

# Dokumentation Projekt questMe

Pavithra Sureshkumar, Kevin Sautner, Ralf Zeller

Geändert 13.01.2022

## Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>6</b>
<b>Tabellenverzeichnis</b>	<b>7</b>
<b>1 Vorwort</b>	<b>8</b>
<b>2 About Us</b>	<b>8</b>
<b>3 UI Designs</b>	<b>9</b>
3.1 Recherche UI Designs . . . . .	9
3.2 Erfahrungen zu der Recherche von unseren UI Designs . . . . .	11
3.3 Version 1 vom UI-Konzept . . . . .	11
3.3.1 Version 1 Webchat . . . . .	12
3.3.2 Version 1 Admin Interface Allgemein . . . . .	13
3.3.3 Version 1 Admin Interface Korpus . . . . .	14
3.3.4 Version 1 Admin Interface Login . . . . .	17
3.4 Version 2 von UI-Konzept . . . . .	18
3.4.1 Version 2 Webchat . . . . .	18
3.4.2 Version 2 Admin Interface Allgemein . . . . .	21
3.4.3 Version 2 Admin Interface Korpus . . . . .	24
3.4.4 Version 2 Admin Interface Einstellungen . . . . .	29
3.4.5 Version 2 Admin Interface Login . . . . .	31
3.5 Bisheriger Prototyp vom UI-Design . . . . .	33
3.5.1 Prototyp Webchat . . . . .	33
<b>4 Usability Test</b>	<b>36</b>
4.1 Kriterien für Usability . . . . .	36
4.1.1 Responsive Webdesign . . . . .	36
4.1.2 Gute Lesbarkeit . . . . .	36
4.1.3 Gute Navigation . . . . .	36

4.1.4	Schnelle Ladezeiten . . . . .	36
4.1.5	Interessantes Design . . . . .	36
4.2	Unser Usability Test . . . . .	37
4.2.1	Remote User Testing . . . . .	37
4.2.2	Durchführung der Remote Usability Testsitzung . . . . .	37
<b>5</b>	<b>User Stories</b>	<b>39</b>
5.1	Struktur der User Stories . . . . .	39
5.2	User Stories Version 1 . . . . .	39
<b>6</b>	<b>Use Cases</b>	<b>42</b>
6.1	Struktur der Use Cases . . . . .	42
6.2	Student . . . . .	42
6.3	Unregistrierter Nutzer . . . . .	43
6.4	Professor (Admin) . . . . .	43
<b>7</b>	<b>Zielgruppen</b>	<b>44</b>
7.1	Zielgruppe: Nicht registrierte Nutzer . . . . .	44
7.2	Zielgruppe: Student . . . . .	45
7.3	Zielgruppe: Professor . . . . .	46
<b>8</b>	<b>Technologien</b>	<b>47</b>
8.1	Kriterien für die Technologien . . . . .	47
8.2	Technologie Vergleich . . . . .	47
8.2.1	Vergleich zwischen Angular und Vue.js . . . . .	48
8.2.2	Vergleich zwischen MongoDB und PostgreSQL . . . . .	49
8.3	Technologie Versionen . . . . .	50
8.3.1	Node.js 16.13.0 LTS . . . . .	50
8.3.2	Angular 13.0.1 . . . . .	51
8.3.3	Socket.io 4.3.2 . . . . .	51
8.3.4	MongoDB Community Edition 5.0.3 . . . . .	51
8.3.5	Keycloak 15.0.2 . . . . .	51
8.3.6	RegEx . . . . .	52
8.3.7	NLPjs 4.22.2 . . . . .	52
8.4	Technologie Diagramme . . . . .	53
8.4.1	UML Komponentendiagramme . . . . .	53
8.4.2	UML Komponentendiagramm: Client . . . . .	53
8.4.3	UML Komponentendiagramm: Server . . . . .	54
8.4.4	UML Komponentendiagramm . . . . .	55
8.4.5	UML Verteilungsdiagramm . . . . .	56
8.5	Datenbank . . . . .	57
8.5.1	Datenhaltung . . . . .	57
8.5.2	ER Diagramme . . . . .	57

8.6	Sicherheit . . . . .	59
8.6.1	Frontend . . . . .	59
8.6.2	Backend . . . . .	59
8.6.3	Angriffs-Szenarios . . . . .	60
<b>9</b>	<b>Meilensteine</b>	<b>61</b>
9.1	Recherche 13.10.21 . . . . .	61
9.2	Zwischenpräsentation 22.10.21 . . . . .	61
9.3	Implementation 25.10.21 . . . . .	62
9.4	MVP 02.12.21 . . . . .	62
9.5	Endpräsentation und Enddokumentation 14.01.22 . . . . .	62
<b>10</b>	<b>Zeitmanagement</b>	<b>63</b>
10.1	Gantt-Diagramm . . . . .	63
<b>11</b>	<b>Risikoanalyse</b>	<b>64</b>
11.1	Fazit . . . . .	66
<b>12</b>	<b>Installationshandbuch</b>	<b>67</b>
12.1	Erste Schritte um die Software zum Laufen zu bringen . . . . .	67
12.1.1	Setup to run a Angular project . . . . .	67
12.1.2	Docker Comnpose zum Laufen bringen . . . . .	67
12.1.3	Docker Desktop mit den Container . . . . .	69
<b>13</b>	<b>Aufteilung des Teams</b>	<b>70</b>
13.1	Herr Ralf Zeller . . . . .	70
13.2	Frau Pavithra Sureshkumar . . . . .	72
13.3	Herr Kevin Sautner . . . . .	74
<b>14</b>	<b>Reflektion Projektmanagement</b>	<b>75</b>
14.1	Geplante Meilensteine für Reflektion . . . . .	75
14.1.1	Recherche <b>11.01.22</b> . . . . .	75
14.1.2	Zwischenpräsentation <b>22.10.21</b> . . . . .	76
14.1.3	Implementation <b>12.01.22</b> . . . . .	76
14.1.4	MVP und Codereview <b>10.12.21</b> . . . . .	76
14.1.5	Endpräsentation und Enddokumentation <b>14.01.22</b> . . . . .	76
<b>15</b>	<b>Reflektion Lernfortschritt</b>	<b>77</b>
15.1	Reflektion Lernfortschritt von Frau Pavithra Sureshkumar . . . . .	77
15.2	Reflektion Lernfortschritt von Herr Kevin Sautner . . . . .	79
<b>16</b>	<b>Ausblick: Pläne für die Zukunft</b>	<b>80</b>
16.1	Mögliche Ergänzungen in der Zukunft . . . . .	80

<b>17</b>	<b>Appendix</b>	<b>81</b>
17.1	Ältere Versionen des Komponentendiagramms . . . . .	81
17.1.1	Komponentendiagramme . . . . .	81
<b>18</b>	<b>Meeting Protokolle</b>	<b>84</b>
18.1	Protokolle in PDFs eingebunden . . . . .	84
<b>Literatur</b>		<b>118</b>

# Abbildungsverzeichnis

1	Old version UI Design WebChat . . . . .	12
2	Old version UI Design Admin Interface Allgemein . . . . .	13
3	Old version UI Design Admin Interface Korpus 00 . . . . .	14
4	Old version UI Design Admin Interface Korpus 01 . . . . .	15
5	Old version UI Design Admin Interface Korpus 02 . . . . .	16
6	Old version UI Design Admin Interface Login . . . . .	17
7	New version UI Design Webchat . . . . .	18
8	New version UI Design Webchat Hochschuldomäne . . . . .	19
9	New version UI Design Webchat mobile version . . . . .	20
10	New version UI Design Admin-Interface Allgemein . . . . .	21
11	New version UI Design Admin-Interface Allgemein dropdown menu . . . . .	22
12	New version UI Design Admin-Interface Allgemein mobile version . . . . .	23
13	New version UI Design Admin-Interface Korpus 00 . . . . .	24
14	New version UI Design Admin-Interface Korpus 01 . . . . .	25
15	New version UI Design Admin-Interface Korpus 02 . . . . .	26
16	New version UI Design Admin-Interface Korpus 03 . . . . .	27
17	New version UI Design Admin-Interface Korpus mobile version . . . . .	28
18	New version UI Design Admin-Interface Einstellungen . . . . .	29
19	New version UI Design Admin-Interface Einstellungen mobile version . . . . .	30
20	New version UI Design Admin-Interface Login . . . . .	31
21	New version UI Design Admin-Interface Login mobile version . . . . .	32
22	Prototyp Chat Interface . . . . .	33
23	Admin Interface Infoseite . . . . .	34
24	Admin Interface Hamburgermenu . . . . .	35
25	UML Komponentendiagramm Client . . . . .	53
26	UML Komponentendiagramm Server . . . . .	54
27	UML Komponentendiagramm . . . . .	55
28	UML Verteilungsdiagramm . . . . .	56
29	ER Diagramm Korpus . . . . .	57
30	ER Diagramm Professor . . . . .	58
31	ER Diagramm Student . . . . .	58
32	ER Diagramm Unregistrierter Nutzer . . . . .	58
33	Frontend-Sicherheit . . . . .	59
34	Gantt-Diagramm . . . . .	63
35	node.js 16.03.0 LTS . . . . .	67
36	Docker Desktop Windows . . . . .	68
37	Docker Desktop Container . . . . .	69
38	Komponentendiagramm Client . . . . .	81
39	Komponentendiagramm Server . . . . .	82
40	Komponentendiagramm v1.0 . . . . .	83
41	Komponentendiagramm v1.1 . . . . .	83

42 Komponentendiagramm v1.2 . . . . .	84
---------------------------------------	----

## **Tabellenverzeichnis**

1	Vergleich zwischen Angular und Vue.js . . . . .	48
2	Vergleich zwischen MongoDB und PostgreSQL . . . . .	49
3	Technologie Liste . . . . .	50
4	Meilenstein Liste . . . . .	61
5	Risikoanalyse Tabelle Teil 1 . . . . .	64
6	Risikoanalyse Tabelle Teil 2 . . . . .	65
7	Meilenstein Liste: Neuer Stand . . . . .	75

# 1 Vorwort

In dieser Dokumentation möchten wir alle relevanten Informationen zum Projekt sammeln.

## 2 About Us

Hier stellen wir uns kurz vor.

### Wer sind wir ?



Wir sind questMe und wir bearbeiten das Thema ChatBot. In unserem Projekt möchten wir nicht nur den ChatBot programmieren oder erstellen, sondern auch neue Technologien kennenlernen. Wir möchten neue Erfahrungen sammeln, lernen wie man organisiert mit Konflikten umgeht und auch bestimmte Probleme angeht. Wir sind als Team immer offen für Neues.

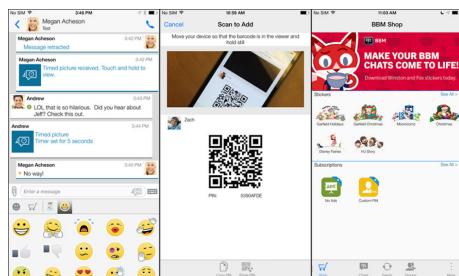
**Wir sind das Team questMe.**

# 3 UI Designs

In diesem Kapitel sollen die verschiedenen Versionen unseres UI Designs geführt werden.

## 3.1 Recherche UI Designs

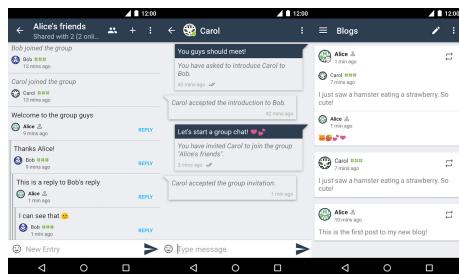
Zuerst haben wir uns mehrere UI Designs von verschiedenen Quellen angeschaut, um einen besseren Überblick über die Designmöglichkeiten zu bekommen.



Bildquelle: *Blackberry-messenger-live-free*, [o. D.]

### Blackberry Messenger Live

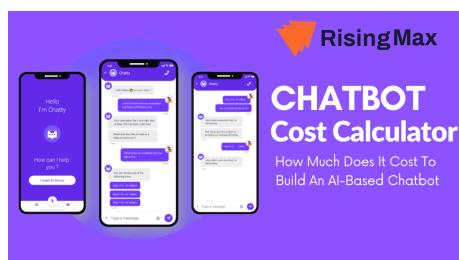
Zuerst haben wir uns umgeschaut und nach Chatfenstern gesucht. Hier haben wir uns die Austauschung von Sprechblasen angeschaut und deren Ausrichtung.



Bildquelle: *Briar*, [o. D.]

### Briar

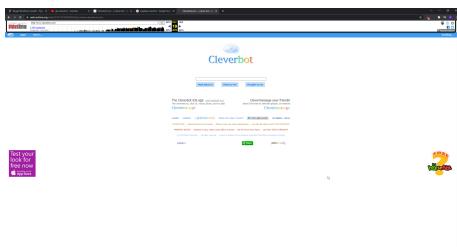
In diesem Beispiel haben wir uns wieder das Chatfenster und die verschiedenen Symbole angeschaut. Wie den Editierbutton oder den Hinzufügebutton.



Bildquelle: *Chatbot-cost-calculator*, [o. D.]

### CHATBOT Cost Calculator

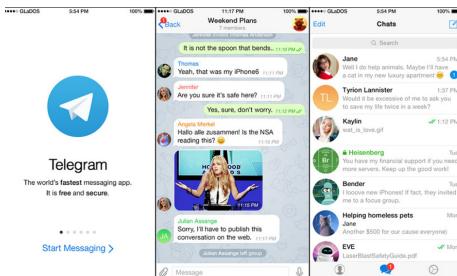
Hier haben wir uns ein ChatBot-Fenster angeschaut und dabei die Icons und die verschiedenen Elemente angeschaut, wie den Balken an der Oberseite oder den Sendebutton.



## Cleverbot

Diesen Bot haben wir uns genauer angeschaut, weil dieser auch uns bekannt war.

Bildquelle: *Cleverbot*, [o. D.]



## Telegram

Telegram haben wir uns wiederum das Chatfenster und die Icons angeschaut.

Bildquelle: *Telegram Bild*, [o. D.]



## WhatsApp

Hier haben wir uns mehr auf die Ausrichtungen des Chatfensters angeschaut und wie die Sprechblasen ausgerichtet sind. Fast jeder benutzt WhatsApp, deswegen haben wir uns die Struktur angeschaut, weil diese vielen Benutzern bekannt ist.

Bildquelle: *Tim WhatsApp*, [o. D.]

## **3.2 Erfahrungen zu der Recherche von unseren UI Designs**

In Recherche von den UI Designs haben die einzelnen Komponenten wie der Aufbau eines Chats und die Anordnung eine große Rolle gespielt. Die Erfahrungen, die man aus den einzelnen Elementen gesammelt hat, sind:

Die Chatfenster sind immer gleich aufgebaut. Sie haben alle eine große Fläche, wo die Chats angezeigt werden. Außerdem haben alle Chat Designs ein Textfeld, wo man seine Fragen und Anliegen schreiben kann. Jeder User besitzt ein Profilbild. So- gar der ChatBot besitzt ein Profilbild. Das Chatfeld vom ChatBot Calculator hat ein gutes Design, wo das Profilbild neben dem Chatfeld zu sehen ist. Demnach ist nach- vollziehbar, wer was geschrieben hat. Neben dem Textfeld, wo man seine Anliegen eingeben kann, steht immer der Sendebutton. Die Benutzer sind also mehr auf ein UI Design eingestellt, welches ihnen immer vorgelegt wird. Darum wird das quest- Me Chatfenster auch ein UI Design erhalten, welches sich von der Basisstruktur der anderen Chats im Alltag nicht unterscheidet.

## **3.3 Version 1 vom UI-Konzept**

Hier sind die älteren Entwürfe des Designs. Bei diesen Entwürfen haben wir uns nur auf die Struktur konzentriert und eine grobe Version erstellt. Uns war es hier haupt- sächlich wichtig die Ideen, die wir durch die Recherche aufgenommen haben zu pro- jizieren.

### 3.3.1 Version 1 Webchat

Hier sieht man die älteren UI Designs Versionen, die zum Teil Chatfenster gehören.

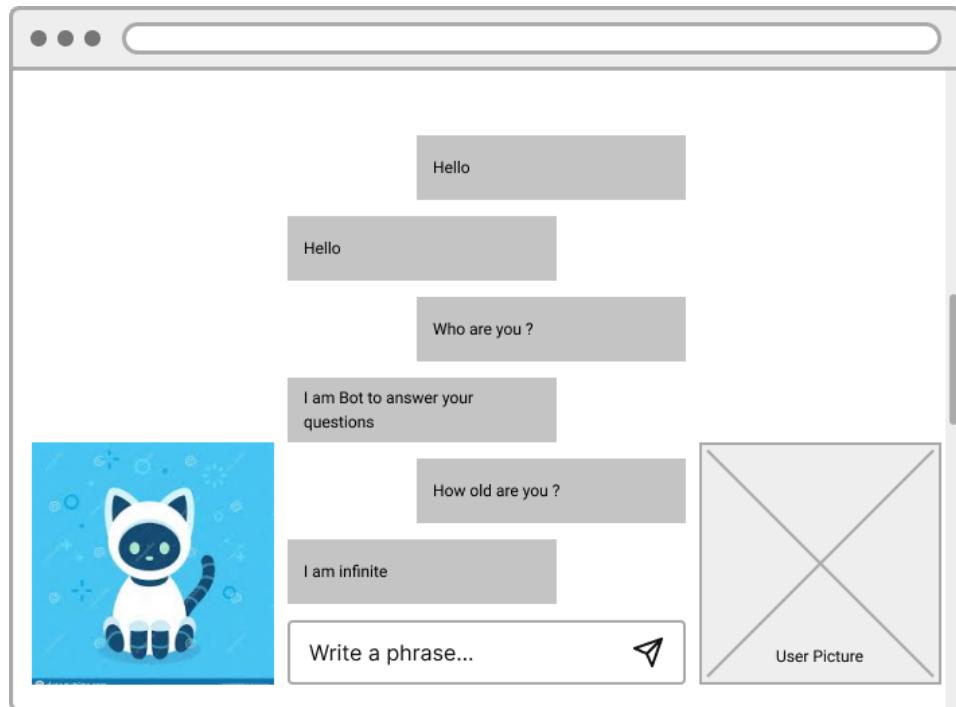


Abbildung 1: Old version UI Design WebChat

### Webchat

In der alten Version unseres Webchat Designs haben wir gedacht, auch für den Nutzer ein Profilbild hinzuzufügen. Diese Idee schien aber zu aufwendig zu sein und wir versuchten zuerst ein Minimal Viable Product zu erschaffen.

### 3.3.2 Version 1 Admin Interface Allgemein

Hier sieht man die älteren UI Designs Versionen, die zum Teil Admin-Interface Allgemein gehören.

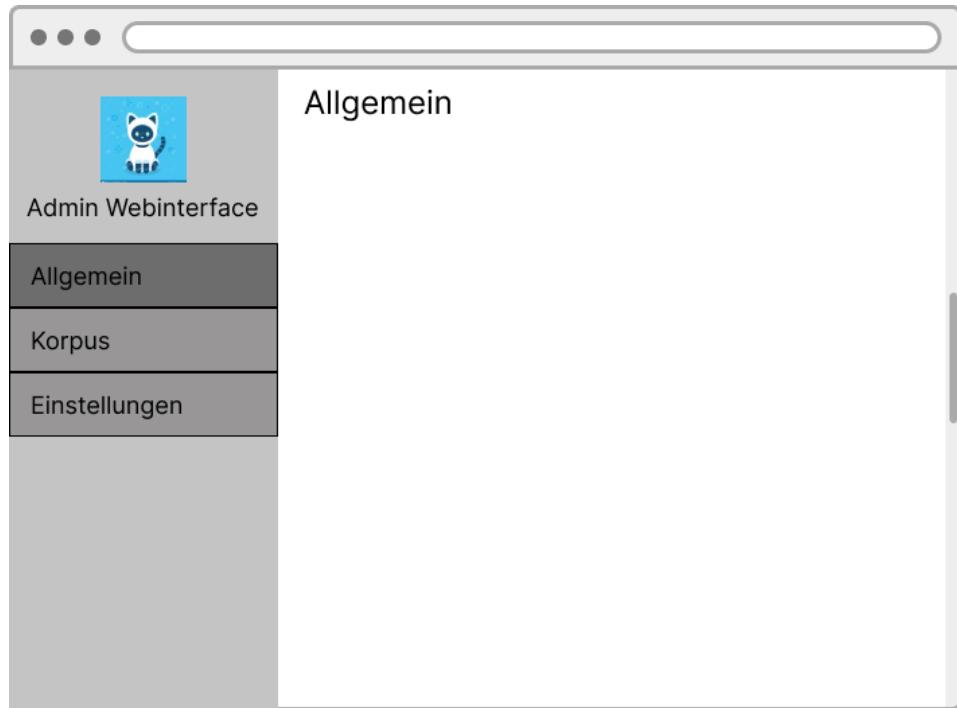


Abbildung 2: Old version UI Design Admin Interface Allgemein

#### Admin Interface: Allgemein

Im Admin Interface kann man auf der linken Seite das Menü sehen. Was aber hier fehlt ist der Logout Button. Wir hatten auch keine richtigen Vorstellungen, was wir einführen möchten.

### 3.3.3 Version 1 Admin Interface Korpus

Hier sieht man die älteren UI Designs Versionen, die zum Teil Admin-Interface Korpus gehören.

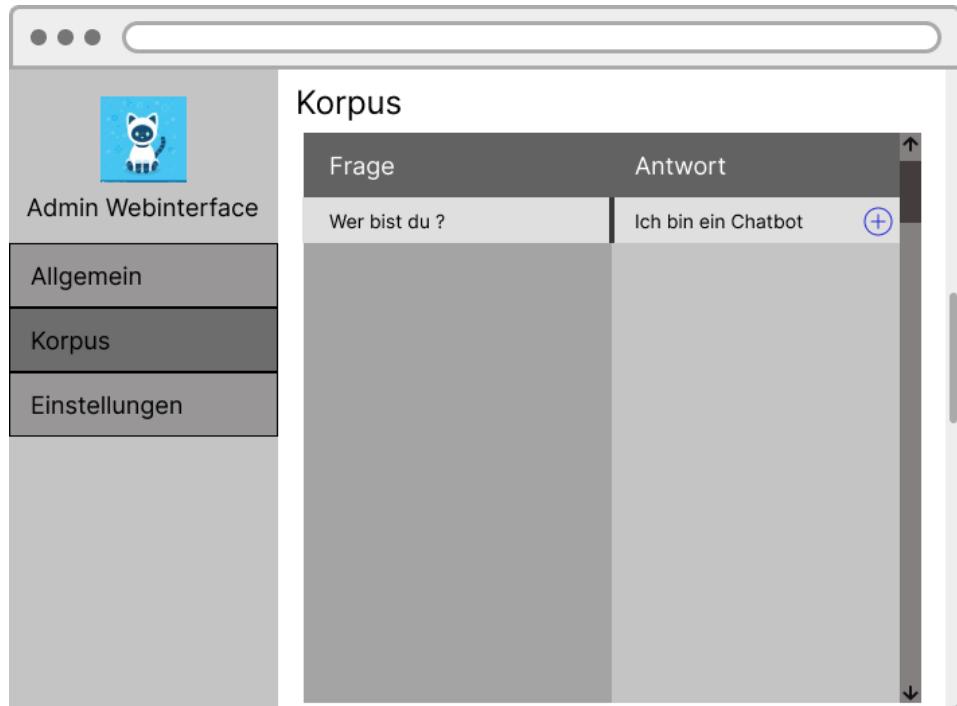


Abbildung 3: Old version UI Design Admin Interface Korpus 00

#### Admin Interface: Korpus 00

Im Korpus wollten wir schon von Anfang an das Hinzufügen darstellen, wussten aber nicht wie und haben herumprobiert.

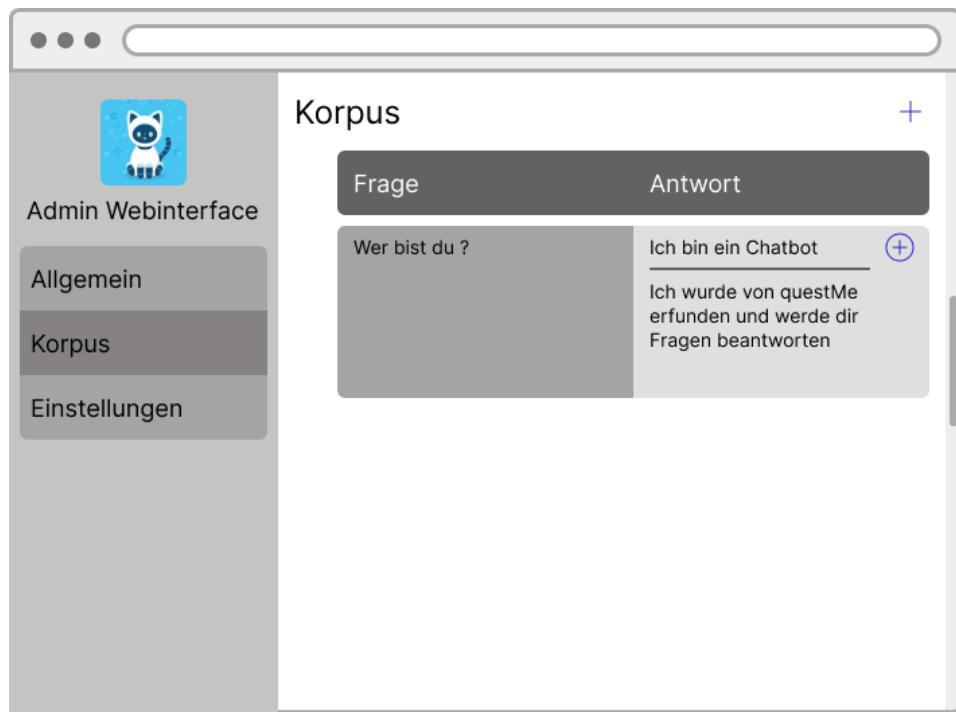


Abbildung 4: Old version UI Design Admin Interface Korpus 01

### Admin Interface: Korpus 01

In diesem Beispiel sieht man ein Basisbeispiel mit einer Frage, die im Alltag gestellt wird und die dazugehörigen Antworten. Wie wir diese aber editieren haben wir hier noch nicht gezeigt.



Abbildung 5: Old version UI Design Admin Interface Korpus 02

### Admin Interface: Korpus 02

Im nächsten Beispiel sieht man eine weitere Frage und die dazugehörenden zwei Antworten. Es ist immer noch nicht bekannt, wie man Antworten und Fragen editiert oder Fragen und Antworten von verschiedenen Domänen bearbeitet.

### 3.3.4 Version 1 Admin Interface Login

Hier sieht man die älteren UI Designs Versionen, die zum Teil Admin-Interface Login gehören.

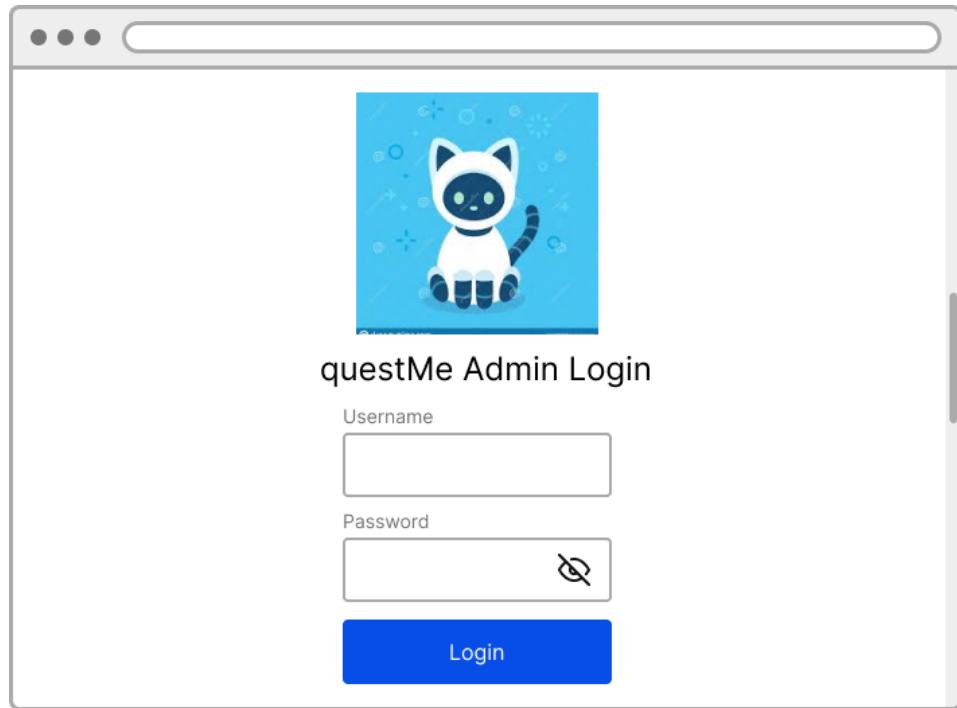


Abbildung 6: Old version UI Design Admin Interface Login

#### Admin Interface: Admin Login

Hier haben wir nur ein klassisches Login Eingabefeld dargestellt, weil wir uns nicht klar waren, wie es mit der Authentifizierung und dem Admin Login funktioniert.

### 3.4 Version 2 von UI-Konzept

Hier sieht man die neuen Entwürfe des UI Designs. In der neuen Version werden auch die mobilen Versionen entworfen. Weil der Plan ist, zuerst eine mobile first Anwendung herzustellen.

#### 3.4.1 Version 2 Webchat

Hier werden die neueren Versionen des UI Designs für den Webchat vorgestellt.

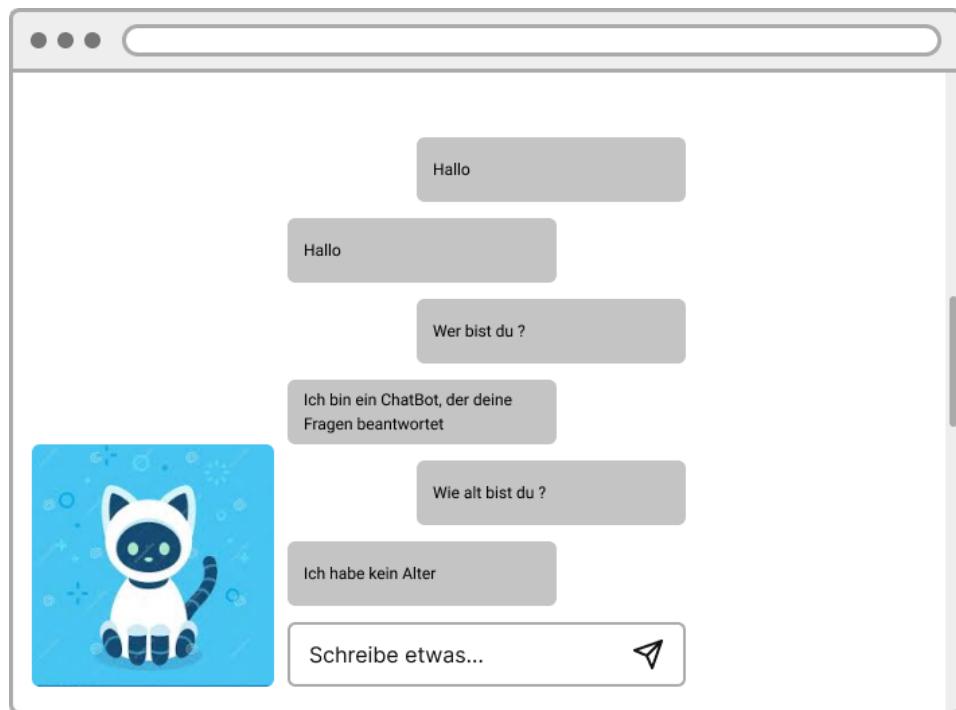


Abbildung 7: New version UI Design Webchat

#### Webchat

Hier haben wir einen Beispielchat mit dem Bot dargestellt. In diesem Beispiel haben wir ein Basisgespräch geführt.

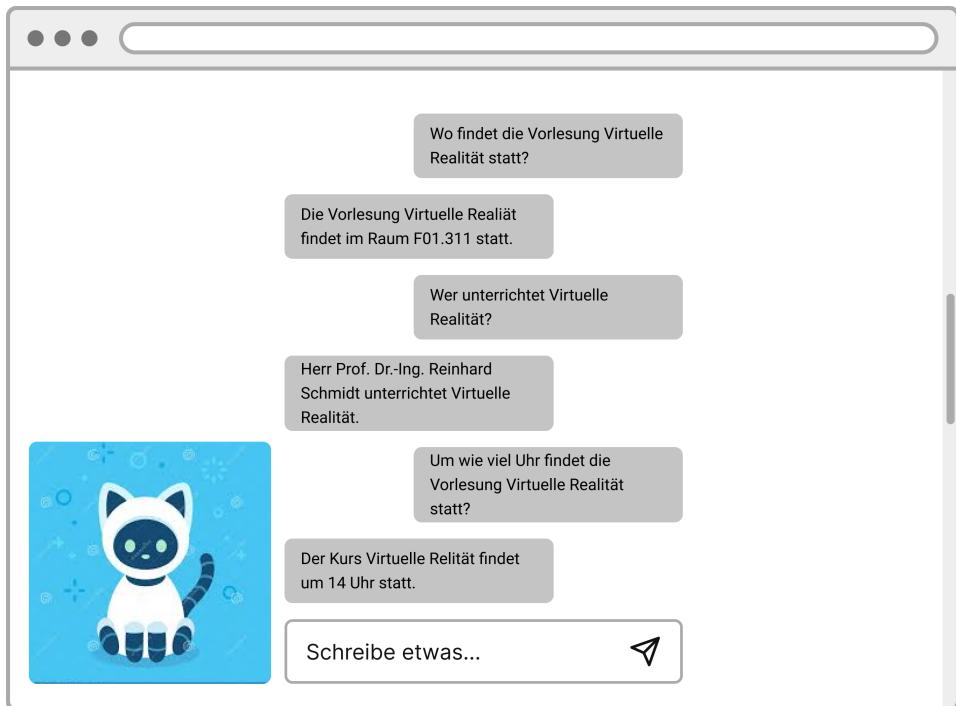


Abbildung 8: New version UI Design Webchat Hochschuldomäne

### Webchat mit der Hochschuldomäne

Diesmal haben wir konkrete Hochschulfragen gestellt und die Hochschuldomäne dargestellt.

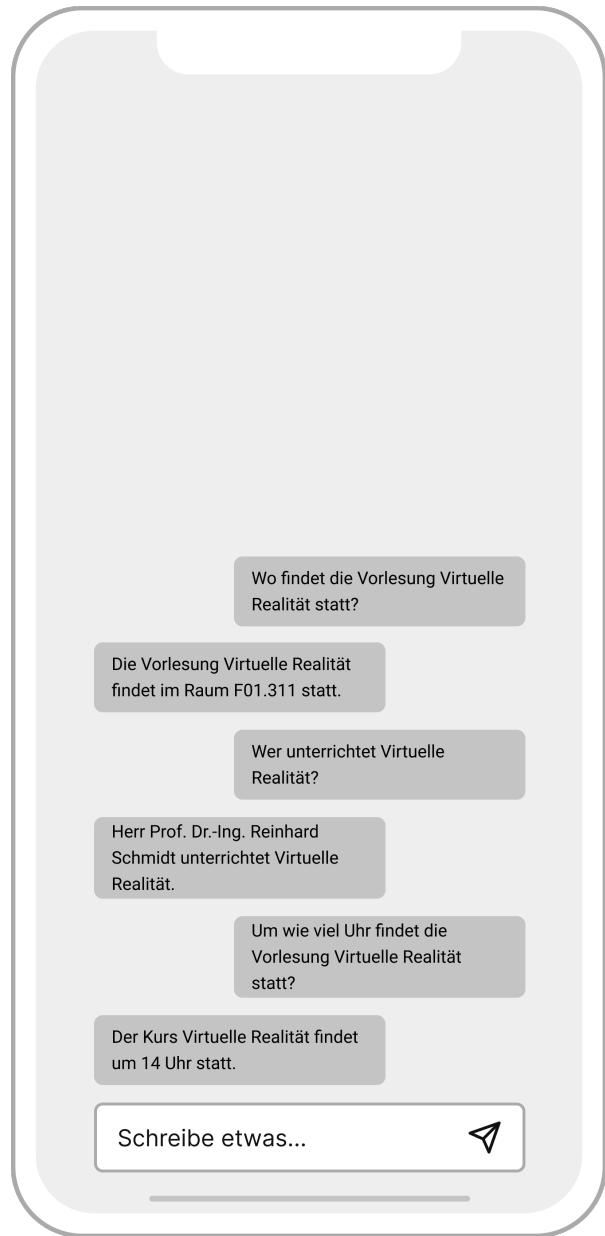


Abbildung 9: New version UI Design Webchat mobile version

### Webchat als mobile Version

Dies ist die mobile Version des Webchats. Wir haben die Darstellung so einfach wie möglich dargestellt. Hier haben wir auch die Hochschulbezogenen Fragen gestellt.

### 3.4.2 Version 2 Admin Interface Allgemein

Hier werden die neueren Versionen des UI Designs für das Admin-Interface Allgemein vorgestellt

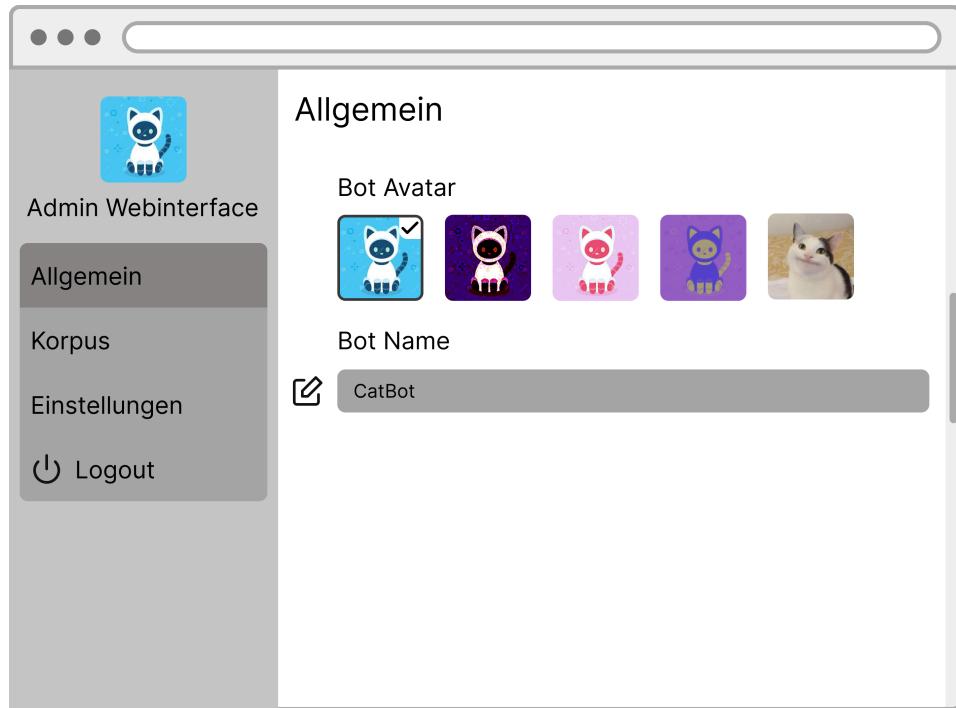


Abbildung 10: New version UI Design Admin-Interface Allgemein

#### Admin Webinterface: Allgemein

Auf der linken Seite sieht man die Kategorien, die der Admin bearbeiten kann. Im Allgemeinen kann der Admin den Bot Avatar wechseln, dieser wird dann mit einem Haken gekennzeichnet. Außerdem kann der Admin den Bot Namen ändern, indem er den Editierbutton drückt.

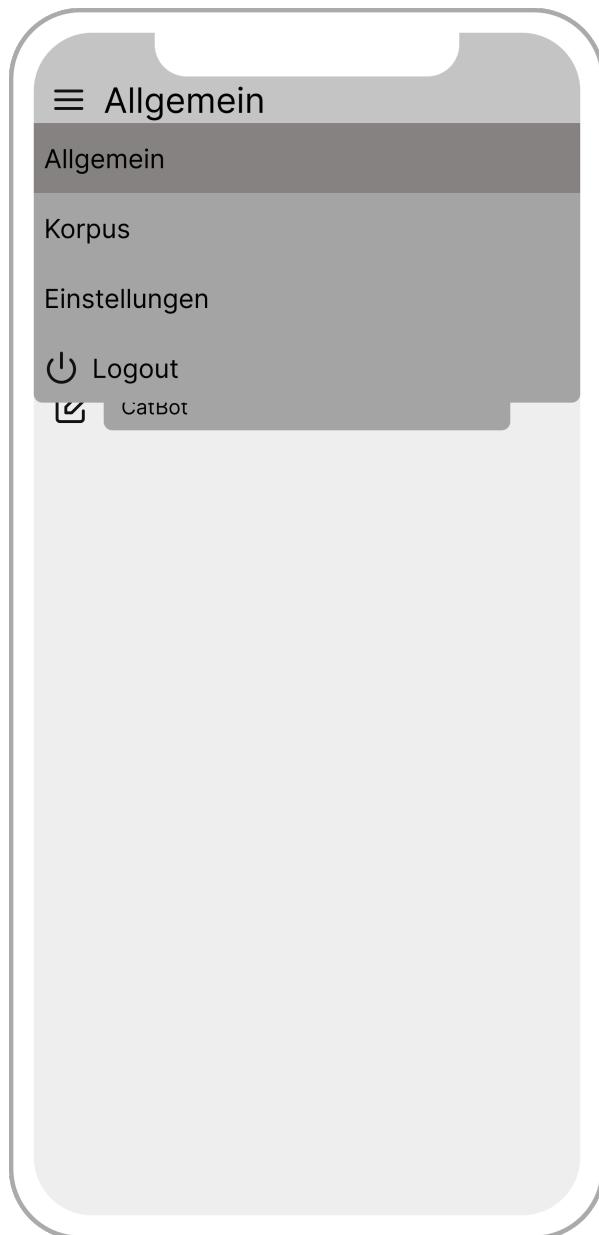


Abbildung 11: New version UI Design Admin-Interface Allgemein dropdown menu

#### **Admin Webinterface mobil: Allgemein dropdown Menü**

In der mobilen Version haben wir die Kategorien, die der Admin bearbeiten kann im dropdown Menü dargestellt.

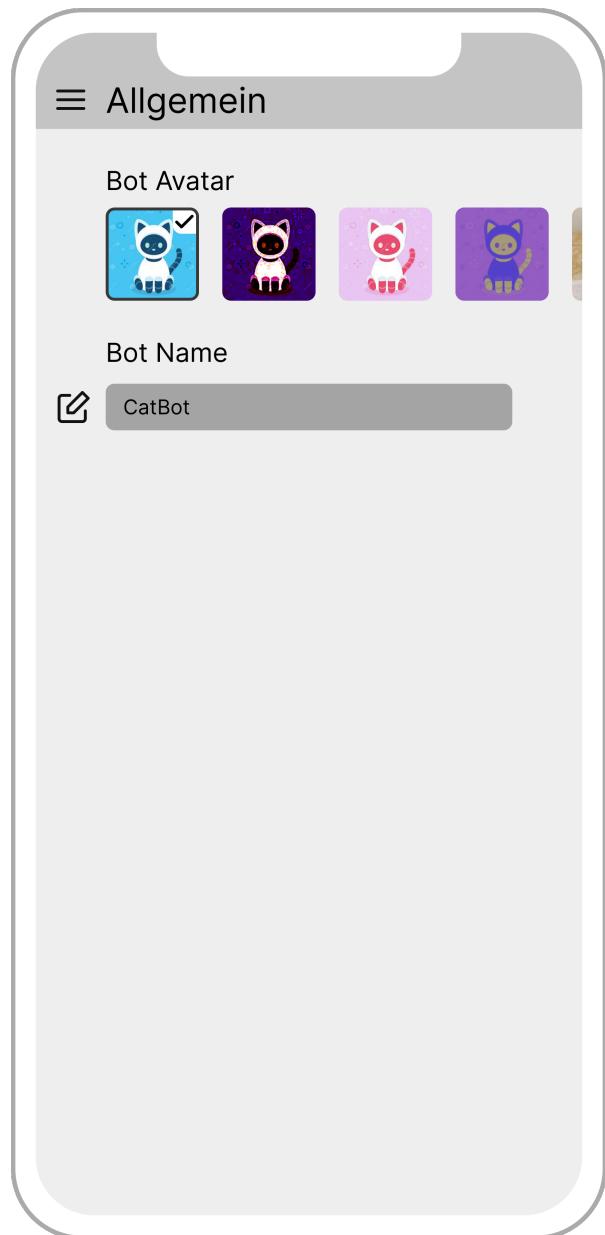


Abbildung 12: New version UI Design Admin-Interface Allgemein mobile version

#### **Admin Webinterface mobil: Allgemein**

Die mobile Variante funktioniert genauso wie die Webvariante. Man kann den Bot Avatar wechseln und den ChatBot Namen frei bestimmen.

### 3.4.3 Version 2 Admin Interface Korpus

Hier werden die neueren Versionen des UI Designs für das Admin-Interface Korpus vorgestellt.

The screenshot shows a web browser window with a light gray header bar containing three dots and a search bar. The main interface has a white background. On the left, there is a sidebar with a blue icon of a white cat and the text "Admin Webinterface". Below this are four menu items: "Allgemein", "Korpus" (which is highlighted in a dark gray box), "Einstellungen", and "Logout". In the center, the word "Korpus" is displayed in large black font above a table. The table has two columns: "Frage" and "Antwort". There are three rows of data:

Frage	Antwort
Wo findet die Vorlesung Virtuelle Realität statt?	Die Vorlesung Virtuelle Realität findet im Raum F01.311 statt.
Wer unterrichtet Virtuelle Realität?	Herr Prof. Dr.-Ing. Reinhard Schmidt unterrichtet Virtuelle Realität.
Um wie viel Uhr findet die Vorlesung Virtuelle Realität statt?	Der Kurs Virtuelle Realität findet um 14 Uhr statt.

A blue plus sign button is located in the top right corner of the central area.

Abbildung 13: New version UI Design Admin-Interface Korpus 00

#### Admin Webinterface: Korpus 00

Im Korpus kann der Admin weitere Domänen, Fragen und Antworten einsehen.

The screenshot shows a web-based administration interface for a corpus. On the left, a sidebar menu includes 'Admin Webinterface' with a cat icon, 'Allgemein', 'Korpus' (which is selected and highlighted in grey), 'Einstellungen', and 'Logout'. The main content area is titled 'Korpus' and features a dropdown menu 'Hochschule' with a blue '+' button next to it. Below this is a table with three columns: 'Frage' (Question), 'Antwort' (Answer), and edit icons. The first row contains the question 'Wo findet die Vorlesung Virtuelle Realität statt?' and the answer 'Die Vorlesung Virtuelle Realität findet im Raum F01.311 statt.' The second row contains the question 'Wer unterrichtet Virtuelle Realität?' and the answer 'Herr Prof. Dr.-Ing. Reinhard Schmidt unterrichtet Virtuelle Realität.' The third row contains the question 'Um wie viel Uhr findet die Vorlesung Virtuelle Realität statt?' and the answer 'Der Kurs Virtuelle Realität findet um 14 Uhr statt.'

Frage	Antwort
Wo findet die Vorlesung Virtuelle Realität statt?	Die Vorlesung Virtuelle Realität findet im Raum F01.311 statt.
Wer unterrichtet Virtuelle Realität?	Herr Prof. Dr.-Ing. Reinhard Schmidt unterrichtet Virtuelle Realität.
Um wie viel Uhr findet die Vorlesung Virtuelle Realität statt?	Der Kurs Virtuelle Realität findet um 14 Uhr statt.

Abbildung 14: New version UI Design Admin-Interface Korpus 01

### Admin Webinterface: Korpus 01

Mit dem Editierbutton kann er Domänen, Fragen und Antworten hinzufügen und entfernen.

The screenshot shows the Admin Webinterface for 'Korpus 02'. On the left, a sidebar menu includes 'Admin Webinterface' with a cat icon, 'Allgemein', 'Korpus' (selected), 'Einstellungen', and 'Logout'. The main area is titled 'Korpus' and has a dropdown 'Hochschule' set to 'Hochschule'. A blue '+' button is in the top right. Below, a table lists questions and their answers:

Frage	Antwort
Wo findet die Vorlesung Virtuelle Realität statt?	Die Vorlesung Virtuelle Realität findet im Raum F01.311 statt. Die Vorlesung findet im Raum F01.311 statt.
Wer unterrichtet Virtuelle Realität?	Herr Prof. Dr.-Ing. Reinhard Schmidt unterrichtet Virtuelle Realität.
Um wie viel Uhr findet die Vorlesung Virtuelle Realität statt?	Der Kurs Virtuelle Realität findet um 14 Uhr statt.

Abbildung 15: New version UI Design Admin-Interface Korpus 02

### Admin Webinterface: Korpus 02

So kann man wie im Beispiel eine weitere Antwort zu der Frage: "Wo findet die Vorlesung Virtuelle Realität statt?", hinzufügen.

The screenshot shows the Admin Webinterface for 'Korpus'. On the left, there's a sidebar with a logo of a white cat with blue spots, labeled 'Admin Webinterface'. Below the logo are buttons for 'Allgemein', 'Korpus' (which is selected), 'Einstellungen', and 'Logout'. The main area is titled 'Korpus' and has a dropdown menu 'Hochschule' with a blue plus sign icon. Below this is a table with two columns: 'Frage' and 'Antwort'. There are four rows of data:

Frage	Antwort
<input type="text"/> Wo findet die Vorlesung Virtuelle Realität statt?	<input type="text"/> Die Vorlesung Virtuelle Realität findet im Raum F01.311 statt.  <input type="text"/> Die Vorlesung findet im Raum F01.311 statt. <span style="color:red;">[trash]</span>
<input type="text"/> Wer unterrichtet Virtuelle Realität?	Herr Prof. Dr.-Ing. Reinhard Schmidt unterrichtet Virtuelle Realität.
<input type="text"/> Um wie viel Uhr findet die Vorlesung Virtuelle Realität statt?	Der Kurs Virtuelle Realität findet um 14 Uhr statt.

Abbildung 16: New version UI Design Admin-Interface Korpus 03

### Admin Webinterface: Korpus 03

Natürlich kann man auch die hinzugefügte Antwort entfernen. Man klickt auf das Feld und das Feld erscheint rötlich und ein Müllimersymbol entsteht. Wenn man jetzt auf das Eimer-Symbol klickt, kann man die Antwort löschen.

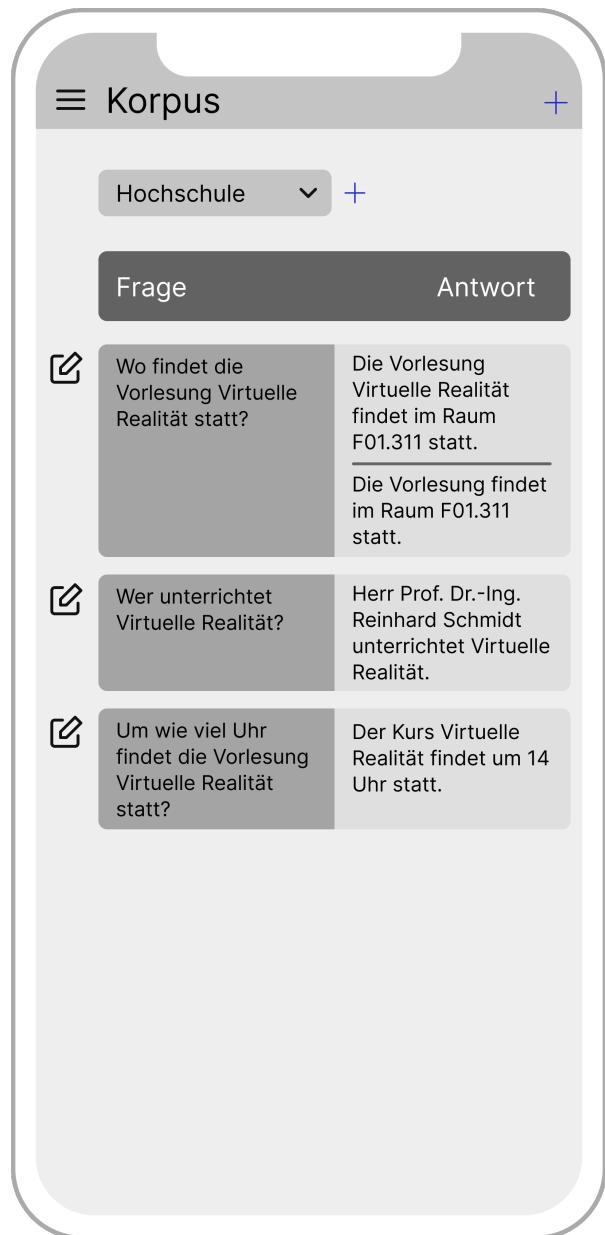


Abbildung 17: New version UI Design Admin-Interface Korpus mobile version

### **Admin Webinterface mobil: Korpus**

In der mobilen Version des Korpus kann man die Elemente genauso editieren wie im Webbrowser.

### 3.4.4 Version 2 Admin Interface Einstellungen

Hier werden die neueren Versionen des UI Designs für das Admin-Interface Einstellungen vorgestellt.

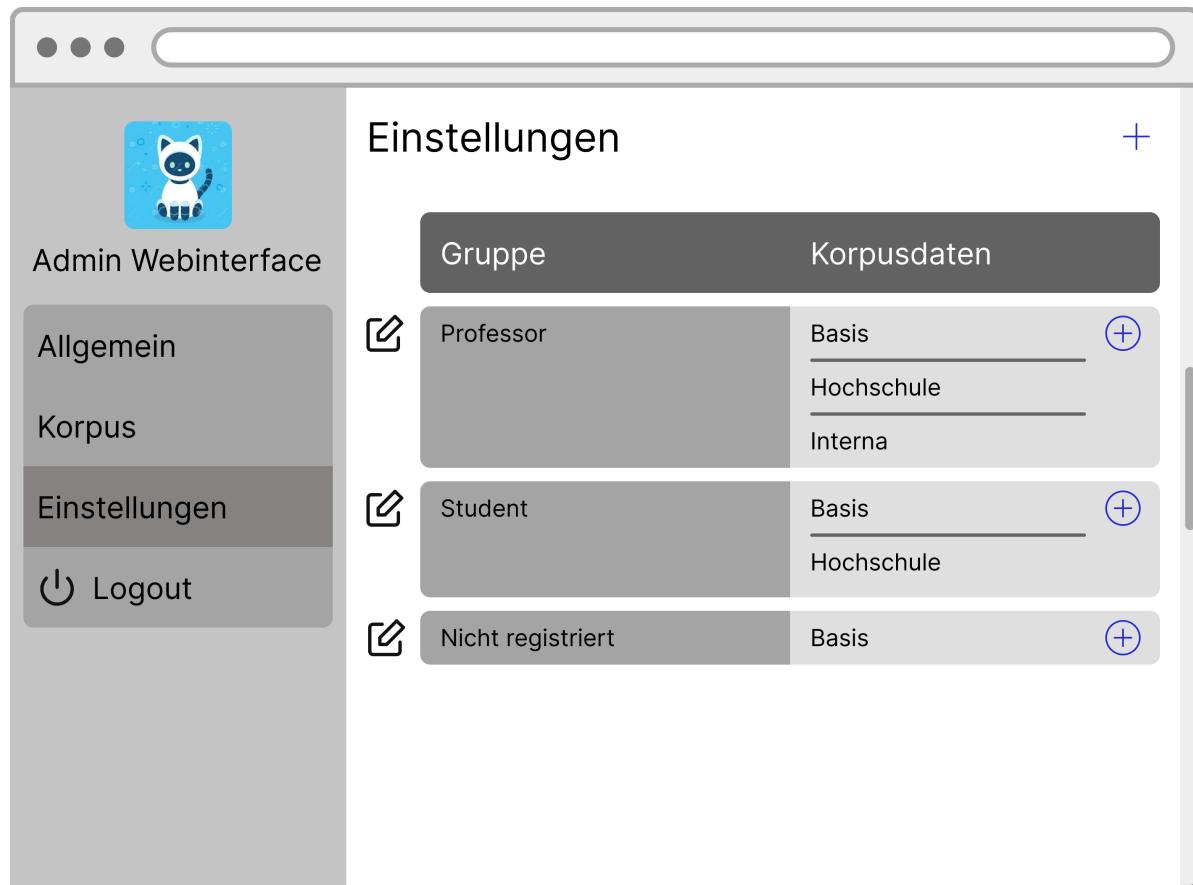


Abbildung 18: New version UI Design Admin-Interface Einstellungen

#### Admin Webinterface: Einstellungen

In Einstellungen kann der Admin seine Gruppen einsehen, hinzufügen und entfernen. Er kann auch die dazugehörigen Korpusdaten sehen und bearbeiten. In den Korpusdaten sind die Domänen der jeweiligen Gruppe eingetragen.

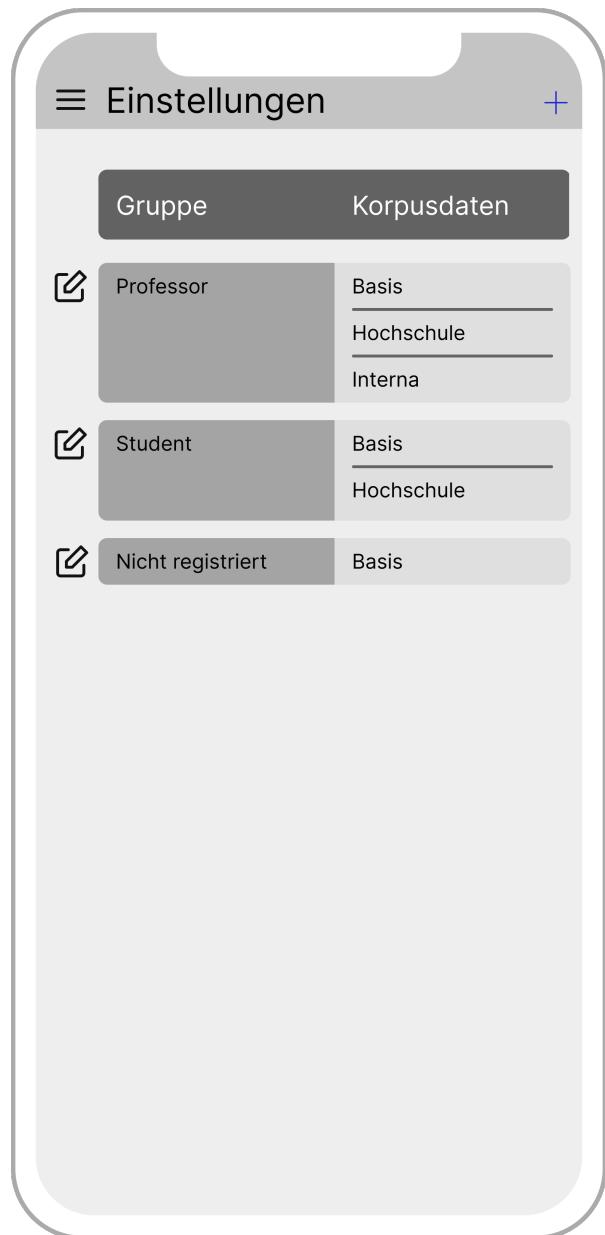


Abbildung 19: New version UI Design Admin-Interface Einstellungen mobile version

#### **Admin Webinterface mobil: Einstellungen**

In der mobilen Version kann man ebenso die gleichen Features nutzen.

### 3.4.5 Version 2 Admin Interface Login

Hier werden die neueren Versionen des UI Designs für das Admin-Interface Login vorgestellt.

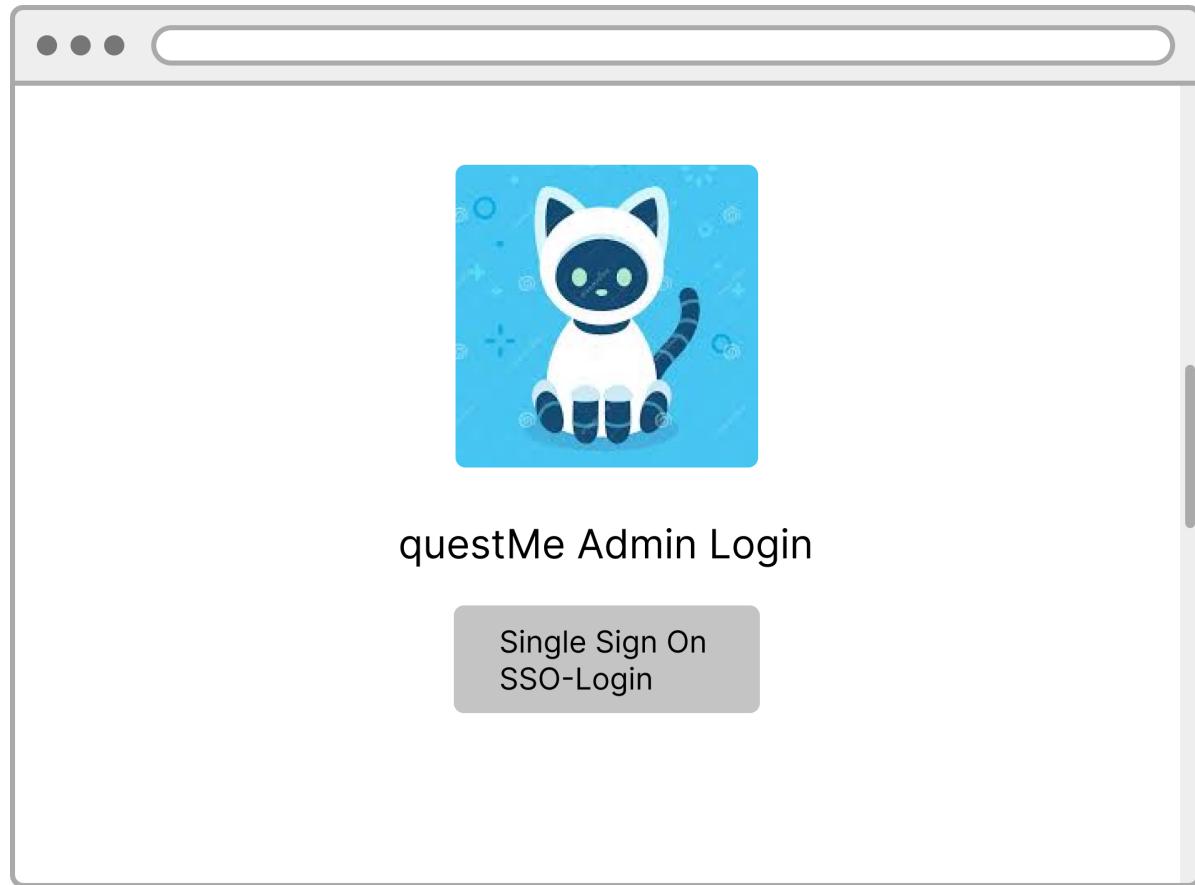


Abbildung 20: New version UI Design Admin-Interface Login

#### Admin Webinterface: Single Sign On

Mit einem Link gelangt der Admin zu der Admin Login Seite, wo er mit Shibboleth sich einloggen kann.

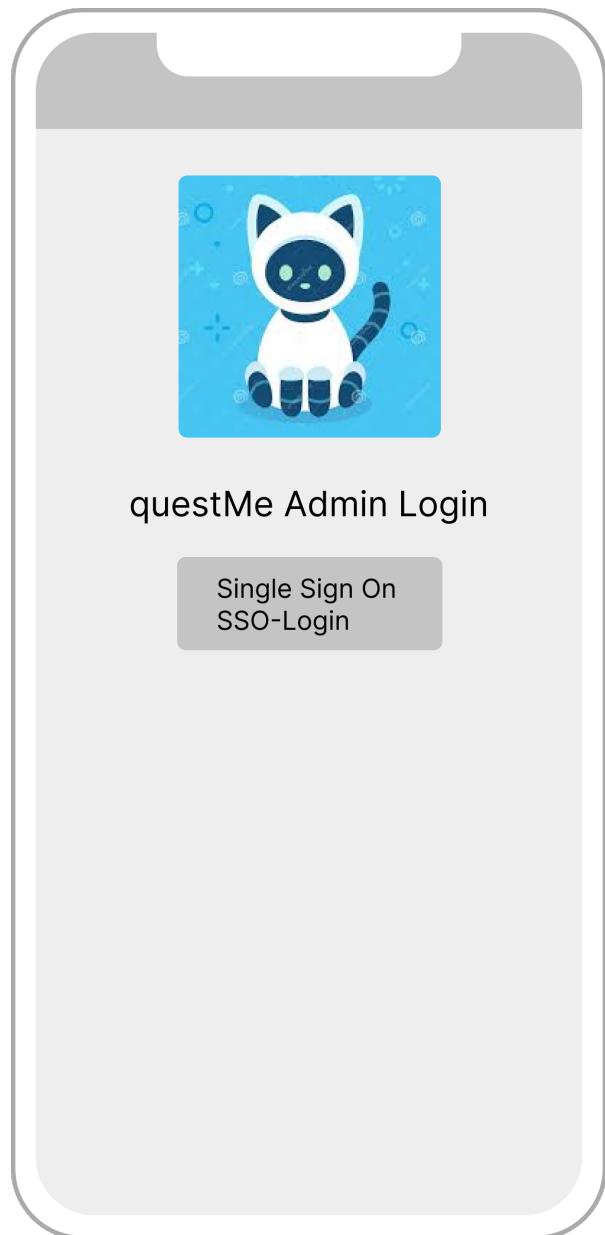


Abbildung 21: New version UI Design Admin-Interface Login mobile version

#### **Admin Webinterface mobil: Single Sign On**

In der mobilen Version wird die gleiche Prozedur benutzt.

### 3.5 Bisheriger Prototyp vom UI-Design

Hier wird der bisherige Prototyp des UI-Designs, welche wir mit Angular bis jetzt programmiert haben, dargestellt.

#### 3.5.1 Prototyp Webchat

Hier wird der Prototyp vom Webchat vorgestellt.



Abbildung 22: Prototyp Chat Interface

#### Chat Interface

Dies ist der bisherige Prototyp des Chat Interfaces. Wie man hier sieht haben wir schon auf eine gute Lesbarkeit des Chats geachtet und auch mobile-first entwickelt.



Abbildung 23: Admin Interface Infoseite

### **Admin Interface Infoseite**

Hier haben wir die Admin Interface Infoseite, wo der Admin begrüßt wird.

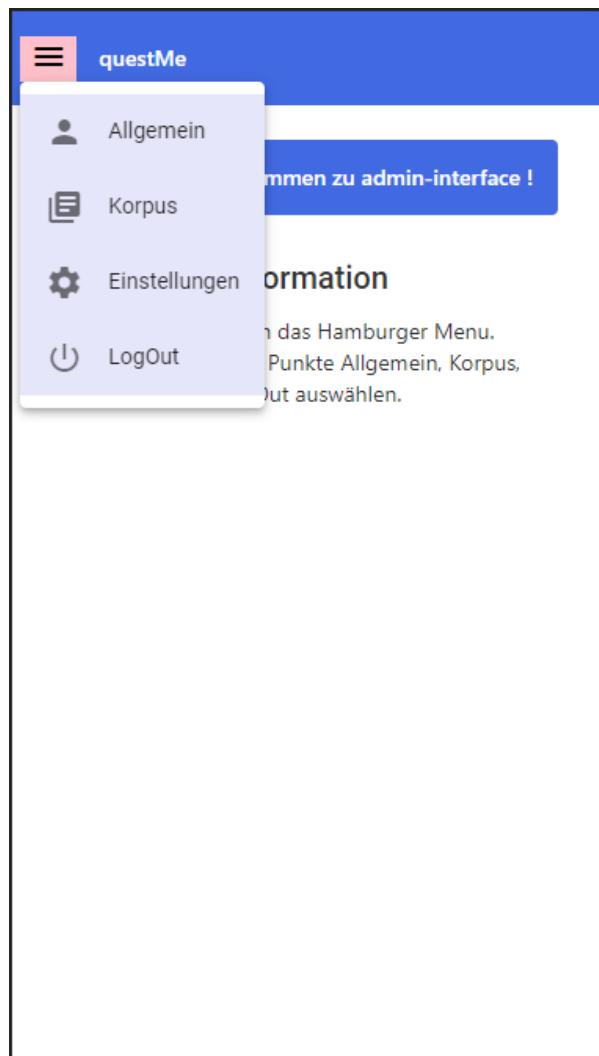


Abbildung 24: Admin Interface Hamburgermenu

### **Admin Interface Infoseite**

Hier sieht man das ausgeklappte Hamburgermenü auf der Infoseite.

## **4 Usability Test**

Hier werden wir unser Vorgehen des Usability Tests beschreiben und die ausführliche Durchführung des Tests.

### **4.1 Kriterien für Usability**

Unsere Kriterien für Usability, die uns wichtig sind und für unser User Interface essentiell sind, werden hier kurz erläutern.

#### **4.1.1 Responsive Webdesign**

Wir möchten unseren ChatBot auf allen Geräten ohne Probleme ausführen lassen. Das heißt, dass wir auch auf mobilen Geräten unsere Software im Browser laufen lassen wollen. Die mobile Seite sollte dann auch auf die wichtigsten Elemente beschränkt werden, damit der Benutzer es leichter hat die Icons treffsicher mit ihren Fingern zu benutzen.

#### **4.1.2 Gute Lesbarkeit**

Unsere Anwendung sollte leicht zu lesen sein, weil das Lesen auf dem Bildschirm grundsätzlich schwieriger ist. Deswegen sollten wir auf jeden Fall auf Textgröße und Kontrastreiche Farben achten. Die Textgröße sollte mindestens 12pt haben. Auch sollten wir lange Textblöcke und Schachtelsätze vermeiden.

#### **4.1.3 Gute Navigation**

Eine übersichtliche und eine verständliche Navigation ist das wichtigste bei einer Webseite. Eine gute Navigation verhindert Verwirrung und unterstützt den Benutzer zu seinem Ziel zu gelangen. Bei der Navigation sollte man darauf achten, dass alle Verlinkungen funktionieren.

#### **4.1.4 Schnelle Ladezeiten**

Die Webseite sollte ihre Inhalte schnell laden und keine großen Verzögerungen aufzeigen. Sie sollte bei ungefähr drei Sekunden liegen, um keine User zu verlieren.

#### **4.1.5 Interessantes Design**

Eine Konsistente Einhaltung von bestimmten Farben ist sehr wichtig. Der erste Eindruck von einer Webseite zeigt schon, ob die User die Webseite benutzen möchten

oder nicht. So können Benutzer durch Bilder mit schlechter Qualität oder zu grellen Farben abgeschreckt werden. Pop-Ups sollten in Grenzen gehalten werden oder vermieden werden.

## 4.2 Unser Usability Test

Hier beschreiben wir welche Methode wir zum Testen unserer Usability Kriterien benutzen möchten.

### 4.2.1 Remote User Testing

Zuerst planen wir was wir testen möchten und dann, wie wir testen möchten. In unserem Fall haben wir uns für remote testing entschieden. Als nächstes überlegen wir uns besondere Tasks, die der User absolvieren muss, um zu erkennen, ob unsere Kriterien eingehalten werden und was wir Verbessern sollten. Die Szenarien sollten so einfach und realistisch sein, dass der Benutzer es leicht durchführen kann. Dann müssen wir auch Tester finden, welche in unserer Zielgruppe passen.

### 4.2.2 Durchführung der Remote Usability Testsitzung

Der erste Schritt beinhaltet den Usability-Testplan. Dieser beinhaltet den Zweck des Tests, die Kosten und Zeiteinschätzung zur Durchführung des Tests, das Testskript mit den Usability-Testaufgaben und die Rekrutierung der Testteilnehmer.

Bei der Rekrutierung werden die User so ausgesucht, dass am besten alle Zielgruppen gedeckt sind. Jeder Testteilnehmer/in wird einzeln durch das Usability Test durchgeführt. Geplant sind insgesamt vier bis sechs Teilnehmer. Nach dem Aussuchen der Testteilnehmer wird am Tag des Tests die Einverständniserklärung für die Teilnahme und Datenschutz von den Usern unterschrieben. Die Einverständniserklärung sollte also schon vorbereitet sein. Auch die Tasks werden von vorneherein mitbestimmt. Nach dem Unterschreiben werden die Testteilnehmer mittels eines Briefings benachrichtigt, wie der Test abläuft und was zu beachten ist. Außerdem wird ausdrücklich auch vermittelt, dass es kein Test ist, um ihre Leistungsfähigkeit zu beurteilen, sondern dient nur für die Weiterentwicklung und zur Bewertung unserer Software.

Als nächstes findet die Pre-Session statt. In dieser Session finden wir heraus, welche Erfahrungen die Testteilnehmer mit dem zu testenden System hat und welche Interessen er vertritt und was er von Chatbots hält. Nach der Pre-Session werden die Testteilnehmer gebeten, laut-denkend ihre Testaufgaben durchzuführen. Der Moderator sollte still zuhören und beobachten. Ganz wichtig ist es nicht bei Schwierigkeiten einzugreifen, weil es sonst den Test manipuliert.

Eine Aufgabe könnte sein den Testteilnehmer zu bitten, dass Sie den Chatbot dazu bringen eine Information zu vermitteln, welche die Teilnehmer haben möchten. Der Testteilnehmer könnte Fragen stellen, welche der Chatbot kennt oder auch nicht. Der Moderator schreibt sich auf, wie der Testteilnehmer auf sein Ergebnis kommt, oder auch gewisse Schwierigkeiten zeigt und bei Aufgaben stockt. Es könnte sein, dass man dann den Korpus erweitern oder anpassen müsste. Alles wird gründlich dokumentiert, während der Testteilnehmer laut-denkend seine Aufgaben erledigt.

Bei uns ist der Moderator der Protokollant, da es remote abläuft. Wir überlegen außerdem die Session aufzunehmen.

Abschließend, in der Post-Session, werden dann die Teilnehmer auf den Gesamteindruck des zum testenden System befragt. Es ist sehr wichtig eine ausführliche Dokumentation zu schreiben, um jeden Eindruck und jedes Vorgehen festzuhalten.

Nach der Usability-Testsitzung werden die Befunde zusammengeschrieben und die Ergebnisse werden ausgewertet. Anschließend wird der Usability-Testbericht geschrieben. Der Usability-Testbericht beinhaltet nicht nur die Usability-Probleme, sondern auch die gelungenen Usability-Befunde.

# 5 User Stories

In diesem Kapitel haben wir unsere User Stories gesammelt.

## 5.1 Struktur der User Stories

Als <Akteur> möchte ich <Funktion>, um <Nutzen> zu erreichen.

## 5.2 User Stories Version 1

Hier listen wir unsere ersten Ideen auf, die wir erfüllen möchten.

### Chatfenster

- u10001. Als Nutzer möchte ich eine Nachricht abschicken können, um mit dem Chatbot zu interagieren.
- u10002. Als Nutzer möchte ich eine Frage stellen können, um eine Antwort zu erhalten.
- u10003. Als Nutzer möchte ich eine Antwort dargestellt bekommen, um die Antwort lesen zu können.
- u10004. Als Nutzer möchte ich meine Nachrichten, von denen des Bots unterscheiden können, um zu erkennen, welche Nachricht die Antwort ist.
- u10005. Als Nutzer möchte ich darauf hingewiesen werden, wo ich zu Schreiben habe, um eine Nachricht verfassen zu können.

**Hinweis:** Ein Teil der Professoren übernimmt administrative Tätigkeiten des Chatbots. Dadurch ist mit dem erwähnten Administrator immer ein administrativ tätiger Professor gemeint.

## **Admin Interface: Allgemein**

- u20001. Als Admin möchte ich eine Möglichkeit haben, um zwischen den Seiten des Admin-Interfaces wechseln zu können.
- u20002. Als Admin möchte ich verschiedene optische Konfigurationen zur Auswahl haben, um mein ChatBot zu individualisieren.
- u20003. Als Admin möchte ich erkennen können, welche Option ich ausgewählt habe, um zu wissen, was aktuell ausgewählt ist.
- u20004. Als Admin möchte ich den Namen des ChatBots ändern, um meinen ChatBot zu individualisieren.
- u20005. Als Admin möchte ich ein Eingabefeld erkennen können, um zu wissen, wo ich etwas eingeben kann.
- u20006. Als Admin möchte ich die ausgewählte Seite des Admin-Interfaces in einer anderen Farbe sehen, um zu erkennen auf welcher Seite ich bin.

## **Admin Interface: Korpus**

- u30001. Als Administrator möchte ich eine Liste mit allen Fragen und Antworten, um einen Überblick über den Korpus zu haben.
- u30002. Als Administrator möchte ich einen neuen Eintrag hinzufügen, um neue Fragen und Antworten hinzuzufügen zu können.
- u30003. Als Administrator möchte ich eine Möglichkeit zum Hinzufügen von Fragen, um neue Fragen hinzuzufügen zu können.
- u30004. Als Administrator möchte ich eine Möglichkeit zum Hinzufügen von Antworten, um neue Antworten hinzuzufügen zu können.
- u30005. Als Administrator möchte ich eine Möglichkeit zum Entfernen von Fragen, um Fragen entfernen zu können.
- u30006. Als Administrator möchte ich eine Möglichkeit zum Entfernen von Antworten, um Antworten entfernen zu können.
- u30007. Als Administrator möchte ich eine Möglichkeit einen Eintrag „Fragen und Antworten“ bearbeiten zu können, um den Eintrag zu ändern.

## **Node.js Allgemein**

- u40001. Als Nutzer möchte ich eine bidirektionale Kommunikation zwischen dem Client und Server, um direkt mit dem Bot kommunizieren zu können.
- u40002. Als Nutzer möchte ich, dass der ChatBot meinen Kontext versteht, um mit dem Bot nach Kontext zu chatten.
- u40003. Als Administrator möchte ich die Möglichkeit den Korpus des ChatBots persistent zu speichern, um auf den Korpus zuzugreifen zu können.
- u40004. Als Nutzer möchte ich, dass der ChatBot über eine Webadresse erreichbar ist, um mit dem ChatBot online zu kommunizieren.

## **KeyCloak**

- u50001. Als Admin möchte ich mich in KeyCloak einloggen können, um es zu verwalten.
- u50002. Als Admin möchte ich mich in das Admin-Interface einloggen können, um die Einstellungen des Chatbots zu verwalten.
- u50003. Als Hochschulangehöriger möchte ich mich mit dem Shibboleth SSO der Hochschule einloggen, um relevante Daten mitzuteilen.
- u50004. Als Admin möchte ich schnellen Zugriff auf das KeyCloak-Webinterface über das Admin-Interface, um Zeit zu sparen.
- u50005. Als Admin möchte ich neue Nutzergruppen erstellen, um zielgerichteter Fragen beantworten zu können.
- u50006. Als Admin möchte ich einer Nutzergruppe einen neuen Fragensatz zuweisen, um die möglichen Fragen für diese Gruppe zu erweitern.
- u50007. Als Admin möchte ich einen, zu einer Nutzergruppe zugewiesenen, Fragensatz entfernen, um möglichen Fragen für diese Gruppe einzuschränken.
- u50008. Als Admin möchte ich Nutzer verwalten, um bei Bedarf Änderungen vorzunehmen.
- u50009. Als Admin möchte ich die Login-Seite anpassen, um sie nach meinen Vorstellungen zu ändern.

# 6 Use Cases

Hier werden wir unsere User Stories zu den einzelnen Zielgruppen auflisten. Wir haben die Zielgruppen nicht registrierte Nutzer, Professor und Student.

## 6.1 Struktur der Use Cases

### Use Case:

Als <Akteur> möchte ich <Funktion>, um <Nutzen> zu erreichen.

### Akzeptanzkriterien:

Szenario:

kurze Beschreibung des Szenarios

Wenn ich ...

und ...

Dann ...

## 6.2 Student

### Use Case:

Als Student möchte ich meine Informationen von meinem Stundenplan abrufen können, um meine Vorlesungen einsehen zu können.

### Akzeptanzkriterien:

Szenario:

Der angemeldete Student fragt nach, wann die Vorlesung Virtuelle Realität stattfindet.

Wenn ich nach der Uhrzeit von der Vorlesung Virtuelle Realität frage und mich im Chatfenster befindet.

Dann bekomme ich meine Vorlesungsinformationen zu Virtuelle Realität.

## **6.3 Unregistrierter Nutzer**

### **Use Case:**

Als unregistrierter Nutzer, möchte ich Zugang zur Chatbot Chat Seite haben, damit ich einen Smalltalk mit dem Chatbot führen kann.

### **Akzeptanzkriterium:**

Szenario:

Als unregistrierter Nutzer, möchte ich Zugang zur Chatbot Chat Seite haben, damit ich einen Smalltalk mit dem Chatbot führen kann.

Wenn ich ein unregistrierter Nutzer bin und mich auf der Chatbot Chatseite befinde und in das Chatfenster Fragen schreibe und diese abschicke.

Dann antwortet der Chatbot auf meine Fragen mit allgemeinen Antworten.

## **6.4 Professor (Admin)**

### **Use Case:**

Als Professor möchte ich den Korpus des Chatbots um eine Frage und Antwort Möglichkeit erweitern, um den Nutzern ein größeres Repertoire zu bieten.

### **Akzeptanzkriterium:**

Szenario:

Der angemeldete Professor möchte eine neue Frage-Antwort Möglichkeit hinzufügen.

Wenn ich mich auf der Korpus-Seite des Admin-Interfaces befinde und auf den Button zum Hinzufügen einer Frage-Antwort Möglichkeit drücke. Dann kann ich eine neue Frage und eine dazugehörige Antwort eintippen.

## **7 Zielgruppen**

In diesem Abschnitt Listen wir unsere drei Zielgruppen: Nicht registrierte Nutzer, Student und Professor. Zu den Zielgruppen werden wir jeweils, ihre Probleme, was ihre geforderten Eigenschaften sind beschreiben und was bei unserer Software für Sie einzigartig ist.

### **7.1 Zielgruppe: Nicht registrierte Nutzer**

#### **Probleme:**

- Interesse an einem Smalltalk mit einem Chatbot
- Möchten einfache Fragen beantwortet bekommen
- Haben bisher mit sehr monoton antworteten Chatbots gechattet

#### **Eigenschaften**

- Nicht unbedingt technikaffin
- Möchten eine leichte Bedienung
- Haben sehr wenig Erfahrung mit Chatbots
- Interesse an einem Smalltalk
- Möchten unterhalten werden
- Kennen WhatsApp, Skype, etc.

#### **Alleinstellungsmerkmal / Einzigartigkeit:**

- Möchten sich sehr schnell in die Chatoberfläche einfinden
- Möchte ein einfaches und simples Userinterface
- Möchten, dass der Chatbot wie ein Mensch mit ihnen chattet

## **7.2 Zielgruppe: Student**

### **Probleme:**

- Studenten finden ihre Raumnummer nicht mehr
- Studenten wissen nicht, wann Ihre Vorlesung stattfindet
- Studenten müssen immer ihren Stundenplan einsehen und verlieren dabei Zeit
- Studenten müssen immer manuell nach Informationen zu ihren Kursen suchen

### **Eigenschaften**

- Studenten möchten alles einfacher
- Keine komplizierten Umwege
- Einfache Bedienung
- Gutes bzw. ansprechendes Design
- Gezielte Antworten auf bestimmte Fragen

### **Alleinstellungsmerkmal / Einzigartigkeit:**

- Studenten können mehrere Fragen stellen
- Eine leichte, aber ansprechende Bedienung des Chats
- Bekannte Chatoberfläche
- Keine komplizierten Umwege

## **7.3 Zielgruppe: Professor**

### **Probleme:**

- Professoren finden ihre Raumnummer nicht mehr
- Professoren wissen nicht, wann Ihre Vorlesung stattfindet
- Professoren müssen immer ihren Stundenplan einsehen und verlieren dabei Zeit
- Professoren wissen nicht, ob ein Raum frei ist, wenn sie sich z.B. einen größeren suchen müssen
- Professoren wissen nicht, wann der Prüfungstermin für ihre Vorlesung ist
- Professoren wollen ihren Studenten Informationen übermitteln
- Professoren wollen ihren Studenten bei Problemen helfen

### **Eigenschaften**

- Professoren möchten alles einfacher
- Keine komplizierten Umwege
- Einfache Bedienung
- Gutes bzw. ansprechendes Design
- Gezielte Antworten auf bestimmte Fragen

### **Alleinstellungsmerkmal / Einzigartigkeit:**

- Professoren können mehrere Fragen stellen
- Eine leichte, aber ansprechende Bedienung des Chats
- Bekannte Chatoberfläche
- Keine komplizierten Umwege
- Professoren übernehmen die Rolle des Admins
- Professoren können die Einstellungen des Chatbots verwalten
- Professoren können den Korpus um Fragen und Antworten erweitern

# **8 Technologien**

In diesem Kapitel soll alles zum Thema Technologien zusammengefasst sein. Zusätzlich sollen Schaubilder und Diagramme das Verständnis für die Technologien erleichtern. Sowie Beschreibungen und Erklärungen zu der Wahl der einzelnen Technologien.

## **8.1 Kriterien für die Technologien**

Bei der Wahl der Technologien haben wir nach Möglichkeit LTS Versionen gesucht. Damit wir eine Applikation erstellen können, die für lange Zeit Sicherheitsupdates bekommt. Zusätzlich haben wir als Ziel, dass die Software als insgesamtes Paket sehr lange stabil läuft und dadurch Ausfälle minimiert werden.

## **8.2 Technologie Vergleich**

In diesem Bereich wollen wir unsere Design Entscheidung für eine Technologie durch einen Vergleich darstellen. Dafür haben wir für die Themen "Vergleich zwischen Angular und Vue.js" und "Vergleich zwischen MongoDB und PostgreSQL" eine Tabelle erstellt.

### 8.2.1 Vergleich zwischen Angular und Vue.js

Als wir uns für ein Framework zur Entwicklung des Frontend entscheiden mussten, kamen Angular und Vue.js in unsere engere Auswahl. Damit wir uns einen besseren Überblick über die Eigenschaften dieser Kandidaten verschaffen und ihre Vor- und Nachteile besser abwägen können, haben wir uns diese Tabelle erstellt.

	<b>Angular</b>	<b>Vue.js</b>
<b>Framework, Library, Platform</b>	Entwicklungsplattform (Development platform)	Progressive Framework
<b>Gründer</b>	Google	ehemaliger Google Mitarbeiter
<b>Technology Typ</b>	MVC Framework	MVVM Framework
<b>Programmiersprache</b>	TypeScript	JavaScript
<b>Performance</b>	niedrig	hoch
<b>Größe</b>	500 kB	80 kB
<b>Lernkurve</b>	Eine steile Lernkurve	Eine geringe Lernkurve
<b>Dokumentation</b>	vorhanden	vorhanden
<b>Datenbindung</b>	Bi-directional	Bi-directional
<b>Rendering</b>	beim Client	beim Server
<b>Code reuse</b>	möglich	Ja, CSS und HTML
<b>Skalierbarkeit</b>	sehr hoch	hoch
<b>Testbarkeit</b>	mit einem Tool	mehrere Tools benötigt
<b>Vollständige Web App</b>	Kann als standalone Basis verwendet werden	benötigt Third Party Tools
<b>Lizenz</b>	MIT License	MIT License

Tabelle 1: Vergleich zwischen Angular und Vue.js

Unsere Entscheidung fiel schlussendlich auf Angular. Da es bereits länger auf dem Markt ist und auch einen höheren Marktanteil hat, ist es generell schon interessant es sich mal anzuschauen und damit zu arbeiten. Hinzu kommt die sehr gute Testbarkeit, für die man lediglich ein einziges Tool benötigt. Die sehr hohe Skalierbarkeit und mögliche Wiederverwendbarkeit des Codes verspricht viele Möglichkeiten, auch für die Zukunft. Alle Teammitglieder haben bereits Erfahrung mit JavaScript als Programmiersprache, aber keine mit TypeScript. Der Umgang mit TypeScript muss also erst erlernt werden. Ein großer Nachteil, der beim Blick auf die Tabelle sofort ins Auge sticht, ist aber die Performance. Allerdings haben wir sie nur auf niedrig eingestuft, da lediglich am Anfang viel mitgeladen werden muss. Bei größeren Projekten bietet Angular mehr Stabilität und Performance als seine Konkurrenz in diesem Vergleich.

### 8.2.2 Vergleich zwischen MongoDB und PostgreSQL

Ähnliche Gedanken wie für unser Frontend mussten wir uns auch für unser Backend stellen. Genauer, welche Datenbank wollen wir verwenden? Da wir in der Datenbank den Korpus unseres Chatbots in Form von JSON-Dateien speichern wollen, haben wir uns zwei der populärsten Lösungen rausgesucht und diese in einer Tabelle gegenübergestellt.

	<b>MongoDB</b>	<b>PostgreSQL</b>
<b>Primäres Datenbankmodell</b>	Dokumentenorientiert	Relationales DBMS
<b>Entwickler</b>	MongoDB, Inc	PostgreSQL Global Development Group
<b>Datenschema</b>	Schemafrei (NoSQL)	Ja
<b>Programmiersprachen</b>	JavaScript + 28 weitere	JavaScript + 9 weitere
<b>Query Language</b>	MQL	SQL
<b>Maximale Dateigröße</b>	16 MB	1 GB
<b>Lernkurve</b>	Eine geringe Lernkurve	Eine geringe Lernkurve
<b>Dokumentation</b>	vorhanden	vorhanden
<b>Skalierbarkeit</b>	sehr hoch	hoch
<b>Lizenz</b>	Open Source	Open Source

Tabelle 2: Vergleich zwischen MongoDB und PostgreSQL

Wir haben uns schlussendlich dafür entschieden, MongoDB zu nutzen. Da es bereits ein schemafreies und Dokumentenorientiertes Datenbankmodell ist, kommt uns das sehr gelegen. Beide Datenbanksysteme haben eine geringe Lernkurve und eine ähnlich gute Skalierbarkeit. SQL ist zwar allen Teammitgliedern bereits bekannt, aber dank der umfangreichen Dokumentation von MongoDB sollte es leichtfallen, MQL zu erlernen. Ein Nachteil, der sofort auffällt, ist die Maximale Dateigröße von gerade einmal 16 MB. Da wir aber nicht davon ausgehen, dass unsere Dateien diese Größe erreichen werden, stellt das für uns kein Hindernis dar.

## 8.3 Technologie Versionen

Hier sollen die einzelnen Technologien im Einzelnen beschrieben werden. Sowie ihre geplante Funktion in unserem Projekt. In der unten angegebenen Liste soll ein Überblick über die im Projekt verwendeten Technologien bereitgestellt werden.

	Node.js 16.13.0 LTS
	Angular 13.0.1
	Socket.io 4.3.2
	MongoDB Community Edition 5.0.3
	Keycloak 15.0.2
	NLP.js 4.22.2

Tabelle 3: Technologie Liste

### 8.3.1 Node.js 16.13.0 LTS

Als Basis für Angular, den Chatbot und Keycloak nutzen wir eine LTS Version von Node.js. Auf dem Node.js Server verwenden wir zusätzlich die Express Erweiterung, um eine Kommunikation zu den Servern herstellen zu können. Für uns war sehr wichtig, dass wir eine sehr stabile Basis für die Anwendung, die wir entwickeln haben. Deswegen haben wir uns ebenfalls informiert, ob alle geplanten oder in Frage kommenden Technologien mit der LTS Version kompatibel sind. Nach unserem derzeitigen Stand empfehlen die Entwickler der Bibliotheken und Frameworks eine LTS Version von Node.js zu verwenden. Bildquelle: *Node.js Icon*, [o. D.]

### **8.3.2 Angular 13.0.1**

Wir verwenden Angular für die Frontend-Entwicklung. Angular ist die Basis, um das Chat- und das Admin-Interface zu erstellen. Um unsere UIs noch effektiver zu gestalten, werden wir vereinzelt auf Angular Material zurückgreifen. Dort gibt es zu häufig genutzten UI Komponenten "Code Snippets", die ausführlich getestet wurden. Bildquelle:*Angular Icon*, [o. D.]

### **8.3.3 Socket.io 4.3.2**

In unserem Projekt nutzen wir Socket.io, um eine bidirektionale Kommunikation zwischen dem Bot und der Benutzeroberfläche herzustellen. Es ermöglicht uns, dass ein User mit dem Chatbot ohne merkbare Verzögerung chatten kann. Bildquelle:*Socket.io Icon*, [o. D.]

### **8.3.4 MongoDB Community Edition 5.0.3**

Wir haben uns für die Datenbank mongoDB entschieden, da es ein sehr einfaches Datenbank Modell bereitstellt. MongoDB ist eine NoSql Datenbank, wodurch auch der Verwaltungsaufwand der Datenbank minimiert wird. In der Datenbank wird der Korpus des Chatbots gespeichert. Bildquelle:*MongoDB Icon*, [o. D.]

### **8.3.5 Keycloak 15.0.2**

In unserem Projekt verwenden wir Keycloak, damit sich der Admin sicher in das Admin-Webinterface einloggen kann. Zusätzlich werden wir das Shibboleth Single-Sign-On Verfahren von der Hochschule integrieren. Damit Hochschulangehörige die Möglichkeit haben sich mit ihrem Hochschul-Account einloggen zu können. Dadurch können wir zusätzlich begrenzen welche Personengruppen auf welche Daten Zugriff haben. Bildquelle:*Keycloak Icon*, [o. D.]

### **8.3.6 RegEx**

Wir haben in unserem Projekt anfangs aus Sicherheitsgründen RegEx eingeplant. RegEx dient in unserem Projekt als Fallback. Um ein Scheitern des Projektes zu verhindern, falls NLP.js sich nicht in unser Projekt integrieren lassen sollte. Damit der Chatbot dennoch Begriffe und Sätze verstehen kann. Nach unserem derzeitigen Stand brauchen wir nicht auf eine RegEx Implementation zurückzugreifen, da NLP.js sich sehr gut in unser Projekt integrieren lässt.

Als Beispiel für den Satz "Wie geht es dir?". Müsste ein User oder Admin folgenden RegEx Befehl integrieren:

(Wie|wie)\s\*geht\s\*es\s\*dir\s\*\?

Damit man nach "Wie/wie geht es dir ?" mit und ohne Leerzeichen prüfen kann.

Bildquelle:*Regex Icon*, [o. D.]

### **8.3.7 NLP.js 4.22.2**

Damit wir NLP.js nutzen können benötigen wir eine LTS Version von Node.js. Diese Begrenzung haben wir zusätzlich in die Technologie Entscheidung einbezogen. Indem NLP.js ab der Version 4 und höher modular ist. Wird uns ermöglicht leichter eigene Module/Plugins für das Framework zu schreiben. Dadurch können wir einen ChatBot mit verbesserter Spracherkennung gestalten. Bildquelle:*NLP Icon*, [o. D.]

## 8.4 Technologie Diagramme

Hier werden die Zusammenhänge der Technologien in Form von verschiedenen Diagrammen vorgestellt.

### 8.4.1 UML Komponentendiagramme

Hier werden die UML Verteilungsdiagramme von der Server- und Client-Seite dargestellt.

### 8.4.2 UML Komponentendiagramm: Client

Hier wird ein UML-Komponentendiagramm Client dargestellt.

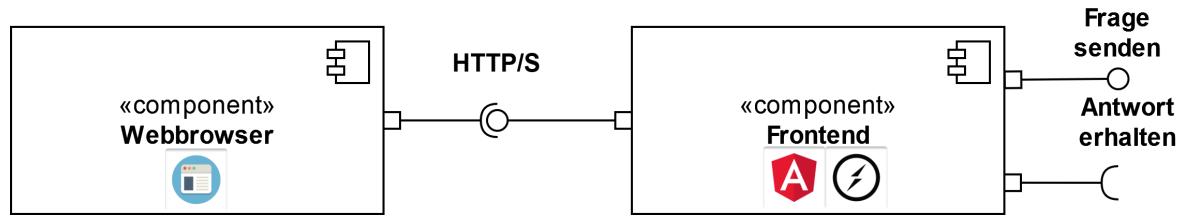


Abbildung 25: UML Komponentendiagramm Client

In der Abbildung 25 sieht man den Webbrower und das Frontend auf der Client Seite. Das Frontend wird mit Angular entwickelt. Das besitzt aber selbst auch einen node.js Server. Per Socket.io Client wird dann eine Verbindung zwischen dem Backend produziert. Per HTTP/S wird eine Verbindung zwischen Webbrower und dem Frontend entwickelt. Das Frontend component sendet eine Frage und erhält anschließend eine Antwort vom Backend.

### 8.4.3 UML Komponentendiagramm: Server

Hier wird der UML-Komponentendiagramm Server dargestellt.

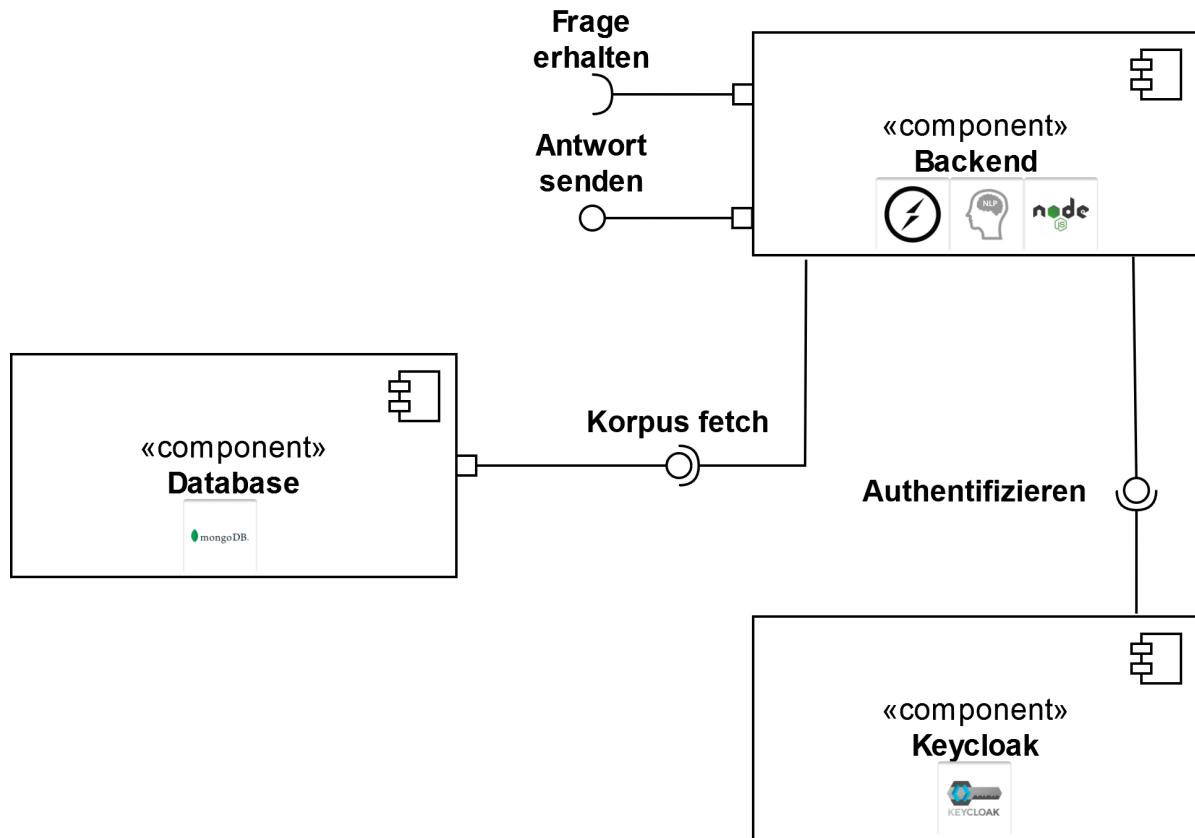


Abbildung 26: UML Komponentendiagramm Server

In der Server-Seite befindet sich das Backend, die Datenbank und KeyCloak. Im Backend befindet sich der Socket.io Server, NLP und ein node.js Server. Das Backend erhält die Frage vom Frontend und schickt daraufhin eine Antwort mit Hilfe des Socket.io Servers zurück. Von der Datenbank wird der Korpus an das Backend vermittelt. KeyCloak dient zur Authentifizierung und hat ebenfalls einen node.js Server.

#### 8.4.4 UML Komponentendiagramm

Hier sieht man die ganze Darstellung von dem Komponentendiagramm mit Server- und Client-Seite.

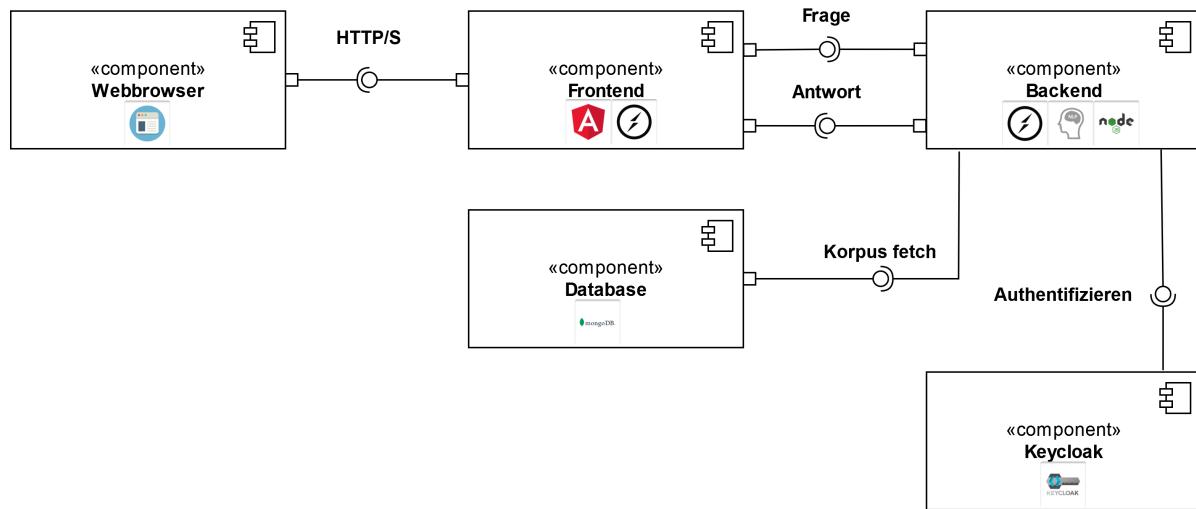


Abbildung 27: UML Komponentendiagramm

In dieser Abbildung 27 sieht man die Client- und Server-Seite. Als Austausch zwischen Frontend und Backend haben wir eine Frage und eine Antwort dargestellt. Des Weiteren besitzt das Frontend, das Backend und Keycloak einen node.js Server.

#### 8.4.5 UML Verteilungsdiagramm

Hier sieht man die ganze Darstellung unseres Verteilungsdiagramms

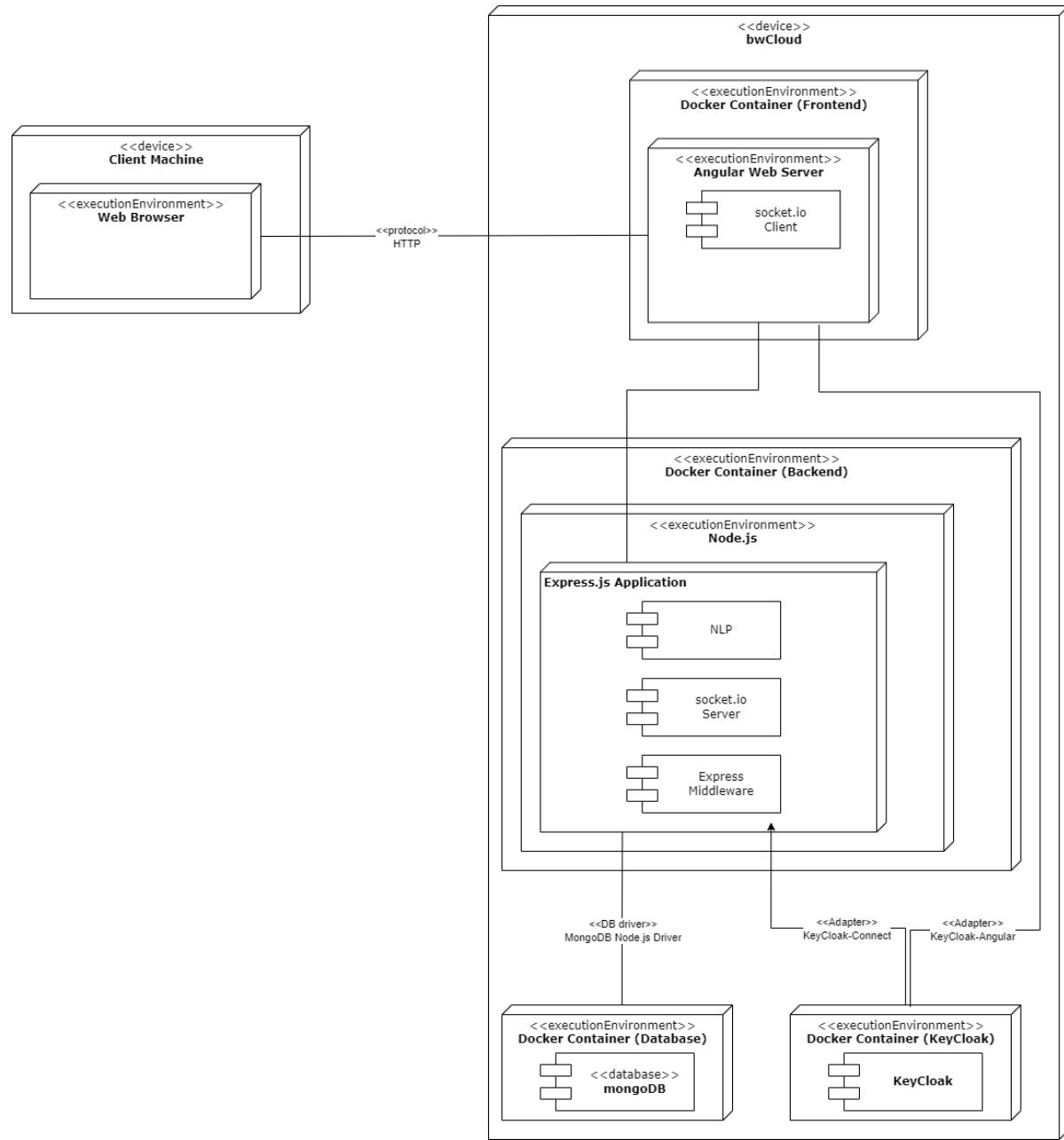


Abbildung 28: UML Verteilungsdiagramm

In unserem Verteilungsdiagramm kann man sehen, wie die verschiedenen Technologien verschachtelt und wie sie miteinander verbunden sind. Außerdem kann man sehen, was wir in unsere einzelnen Docker Container hineinlegen.

## 8.5 Datenbank

In folgendem soll Aufschluss über die Struktur der Datenbank des ChatBots gegeben werden. Die Datenbank wird in einer NoSql mongoDb Datenbank gehalten.

### 8.5.1 Datenhaltung

Die Daten, die in der mongoDb Datenbank gespeichert sind, sollen über ein Webinterface verändert werden können. Wenn der Chatbot gestartet wird lädt der Chatbot den Korpus aus der mongoDb. Sollten die Daten der Datenbank verändert worden sein, dann muss der Chatbot den Korpus erneut laden. Damit die Veränderungen der Datenbank auch beim Chatbot geändert werden.

### 8.5.2 ER Diagramme

Hier werden die ER Diagramme für die Datenbank aufgeführt. Zusätzlich wird erklärt welche Daten bei den Entities enthalten sind.

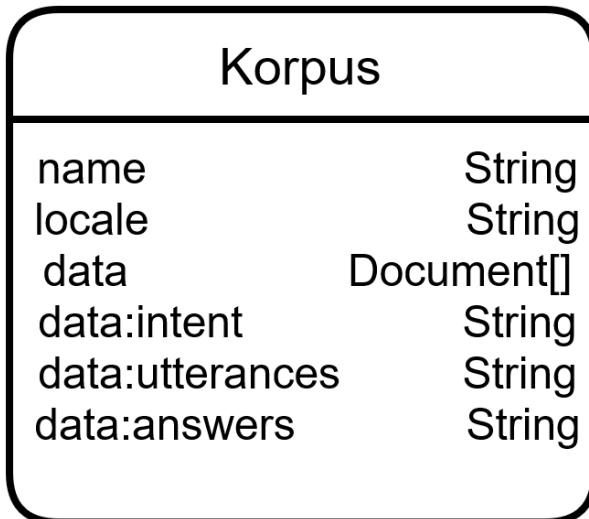


Abbildung 29: ER Diagramm Korpus

Der Korpus besteht aus name, locale und data. Der "name" ist der Name des Korpus. Das "locale" gibt an in welcher Sprache der Korpus verfasst wurde. Das "data" ist gefüllt mit Intents, die benötigt werden, um einschätzen zu können in welchem Kontext der ChatBot antworten soll. Der Intent besteht aus Utterances (Äußerung/ Frage des Nutzers) und Antworten. Die Utterances sind die möglichen Fragen des Nutzers und die Antworten sind die möglichen Antwortmöglichkeiten des Chatbots. Ein Korpus hat sehr viele Intents, die den Wissensschatz des ChatBots abbilden.

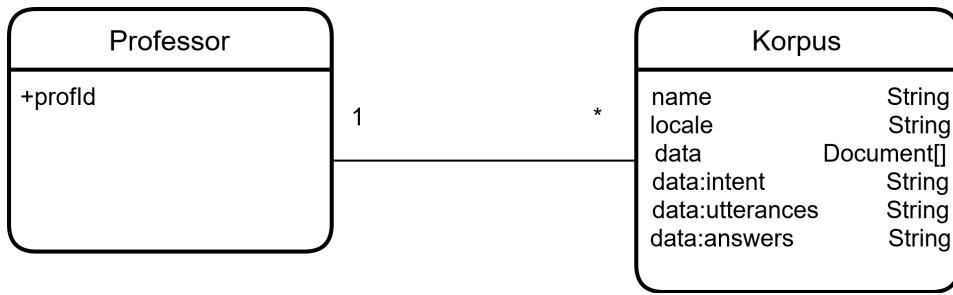


Abbildung 30: ER Diagramm Professor

In der Abbildung 30 soll dargestellt werden, dass ein Professor auf mehrere Korpusse zugreifen kann. Der Professor wird mit einer "profID" ausgestattet, damit man bestimmen kann welche Korpusse ihm zugeordnet sind.

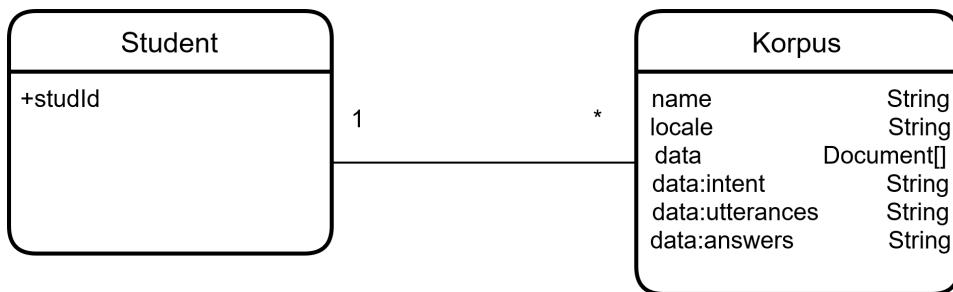


Abbildung 31: ER Diagramm Student

In der Abbildung 31 soll dargestellt werden, dass ein Student auf mehrere Korpusse zugreifen kann. Der Nutzer bekommt eine "studId" damit man bestimmen kann, welche Korpusse für den Studenten bereitgestellt werden müssen.

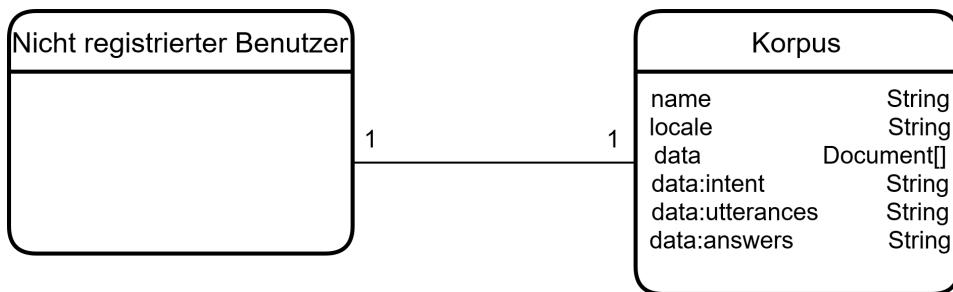


Abbildung 32: ER Diagramm Unregistrierter Nutzer

In der Abbildung 32 soll dargestellt werden, dass ein nicht registrierter Benutzer nur Zugriff auf einen Korpus hat. Für unregistrierte Benutzer soll nur der Korpus "Allgemein" zur Verfügung stehen.

## 8.6 Sicherheit

In diesem Kapitel geht es darum, wie wir uns vor zugriffen von außen schützen wollen. Dazu schauen wir uns an, welche Sicherheitsvorkehrungen wir sowohl im Frontend als auch im Backend getroffen haben.

### 8.6.1 Frontend

Im Frontend ist Keycloak so konfiguriert, dass man sich dort einloggen kann, wenn man möchte. Um die Seite des Chats aufrufen zu können muss man nicht eingeloggt sein. Anders sieht das beim Admin-Interface aus. Möchte man auf das Admin-Interface zugreifen, so muss man sich zuerst einloggen.



Abbildung 33: Frontend-Sicherheit

Kommt man jetzt noch nicht in den Admin-Bereich, dann hat wohl der benutzte Account nicht die benötigten Rechte. Jedem Account sind Rollen zugeordnet, wie zum Beispiel "Admin", "Student" und "Professor". Nur wenn man mit einem Account, welcher die Admin-Rolle besitzt eingeloggt ist, kann auf das Admin-Interface zugreifen.

### 8.6.2 Backend

Für das Backend ist Keycloak so konfiguriert, dass man sich nicht einloggen kann. Es dient nur zur Verifizierung von Tokens. Geschützt ist jeder mögliche Request an unsere Rest-API. Bei jeder Anfrage an die Rest-API muss ein valider Security Bearer Token im Header mitgeschickt werden. Ist dies nicht der Fall, so wird die Anfrage sofort abgelehnt. Sollte ein Token vorhanden sein, wird dieser vom Keycloak-Server geprüft. Ist der Token valide wird geprüft, ob der Nutzer, zu dem dieser Token gehört, über die nötigen Rollen verfügt. So kann auch niemand ohne gültigen Token zu einem Admin-Account über die Rest-API auf unsere Daten zugreifen, sie löschen oder ändern. Bei einer Anfrage vom Frontend wird automatisch der aktuelle Token des eingeloggten Nutzers mitgeschickt.

### **8.6.3 Angriffs-Szenarios**

#### **Szenario 1: API**

Ein Angreifer findet die Pfade zu unserer Rest-API heraus. Mithilfe dieser Information möchte er Zugriff auf unsere Daten in der Datenbank erlangen.

#### **Lösung:**

Da der Angreifer aber keinen gültigen Security Token hat, den er mitschicken kann, ist es ihm nicht möglich eine Anfrage an die API zu stellen.

#### **Szenario 2: API mit Token**

Der Angreifer hat einen gültigen Token in die Hände bekommen und sendet diesen nun mit in der Anfrage.

#### **Lösung:**

Selbst wenn der Token zu einem Account mit der Admin-Rolle gehören sollte, so muss der Angreifer sich trotzdem sehr beeilen, da die Token regelmäßig in kurzen Abständen erneuert werden.

#### **Szenario 3: Admin-Interface**

Der Angreifer möchte sich Zugang zum Admin-Interface verschaffen, indem er den Login-Prozess überspringt und direkt auf einen Pfad des Admin-Interfaces zugreift.

#### **Lösung:**

Da jeder Pfad des Admin-Interfaces durch Keycloak geschützt ist, wird für jeden Pfad einzeln geprüft, ob der Nutzer die nötigen Rechte besitzt.

## 9 Meilensteine

In der nachfolgenden Tabelle 4 sind alle einzelnen Meilensteine aufgelistet. Jeder Meilenstein wird noch etwas genauer weiter unten erklärt.

Recherche	<b>13.10.21</b>
Zwischenpräsentation	<b>22.10.21</b>
Implementation	<b>25.10.21</b>
MVP	<b>02.12.21</b>
Endpräsentation und Enddokumentation	<b>14.01.22</b>

Tabelle 4: Meilenstein Liste

### 9.1 Recherche 13.10.21

Während der Recherche haben wir alle Informationen gesammelt, die wir für das Projekt benötigen. Die Themen Node.js, Keycloak, Angular, Socket.io und NLP waren vorrangige Themen, um Risiken zu minimieren. Deswegen haben wir uns in dieser Phase möglichst ausführlich informiert und Bücher, Webadressen und weitere Materialien besorgt. Außerdem haben wir nach Möglichkeit alle Unklarheiten geklärt.

### 9.2 Zwischenpräsentation 22.10.21

Wir haben eine Woche früher (12.10.21) angefangen alle relevanten Themen zu sammeln, um Materialien für die Präsentation zu haben. Damit die Präsentation sehr interessant für alle Teilnehmer ist, haben wir die wichtigsten Themen optisch ansehnlich gestaltet. In unserer Zeitplanung ist auch die praktische Übung der Folien im Team eingeplant.

### **9.3 Implementation 25.10.21**

In der Implementierung wollen wir die recherchierten Materialien umsetzen und praktische Erfahrung sammeln. Während wir versuchen alle relevanten Informationen in einen MVP umzusetzen. Zusätzlich wird in dieser Phase ein Teil der Recherche in das L<sup>A</sup>T<sub>E</sub>X-Format übertragen.

### **9.4 MVP 02.12.21**

Der MVP ist unser angestrebtes Ziel. Damit wir ein Produkt zum Präsentieren haben. Während wir unser angesammeltes Wissen in die Praxis umsetzen, versuchen wir frühzeitig ein funktionierendes Produkt mit den Mindestanforderungen umzusetzen. Wir hatten vorerst geplant unseren MVP zum 03.12.21 zu liefern. Interessanterweise wurde später der Termin des MVP vom Professor auf den 02.12.21 gelegt. Wodurch wir unseren geplanten Zeitraum weiterhin nachverfolgen können und den zeitlichen Rahmen minimal korrigieren müssen. Demnach haben wir relativ gut eingeschätzt, bis wann der MVP fertig sein sollte.

### **9.5 Endpräsentation und Enddokumentation 14.01.22**

Für die Endpräsentation ist eine Woche früher (07.01.22) der Beginn der Erstellung der Präsentation eingeplant. Ziel ist hierbei, dass ein Prototyp mit interessanten Features und Funktionen vorgestellt werden kann. Für die Enddokumentation werden wir ab dem Start der Implementation anfangen, alle wichtigen Informationen zu dokumentieren. Sehr wichtig ist hierbei für unser Team, dass wir alle kontinuierlich wichtige Themen zu unserem Projekt dokumentieren. Für unsere Dokumentation wählen wir das L<sup>A</sup>T<sub>E</sub>X-Format, da es uns hilft besser kooperativ über Gitlab zu arbeiten.

# 10 Zeitmanagement

In diesem Abschnitt wollen unsere Pläne zum Zeitmanagement des Projektes vorstellen.

## 10.1 Gantt-Diagramm

Im nachfolgenden Gantt-Diagramm sieht man, welche Tasks wir eingeplant haben. Wann wir mit diesen beginnen wollen und wann wir wird mit ihnen fertig sein wollen.

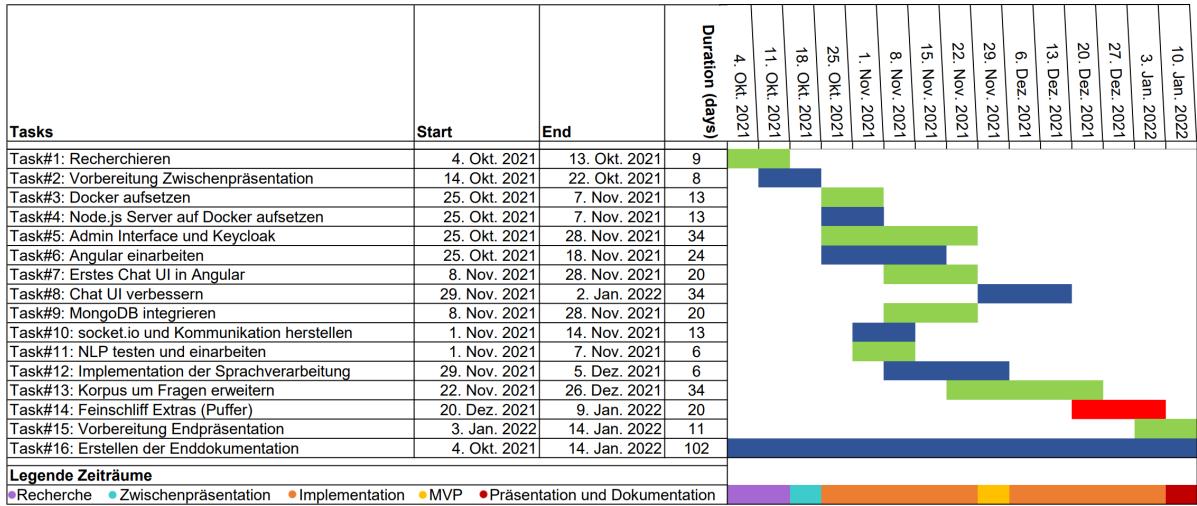


Abbildung 34: Gantt-Diagramm

In der Darstellung der Balken stellt eine Spalte, eine Woche dar, beginnend mit jedem Montag. Am unteren Ende des Diagramms sieht man, wie wir unsere Milestones eingeplant haben.

## 11 Risikoanalyse

Auf den folgenden Seiten haben wir in einer Tabelle möglicher Risiken für unser Projekt aufgelistet. Sowie Maßnahmen wie wir diese Risiken verringern möchten.

Risiko	Eintrittswahrscheinlichkeit	Auswirkung	Maßnahme
Team schafft es nicht schnell genug Typescript zu lernen	niedrig	hoch	Alle Teammitglieder beginnen frühzeitig sich mit Typescript zu beschäftigen
Team schafft es nicht Keycloak zu integrieren	niedrig	hoch	Alle Teammitglieder schauen sich rechtzeitig die Einführungsvideos von Herr Rößler zu Keycloak an
Das Team hat Schwierigkeiten das Userinterface in Angular zu entwickeln	niedrig	hoch	Das Team greift auf gut bewährte Designs zurück
Das Team hat Schwierigkeiten eine Verbindung mit socket.io herzustellen	niedrig	hoch	Das Team informiert sich rechtzeitig auf der Socket.io Website, wie eine Verbindung aufgebaut wird
Das Admin Interface lässt sich nicht flexibel genug anpassen	niedrig	niedrig	Das Team informiert sich rechtzeitig welche Möglichkeiten es in das UI einbauen möchte, um Flexibilität zu garantieren
Der ChatBot antwortet stark verzögert	niedrig	mittel	Das Team muss mit einplanen, dass der ChatBot in kleine saubere Module aufgeteilt wird

Tabelle 5: Risikoanalyse Tabelle Teil 1

Risiko	Eintritts-wahrscheinlichkeit	Auswirkung	Maßnahme
Hohe Latenz durch alle Komponenten	unwahr-scheinlich	mittel	Das Team muss den ChatBot testen, um z.B. Endlosschleifen zu verhindern
Der ChatBot hat Schwierigkeiten Sätze zu verstehen	mittel	mittel	Das Team muss bei einem Regex Ansatz mehrere Regex Befehle vordefinieren, um ein großes Spektrum abzudecken
Die Hardware des Kunden ist nicht kompatibel mit der Software	niedrig	hoch	Das Team muss frühzeitig mit dem Kunden klären für welche Hardware der ChatBot entwickelt werden soll
Das Team scheitert einen MVP zu entwickeln	niedrig	hoch	Das Team muss sehr früh mit der Implementierung beginnen und ausführlich genug recherchieren
Das Team scheitert rechtzeitig genug alle Technologien zu lernen	niedrig	hoch	Das Team recherchiert frühzeitig und versucht für alle möglichen Probleme Lösungen zu finden

Tabelle 6: Risikoanalyse Tabelle Teil 2

## 11.1 Fazit

Bei Beginn unseres Projektes haben wir mögliche Risiken aufgelistet, die in unserem Projekt auftauchen könnten und wie wir diese nach Möglichkeit verhindern möchten. Wir haben dabei herausgefunden, dass wir sehr viele Risiken haben, die von niedriger Eintrittswahrscheinlichkeit sind. Dennoch ist die Mehrheit der Risiken von hoher Auswirkung im Projekt. Als Beispiel, das Team schafft es nicht schnell genug TypeScript zu lernen. Die wichtigsten Risiken für uns waren die oben genannten Risiken zu Angular, Keycloak und Socket.io. Wenn eines dieser Risiken von unserem Team nicht ausgleichbar wäre, dann würden wir gar nicht die Möglichkeit haben einen MVP oder später ein fertiges funktionierendes Produkt abzuliefern. Mit den Tabellen 5 und 6 möchten wir für unsere Gruppe festhalten, welche Gedanken wir uns über die möglichen Risiken in unserem Projekt gemacht haben. Damit wir besser und effizienter unser Projekt vorantreiben können und vorbereitet sind auf mögliche Schwierigkeiten.

## 12 Installationshandbuch

In diesem Abschnitt wird die Installation von den einzelnen Komponenten ausführlich erklärt, damit die Software, nach dem herunterladen der questMe repository, erfolgreich gestartet werden kann.

### 12.1 Erste Schritte um die Software zum Laufen zu bringen

Hier werden die ersten Schritte erklärt, die ein Benutzer haben sollte um die Software ohne Probleme ausführen zu können.

#### 12.1.1 Setup to run a Angular project

Hier werden die ersten Schritte für das Einrichten von Angular vorgestellt. Bei mehr Fragen kann man auch auf das Angular Doc Setup zugreifen, <https://angular.io/guide/setup-local>.

RELEASE	STATUS	CODENAME	INITIAL RELEASE	ACTIVE LTS START	MAINTENANCE LTS START	END-OF-LIFE
v12	Maintenance LTS	Erbium	2019-04-23	2019-10-21	2020-11-30	2022-04-30
v14	Maintenance LTS	Fermium	2020-04-21	2020-10-27	2021-10-19	2023-04-30
v16	Active LTS	Gallium	2021-04-20	2021-10-26	2022-10-18	2024-04-30
v17	Current		2021-10-19		2022-04-01	2022-06-01
v18	Pending		2022-04-19	2022-10-25	2023-10-18	2025-04-30

Abbildung 35: node.js 16.03.0 LTS

Die node.js Version die in der Software benutzt wird ist die Long Time Support Version **16.03.0 LTS**, <https://nodejs.org/en/about/releases/>. Diese sollte der Nutzer auf seinen Rechner installiert haben. Außerdem sollte auch das npm client installiert sein. Zu prüfen ob diese vorhanden ist, gibt man in dem Terminal Window **npm -v** ein.

Der User sollte auch den Angular CLI installieren mit: **npm install -g @angular/cli**

#### 12.1.2 Docker Compose zum Laufen bringen

Der Benutzer sollte Docker Desktop installiert haben um die images zu pullen und den Docker compose zu starten.

# Install Docker Desktop on Windows

Estimated reading time: 9 minutes

## 1 Update to the Docker Desktop terms

Professional use of Docker Desktop in large organizations (more than 250 employees or more than \$10 million in annual revenue) requires users to have a paid Docker subscription. While the effective date of these terms is August 31, 2021, there is a grace period until January 31, 2022, for those that require a paid subscription. For more information, see the blog [Docker is Updating and Extending Our Product Subscriptions](#) and the [Docker Desktop License Agreement](#).

Welcome to Docker Desktop for Windows. This page contains information about Docker Desktop for Windows system requirements, download URL, instructions to install and update Docker Desktop for Windows.

## 1 Download Docker Desktop for Windows

[Docker Desktop for Windows](#)

## System requirements

Your Windows machine must meet the following requirements to successfully install Docker Desktop.

WSL 2 backend

Hyper-V backend and Windows containers

Abbildung 36: Docker Desktop Windows

Wir benutzen hier bei unserer Software die Windows Version vom Docker Desktop.

Da wir auf einem Docker arbeiten muss der User nur die docker images installieren um den Docker compose zu starten.

Dabei muss er ins Terminal Window und die Images pullen.

Die Images die der Benutzer pullen sollte.

- docker pull mongo:5.0.4-focal
- docker pull alpine:3.15
- docker pull node:16.13.0-alpine3.14
- docker pull openjdk:11

Nachdem die images gepullt sind kann der Nutzer dann mit **docker compose build** den docker compose erstellen, indem er es diese im Terminal eingibt.

Um den Docker hochzufahren muss der Benutzer im Terminal **docker compose up** eingeben.

Es könnten Schwierigkeiten beim builden entstehen, weil wir unsere Daten im mongoDb in einer file .db-data speichern. Um die existierenden Korpus zu löschen muss diese file gelöscht werden und das build erfolgt dann ohne Probleme.

### 12.1.3 Docker Desktop mit den Container

So sollten die Container aussehen und wenn der **docker compose up** geschieht leuchten die Container grün, wenn nichts schief gelaufen ist. Leuchten Sie orange oder rot ist ein Fehler aufgetreten.

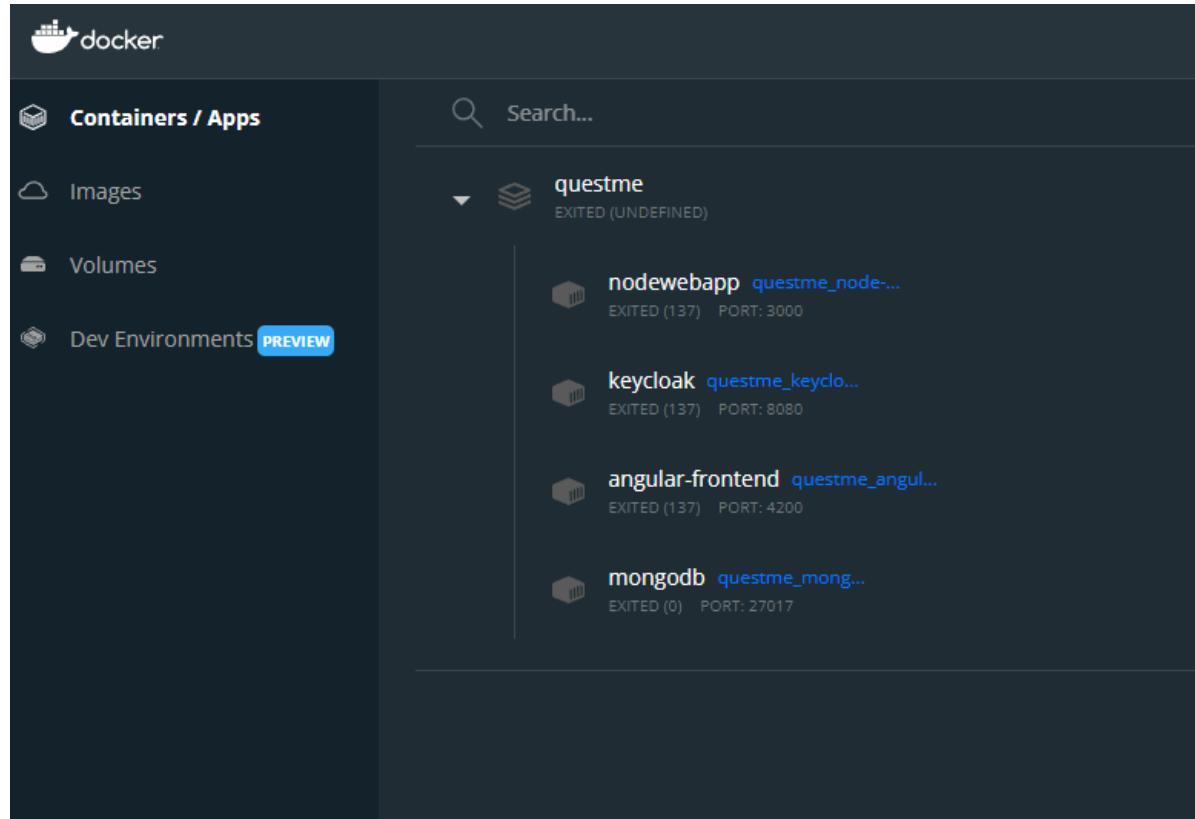


Abbildung 37: Docker Desktop Container

## **13 Aufteilung des Teams**

Hier werden die einzelnen Aufgaben, die wir bearbeitet haben in Issues mit dem Namen der Beteiligten aufgezählt.

### **13.1 Herr Ralf Zeller**

Hier werden die Issues von Herr Ralf Zeller aufgelistet.

1. Angular vs. React.js vs, Vue.js recherchiert
2. Die Zwischenpräsentationsfolien erstellt
3. Latex Grundstruktur der Dokumentation erstellt
4. Die Meilensteine in der Dokumentation erstellt
5. Die Risk Liste erstellt
6. Technologien in der Dokumentation eingetragen
7. Die Commits und die Labeling bestimmen (in welcher Sprache und in welcher Form diese geschrieben werden)
8. Die Datenbank für den Korpus erstellt
9. Ausgetestet ob NLP mit mongoDB benutzt werden kann
10. Datenbankstruktur (ERM) erstellt
11. Die Technologien in der Dokumentation aktualisiert
12. Den SRC von der "Literatur.bib" geprüft
13. Risikoanalyse geprüft und ein Review geschrieben
14. Meilensteine korrigiert
15. Die Technologien geprüft und die Liste neu gestaltet
16. Die Use Cases enummeriert
17. Die Dokumentation Tree im Gitlab gesäubert
18. Den UML Komponentendiagramm hinzugefügt
19. Den NLP Node.js Server von 14+ zu 16+ aktualisiert
20. Die Dokumentation korrigiert

21. Die Admin development branch gesäubert
22. Den ersten Commit umbennant
23. Docker Compose erstellt
24. NLP Test branch und presentation branch rebased
25. Chat/Admin UI, NLP und mongoDb kombiniert
26. Admin Interface in Docker Compose integriert
27. Funktionalität zu Admin Corpus Website hinzugefügt
28. Keycloak in Docker Compose integriert
29. Erste Version von der Dokumentation für die Abgabe hinzugefügt
30. Vesionen von allen Technologien überprüft
31. Team Building Retrospective gehalten
32. Corpus for Interna erstellt
33. Die main branch gesäubert
34. Hinzufüge-/ und Entfern-/ Button für die Intent Cards im Admin Interface implementiert
35. Die essentielle Funktion für die Rest API implementiert
36. Allgemein und Einstellungen im Admin Interface gearbeitet
37. Verbindung zwischen der Allgemein Seite und Backend implementiert
38. Verbindung zwischen Einstellungen Seite und Backend implementiert
39. JSON als Korpus in mongoDb integriert
40. Den Hinzufüge Button auf der Korpus Seite ändern und den Entfern Button integriert
41. Installations- und Administrationshandbuch hinzugefügt
42. Benutzte licences und project licence eingetragen
43. Ausblick ausgedacht
44. Fix Intent Cards in container HTML intent-array Intent Karten in Container HTML intent-array fixiert
45. Nicht benutzte Branches gelöscht

46. Readme über Gitlab clean hinzugefügt
47. Den Icon von der Katze im Chat Interface geändert
48. READE.ME für questME Gitlab Repository hinzugefügt
49. Die Technologien in der Dokumentation aktualisiert
50. Alle Meeting Protokolle verfasst
51. Alle Meeting Protokolle in die Dokumentation branch gepusht
52. UI in englisch umschreiben

### **13.2 Frau Pavithra Sureshkumar**

Hier werden die Issues von Frau Pavithra Sureshkumar aufgelistet.

1. Am Anfang des Projekts: Aufgabenverteilung und Milestones mit Prioritäten erstellen
2. Angular vs. React.js vs, Vue.js recherchiert
3. Die Zwischenpräsentationsfolien erstellt
4. UI Designs erstellt und dokumentiert
5. About Us hinzugefügt
6. User Stories erstellt und dokumentiert
7. Angular getestet und Chat Interface erstellt
8. User stories neu formuliert
9. getrennte Komponentendiagramm nerstellt
10. Kriterien für Usability erstellt
11. Zielgruppe, Problem, Eigenschaften, Alleinstellungsmerkmal erstellt
12. merge fault von User Stories korrigiert
13. Admin Interface mit Angular erstellt
14. An Corpus gearbeitet
15. Usability-Test erstellt und hinzugefügt
16. UI designs korrigiert

17. Appendix erstellt
18. Die Dokumentation Tree im Gitlab gesäubert
19. User Story mit Acceptance Criteria erstellt und dokumentiert
20. Den UML Komponentendiagramm hinzugefügt
21. Die Admin development branch gesäubert
22. Funktionalität zu Admin Corpus Website hinzugefügt
23. UI Designs Version1/2 korrigiert
24. Team Building Retrospective gehalten
25. Corpus for University erstellt
26. Corpus for Interna erstellt
27. Angular CI/CD for automated testing recherchiert (aber keine Zeit gehabt auszutesten)
28. Hinzufüge-/ und Entfern-/ Button für die Intent Cards im Admin Interface implementiert
29. Allgemein und Einstellungen im Admin Interface gearbeitet
30. Chat/Admin UI, NLP und mongoDb kombiniert
31. Verbindung zwischen der Allgemein Seite und Backend implementiert
32. Verbindung zwischen Einstellungen Seite und Backend implementiert
33. Ein Hintergrundbild für den Chat hinzugefügt
34. Angefangen an der Enddokumentation zu arbeiten
35. Installations- und Administrationshandbuch hinzugefügt
36. Die Aufgabenverteilung von Ralf und Pavithra dokumentiert
37. Die Reflektion vom Projektmanagement hinzugefügt
38. Benutzte licences und project licence eingetragen
39. Ausblick ausgedacht und dokumentiert
40. Team Reflektion vom Lernfortschritt verfasst und dokumentiert
41. Angefangen die Endpräsentation vorzubereiten und Präsentationsfolien erstellt

42. Alle Meeting Protokolle verfasst
43. Meeting Protokolle als PDF in Appendix (in Latex) hinzugefügt
44. Aufgabenverteilung korrigiert

### **13.3 Herr Kevin Sautner**

Hier werden die Issues von Herr Kevin Sautner aufgelistet.

1. Recherche zu MongoDB vs. PostgreSQL
2. Tabellen MongoDB vs. PostgreSQL und Angular vs. Vue.js hinzugefügt
3. Gantt-Diagramm überarbeitet und hinzugefügt
4. UML-Verteilungsdiagramm erstellt und hinzugefügt
5. Fazit zu den Technologien-Tabellen hinzugefügt
6. Tests mit Keycloak durchgeführt
7. Gantt-Diagramm um Milestones erweitert
8. Zielgruppe Professor erstellt
9. Beschreibung zum Gantt-Diagramm hinzugefügt
10. Keycloak implementiert
11. Gantt- und Verteilungsdiagramm überarbeitet
12. Fazit zu Tabellen überarbeitet
13. Teile der Use Cases und User Stories erstellen und überarbeitet
14. Unterpunkte von Technologien neu angeordnet
15. Rechtschreibprüfung der Dokumentation zur Zwischenabgabe
16. Versucht Keycloak auf SAML umzustellen
17. Grundstruktur der Rest-API
18. Keycloak Authentifizierung in die Rest-API implementiert
19. Auslesen und senden der Nutzerrollen mit einer Chat-Nachricht
20. Kapitel zur Sicherheit hinzugefügt
21. UML-Verteilungsdiagramm angepasst
22. Keycloak Administrationshandbuch hinzugefügt

## 14 Reflektion Projektmanagement

In diesem Abschnitt wird die Planung, die man vorgenommen hat (in Meilensteine) reflektiert. Reflektiert wird, ob die Planung wie gehabt durchgeführt wurde und wie die Aufwandschätzung vom Geplanten und der Realisierung war.

Meilensteine	alter Stand	neuer Stand
Recherche	13.10.21	11.01.22
Zwischenpräsentation	22.10.21	22.10.21
Implementation	25.10.21	12.01.22
MVP und Code Review	02.12.21	10.12.21
<b>Endpräsentation und Enddokumentation</b>	14.01.22	14.01.22

Tabelle 7: Meilenstein Liste: Neuer Stand

### 14.1 Geplante Meilensteine für Reflektion

Hier werden die Meilensteine, die man als Ziel gesetzt hat verglichen mit dem realen Aufwand.

#### 14.1.1 Recherche 11.01.22

Die Recherche lief nach Plan. Was aber nicht ging, alles bis 13.10.21 zu lernen, da wir sehr viel zu Erfüllen hatten und nur drei Personen waren, mussten wir kontinuierlich Lernen. So musste nicht nur ein Bereich gelernt werden, sondern wir mussten uns Austauschen und unsere Codes vergleichen und zusammenfügen. Dies hat uns dann noch mehr Zeit gekostet als erwartet und wir sind auf kontinuierliche Recherche umgestiegen.

#### **14.1.2 Zwischenpräsentation 22.10.21**

Bei der Zwischenpräsentation haben wir uns sehr viel Zeit genommen uns eine kreative Idee auszudenken, was uns von den anderen Gruppen unterscheidet und wie wir unsere Präsentation so einfach wie möglich darstellen. Die Zwischenpräsentation mussten wir auch sehr schnell erledigen und hatten kaum Zeit zu üben, da diese nach einem kurzen Zeitraum stattfand. Wir haben uns aber trotzdem in der Hochschule getroffen und haben im Präsentationsraum unsere Präsentation Test geprobt, um uns vorzubereiten und unsere Ressourcen auszutesten, wie Ton, Laptop verbinden und das Reden mit der Mundschutzmaske nicht zu vergessen.

#### **14.1.3 Implementation 12.01.22**

Die Implementation dauert noch bis 12.01.22 an und ist gerade dabei noch ein Feinschliff durchzulaufen. Das wichtigste für uns ist hierbei das Produkt so weit wie möglich präsentabel zu programmieren. Die Implementation lief sehr schwergängig, da wir auch sehr viel zu tun hatten mit nur drei Gruppenmitglieder. Hinzu kam es noch, dass sich Probleme beim Implementieren ergeben haben, aber diese von uns wieder durch Pair Programming gelöst wurde.

#### **14.1.4 MVP und Codereview 10.12.21**

Das MVP oder auch technischer Durchstich genannt haben wir sehr gut hinbekommen, jedoch mussten wir diese eine Woche später durchführen. Das wichtigste hierbei war das Codereview. Beim Codereview mussten wir unsere Codes vorstellen, damit wir auch zeigen können wer welche Aufgaben gelöst hat. Wir wollten unser MVP mehr ausreifen, aber hatten wieder das zeitliche Problem, Probleme Wissenslücken zu füllen und das Problem Codes zu kombinieren damit diese auch funktionieren.

#### **14.1.5 Endpräsentation und Enddokumentation 14.01.22**

Bei der Endpräsentation haben wir uns vorgenommen früher Anzufertigen, aber wir konnten es aus zeitlichen Gründen nicht eine Woche früher vorbereiten. Wir müssen nämlich auch den ChatBot ausbessern und einen Feinschliff durchführen. Die Präsentation kann nur vorbereitet werden, wenn der ChatBot fertig und vorführbar ist. Diese haben wir geplant am 12.01.22 fertig zu haben, damit wir noch bis zu der Präsentation üben können, um die Präsentation so gut wie möglich vorzuführen. Die Enddokumentation wird schon von Anfang an verfasst und wird dann mit der Endpräsentation abgegeben.

# 15 Reflektion Lernfortschritt

Hier wird der Lernforschritt, den wir als Gruppe gesammelt haben beschrieben.

## 15.1 Reflektion Lernfortschritt von Frau Pavithra Sureshkumar

Hier wird der Lernfortschritt von Frau Pavithra Sureshkumar zusammengefasst. Dabei werden nicht nur die technischen Lernfortschritte sondern auch die Teamarbeit als Reflektion erläutert.

In unserem Projekt habe ich sehr viel Neues gelernt. Meine Aufgabe war es im Team das Frontend mit Angular zu programmieren und es aufzustellen. Da ich kaum wissen über Typescript oder Angular hatte, musste ich mich erst mal damit beschäftigen, wie ich diese einsetzen kann. Dafür habe ich mich mit der Angular docs Seite beschäftigt <https://angular.io/docs>, auch noch weitere Tutorials angeschaut und natürlich auch bei Problemen in Stackoverflow nachgeschaut. Als Richtlinie für die Komponenten habe ich Angular Material benutzt, abgeändert und versucht diese in das Interface einzusetzen <https://material.angular.io/>. Ich habe gleich in der ersten Woche versucht das Chatinterface während dem Lernen zu programmieren. Ich habe im Projekt sehr viel mit Ralf gearbeitet, weil ich mit ihm das Chat Interface was ich programmiert habe mit dem backend verbinden musste. Ich habe auch das Angular Routing angewandt um durch die ganzen Seiten zu navigieren. Nachdem der Chat mit dem backend verbunden wurde habe ich mich mit dem Admin Interface befasst. Dort musste ich das Routing einsetzen und die ganzen Seiten verbinden. Ich habe mich mit Kevin ausgetauscht, wie es für ihn einfacher wäre auf die Routen zuzugreifen und habe diese auch so umgesetzt, wie er es haben wollte. Nachdem ich die Komponenten mit Angular im Admin Interface programmiert habe, konnte ich mein Wissen mit Ralf teilen und das backend mit dem frontend der Admin Interface Seiten verbinden, um den Seiten eine Funktion zu erschaffen. Dabei haben wir beide gelernt, dass es viel einfacher gewesen wäre ein Datenmodell von vorne herein zu erschaffen. Ich und Ralf haben es aber trotzdem geschafft das Chat und das Admin Interface so weit wie möglich zu programmieren. Ich habe mich kurz mit dem CI/CD beschäftigt konnte diese aber nicht durchführen, weil ich andere Aufgaben erledigen musste und kaum Zeit gefunden habe. Trotzdem habe ich mit Ralf gelernt, wie man eine branch im Gitlab cleaned und wie man mit den erstellten Issues arbeitet. Alles was ich gemacht und gelernt habe, habe ich in Gitlab in den Issues ausführlich dokumentiert. Ich habe mich nicht nur mit dem programmieren beschäftigt, sondern habe auch Richtlinien für das dokumentieren erstellt und das meinen Teammitgliedern weitergegeben. In der Dokumentation habe ich während ich programmiert habe gearbeitet. Bei der Zwischenpräsentation und bei der Endpräsentation habe ich den ersten Schritt gewagt für die Gruppe die Präsentationsfolien zu erstellen und die allgemeine Gliederung zu erstellen. Die Enddokumentation habe ich auch selber angefangen und habe dann mich mit Ralf ausgetauscht. Da Kevin sich nach den Weihnachtsferien nicht gemel-

det hat. Habe ich seine Aufgaben ausgelassen und auf eine Ergänzung gewartet.

Ich habe nicht nur fachliches Wissen erweitert und eingesetzt, sondern auch Teambildung und Teamarbeit ausgeführt. Wir haben trotz dass wir nur drei Personen sind, versucht Scrum auszuführen. Das ein Team nicht immer einwandfrei läuft ist selbstverständlich. Es können immer Probleme bei Kommunikation entstehen und man könnte etwas falsch verstehen oder übermitteln. Dafür haben wir eine Retrospektive gemacht. Wir haben unsere Probleme angesprochen und eine Lösung gefunden. Trotzdem hat sich das Arbeitsverhalten von Kevin nicht geändert. Auch haben wir gelernt, dass manche nicht die Motivation gehabt haben etwas im Team zu leisten. Ich und Ralf haben beschlossen dann etwas als Team zu unternehmen, um den Teamgeist zu steigern. Mit unserem Betreuer haben wir immer versucht jede Woche eine Besprechung zu halten und haben diese auch immer dokumentiert und immer nach der Besprechung zusammengefasst und es dem Meeting Mitglieder geschickt, um den Überblick zu behalten. Auch wenn man sich austauscht, sollte aber in jedem Team das selbstständige Arbeiten nicht ausfallen. Mit dem selbstständigen Arbeiten ist gemeint, dass man Aufgaben übernimmt und nicht nur die Aufgaben zugeteilt bekommt, das heißt Eigeninitiative sollte bei jedem vorhanden sein. Diese erwies sich schwer, weil es immer verschiedene Arbeitscharaktere gibt. Manche können nicht sich organisieren oder Selbstständigkeit vermitteln oder müssen extra Motivation finden. Das alles haben wir im Seminar gelernt und ist für uns eine große Lehre.

Außerdem haben wir die Planung unterschätzt. Wir hätten strenge Regeln setzen müssen, damit die Aufgabenverteilung gerecht verteilt wird. Da Kevin keine Eigeninitiative gezeigt hat mussten ich und Ralf ihm immer Aufgaben zuweisen. Am Ende hat er diese nicht mal ausgeführt und sich erst später damit beschäftigt und sich nicht mal damit auseinandergestzt, wie man es hätte machen sollen. Ich und Ralf in der Gruppe mehr machen müssen und haben auf Hilfe von Kevin gewartet, die nie ankam. Am Anfang wurde die Planung so ausgeführt, dass sich jeder mit seinem Thema beschäftigt und schon was macht. Leider hat Kevin diese Aufgabe nicht ernst genommen oder halbherzig ausgeführt und es kam dazu, dass sich ein paar Gruppenmitglieder mit sehr viel Arbeit überbelastet haben. Am Ende haben wir noch versucht die Aufgabenverteilung gerechter zu machen. Trotz das ich und Ralf sich mehr Mühe gegeben haben und schon mittlerweile auch mental nicht instande waren auf Kevin aufzupassen, hat er sich nicht einmal nach den Ferien bei uns gemeldet. Wir sind nicht seine Aufsichtsperson und haben uns auch beschlossen dann auf unsere Aufgaben zu kümmern. Es ist nicht selbstverständlich für mich einer Person hinterher zu laufen, wenn wir das gleiche Arbeitspensum abliefern müssen. Ich habe dabei nur gelernt, dass ich mich nächstes Mal nicht mit so einer Person annährend beschäftigen möchte und wenn, dann es schon früher dem Betreuer vermitteln würde, auch wenn es sehr offensichtlich war, dass diese Person sich nicht viel Mühe im Projekt gegeben hat. Ich sehe dieses Projekt als eine große Lehre und werde in den nächsten Projekten, die vor mir stehen meine Erfahrungen die aus diesem Projekt hervorgegangen sind Einsetzen.

## 15.2 Reflektion Lernfortschritt von Herr Kevin Sautner

Da es von meiner Seite nicht allzu viel Gutes zu sagen gibt, starte ich auch gleich einmal damit. Durch das Projekt wurde mein Interesse an Full-Stack-Entwicklung geweckt und ich konnte mir ein paar Gedanken machen, zu eigenen Projekten, die ich in Zukunft vielleicht angehen werde. Es war schön mal die Erfahrung gemacht zu haben in einem Team an einem Projekt zu arbeiten und Tools wie GitLab auch mal halbwegs richtig zu benutzen, auch wenn das mit dem Team am Ende nicht so gut funktioniert hat. Und ja, da kommen wir auch schon zum negativen Teil.

Je weiter das Projekt fortgeschritten ist, desto mehr Probleme hatten wir mit der Kommunikation. Das ist so weit gegangen, dass wir uns auch Anfang Dezember zusammengesetzt haben und darüber gesprochen haben. Bis zu diesem Punkt würde ich sagen, wurden Fehler auf beiden Seiten begangen. Nach dem Gespräch hatte es sich dann deutlich gebessert, aber über die Ferien ist die Kommunikation dann wieder vollständig zusammengebrochen. Besonders über diesen Zeitraum, von Ferien bis Abgabe, würde ich die volle Verantwortung übernehmen. Damit komme ich auch zu dem wichtigsten was ich im Laufe des Projekts gelernt habe.

Ich muss mehr auf meine mentale Gesundheit achten. Damit kämpfe ich schon seit einigen Jahren, aber es hat sich immer recht in Grenzen gehalten und hat sich nie groß auf meine Leistungen ausgewirkt. Aber gerade durch die aktuellen Umstände und häufige Rückschläge hat es leider in den letzten Wochen und Monaten einen absoluten Hochpunkt erreicht. Trotz meiner geringen Beteiligung hat mich das Projekt extrem mitgenommen. Ich habe mich generell immer und immer mehr abgekapselt und über die Ferienzeit dann quasi komplett abgeschalten. Hätte mir jemand geschrieben hätte ich das noch mitbekommen, aber ansonsten habe ich mich komplett von Kommunikationswegen abgeschnitten.

Am Ende tut es mir einfach nur leid, dass ich meine Teammitglieder zusätzlich belastet habe und kann mich dafür auch nicht genug entschuldigen.

# **16 Ausblick: Pläne für die Zukunft**

Hier werden die möglichen Ergänzungen in unserem Projekt erwähnt.

## **16.1 Mögliche Ergänzungen in der Zukunft**

Hier werden die Ergänzungen aufgelistet.

- CSV importieren von Intents
- Mehr individuelles Gestalten der Chat Seite
- Mehr Korpusdaten
- Mehr Domänen und Gruppen
- Nicht nur lokaler Zugang
- questMe in mehreren Plattformen integrieren
- Paginator für Korpusdaten

# 17 Appendix

Hier wird der zusätzliche Content abgebildet.

## 17.1 Ältere Versionen des Komponentendiagramms

Hier werden die älteren Versionen und Entwürfe des Komponentendiagramms abgelegt.

### 17.1.1 Komponentendiagramme

Im folgenden sind die älteren Versionen unserer Komponentendiagramme

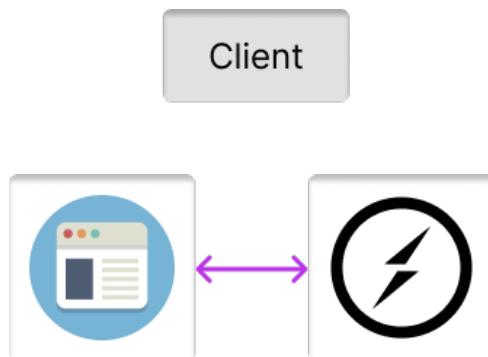


Abbildung 38: Komponentendiagramm Client

Hier wird die Client Seite bildlich dargestellt. Man sieht, dass der Webbrowseer durch die Socket.io Client deployed wird und dadurch wird dann eine Beziehung zur Serverseite aufgebaut. (Siehe Abbildung 38)

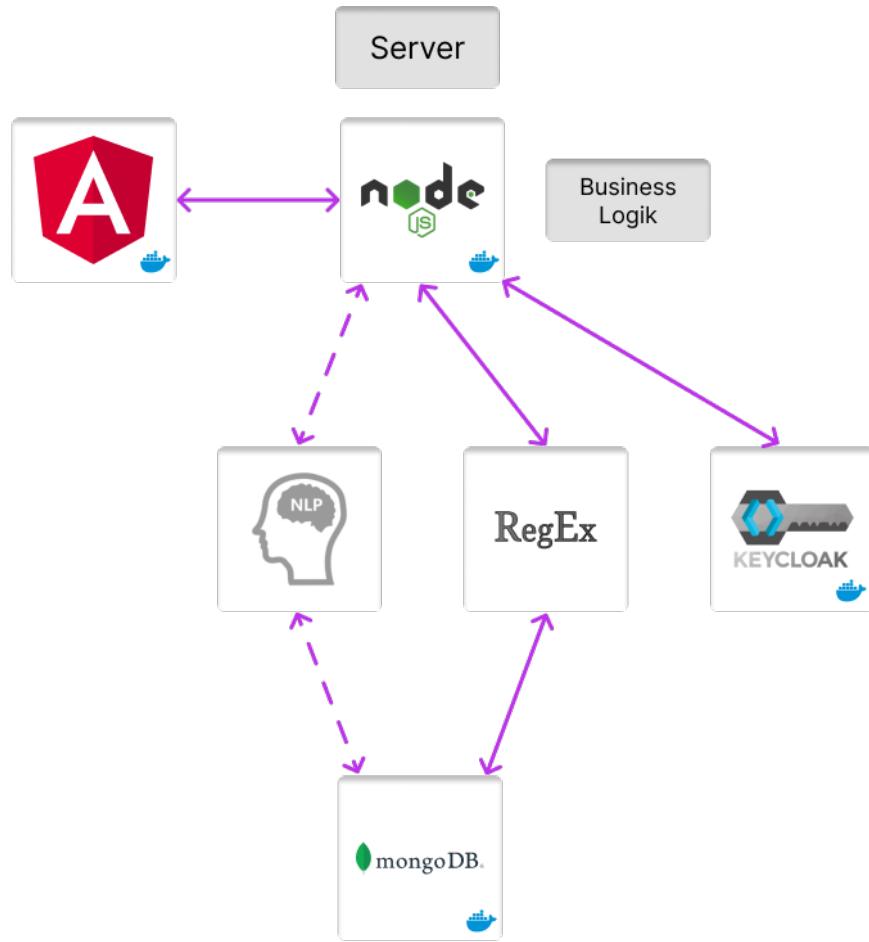


Abbildung 39: Komponentendiagramm Server

In der Serverseite wird dann durch den Socket.io Server die Verbindung zum Client aufrecht gehalten. Im Server befindet sich Angular, node.js, KeyCloak und die Datenbank mongoDB und haben jeweils einen eigenen Dockercontainer. Die Erkennung von der eingegebenen Sprache möchten wir zunächst mit RegEx ermöglichen, um das Minimal Viable Product hinzubekommen. Optional dann mit NLP (Natural Language Processing) erweitern.

In dieser Version haben wir erst einmal die Struktur von unseren Komponenten gesucht und eine grobe Darstellung erstellt. Was wir hier aber nicht wussten ist, wie wir das NLP darstellen sollten. NLP war für uns vorher eine optionale Möglichkeit.

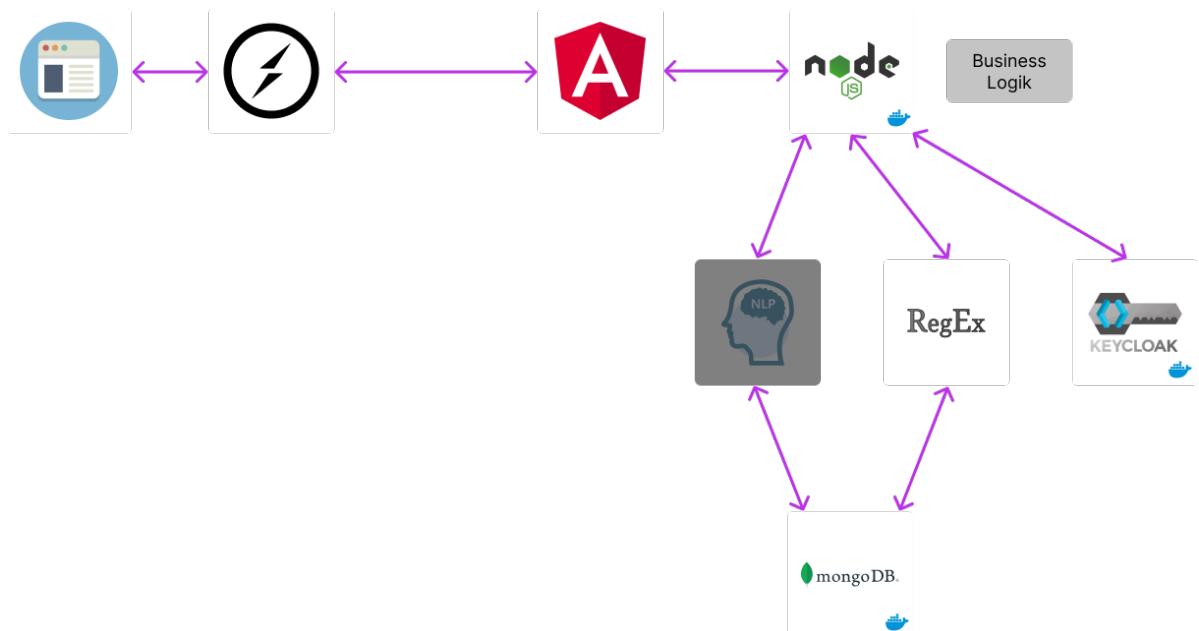


Abbildung 40: Komponentendiagramm v1.0

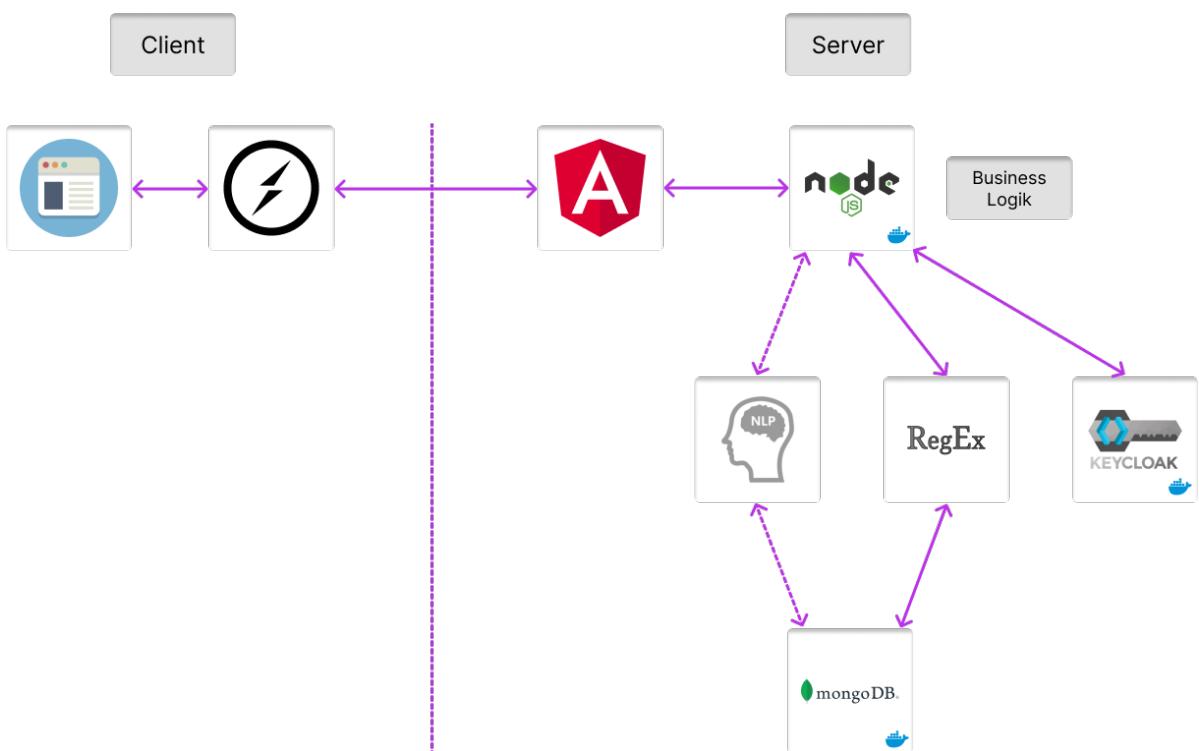


Abbildung 41: Komponentendiagramm v1.1

In dieser Darstellung haben wir die einzelnen Komponenten in Server und Client eingeteilt, um die Struktur besser zu verstehen. Wir haben aber die Abtrennung nicht

richtig anzeigen können und das NLP haben wir auch nicht klar als optional zeigen können.

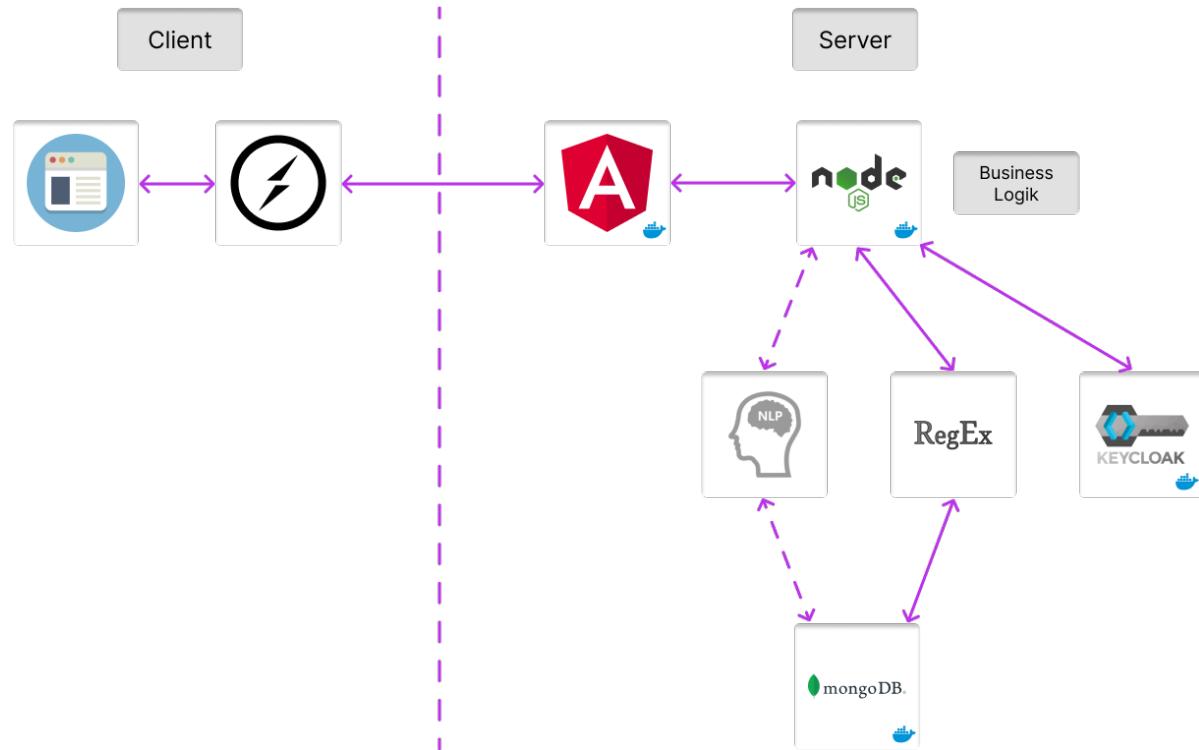


Abbildung 42: Komponentendiagramm v1.2

## 18 Meeting Protokolle

Hier werden die Meeting Protokolle von unseren Besprechungen mit Herr Watzko, Herr Renz und Herr Rößler protokolliert.

### 18.1 Protokolle in PDFs eingebunden

Hier werden die Protokolle als PDFs eingebunden. Diese werden von alt bis neu aufgelistet.

# Protokoll 08.10.21: Erste Besprechung

## Teilnehmer:

Michael Watzko, Pavithra Sureshkumar, Kevin Sautner, Ralf Zeller

## Inhalte:

- Vorstellung der einzelnen Teilnehmer
- Besprechung der Organisation des Projekts (Kunden, Betreuer, Entwickler)
- Hinweise auf was man beim Projekt achten soll
- Information über welche Details mit Herr Renz noch geklärt werden müssen.
- Welche Diagramme man am besten zum Kundengespräch mitnehmen soll
- Tipp: Zum Erstellen von Diagrammen „draw.io“
- Tipp: Diagramme als UMLs erstellen
- Beantragung des Gitlab repo bei Herr Rößler
- Neue Terminplanung für einen Kundentermin mit Herr Renz planen
- Grobe Zeitplanung des Projekts vorab vorschlagen
- Gantt Diagramm für Zeitplanung
- Erste Erkenntnisse zu Technologien auflisten
- Erstellen eines Kanban Boards
- Meilensteine definieren
- Functional und Non-Functional Requirements formulieren (notwendig für Doku)
- Infos zum Erstellen von Tickets für Anforderungen
- etwa 40 Tickets sind geplant müssen aber nicht beim ersten Termin schon speziell definiert werden
- Risiken auflisten und Erstellen
  
- Gespräch darüber wie KeyCloak verwendet werden soll, um die Umsetzung besser planen zu können
  
- Gespräch über welche Frontend Umgebung für das Projekt geeignet ist (Angular, React, Vue)
- Ergänzung zu der erstellten Tabelle warum man Angular oder Vue verwendet werden sollte
  
- Welche NLP geeignet sein sollte
  
- Info über socket.io (bidirektionale Kommunikation)
  
- Information über welche Datenbank geeignet wäre (MongoDB, PostgreSQL)
- Entscheidung des Teams für MongoDB
  
- Gespräch darüber wie man den Korpus des ChatBots gestalten kann
- Beispiel Aufteilung des Korpus in Domänen
- Information zum Erstellen einer Doku zur Bedienung des Chatbots

- Welche Funktionen soll der Bot können?
- Welche Funktionen soll der Bot nicht können?
- Klärung für welche Plattform entwickelt werden soll (Tablet, PC, Smartphone,...)
- Klärung soll der Bot Tonausgabe haben
- Gitlab Infos wie man damit am besten arbeitet
- Gespräch darüber wie oft man etwas „commiten“ soll
- Tipps zur Vermeidung von Problemen in Gitlab
- Gitlab Infos welche automatischen Testverfahren möglich sein könnten (nach Erkenntnisstand)
- Infos zu IDEs besprochen
- Welche IDEs für das Projekt hilfreich sein könnten (IntelliJ)
- Info darüber wie viel man ins Projekt als „muss“ rein nehmen soll
- Erweiterte Funktionen besprochen.
- Erweiterte Funktion: Themes angesprochen
- Erweiterte Funktion: Animierter Chatbot
- Erweiterte Funktion: Kommentar vom Bot bei Bearbeitung
- Mockups, Diagramme, etc. zu Kundentermin mitbringen, um Design Entscheidungen einfacher zu ändern.
- grobe Architektur Diagramme erstellen
- falls möglich User Stories erstellen
- Erste Erkenntnisse zum Projekt zusammenfassen und präsentabel gestalten
- Erste Wireframe Diagramme zum ersten Kundengespräch mitnehmen

# Protokoll 13.10.21: Erster Kundentermin

## Teilnehmer:

Alexander Renz, Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

## Inhalte:

- Meilensteine zur Abgabe vom Softwareprojekt angesprochen
- 1. Meilenstein Zwischenpräsentation 22.10.21
- 2. Meilenstein Product Vision, User Stories, UI Mockup
- 3. Meilenstein „Technischer Durchstich“, Code-Review
- 4. Meilenstein Finale Abgabe, Endpräsentation, Enddokumentation
- Besprechung wie benotet wird
- Anschauen der Benotungsgrundlage „Bewertung\_PSWTM\_S21.pdf“
- Vorzeitige Architektur vorgestellt
- NLP ist nicht gefordert (optional), Für das Projekt wäre Regex ausreichend
- Text-to-Speech nur wenn es nicht zu gruselig ist (optional)
- Feedback zur vorzeitigen vorgestellten Architektur:
  - socket.io Client ist im Browser
  - Frontend ist Teil des Servers
  - Nochmal das UML (Komponente-/ Verteilungsdiagramm) anschauen
  - UML gut für Darstellung von Architekturen, weil der Kunde es leichter versteht
  - Herr Rößler hat im GitHub, eine Demo zu Keycloak und node.js (<https://github.com/go-hse?tab=repositories>)
  - Keycloak benutzt man hauptsächlich für Authentifizierung
  - Besprechung mögliche Integration des geforderten Admin Webseite für den Bot in Keycloak
  - Mögliche Schwierigkeiten bei der Anpassung der Keycloak Admin Webseite
  - Empfehlung separate Admin Webseite für den Bot

- Besprechung wie man die Docker Container einteilen sollte
- Empfohlene Motivation für die Docker Container, sie „modular“ halten
- Empfehlung zur Nutzung von „Docker compose“
  
- Die Webseite vom Chatbot und der Admin Webseite als „Responsive“ entwickeln
- Besprochen das „Mobile First“ in Ordnung ist
- Vorstellung der Tabelle „Angular vs Vue.js“
- Besprechung für die Entscheidung für Angular
- Empfehlung: Fertige UI Komponenten für Angular verwenden
- Vorstellung der Tabelle „mongoDB vs PostgreSQL“
- Besprechung für die Entscheidung für PostgreSQL
- Besprechung welche Versionen wahrscheinlich von den einzelnen Technologien verwendet werden
- Empfehlung: Entscheidungen für eine Technologie in die Dokumentation aufnehmen
  
- Selbstdefinierte Meilensteine vom Team besprochen
- Vorstellung eines vorzeitigen Gantt-Diagramms
- Verbesserungen des Gantt-Diagramms um Termine wie Projekt Meilensteine, Dokumentation erstellen, etc.
- Empfehlung nach Möglichkeit so viel wie möglich parallelisieren
- Entscheidung für eine Haupt IDE „Visual Studio Code“ für das Projekt
- Anstrengung so früh wie möglich ein MVP zu erstellen
- Besprechung der vorzeitigen Risiken für das Projekt, die von der Gruppe ermittelt wurden
- Besprechung der Antwort Verzögerung des Chat Bots
- Besprechung welche Informationen in das Repertoire aufgenommen werden sollten
- Besprechung auf welche Lizenzen für das Projekt zugegriffen werden kann (OpenSource, MIT, BSD)
- Vorstellung einer vorzeitigen ChatBot Webseite
- Besprechung was verbessert werden soll an der ChatBot Webseite
- Mögliche Probleme bei dem vorzeitigen Entwurf der Chatbot Webseite
- Vorstellung eines vorzeitigen Webseiten Admin Logins

- Besprechung wegen Weiterleitung durch KeyCloak zum Admin Login
- Vorstellung eines vorzeitigen Webseiten Admin Interfaces
- Besprechung welche Elemente ausgebessert werden sollten
- Besprechung von einem möglichen Muster für Eingabe und Antworten
- Besprechung welche Landing Page für den ChatBot geeignet wäre
- Empfehlung was man beachten sollte für die Präsentation
- Besprechung welche Sprache als Hauptsprache für den Korpus genutzt wird  
(1. Deutsch, 2. Englisch)
- Besprechung über die Einplanung für einen Termin zur Vorstellung einer „Demo“  
(so früh wie möglich)

# Protokoll 19.10.21: Zweite Besprechung

## Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

## Inhalte:

- Konzept der Zwischenpräsentation besprochen
- Folie zu verwendeten Technologien besprochen
  - Andeuten, was wozu benutzt wird
  - Wie interagieren die Technologien miteinander?
  - NLP nur als optionales Feature einbauen
  - Einzelne Technologien mit Piktogrammen nach Möglichkeit darstellen
  - etwa drei bis vier Technologien pro Folie
- Folie mit der Produkt Roadmap besprochen
  - Meilensteine mit Datum angeben
  - MVP einbauen
  - Überlegen welche Informationen noch hinzugefügt werden müssen
- Folie mit dem UI Design besprochen
  - Mockups für die Webseitendarstellung auf mehrere Folien aufteilen für die bessere Übersicht
  - Kurze Einblendung, wie es auf dem Handy aussehen soll
  - Was soll im Menüpunkt "Allgemein" und "Einstellungen" passieren?
- Wie soll der Schluss der Präsentation aussehen
  - Zusammenfassung an konkreten Punkten
  - Quellenverzeichnis als letzte Folie
- Einbauen einer Feature Liste an das Ende der Präsentation
- Besprechung über einen guten Abschluss für die Präsentation
- Mockup für das Admin Webinterface besprochen

- Inhalt des Unterpunktes "Korpus"
  - Neues Plus zum Hinzufügen eines Frage-Antworten Blocks
  - Mögliches Hinzufügen eines Plus bei den Fragen, um mehrere Fragen mit mehreren Antworten zu verknüpfen
  - Frage-Antwort Blöcke möglicherweise zusammenfassen zu Fragesätzen die dann auf bestimmte Gruppen wie "Dozenten" und "Studenten" abgestimmt werden
  - Hinzufügen von z.B. einem Zahnrad über das man dann eine Frage oder Antwort editieren bzw. löschen kann
  - Einbauen eines Dropdown Menüs zum Auswählen der Domaine des Korpus
- Möglicher Inhalt des Unterpunktes "Allgemein"
  - Icon des Chatbots ändern
  - Name des Chatbots ändern
- Möglicher Inhalt des Unterpunktes "Einstellungen"
  - Verlinkung auf das Webinterface von Keycloak
- Hinzufügen einer „Logout“ Möglichkeit
- Anwendungsarchitektur nochmal besprochen
- Einbauen eines Komponenten Diagramms in die Präsentation
- Informationen sammeln um eine Korpus Domaine zum Thema „Hochschule“ erzeugen zu können
- Festlegung auf Domaine „Basis“ und „Hochschule“ für den Korpus
- Anregung bereits Fragen und Antworten zu sammeln und diese evtl. auch beispielhaft in der Zwischenpräsentation einbringen
- Besprochen, wie detailliert die Protokolle ausfallen sollen
  - Sollten nicht detaillierter werden
- Anregung möglichst viel zu testen
- Besprochen, wie wir die Risikoliste überarbeiten sollten
  - Eine Tabelle anlegen und zu jedem Risiko zuordnen, wie gefährlich ist es, was für einen "impact" hat es und was tun wir dagegen
- Festlegung auf das Nutzen von LaTeX für das Erstellen der Dokumentation

# Protokoll 20.10.21:

## Zwischenpräsentation Besprechung

### Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

### Inhalte:

- Gemeinsames durchgehen der Präsentationsfolien
- Änderungen die für Folie „ChatBot erste Seite“ geplant sind:
  - Ergänzen von Datum, Titel „Zwischenpräsentation“
- Änderungen die für Folie „Technologien“ geplant sind:
  - Technologien Komponenten Diagramm
    - um gestrichelte Pfeile bei NLP ergänzen
    - Trennung Client und Server zeigen
  - Technologien einzelne Folien
    - Bilder links um Name ergänzen
- Änderungen die für Folie „UI Design“ geplant sind:
  - Chat UI austauschen gegen mit Korpus „Hochschule“ gefüllte Informationen
  - Ergänzen bei Gruppe „Professor“ um „Interna“
- Änderungen die für Folie „Meilensteine/Roadmap“ geplant sind:
  - Datum ergänzen
  - Reihenfolge ändern
- Änderungen die für Folie „Feature List“ geplant sind:#
  - Ergänzen um Funktionen aus den UI Designs
  - Interessant formulieren
- Änderungen die für Folie „About us“ geplant sind:
  - nach der ersten Folie einfügen

- Änderungen die für Folie „Abschlussfolie“ geplant sind:
  - um Sprechblase „noch Fragen?“ ergänzen
- Geplante Probe für morgen 21.10.21
  - Technik check (Audio, etc.)

# Protokoll 29.10.21:

## Dritte Besprechung

Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

Inhalte:

- Information über das Ergebnis der Zwischenpräsentation
- Information über die nächsten Meilensteine der Projektarbeit
  - Technischer Durchstich
  - Code Reviews
  - Abgabe der Dokumentation
- Dokumentation auf Gitlab
  - Doku wurde größtenteils bis zum aktuellen Datum hochgeladen
  - verfasst im Latex-Format
- Verbesserungsvorschläge für die Doku
  - Blocksatz einfügen
  - Nutzen von Bibtex um Quellenverzeichnis zu erstellen
- Zeigen des ChatBot Beispiels der Bibliothek NLP.js
- Besprechung wie Gitlab genutzt werden soll
  - Team möchte „Gitlab Flow“ verwenden (erleichtert CI/CD)
  - Vorstellen der einzelnen Branches
  - Wie man Pull Requests verwenden kann
  - Empfehlung von Git Rebase um Branch Struktur übersichtlich zu machen

- Kurze Besprechung von CI/CD
  - Wie man es benutzen kann
  - Ausführliche Einführung davon von Herr Rößler geplant
  - Kann automatisiert Branches deployen
- Verbesserungsvorschlag zum Nutzen der ReadMe.md
  - Markdown Language nutzen
  - Links zum Nachlesen bekommen
  - Bisher verwendetes ReadMe korrigieren
- Kurze Besprechung wie Branches aktualisiert werden können
- Empfehlung: Nutzen von Tags für Versionen des ChatBots
  - Können direkt verwendet werden (Sprungadresse)
- Empfehlung: Commits verknüpfen mit Tickets
  - Links zum selbständigen Nachlesen erhalten

Änderungen die für Folie „Abschlussfolie“ geplant sind:

- um Sprechblase „noch Fragen?“ ergänzen
- Geplante Probe für morgen 21.10.21

Technik check (Audio, etc.)

# Protokoll 12.11.21:

## Vierte Besprechung

### Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

### Inhalte:

- Information über das Praxissemester
  - Immatrikulationsbescheinigung mitschicken
  - Verweis auf unseren Betreuer in der Bewerbung
  - Adressierung Sehr geehrte Damen und Herren
  - Initiativbewerbung
  - Vorlieben gerne erwähnen (Fullstack, Frontend,...)
- Allgemein
  - Immer nach einer Überschrift einen beschreibenden Text schreiben
  - Tabellen erklären warum sie verwendet werden
- Veränderung beim eingebundenen Literaturverzeichnis
  - Biblatex ist in Ordnung
  - GoogleUrls kürzen
  - falls kein Erscheinungsdatum findbar auch in Ordnung
  - Zitate gerne Experimentieren (Nummerierung, Autor Jahr,...)
  - Zitate auf jeden Fall einheitlich halten

- UI Konzepte
  - in einem extra Unterpunkt ordnen
  - von Version 1 zu Version 2 auf Veränderungen eingehen
- Kriterien für Usability
  - auf eine neue Seite packen
  - Unterpunkte als „Subsubsections“ setzen
- Usability Test
  - um Task ergänzen
  - Task soll etwa eine halbe Seite lang sein
- User Stories
  - bisherige „User Stories“ in „Use Cases“ umbenennen
  - bei den Use Cases für die Bereiche absolute Nummern vergeben
  - als Beispiel u10000, u20000,... für die Bereiche
  - 3 Exemplarische User Stories
  - als Beispiel für jede Zielgruppe eins
- Use Cases Admin Interface
  - Beschreibung Teil der Professoren hat auch administrative Aufgaben
- Technologien
  - frühere Versionen in den Anhang z.B. Appendix
  - Technologie Liste versuchen kleiner zu machen
- Entscheidung für die Technologien
  - Umbenennen in „Kriterien für die Technologien“

- RegEx
  - Als Fallback beschreiben, falls es nicht mit NLP.js funktioniert
  - Beispiel erwähnen wie RegEx funktioniert
- NLP.js
  - jetzt definitiv eingebunden
  - welche Aufgabe es in unserem Projekt hat Technologie Tabellen
  - Fazit für die Wahl einer Technologie schreiben
  - optional kann entfernt werden
- Technologie Diagramme
  - UML Verteilungsdiagramm korrigieren und einfügen
  - bisherige Komponentendiagramme in UML umwandeln
- UML Komponentendiagramme
  - Abgleichen mit Verteilungsdiagramm
  - Client Seite in einen Container packen
  - Server Seite in einen Container packen
  - Beispiel für die Benennung Frontend-, Authentifizierung
  - Alte Komponentendiagramme als Appendix
- Meilensteine
  - Text nach der Überschrift
  - erwähnen was vorgeplant wurde
- Meilenstein Liste
  - Meilensteine „Endpräsentation“ und „Enddokumentation“ zusammen darstellen
  - ebenfalls die Beschreibung zusammen tun

- Zeitmanagement
  - Eingehen auf die Meilensteine
  - Sätze zur Beschreibung warum es so geplant wurde
- Risikoanalyse
  - Sätze was wir besonders wichtig fanden
  - Fazit dazu schreiben
- Datenbank
  - Doku wird um ERM Diagramme ergänzt
  - ERM Diagramm nach empfohlener UML Notation angeben oder Chen-Notation
  - Beschreibung wie der Chatbot mit der Datenbank arbeitet
- Vorstellen des Chatbot Prototypen
  - erste Chatmöglichkeiten als Beispiel
  - Beschreibung der Funktion
- Korpus des ChatBots
  - bisher ist der Korpus lokal beim ChatBot
  - geplant ist Korpus aus der mongoDb zu holen
  - der Bot wird bei Änderungen informiert
- Vorstellung Prototyp des Admin Interfaces
  - Menüs gezeigt
  - Probleme beschrieben

# Protokoll 19.11.21:

## Fünfte Besprechung

Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

Inhalte:

- Besprechung der erstellten Doku
  - Ergebnisse gezeigt
  - Gelerntes erzählt
  - bisher muss nichts korrigiert werden
  - dennoch bis zum MVP geupdatet werden
- Sicherheitsaspekt bei Dockercontainern
  - für kleine Projekte ist es in Ordnung mongoDb ohne Password und Username
  - evtl. erwähnen, dass mongoDb nur local zugänglich ist (Doku)
- Datenbank ER Struktur
  - vorerst in Ordnung
  - muss evtl. geändert werden im Laufe des Projektes
  - Nicht registrierte Nutzer evtl. um ein sinnvolles Attribut ergänzen (Cookield, SessionId,...)
- Risikoanalyse
  - Fazit in eine section umändern

- Admin Interface vorgestellt
  - Struktur der Adminseite
  - Probleme mit einbinden eines “Mat-Carousel” (Version)
  - Carousel wird evtl. selber programmiert oder eine Alternative gefunden
- Keycloak
  - Container läuft bisher
  - muss noch integriert werden
- Zeigen der gereinigten Git Branches
  - Besser erkennlich was geändert wurde
  - Merges sind integriert in die Branch
- CI/CD Zeitpunkt
  - in Ordnung erst nach dem MVP
  - Besser so früh wie möglich
  - kann unterstützend sein
- Besprechung der Meilensteine
  - Mehr Issues sollen für den Meilenstein MVP erstellt/eingeordnet werden
  - Differenzieren zwischen Features von Implementation und MVP
  - Damit Team besser eigenen Fortschritt sieht
- Issues
  - soweit in Ordnung
  - Kommentare in den Issues zu Problemen sehr gut
  - evtl. Issues mit Branches verlinken

- Nachfrage wegen einem Kundentermin vor/nach MVP
  - muss nicht unbedingt vor dem MVP Termin sein
  - ist nicht notwendig reicht auch später
- MVP Termin
  - findet in Präsenz statt
  - Kundentermin wird nicht benötigt, da dem Kunden vorgestellt wird
  - soll eine funktionierende App zeigen mit den vorgestellten Technologien
  - muss nicht bugfrei sein

# Protokoll 26.11.21:

## Sechste Besprechung

Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

Inhalte:

- Besprechung der erstellten Doku
  - Ergebnisse besprochen
  - für wissenschaftliches Arbeiten angewöhnen „Wir“ und „Ich“ nicht zu verwenden
  - auf sachliche Formulierungen zurückgreifen
  - Allgemein Rechtschreibung überarbeiten
- 3.1 Recherche UI Designs
  - Bildunterschriften ergänzen
  - was für uns wichtig bei dem gezeigten Bild war eingehen
- 3.2 Version 1 und 3.3 Version 2 von UI Konzept
  - Bildunterschriften ergänzen
  - Bilder größer machen
- 4.1.1 Responsive Webdesign und 4.2.2 Zielgruppen eingehen
  - Formulierungen bearbeiten: „Zielgruppen..“, „alle Geräte...“

- 5 Use Cases
  - in User Stories umbenennen
  - Formulierungen weniger technisch formulieren
- 6 User Stories
  - in Use Cases umbenennen
- 8 Technologien
  - Technologie Versionen mit Vergleich tauschen
- 8.3.2 Vergleichstabellen
  - Fazit um 2-3 Punkte der Tabelle ergänzen
  - mehr Bezug zum recherchierten erstellen
- Seite 45 Fazit
  - als Überschrift umändern
- Besprechung des Code Reviews nächste Woche
  - jeder stellt seinen Code vor (10-15min)
  - Ticket verlinken
- Angular Admin Webinterface vorgestellt
  - muss noch mit Backend verknüpft werden
- node.js Anbindung zu mongoDb gezeigt
  - nochmals recherchieren wegen Promises

# Protokoll 03.12.21:

## Siebte Besprechung

Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

Inhalte:

- Besprechung gestriges Vorstellungsgespräch
  - Wie es gelaufen ist
  - Was dabei falsch gelaufen ist
- Michael hat uns folgende Vorschläge gemacht, um als Team wieder besser miteinander zu arbeiten
  - Daily Standups, morgens 5 Minuten berichten was getan wurde und was gemacht werden soll
  - Pair Programming
  - Mehrere Meetings wöchentlich (Chat, Audio,...)
  - Wöchentliche Retrospektiven
- Besprechung wie das Team aus der Krise kommt
  - gestern wurde ausführlich eine Retrospektive gemacht
  - Alle Teammitglieder wollen: Mehr Kommunikation, Mehr Information darüber was jeder gemacht hat und gemeinsam wieder etwas zusammen spielen
  - Für Mehr Kommunikation ist geplant, dass man z.b montags, donnerstags und Tage an denen gemeinsam gearbeitet wird ein „Daily Standup“ macht
  - Für Mehr Information, dass jeder am Ende des Tages erzählt was er alles geschafft hat und evtl. ein Discord Team Gespräch stattfindet wo alle gemeinsam zusammen programmieren.

- Gemeinsam etwas zusammen spielen z.B. Minecraft, um etwas gemeinsam zu unternehmen und den Teamgeist zu stärken
  - Probleme dokumentieren
  - Probleme frühzeitig ansprechen und gemeinsam eine Lösung finden
- 
- Keycloak SAML
    - sehr aufwendig umzusetzen
    - wird für den MVP ausgelassen
    - evtl. wird es nicht umgesetzt
    - geplant ist die Dokumentation der Schwierigkeiten
    - Dokumentiert wird der Aufwand von Kevin
- 
- Nächste Woche Code-Review mit Herr Rößler
    - Admin Interface ausbessern
    - Webinterface Corpus mit REST verbinden
    - Chatbot soll Änderungen dynamisch übernehmen
    - Code vorbereiten zum Vorstellen
- 
- Besprechung für die Basis der Docker Container
    - Alpine wird als Basis verwendet
    - Images mit Ubuntu sind deutlich größer
    - Ubuntu Basis ist mindestens doppelt so groß wie Alpine Images
    - Entscheidungen zur Wahl dokumentieren
    - Wird von Ralf dokumentiert

- Entwicklungsumgebung System Voraussetzungen
  - nicht unbedingt relevant
  - müsste deutlich stärker recherchiert werden warum man System Ressourcen benötigt
  - evtl. sollte das Team herausfinden wie viel System Ressourcen das Programm benötigt
- Wie viel wird noch benötigt bis zum Kundengespräch
  - dieselben Voraussetzungen wie bei Herr Rößler
  - Schönheitsfehler ausbessern
  - optisch und funktional optimal gestalten
- Welche Aufgaben hat jeder bis zum Code-Review
  - Kevin REST Interface entwickeln/ausbauen
  - Pavi Corpus für SWB4 entwickeln und Angular UI Fixes(responsive,...)
  - Ralf Verbindung Angular Admin Interface Funktionalitäten bearbeiten. Mit Kevin und Pavi Kompatibilität zwischen REST und Angular absprechen
  - Jeder seinen Code besprechen und vorerst im Team üben

# Protokoll 10.12.21:

## Achte Besprechung/ Code Review

Teilnehmer:

Andreas Rößler, Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

Inhalte:

- Vorstellung des derzeitigen Standes des Projektes
  - Chat-Interface
  - Admin Interface
  - Korpus Interface (Fragen, Antworten hinzugefügt)
  - Funktionalität des Chatbots
- Besprechung des Codes
  - Welche Architektur wir benutzen
  - Verwendung des NLP.js Framework der AXA Group für den Chatbot
  - REST API in der NodeWebApp
  - Angular Struktur der Komponenten
  - Socketio bidirektionale Verbindung
- Socketio
  - Angular ChatInterface hat einen Socketio Client
  - NodeWebApp hat einen Socketio Server
  - Angular leitet die Fragen des Chats an den WebServer weiter
  - Der Webserver reagiert auf die Fragen und sendet die Antwort

- Node Server
  - hat zusätzlich das NLP.js Framework eingebunden
  - in NLP.js werden Module in Containern gebootstrapped (angehängt)
  - Durch das Container System kann man eigene Module oder Plugins für das Framework schreiben
  - Socketio Server wurde als Module für das Framework implementiert
  - REST API wurde für den Zugriff auf mongoDB entwickelt
- REST API
  - Methoden für CRUD wurden implementiert
  - Bisher werden noch nicht alle Methoden verwendet
- Verbesserungen, die geplant sind
  - Keycloak soll beim Erstellen des Docker Container „UNIX LF“ verwendet statt „Windows CR LF“
  - Keycloak hat Probleme „CR LF“ Formatierte Shell Scripte auszuführen
  - Code soll soweit möglich verbessert werden
  - REST API soll durch Keycloak geschützt werden
  - Allgemein ein Authentifizierungsverfahren, um Personen zu unterscheiden (bisher nur im Adminbereich möglich)
  - User Interface CSS soll allgemein ausgebessert werden
  - Intents sollen per Formular eingefügt werden können
- Recherche für das Team
  - „Connection Pool“ für Restanfragen
  - Mongodb Client auch einmalig initialisieren
  - Angriffsszenarien des Chatbots von Dritten

- Geplante Projektschritte
  - CI/CD Tests des Chatbots
  - Kundengespräch mit Herr Renz
  - Remote Testing der Chat App (Tester schon vorhanden)
- Besprechung des Praxissemesters und späteren Bachelorarbeit
  - wie man sich bewerben sollte
  - Laptop wird von der Firma gestellt
  - was man für Projekte machen kann
  - wo man die Arbeit ausübt
  - wenn man ansprechen kann
  - wann man den Betreuer der Hochschule informieren sollte
  - Booster Impfung in Stuttgart
- Teambesprechung
  - geplante Änderungen der Retrospektive wurden weitgehendst umgesetzt
  - einen Abend wurde gemeinsam im Team ein Spiel gespielt
  - Alle haben am Mittwoch allen Ihren bisherigen Stand des Projektes vorgestellt und Fragen geklärt
  - Team versteht sich jetzt besser und harmoniert im Bearbeiten der Aufgaben
  - die nächsten Wochen wird beobachtet welche Änderungen benötigt werden, um gemeinsam geplantes effizienter umzusetzen

# Protokoll 17.12.21:

## Neunte Besprechung

Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

Inhalte:

- Vorstellung des derzeitigen Standes des Projektes
  - Änderung am Admin Interface
  - Bereich Allgemein verbunden mit REST
  - Keycloak Authentifizierung für REST integriert
- Admin Interface Allgemein
  - Icon lässt sich auswählen
  - Icon wird auf allen angemeldeten Seiten angezeigt
  - Name lässt sich ändern
- Admin Interface Corpus
  - Styling komplett geändert
  - Dialog zum Hinzufügen von Intents
  - Tooltips für Button

- Admin Interface Corpus geplante Änderungen
  - Add Button sichtbarer machen (z.B. Kreis)
  - Add Intent Button in die Karte nehmen
  - Dropdown Menü anpassen
- MongoDB Init Script.js
  - corpus als json einbinden
- RESTAPI mit Keycloak gezeigt
  - Routes haben Rollen zum Authentifizieren
  - Client Login Authentifizierung soll im Chat Interface eingebunden werden
  - Erklärt wie die Autorisierung mit Keycloak im Backend funktioniert
- Nächster Termin
  - Kundengespräch
  - geplante Ausbesserungen soweit wie möglich erledigt

# Protokoll 22.12.21:

## Zweiter Kundentermin

Teilnehmer:

Alexander Renz, Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

Inhalte:

- Vorstellung des ChatBots
  - UI und Funktion des Chat Interfaces vorgestellt
  - UI und Funktion des Admin Interfaces vorgestellt
- Chat Interface
  - Funktion vorgestellt
  - gezeigt wie man mit dem ChatBot schreiben bzw. reden kann
  - gezeigt wie der ChatBot antwortet
- Admin Interface Allgemein
  - gezeigt wie das Icon des ChatBots geändert werden kann
  - gezeigt wie man den Namen des ChatBots ändern kann
- Admin Interface Korpus
  - vorgestellt wie man neue Einträge für den ChatBot erstellt
  - das Wissen des ChatBots wird in Intent, Utterances und Answers abgebildet
  - gezeigt wie der ChatBot antwortet

- Verbesserungen für das Chat Interface
  - Sicherstellen dass der ChatBot Fragen und Antworten oder Intents, die neu eingegeben wurden beantworten kann.
- Verbesserungen für das Admin Interface
  - darauf achten, dass die Sprache des Interfaces einheitlich ist
  - entweder alles in Englisch oder Deutsch
  - Sicherstellen, dass die Elemente der Website alle korrekt dargestellt werden
- Verbesserungen für das Admin Interface Korpus
  - Darauf achten, dass Eingaben übernommen werden
  - Evtl. die Eingabe von Intents mit Auto-Vervollständigung ergänzen
  - Die Buttons zum Hinzufügen der Einträge sind nicht selbsterklärend
  - Evtl. ein Tutorial zum Eingeben von Intents erstellen.
  - Evtl. erklärende Tooltips für den Kontext einbauen
  - Die Dynamisch geladenen Intent Cards könnten mit einem Paginator besser dargestellt werden.
  - Optische Ausbesserungen, dass Buttons nicht in Komponenten herumrutschen
- Besprechung welche Änderungen sehr wichtig sind bis zur End-Präsentation
  - Am Besten, die Features die bereits implementiert sind soweit ausbessern das alles stimmig aussieht
  - neue Features könnten Schwierigkeiten erzeugen, die viel Zeit kosten
  - alle eingetretenen Fehler wie der ChatBot antwortet nicht auf den Kontext usw. müssen ausgebessert werden
- Empfehlungen zum Abliefern eines guten Ergebnisses
  - Gemeinsam die Arbeit ordentlich aufteilen auf alle Teammitglieder
  - funktionale Features müssen funktionieren
  - optische Darstellung sollte fehlerfrei dargestellt werden, damit ein besserer Eindruck entsteht

# Protokoll 12.01.22:

## Besprechung der Endpräsentation

### Teilnehmer:

Michael Watzko, Kevin Sautner, Pavithra Sureshkumar, Ralf Zeller

### Inhalte:

- Vorstellung der Präsentation
  - Struktur der Präsentation
  - ChatBot Anwendung
- Präsentation
  - Struktur kann beibehalten werden
  - evtl. Zeit mit Erklären von verwendeten Technologien füllen
  - Gruppe soll Präsentation proben
  - Sicherheits Slides über den ChatBot von Kevin muss noch ergänzt werden
- ChatBot Anwendung
  - auf 300% Vergrößern beim Vorstellen
  - ChatBot Anwendung
  - 1-2 Intents zum Vorstellen vorbereiten
  - Änderungen zeigen
  - UI in Englisch verfassen und Inhalte des Korpus in Deutsch
  - Fragen in einem Textdokument vorbereiten
- Reflektion Lernfortschritt
  - Reflektion für die Gruppe bisher vorhanden

- Milestones Liste Reflektion
  - Änderungen an Terminen rot kennzeichnen und beschreiben
  - evtl. ein einfaches Gantt-Diagramm anfertigen oder Tabelle um Spalte zum Vergleich ergänzen
- Aufteilung des Teams
  - jedes Teammitglied Aufgabenbereiche und Tätigkeiten ergänzen
  - evtl. Unterpunkte zu den Aufgaben
- Installations- und Administrationshandbuch
  - bisheriges Readme im Repo ausführlicher ergänzen
  - Angular muss beim Rechner installiert sein
- Lizenzen
  - Technologien verwenden Opensoure oder MIT Lizenz
  - Bilder austauschen gegen Opensource lizenzierte Bilder
- Enddokumentation
  - Protokolle ergänzen
  - Kapitel über Sicherheit wird von Kevin verfasst
- Restliche Aufgaben der Gruppe
  - Deadlines setzen und verbindlich Aufgaben vereinbaren
- Endpräsentation am Freitag
  - Präsenz oder Remote Herr Rössler anschreiben und Information dazu holen

## Literatur

- Angular Icon*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://angular.io/assets/images/logos/angular/angular.svg#>.
- Blackberry-messenger-live-free*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://cdn.geckoandfly.com/wp-content/uploads/2016/01/blackberry-messenger-live-free.jpg>.
- Briar*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://cdn4.geckoandfly.com/wp-content/uploads/2018/08/briar.jpg>.
- Chatbot-cost-calculator*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://risingmax.com/blog/ai-based-chatbot-cost-calculator/>.
- Cleverbot*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://www.cleverbot.com/>.
- Keycloak Icon*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://www.katacoda.com/sebastienblanc/avatar>.
- MongoDB Icon*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: [https://webimages.mongodb.com/\\_com\\_assets/cms/kpo5kblefbjq79065-Horizontal\\_Default.svg?auto=format%252Ccompress](https://webimages.mongodb.com/_com_assets/cms/kpo5kblefbjq79065-Horizontal_Default.svg?auto=format%252Ccompress).
- NLP Icon*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://ecomschool.uk/wp-content/uploads/2020/01/NLP.png>.
- Node.js Icon*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://cdn.freebiesupply.com/logos/thumbs/2x/nodejs-1-logo.png>.
- Regex Icon*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: [https://upload.wikimedia.org/wikipedia/commons/thumb/d/d3/Toolbaricon\\_RegEx.svg/1280px-Toolbaricon\\_RegEx.svg.png](https://upload.wikimedia.org/wikipedia/commons/thumb/d/d3/Toolbaricon_RegEx.svg/1280px-Toolbaricon_RegEx.svg.png).
- Socket.io Icon*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://socket.io/images/logo.svg>.
- Telegram Bild*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: <https://cdn2.geckoandfly.com/wp-content/uploads/2016/01/telegram-chat-secure.jpg>.
- Tim WhatsApp*, [o. D.] **online**[besucht am 2021-11-11]. Abger. unter: [https://s3.amazonaws.com/cdn.freshdesk.com/data/helpdesk/attachments/production/42438944/original/Aw4-xHy7s\\_EILasM39zbaxogNI7PZdtvzw.png?1545221722](https://s3.amazonaws.com/cdn.freshdesk.com/data/helpdesk/attachments/production/42438944/original/Aw4-xHy7s_EILasM39zbaxogNI7PZdtvzw.png?1545221722).