wird er für die Posten, die er noch nicht belegt, als potenzieller Verantwortlicher vermerkt.

5. Zufällige Auswahl eines zuständigen Therapeuten

An dieser Stelle werden Therapeuten für alle drei Positionen nach dem Zufallsprinzip ausgewählt und an die Datenbank gesendet.

Abschluss

Sind alle Gruppen durchlaufen und eingeteilt, wird der Dienstplan fertig generiert und abgespeichert. Die Raumpläne können automatisch aus diesen Daten herausgelesen und angezeigt werden.

Kapitel 4

App (JK)

4.1 Einführung

Als App wird eine Anwendungssoftware für mobile Geräte (Smartphones) bezeichnet. Bei einer App unterscheidet man zwischen zwei verschiedenen Typen¹:

- **Plattformabhängige App:** wird auch Native-App genannt. Diese Apps sind speziell auf eine einzige Plattform angepasst, das heißt eine programmierte Android App ist zum Beispiel nicht mit einem iPhone kompatibel, da es zwei verschiedene Plattformen sind.

Es wird für jede Plattform eine andere Programmiersprache verwendet (für Android wird Java verwendet, für iOS Objective-C und Swift und für Microsoft Windows C#, C++ und C).

- **Plattformunabhängige App:** hier unterscheidet man zwischen:

Web-App: Diese Variante erfordert keine direkte Installation und wird nur über den Webbrowser auf dem Smartphone aufgerufen.

Hybrid-App: Diese Variante kann auf unterschiedlichen Geräten installiert werden und auch auf mehreren Betriebssystemen laufen. Bei einer Hybrid-App nützt man die Vorteile von nativen und Web-Apps.

Cross-Plattform-App: Eine Cross-Plattform-App ist ähnlich einer Hybrid-App, nur kann diese Anwendung unabhängig der Plattform auf verschiedensten Betriebssystemen ausgeführt werden. Die Benutzeroberfläche wird mit dem APIs des Betriebssystems gebaut.

In dieser Diplomarbeit war das Ziel eine Cross-Plattform App zu programmieren. Der Grund für diese Entscheidung wird im nächsten Abschnitt erklärt.

¹Vgl. Wikipedia, *Mobile App* — Wikipedia, Die freie Enzyklopädie.

4.2 Evaluierung von Ionic

Ionic ist ein von AngularJS bereitgestelltes Frontend-Framework mit einem integrierten Kommandozeilentool, welches den Code als App verpackt, welche dann ganz leicht auf den Plattformen Android, iOS oder Windows-Phone ausgeführt werden kann². Eine der Anforderungen der Auftraggeberin war eine iOS- und eine Android-App.

Als problematisch stellte sich am Anfang das Fehlen eines Macbooks für die Programmierarbeit heraus, daher musste auf andere Möglichkeiten zurückgegriffen werden. Es wurde nach einem Framework gesucht, mit welchem es möglich war, einen Code, ohne ihn verändern zu müssen, auf mehreren Plattformen auszuführen. Dafür eignete sich Ionic perfekt, da man damit eine hybride Cross-Plattform App mit HTML5, CSS, JavaScript, Sass und AngularJS erstellen kann. Der Vorteil davon ist die einfache Nutzung von Webtechnologien anstatt Objective-C, Swift oder Java erlernen bzw. programmieren zu müssen.

In der folgenden Abbildung 4.1 sieht man die Technologien und Abhängigkeiten von Ionic³.

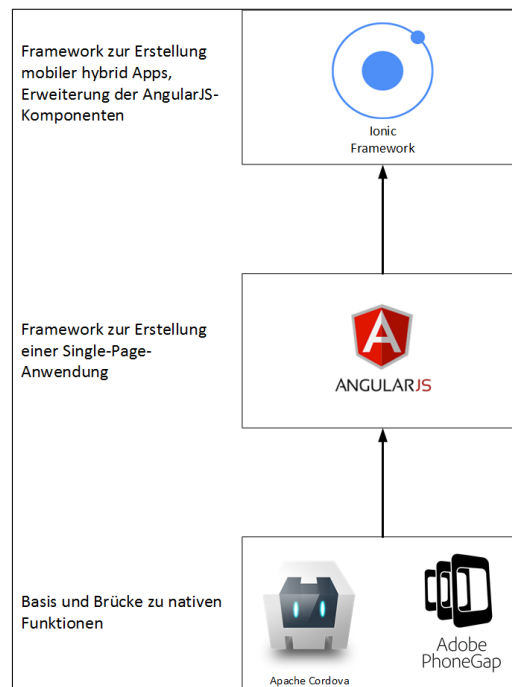


Abbildung 4.1: Übersicht der Technologien und Abhängigkeiten

4.3 Verwendete Entwicklungsumgebung

Als Entwicklungsumgebung für die Programmierung der Ionic App wurde PhpStorm verwendet.

Es wurde die Entscheidung für diese Umgebung getroffen, weil sie diverse verwendete Plugins (auch 'Software Erweiterung' - Software-Modul, welches eine Software mittels verschiedenste Zusatzmodule erweitert⁴) und ebenso die Fehlersuche des Ionic Frameworks unterstützt.

²Vgl. Weiße, *AngularJS Ionic Framework*, S. 6.

³Vgl. Weiße, *AngularJS Ionic Framework*, S. 5.

⁴Vgl. Wikipedia, *Plug-in* — *Wikipedia, Die freie Enzyklopädie*.

4.4 Einrichten von Ionic

Zuerst musste auf dem System 'node' und 'npm' (Node Package Manager) installiert werden. 'Node.js' ist eine Laufzeitumgebung, womit JavaScript auf der Serverseite geschrieben werden kann und dies wird verwendet, um Entwickler-Tools zu bauen⁵. Mit dem Node Package Manager wird dann z.B. Ionic installiert.

Grundsätzlich ist Ionic nicht schwierig zu installieren und einzurichten. Um eine App zu erstellen sind nur 2 Kommandozeilenbefehle notwendig, jedoch wird für die Verwendung des Ionic Frameworks auch eine Cordova Installation vorausgesetzt. Dieses Cordova Framework stellt die Grundfunktionalität zur Verfügung. Mit 'npm-install -g ionic cordova' installiert man Cordova und das Ionic Framework auf dem System und mit 'ionic start <app_name> [template]' wird ein neues Projekt mit dem Namen <app_name> erstellt⁶. Die Angabe des Templates ist optional und gibt an, welches der vordefinierten Templates verwendet werden soll.

Es sind folgende Templates vorhanden: tabs, sidemenu und blank. Das Template 'tabs' erstellt eine App mit Tab-Navigation (Standard), 'sidemenu' eine App mit einem Seitenmenü und 'blank' ein leeres Projekt⁷.

4.4.1 Projektstruktur

Beim Anlegen eines Ionic-Projekts wird auch automatisch ein Ordner mit einer vordefinierten Projektstruktur erzeugt⁸.

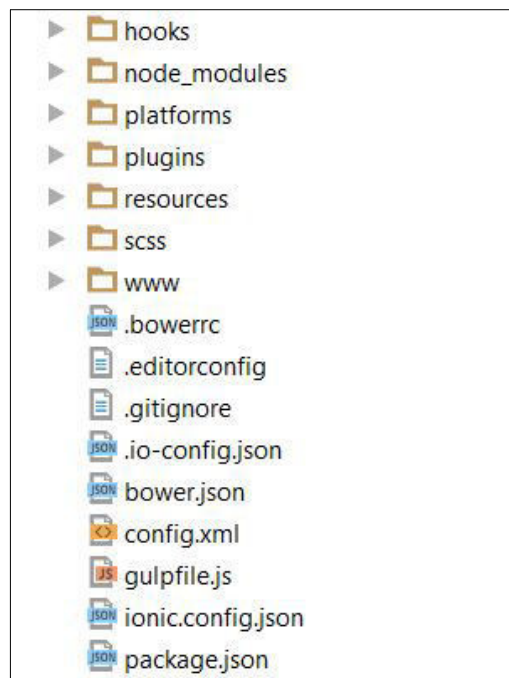


Abbildung 4.2: Projektstruktur eines Ionic-Projekts

⁵Vgl. Nodejs, *Über Node.JS*.

⁶Vgl. Framework, *Installing Ionic and its Dependencies*.

⁷Vgl. *Starting an Ionic App*.

⁸Vgl. Weiße, *AngularJS Ionic Framework*, S. 168-170.

Zuerst werden die einzelnen Ordner der Projektstruktur kurz beschrieben und danach die einzelnen Dateien.

Die hauptsächliche Programmierung erfolgt im `www`-Ordner, in dem sich folgende wichtige Ordner/Dateien befinden:

- **index.html:** Dies ist die Hauptdatei der App und der Inhalt dieser Datei wird standardmäßig beim Starten der App geladen. Alle benötigten Quellen müssen hier eingebunden werden und sie muss `'cordova.js'` laden.
- **css:** Beinhaltet Styling-Dateien zum Design der App.
- **img:** Beinhaltet alle in der App verwendeten Bilder.
- **js:** Beinhaltet JavaScript Dateien und Quellen.
- **lib:** Beinhaltet interne und externe Quellen des Ionic Frameworks.
- **templates:** Beinhaltet Vorlagen für HTML-Dateien.

Der Ordner `'hooks'` beinhaltet Skripte für den Bauprozess der App, in `'plugins'` werden zusätzliche Erweiterungen zur Nutzung nativer Schnittstellen gespeichert und in `'platforms'` werden die Android- und iOS-Projekte erstellt.

Im Letzteren werden dann angepasste Unterordner für die Plattformen angelegt (`'Android'`, `'iOS'`) und im Android Ordner findet man zum Beispiel die APK (Applikation) für eine erstellte Android App.

Der Ordner `'resources'` wird für das Hinzufügen von Ressourcen wie z.B. für das Icon und dem Splash-Screen verwendet.

Der Ionic Kern wird mit Sass gebaut und in dem `'scss'` Ordner befindet sich die Sass-Datei. In der Diplomarbeit wurde jedoch nicht Sass verwendet, sondern das Design wurde mit Hilfe von CSS erledigt.

Im Folgenden sollen nun die einzelnen Dateien der Projektstruktur beschrieben werden.

Die wohl wichtigste Datei ist `'config.xml'`, in der sich die wichtigen Informationen, die zum Ausführen und Erstellen der App benötigt werden, befinden z.B. die Datei, welche beim Starten ausgeführt werden soll.

Die Dateien `'gulp.js'`, `'.bowerrc'`, `'editorconfig'` sind nur verschiedenste, eher unwichtige, Konfigurationsdateien.

In der Datei `'gitignore'` kann man Anweisungen geben, damit Teile der App ignoriert werden sollen.

`'bower.json'` wird verwendet, um Bower Komponenten zu erhalten. Bower ist ein Paketverwaltungstool um Frameworks oder Bibliotheken einfach zu installieren und aktualisieren.

`'package.json'` enthält nur Informationen über die App und Plugins.

Aus der Konfigurationsdatei `'io-config.json'` kann man die `APP_ID`, den API (Anwendungsprogrammierschnittstelle) -Key, den GCM-Key (Google Cloud Messaging Key) und verschiedenste eingegebene Konfigurationsbefehle herauslesen.

Die zweite Konfigurationsdatei `'ionic-config.json'` enthält ebenso die `APP_ID`, zusätzlich aber auch noch den App-Namen⁹.

⁹Vgl. Weiße, *AngularJS Ionic Framework*, S. 170.

4.5 Verwendung / Testen

Bevor mit der Arbeit beim Projekt begonnen werden kann, muss man noch mit dem Befehl `'ionic platform add [Plattform]'` eine Plattform hinzufügen ('iOS', 'Android' oder 'Windows'). Es gibt verschiedene Varianten, um eine Ionic App zu testen bzw. auszuführen:

- **Direkt als Website:** Die am meisten verwendete Variante zum Testen der App. Dies funktioniert mit dem Befehl `'ionic serve'` in der Kommandozeile. Der Vorteil dieser Variante ist, dass man schnell testen kann, ob man Programmierfehler im HTML-, CSS-, AngularJS- oder JavaScript-Code gemacht hat, weil die Konsole zum Debugging zur Verfügung steht. Der Nachteil jedoch ist, dass verschiedenste Funktionen, welche bei dieser Variante funktionieren, am Smartphone nicht unterstützt werden. Diese Art des Testens wurde hauptsächlich während der Entwicklungsphase verwendet, weil man das Design bei dieser Variante am leichtesten testen und Programmierfehler am besten debuggen kann.
- **Direkt am Smartphone:** Eine ziemlich aufwändige Variante zum Testen, es ist jedoch sichergestellt, dass alle Codeteile auch auf den Smartphones funktionieren. Um die App am Smartphone zu testen, muss man zuerst den Befehl `'ionic build [Plattform]'` in der Kommandozeile ausführen, womit eine App entweder nur für die gewählte Plattform oder für alle Plattformen erstellt wird. Auf Android kann diese erstellte APK entweder über die ADB (Android Debug Bridge) oder über ein angestecktes Kabel auf das Smartphone übertragen werden und danach standardmäßig auf dem Smartphone installiert werden. Auf iOS übernimmt das Übertragen und Installieren der App auf dem Smartphone ein von Apple installiertes Programm, xCode. Nach der Installation kann auf dem Smartphone getestet werden. In dieser Diplomarbeit wurde diese Variante in regelmäßigen Abschnitten verwendet. Immer wenn ein Teil fertig programmiert war und bei der Testvariante 'Direkt als Webseite' einwandfrei funktionierte, wurde die App gebaut und am Smartphone getestet. Die Auftraggeberin testete nur mit dieser Variante, weil die App auch im Echtzeitbetrieb auf dem Smartphone betrieben wird und dann alles einwandfrei funktionieren muss.
- **Auf einem Emulator:** Aus Erfahrungswerten dieser Diplomarbeit ist diese Variante nicht vorteilhaft, weil das Testen mit dem Emulator vergleichsweise langsam und dadurch zeitaufwendig ist. Das Starten des Emulators dauert ziemlich lange (ca. eine Minute) und dies macht die oben genannten Möglichkeiten zweckmäßiger. Der einzige Vorteil ergibt sich daraus, dass dadurch das Smartphone direkt auf dem Laptop simuliert werden konnte, was die Nachteile der anderen Varianten ausmerzt. Diese Variante wurde zu Beginn der Programmierung auf einem Android Emulator getestet und wurde zum Testen der Android App nie weiter verwendet. Auf iOS wurde öfters mit dem Emulator getestet, weil der iOS Emulator eine höhere Geschwindigkeit aufweist, wenn man von dem Startvorgang absieht.

4.5.1 Server

Für das Testen wurde ein Testserver von dem Anbieter 'world4you' gemietet. Da dieser gemietete Server bereits vorkonfiguriert war und alle benötigten Pakete installiert waren, musste nicht viel eingerichtet werden. Lediglich die PHP Version musste konfiguriert werden. In der Administrationsoberfläche konnte die zu verwendende PHP Version eingestellt werden und als logische Schlussfolgerung wurde die neueste PHP Version (PHP7) ausgewählt.

Damit die neuesten Dateien nicht jedes Mal zum Testen der neu implementierten Features auf dem Server hochgeladen werden müssen, wurde bei jedem Diplomatsmitglied ein lokaler Testserver eingerichtet. Dafür wurden im Kreise des Diplomarbeitsteams 2 verschiedene Server verwendet:

- **WAMP-Server:** ist die Abkürzung für Windows (Betriebssystem), Apache (Web-Server), MySQL (Datenbank), PHP (Sprache)¹⁰.
Den WAMP-Server verwendete aufgrund bereits vorhandener Erfahrung nur Frau Karner.
- **XAMPP-Server:** XAMPP ist die Abkürzung für Apache (Web-Server), MariaDB (Datenbank), Perl und PHP (Sprachen). Das 'X' steht dafür, dass man es auf verschiedenen Betriebssystemen einsetzen kann¹¹.
Diese Variante nutzten Herr Ganster und Herr Tarnok, weil beide mit diesem Server mehr Erfahrung mitbrachten.

Ein XAMPP Server baut auf einen WAMP Server auf, das heißt zwischen den beiden Servern ist der Unterschied, abgesehen einiger Neuerungen und zusätzlichen Möglichkeiten, die auf dem XAMPP Server geboten werden, gering. Diese weiteren Möglichkeiten wurden in dieser Diplomarbeit aber nicht unbedingt benötigt.

¹⁰Vgl. WampServer, *WAMPSEVER, a Windows web development environment*.

¹¹Vgl. Wikipedia, *XAMPP — Wikipedia, Die freie Enzyklopädie*.

4.6 Funktionen der App

In diesem Kapitel werden die Funktionalitäten der App beschrieben und das Design gezeigt.

In Abbildung 4.3 rechts sind alle Funktionen aufgelistet. Als aktiver Therapeut kann man in der App seine eigenen Daten abrufen und bearbeiten. Die eigenen Arbeitszeiten, Abwesenheiten, Blockaden und der Dienstplan können überprüft werden. Die Vertretungsverwaltung wird später genauer beschrieben.

Die Lupe ist für das Suchen zuständig und durchsucht den ganzen Inhalt auf der aktiven Seite und die Glocke verweist auf die Benachrichtigungsseite. Wenn auf 'Ausloggen' geklickt wird, wird die Session geschlossen und die Benutzerdaten werden nicht mehr zwischengespeichert. Auf der Seite 'About' können zum Beispiel Informationen über das Diplomarbeitsteam gefunden werden.

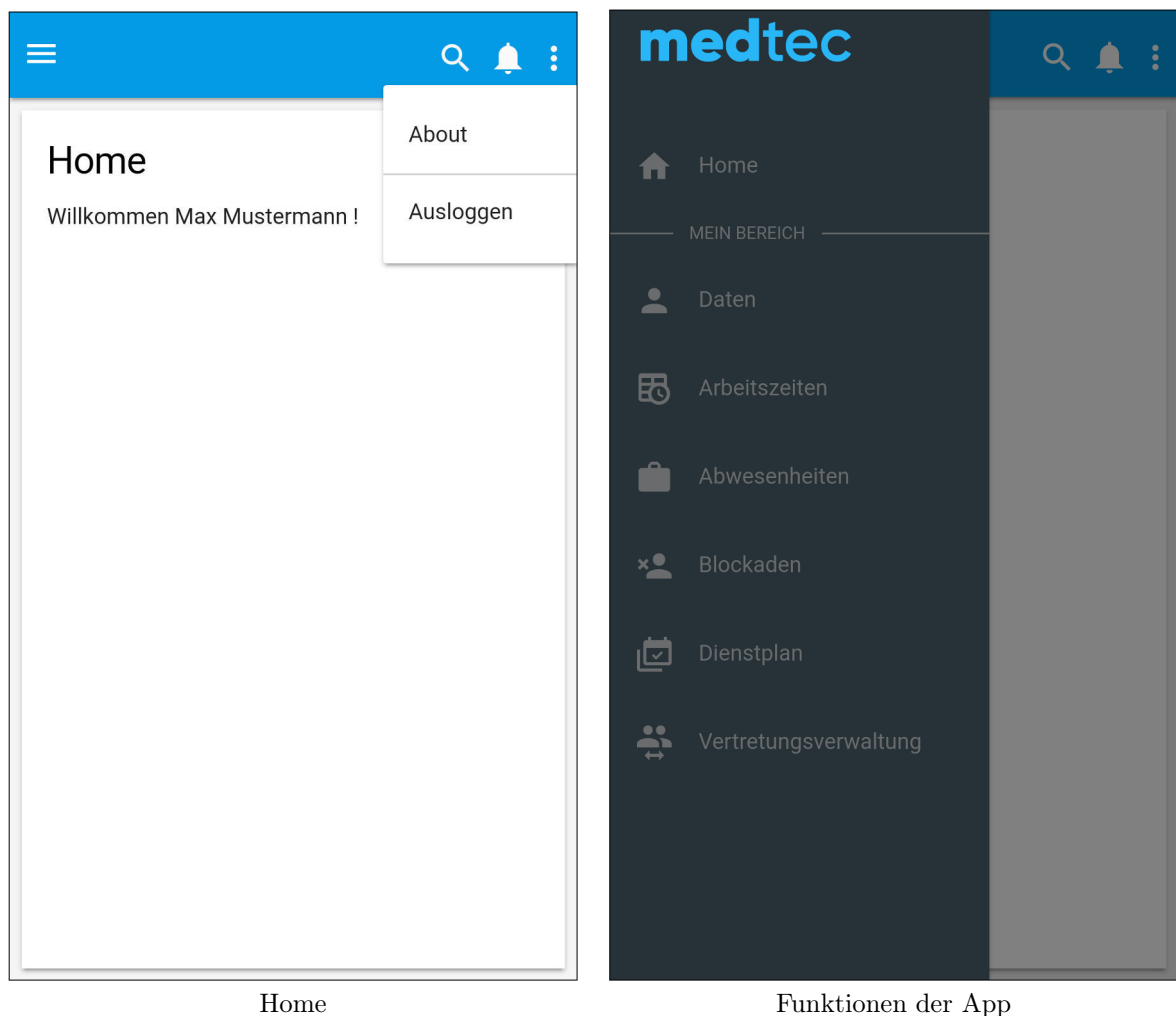


Abbildung 4.3: Links ist zu sehen, wie die App beim Start aussieht und rechts sieht man die Funktionen der App.

4.6.1 Datenanzeige / Datenbearbeitung

Unter dem Tab 'Daten' sieht man, wie in Abbildung 4.4 gezeigt, die eigenen Daten, wie den Nachnamen, Vornamen, Email, den Bereich / die Abteilung, in der man arbeitet und die eigene Telefonnummer.

Klickt man auf das Bearbeiten-Symbol, wird ein Dialog zum Bearbeiten der Daten geöffnet (wird in Abbildung 4.4 rechts gezeigt). Dies wurde mit dem von Ionic bereitgestellten Dialog ('ionicPopup') gelöst.

Wenn Änderungen vorgenommen worden sind und auf Speichern geklickt wurde, werden sie in die Datenbank geschrieben und die Seite mit den Änderungen neu geladen, damit die neuesten Daten angezeigt werden. Wenn man abbricht, gelangt man wieder zur Datenanzeige.

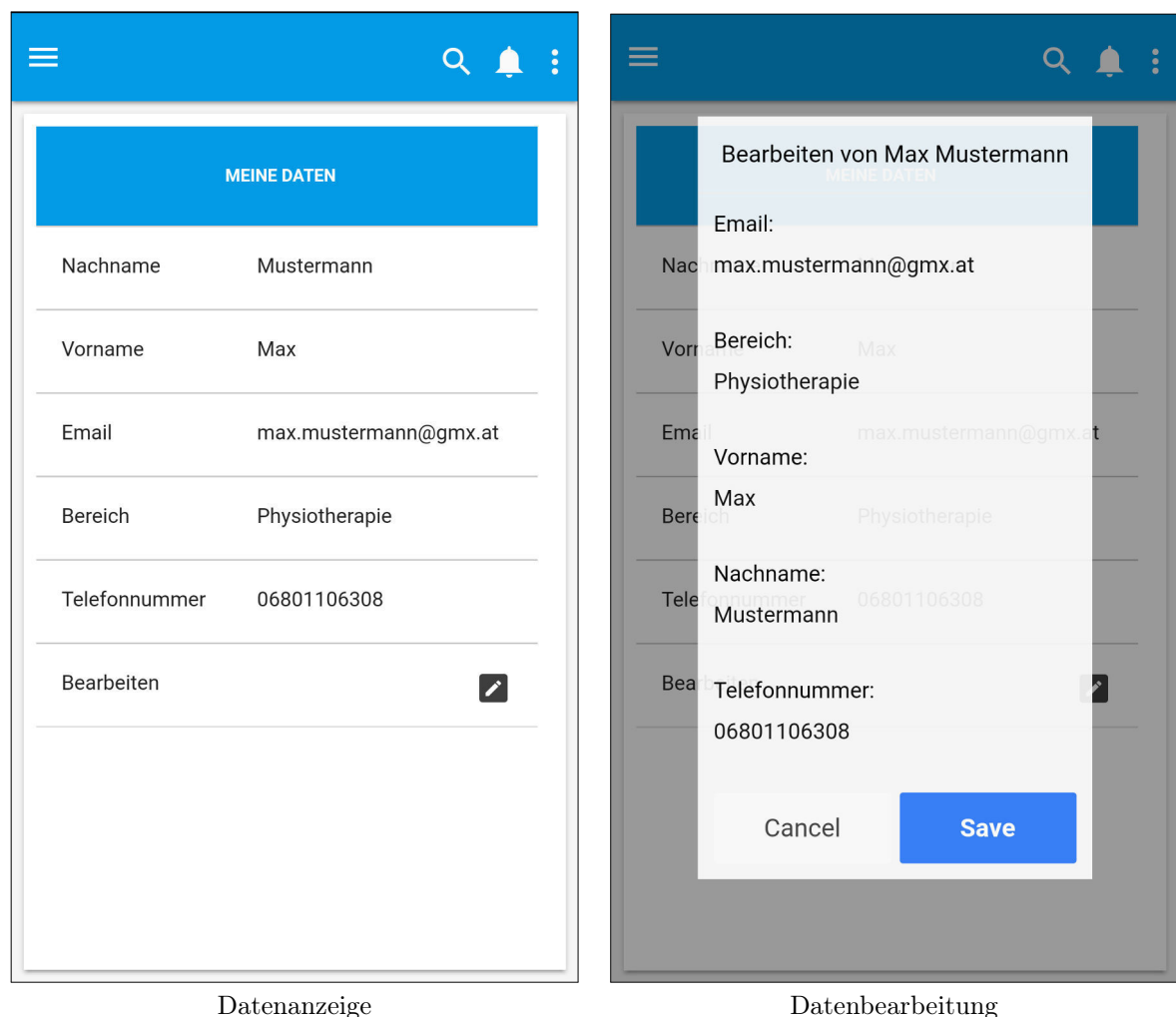


Abbildung 4.4: In dieser Abbildung ist das Design der Datenanzeige und der Datenbearbeitung zu sehen

4.6.2 Arbeitszeiten-, Abwesenheiten- und Blockadenanzeige

Unter den Tabs 'Arbeitszeiten', 'Abwesenheiten' und 'Blockaden' kann der eingeloggte Therapeut nachsehen, von wann bis wann er an welchem Tag im Krankenhaus sein muss und wann er auf Urlaub ist bzw. wann er verhindert ist, in die Arbeit zu kommen, z.B. wenn ein Arzttermin eingeteilt ist.

Bei den Arbeitszeiten gibt es zwei weitere Tabs - aktuelle und zukünftige - weil sie z.B. in zwei Wochen anders sein können als in dieser Woche.

Wenn man auf eine Tabellenzeile klickt, wird die auf der Abbildung 4.5 gut zu sehende dunkelgraue Zeile ausgeklappt. Diese verschafft genauere Informationen über diesen einen Eintrag.

Bei den Abwesenheiten und Blockaden werden in der ausgeklappten Zeile der Grund angezeigt und bei den Arbeitszeiten wird der Zeitraum angezeigt, von wann bis wann diese gelten. ('Akkordeon' - siehe Kapitel 4.7.3)

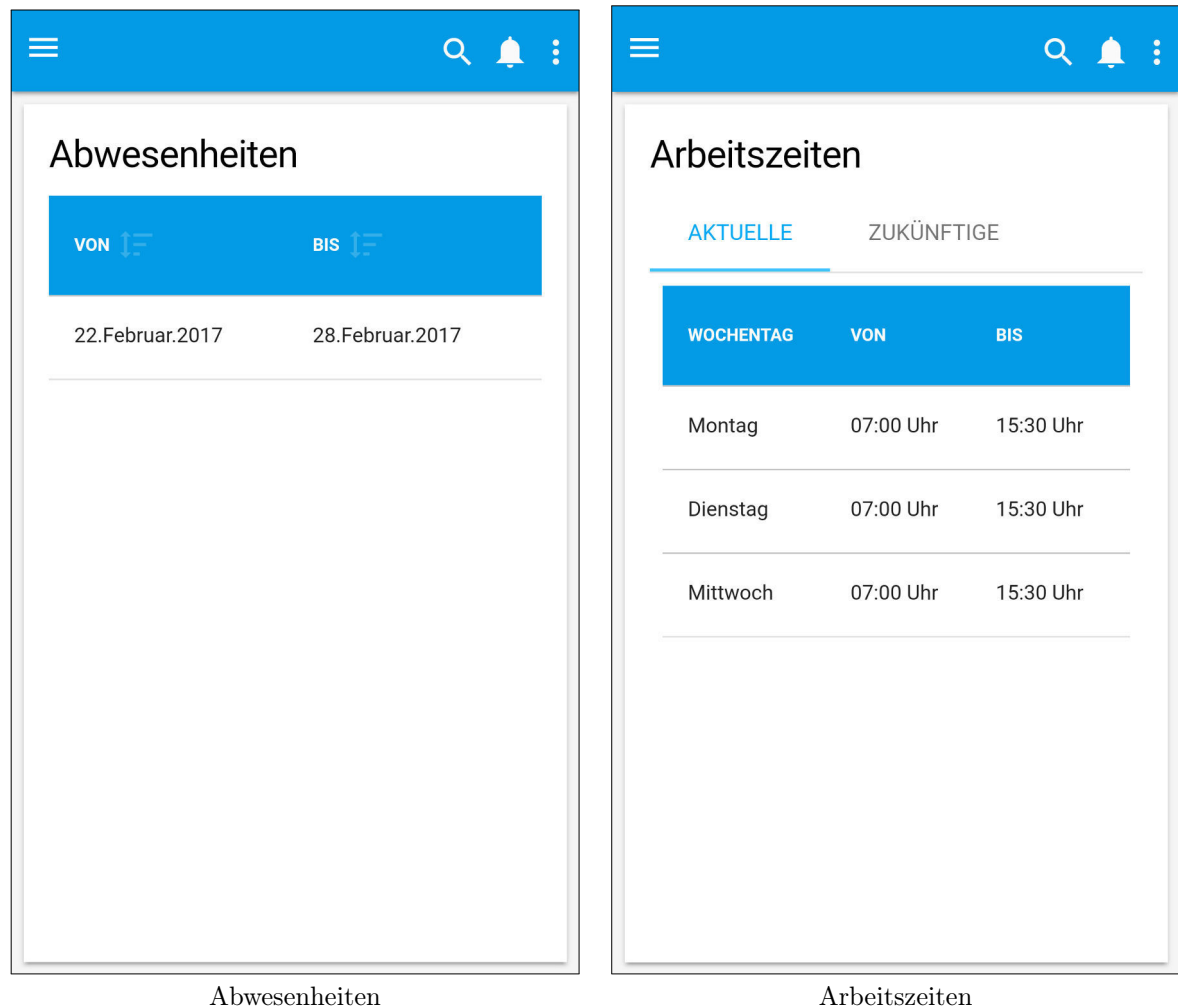


Abbildung 4.5: In dieser Abbildung ist das Design der Abwesenheiten und der Arbeitszeiten zu sehen

4.6.3 Dienstplananzeige

Die Dienstplananzeige ist für die aktuelle Woche konfiguriert und zeigt an, wann an welchem Tag in welcher Gruppe gearbeitet werden soll. Ohne ausgeklappte Zeilen wird nur der Tag und die Gruppe, welche betreut werden muss, angezeigt.

Wenn darauf geklickt wird, werden vier weitere Zeilen ausgeklappt.

Die erste Zeile zeigt an, von wann bis wann diese Gruppe zu betreuen ist. Von den weiteren drei kann abgelesen werden, wer der hauptverantwortliche Therapeut ist (zweite Zeile) und wer die zwei Vertretungen sind (dritte und vierte Zeile).

Wenn der Tag in der zweiten Spalte in den ausgeklappten Zeilen rot hinterlegt ist, heißt das, dass dieser Therapeut an diesem Tag eigentlich nicht Dienst hat bzw. verhindert ist. Wenn die Farbe des Tages grün ist, dann sollte der Therapeut anwesend sein.

TAG	GRUPPE
Fr	Schulter 1 ▼
08:00 Uhr	10:00 Uhr
Mustermann Max	Fr
Haas Gabriele	Fr
Ganster Lukas	Fr
Do	Schulter 1 ▼

Abbildung 4.6: Dienstplananzeige in der App

4.6.4 Vertretungsverwaltung

In der Vertretungsverwaltung werden alle Gruppen angezeigt, in der der eingeloggte Therapeut am heutigen Tag entweder als Hauptverantwortlicher oder als Vertretung eingeteilt ist. Dies wird, wie in Abbildung 4.7 zu sehen ist, durch zwei verschiedene Bilder gekennzeichnet.

Falls derjenige Therapeut wegen einem beliebigen Grund verhindert ist, kann er sich über den Button 'Für den heutigen Tag abmelden' abmelden. Danach muss er noch bestätigen, dass die Vertretung informiert werden soll, und dann wird eine Push-Benachrichtigung an die Vertretungen der betroffenen Gruppen gesendet, dass diese für den heutigen Tag einspringen müssen. Dies wird aber noch genauer im nächsten Abschnitt nach dem Bild, in 'Implementierung des Abmeldeverfahrens', mit einem Code Beispiel erklärt.

Falls am heutigen Tag keine Gruppen vorhanden sind, wird der eingeloggte Therapeut mit der Anzeige von 'Keine passenden Treffer' darüber informiert.

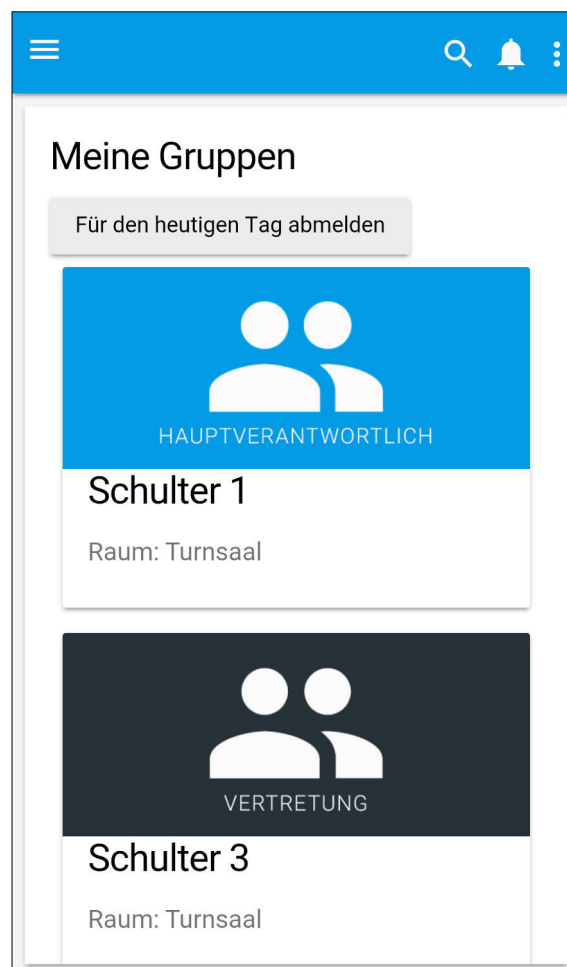


Abbildung 4.7: Vertretungsverwaltungsanzeige in der App

Implementierung des Abmeldeverfahrens

Zuerst wird ein Datensatz mit der eigenen Therapeutennummer und dem heutigen Datum in die 'AbsentToday' Tabelle in der Datenbank geschrieben. Dann werden von der DB alle involvierten Therapeuten der Gruppe geholt und es wird zwischen der Abmeldung des Hauptverantwortlichen und der ersten Vertretung unterschieden.

Falls sich der hauptverantwortliche Therapeut abmeldet, wird die erste Vertretung informiert und wenn sich dann die erste Vertretung auch noch abmeldet, wird dasselbe mit der zweiten Vertretung erledigt.

Wenn aber der erstvertretende Therapeut schon abgemeldet ist, bevor sich der Hauptverantwortliche entschuldigt, wird direkt die zweite Vertretung informiert.

Beim Informieren wird zuerst die Benachrichtigung in die Datenbank geschrieben und danach mit der 'sendPush' Funktion die Push-Benachrichtigung an den richtigen Therapeuten gesendet.

Listing 4.1: Implementierung der Unterscheidung der Abmeldungen

```

1 $senr = null;
2 $token = "";
3 $msg = "Gruppe " . $group["descr"] . " ist heute von dir zu uebernehmen!";
4 if($group["mnr"] = $senr){
5     if(isAbsentTodayController($group["d1nr"]) > 0){
6         $token = getEmployeesTokenController($group["d2nr"]);
7         $senr = $group["d2nr"];
8     }else{
9         $token = getEmployeesTokenController($group["d1nr"]);
10        $senr = $group["d1nr"];
11    }
12    insertNewNotificationController($senr, "Abwesenheit", $msg);
13    sendPush($token, "Abwesenheit", $msg, false);
14 }elseif($group["d1nr"] = $senr){
15     if(isAbsentTodayController($group["mnr"]) > 0){
16         $token = getEmployeesTokenController($group["d2nr"]);
17         $senr = $group["d2nr"];
18         insertNewNotificationController($senr, "Abwesenheit", $msg);
19         sendPush($token, "Abwesenheit", $msg, false);
20     }
21 }
```

4.7 Webapplikation zu Ionic-App

Die Ionic App wurde zeitgleich mit der dazugehörigen Webapplikation programmiert. Da die Webapplikation mit AngularJS aufgebaut ist und Seiten der Webapplikation auch in der App verfügbar sind, konnten diese teilweise übernommen werden.

Natürlich gab es einige Probleme und die Anzeige musste auch verändert werden, aber der Sinn dahinter, ist derselbe. In den nächsten Kapiteln werden die Kommunikation mit dem Backend, Probleme bei der Übernahme des Codes und Vor- und Nachteile beschrieben.

4.7.1 Kommunikation mit der API

Worüber zu Beginn der Diplomarbeit viel nachgedacht wurde, ist die Kommunikation zwischen App und Web-Applikation.

Zuerst wurde bestimmt, dass die Kommunikation direkt am Server geschrieben wird, damit die Synchronisation der Daten zwischen App und Web-Applikation korrekt ist.

Im Laufe der Programmierung von der App wurde hinsichtlich der Art der Kommunikation umentschieden und die App und die Web-Applikation schicken direkte Requests zum Server (AjaxController.php - siehe 4.2).

Für diese Implementierung wurde für jede HTML-Seite, welche Daten vom Server benötigt, ein eigener Controller geschrieben. Dieser sendet POST- bzw. GET-Requests an den AjaxController.

Listing 4.2: Der GET-Request für die Abwesenheiten an den AjaxController

```

1 function AbwesenheitenCtrl($scope, $http, $log) {
2     $http.get("http://" + ipaddress + "/api/AjaxController.php", {
3         params: {
4             req: 'getFutureAbsencesOwn',
5             fromApp: true,
6             email: localStorage.getItem("email"),
7             enr: localStorage.getItem("enr")
8         }
9     }).then(function (response) {
10         $scope.absences = response.data;
11     });
12 }
```

Der Codeteil 4.2 zeigt den verwendeten GET-Request, um Daten für die Abwesenheit eines Therapeuten zu bekommen.

Der Befehl 'req: getFutureAbsencesOwn' ruft auf dem Server den AjaxController mit der angegebenen Funktion auf, welche dann die benötigten Daten zurückliefert.

Wieso 'fromApp: true' als Parameter mitgegeben werden muss, wird in Kapitel 4.7.4 genauer beschrieben.

Die E-Mail und enr (Therapeutennummer) des eingeloggten Therapeuten muss zur Identifikation mitgegeben werden.

4.7.2 Vergleich localStorage zu Cookies

Beide Varianten bieten die Möglichkeit zur clientseitigen Datenspeicherung, jedoch mit ein paar bedeutenden Unterschieden.

Bei localStorage werden die Daten lokal abgespeichert und nicht wie bei den Cookies immer mitgeschickt.

Ein weiterer Unterschied ist die Speicherkapazität, bei Cookies können nur 80 Kilobyte gespeichert werden, wobei bei localStorage eine Speicherung mehrerer Megabytes möglich ist. Es wurde trotzdem für die Verwendung von localStorage entschieden, da man auf den Key von überall zugreifen kann und diese Variante für eine App praktischer ist.

Es gibt 2 unterschiedliche Varianten zur Storage-Speicherung:

- **localStorage**
- **sessionStorage**

Der Unterschied ist, dass bei sessionStorage die gespeicherten Daten beim Schließen der App/des Browsers verloren gehen.

Die Verwendung von localStorage ist sehr einfach und man kann diese mit vier einfachen Befehlen kontrollieren. Mit 'localStorage.setItem('key', 'value')' wird ein Key mit einem

definierten Namen gesetzt, welcher die definierten Informationen enthält.

'localStorage.getItem('key')' gibt für diesen Key die gespeicherte Information zurück und wenn dieser nicht existiert, wird 'undefined' zurückgegeben.

Einen gespeicherten Wert kann man mit 'localStorage.removeItem('key')' wieder löschen und der Speicher wird freigegeben.

Um den ganzen localStorage Speicher zu löschen (wird z.B. beim Abmelden eines Therapeuten benötigt) gibt es den Befehl 'localStorage.clear()' ¹².

In dieser Diplomarbeit wurde localStorage zur Speicherung der Anmeldedaten (Therapeutennummer und Email) und der Tokeninformationen verwendet.

4.7.3 UI-Elemente

Akkordeon

Ein mit Javascript gelöstes Akkordeon ist eine gute und einfache Lösung für die Anzeige einer Tabelle in einer App.

Die wichtigsten Daten werden angezeigt und durch einen Klick auf eine Zeile bekommt man nähere Informationen für diesen einen Datensatz.

Es gibt zwei Varianten für die Implementierung eines Akkordeons:

- **Es kann immer nur eine Zeile ausgeklappt werden:** Diese Variante wurde in dieser Diplomarbeit bei den Tabellen verwendet, weil die Anzeige übersichtlich bleibt. Wenn man auf eine Zeile klickt, werden die dazugehörigen Informationen ausgeklappt. Wenn wieder auf dieselbe Zeile geklickt wird, wird sie eingeklappt und es werden wieder nur die wichtigsten Daten angezeigt. Falls schon eine Box ausgeklappt ist und es wird auf eine andere Zeile geklickt, wird die schon ausgeklappte Box eingeklappt und die andere ausgeklappt.
- **Es können alle Zeilen ausgeklappt werden:** Bei dieser Variante könnte die Ansicht der Tabelle sehr unübersichtlich werden. Das Prinzip ist dasselbe wie bei der vorigen Variante, nur wird bei einem zweiten Klick auf eine andere Zeile, die vorherige nicht eingeklappt. Somit können alle Zeilen auch ausgeklappt sein, was die Tabelle sehr lang machen könnte.

Für die Implementierung wird in der Tabellendefinition bestimmt, auf welche Zeilen geklickt werden darf und auf welche nicht.

Dies wird durch das HTML-Attribut 'data-clickable' definiert. Die Box mit dem Inhalt, welcher ausgeklappt wird, bekommt das HTML-Attribut 'data-expandable' zugewiesen.

Für das Aufrufen des dazugehörigen Javascript-Codes wird eine sogenannte Direktive von AngularJS verwendet, weil gewartet werden muss, bis die ganze Tabelle fertig aufgebaut ist. Dies kann vom abgebildeten Codeteil 4.3 abgelesen werden.

¹²Vgl. *HTML5 localStorage*.

Listing 4.3: Der Body-Tabellenaufbau für die Abwesenheiten eines Therapeuten

```

1 <tbody>
2   <tr data-clickable ng-repeat-start="x in absences = (absences | filter:search)" my-
     repeat-directive>
3     <td>{{ x.startdate | dateFilter : 'dd.LLLL.yyyy'}} </td>
4     <td>{{x.enddate | dateFilter : 'dd.LLLL.yyyy'}}</td>
5   </tr>
6   <tr ng-repeat-end data-expandable style="display:none; background-color:lightgrey;">
7     <td colspan="2">{{x.reason}}</td>
8   </tr>
9 </tbody>

```

Der dazugehörige in 4.4 abgebildete JavaScript-Code ist nicht schwierig und auch nicht lang.

Er beginnt mit einem Klick auf ein 'data-clickable' Element (warum '.unbind('click')' benötigt wird, wird im Kapitel 4.7.4 beschrieben).

In der Klick-Funktion werden zuerst alle 'data-expandable' Zeilen (zusätzliche Inhalte) eingeklappt, welche nicht meine eigenen zusätzlichen Inhalte sind. Danach wird mein 'data-expandable' Objekt ausgeklappt. Somit werden zwei Bedingungen sichergestellt:

1. Dass immer nur von einer Zeile die zusätzlichen Inhalte angezeigt werden können und
2. dass auch durch einen Klick auf eine schon ausgeklappte Zeile, diese eingeklappt wird

Listing 4.4: Der JavaScript Code für die Logik des Akkordeons

```

1 ctrl.directive('myRepeatDirective', function () {
2   return function (scope, element, attrs) {
3     $(' [data-clickable] ').unbind('click').click(function () {
4       $(" [data-expandable] ").not($(this).next(" [data-expandable] ")).slideUp();
5       $(this).next(" [data-expandable] ").slideToggle("slow");
6     });
7   };
8 });

```

4.7.4 Probleme und Lösungen

Im Laufe der Programmierung der App traten einige Probleme auf und in diesem Kapitel werden diese mit den Lösungen aufgelistet und beschrieben.

Warum wurde zuerst der Zugriff auf den Server verweigert?

Weil im AjaxController zuerst Zugriffe von außen erlaubt werden müssen, ansonsten kann nur innerhalb der gleichen Domäne zugegriffen werden.

Listing 4.5: Zugriffe von außen erlauben

```

1 header('Access-Control-Allow-Origin: *');

```

Warum muss bei jedem Request an das Backend 'fromApp: true' mitgegeben werden?

Weil bei Cross Site Requests kein Session-Speicher in PHP vorhanden ist. Deswegen muss im AjaxController jede App-Variable neu zugewiesen werden.

Listing 4.6: Zuweisung der Email Variable im AjaxController

```

1 if(array_key_exists('email', $_GET)){
2     $_SESSION["user"] = $_GET["email"];
3 }

```

Warum muss beim Akkordeon dieses `.unbind('click')` verwendet werden?

Weil beim Event zwei Klicks ausgelöst wurden und der Befehl löscht das vorige ausgelöste Click und ersetzt es durch ein neues.

Warum wurde die Navigationsbar nicht eingefahren, wenn man auf ein Element geklickt hat?

Weil sich bei einem Klick auf ein Element die Klassen nicht ändern. Somit wird bei jedem Klick auf ein Navigationselement folgende Funktion aufgerufen:

Listing 4.7: hideDrawer Funktion

```

1 function hideDrawer(){
2     $(".mdl-layout__drawer").toggleClass("is-visible");
3     $(".mdl-layout__obfuscator").toggleClass("is-visible");
4 }

```

Die beiden Befehle setzen die Eigenschaft 'is-visible' immer auf falsch, somit wird sie bei jedem Klick auf ein Element wieder eingeklappt.

Warum wurden die Benachrichtigungen zuerst mit `PushNotification` gelöst, aber danach doch mit `cordovapushv5` umgesetzt?

Weil `PushNotification` als veraltet markiert worden ist und deswegen der Push-Service keinen Token mehr zurückgeliefert hat. Nach langem Suchen wurde `cordovapushv5` verwendet, wofür noch zusätzlich `ng-cordova` benötigt wurde.

Warum wurde bei den Push Benachrichtigungen nicht `onnotification` verwendet sondern die Datenbank-Lösung verwendet?

Weil es zuerst nicht funktioniert hat und der darin enthaltene Code nicht ausgeführt wurde. Danach wurde zwischen der Datenbank-Lösung oder der Fehlersuche entschieden. Es wurde für die Datenbank-Lösung abgestimmt, da jetzt die Benachrichtigungen auch auf iPhones angezeigt werden können und dies wäre mit `onnotification` nicht möglich gewesen, da kein Apple Entwickler Account existierte (siehe 4.8.3).

Warum war ein Teil der App-Ansicht auf iOS über der Statusbar?

Weil dafür das 'org.apache.cordova.device' Plugin installiert und folgender Code in die Datei 'app.js' eingefügt werden musste:

Listing 4.8: Benötigter Code für das Verhindern der Überlappung der Statusbar auf iPhones

```

1 $ionicPlatform.ready(function(){
2     StatusBar.overlaysWebView(false);
3     StatusBar.styleDefault();
4     ....
5 });

```

4.7.5 Vor- und Nachteile dieser Umsetzung

In den Seiten zuvor wurden nebenbei immer wieder Vor- und Nachteile erwähnt. In diesem Kapitel werden diese zusammengefasst.

Der wesentlichste Vorteil ist, dass eine Ionic-App plattformübergreifend ist und dadurch wird der Entwicklungsaufwand reduziert.

Zur Programmierung wird AngularJS verwendet, dies nutzt das Model-View-View-Model Entwurfsmuster. Das heißt, dass Änderungen immer auf beiden Seiten (View und Model) synchronisiert werden, was für die Funktionalität die App sehr wichtig ist.

Ein weiterer Vorteil ist, dass Ionic auf AngularJS basiert und auch HTML, JavaScript und CSS zum Einsatz gebracht werden können. Deswegen mussten keine Technologien von Grund auf neu erlernt werden.

Ionic bringt auch bedienbare Komponenten mit, welche sehr benutzerfreundlich und leicht zu verwenden sind.

Natürlich hat Ionic nicht nur Vorteile, sondern auch Nachteile, zum Beispiel treten bei komplexen Anwendungen Probleme bei der Performance auf. Es kann auch passieren, dass plattformspezifische Anpassungen notwendig sind. Teilweise werden Code- oder Design-Änderungen von einem Betriebssystem zum anderen nicht übernommen und diese müssen dann extra angepasst werden.¹³.

Fazit

Schlussendlich wurde Ionic verwendet, da in kurzer Zeit eine App mit schönem Design entwickelt werden kann.

Da schon etwas Erfahrung mit AngularJS mitgebracht wurde, war der Einstieg ziemlich unkompliziert und es konnte von den vielen durch Ionic bereitgestellten Komponenten profitiert werden.

4.8 Push Benachrichtungen

4.8.1 Allgemeines zu Push Benachrichtungen / Alternativen

Push Benachrichtigungen sind Meldungen bzw. Benachrichtigungen, welche ein Smartphone empfangen kann.

Sie werden angezeigt, ohne dass die jeweilige App, welche die Benachrichtigungen empfängt, geöffnet ist. Die Meldungen werden gezielt von einem Sender (in dieser Diplomarbeit von dem Server, welcher beim Kapitel 4.5.1 Server erklärt wurde) an einen Client (in dieser Diplomarbeit an die App) gesendet.

Alternativen zu Push Benachrichtigungen wären:

- **Polling:** beim Polling sendet ein Client, also das Smartphone, in regelmäßig festgelegten Abständen eine Anfrage an den Server.
Der Server antwortet entweder mit einer kurzen Nachricht oder er sendet eine leere Antwort zurück¹⁴.

¹³Vgl. *Hybrid Mobile App Entwicklung mit Ionic*.

¹⁴Vgl. *Skalierbares HTTP Long Polling*.

- **Long Polling:** beim Long Polling sendet ein Client auch eine Anfrage an den Kommunikationspartner und falls eine Nachricht vorhanden ist, schickt der Server diese Nachricht zurück.
Falls jedoch keine vorliegt, wartet der Server und lässt die Verbindung entweder so lange offen bis eine Antwort vorliegt oder so lange bis eine gewisse festgelegte Zeitspanne vorüber ist. Sobald der Client eine Antwort erhält, sendet er eine weitere Anfrage an den Server.
Diese Variante reduziert im Gegensatz zu Polling den Netzwerkverkehr und hat insgesamt eine bessere Performance, weil nicht ständig Anfragen an den Server geschickt werden müssen¹⁵.
- **Benachrichtigung über E-Mails:** Die Benachrichtigung über E-Mail ist der meistgenutzte Dienst des Internets.
Diese Variante eignete sich deswegen nicht für die Aufgabenstellung, weil man sich nicht sicher sein kann, ob die empfangene E-Mail vom Client wirklich gelesen worden ist. Die meisten E-Mail Programme nutzen ebenso Push Benachrichtigungen, aber bei solchen Programmen/Apps kann man diese Option auch ausstellen.
Bei der App gibt es diese Option nicht und die Meldung kommt sicher auch auf einem gesperrten Smartphone an.

In dieser Diplomarbeit sind Push Benachrichtigungen auch effektiver wie Polling oder Long Polling, weil der Client wirklich nur benachrichtigt wird, wenn eine Nachricht für ihn vorliegt und er selbst nicht immer 'nachfragen' muss.

4.8.2 Anwendung

Push Benachrichtigungen generell werden in dieser Diplomarbeit benötigt, damit Vertretungen informiert werden können, falls sich der hauptverantwortliche Therapeut oder seine Vertretung für den aktuellen Tag abmeldet.

Dafür sendet die App einen Abmelde-Request an das Backend, wo dieser Fall behandelt wird, und die erforderlichen Benachrichtigungen an die betroffenen Personen.

4.8.3 Implementierung

Für diese Push Benachrichtigungen wurde der von ionic.io bereitgestellte Service genutzt.

Zusätzlich wurden zwei Plugins benötigt und genutzt. Das erste heißt 'ionic-platform-web-client' und sorgt für die Verbindung zu ionic.io und das zweite auf Cordova basierte Plugin 'phonegap-plugin-push' sorgt für das Empfangen der Push Benachrichtigungen innerhalb der App¹⁶.

Die Push-Benachrichtigungen wurden jedoch nur auf dem Android Betriebssystem implementiert, weil für den Apple Entwickler Account zu viele Kosten anfielen. Auf iOS-Systemen werden die Benachrichtigungen demnach nur in der App angezeigt.

Android Frontend

In diesem Punkt wird die Implementierung der Push Benachrichtigungen auf Android kurz erklärt. Technische Anforderungen für diese Implementierung sind¹⁷:

¹⁵Vgl. *Skalierbares HTTP Long Polling*.

¹⁶Vgl. Simon, *The Complete Ionic Push Notifications Guide*.

¹⁷Vgl. Simon, *The Complete Ionic Push Notifications Guide*.

- Security Profile
- API Key
- Android Keystore Datei
- Google API Projekt mit einer GCM (Google Cloud Messaging) Projekt Nummer
- Firebase Account

Das Security Profile und der API Key werden benötigt, damit diese von Ionic.io bereitgestellten Dienste auch von außen kontaktiert werden können. Zum Erstellen dieser wird ein Ionic.io Account benötigt. Danach kann man diese auf dem Account unter Einstellungen - Certificates und unter Einstellungen - API Keys anlegen.

Es wird auch eine Android Keystore Datei benötigt, um die App zu signieren. Diese Datei kann durch ein Keytool generiert werden, welches im Java Development Kit inkludiert ist. Diese wird dann in den Ionic.io App Einstellungen eingetragen¹⁸.

In den Einstellungen wird auch nach einem GCM API Key gefragt und dieser wird durch das Erstellen des Google API Projektes angelegt. Als letztes wird noch ein Firebase Projekt benötigt, welcher eine Google FCM (Firebase Cloud Messaging) Sender ID und Sender Key erstellt, um Push Benachrichtigungen an Android Geräte zu schicken.

Um das Plugin „phonegap-plugin-push“ für die App zu aktivieren benötigt man noch 2 Befehle¹⁹:

- `ionic plugin add phonegap-plugin-push --variable SENDER_ID='GCM_PROJECT_NUMBER'`
- `ionic config set gcm_key <gcm-project-number>`

Zusätzlich gehört noch 'ngCordova' in dem Projekt lib-Ordner hinzugefügt. Im Codeteil 4.9 wird die Registration beim Push-Service gezeigt. Zuerst wird 'cordovaPushv5' initialisiert und wenn dies erfolgreich ist, wird man beim Push-Service registriert²⁰. Beim 'register(resultreg)' wird eine eigene geschriebene Funktion aufgerufen, welche den vom Push-Service generierten lokalen Token in die Datenbank schreibt.

Listing 4.9: Registration beim Push-Service

```

1 $cordovaPushV5.initialize(
2 {
3   android: {
4     senderID: "xxxxxxxxxxxx",
5     clearNotifications: false
6   }
7 }).then(function (result) {
8   $cordovaPushV5.register().then(function (resultreg) {
9     register(resultreg);
10  }, function (err) {
11    console.log("medtec_push register error: ", JSON.stringify(err));
12  });
13 });
```

Beim Versenden einer Benachrichtigung wird die Nachricht in die Datenbank mit einem Status (geöffnet oder nicht geöffnet) geschrieben. Durch einen Request bekommt die App die Information, wieviel neue und welche Benachrichtigungen der eingeloggte Therapeut

¹⁸Vgl. *Android App Keystore*.

¹⁹Vgl. Simon, *The Complete Ionic Push Notifications Guide*.

²⁰Vgl. ngcordova, *cordovaPushV5*.

erhalten hat. Wenn danach auf die Benachrichtigungsseite geklickt wird, wird der Status aller Nachrichten auf 'geöffnet' geschaltet.

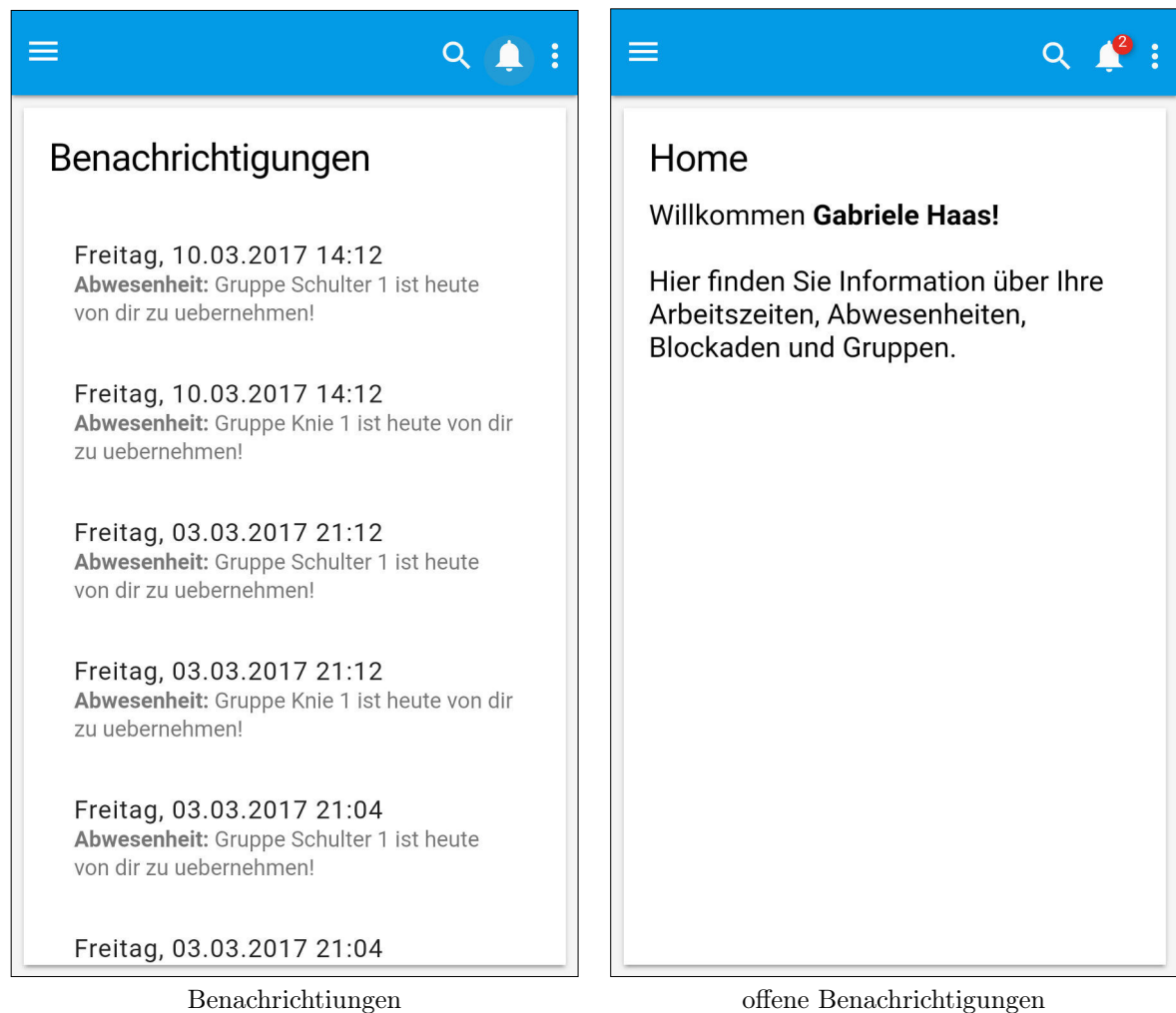


Abbildung 4.8: Links sieht man die Benachrichtigungsübersicht und rechts wie die offenen Benachrichtigungen angezeigt werden

Senden der Push-Benachrichtigung vom Backend

Wenn man sich bei der App anmeldet, wird für diese Person auch die in 4.8.3 erklärte 'register' Funktion aufgerufen und der generierte Token in die Datenbank geschrieben. Dieser Token wird wieder von der Datenbank gelöscht, wenn sich die Person von der App abmeldet. An diesen Token wird die Benachrichtigung vom PHP-Backend gesendet, welches jetzt genauer erklärt wird.

Listing 4.10: Benötigte Variablen zum Senden einer Push-Benachrichtigung

```
1 function sendPush($devicetoken, $title, $message, $sendtoall){
2   $url = 'https://api.ionic.io/push/notifications';
3   $profile = "dev";
4
5   if($sendtoall)
6   {
7     $data = array(
8       'send_to_all' => true,
9       "profile" => $profile,
10      'notification' => array('title' => $title, 'message' => $message)
11    );
12  }
13  else
14  {
15    $data = array(
16      'tokens' => array($devicetoken),
17      "profile" => $profile,
18      'notification' => array('title' => $title, 'message' => $message)
19    );
20  }
21  $content = json_encode($data);
```

Der Codeteil in der Abbildung 4.10 zeigt einen Teil der Funktion, welche die Push Benachrichtigung an die betroffene Person sendet. Eine Benachrichtigung wird mit 'cURL' (Curl URL Request Library bzw. Client for URLs) vom Backend gesendet, welches später noch genauer erklärt wird.

Der Funktion werden vier Parameter übergeben, wobei die wichtigsten zwei 'sendtoall' und 'devicetoken' sind.

Im Code ist zu sehen, dass bei 'sendtoall' zwischen true und false unterschieden wird. Wird bei diesem 'true' mitgegeben, wird der Token ignoriert und es wird die Benachrichtigung an jedem beim Push-Service registrierten Benutzer gesendet. Falls 'false' mitgegeben wird, wird der Inhalt des mitgegebenen Parameters 'devicetoken' (welches ein Array ist) verwendet.

Die anderen zwei Parameter 'title' und 'message' sind nur die Informationen für das eigentliche Benachrichtigungsobjekt, welche an die Personen gesendet werden sollen.

Damit libcurl, eine Bibliothek von cURL, mit Daten-Arrays umgehen kann, muss es noch in JSON-Format mit 'json_encode' konvertiert werden.

Es werden zusätzlich zu den Parametern zwei extra Variablen benötigt, nämlich url und profile. 'Profile' ist der Name des oben erstellten Security Profile, welches die Push-Anmeldeinformationen für die Benachrichtigungen enthält. Die 'url' muss angegeben werden, damit cURL später die Information hat, an welche Adresse ein POST-Request gemacht werden soll.

Listing 4.11: Eigentliches Senden der Benachrichtigung mit cURL

```

1 $ch = curl_init();
2 curl_setopt($ch, CURLOPT_URL, $url);
3 curl_setopt($ch, CURLOPT_POST, TRUE);
4 curl_setopt($ch, CURLOPT_POSTFIELDS, $content);
5 curl_setopt($ch, CURLOPT_HTTPHEADER, array(
6     'Content-Type: application/json',
7     'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9....'
8 ));
9 $result = curl_exec($ch);
10 echo $result;
11 curl_close($ch);

```

cURL ist ein Kommandozeilenprogramm zum Übertragen von Daten in einem Netz, ohne Interaktionen eines Benutzers. Die Bibliothek libcurl wird von PHP unterstützt und verwendet und deswegen können Benachrichtigungen problemlos über cURL an den Client gesendet werden²¹

Mit 'curl_init' wird ein neues cURL Objekt angelegt, mit dessen die Benachrichtigung gesendet wird.

Danach werden die benötigten Optionen gesetzt. Die ersten zwei Optionen bedeuten, dass cURL einen POST-Request an die URL sendet, deswegen wird 'CURLOPT_POST' auf true gesetzt. Danach wird noch der Inhalt (Titel und Nachricht) hinzugefügt.

Die letzte zu setzende Option ist der sogenannte 'Header'. Dieser besteht aus einem Array aus zwei Eigenschaften, nämlich dem Content-Type und der Authorization.

Der 'Content-Type' gibt an, in welchem Typ/Format der Inhalt abgespeichert ist. Die 'Authorization' muss wie folgt angegeben werden: 'Authorization: Bearer <dein-authentication-token>'²².

Ein Bearer-Token ist ein Sicherheitstoken und ist ein Teil von dem OAuth V2 Standard, welches ein Protokoll für eine sichere Autorisierung für Anwendungen erlaubt²³. Den eigenen 'Authentication-Token' findet man in den Einstellungen unter API Keys auf ionic.io und dieser besteht aus 143 Zeichen.

Mit 'curl_exec(<Objekt>)' wird die Benachrichtigung versendet und sollte nach dem Befehl beim Client angekommen sein und danach wird das erstellte Objekt wieder geschlossen²⁴.

²¹Vgl. curl, *curl - command line tool and library for transferring data with URLs*.

²²Vgl. *CURL*.

²³Vgl. *The OAuth 2.0 Authorization Framework: Bearer Token Usage*.

²⁴Vgl. *CURL*.

Kapitel 5

Datenbank (PT)

5.1 Einleitung

Dieses Kapitel befasst sich mit den Überlegungen und der Umsetzung, der Datenbank. Die Datenbank speichert alle relevanten Daten, die in der Webapplikation verwendet werden. Zusätzlich ist diese möglichst performant und sicher gestaltet, um mit geringsten Erhaltungskosten den maximalen Nutzen aus der Datenbank zu holen.

5.2 Evaluierung von MySQL

Bei dem relationalen Datenbankmanagementsystem (DBMS) MySQL handelt es sich nicht um irgendein Datenbanksystem, sondern um die populärste Open-Source-Datenbank der Welt¹. Das Datenbanksystem ist seit 1995 verfügbar und ist heute in fünf verschiedenen Versionen erhältlich:

- MySQL Standard Edition
 - kostenpflichtige Version mit Oracle Premier Support
- MySQL Community Server
 - Frei erhältliche Open-Source Version der Standard Edition
- MySQL Enterprise Edition
 - kostenpflichtige Version mit Oracle Premier Support und Enterprise Manager
- MySQL Cluster Carrier Grade Edition(CGE)
 - kostenpflichtige Version mit Oracle Premier Support und Enterprise Manager
- MySQL Cluster
 - Frei erhältliche Open-Source Version der CGE

Die Firma Oracle² übernahm MySQL im Jahr 2010³ und entwickelt das Datenbanksystem seither weiter. Seit dieser Übernahme steht das System immer häufiger unter Kritik, da der Unterschied zwischen den freien und kommerziellen Versionen von MySQL immer gravierender wird.

¹Vgl. Pröll, Zangerle und Gassler, *MySQL - Das umfassende Handbuch*, S. 40.

²Vgl. <https://www.oracle.com/index.html>, besucht am 03.04.2017

³Vgl. *Oracle Completes Acquisition of Sun*.

Der Open-Source MySQL Community Server ist frei erhältlich und bietet alle benötigten Funktionen für die meisten Anwendungsfälle und ist jener der für die medtec-Webanwendung, auf Version 5.7, verwendet wird. Das DBMS wird dabei im „Data Tier“ der 3-Tier-Architektur (siehe Abb. 3.1) der Webanwendung verwendet.

Ein Hauptvorteil von MySQL ist die äußerst einfache Bedienung des Systems, außerdem findet man sehr einfach Anleitungen und Hilfestellungen zu allen Problemen, da das DBMS Open-Source ist. MySQL gilt als hochleistungsfähiges, hochverfügbares und sicheres System, das selbst auf älterer Hardware stabil läuft und erfüllt durch die kostenlose Anschaffung die Anforderung, die Webanwendung möglichst günstig in der Erstellung und Erhaltung zu ermöglichen⁴. Durch die vielen unterschiedlich verfügbaren „Storage Engines“ ist die Datenbank weiter anpassbar.

Ein großer Nachteil des DBMS ist jedoch die fehlende „check“-Funktionalität, welche den Benutzer dazu zwingt Trigger zur Validierung, beim Einfügen oder Ändern von Datensätzen, zu verwenden.

5.3 Aufbau der Datenbank

5.3.1 Ausgangslage

Bisher wurden in der Abteilung Microsoft Excel Dokumente zur Verwaltung aller Therapeuten, deren Aufgaben bzw. Zuständigkeiten und Abwesenheiten verwendet.

Die Erstellung eines neuen Dienstplanes entsprach mehreren Arbeitstagen Aufwand, da auch die persönlichen Wünsche, welche Gruppe ein Therapeut übernehmen möchte, miteinbezogen wurden.

Leider konnten aus keinem dieser Dokumente brauchbare Daten gewonnen werden. Dies führte dazu, dass die Datenbank von Grund auf neu entwickelt wurde.

5.3.2 Datenbankstruktur

Da die Datenbank der medtec-Webanwendung keine große Anzahl an gleichzeitigen Zugriffen verarbeiten muss, konnte diese stark normalisiert werden um Redundanzen zu verhindern und garantiert trotzdem eine geringe Reaktionszeit.

Die Komplette Datenbankstruktur kann als ERD im Anhang gefunden werden.

5.3.3 Die „Employees“-Tabelle

Die Tabelle „Employees“ beinhaltet alle Therapeuten die in der Abteilung tätig sind. Zusammen mit ihren Stammdaten werden weiters die Email und das Passwort für den Zugang zur Webanwendung sowie zur App gespeichert. Außerdem wird das Berechtigungslevel, um herkömmliche und leitende Therapeuten zu unterscheiden, gespeichert.

Das „Token“ Attribut speichert den identifizierenden Token für die App bzw. das Smartphone des Benutzers. Über diesen werden die Benachrichtungen ausgestellt.

Damit bildet diese Tabelle zusammen mit der „Groups“-Tabelle das Herzstück der Datenbank.

⁴Vgl. Pröll, Zangerle und Gassler, *MySQL - Das umfassende Handbuch*, S. 42-43.

5.3.4 Die „Groups“-Tabelle

Die Tabelle „Groups“ beinhaltet alle Daten zu allen Therapiegruppen, die die Abteilung betreut. Dabei wird zusätzlich zu der Beschreibung der Gruppe das Arbeitsjahr, der Hauptverantwortliche sowie die beiden Stellvertreter und der Raum in dem die Therapie stattfindet abgespeichert. Das Arbeitsjahr wird deshalb gespeichert, damit über mehrere Jahre nachvollzogen werden kann, welche Therapeuten welche Gruppe betreut haben.

Abb. 5.1 stellt die beiden Tabellen mit ihren Attributen und jeweiligen Datentypen dar. Ebenfalls abgebildet sind die Beziehungen der Tabellen zueinander.

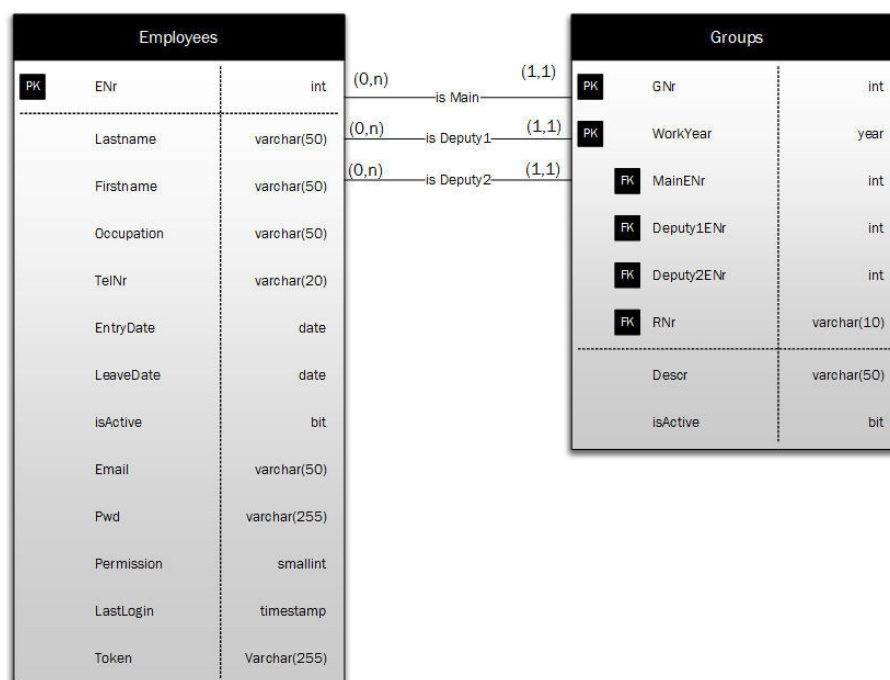


Abbildung 5.1: Employees & Groups Tabelle

5.3.5 Erweiterungen zur „Employees“-Tabelle

Um Redundanzen zu vermeiden werden alle zusätzlichen Informationen zu den Therapeuten in den Tabellen „Absences“, „WorkingHoursWeek“, „WorkingHours“ und „WorkingHours-Blockade“ abgespeichert. Darin sind sämtliche Informationen zu Abwesenheiten, Dienstzeiten und Blockaden, an denen der Therapeut nicht für Gruppen zur Verfügung steht, gespeichert. Außerdem können mit der „WorkingHoursWeek“ vergangene und zukünftige Änderungen in den Dienstzeiten abgebildet werden. Eine auf Abb. 5.2 nicht⁵ abgebildete Tabelle ist „AbsentToday“, in dieser werden die unerwarteten Abmeldungen über die App gesammelt und für die **Vertretungsverwaltung** verwendet.

⁵siehe ERD im Anhang

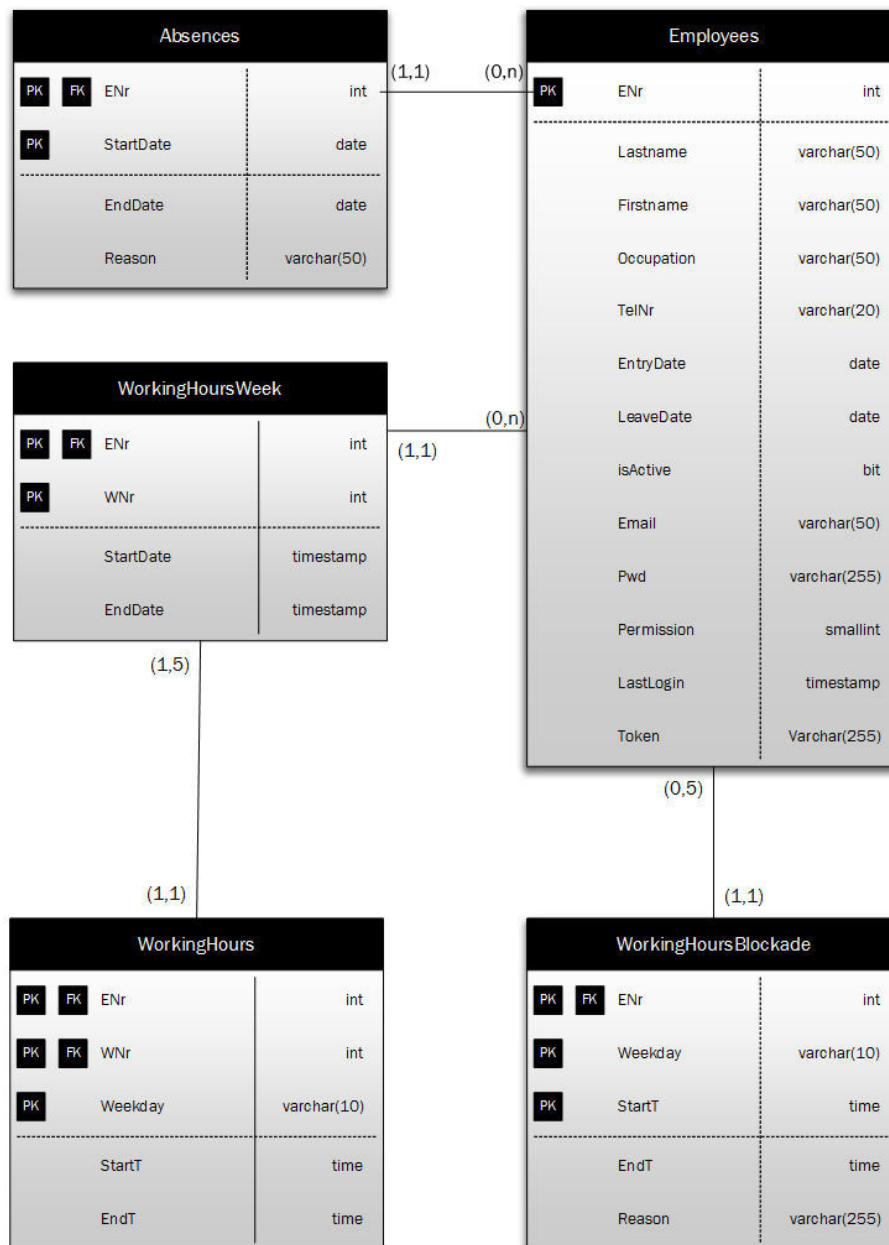


Abbildung 5.2: Employees Tabelle & Erweiterungen

5.3.6 Erweiterungen zur „Groups“-Tabelle

Auch bei der „Groups“-Tabelle wurde versucht, Redundanzen zu vermeiden. Es werden alle zusätzlichen Informationen, die für die Gruppen wichtig sind, in den Tabellen „GroupHours“, „GroupInactivity“ und „Rooms“ abgespeichert. Diese beinhalten die Wochentage und Zeiten zu denen die Gruppen stattfinden, in welchen Zeitintervallen eine Gruppe eventuell pausiert wird und in welchem Raum die Gruppe abgehalten wird.

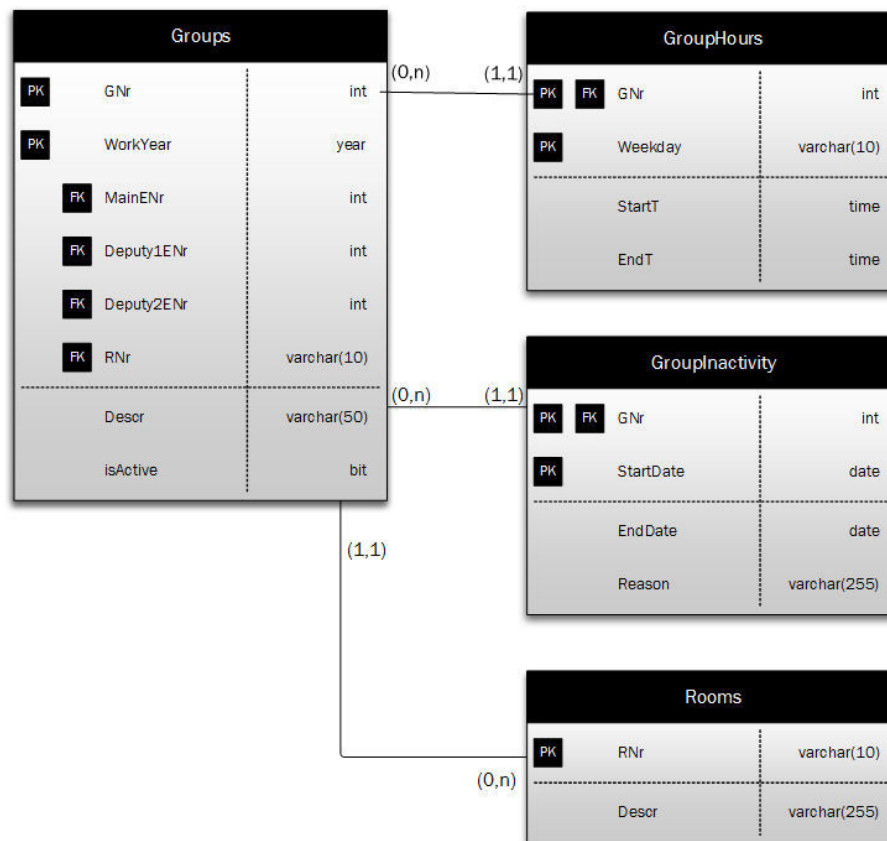


Abbildung 5.3: Groups Tabelle & Erweiterungen

5.4 Stored Procedures (SP)

Da alle Kommunikationen mit der Datenbank vorhersehbar sind und die Gefahr vor **SQL Injections** durch die Verwendung von Stored Procedures erheblich gesenkt werden kann, sind in der Datenbank 82 Prozeduren vordefiniert.

Der größte Teil dieser sind einfache SELECT, UPDATE, INSERT oder DELETE Anweisungen.

Ein simples Beispiel zu einer SELECT Abfrage, die alle aktiven Therapeuten liefert:

Listing 5.1: Abfrage aller Therapeuten

```

1 create procedure sp_getEmployees()
2 begin
3     select ENr, Lastname, Firstname, Occupation, TelNr,
4           DATE_FORMAT(EntryDate, "%Y/%m/%d") AS EntryDate,
5           DATE_FORMAT(LeaveDate, "%Y/%m/%d") AS LeaveDate,
6           isActive, Email, Pwd, Permission,
7           DATE_FORMAT>LastLogin, "%Y/%m/%d %H:%i:%s") AS LastLogin
8     from Employees
9     where isActive = 1;
10 end;
```

Die Definition solcher Prozeduren bringt zusätzlich den Vorteil einer geringeren Antwortzeit und der einfachen Wiederverwendbarkeit des Codes.

Zu diesen Routinen gehören weiters Abläufe, die zum Beispiel die Verfügbarkeit eines

Therapeuten an gewissen Tagen liefern. Diese liefern eine Anzahl zurück, welche bei keinem Ergebnis 0 bzw. bei einem gefundenen Ergebnis größer 0 ist. So überprüft folgendes Beispiel, ob der mitgeschickte Therapeut an dem mitgeschickten Wochentag bereits hauptverantwortlich für eine Gruppe ist:

Listing 5.2: Abfrage auf Hauptverantwortlichkeit

```

1 create procedure sp_hasMainResponsibility(_ENr int, _Weekday varchar(10))
2 begin
3     select count(g.GNr) as Cnt
4     from Groups g
5     join GroupHours gh on gh.GNr = g.GNr
6     where g.MainENr = _ENr and gh.Weekday = _Weekday and Workyear = year(current_date)
7     ;
8 end;
9 $$

```

Die Rückgabe des Dienstplanes, inklusive der momentanen Abwesenheiten, erfolgt ebenfalls über solch eine Prozedur. Dabei greift diese auch auf **User-Defined Functions (UDF)** zurück, die in der Datenbank vordefiniert sind, um zu ermitteln ob der abgefragte Therapeut an diesem Tag anwesend ist.

Listing 5.3: Abfrage des Dienstplanes

```

1 create procedure sp_getSP()
2 begin
3     select g.GNr, g.Descr, gh.Weekday, gh.StartT, gh.EndT,
4         coalesce(me.Lastname, 'Nicht') as LastMa,
5         coalesce(me.Firstname, 'Zugeteilt') as FirstMa,
6         fnc_hasAbsenceThisDay(me.Enr, fnc_WeekdayToDate(gh.Weekday)) as MainPresence,
7         coalesce(d1e.Lastname, 'Nicht') as LastD1,
8         coalesce(d1e.Firstname, 'Zugeteilt') as FirstD1,
9         fnc_hasAbsenceThisDay(d1e.Enr, fnc_WeekdayToDate(gh.Weekday)) as Dep1Presence,
10        coalesce(d2e.Lastname, 'Nicht') as LastD2,
11        coalesce(d2e.Firstname, 'Zugeteilt') as FirstD2,
12        fnc_hasAbsenceThisDay(d2e.Enr, fnc_WeekdayToDate(gh.Weekday)) as Dep2Presence
13    from Groups g
14    join GroupHours gh on gh.GNr = g.GNr
15    left join Employees me on g.MainENr = me.Enr
16    left join Employees d1e on g.Deputy1ENr = d1e.Enr
17    left join Employees d2e on g.Deputy2ENr = d2e.Enr
18    where g.Workyear = year(current_date) and g.isActive = 1
19    order by gh.StartT, g.GNr, gh.EndT;
20 end;

```

5.5 User-Defined Functions (UDF)

Um den Umgang mit Daten zu vereinfachen sind mehrere User-Defined Functions (UDF) in der Datenbank angelegt. Durch diese können Datensätze, zum Beispiel der Dienstplan mit seinen Abwesenheiten, direkt in der Datenbank vorbereitet werden. Dies senkt die Anzahl der benötigten Anfragen, an den Server, um im Frontend alle erforderlichen Daten, zum Anzeigen eines Menüpunktes, zu sammeln.

Die in Listing 5.4 gezeigte Funktion gibt zurück, ob ein Therapeut an einem gewissen Datum anwesend ist. Diese Funktion wird zum Beispiel in jener Prozedur verwendet, die den Dienstplan zurückgibt (siehe Listing 5.3).

Listing 5.4: Funktion zur Bestimmt der Anwesenheit eines Therapeuten

```

1 create function fnc_hasAbsenceThisDay(enr_ int, date_ date)
2 returns int
3 begin
4     declare _cnt int;
5     select count(enr)
6     into _cnt
7     from Absences a
8     where date_ between a.StartDate and a.EndDate and a.ENr = enr_;
9     if _cnt > 0 then
10         return 0;
11     else
12         return 1;
13     end if;
14 end;
```

5.6 Events

Damit die täglichen Abmeldungen aufgezeichnet werden können und trotzdem der Datenbankspeicher nicht unnötig befüllt wird, wird in der Datenbank ein Event verwendet. Ein Event in MySQL ist eine vorbereitete SQL Anweisung die nach einem **Schedule**, zu einem gewissen Zeitpunkt, abläuft.

Dieses Event (siehe 5.5) löscht alle eingetragenen Abmeldungen im Abstand von zwei Wochen.

Listing 5.5: Event zum leeren der „AbsentToday“-Tabelle

```

1 create event absent_reset_weeks
2 on schedule
3     every 14 day
4     starts '2017-04-03 20:00:00'
5     comment 'clears out the temporary absences every 2 weeks'
6 do
7     DELETE
8     FROM AbsentToday;
```

5.7 Trigger

Durch den Umstand, dass in MySQL keine „checks“ innerhalb einer Tabelle existieren und damit eingehende Daten nicht direkt beim Einfügen oder Abändern überprüft werden können, muss diese Überprüfung durch Trigger erledigt werden. Dies ist umständlich, aber notwendig um die Richtigkeit und Konsistenz der zu speichernden Daten zu gewährleisten.

Um den damit verbundenen Aufwand besser zu illustrieren, zeigen die Listings 5.6 & 5.7 den Unterschied bei der Implementierung.

Außerhalb der „check-Trigger“ werden noch weitere Trigger verwendet um zum Beispiel, Abwesenheiten innerhalb von anderen Abwesenheiten des gleichen Therapeuten zu verhindern.

Wie zu erkennen ist, werden statt einer Code-Zeile, 15 Code-Zeilen für die gleiche Funktionalität gebraucht. Außerdem muss der Trigger für das Einfügen als auch das Updaten eines Datensatzes existieren. Damit sind es sogar 30 Code-Zeilen, die für die Abbildung der genannten Funktionalität gebraucht werden.

Listing 5.6: Check Zeile innerhalb einer Tabelle zur Überprüfung des Berechtigungslevels

```
1 check(Permission = 1 or Permission = 0)
```

Listing 5.7: Trigger zur Überprüfung des Berechtigungslevels

```
1 create trigger tr_upd_Permission_in_Employees
2 after update
3 on Employees
4 for each row
5 begin
6     declare _newPerm smallint;
7     set _newPerm = new.Permission;
8
9     if new.Permission <> old.Permission then
10         if (_newPerm <> 0) and (_newPerm <> 1)
11             then
12                 signal sqlstate '45000' set message_text = 'Kein gueltiges Permission level!';
13             end if;
14         end if;
15 end;
```

Kapitel 6

Fazit

Zusammenfassend kann gesagt werden, dass die Diplomarbeit überwiegend positiv in Erinnerung bleiben wird.

Es war eine unglaublich wertvolle Erfahrung, für das spätere Arbeitsleben, in einem begrenzten Zeitraum, ein größeres Projekt umzusetzen. Durch eine konstruktive Zusammenarbeit und guter Abstimmung innerhalb des Teams konnten die Projektziele bis zur Gänze und sogar darüber hinaus erreicht werden. Bei Fragen oder Anregungen standen wir uns gegenseitig immer zur Verfügung.

Durch gutes Zeitmanagement und ausreichend Planung konnte auch die stressige Endphase in der Schulzeit, mit dem Ziel, die einzelnen Fächer gut abzuschließen, überwunden werden. Innerhalb des Teams wurde aus dieser Diplomarbeit gelernt, wie mit kniffligen Problemstellungen umgegangen wird und wie diese am besten gelöst werden können. Oft schien die Aufgabenstellung zu einem Thema auf den ersten Blick sehr trivial, was sich bei genauerer Beschäftigung mit dem Thema aber doch als herausfordernd entpuppte. Es ist der guten Zusammenarbeit des Teams zu verdanken, dass das Projekt mit Erfolg abgeschlossen werden konnte.

Das Team konnte viele neue Erfahrungen gewinnen, aber auch bereits vorhandenes Wissen auf dem Sektor der verwendeten Technologien einsetzen. Es ist schön mitanzusehen, wie sehr die Arbeit in der medizinisch-technischen Abteilung im Landesklinikum Wiener Neustadt durch die geschriebene Diplomarbeit erleichtert werden kann.

Glossar

Overhead

Ist die Menge der Rechenleistung die zur Verbindung verbraucht wird, im Verhältnis zur benötigten Rechenleistung für die eigentliche Abfrage.. 29, 66

Schedule

Ein Schedule ist ein geplanter Ablauf von (Datenbank-)Abfragen und Änderungen.. 63, 66

Scheduler

"*Ein (Datenbank-)Scheduler dient der Verwaltung von Schreib- und Lesezugriffen (sog. Operationen) auf Datenbankobjekten¹.*". 66

Session

Eine PHP Session dient der Authentifizierung des Benutzers am Server und bleibt so lange geöffnet bis diese durch einen Logout zerstört wird oder nach einem gewissen Zeitintervall ausläuft.. 33, 66

¹*Scheduler (Datenbank).*

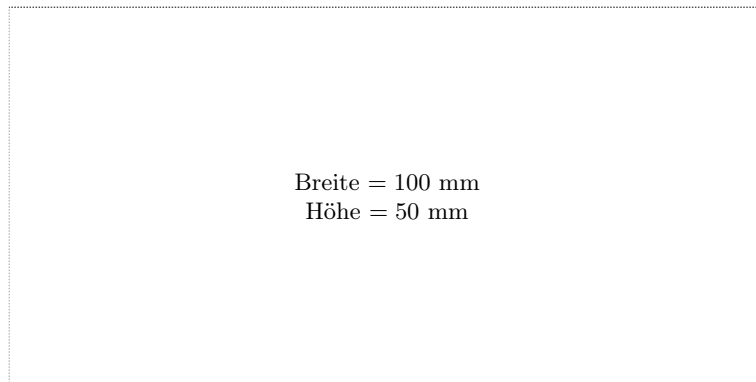
Literatur

- Android App Keystore*. Englisch. Ionic Docs. URL: <http://docs.ionic.io/services/profiles/#android-app-keystore>.
- Balzert, Helmut. *Java: Objektorientiert programmieren*. W3L GmbH, 2010.
- CURL*. PHP. URL: <http://php.net/manual/de/book.curl.php>.
- curl. *curl - command line tool and library for transferring data with URLs*. Englisch. URL: <https://curl.haxx.se/>.
- Eric Freeman, Elisabeth Robson. *Entwurfsmuster von Kopf bis Fuß*. O'Reilly, 2015.
- Fender, Luke. *YAML vs. JSON*. Englisch. 2017. URL: <https://www.json2yaml.com/yaml-vs-json> (besucht am 08.03.2017).
- Florin, Alexander. *User - Interface - Design*. Books on Demand GmbH, 2015.
- Framework, Ionic. *Installing Ionic and its Dependencies*. Englisch. 2016. URL: <http://ionicframework.com/docs/guide/installation.html>.
- Hillebrand, Kurt. *Datenbanken und Informationssysteme*. Eigenverlag K.Hillebrand, 2014.
- HTML5 localStorage*. 2012. URL: <http://html5-webdesign.de/localStorage.html>.
- Hybrid Mobile App Entwicklung mit Ionic*. 2015. URL: <https://www.creativeworkline.at/2015/09/hybrid-mobile-app-entwicklung-ionic/>.
- Interpreter Injection*. Englisch. Uniface. URL: http://unifaceinfo.com/docs/0906/Uniface_Library_HTML/ulibrary/webSecurity_InterpreterInjection_241236CD89757302E3CB327D0.html (besucht am 25.03.2017).
- Ippen, Johannes. *Web Fatale: Wie Du Webseiten und Web-Apps gestaltest, denen niemand widerstehen kann: Usability, User Experience und Interaktion*. Rheinwerk Verlag, 2016.
- Lamb, Daniel. *jQuery vs. AngularJS: A Comparison and Migration Walkthrough*. Englisch. 2014. URL: <https://www.airpair.com/angularjs/posts/jquery-angularjs-comparison-migration-walkthrough> (besucht am 14.03.2017).
- Mattschek, Markus. *Marken, Logos und Symbole - Einfluss und Bedeutung*. 2015. URL: <http://www.onlinemarketing-praxis.de/basisinformationen/marken-logos-und-symbole-einfluss-und-bedeutung> (besucht am 08.03.2017).
- Net Losses: Estimating the Global Cost of Cybercrime*. Englisch. McAfee. URL: <https://www.mcafee.com/us/resources/reports/rp-economic-impact-cybercrime2.pdf> (besucht am 20.03.2017).
- ngcordova. *cordovaPushV5*. Englisch. [Online; Stand 18. März 2017]. URL: <http://ngcordova.com/docs/plugins/pushNotificationsV5/>.
- Niemann, Alexander. *JSON (JavaScript object notation)*. 2013. URL: <http://www.itwissen.info/JSON-JavaScript-object-notation.html> (besucht am 08.03.2017).
- Nodejs. *Über Node.JS*. 2017. URL: <https://nodejs.org/de/about/>.
- Oracle Completes Acquisition of Sun*. Englisch. Oracle. URL: <http://www.oracle.com/us/corporate/press/044428> (besucht am 27.01.2010).
- PDO Object*. Englisch. PHP Group. URL: <http://php.net/manual/en/intro.pdo.php> (besucht am 20.03.2017).

- Pechhacker, Dominik. "Single Page Webapplikationsentwicklung anhand des JavaScript Frameworks Ember.js". Bachelorarbeit. FH Joanneum, 2014.
- Pröll, Stefan, Eva Zangerle und Wolfgang Gassler. *MySQL - Das umfassende Handbuch*. Rheinwerk Verlag, 2015.
- Request for Comments*. Englisch. Internet Engineering Task Force(IETF). URL: <https://tools.ietf.org/html/rfc760> (besucht am 01.01.1980).
- Rohr, Matthias. *Sicherheit von Webanwendungen in der Praxis*. Springer Vieweg, 2015.
- Scheduler (Datenbank)*. Wikipedia. URL: [https://de.wikipedia.org/wiki/Scheduler_\(Datenbank\)](https://de.wikipedia.org/wiki/Scheduler_(Datenbank)) (besucht am 02.04.2017).
- Schonmann, Frank. *Einfuehrung in XML*. Techn. Ber. UNI Bielefeld, 2003.
- Simon. *The Complete Ionic Push Notifications Guide*. Englisch. 2016. URL: <https://devdactic.com/ionic-push-notifications-guide/>.
- Skalierbares HTTP Long Polling*. 2013. URL: <http://accso.de/magazin/skalierbares-http-long-polling/>.
- Skolski, Pawet. *Single Page Application vs. Multiple Page Applications*. Englisch. 2017. URL: <https://neoteric.eu/single-page-application-vs-multiple-page-application/> (besucht am 08.03.2017).
- Starting an Ionic App*. Englisch. Ionic Framework. URL: <http://ionicframework.com/docs/cli/start.html>.
- Steyer, Manfred. *Angular JS - Moderne Webanwendungen und SPAs mit JavaScript*. O'Reilly, 2015.
- The OAuth 2.0 Authorization Framework: Bearer Token Usage*. Englisch. Microsoft. 2012. URL: <https://tools.ietf.org/html/rfc6750>.
- WampServer. *WAMPSEVER, a Windows web development environment*. Englisch. URL: <http://www.wampserver.com/en/>.
- Weiß, Bengt. *AngularJS Ionic Framework*. Carl Hanser Verlag, 2016.
- Wikipedia. *Mobile App* — *Wikipedia, Die freie Enzyklopädie*. [Online; Stand 18. März 2017]. 2017. URL: https://de.wikipedia.org/w/index.php?title=Mobile_App&oldid=163530806.
- *Plug-in* — *Wikipedia, Die freie Enzyklopädie*. [Online; Stand 22. März 2017]. 2017. URL: <https://de.wikipedia.org/w/index.php?title=Plug-in&oldid=163758413>.
- *XAMPP* — *Wikipedia, Die freie Enzyklopädie*. [Online; Stand 18. März 2017]. 2017. URL: <https://de.wikipedia.org/w/index.php?title=XAMPP&oldid=163325861>.

Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —



— Diese Seite nach dem Druck entfernen! —