



**Technische Hochschule**  
**Brandenburg**  
University of  
Applied Sciences

## **Technische Dokumentation**

Die Entwicklung eines Entscheidungsunterstützenden Systems am  
Beispiel Brustkrebs

Vorgelegt von Ali Ridha Haouari  
Matr-Nr:20020445

Am SS 2016/2017

Betreuer: Prof. Dr. Eberhard Beck

# Inhaltsverzeichnis

Inhaltsverzeichnis.....

Abbildungsverzeichnis.....

<b>1</b>	<b>Einleitung.....</b>	<b>5</b>
1.1	Einführung.....	5
1.2	Motivation und Ziel der Arbeit.....	5
1.3	Prävention & Früherkennung.....	6
<b>2</b>	<b>Anforderungsmanagement und Anforderungsdefinitionen.....</b>	<b>6</b>
2.1	Technische Anforderungen an das System(Back-End).....	7
2.2	Visuelle Anforderung an das System (Front-End).....	7
2.3	Planung des Projektes.....	8
<b>3</b>	<b>Funktionalität des Systems(Entwurf).....</b>	<b>9</b>
3.1	Einleitung .....	11
3.2	User Case.....	11
3.3	Sequenzen Diagramm.....	12
3.4	Komponenten Diagramm.....	12
3.5	Sequenzen Diagramm (Registrieren). ....	12
3.6	Sequenzen Diagramm (Anmeldung). ....	13
3.7	Sequenzen Diagramm (Abtastung). ....	14
3.8	Datenbank-Design .....	15
<b>4</b>	<b>Material und Methoden ( Font End und Back End).....</b>	<b>15</b>
4.1.1	RESTful JSON Web Services.....	15
4.1.2	Framework Spring Boot.....	17
4.1.3	Framework Angular JS.....	18
4.1.4	Framework Hibernate und MySQL server.....	18

4.1.5	Framework Bootstrap .....	18
4.1.6	Maven Apache Tomcat.....	19
4.1.7	Hibernate.....	19
4.1.8	jQuery .....	19
4.1.9	CSS .....	19
<b>5</b>	<b>Lösungsarchitektur und Umsetzung .....</b>	
4.1	Client/Server .....	20
<b>5</b>	<b>Ergebnisse.....</b>	
5.1	Implementierung.....	26
<b>6</b>	<b>Technologie.....</b>	29
<b>7</b>	<b>Diskussion, Ausblick und Schlussfolgerung .....</b>	
7.1	Ausblick.....	
7.2	Fazit.....	
<b>8</b>	<b>Literaturverzeichnis .....</b>	33
<b>9</b>	<b>Anhang.....</b>	

## Abbildungsverzeichnis

Abbildung 1. Behandlungsgrundstruktur.....	6
Abbildung 2. Projekt Planung.....	6
Abbildung 3. User Case Patient .....	11
Abbildung 4. User Case Azt .....	10
Abbildung 5. Diagramm Class .....	12
Abbildung 6. Sequenzen Diagramm (Registrieren). .....	13
Abbildung 7. Sequenzen Diagramm Abtestung .....	14
Abbildung 8. Datenbank-Design .....	15
Abbildung 9. Architektur Client/Server .....	16
Abbildung 2. Startseite Erreichbarkeit .....	20
Abbildung 11. Architektur (MVC) .....	21
Abbildung 12. Architektur (MVC)[AG] .....	23
Abbildung 13. Login .....	24
Abbildung 14.Registregung .....	25
Abbildung 15. Password vergessen.....	25
Abbildung 16. Startseite Erreichbarkeit .....	26
Abbildung 17. Seite Selbstuntersuchung.....	27
Abbildung 18. Seite Selbstuntersuchung.....	27
Abbildung 19. Navigation .....	28
Abbildung 20. Musik Hören .....	29
Abbildung 21.Resultat des Testes .....	29

# **1 Einleitung**

In dem folgenden Kapitel geht es darum, die Ziele, die Motivationen und die Nachhaltigkeit dieser Arbeit zu erläutern. Weiterhin wird auf die Herausforderungen und die Problemstellungen eingegangen, die mit der Entwicklung und der Konzeption der Anwendung zu erwarten waren.

## **1.1 Einführung**

Brustkrebs ist die häufigste Krebserkrankung der Frauen in Deutschland. Rund 5'350 Frauen erkranken jährlich an Brustkrebs, ca. 1'350 sterben daran. Damit hat Brustkrebs sowohl die höchste Inzidenz- als auch Mortalitätsrate aller Krebsarten[FR]. Aktuelle Trends zeigen jedoch, dass die Überlebenschancen steigen und die Behandlungsmöglichkeiten stetig besser werden. Bisher gibt es noch kein Medikament, welches der Brustkrebserkrankung vorbeugen kann. Daher ist die Früherkennung die wichtigste Maßnahme, die Heilungs- und Überlebenschance zu verbessern. Je früher der Krebs entdeckt wird, umso besser sind diese Chancen und umso schonender ist die Behandlung. Es gibt unterschiedliche Methoden, einen Tumor möglichst früh zu erkennen. Auf eine dieser Methoden gehe ich in meiner Arbeit genauer ein, das Mammographie-Screening. In vielen Kantonen hat sich ein systematisches Screening bereits durchsetzen können, in anderen wird noch heftig über dessen Nutzen diskutiert. In meiner Arbeit gehe ich sowohl auf die theoretischen Grundlagen des Brustkrebses ein, als auch spezifisch auf das Mammographie-Screening mit dessen Wirksamkeit und Entwicklung, sowie dessen Vor- und Nachteilen[FR].

## **1.2 Motivation und Ziel der Arbeit**

Medizinische Entscheidungen sind durch die teilweise unzureichende Dokumentation nicht immer transparent und daher schwer zu objektivieren. Ziel dieser Arbeit ist die Entwicklung eines

entscheidungsunterstützenden Systems am Beispiel Brustkrebs.

Bei der Entwicklung und Implementierung des Systems lagen die Faktoren auf der medizinischen Aussagekraft, sowie bei der Programmierung einer funktionierenden Webanwendung. Die hilft die Frauen um Brustkrebs früher wie möglich zu erkennen, damit die Risiken reduziert werden.

### 1.3 Prävention & Früherkennung

Die Prävention ist klar von der Früherkennung zu unterscheiden, denn es werden jeweils vollkommen andere Ziele angestrebt. Während die Prävention das Ausbrechen einer Krankheit zu verhindern versucht, zielt die Früherkennung auf eine möglichst frühe Diagnose einer Krankheit ab[FR].

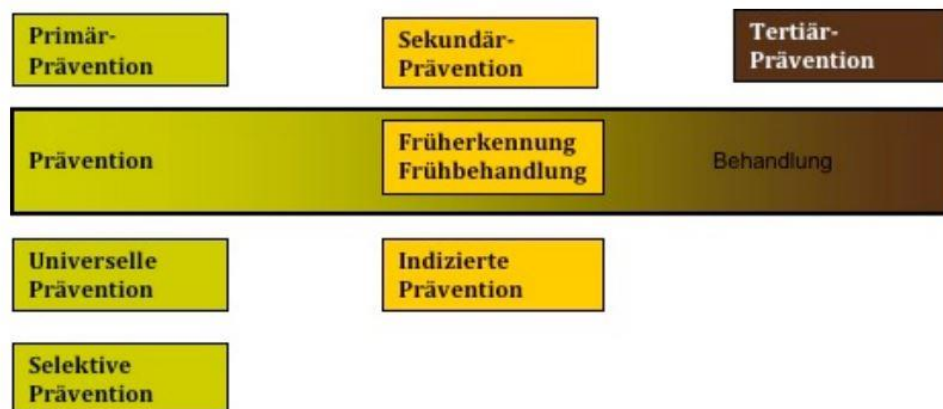


Abbildung 1. Behandlungsgrundstruktur

Primärprävention liegt dann vor, wenn die Maßnahmen sich an die Frauen richten, bei denen ein Problem noch nicht aufgetreten ist. Von Sekundärprävention ist die Rede, wenn sich die Maßnahmen auf die Früherkennung der betreffenden Probleme konzentriert. Von der Tertiärprävention sprechen wir, wenn das Problem bereits manifest geworden ist und die Maßnahmen zum Ziel haben, einer Verschlimmerung des Problems oder der Folgeprobleme zu verhindern[FR].

## 2 Anforderungsmanagement und Anforderungsdefinitionen

Um die Arbeit erfolgreich zu realisieren, wurde in mehreren Projekttreffen ein Anforderungskatalog erstellt. Dieser beinhaltet die definierten Ziele. notwendigen Umsetzungen, aufgeteilt nach:

technischen Anforderungen, visuellen Anforderungen und inhaltlichen Anforderungen. Das Anforderungsmanagement ist ein wichtiger Bestandteil, um dem angeforderten Bedarf gerecht zu werden. In dem folgenden Abschnitt werden die zusammengetragenen Anforderungen aufgeführt und spezifiziert[\*].

## **2.1 Technische Anforderungen an das System (Back-End)**

Bei der Webanwendung handelt es sich um ein System, in das gesundheitsbezogene Daten eingegeben werden können. Die Eingabe erfolgt. Dennoch wird auf einer Datenbank gespeichert. Hierbei ist es notwendig, dass die Webanwendung auf einem Server läuft, der gegen Angriffe resistent und der Zugang mit Passwörtern geschützt ist. Neben der Eingabe bietet das System die Möglichkeit, die Daten aus einer Datenbank auszulesen.

Die Webanwendung besteht aus Zwei Teile Client und Server in dieser Beziehung fordert das eine Programm, der Client, einen Dienst beim anderen Programm, dem Server, an. Dieser erfüllt dann die Anforderung. Die Client/Server kann auch von Programmen innerhalb eines einzelnen Computers verwendet werden, wichtiger aber ist sie für Netzwerke. In einem Netzwerk stellt das Client/Server-Modell einen probaten Weg dar, um die Verbindung zwischen Programmen herzustellen, die aus Effizienzgründen an verschiedenen Orten liegen. Die Webanwendung bietet die Möglichkeit, dass ein Patient und eine Arzt ein Konto erstellen konnte, Außerdem ein Test bzw. die Monatliche Selbstuntersuchung der Brust in 8 Schritten, jeder Test hat ein Ergebnis. Als die Resultat bekommt man die gute Empfehlung entweder einen Termin vereinbaren oder nimmt sich im nächste Monat, am besten kurz nach der Regelblutung, Zeit für eine gründliche Selbstuntersuchung der Brust einen neue Test damit, erhält man eine E-Mail der Erinnerung an die Selbstuntersuchung der Brust zur Vorsorge von Brustkrebs. Auch bietet, dass Patient durch die Test Entspannungsmusik hören kann und einen Termin in der Nähe zu buchen wo man wohnt.

Das System enthält ein Login- und Registrierungssystem. Die Passwörter der Benutzer sind durch gängige Standards und im notwendigen Umfang vor Angriff und Missbrauch geschützt. Der Programmcode ist übersichtlich, nachvollziehbar, und transparent, sodass Änderungen an der Funktionalität jederzeit durch befugte Personen durchgeführt werden können.

## **2.2 Visuelle Anforderung an das System (Front-End)**

Die Programmierung wurde mit aktuellen Versionen von Java AngularJs Gulp, und HTML 5 C CSS3 JQuery Und Bootstrap Java Script RestFull Web Service durchgeführt.

Alle Unterseiten, die von der Hauptseite erreicht werden, werden in mit einem CSS Front-End ausgestattet. Die mobilen Endgeräte skalieren nach der Implementierung von dem aktuellen Bootstrap CSS 3 die Haupt-und Unterseiten selbstständig auf die jeweilige Displayauflösung. Durch diese Funktion ist eine Skalierung für jede Displaygröße realisierbar. Die Oberfläche wurde minimalistisch aufgebaut, damit die Übersicht gewährleistet wird. Die Startseite bietet Verlinkungen zu den Internetauftritten Die Monatliche Selbstuntersuchung der Brust auch Feedback zu geben, Musik hören, Online Termin buchen. Die web Anwendung hilft bei Durchführung der und Erinnerung an die Selbstuntersuchung der Brust zur Vorsorge von Brustkrebs. Die Hauptidee ist es, die sieben Schritte einer Selbstuntersuchung zu zeigen und eine einfache Anleitung zu geben. Die Untersuchung selbst ist ganz simpel und in weniger als fünf Minuten durchgeführt. Der beste Zeitpunkt dafür ist unmittelbar nach der Monatsblutung. Die Anwendung zeigt mit anschaulichen Bildern die notwendigen Schritte und liefert Informationen zur Ursache von Brustschmerzen und anderen Beschwerden.

Wenn ein Benutzer ein Login durchführt, wird er mit zwei verschiedenen Meldungen auf das jeweilige Verhalten aufmerksam gemacht.

Falsche Benutzerdaten werden rot markiert und beinhalten die Benachrichtigung, dass die falschen Daten eingegeben wurden.

Sie können ein neues Passwort einrichten, wenn Sie Ihr altes aus Sicherheitsgründen ändern möchten oder es vergessen haben. Das Passwort für Ihr E-Mail bekommen.

## **2.3 Planung des Projektes**



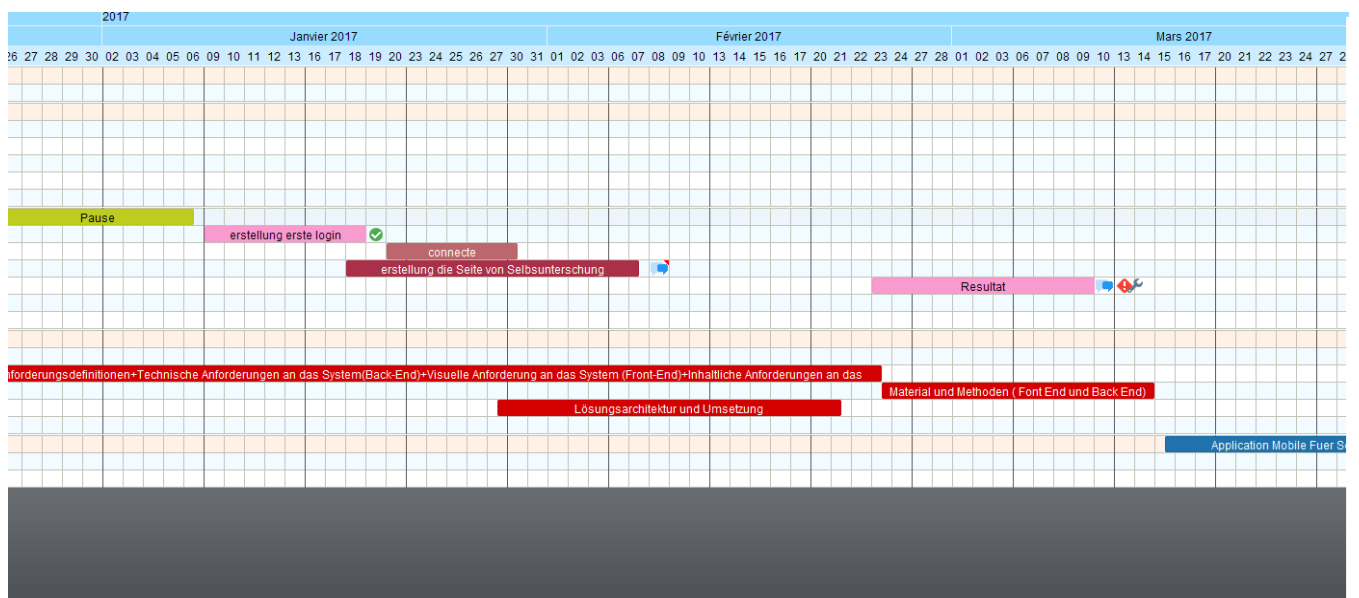
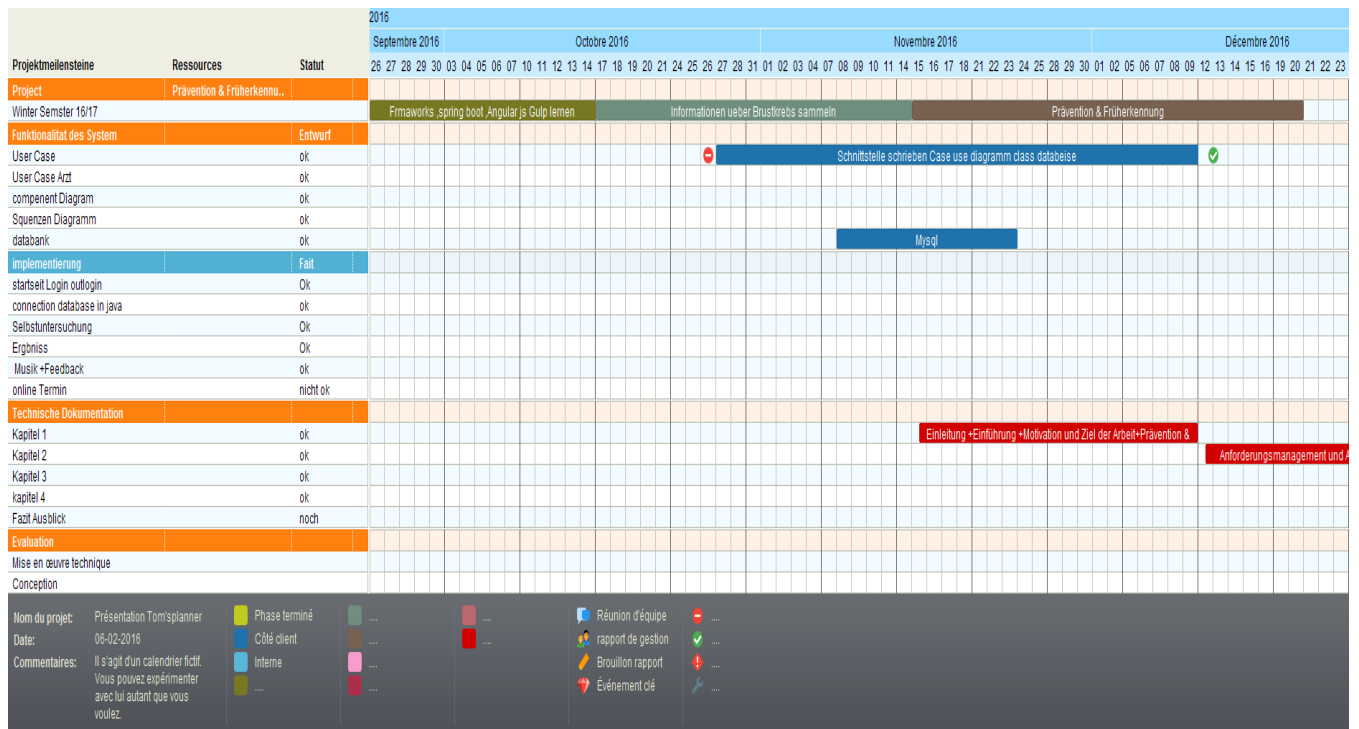


Abbildung 2. Projekt Planung

### 3. Funktionalität des Systems(Entwurf )

### 3.1 Einleitung

In dem folgenden Kapitel geht es darum, die Funktionalität des Systems, von Analyse zum Entwurf dieser Arbeit zu erläutern.

### 3.2 User Case (Patient)

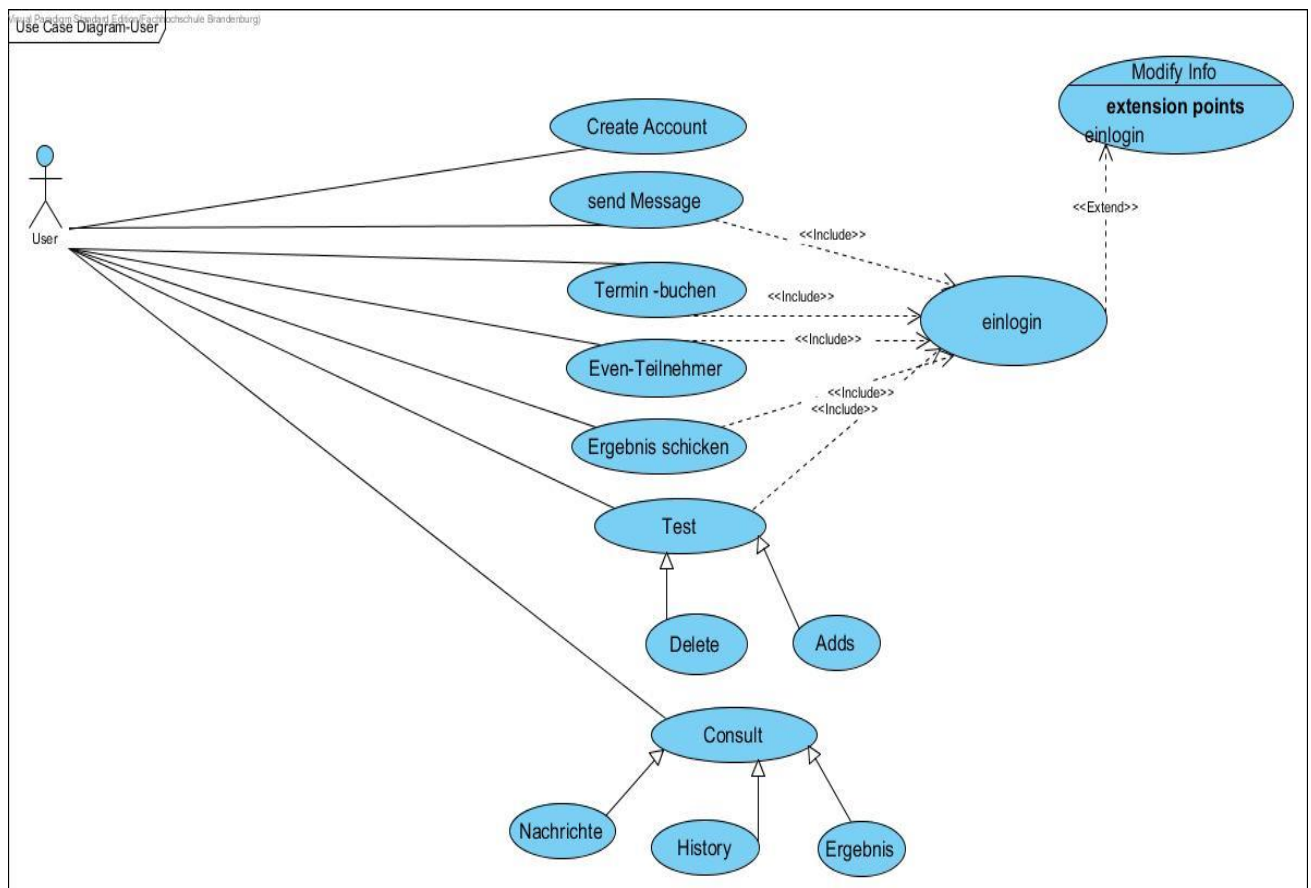


Abbildung 3. User Case Patient

Name	Value
Name	Use Case MVC

Jeder Patient kann eine oder Mehrere Konto erstellen, Auch Nachrichten senden und Feedback geben .Außerdem Test durchführt und es ist in der Lager einen Termin zu vereinbaren. Die Selbstuntersuchung der Brust bzw., Test kann man löschen oder hinzufügen auch die Verlauf der Ergebnis konsultiere

### 3.3 User Case (Arzt).

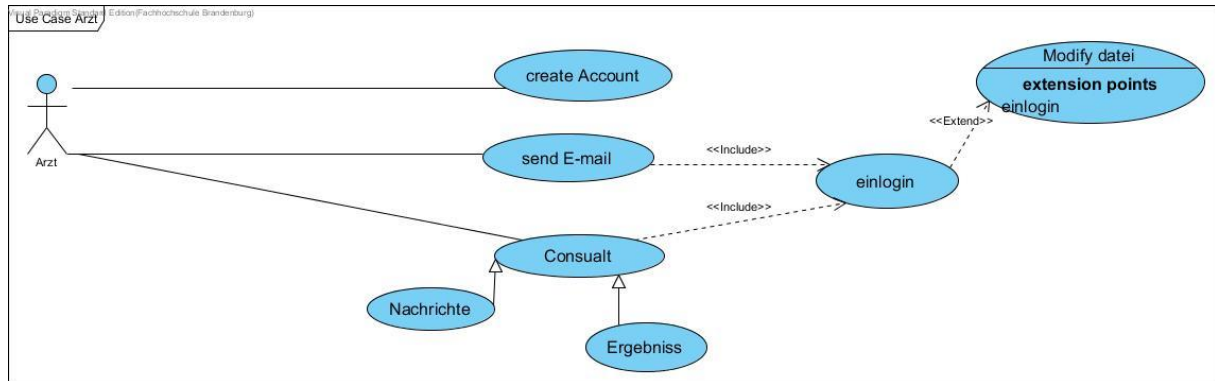


Abbildung 4. User Case Azt

Name	Value
Name	Use Case Arzt

### 3.4 Component Diagramm Class

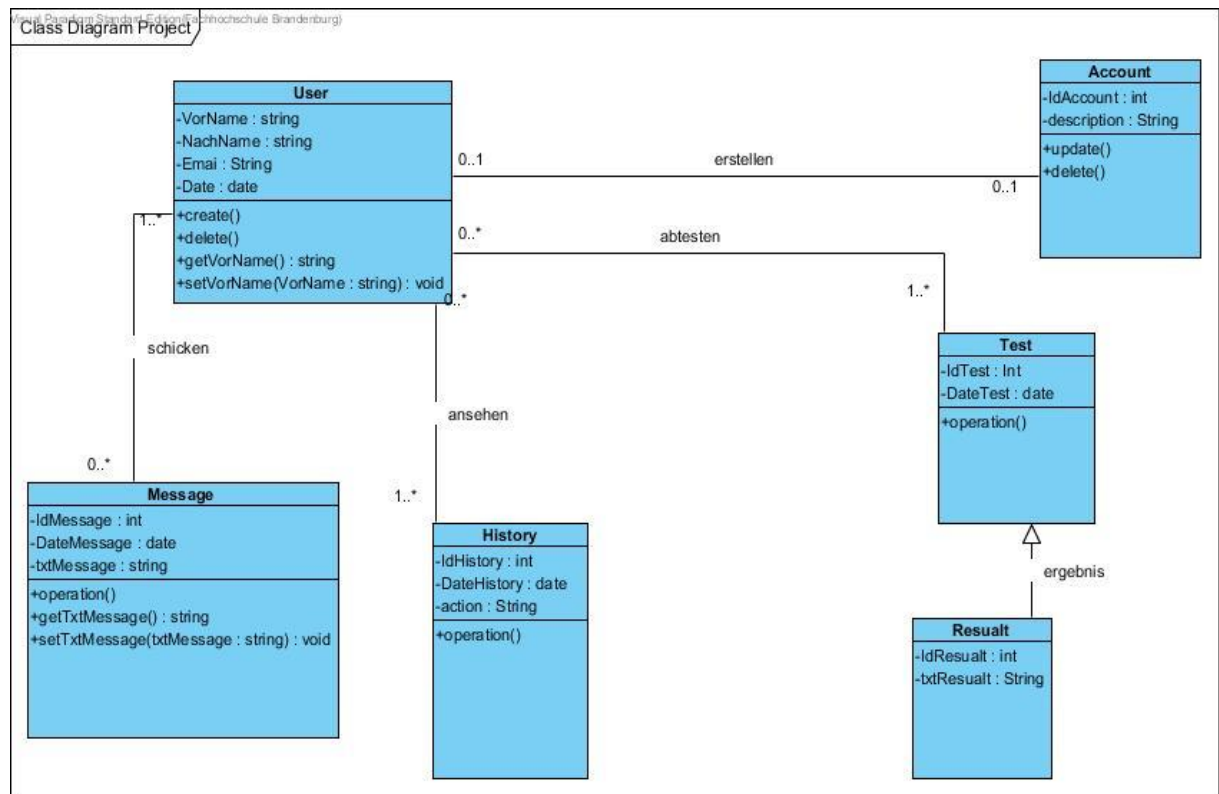


Abbildung 5. Diagramm Class

Name	Value
Class	Diagram class App

### 3.5 Sequenzen Diagramm (Registrieren).

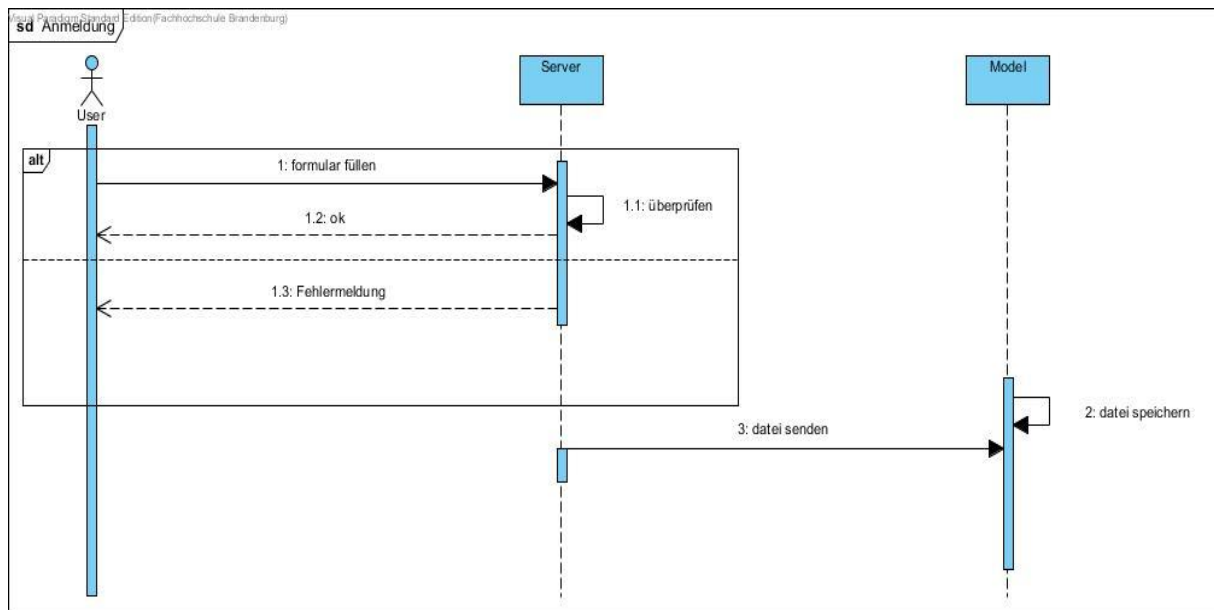


Abbildung 6. Sequenzen Diagramm (Registrieren).

Name	Value
Name	Sequence Diagram « registrierung der User in Webseite »

### 3.6 Sequenzen Diagramm (Anmeldung).

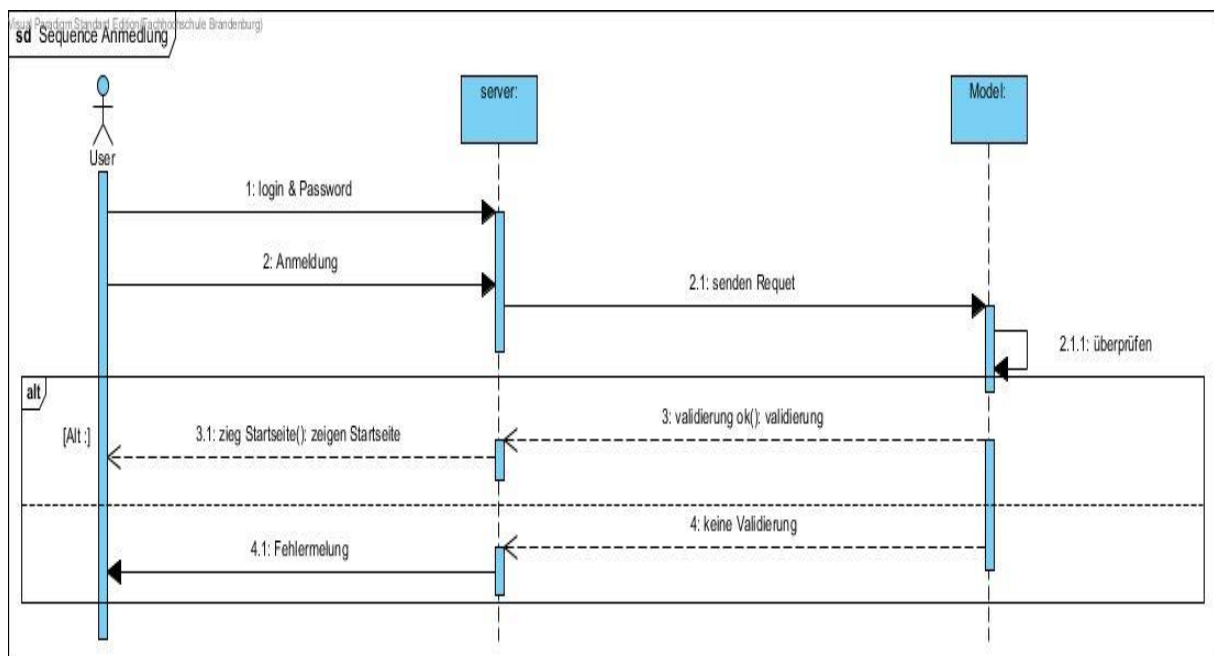


Abbildung 6. Sequenzen Diagramm (Anmeldung)

Name	Value
Name	Sequence Diagram « anmeldung der User in Webseite »

### 3.7 Sequenzen Diagramm (Abtastung).

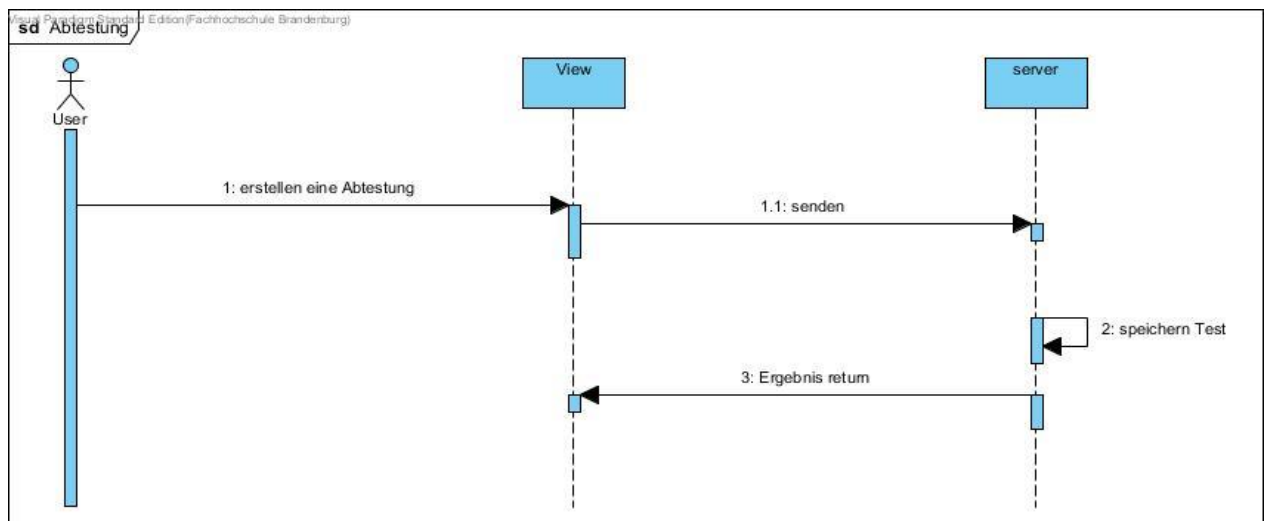


Abbildung 7. Sequenzen Diagramm Abtastung

Name	Value
Name	Sequence Diagram « Test der User in Webseite »

### 3.8 Datenbank-Design

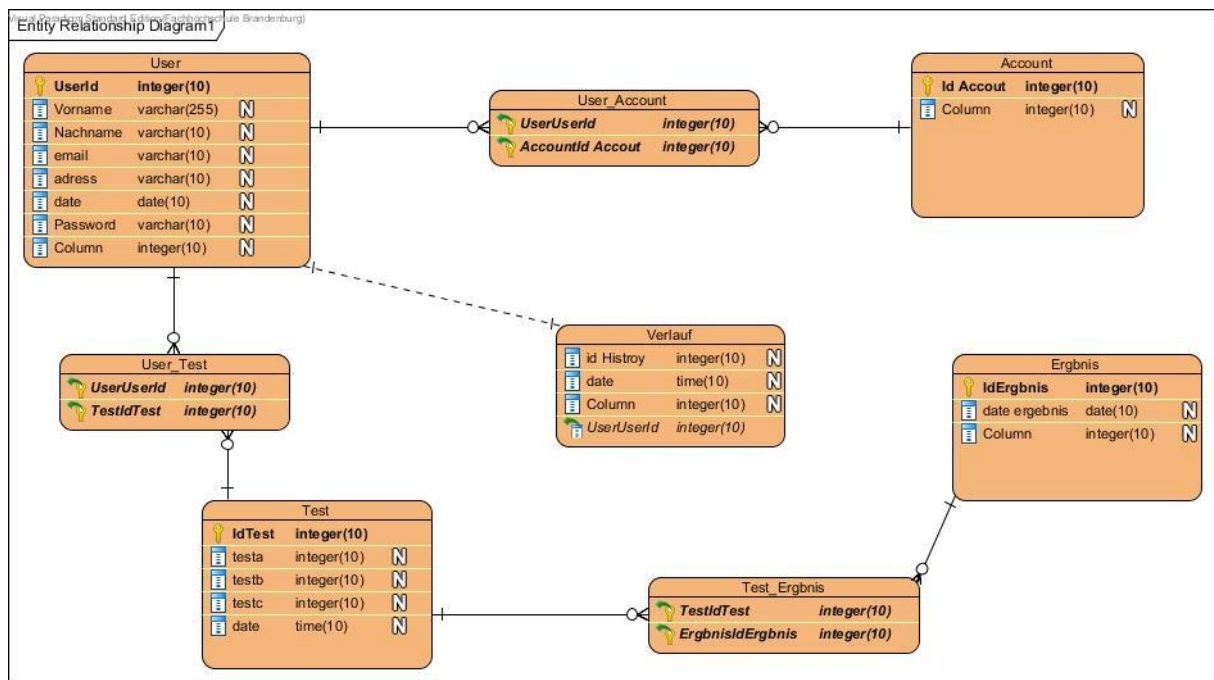


Abbildung 8. Datenbank-Design

Die Folgende Abbildung zeigt die Darstellung Daten Zwischen die Class

## 4. Material und Methoden ( Font End und Back End)

In dem folgenden Unterkapiteln werden die Grundlagen der verwendeten Materialien und Methoden beschrieben und im weiteren Verlauf der Arbeit das Einsetzen dieser Methoden genauer erläutert.

### 4.1.1 RESTful JSON Web Services

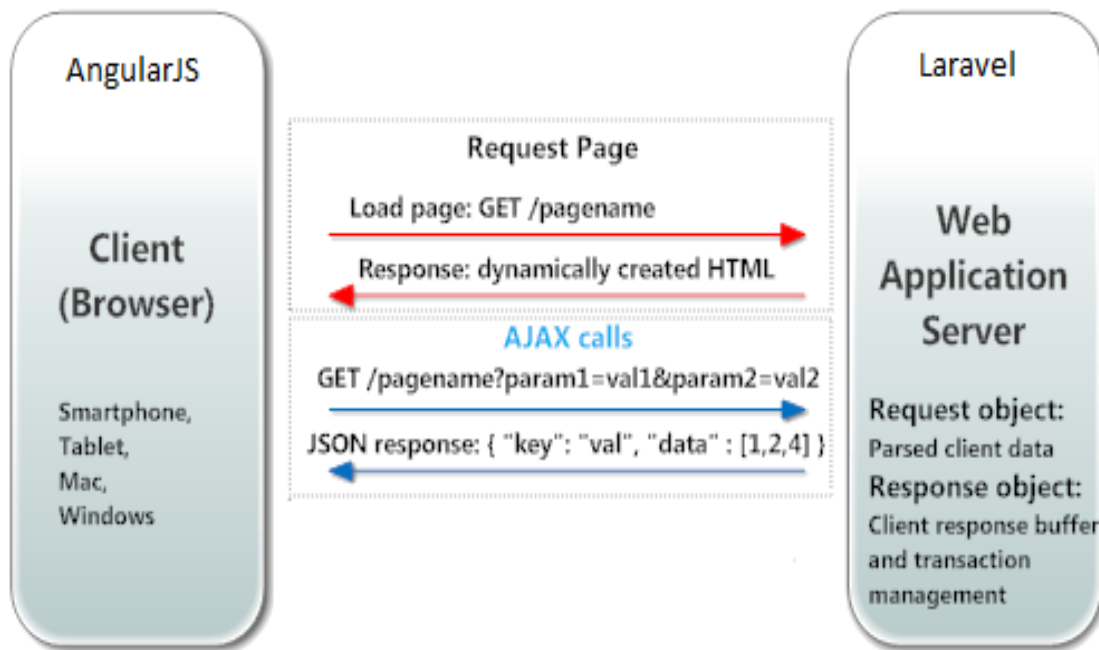


Abbildung 8. Architektur Client/Server

Für die Kommunikation zwischen dem Java-Server und den Clients dienen JSON, die in der Kommunikationsbibliothek festgelegt und gesteuert werden. Mit JSON lassen sich Verbindungen für einen Informationsaustausch realisieren. Ein json stellt einen Adressen-Struktur dar und dient als Kommunikationsendpunkt. Diese Adressen-Struktur ist eine Kombination aus einer IP-Adresse und einem Port. Damit kann ein Programm, in unserem Fall die Clients, ein bestehendes System (Server) ansprechen.

Ein Webservice (auch Webdienst) ermöglicht die Maschine-zu-Maschine-Kommunikation auf Basis von HTTP oder HTTPS über Rechnernetze wie das Internet. Dabei werden Daten ausgetauscht und auf entfernten Computern Funktionen aufgerufen. Jeder Webservice besitzt einen Uniform Resource Identifier (URI), über den er eindeutig identifizierbar ist, sowie eine Schnittstellenbeschreibung in maschinenlesbarem Format (als XML-Artefakt, z. B. WSDL), die definiert, wie mit dem Webservice zu interagieren ist. Die Kommunikation kann (muss aber nicht) über Protokolle aus dem Internetkontext wie HTTP laufen und kann XML oder JSON basiert sein.[AG][WEB]



#### 4.1.2 Spring Boot

Mit Spring Boot können **autonomen** (autarke oder **selbständig ausführbare**) Java Anwendungen (JAR) auf Basis des **Spring Ökosystems** erstellt werden (*Self-contained*).



Die erstellten Programme beinhalten alle benötigten Komponenten und Bibliotheken. Der Webserver wird, wie in diesem Beispiel, vorkonfiguriert in die Anwendung integriert (*Auto-Konfiguration*).

Durch das Verwenden des Paradigmas *Konvention vor Konfiguration* kommt die Anwendungen ohne XML-Konfiguration aus (*JavaConfig*).

Gesteuert wird der Zusammenbau der **Spring Boot Webanwendung** durch das Buildsystem Apache Maven mit einem speziellen Plug-In von Spring.

Spring Boot eignet sich auf Grund dieser genannten Eigenschaften hervorragend als Grundlage für Micro Services (**Martin Fowler**) oder Self-contained Systems (**SCS**).

Durch die genannten Eigenschaften werden folgende Ziele mit von Spring Boot erreicht.

- **Schnelleres Aufsetzen** von Spring Projekten durch Vereinfachungen.
- Out-of-the-box Verhalten durch Autokonfigurationen mit sinnvollen Anfangswerten.
- Nichtfunktionale Features werden bereitgestellt. [SP]

#### 4.1.3 Angular JS



AngularJS – oft einfach als Angular bezeichnet – ist ein clientseitiges JavaScript-Webframework zur Erstellung von Single-page-Webanwendungen nach einem Model-View-ViewModel-Muster. Die Softwareentwicklung und das Komponententesten können damit vereinfacht werden. Es wird als Open-Source-Framework vom US-amerikanischen Unternehmen Google Inc. entwickelt [AN].

#### 4.1.4 MySQL



MySQL ist eines der weltweit verbreitetsten relationalen Datenbankverwaltungssysteme. Es ist als Open-Source-Software sowie als kommerzielle Enterprise Version für verschiedene Betriebssysteme verfügbar und bildet die Grundlage für viele dynamische Webauftritte.

MySQL wurde seit 1994 vom schwedischen Unternehmen MySQL AB entwickelt. Im Februar 2008 wurde MySQL AB vom Unternehmen Sun Microsystems übernommen, das seinerseits im Januar 2010 von Oracle gekauft wurde.

Der Name MySQL setzt sich zusammen aus dem Vornamen My, den die Tochter des MySQL AB Mitbegründers Michael Widenows trägt, und SQL. [1]

#### 4.1.5 Maven



Maven ist ein Build-Management-Tool der Apache Software Foundation und basiert auf Java. Mit ihm kann man insbesondere Java-Programme standardisiert erstellen und verwalten. Der Name Maven kommt aus dem Jiddischen und bedeutet so viel wie „Sammler des Wissens“.[2]

#### 4.1.6 Bootstrap



Bootstrap ist ein freies CSS-Framework. Es enthält auf HTML und CSS basierende Gestaltungsvorlagen für Typografie, Formulare, Buttons, Tabellen, Grid-Systeme, Navigations- und andere Oberflächengestaltungselemente sowie zusätzliche, optionale JavaScript-Erweiterungen. Es wird unter anderem von der NASA und dem US-amerikanischen Nachrichtensender MSNBC eingesetzt.[3]

#### 4.1.7 Hibernate



Bei Hibernate (englisch für Winterschlaf halten) ist ein Open-Source-Persistenz- und ORM-Framework für Java. Für .NET ist eine portierte Version namens Hibernate verfügbar. Hibernates Hauptaufgabe ist die objektrelationale Abbildung (englisch O-R-Mapping, kurz ORM). Dies ermöglicht es, gewöhnliche Objekte mit Attributen und Methoden (im Java-Umfeld POJOs (für Plain Old Java Objects) genannt) in relationalen Datenbanken zu speichern und aus entsprechenden Datensätzen wiederum Objekte zu erzeugen. Beziehungen zwischen Objekten werden auf entsprechende Datenbank-Relationen abgebildet. Darüber hinaus bietet Hibernate Mechanismen zur Kompatibilität mit verschiedenen Datenbanken. Die zum Datenbankzugriff erforderlichen SQL-Anweisungen werden nicht explizit in SQL programmiert, sondern von Hibernate in Abhängigkeit vom SQL-Dialekt der verwendeten Datenbank generiert. Anwendungsseitig kann Hibernate in Java-Applikationen und Servlet-Engines benutzt werden oder in einen Applikationsserver integriert werden[HB].

#### 4.1.8 jQuery

Bei jQuery handelt es sich um eine freie JavaScript- Bibliothek, die eine Vielzahl von Möglichkeiten bietet, die Objekte einer Webseite zu manipulieren. Bei einer Manipulation handelt es sich zum

Beispiel um eine Gruppierung von Elementen, oder das Ausrichten von Seiteninhalten. Ungefähr die Hälfte aller Internetseiten benutzt bereits jQuery (Stand Juli 2014) [JQ]. Allerdings setzt es das Vorhandensein von Java voraus. Die Plattformunabhängigkeit dieser Bibliothek unterstreicht die vielseitigen Einsatzmöglichkeiten von jQuery. Die Veröffentlichung, unter dem führenden Entwickler John Resig, fand im Jahr 2006 statt. Zu den Funktionen neben der Objektmanipulation gehören zum Beispiel das Content- Management- System, was auch bei Wordpress standardmäßig zum Einsatz kommt, Animationen und Effekt, Ajax- Funktionalitäten, sowie ein erweiterbares Event- System. Zusätzlich zu diesen Funktionalitäten lässt sich jQuery durch weitere Plug-Ins erweitern. Für den Einsatz bedarf es eines Grundgerüsts von HTML und/ oder PHP, in das der Script eingebunden wird [1].

#### **4.1.9 CSS**

Bei CSS, dem Cascading Style Sheet, handelt es sich um ein Werkzeug, mit dem es möglich ist, Webseiten auf die gewünschte Weise zu präsentieren. Durch CSS lassen sich Dokumente und deren Anordnungen einheitlicher darstellen. Der Inhalt einer HTML Seite wird durch das Einführen von CSS zwar im Programm-code geringer, aber nicht vom Kontext, den die Seite vermittelt. Dadurch lassen sich Seiten besser auf Fehler überprüfen und besser warten, sowie updaten. HTML wäre nicht in dem Umfang nutzbar, wie es CSS ist.

Durch das Cascading Style Sheet können jedem Element unterschiedliche Lay-out-Informationen übergeben werden, was sich in unterschiedlichen Formen, Schriftarten oder Größen widerspiegelt. Der Quelltext kann in einem separaten Dokument ausgelagert, angepasst und dokumentiert werden. Der Gedanke liegt darin, den Inhalt von der Darstellung zu trennen. Die Veröffentlichung fand am 10. Oktober 1994 durch Håkon Wium Lie in Chicago statt. Seit 2000 befindet sich CSS3 in Entwicklung und ist derzeit die aktuellste Version. Die Version wird in den Prototypen dieser Arbeit angewendet .Da der Anforderungskatalog eine mobile Ansteuerung der Webanwendung forderte, wird das Bootstrap CSS eingesetzt. Bootstrap ist frei verfügbar und ist ein Frontend, das eine Sammlung von Bibliotheken beinhaltet, die man mit kurzen Klassendefinitionen in die Webseite einbauen kann. [1]

## **5 Lösungsarchitektur und Umsetzung**

In diesem Abschnitt wird die Umsetzung der im Kapitel 2 genannten Anforderungen erläutert und beschrieben. Dabei wird auf die Programmierung im Front-End und im Back-End Bereich eingegangen. Die eingesetzten Methoden aus Kapitel 3 und deren Einsatz wird verdeutlicht und deren Funktionsweise im System beschrieben.

## 5.1 Visualisierung des System – und Programmaufbaus

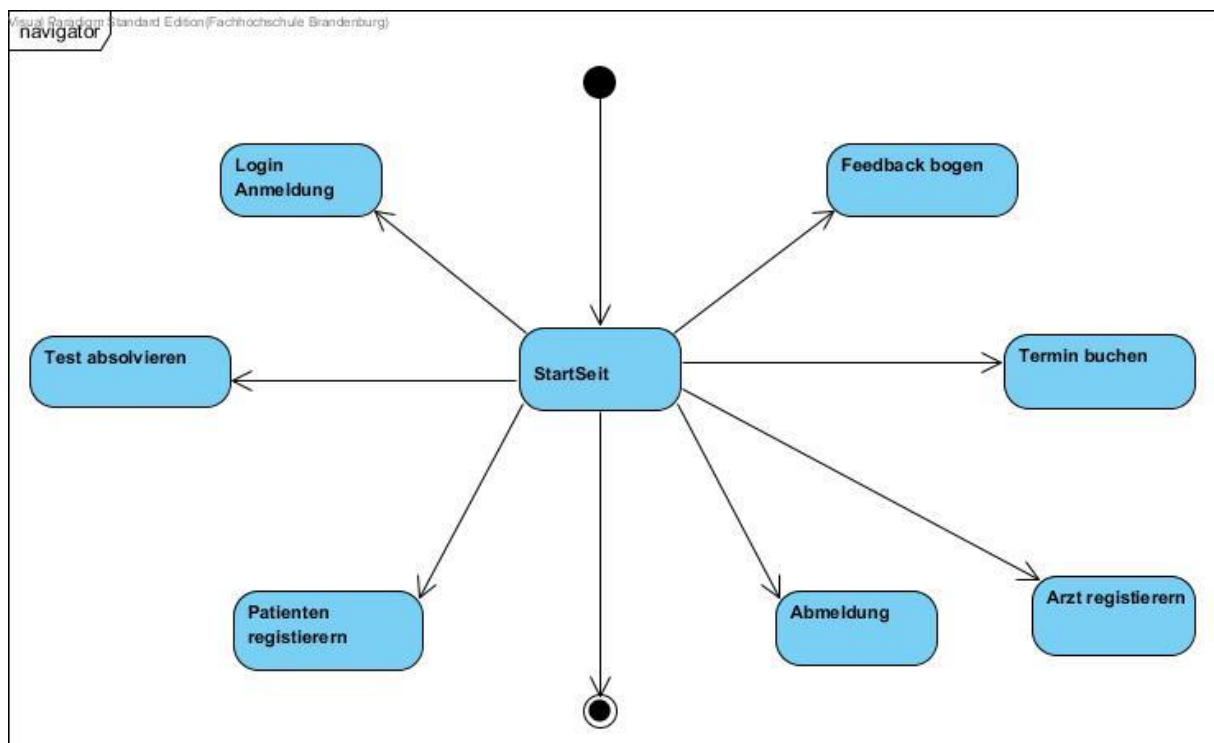


Abbildung 10. Startseite Erreichbarkeit

Die Folgende Abbildung zeigt die Darstellung der Hauptseite mit den integrierten Links zu den Informationsseiten, auf denen sich der Anwender informieren kann.

## 5.2 Client /Server

Der Begriff Client/Server beschreibt die Beziehung zwischen zwei Computer-Programmen. In dieser Beziehung fordert das eine Programm, der Client, einen Dienst beim anderen Programm, dem Server, an. Dieser erfüllt dann die Anforderung. Die Client/Server-Idee kann auch von Pro-

grammen innerhalb eines einzelnen Computers verwendet werden, wichtiger aber ist sie für Netzwerke. In einem Netzwerk stellt das Client/Server-Modell einen probaten Weg dar, um die Verbindung zwischen Programmen herzustellen, die aus Effizienzgründen an verschiedenen Orten liegen.

Computer-Transaktionen, die das Client/Server-Modell verwenden, sind weit verbreitet. Wenn Sie beispielsweise Ihren Kontostand von Ihrem Computer aus überprüfen möchten, dann sendet ein Client-Programm Ihre Anfrage an ein Server-Programm in der Bank. Dieses Programm leitet unter Umständen die Anforderung an ein eigenes Client-Programm weiter, das dann eine Anfrage an den Datenbank-Server sendet, um Ihren Kontostand abzurufen. Der Saldo wird zurück an den Bankdaten-Client gesendet, der diesen wiederum an Ihren Computer weiterleitet. Dieser zeigt Ihnen schließlich die Information an[cs].

Das Client/Server-Modell hat sich zu einer der zentralen Ideen des Network-Computing entwickelt. Die meisten geschäftlichen Anwendungen, die heutzutage entwickelt werden, verwenden es. Das gilt auch für das Hauptprogramm des Internets: TCP/IP[cs]. In der Sprache des Marketings wurde der Begriff anfangs zur Unterscheidung verteilter Rechenleistung in Form vieler kleinerer Computer von der „monolithischen“ zentralisierten Rechenleistung bei Mainframe-Computern verwendet. Aber diese Unterscheidung ist inzwischen weitestgehend verschwunden, da Mainframes und die zugehörigen Anwendungen mittlerweile ebenfalls das Client/Server-Modell nutzen und damit selbst ein Bestandteil des Network-Computing geworden sind.

## **5.3 Architekturstil :**

In diesem Abschnitt wird die Umsetzung der im Kapitel 2 genannten Anforderungen erläutert und beschrieben. Dabei wird auf die Programmierung im Front-End und im Back-End Bereich eingegangen. Die eingesetzten Methoden aus Kapitel 3 und deren Einsatz wird verdeutlicht und deren Funktionsweise im System beschrieben.

### **5.3.1 Klassifikation der Architektur (Back End):**

Nachdem die in Frage kommenden Architekturmuster kurz vorgestellt wurden, ist die Entscheidung gefallen, das Projekt auf der Basis des MVC- Model View Ansatzes zu realisieren. Zu dieser Entscheidung haben solche Merkmale von Controller View wie leichte Testbarkeit, sowie klare und überschaubare Kommunikationslogik beigetragen.

Das Projekt besteht aus Zwei Teile Client bzw., view w und Server, die Kommunikation zwischen der Client und Server enthält entsprechende Technologie RestFull Webservice.

Des Weiteren wurde das MVC-Muster als Client-Server-Architektur umgesetzt. Dabei wurde die gesamte Geschäftslogik in den Server ausgelagert und die View in den Client. Als Controlle wird eine eigene Kommunikationsbibliothek benutzt, die generisch gestaltet ist, um die wiederverwendbarkeit der funktionalen Klassen dieser Bibliothek zu erhöhen.

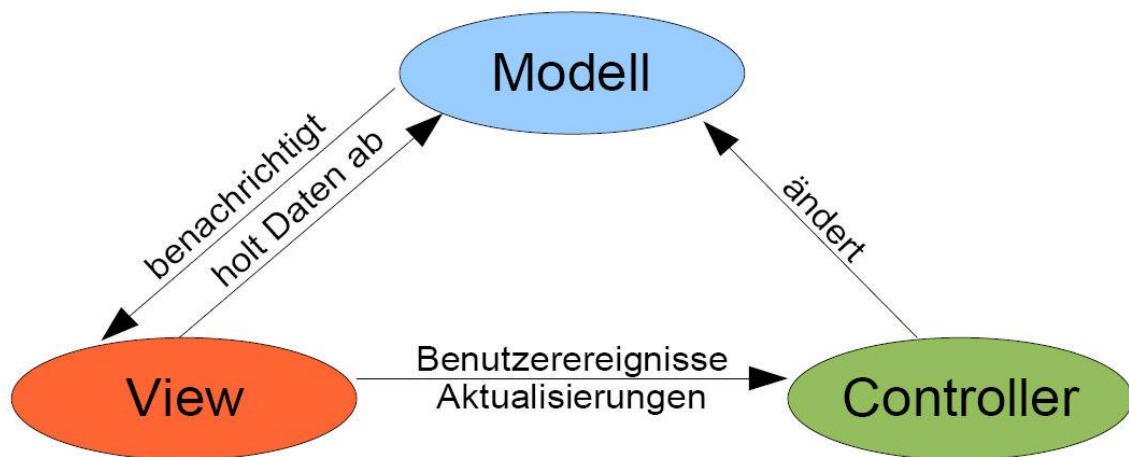


Abbildung 11. Architektur (MVC)

### 5.3.2 Klassifikation der Architektur (Font- End):

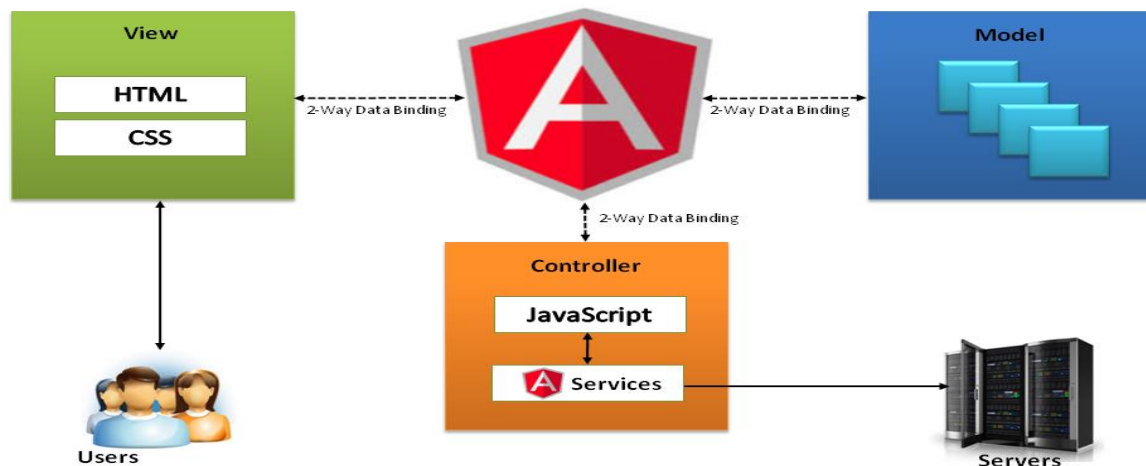


Abbildung 12. Architektur (MVC)[AG]

### Modell (Model).

Das Modell enthält die darzustellenden Daten (und in manchen Umsetzungen des MVC-Musters auch die Geschäftslogik). Es ist von Präsentation und Steuerung unabhängig. Die Bekanntgabe von Änderungen an relevanten Daten im Modell geschieht nach dem Entwurfsmuster „Beobachter“. Das Modell ist das zu beobachtende Subjekt, auch Publisher, also „Veröffentlicht“, genannt[WEB].

### Präsentation (View).

Die Präsentationsschicht ist für die Darstellung der benötigten Daten aus dem Modell und die Entgegennahme von Benutzerinteraktionen zuständig. Sie kennt sowohl ihre Steuerung als auch das Modell, dessen Daten sie präsentiert, ist aber nicht für die Weiterverarbeitung der vom Benutzer übergebenen Daten zuständig. Im Regelfall wird die Präsentation über Änderungen von Daten im Modell mithilfe des Entwurfsmusters „Beobachter“ unterrichtet und kann daraufhin die aktualisierten Daten abrufen. Die Präsentation verwendet oft das Entwurfsmuster „Kompositum[WEB]“.

### Steuerung (Controller).

Die Steuerung verwaltet eine oder mehrere Präsentationen, nimmt von ihnen Benutzeraktionen entgegen, wertet diese aus und agiert entsprechend. Zu jeder Präsentation existiert eine eigene Steue-

rung. Die Steuerung sorgt dafür, dass Benutzeraktionen wirksam werden, z. B. durch Änderung der Präsentation (z. B. Verschieben des Fensters) oder durch Weiterleiten an das Modell (z. B. Übernahme von Eingabedaten oder Auslösen von Verarbeitungen). Als es noch keine Objektorientierung gab, bestand ein Modell nur aus Daten, und die Steuerung hat die Daten oft direkt aktualisiert. In einer objektorientierten Umgebung ist es dagegen besser, wenn das Modell die Geschäftsobjekte enthält und die Steuerung sich darauf beschränkt, Benutzereingaben (Daten und Methodenaufrufe) weiterzuleiten, von der Präsentation an das Modell. Die Steuerung enthält weiterhin Mechanismen, um die Benutzerinteraktionen der Präsentation einzuschränken. Die Steuerung kann in manchen Implementierungen ebenfalls zu einem „Beobachter“ des Modells werden, um bei Änderungen der Daten die Präsentation direkt zu manipulieren[MVC].

## 6. Implementierung

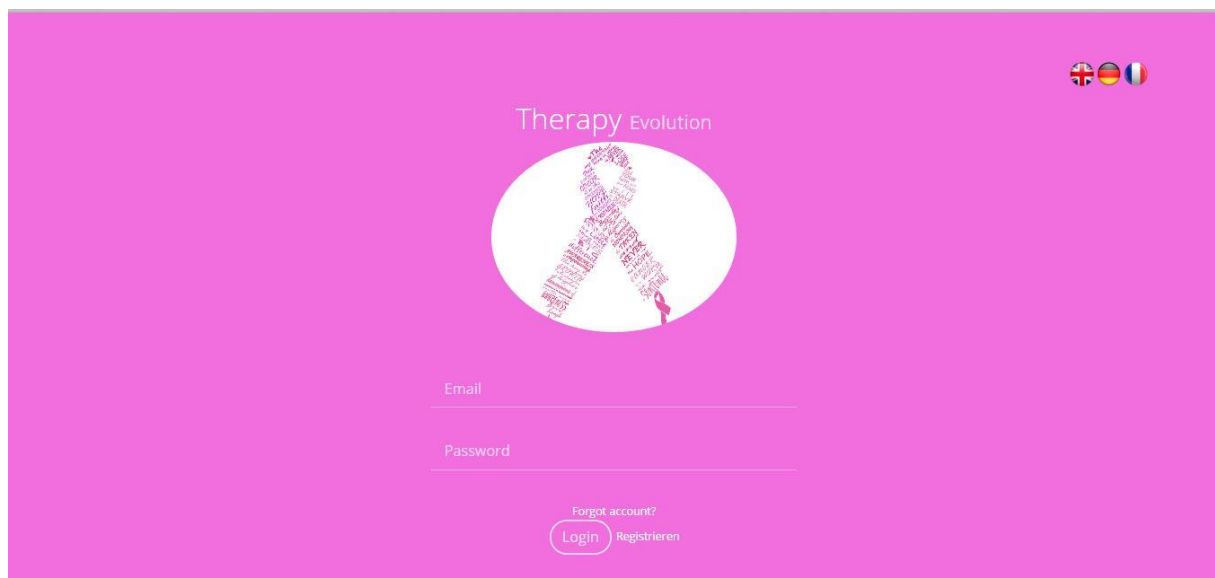
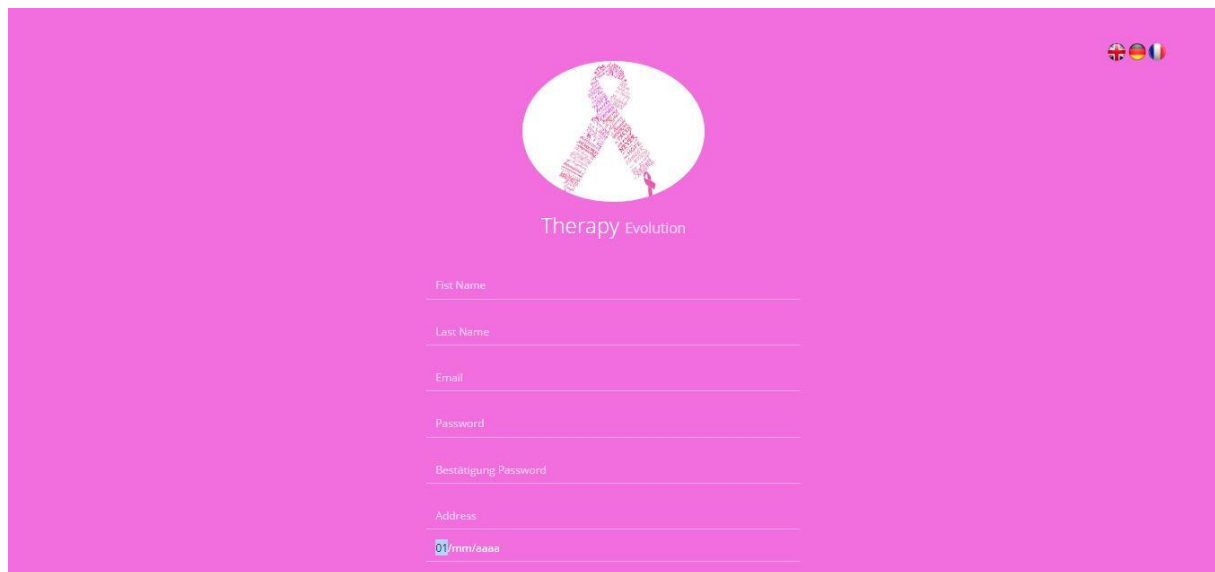


Abbildung 13. Login





The registration form is set against a solid magenta background. At the top center is a circular logo featuring a white ribbon made of text. To the right of the logo, three small flags (UK, Germany, France) are displayed. Below the logo, the text 'Therapy Evolution' is centered. The form consists of several input fields with labels to their left: 'First Name', 'Last Name', 'Email', 'Password', 'Bestätigung Password', 'Address', and a field with a placeholder '01/mm/aaaa'. Each field is a simple white line.

Therapy Evolution

First Name

Last Name

Email

Password

Bestätigung Password

Address

01/mm/aaaa

**Abbildung 14.Registregung**



The 'Find your account' form is on a light gray background. The title 'Finde dein Konto' is centered in a large, bold font. Below it, a subtitle reads 'E-Mail-Adresse, Telefonnummer, Benutzername oder vollständiger Name'. There is a single input field labeled 'Email Address' on the left. To its right are two buttons: a red one labeled 'suchen' and a gray one labeled 'Abbrechen'.

Finde dein Konto

E-Mail-Adresse, Telefonnummer, Benutzername oder vollständiger Name

Email Address

suchen Abbrechen

**Abbildung 15. Password vergessen**

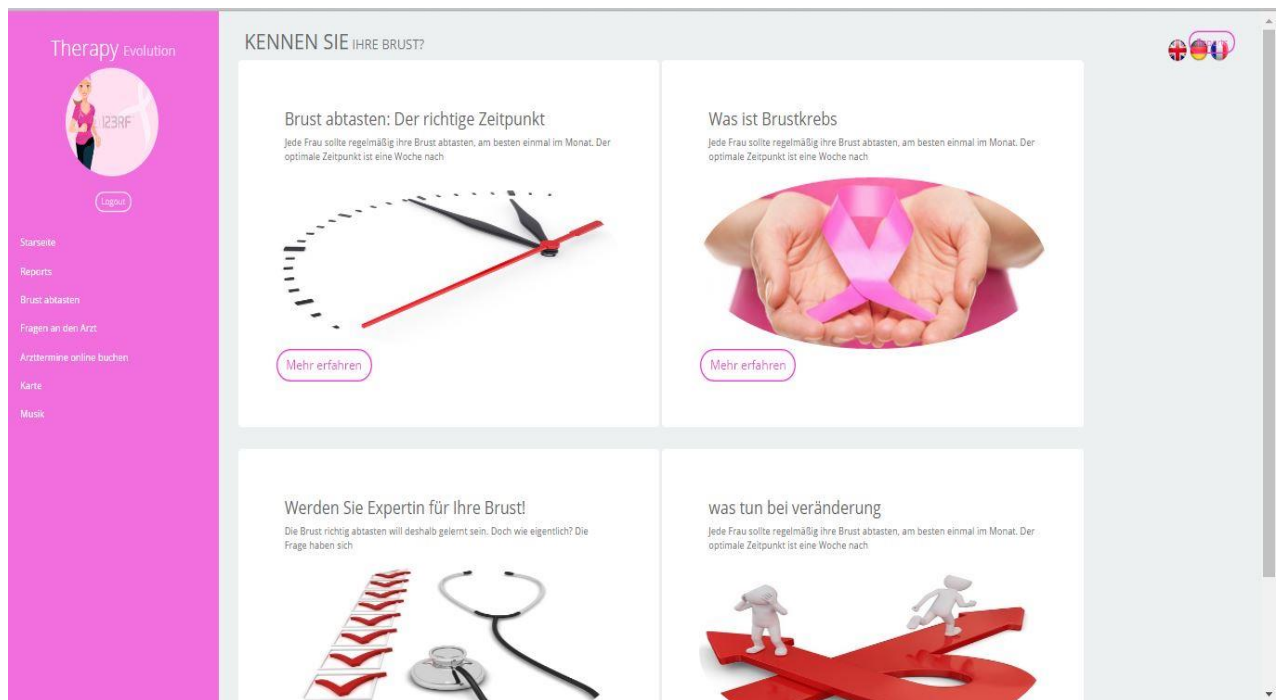


Abbildung 16. Startseite Erreichbarkeit



Abbildung 17. Seite Selbstuntersuchung

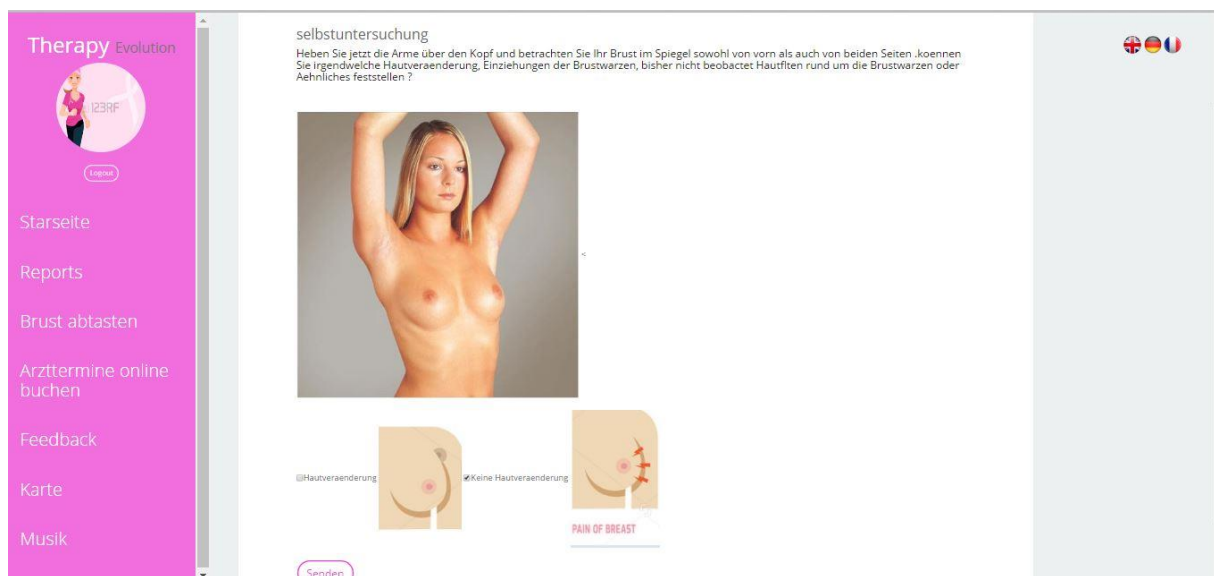


Abbildung 2. Seite Selbstuntersuchung

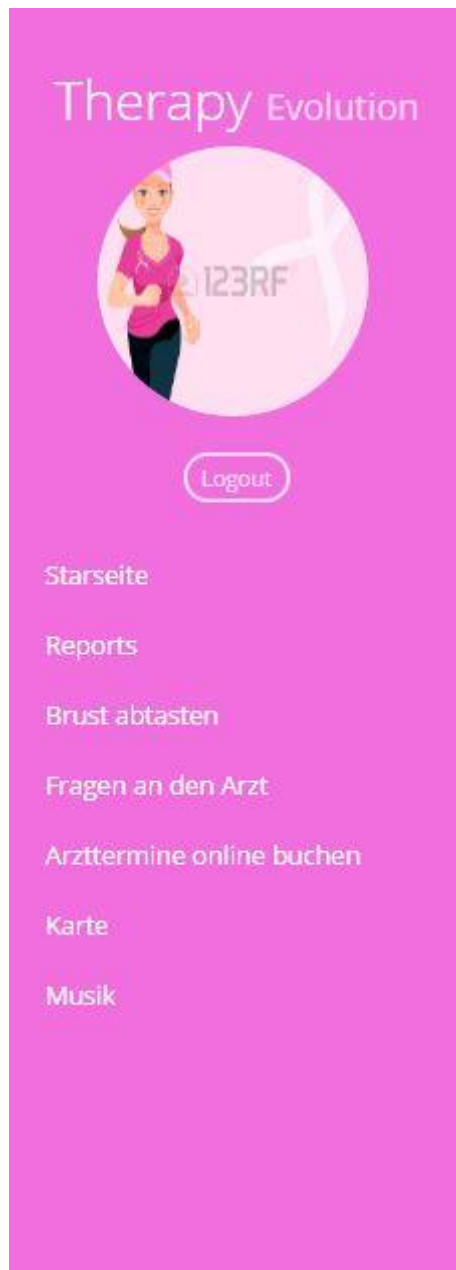


Abbildung 19. Navigation

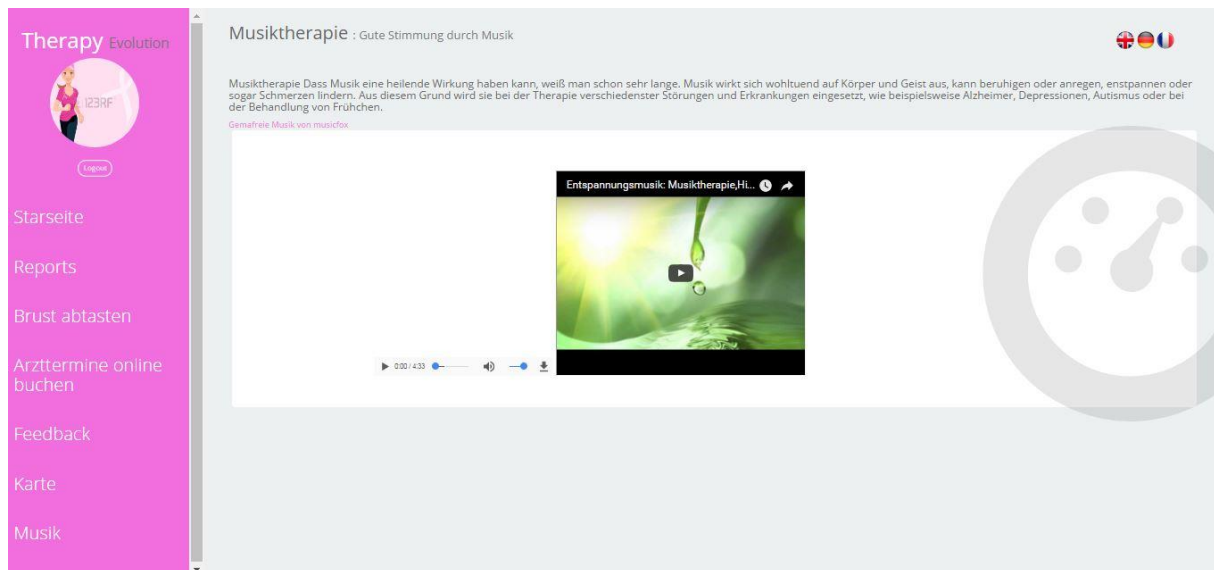


Abbildung 20. Musik Hören



Abbildung 21.Resultat des Testes

## 7. Technologie

Für das Projekt wurde ein Java-Server, Js-Client und ein Js-Client programmiert.

Die verwendete Programmiersprache bei dem Projekt ist Java.

Die Diagramme wurden mit Visual Paradigm mit der Version 13.0 erstellt. Das Dokument wurde mit Microsoft Word erstellt.

- Sublime
- Eclipse (Version 3.8)
- Wampserver
- Visual Paradigam
- GitHub

# Literaturverzeichnis

**[LJ15] Offizielle JQuery Seite [12.06.2016]**

<https://learn.jquery.com/about-jquery/>

**[FR] pädagogisch maturitätsschule kreuzlingen Maturaarbeit Okt 2012 [Aufruf: 05 01.01.2017]** [http://assets.krebsliga.ch/downloads/maturaarbeit\\_larissia\\_ullmann.pdf](http://assets.krebsliga.ch/downloads/maturaarbeit_larissia_ullmann.pdf)

**[AG] AngularJS API Docs [08/2016]**

<https://docs.angularjs.org/api>

**[BS] Bootstrap Docs**

<http://getbootstrap.com/>

**[SP] Spring Boot) Einführung [01/2017]**

<https://www.kade.de/start/>

**[CSS07] Eric A. Meyer CSS Das Umfassende Han**

2007 O'Reilly Verlag GmbH & Co. KG, 2 Auflage

**[HB]Framework Hibernat [01/2017]**

<http://hibernate.org/orm/documentation/5.0/>

**[MVC] Model View Controller**

[https://de.wikipedia.org/wiki/Model\\_View\\_Controller](https://de.wikipedia.org/wiki/Model_View_Controller)

**[App] Selbstuntersuchung der Brust**

<https://play.google.com/store/apps/details?id=de.kade.brustcheck>

**[\*] Bericht Masterarbeit Peter Jentsch**

**[PHZ] Tumorthherapie zwischen Nutzen und Schaden [01/2017]**

<http://www.pharmazeutische-zeitung.de/index.php?id=33021>

**[AG] Angular js modelviewControlle [01/2017]**

<https://avaldes.com/angularjs-introduction-and-sample-programs/>

<https://www.visual-paradigm.com/VPGallery/diagrams/Sequence.html>

(24.11.2016, 16:47 Uhr)

<https://www.visual-paradigm.com/tutorials/writingeffectiveusecase.jsp>

(29.11.2016, 13:45 Uhr)

<https://www.visual->

[paradigm.com/support/documents/vpuserguide/94/2575/6362\\_drawinguseca.html](https://www.visual-paradigm.com/support/documents/vpuserguide/94/2575/6362_drawinguseca.html)

**[CS] ClientServer [01/2017]**

<http://www.searchnetworking.de/definition/Client-Server>

**ProjektPlanung[PP]**

<https://www.tomsplanner.fr/?template=example>