



# Schriftliche Ausarbeitung

Dokumentation Anwendung “Digitaler Briefkasten”  
im Rahmen des Modul “AWE1”

Prüfer:

Christian Heuermann

Erstellt von:

Jonathan Brockhausen, Phillip Röring, Julius Figge

Studiengang:

Angewandte Informatik B.Sc.

Eingereicht am:

Wednesday 27<sup>th</sup> May, 2020

# Contents

<b>List of Figures</b>	<b>IV</b>
<b>List of Tables</b>	<b>V</b>
<b>Listingverzeichnis</b>	<b>VI</b>
<b>1 Installation</b>	<b>1</b>
<b>2 Anleitung</b>	<b>2</b>
<b>3 Fachkonzept</b>	<b>3</b>
3.1 Grundlagen . . . . .	3
3.2 Frameworks . . . . .	3
3.3 Datenbank . . . . .	4
<b>4 Entity-Relationship Diagramm</b>	<b>5</b>
<b>5 Security [Julius Figge]</b>	<b>7</b>
5.1 Aktive Security Bestandteile . . . . .	7
5.2 Passive Security Bestandteile . . . . .	7
<b>6 Test [Julius Figge]</b>	<b>9</b>
6.1 manuelle-“Klicktests” . . . . .	9
<b>7 Use-Cases [Julius Figge]</b>	<b>10</b>
<b>8 GUI-Konzept [Julius Figge]</b>	<b>12</b>
<b>9 Konzepte [Julius Figge]</b>	<b>15</b>
9.1 Arbeitskonzept . . . . .	15
9.2 MVC-Pattern . . . . .	15
9.3 Jackson-JSON . . . . .	15
<b>10 Projektplanung</b>	<b>16</b>
10.1 Projektstrukturplan . . . . .	16
10.2 Soll-Ist-Vergleich . . . . .	16
<b>Anhang</b>	<b>17</b>
<b>Quellenverzeichnis</b>	<b>35</b>



## List of Figures

Abbildung 1: ERD des Projekts . . . . .	5
Abbildung 2: Use-Case Diagramm . . . . .	10
Abbildung 3: Farben Konzept . . . . .	12
Abbildung 4: Ideen Konzept . . . . .	13
Abbildung 5: Rechtsklick Umsetzung . . . . .	14
Abbildung 6: Dropdown Umsetzung . . . . .	14
Abbildung 7: Administrator - Use-Case Diagramm . . . . .	20
Abbildung 8: Kontaktformular - Use-Case Diagramm . . . . .	21
Abbildung 9: GUI-Konzept - Login . . . . .	22
Abbildung 10: GUI-Konzept - Registrierung . . . . .	23
Abbildung 11: GUI-Konzept - Willkommen . . . . .	23
Abbildung 12: GUI-Konzept - Idee erstellen . . . . .	24
Abbildung 13: GUI-Umsetzung - Login . . . . .	25
Abbildung 14: GUI-Umsetzung - Registrierung . . . . .	26
Abbildung 15: GUI-Umsetzung - Ideen . . . . .	27
Abbildung 16: GUI-Umsetzung - Idee erstellen . . . . .	28
Abbildung 17: GUI-Umsetzung - Idee ansehen . . . . .	29
Abbildung 18: GUI-Umsetzung - Admin Ansicht . . . . .	30
Abbildung 19: GUI-Umsetzung - Spezialist Ansicht . . . . .	31
Abbildung 20: Projektstrukturplan . . . . .	32

## List of Tables

Tabelle 1: GUI-Testdurchführung . . . . .	18
---	----

## **Listingverzeichnis**

## **1 Installation**

## **2 Anleitung**



## 3 Fachkonzept

Im Folgenden werden die im Projekt verwendeten Technologien aufgeführt.

### 3.1 Grundlagen

#### Java 11

Zum Ziele der größten Kompatibilität sowohl mit den weiteren Technologien haben wir uns für Version 11 des Java Development Kits als grundlegende Java-Version entschieden.

#### Maven (Wrapper)

Um die notwendigen Dependencies des Projekts bereitzustellen nutzen wir Maven mit einem Wrapper. Maven erlaubt in der POM.XML-Datei die einfache und übersichtliche Verwaltung von Dependencies. Die wichtigsten Dependencies sind im nächsten Überabschnitt aufgeführt.

#### GitHub

GitHub ist die populärste und weitverbreiteste Plattform für Kollaboration auf Basis der Open-Source-Versionsverwaltung Git. Da wir auch alle drei mit der Plattform zumindest ansatzweise vertraut sind und die Integration in die verwendete IDE problemlos möglich ist, wird GitHub für das Projekt genutzt.

### 3.2 Frameworks

#### Springboot

Springboot wird genutzt um unseren Spring-basierten Code auf den Webserver zu bringen. Springboot ist ein in der Branche übliches Framework, um die Produktion von Enterprise-Web-Anwendungen zu vereinfachen. Die bereitgestellte Standardkonfiguration bietet eine gute Grundlage. Zusätzlich dazu wurden in den application.properties einige weitere Einstellungen vorgenommen, die beispielsweise den Port und die Adresse des Webserver angeben.

#### Thymeleaf

Kurzgesagt macht Thymeleaf HTML-Dateien intelligent. Das Framework erlaubt es uns, Informationen im Frontend sauber anzuzeigen und gleichzeitig sauber zu implementieren. Mithilfe von Thymeleaf vermeiden wir an einigen Stellen Konflikte oder Umwege. Weiterhin bietet Thymeleaf gute Integration mit Spring und dessen Security-Tools.

### 3.3 Datenbank

h2 ist eine auf Java basierte SQL-Datenbank-Engine. Die Verwendung von h2 für die Java-Anwendung macht die Einbindung der Datenstruktur direkt in Java möglich und macht somit eine externe Datenbank überflüssig. Wir haben uns bewusst gegen eine Lösung wie MSSQL oder MySQL entschieden, da diese bei weitem nicht so spurlos in Java integrierbar sind. Besonders die Vermeidung von hard-codetem SQL wollten wir in unserer Anwendung vermeiden. Trotzdem lässt h2 normale Auswertungen und Anbindungen sowie das Absetzen von SQL-Statements zu. Damit sind externe Zugriffe auf die Datenbank trotzdem umsetzbar.



der Auslieferung bereits vorhanden sind. Da interne Ideen gemäß der Anforderungen keine Produktsparte besitzen, bekommen Sie die dem Benutzer verborgene Produktsparte INTERNAL zugewiesen. Diese ermöglicht für interne Ideen dieselbe Logik zu verwenden. Durch diese Umsetzung ist auch eine Erweiterung um weitere Ideenkategorien ohne Änderungen am übrigen Programm möglich.

#### **Umsetzung von Status**

Wir haben uns dagegen entschieden, den Status in eine eigene Entität im Sinne des ERD auszulagern. Erweiterungen und Änderungen der Status im Echtbetrieb erfordern dann zwar unter Umständen an einigen Stellen Änderungen in der Programmlogik aber die enum bieten insgesamt Performancevorteile gegenüber der Auslagerung als vollwertige Entität und sind leichter im Code umzusetzen.

#### **Vererbung von Ideen zu interne und Produktidee**

Das Anlegen von Klassen mit Vererbung (wie im Projekt bei Ideen und Usern) kann in relationalen Datenbanken zu zwei Schwierigkeiten führen:

1. Lese- und Speicherzugriffe betreffen mehrere Tabellen und erfordern Joins. Das kann zu unübersichtlichen Strukturen und SQL-Kommandos sowie zu geringerer Performance führen
2. Nicht-polymorphe Abfragen (z.B. Namen/Preise nur der Getraenke) sind umständlich (z.B. Unterscheidung per Diskriminator)<sup>1</sup>

Durch die Verwendung von Hibernate mit seiner nativen Java-Integration und die Vermeidung von hard-codeten SQL-Statements im Code, fallen diese Punkte kaum ins Gewicht. Die Struktur der Vererbung ermöglicht es darüber hinaus sogar weitere Kategorien von Ideen anzulegen.

---

<sup>1</sup>vgl. Horn, Torsten (2007)

## 5 Security [Julius Figge]

Die Security der Anwendung wird durch mehrere Bestandteile sichergestellt, diese lassen sich in aktive und passive Elemente unterteilen.<sup>2</sup>

### 5.1 Aktive Security Bestandteile

Zuerst ist der Login sowie die Registrierung abgesichert. Nutzer müssen ihr Passwort mit mindestens 8 Zeichen wählen. Dieses wird im Backend, inklusive Salt, gehashed gespeichert. Hierzu benutzen wir BCrypt als Passwort Encoder. Diesen verwenden wir mit einer Stärke von 10, das bietet für uns die beste Balance zwischen Sicherheit und Performance. Mit dem Login bekommen Nutzer einen Cookie, in Form einer JSession ID, mit dem sie sich in weiteren Requests authentifizieren und über den sie identifiziert werden können.

Der nächste Bestandteil ist die URL-Zugriffskontrolle in der Klasse “SecurityConfig” im Package “config”. In dieser wird festgelegt welche Requests durch Spring Security zugelassen werden. Nicht authentifizierte Nutzer haben hier nur Zugriff auf statische Elemente (wie z.B. Grafiken, Javascript und CSS), die Registrierung und die Ideenansicht. Authentifizierte Nutzer werden anhand ihrer Rolle unterschieden, welche im Backend überprüft wird. Nutzer, Spezialisten und Administratoren können nur auf die jeweils für sie relevanten Seiten zugreifen. Der durch Spring Security erstellte JSession-Cookie wird beim ausloggen invalidiert und gelöscht.

Darüber hinaus ist die Anwendung so konfiguriert, dass ein automatischer Session Timeout nach 15 Minuten erfolgt, auch hierbei wird die Session invalidiert.

Außerdem werden alle Abfragen durch das Backend geprüft. An relevanten Stellen wird in den jeweiligen Controllern bereits vor der Bearbeitung des Requests die Rolle des aktuellen Users überprüft. Damit wird sichergestellt, dass Funktionen die insbesondere dem Administrator oder Spezialisten vorbehalten sind, nur durch diese durchgeführt werden können.

Außerdem werden übertragene Informationen in den bearbeitenden Services um die Berechtigung diese anzufragen, zu verändern oder zu speichern geprüft.

Durch diese Kontrolle an mehreren Stellen erreichen wir es zu kontrollieren und sicherzustellen welche Art von Requests ( un-, authentifiziert), welcher User, welcher Rolle, welche Daten wie verwenden (lesen, bearbeiten, schreiben) dürfen.

### 5.2 Passive Security Bestandteile

Zu den passiven Bestandteilen gehört das Loggen von Anmeldeversuchen, Anmeldung, Registrierung, und Abmeldung vom System.<sup>3</sup> Dies wird durch mehrere Klassen im Package “log” sichergestellt. Diese

---

<sup>2</sup>Zu beachten ist, dass wir das Programm unter der Prämisse entwickelt haben, dass im Livebetrieb eine zusätzliche SSL-Verschlüsselung für den Traffic genutzt wird.

<sup>3</sup>Das Loggen von Session Timeouts konnte aufgrund von Komplikationen zum Abgabezeitpunkt nicht fertiggestellt werden.

implementieren einen jeweiligen Application-Listener, beispielhaft für den fehlerhaften Login der ApplicationListener `AuthenticationFailureBadCredentialsEvent`. Beim auftreten eines passenden Applicationevents wird mit Hilfe eines Loggers, den “slf4j” bereitstellt der aktuelle Zeitstempel sowie Nutzernamen und IP-Adresse geloggt. Hierbei ist anzumerken, dass die Logs zusätzlich außerhalb der Konsole in eine Datei geschrieben werden. Diese ist auf 5Mb begrenzt und rotiert oberhalb dieser Grenze automatisch. Zudem sind die zu schreibenden Logs eingeschränkt. Damit stellen wir sicher, dass nur relevante Informationen festgehalten werden und diese auch unabhängig vom Programm zur Auswertung zur Verfügung stehen. Des weiteren sind Fehlermeldungen eingeschränkt um nicht aus versehen Informationen durchsickern zu lassen. Beispielhaft zeigt der Login ausschließlich eine Fehlermeldung bei fehlerhaften Daten an - jedoch nicht ob der Nutzernamen oder das Passwort falsch war. **Darüber hinaus werden Exceptions gefiltert und nur ausgewählte (respektive unsere eigenen) auf der Error-Seite angezeigt. Damit stellen wir sicher, dass nicht ausversehen Exceptions, Stacktraces oder Debug-Logs an das Frontend gelangen und für den Nutzer sichtbar sein könnten.**

## 6 Test [Julius Figge]

### 6.1 manuelle-“Klicktests”

Zur Überprüfung der “GUI” sollen manuelle Klicktests durchgeführt werden. Diese sollen dokumentiert werden um Fehler möglichst gezielt beheben zu können.

#### Zu notierende Informationen

Zu den notierenden Informationen gehören zum einen die Programmrevision (Git Commit Hash, Datum) sowie der verwendete Branch. Darüber hinaus ist das genutzte Betriebssystem sowie der genutzte Browser (inklusive Build zu notieren). Bei Darstellungsfehlern ist es sinnvoll zudem Screenshots zu hinterlegen sowie die Bildschirmauflösung zu notieren. Diese Informationen sammeln wir gezielt sehr detailliert um Fehler besser eingrenzen zu können.

#### Testvorbereitung

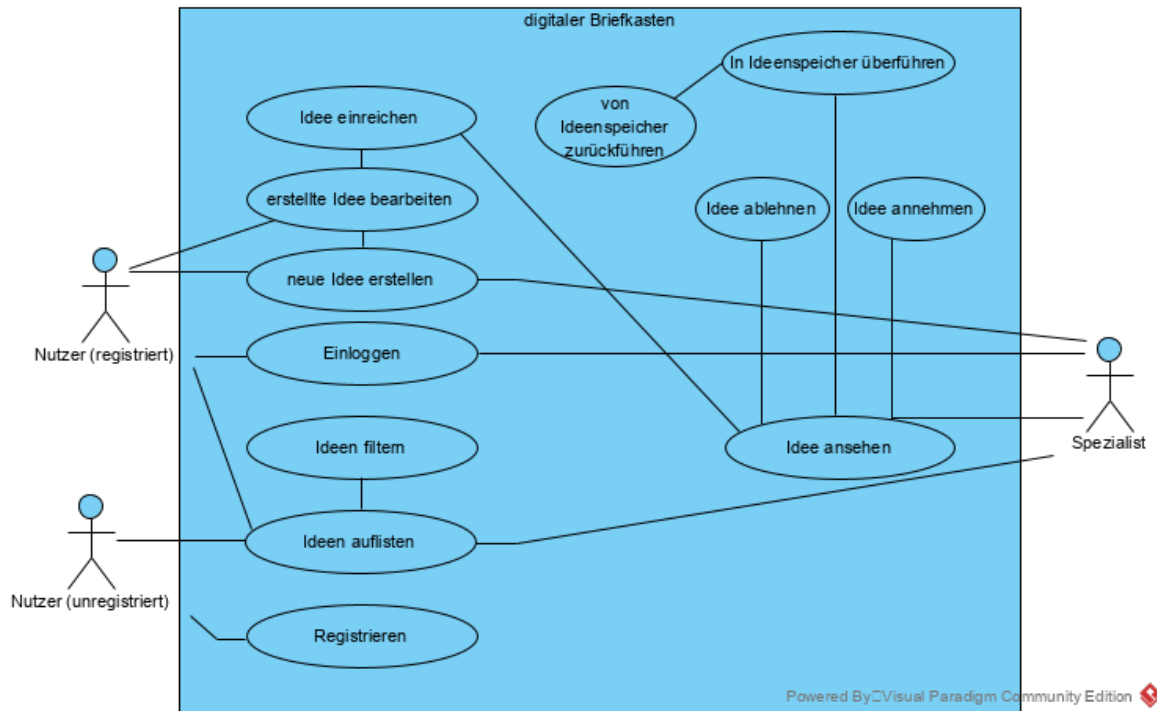
1. Zum Testen wird der neueste Stand des Master-Branches verwendet.
2. Hierzu ist zunächst die Datenbank zu löschen und mit Hilfe der in “HelperScriptsNoTests” vorhandenen Tests zu füllen.
3. Der Code soll kompiliert werden und die entstandene “Jar”-Datei ausgeführt werden.
4. Nach Möglichkeit soll der Test auf mehreren Browsern ausgeführt werden. Hierbei ist zu beachten, dass alle Addons zu deaktivieren sind, um eventuelle Komplikationen auszuschließen.
5. Die Entwicklerkonsole ist zu öffnen um hier entstehende Fehler und Warnungen mit in die Testergebnisse aufzunehmen.
6. Nachdem diese Voraussetzung geschaffen ist, sind die Tests durchzuführen und die obigen Informationen zu notieren.

Zur Testdurchführung ist die Tabelle im Anhang 1 auf S.18 zu verwenden.

## 7 Use-Cases [Julius Figge]

Im nachfolgenden sind die Use-Cases des Programm dargestellt (Siehe Abb. 2). Diese sind den Projektvorgaben entnommen.<sup>4</sup>

Figure 2: Use-Case Diagramm



Quelle: Eigene Darstellung

Für das Use-Case Diagramm sind drei Rollen von Relevanz. Zuerst der **unregistrierte Nutzer**, welche die Sicht des Programmes für die Öffentlichkeit repräsentiert. Des weiteren der **eingeloggte Nutzer** der mehr Möglichkeiten hat, hierzu gehört auch der **Administrator**. Dieser hat über die Möglichkeiten des Nutzers hinaus weitere administrative Rechte.<sup>5</sup> Jedoch besitzt er nicht die Rechte der dritten Rolle des **Spezialisten**.

Die Use-Cases lassen sich in zwei "Kern"-Kategorien unterteilen. Das sind zum einen die **Account** bezogenen Use-Cases.

Hierzu gehören der Vorgang des Einloggens sowie der Registrierung. Zu diesen ist anzumerken, dass Spezialisten sich lediglich einloggen können. Durch ihre administrative Rolle werden diese durch den Administrator angelegt.

Zum anderen ist der zweite Use-Case die **Erstellung und Bewertung von Ideen**.

Eingereichte Ideen lassen sich durch alle Nutzer jeder Rolle einsehen und filtern. Darüber hinaus haben

<sup>4</sup>Vgl. Heuermann, Christian (2020)

<sup>5</sup>Der Admin ist als eigener Use-Case im Anhang dargestellt. Siehe Anhang 2.1 auf S.20



alle eingeloggten Nutzer die Möglichkeit Ideen zu erstellen, zu bearbeiten und zur Bewertung einzureichen.

Diese eingereichten Ideen werden durch Spezialisten bewertet oder gespeichert.

Des weiteren existiert die Möglichkeit für alle Nutzer dem Administrator der Plattform über ein Kontaktformular Nachrichten zu senden.<sup>6</sup>

---

<sup>6</sup>Das zugehörige Use-Case Diagramm findet sich im Anhang 2.2 auf S.20

## 8 GUI-Konzept [Julius Figge]

Wir haben uns entschieden, statt eines GUI-Mockups unser GUI-Konzept direkt im Prototypen mit auszuliefern. Das lässt sich durch mehrere Punkte begründen. Zuerst hatten wir zum Zeitpunkt der ersten Präsentation bereits einen funktionierenden Prototypen und konnten diesen direkt mit dem GUI-Konzept ausstatten. Dadurch hatten wir nicht nur ein Mockup sondern konnten bereits mit der GUI interagieren. Des weiteren hatten wir dadurch die Möglichkeit die Zeit für die Erstellung eines Konzeptes direkt in die Entwicklung funktionierender GUI zu stecken.

Die GUI wurde unter Nutzung von Bootstrap 4 in Kombination mit Font-Awesome für die Icons entwickelt. Dadurch war es uns möglich eine konsistente, verständliche und klare Oberfläche zu entwickeln. Hierbei haben wir uns darauf konzentriert “eine klare Linie zu fahren”. Alle Seiten werden auf weißem Hintergrund dargestellt. Buttons und Informationen sind generell in grau ( beziehungsweise Schwarz) gehalten. Abweichend hiervon treten Farben nur auf um die Aufmerksamkeit des Nutzers auf sich zu ziehen oder um Hinweise hervorzuheben. Diese Farben sind in ?? abgebildet. Die Verwendung wird im weiteren näher erläutert.

**Figure 3:** Farben Konzept



**Quelle:** Eigene Darstellung

Grundlegend sind die Elemente der Anwendung zentriert wie im weiteren zu sehen. Damit erreichen wir in Kombination mit der Nutzung von Bootstrap eine nahezu 100 prozentige Kompatibilität zu mobilen Endgeräten.<sup>7</sup>

Grundlegende Elemente dieses Konzeptes sind zum einen der Login-Screen (Siehe ?? im Anhang auf S.??). In diesem Screenshot ist auch das Logo der Anwendung zu sehen welches ebenfalls in den typischen Farben gehalten wurde. Dieses soll der Anwendung einen Wiedererkennungswert geben durch seine gleichzeitig humorvolle als auch simple Darstellung.










Die zweite zentrale Komponente des Konzeptes ist die Übersicht aller Ideen (Siehe Abb. 4). Auffallend ist hier die Gliederung der Ideen in Tabellen. Bereits zu diesem Zeitpunkt war geplant die Ideen nach Typ zu gliedern und in einer Übersicht mit ihren wichtigsten Eigenschaften darzustellen. Zu diesem Zeitpunkt noch per Klick<sup>8</sup>, sollte es möglich sein die Idee im Detail inklusive aller Informationen darzustellen. Diese Entscheidung begründet sich damit das Gleichgewicht zwischen der verfügbaren Information auf einer Seitenansicht und der Übersichtlichkeit zu wahren. Darüber hinaus findet sich auch hier die Farbgestaltung wieder. Grundsätzlich ist die Oberfläche Monochrom gehalten. Icons dienen der schnelleren Identifikation der verschiedenen Tabellen und der Übersichtlichkeit. Farbakzente sind zum einen zur Führung der Nutzer gedacht, siehe beispielhaft in dem Hyperlink auf den Ersteller

<sup>7</sup>Hierbei ist jedoch anzumerken, dass dieses Feature nicht gefordert war und somit auch nicht weitergehend getestet wurde. Allerdings sind bereits die Voraussetzungen für eine mögliche Erweiterung der Anwendung geschaffen.

<sup>8</sup>Diese Funktionalität wurde im weiteren durch ein Rechtsklick Menü erweitert (Siehe 3.2 S.25).

der Ideen<sup>9</sup>. Zum anderen sind diese in den Status der Ideen mit einbezogen, hierdurch lässt sich erheblich schneller ein Überblick verschaffen.

**Figure 4:** Ideen Konzept

 Digitaler Briefkasten         Home Create Idea List Ideas <span>Logout</span>				
 Not submitted Ideas				
title	description	creator	creation date	status
konzept 1	Testbeschreibung	konzept	2020-05-20	 not submitted
 Product-Ideas				
title	description	creator	creation date	status
weitere Idee	Testbeschreibung	konzept	2020-05-20	 pending
Geht nicht junge	geheime Nachricht	konzept	2020-05-20	 accepted
 Internal-Ideas				
title	description	creator	creation date	status
weitere Idee	Testbeschreibung	konzept	2020-05-20	 pending
Geht nicht junge	geheime Nachricht	konzept	2020-05-20	 accepted

**Quelle:** Eigene Darstellung

Ein weiterer relevanter Punkt der sich beispielhaft in dieser Abbildung (Siehe Abb. 4) findet ist die Navigationsleiste. Diese ist im Konzept nur nach dem Login vorhanden. Im fertigen Produkt wurde diese aber auf jeder Seite inkludiert.<sup>10</sup> Diese ist zentrales Steuerelement der Anwendung. Auf der linken Seite findet sich das Logo dauerhaft präsent wieder. Daneben werden zur Verfügung stehende Seiten angezeigt, wobei die aktuelle hervorgehoben ist.<sup>11</sup> Auf der rechten Seite findet sich der Logout Button, auch dieser ist hervorgehoben um vom Nutzer wahrgenommen zu werden.

Die weiteren Konzeptteile der GUI finden sich im Anhang 3.1 auf S.22.

Hervorzuheben ist, dass gegenüber des Konzeptes in der Umsetzung<sup>12</sup> einige Elemente hinzugekommen sind. Die wichtigsten hierbei sind das bereits genannte Rechtsklick Menü (Siehe Abb. 5). Sowie intelligente Dropdown Menüs (Siehe Abb. 6). Diese sollen dem Nutzer die Möglichkeit geben intuitiv die Anwendung zu bedienen und erweitern diese durch dynamische Menüs welche sich in die Oberfläche einpassen.


<sup>9</sup>Dieser Hyperlink stand beispielhaft für die Weiterleitung auf eine Detailseite auf der Tabellensicht.

<sup>10</sup>Ebenso wurde ein Footer eingefügt. Vgl. 3.2 S.25

<sup>11</sup>Im fertigen Produkt ist diese Sicht abhängig von den verschiedenen Rollen. Vgl. 3.2 S.25


<sup>12</sup>Siehe Anhang 3.2 auf S.25

**Figure 5:** Rechtsklick Umsetzung

 Deine nicht eingereichten Ideen

Filtern..

	Typ	Titel	Beschreibung	Ersteller	Erstellungsdatum	Status
⋮	💡	Geheime Weltherrschaft	Na was denn sonst	Umse tzung	20.05.2020	🔗 Nicht eingereicht



- Anzeigen
- Bearbeiten
- Einreichen
- Löschen

Quelle: Eigene Darstellung

**Figure 6:** Dropdown Umsetzung

Vertriebskanal auswählen:

× Versicherungsmakler
× Kooperation mit Kreditinstituten

Stationärer Vertrieb  
**Versicherungsmakler**  
Kooperation mit Kreditinstituten  
Direktversicherung  
bis zu drei weitere eingeben:

Quelle: Eigene Darstellung

## 9 Konzepte [Julius Figge]

### 9.1 Arbeitskonzept

Unsere Team Arbeit haben wir auf den Austausch untereinander ausgerichtet. So konnten wir uns unsere unterschiedlichen Kompetenzen zu Nutze machen und haben beispielhaft im Mob-Programming<sup>13</sup> Wissen vermittelt und uns gegenseitig unterstützt.

Darüber hinaus haben wir nach dem “All hands on Deck”-Prinzip<sup>14</sup> gearbeitet.

Das haben wir zum einen aufbauend auf Teaminterne Kommunikation (beispielhaft über Telegram), aber insbesondere auch über die “Github CI” erreicht. Diese war konfiguriert bei jedem Commit alle Tests durchzuführen und bei Problemen Email-Benachrichtigungen zu versenden.

Ausserdem haben wir “Sonarlint” eingesetzt um unsere Code-Qualität zu überprüfen und stetig zu verbessern. Dadurch haben wir nicht nur unsere Code Qualität sichergestellt, sondern konnten auch auftretenden Probleme möglichst schnell erkennen und beheben.

Als netter Nebeneffekt lässt sich festhalten, dass durch das gemeinsame arbeiten Wissensilos effektiv aufgebrochen wurden und der Lerneffekt im Zuge des Projekts für alle beteiligten maximiert wurde.

### 9.2 MVC-Pattern

In unserer Anwendung benutzen wir das Architekturmuster Model View Controller. Dieses Muster haben wir explizit ausgewählt, da Springboot zusammen mit Thymeleaf als Frontend hierfür sehr gut geeignet ist. Dadurch erreichen wir eine strikte Trennung der verschiedenen Ebenen und eine bessere Anpassbarkeit des Programmes.

### 9.3 Jackson-JSON

Wir haben uns für die Nutzung der “Jackson”-JSON library entschieden unter anderem, da diese Annotations mitliefert welche wir direkt in unseren Code einbinden können. Diese benutzen wir um die Serealisierung und Deserealisierung von Datenbank-Einträgen zu verwalten. Dadurch ist es einfach und gut im Code lesbar möglich zu kontrollieren welche Einträge wie serealisiert werden. Das ist insbesondere für die Entwicklung der API relevant.

---

<sup>13</sup>Hiermit ist das gemeinse Programmieren über ein Videotelefonat gemeint, bei dem abwechselnd eine Person den Bildschirm teilt.

<sup>14</sup>Dieses bezeichnet den Ansatz, bei auftretenden Problemen und Fehlern sich zuerst auf die Behebung dieser zu konzentrieren, bevor weitergehende Aufgaben bearbeitet werden.

## 10 Projektplanung

### 10.1 Projektstrukturplan

Das Projekt wurde von uns in vier Phasen aufgeteilt, *Vorbereitung, Implementierung, Dokumentation & Tests* und *Abschluss*. Der Projektstrukturplan ist im Anhang auf Seite 32 dargestellt.

### 10.2 Soll-Ist-Vergleich

Der vor dem Projekt von uns festgelegte und in der Präsentation des Fachkonzepts vermittelte Soll-Zustand ist die vollständige Umsetzung der Muss-Features und die in der angegebenen Reihenfolge begonnene Umsetzung der Kann-Features. Im Anhang auf Seite 33 ist die Übersicht der Features dargestellt. Alle Soll-Features wurden anforderungsgemäß umgesetzt. Die Umsetzung der Kann-Features wurde gemäß der im Fachkonzept vorgestellten Priorisierung begonnen. Im Einklang mit dem gesamten Projekt wurde bei allen Features darauf geachtet, dass sie gut erweiter- und wartbar sind.

## Anhang

### Anhangsverzeichnis

Anhang 1:	Testdurchführung <a href="#">[Julius Figge]</a> . . . . .	18
Anhang 2:	Weitere Use-Cases <a href="#">[Julius Figge]</a> . . . . .	20
Anhang 2.1:	Administrator . . . . .	20
Anhang 2.2:	Kontaktformular . . . . .	20
Anhang 3:	GUI-Konzept <a href="#">[Julius Figge]</a> . . . . .	22
Anhang 3.1:	Konzept . . . . .	22
Anhang 3.2:	Umsetzung . . . . .	25
Anhang 4:	Projektplanung . . . . .	32
Anhang 4.1:	Projektstrukturplan . . . . .	32
Anhang 4.2:	Soll-Ist-Vergleich Muss- und Kann-Features . . . . .	33

**Anhang 1 Testdurchführung [Julius Figge]****Table 1:** GUI-Testdurchführung

Aktion	erwartetes Ergebnis	Reaktion
<b>Registrieren eines neuen Nutzers</b>		
Bereits bestehenden Nutzernamen verwenden (admin)	Fehlermeldung - Nutzernamen existiert bereits	
zu kurzer Nutzernamen (<3)	Fehlermeldung - Daten falsch	
zu kurzes Passwort (<=7)	Fehlermeldung - zu kurzes Passwort	
nicht übereinstimmende Passwörter	Fehlermeldung - nicht stimmende Passwort	
mit korrekten Daten	eingeloggt sein	
<b>Ausloggen aus dem Account</b>		
	ausgeloggt sein	
<b>Einloggen in erstellten Account</b>		
mit falschem Passwort	Fehlermeldung	
mit falschem Nutzernamen	Fehlermeldung	
mit richtigem Passwort	eingeloggt sein	
<b>Erstellen von beispielhaften Ideen</b>		
Erstellen einer "internen Idee"	Idee erscheint in Tabelle nicht eingereichter Ideen	
Erstellen einer "Produkt-Idee"	Idee erscheint in Tabelle nicht eingereichter Ideen	
Erstellen einer beliebigen Idee mit fehlerhaften Werten	Fehlerhafte Attribute werden hervorgehoben	
Erstellen einer Idee von der bereits selber Name bei selbem Typ vorhanden	Fehlermeldung über Duplikat	
Bearbeiten der internen Idee	Änderungen werden übernommen	
Bearbeiten der Produkt-Idee	Änderungen werden übernommen	
<b>Löschen von beispielhaften Ideen</b>		
Erstellen einer Idee	Idee erscheint in Tabelle nicht eingereichter Ideen	
Löschen der erstellten Idee	Idee wird ohne Fehlermeldungen gelöscht	
<b>Ideenübersicht</b>		
Filtern der nicht eingereichten Ideen nach Attributen	nur Ideen mit passenden Attributen werden angezeigt	

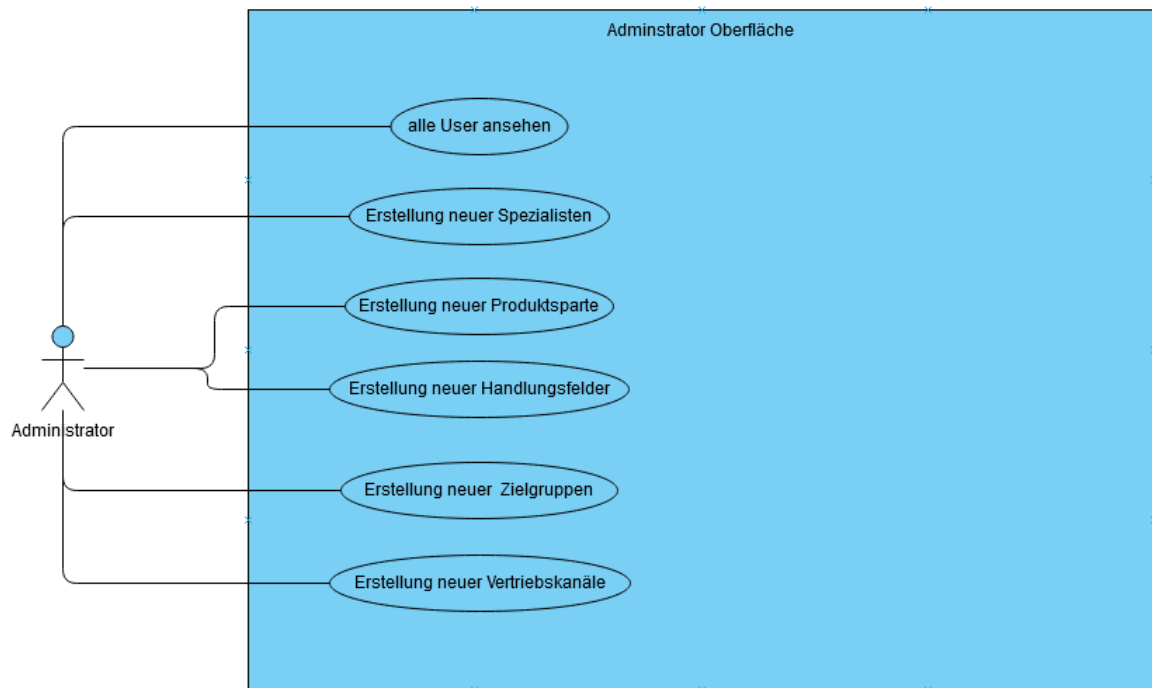


Einreichen der erstellten Ideen	erfolgreicher Transfer in jeweilige Tabelle	
<b>Ausloggen aus dem Account</b>	ausgeloggt sein	
<b>Idee Übersicht als nicht eingelogger Nutzer</b>		
Filtern der Ideen in beiden Tabellen	nur Ideen mit passenden Attributen werden angezeigt	
<b>Spezialist für “internen Idee”</b>		
Einloggen als passender (Ideen sollten ihm zugewiesen sein) Spezialist (Zugangsdaten siehe <code>Manual.md</code> )	eingeloggt sein	
Übersicht zu entscheidender Ideen filtern	nur Ideen mit passenden Attributen werden angezeigt	
Entscheiden ohne Begründung	fehlendes Attribut wird hervorgehoben	
Idee in Ideenspeicher verschieben	Idee liegt in Ideenspeicher	
<b>Spezialist für “Produkt-Idee”</b>		
Account zu anderem Spezialist wechseln	Idee liegt in Ideenspeicher	
Entscheiden über Idee aus Ideenspeicher mit Auswahl “zur Entscheidung freigegeben”	Idee liegt in eigenen zu entscheidenden Ideen	
Idee aus Entscheidungsübersicht bewerten	Idee erscheint auf passender Tabelle in Ideenübersicht	
<b>Administrator</b>		
Account zu Administrator wechseln (Zugangsdaten siehe <code>Manual.md</code> )		
Existierende User prüfen	registrierter Account sowie alle Spezialisten werden aufgelistet	
Alle möglichen anzulegenden Felder durchgehen, bereits bestehenden Namen eingeben	bei jedem Feld wird ein Fehler angezeigt	
Alle möglichen anzulegenden Felder durchgehen	Feld wird angelegt	
Ausloggen	<b>FERTIG!</b>	

## Anhang 2 Weitere Use-Cases [Julius Figge]

### Anhang 2.1 Administrator

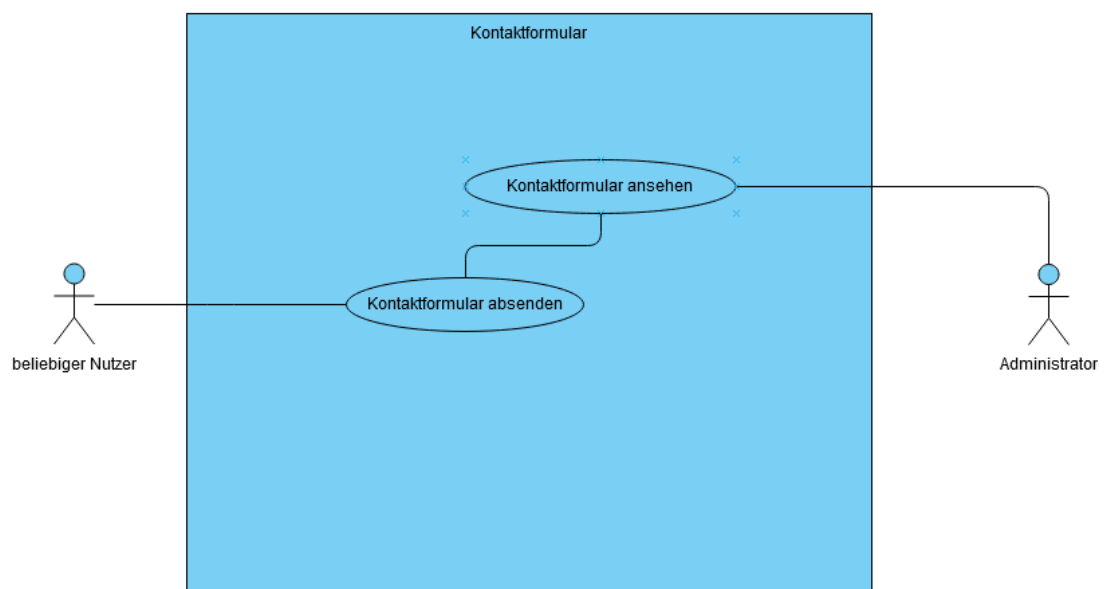
Figure 7: Administrator - Use-Case Diagramm



Quelle: Eigene Darstellung

### Anhang 2.2 Kontaktformular

**Figure 8:** Kontaktformular - Use-Case Diagramm



**Quelle:** Eigene Darstellung

## Anhang 3 GUI-Konzept [Julius Figge]

### Anhang 3.1 Konzept

Figure 9: GUI-Konzept - Login



Digitaler  
Briefkasten

Username

Password

Log in

Create an account

Quelle: Eigene Darstellung

**Figure 10:** GUI-Konzept - Registrierung

The registration form features a logo at the top consisting of a stylized mailbox with a Wi-Fi signal icon above it. Below the logo, the text 'Digitaler Briefkasten' is centered. The form contains three input fields: 'Username', 'Password', and 'Password Confirmation', each with a light gray border. At the bottom is a dark gray 'Register' button with white text.

**Quelle:** Eigene Darstellung

**Figure 11:** GUI-Konzept - Willkommen

The welcome screen has a dark gray header bar. On the left, it contains a logo and the text 'Digitaler Briefkasten', followed by navigation links 'Home', 'Create Idea', and 'List Ideas'. On the right is a red 'Logout' button. The main content area has a light gray background with the text 'Welcome konzept!' in a large font, and a smaller line of text below it: 'This is just a Prototype - so please be aware of that.' The bottom half of the screen is a large white rectangular area.

**Quelle:** Eigene Darstellung

**Figure 12:** GUI-Konzept - Idee erstellen

The screenshot shows a web application interface for creating a new idea. At the top, there is a dark navigation bar with the text "Digitaler Briefkasten" and a home icon, followed by links for "Home", "Create Idea", and "List Ideas". A "Logout" button is located in the top right corner. The main content area is titled "Create new Idea" and contains two input fields: "title" and "description". Below these fields is a dark button labeled "Create Draft".

Digitaler Briefkasten Home Create Idea List Ideas Logout

### Create new Idea

title

description

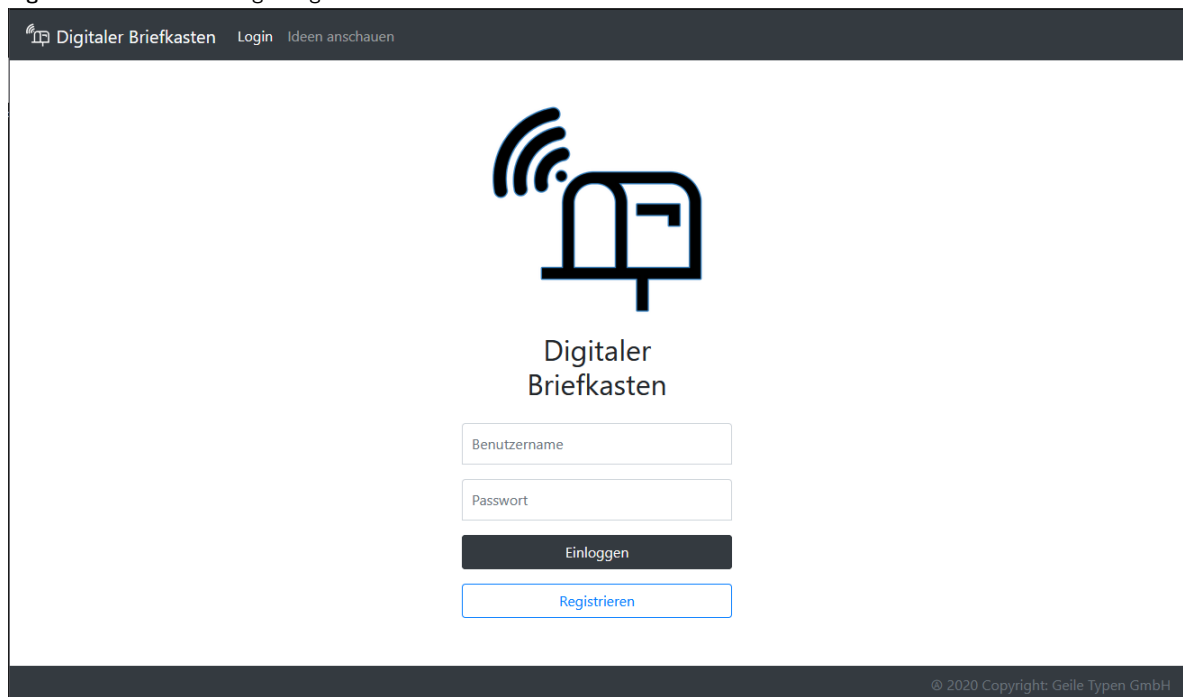
Create Draft

**Quelle:** Eigene Darstellung

## Anhang 3.2 Umsetzung

Die im folgenden dargestellten GUI Bestandteile stellen die wichtigsten Teile der Oberfläche dar. Auf die Abbildung aller Bestandteile wurde aufgrund der zu großen Menge, zur Wahrung der Übersichtlichkeit, verzichtet.

**Figure 13:** GUI-Umsetzung - Login



Digitaler Briefkasten Login Ideen anschauen

Digitaler Briefkasten

Benutzername

Passwort

Einloggen


Registrieren

© 2020 Copyright: Geile Typen GmbH

Quelle: Eigene Darstellung

**Figure 14:** GUI-Umsetzung - Registrierung

Digitaler Briefkasten   Login   Ideen anschauen



Digitaler  
Briefkasten

test

nur

show

.....

Passwort bestätigen

Registrieren

© 2020 Copyright: Geile Typen GmbH

**Quelle:** Eigene Darstellung



Figure 15: GUI-Umsetzung - Ideen

Digitaler Briefkasten
 Home
 Idee erstellen
 Ideen anschauen
 Logout

Deine nicht eingereichten Ideen
 

	Typ	Titel	Beschreibung	Ersteller	Erstellungsdatum	Status
⋮	💡	Geheime Weltherrschaft	Na was denn sonst	Umsetzung	20.05.2020	ⓘ Nicht eingereicht

Produkt-Ideen
 

	Titel	Beschreibung	Ersteller	Erstellungsdatum	Status	Status Begründung
⋮	<a href="#">Eine weitere</a>	Geheimer Test	Umsetzung	20.05.2020	🕒 auf Bewertung wartend	

Interne-Ideen
 

	Titel	Beschreibung	Ersteller	Erstellungsdatum	Status	Status Begründung
⋮	<a href="#">Beispiel</a>	Test	Umsetzung	20.05.2020	🕒 auf Bewertung wartend	

[ Nutzer: Umsetzung , Rolle: Nutzer ]
 [Kontaktformular](#)
 © 2020 Copyright: Geile Typen GmbH

Quelle: Eigene Darstellung

**Figure 16:** GUI-Umsetzung - Idee erstellen

**Digitaler Briefkasten** Home Idee erstellen ▾ Ideen anschauen Logout

### Neue Produktidee erstellen

Titel

Beschreibung

Zielgruppe auswählen:

Vertriebskanal auswählen:

Produktsparte auswählen:

KFZ

☐ Existiert bereits Vergleichbares (bei der Konkurrenz)?

Bis zu drei Vorteile eingeben:

Vorteil

Vorteil

Vorteil

Produkt-Idee erstellen

[ Nutzer: Umsetzung , Rolle: Nutzer ] [Kontaktformular](#) © 2020 Copyright: Geile Typen GmbH

**Quelle:** Eigene Darstellung

Figure 17: GUI-Umsetzung - Idee ansehen

**Digitaler Briefkasten** Home Idee erstellen ▾ Ideen anschauen Logout

### 💡 Geheime Weltherrschaft

Titel	Geheime Weltherrschaft
Beschreibung	Na was denn sonst
Produktlinie	KFZ
Vorteile	Nenene
Zielgruppen	Paare
Vertriebskanäle	Versicherungsmakler
Existiert bereits Vergleichbares?	<input checked="" type="checkbox"/>
Ersteller	Umsetzung
Erstellungsdatum	20.05.2020
Status	Nicht eingereicht
Status-Begründung	

[ Nutzer: Umsetzung , Rolle: Nutzer ] [Kontaktformular](#) © 2020 Copyright: Geile Typen GmbH

Quelle: Eigene Darstellung

**Figure 18:** GUI-Umsetzung - Admin Ansicht

The screenshot displays the Admin View of a web application. At the top, a dark navigation bar contains the following elements from left to right: a mail icon, the text 'Digitaler Briefkasten', 'Adminpanel', a dropdown menu with 'Idee erstellen' and 'Ideen anschauen', and a red 'Logout' button. Below the navigation bar is a large light gray banner with the text 'Willkommen admin!'. The main content area features a series of light gray boxes. The first box contains a link 'Alle registrierten User ansehen'. The second box contains a link 'Fachspezialisten anlegen'. Below this link is a form for creating a specialist, consisting of input fields for 'Username', 'Vorname', 'Nachname', 'Passwort', and 'Passwort bestätigen'. Below these fields is a label 'Fachgebiet auswählen:' followed by a dropdown menu. At the bottom of this form is a dark gray button labeled 'Spezialist anlegen'. Below the form are four more light gray boxes, each containing a link: 'Produktparte anlegen', 'Handlungsfeld anlegen', 'Zielgruppe anlegen', and 'Vertriebskanal anlegen'. At the bottom of the page is a dark footer bar containing the text '[ Nutzer: admin , Rolle: Administrator ]', a link 'Kontaktformular', and the copyright notice '© 2020 Copyright: Geile Typen GmbH'.

**Quelle:** Eigene Darstellung

Figure 19: GUI-Umsetzung - Spezialist Ansicht

The screenshot displays a web application interface for a 'Spezialist' (Specialist) view. At the top, a dark navigation bar contains the text 'Digitaler Briefkasten' with a mail icon, followed by links: 'Ideen bewerten', 'Idee erstellen' (with a dropdown arrow), and 'Ideen anschauen'. A red 'Logout' button is on the right. Below the navigation bar, a large light gray banner displays the welcome message 'Willkommen SpeziusMaximus\_KFZ!'. The main content area is divided into two sections. The first section, 'Anstehende Entscheidungen' (Pending Decisions), is marked with a hammer icon and contains a table with one data row. The second section, 'Ideen im Speicher' (Ideas in Storage), is marked with a floppy disk icon and contains an empty table. The footer shows the user information '[ Nutzer: SpeziusMaximus\_KFZ, Rolle: Spezialist ]', a link to 'Kontaktformular', and the copyright notice '© 2020 Copyright: Geile Typen GmbH'.

**Anstehende Entscheidungen**

	Typ	Titel	Beschreibung	Ersteller	Erstellungsdatum	Status
⋮	💡	<a href="#">Eine weitere</a>	Geheimer Test	Umse tzung	20.05.2020	🕒 auf Bewertung wartend

**Ideen im Speicher**

	Typ	Titel	Beschreibung	Ersteller	Erstellungsdatum	Status
--	-----	-------	--------------	-----------	------------------	--------

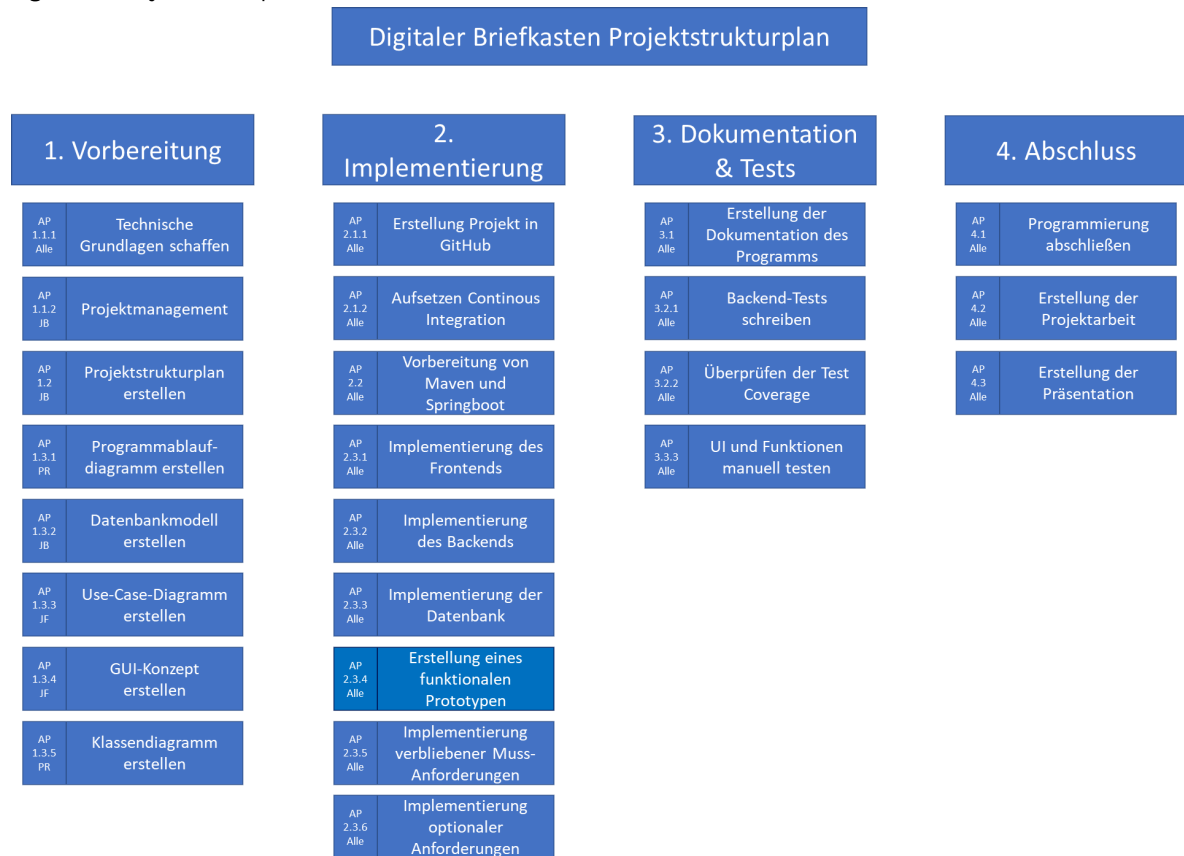
[ Nutzer: SpeziusMaximus\_KFZ, Rolle: Spezialist ] [Kontaktformular](#) © 2020 Copyright: Geile Typen GmbH

Quelle: Eigene Darstellung

## Anhang 4 Projektplanung

### Anhang 4.1 Projektstrukturplan

Figure 20: Projektstrukturplan



Quelle: Eigene Darstellung

**Anhang 4.2 Soll-Ist-Vergleich Muss- und Kann-Features**

	Anforderung	Umsetzung
Muss	Noch nicht registrierte Mitarbeiter können sich am System registrieren	Umgesetzt
Muss	Registrierte Mitarbeiter können sich am System anmelden	Umgesetzt
Muss	Registrierte Mitarbeiter können neue Ideen erfassen	Umgesetzt
Muss	Registrierte Mitarbeiter können sich eine Liste ihrer eingereichten Ideen anzeigen lassen	Umgesetzt
Muss	Registrierte Mitarbeiter können ihre Ideen solange bearbeiten oder auch löschen solange dieses noch nicht zur Bewertung an einen Fachspezialisten übergeben wurden.	Umgesetzt
Muss	Nicht registrierte Mitarbeiter können vorhandene Ideen lesen, sich eine Übersicht der Ideen anzeigen lassen und die Übersicht filtern	Umgesetzt
Muss	Diese Funktionen stehen auch registrierten Mitarbeitern zur Verfügung	Umgesetzt
Muss	Neue Ideen werden Fachspezialisten zur Bewertung zugeordnet	Umgesetzt
Muss	Die Zuordnung erfolgt automatisch sobald die Idee vom registrierten Mitarbeiter zur Bewertung eingereicht wurde	Umgesetzt
Muss	Fachspezialisten können eine Idee entweder annehmen, ablehnen oder für einen späteren Zeitpunkt in einen sog. Ideenspeicher überführen / sie aus dem Ideenspeicher zurückholen	Umgesetzt
Muss	Fachspezialisten begründen ihre Entscheidung transparent und für alle sichtbar in der Anwendung	Umgesetzt
Muss	Fachspezialisten können ihnen zugewiesene Ideen in einer Liste sehen und diese Liste filtern	Umgesetzt

	Anforderung	Umsetzung
Kann	REST-API	Teilweise umgesetzt, lauffähig und erweiterbar
Kann	Kontaktformular auch unregistriert	Umgesetzt, erweiterbar um E-Mail-Einbindung
Kann	Administrator verwaltet Benutzer	Umgesetzt, erweiterbar
Kann	Dokumentenupload zu einer Idee	Nicht umgesetzt, mit Erweiterung der Datenbank umsetzbar
Kann	Profilfoto	Nicht umgesetzt, mit Erweiterung der Datenbank umsetzbar
Kann	Fachspezialist: E-Mail Benachrichtigung bei neuer Idee	Nicht umgesetzt, erfordert E-Mail-Einbindung
Kann	Benutzer: E-Mail Benachrichtigung bei Änderung einer Idee	Nicht umgesetzt, erfordert E-Mail-Einbindung
Kann	PDF-Report über erstellte Ideen quartalsweise	Nicht umgesetzt



## Quellenverzeichnis

### Internetquellen

Heuermann, Christian (2020). *Projektvorgaben Softwareprojekt - Meine Idee Initiative*.

Horn, Torsten (2007). *Vererbung und Polymorphie mit relationalen Datenbanken*. URL: <https://www.torsten-horn.de/techdocs/sql-vererbung.htm> (visited on May 21, 2020).