DOKUMENTATION LB2

Modul 151

Blaser und Esteban

INHALTSVERZEICHNIS

ANALYSE

AUFGABENSTELLUNG

Das Ziel ist eine 3-Tier WEB Applikation zu entwickeln. Dabei wird das Thema, die Programmiersprache und Framework vom Team selbst gewählt. Wichtig ist dabei, dass die Applikation eine übersichtliche 3-Tier und MVC Architektur (Hybride Architektur) aufweist und das eine Webapplikation mit einer Datenbank in Verbindung gebracht worden ist.

PROJEKTIDEE

BESCHREIBUNG

Wir entschieden uns als Team eine ToDo List Applikation zu entwickeln mit dem Ziel unsere Kreativität und Ideen ohne Einschränkung einzusetzen, sowie auch ein Nutzen aus unserer Applikation ziehen zu können. In unsere App hat man unterschiedliche ToDo Listen, die unterschiedliche Tasks beinhalten. Diese können vom Benutzer selbst definiert werden, sodass jeder Benutzer die Möglichkeit hat personalisierte Listen und Tasks zu erstellen.

FUNKTIONALITÄTEN

Unsere Applikation erlaubt es dem User Todo Listen, wie auch Aufgaben in unserem System konsistent abzuspeichern. Da diese Daten empfindlich sein können, werden diese sicher abgespeichert und werden daher nur vom jeweiligen User abgerufen. Für das haben wir eine Login Seite erstellt, welche sich um Authentifizierung und Weiterleitung kümmert.
Der Benutzer hat die Möglichkeit mehrere ToDo Listen zu erstellen. Dabei kann er für jede ToDo Liste spezifische Aufgaben erstellen. Diese kann er jeder Zeit abhacken, löschen oder bearbeiten und dasselbe gilt für die ToDo Liste selbst. Wichtig ist bei den ToDo Listen, dass beim Löschen auch alle dazugehörigen Tasks mitentfernt werden.

ÜBERSICHT DER FUNKTIONALITÄTEN:

- → Registrieren
- → Einloggen
- → To Do Listen erstellen/löschen/bearbeiten
- → Aufgaben der jeweilige To Do Liste ansehen
- → Aufgaben erstellen/ bearbeiten(Name & Status)/löschen
- → Ausloggen

PLANUNG

NAMENSKONVENTION

Die Namenskonventionen für die erstellte und in Zukunft zu wartende Datenbank und Dateien im Backend wie auch im Frontend sind hier festgehalten und geschildert.

FRONTEND

Im Frontend folgt die Struktur der Files den Regeln des Atomic Designs. Um eine bessere Überblick zu ermöglichen, wird für jede File-Gruppe ein Ordner erstellt. Das Ziel wäre die einzelne Komponente der Applikation in den Atomic Design herunter zu brechen.

NAMENSGEBUNG:

Typ	Konvention
Datei	Pascal Case
Ordner (Atomic Design Struktur)	Lower Case
Order der Komponente	Pascal Case

Komponente	Pascal Case (Gleich wie Dateiname)
Interface	Pascal Case
Methode	Camel Case
Variable	Camel Case
Konstante	Upper case und Snake case
CSS IDs und Klassen	Lower Case und Camel Case

BACKEND

Im Backend ist eine 3 Layer Architektur vorhanden. Für jede Entität werden jeweils die drei Layers erstellt. Um die Files besser zu strukturieren, werden pro Entität ein Ordner erstellt, welcher alle drei Layer beinhaltet. Die genauere Erläuterung der drei Layers finden sie im Kapitel Design.

Layer	Konvention
Model	Der Name der Entität Benutzer
Controller	Entitätsname und Suffix Controller z. B. BenutzerController
Service	Entitätsname und Suffix Service z. B. BenutzerService
ServiceImpl	Entitätsname und Suffix ServiceImpl z. B. BenutzerServiceImpl.
Repository	Entitätsname und

NAMENSGEBUNG:

Typ	Namensgebung
Package	Lower Case
Datei	Pascal Case
Klasse / Interface	Pascal Case (Gleich wie Dateiname)
Methode	Camel Case
Variable	Camel Case
Konstante	Upper Case und Snake Case

DATENBANK

Art	Konvention
Umlaute umschreiben	ae, oe, ue
Tabellen	Camel Case z. B. ToDoListe
Primärschlüssel	Entitätsname Singular und Suffix _id
Fremdschlüssel	Entitätsname Singular und Präfix id_
Attribute	Lowercase und bei der Zusammensetzung der Wörter Camel Case

ENTWICKLUNGSUMGEBUNG

Was	Mit Was	Entwicklungsumgebung
Frontend	React und Typescript	Visual Studio
Backend	Java und Springboot	Intellij
Datenbank	MariaDB	Eigener Server -> Externes Hosting
Dokumentation		Word

Use Case, ERD, ERM, Aktivitätsdiagramm		Draw.io

ZEITPLAN

Wie man im untenstehenden Gantt Model sehen kann, haben wir die Planung in verschiedene Abschnitte unterteilt. Der erste Abschnitt heisst «Vorbereitungsphase» und befasst sich mit der Planung, dem Einrichten der Dokumente und Arbeitsstationen, sowie der Entwicklungsumgebung. In der sogenannten «Arbeitsphase» geht es um die effektive Datenbankstruktur, Implementierung vom Backend und Frontend, genauer gesagt ums Umsetzen. Im Abschnitt «Testen» geht es um das Testen der Anwendung und der Speicherung der Daten. Der letzte Abschnitt «Produktentwicklung verfolgt das Ziel das Produkt richtig an den Kunden auszuliefern. Unser Ziel ist es die Dokumentation und ein fertiggestelltes Programm zu liefern.

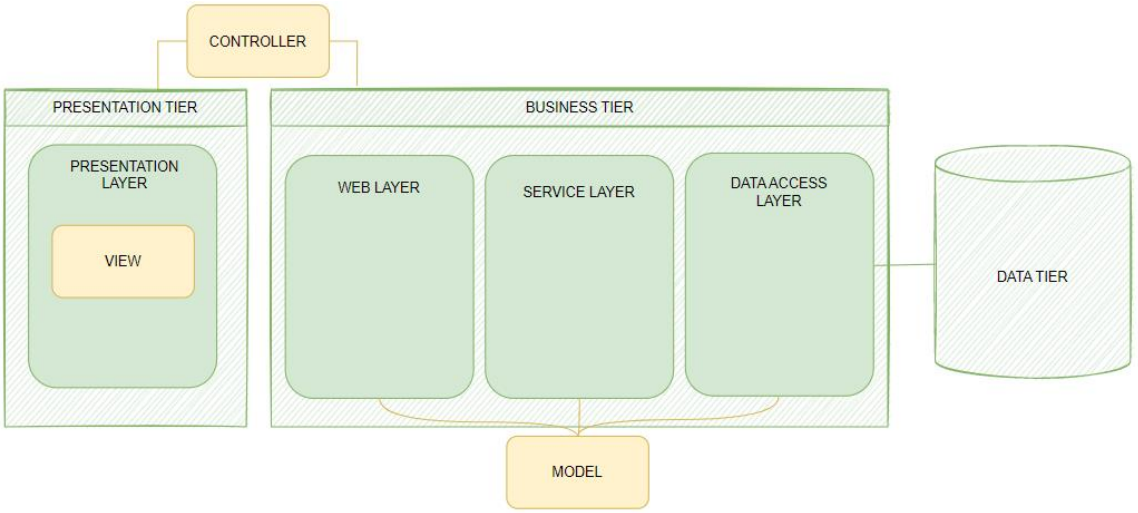
GANTT

VORBEREITUNGSPHASE							
	KW38	**KW39**	**KW40**	**KW41**	**KW42**	**KW43**	**KW44**
Dokumentation	*						
Projekt aussuchen	*						
Kurz Beschreibung des Projekts	*						
Use Cases erstellt	*						
Entwicklungsumgebung vorbereitet	*						
Domain Model		*					
ERM und Tabelle Definition		*					
ARBEITSPHASE							
	KW38	**KW39**	**KW40**	**KW41**	**KW42**	**KW43**	**KW44**
DB implementieren		*					
Backend Struktur 3-Tier Layer		*					
Frontend Struktur		*					
Backend Login							*
Frontend Login Ansicht							
CRUD Endpoints			*				
Navbar mit ToDoListen						*	
ToDoListe und Aufgaben Ansicht						*	
ToDoListe löschen, erstellen und bearbeiten						*	
Aufgabe erstellen und Status auf Done setzen						*	
Registrierung Ansicht						*	
TESTPHASE							
	KW38	**KW39**	**KW40**	**KW41**	**KW42**	**KW43**	**KW44**
CRUD Operationen			*				
ToDoListen erstellen, löschen und bearbeiten			*				
Aufgaben löschen und Status Done			*				
Login, Registrierung und Logout							*
PRODUKT ABGABE							
	KW38	**KW39**	**KW40**	**KW41**	**KW42**	**KW43**	**KW44**
Dokumentation							*
Applikation							*

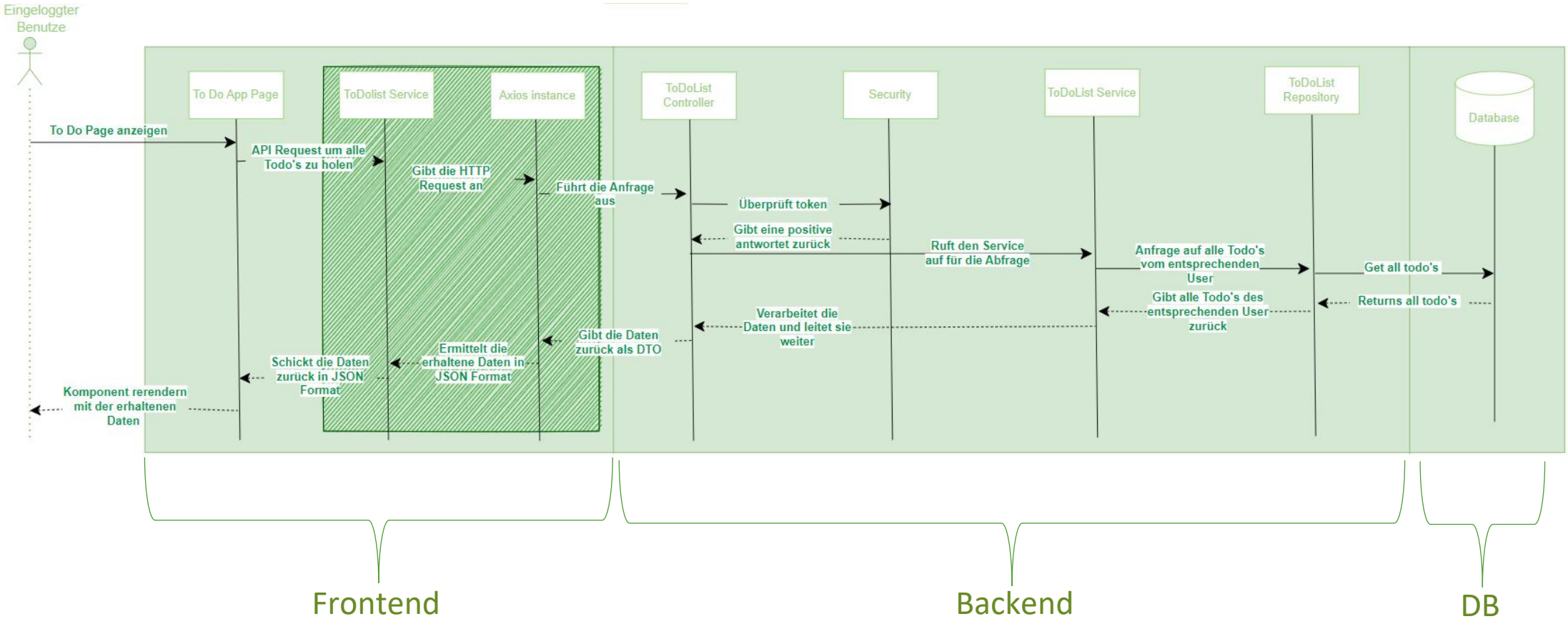
*** TATSÄCHLICHER ABLAUF**

ARCHITEKTUR

In unsere Applikation wird eine Hybride Version von einer drei Tier Architektur und MVC umgesetzt. Um sich das besser vorzustellen siehe Abbildung.

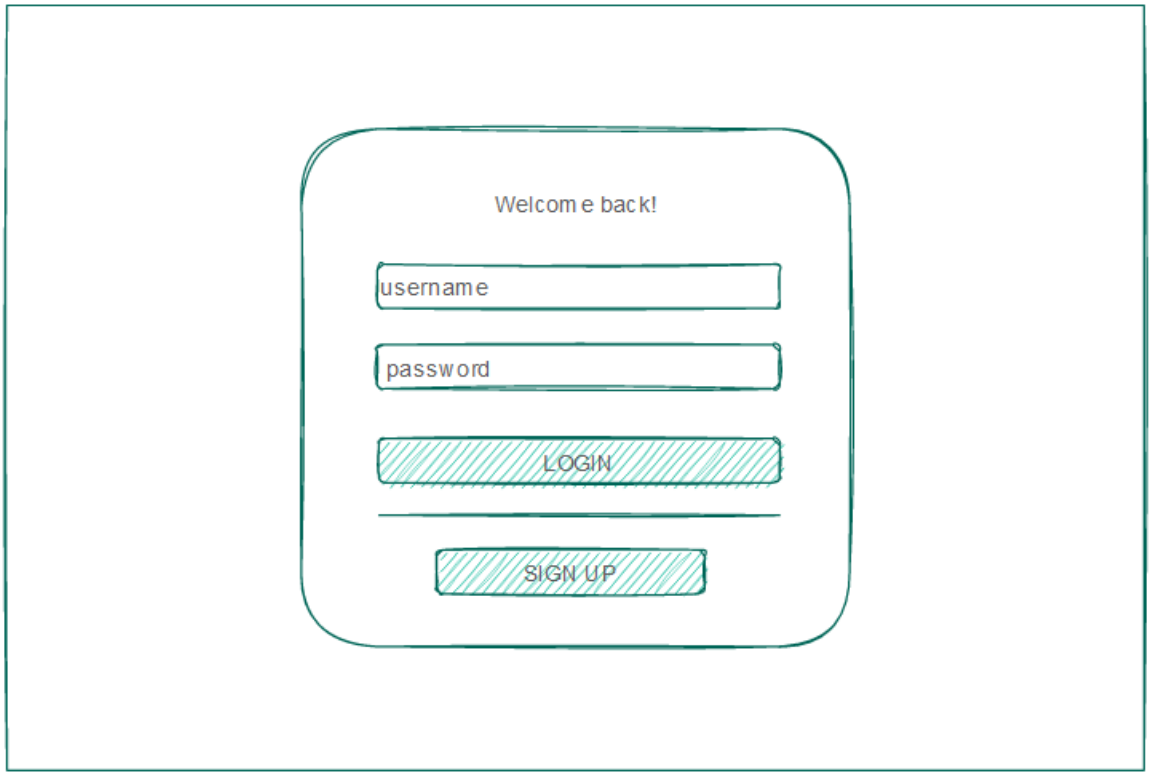


Damit sich die Kommunikation zwischen den Tiers besser vorstellen können siehe Abbildung auf der nächsten Seite.

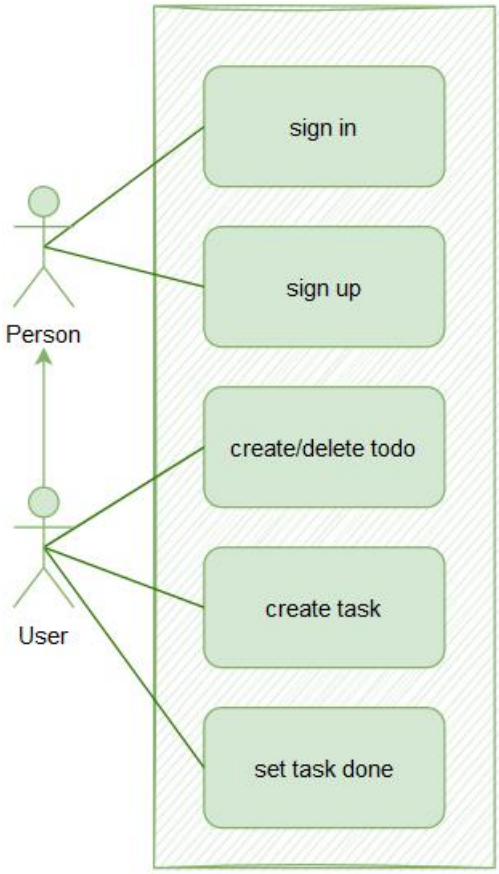

Eingeloggter Benutze
To Do App Page
ToDolist Service
Axios instance
ToDoList Controller
Security
ToDoList Service
ToDoList Repository
Database
To Do Page anzeigen
API Request um alle Todo's zu holen
Gibt die HTTP Request an
Führt die Anfrage aus
Überprüft token
Gibt eine positive antwortet zurück
Ruft den Service auf für die Abfrage
Anfrage auf alle Todo's vom entsprechenden User
Get all todo's
Returns all todo's
Gibt alle Todo's des entsprechenden User zurück
Verarbeitet die Daten und leitet sie weiter
Gibt die Daten zurück als DTO
Ermittelt die erhaltene Daten in JSON Format
Schickt die Daten zurück in JSON Format
Komponent rerendern mit der erhaltenen Daten
Frontend
Backend
DB

DESIGN

SKIZZE



USE CASE

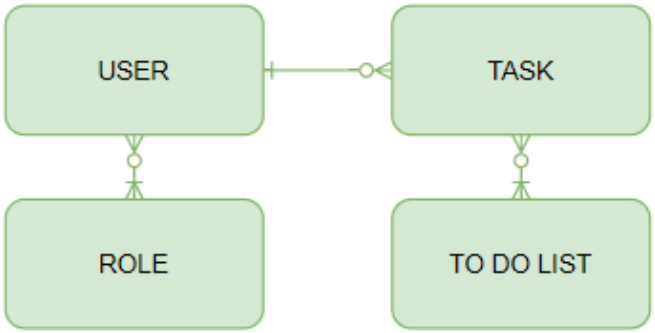


UML

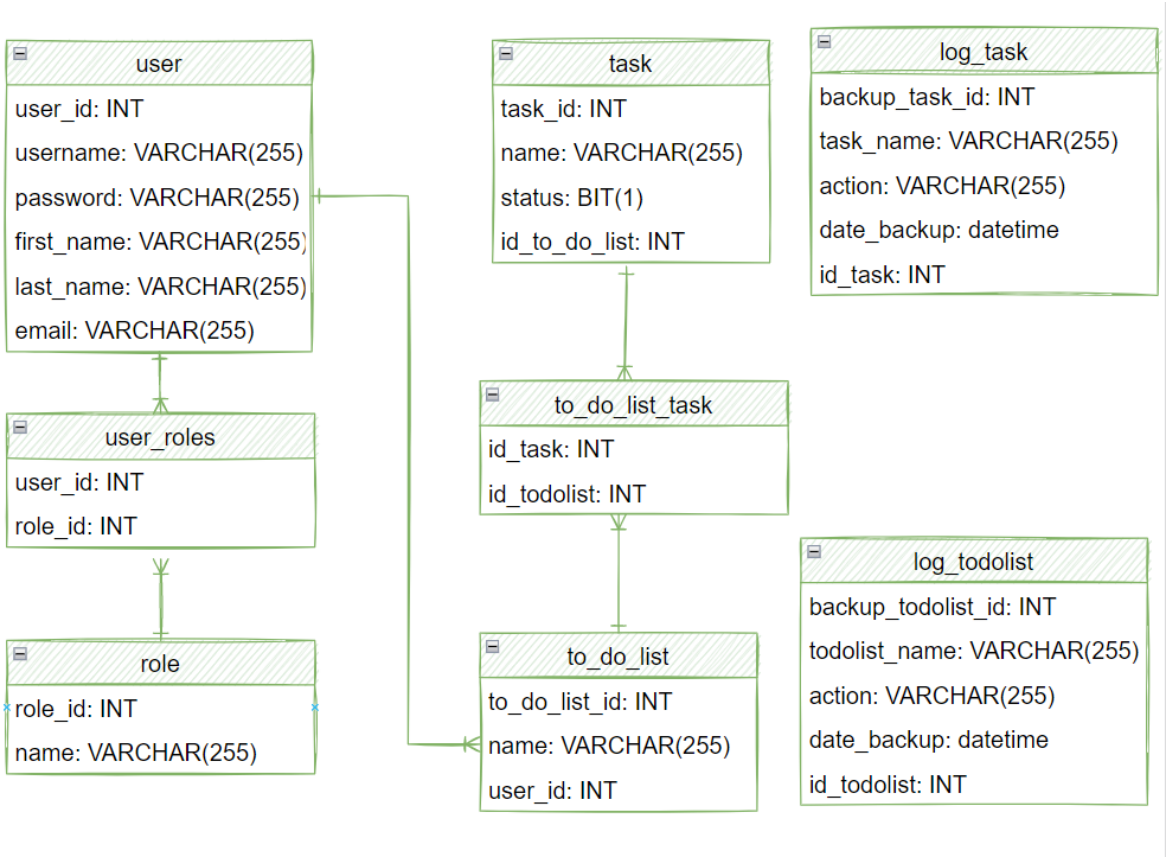
DOMAIN MODEL

Das Domain Model verschafft uns einen Überblick der Tabellen und deren Beziehung. Sodass wir uns ein besseres Verständnis und Überblick verschaffen können. Die genauere Beschreibung der einzelnen Tabellen und deren Beziehung finden sie im ERD oder bei der Tabellenbeschreibung.

VERSION 1.1



ERD



Log_task und Log_todolist sind Tabellen, die von Trigger befüllt werden, wenn eine Änderung an einer der folgenden Tabellen gemacht werden: to_do_list und task

TESTKONZEPT

FRONTEND

Im Frontend überprüfen wir die Funktionalitäten manuell und halten uns dabei an diese Vorlage:

Use Case Nr. #1	
Vorkondition	
Akteur	
Trigger	
Eingabe	
Kurzbeschreib	
Erwartetes Ergebnis	
Erhaltenes Ergebnis	
Status	**OK**

Hier noch ein ausgefülltes Exemplar:

Use Case Nr. #1 User Einloggen	
Vorkondition	Die Seite wurde geöffnet
Akteur	Benutzer mit einem Account
Trigger	Login
Eingabe	Login Daten (username, password)
Kurzbeschreib	Der Benutzer loggt sich mit seinen Daten ein.

Erwartetes Ergebnis	Der Benutzer konnte sich erfolgreich einloggen, erhält eine Nachricht und wurde weitergeleitet.
Erhaltenes Ergebnis	Der Benutzer ist erfolgreich eingeloggt, hat Nachricht erhalten & wurde weitergeleitet.
Status	**OK**

Ebenso kann das Frontend mit Cypress getestet werden, dabei kann man die Einzelne Komponente, wie auch die ganze Applikation getestet werden

BACKEND

Bei Backend-Tests liegt der Schwerpunkt auf der Untersuchung des Systemverhaltens auf der Datenbankebene. Bei fehlerhafter Durchführung kann es zu schwerwiegenden Komplikationen wie Deadlock, Datenbeschädigung, Datenverlust usw. kommen. Wir haben uns vorgenommen das Backend mit JUnit Tests zu testen, dabei werden einzelne Funktionalitäten getestet.

Eine weitere Möglichkeit wäre das Backend mit Postman zu testen und dabei die http Requests und Antworten zu überprüfen.

FAZIT

Im Team sind wir durch die ständige Kommunikation und Austausch sehr schnell vorangekommen. Da wir die Applikation von neu Aufsetzten hatten wir die Möglichkeit unser Wissen zu vertiefen und zu erweitern. Wir haben einiges dazu gelernten vor allem durch unsere Fehler und Probleme während der Entwicklung.

Das Arbeiten an diesem Projekt war sehr erfreulich. Als Gruppe funktionieren wir gemeinsam sehr gut, da wir eine ähnliche Herangehensweise an Projekte haben. Ebenfalls ist uns beiden sehr wichtig, dass jeder in der Gruppe seinen Beitrag leistet und die Arbeit gerecht verteilt ist, daher hatten wir beide den Anspruch unsere beste Leistung zu zeigen und den jeweilig anderen zu unterstützen. Das Arbeiten mit MariaDB, Spring Boot und React hat uns beiden Spass gemacht und wir konnten nicht nur Neues sondern auch bekanntes anwenden. Mit dem Ergebnis sind wir ausserdem sehr zufrieden.

Was uns allerdings schwer fiel, war die Authentisierung im Frontend mit dem im Backend zu verbinden. Wir hatten einige Schwierigkeiten, die uns sehr viel Zeit raubten und mit welchen wir nicht rechneten. Durch viel Recherche und den gegenseitigen Support im Team konnten wir jedoch alle Probleme lösen.

Da in diesem Modul der Fokus auf der Verbindung zwischen Datenbank und Webapplikation steht, wollten wir uns herausfordern in dem wir unsere Datenbank mit einem externen Server verbunden haben. Ebenfalls hat es uns geholfen nach jeder Implementation einer Funktionalität den Ablauf nachzuvollziehen, in dem wir uns Gedanken dazu gemacht haben, welche Schritte das Framework für uns übernimmt und tätigt. Denn nur so konnten wir unser Wissen erweitern und bekanntes erfrischen, so wie auch das Framework richtig anwenden.