DOKUMENTATION ÜK 223
EVENTS «OUR SPACE»

Noah De Boni, Ruben Schneebeli, Sairam Vijayakumar
NOSER YOUNG AG
08.09.2023

Inhalt

1 Einleitung

1.1 Unser Team

Ruben Schneebeli

Noah De Boni

Sairam Vijayakumar

1.2 Ausgangslage

Unser Team wurde beauftragt, die Social Media-Webseite «OurSpace» zu entwickeln. Auf der Webseite sollen mehrere User Blog-Posts erstellt werden können. Die Posts sind zugänglich für die Öffentlichkeit sein. Die Page wird von Admins verwaltet, die Kategorien erstellen und verwalten können. Diese Kategorien können dann von Usern zu Blog-Posts zugeteilt werden. User können Teil einer Gruppe werden.

2 Auftrag

Ziel dieses Projekt ist es sein eine Full-Stack Komponente mithilfe von React, SpringBoot und PostgreSQL zu erstellen, die die unten aufgeführten Bedingungen erfüllt. Dabei wird auf Multiuser-fähigkeit und Sicherheit sowie Dokumentation der Arbeit geachtet. Als Vorgabe erhalten Sie ein funktionierendes rudimentäres Full-Stack Projekt. Dieses Projekt enthält Funktionalitäten, um bestehende User einzuloggen und ermöglicht das Erstellen, Bearbeiten und Löschen von Usern. Login Funktionalität sowie einfaches Routing wurde ebenfalls implementiert.

2.1 Pflicht-Anforderungen

2.1.1 Funktionale Anforderungen

User Rollen & Privilegien:

- Bestehende Rollen und Autoritäten werden bearbeitet oder erweitert um untenstehende Anforderungen zu erfüllen und zu testen.
- Die persönlichen Informationen eines Users sind nur für Administratoren oder den User selbst zugänglich.
- Admins können ausserdem andere Benutzer bearbeiten, erstellen und löschen.

Frontend

- Im Minimum enthält die Applikation:
- Login-Page, die öffentlich zugänglich ist (bereits vorhanden)
- Eine öffentlich zugängliche Homepage (bereits vorhanden)
- eine Homepage für alle eingeloggten User
- Eine Admin-Page (nur für Admins zugänglich).
- Mindestens eine Komponente, um gruppenspezifische Funktionalitäten im Frontend zu ermöglichen.

Security

- Jeder REST-Endpoint soll nur mit sinnvollen Autoritäten zugänglich sein. Dies wird mit automatisierten Tests überprüft.
- Es gibt Bereiche des Front-Ends, die nur für eingeloggte Benutzer zugänglich sind.
- Es gibt Bereiche des Front-Ends, die nur für Admins zugänglich sind.
- Der Authentifizierung-Mechanismus wird mit JSON-Web-Tokens implementiert. (bereits vorhanden)

2.1.2 Gruppenspezifische Aufgabe

- Erstellen Sie ein Event Model, das Informationen über eine geplante Veranstaltung enthält (Gästeliste, Eventname, Datum, Ort).
- Jeder User kann Gast von mehreren Events sein, Admins können nicht an Events teilnehmen.
- Erstellen Sie Endpoints in Ihrer Applikation, um typische CRUD-Operationen an Groups durchzuführen.
- Erstellen Sie einen Endpoint, der alle Teilnehmer einer Veranstaltung auflistet. Benutzen Sie dafür Pagination.

- Alle User können Events erstellen.
- Alle User können alle Information über ein Event einsehen
- Nur der User, der ein Event erstellt hat, kann andere User zur Gästeliste hinzufügen.

2.1.3 Nicht funktionale Anforderungen

Implementation

- Daten werden in einer PostgreSQL Datenbank persistiert, das OR-Mapping wird mit JPA realisiert.
- Ein Frontend mit React (Typescript) wird genutzt.
- Ein Backend Springboot (Java) wird genutzt.
- Der Sourcecode wird täglich in einem GIT-Repository commited.

Testing

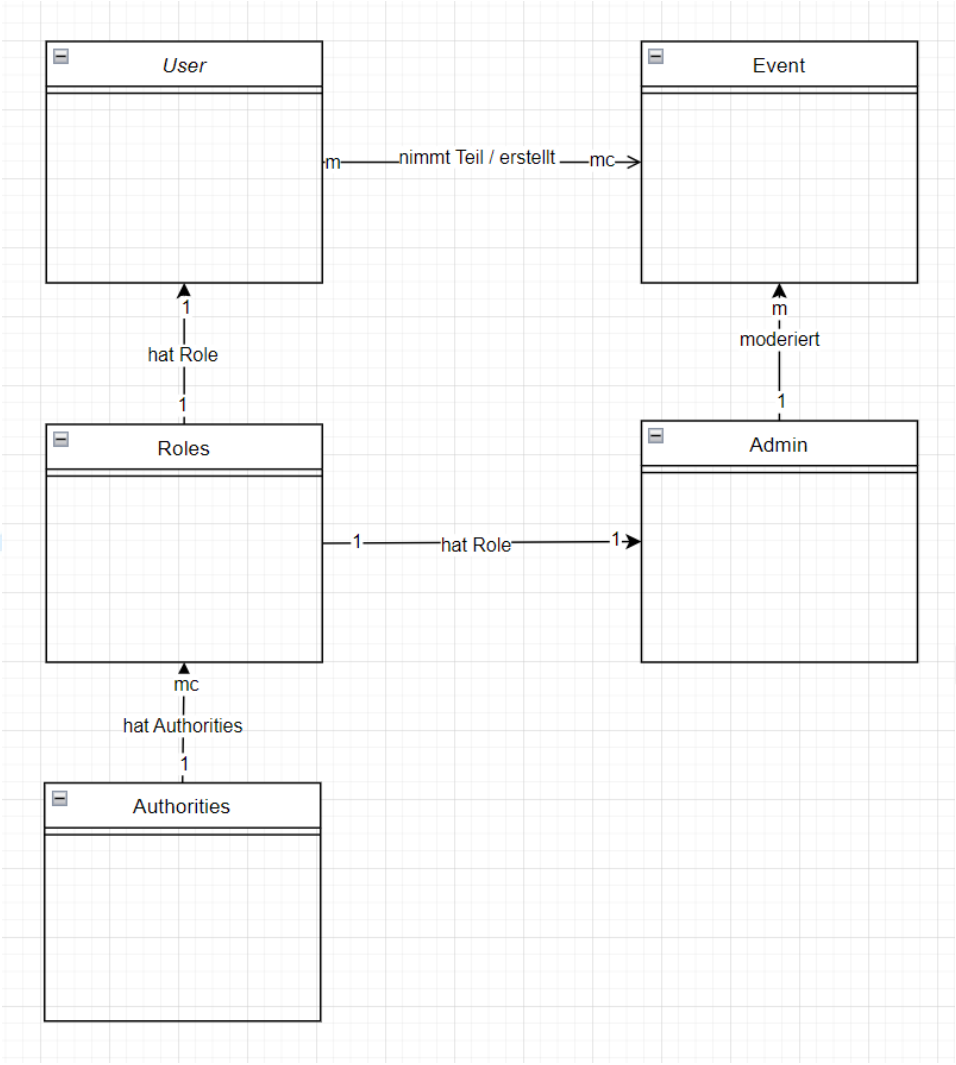
- Generelle Funktionalität aller selbst implementierten Endpoints wird mit Cypress (zwingend), Postman und/oder JUnit getestet.
- Besonderer Wert wird auf das Testen von Zugriffsberechtigungen gelegt.
- Mindestens ein Use-Case wird ausführlicher mit Cypress getestet. Dies beinhaltet im Minimum:
    - Der Endpoint wird mit mehreren Usern & Rollen getestet
    - Mindestens ein Erfolgsfall und ein Errorfall wird getestet.
    - Für diese Fälle werden Use-Cases nach UML-Standard beschrieben.

Multiuserfähigkeit

- Aspekte der Multiuserfähigkeit, wie z.B. Einhaltung der ACID-Prinzipien werden berücksichtigt

3 Domänenmodell

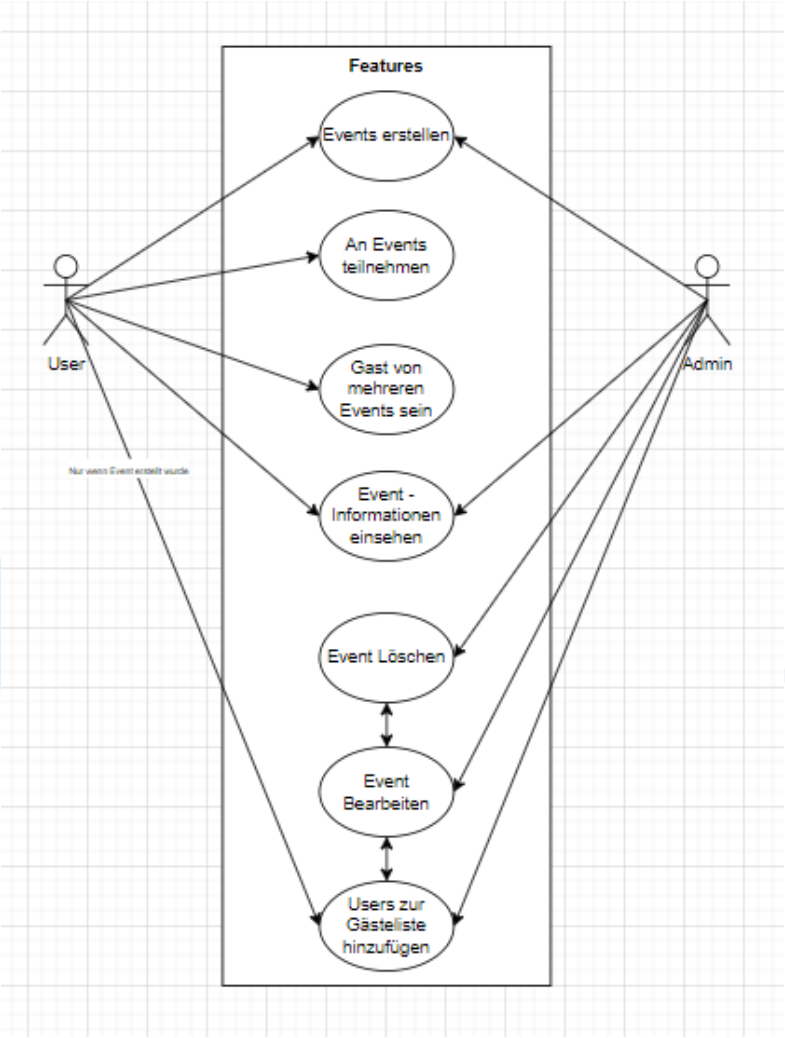
Das Domänenmodell ist eine grafische Darstellung der in der Domäne enthaltenen Entitäten. Die Entitäten zeigen die Eigenschaften und die Beziehungen zwischen den Entitäten. Durch die Ausgangslage und unserem Gruppen spezifischen Auftrag, lässt sich folgendes Domänenmodell resultieren.



4 Use-Case-Diagramm

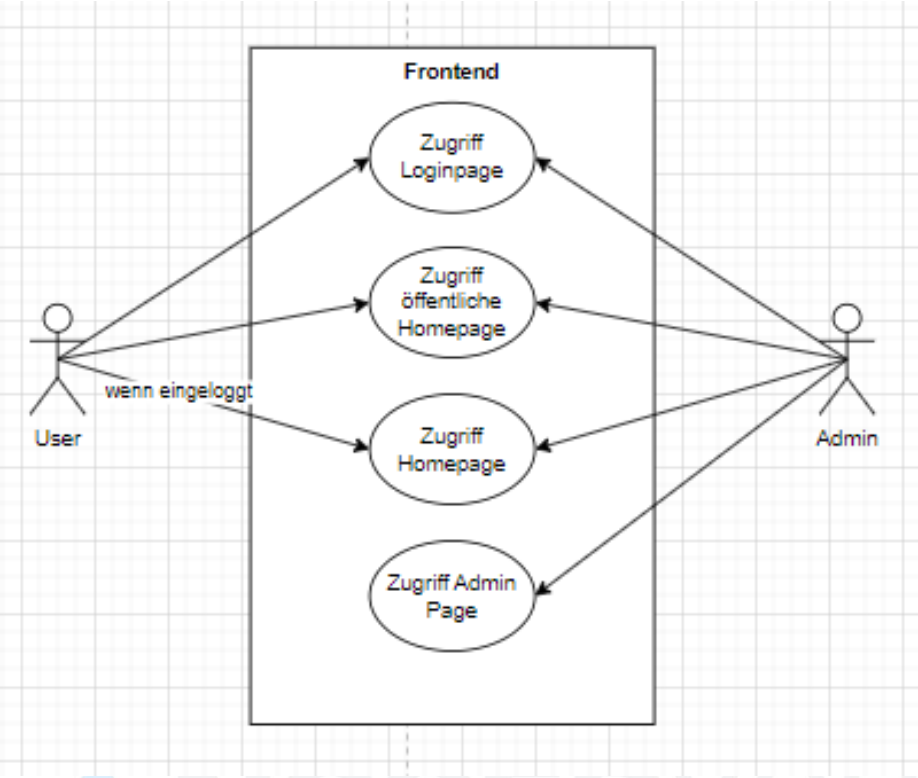
Ein Use Case Diagramm zeigt, die von aussen sichtbaren Interaktionen von verschieden Akteuren an, welche beim System mitwirken. Die folgenden Use Case-Diagramme sind anhand von dem Auftrag erstellt worden.

4.1 Use-Case-Diagram-Features



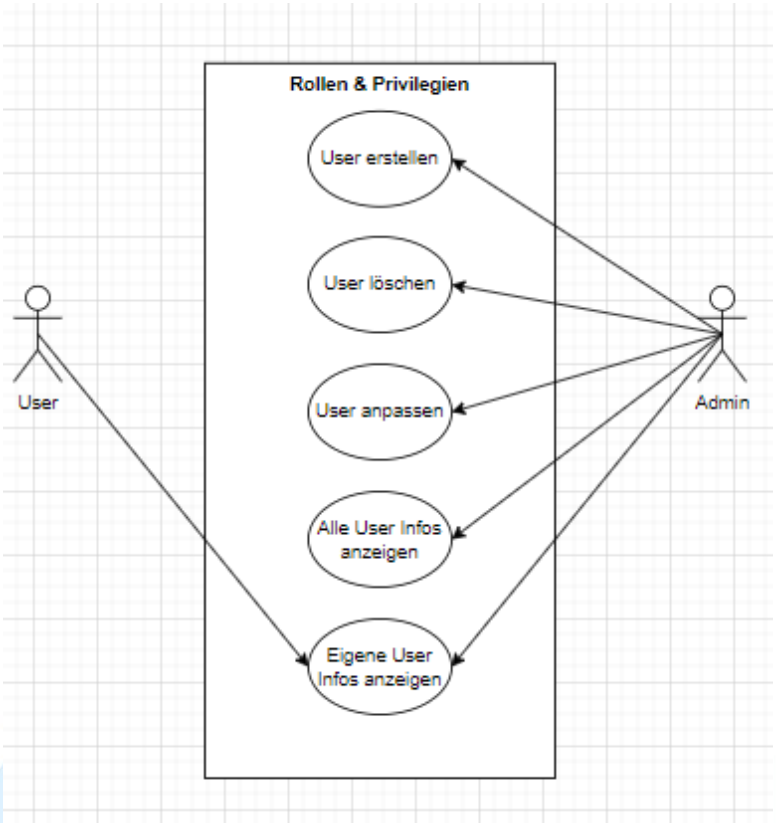
Rollen	User	Admin
Events erstellen	x	x
An Events teilnehmen	x	
Gast von mehreren Events sein	x	
Event Informationen einsehen	x	x
Event löschen		x
Event bearbeiten		x
Users zur Gästeliste hinzufügen	Nur wenn Events erstellt wurde	x

4.2 Use-Case-Diagram-Frontend



Rollen	User	Admin
Zugriff Login Page	x	x
Zugriff öffentliche Homepage	x	x
Zugriff Homepage	Wenn eingeloggt	x
Zugriff Admin Page		x

4.3 Use-Case-Diagram-Rollen & Privilegien

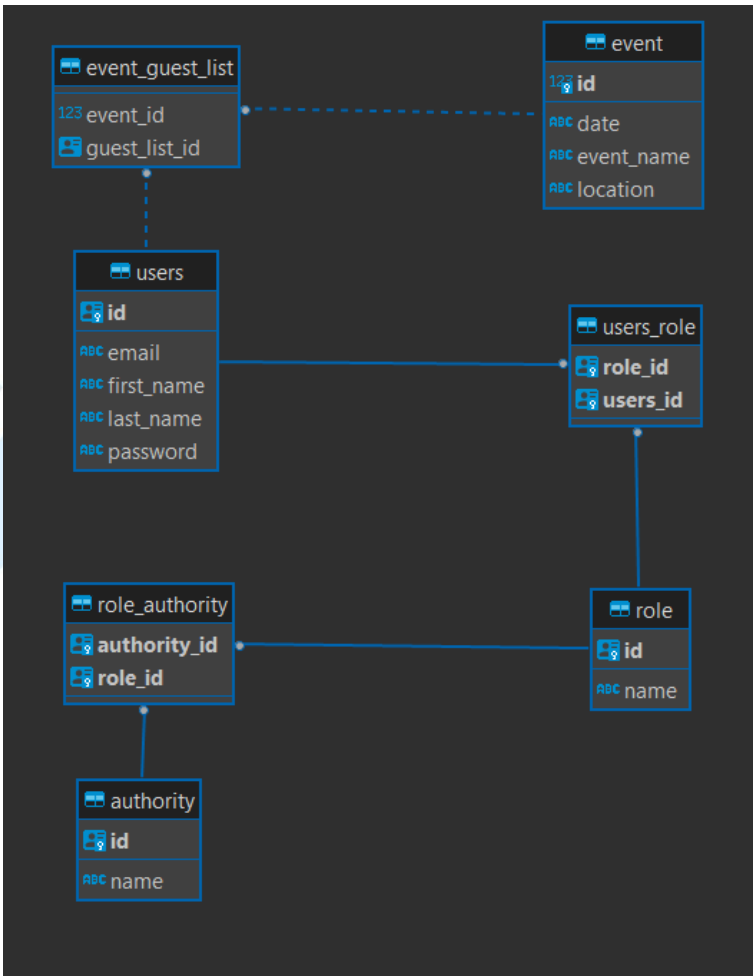


Rollen	User	Admin
User erstellen		x
User löschen		x
User anpassen		x
Alle User Infos anzeigen		x
Eigene User Infos anzeigen	x	x

5 Entity Relationship Diagram (ERD)

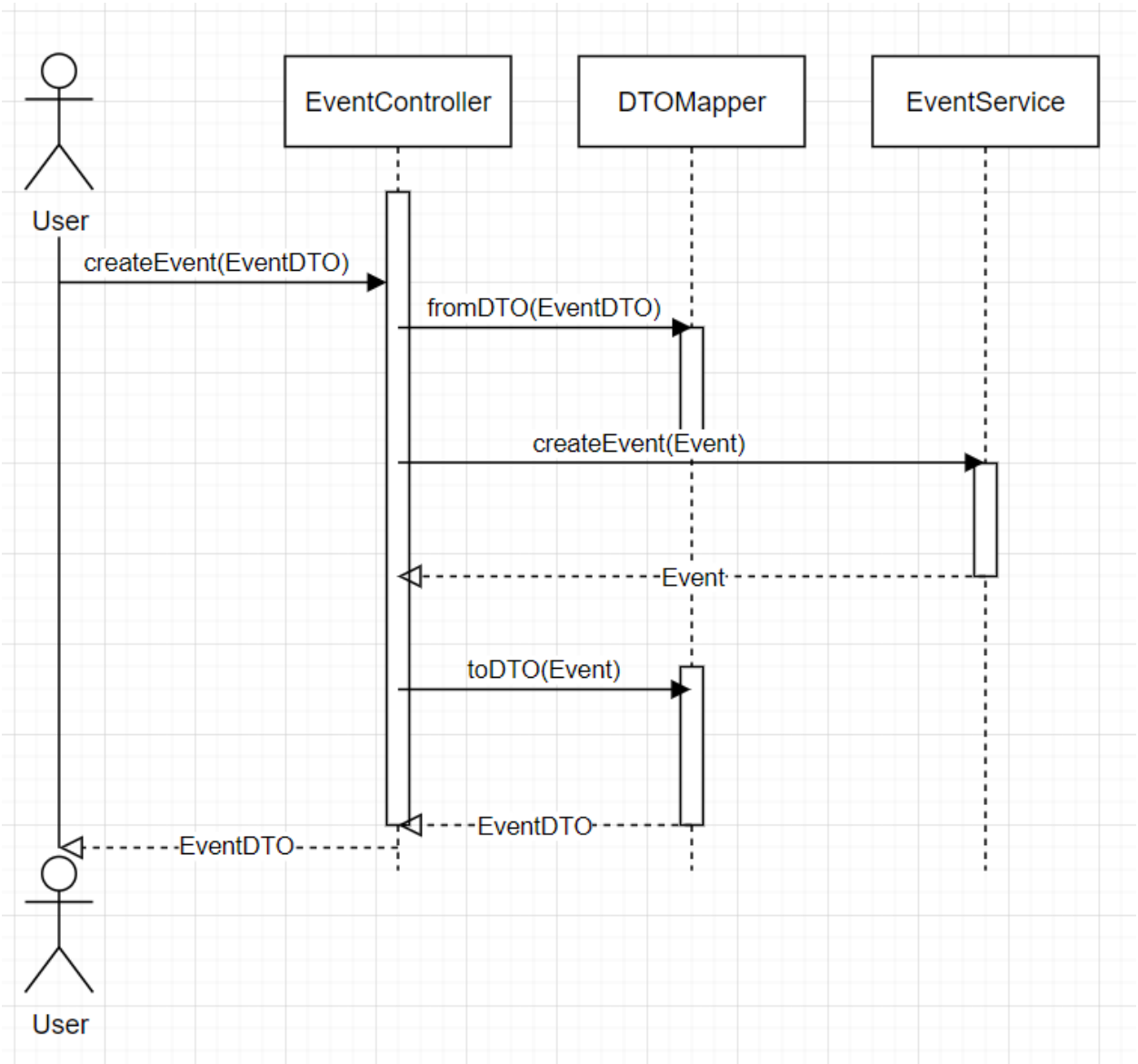
Unsere Daten werden in Postgresql gespeichert. Dies ist unser ERD mit den folgenden Entitäten:

- Authority
- Role
- Role_authority
- Users
- Users_role
- Event
- Even_guest_list



6 Sequence Diagram

Dies ist eine Abbildung von einem Request «Create Event».



7 Testing Strategie

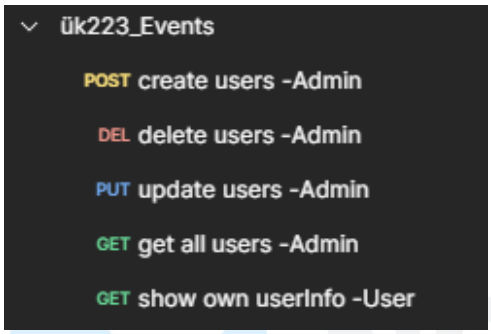
Dieses Projekt wird auf zwei Ebenen getestet. In unsere Gruppe haben wir auf folgende zwei Methoden entschieden:

- Integration tests (postman)
- E2E testing (cypress)

7.1 Postman

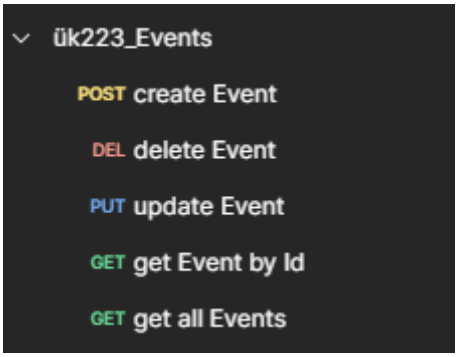
Bei den Postman Tests haben wir uns beim Use Case Diagramm orientiert:

7.1.1 Testfälle Rollen & Privilegien



Dies sind die Testfälle von Rollen & Privilegien. Wir haben folgende Tests geschrieben, weil Admins und Users nicht die gleichen Privilegien haben sollen.

7.1.2 Testfälle Event



Dies sind die Testfälle für die Events. Wir haben folgende Tests geschrieben, weil wir damit die verschiedenen Autoritäten überprüfen wollen.

Die Tests befinden sich als Export beim Backend git Repository:
https://github.com/SairamVijayakumar19/PLJ2023_uek223_team4_event_backend

7.2 Cypress

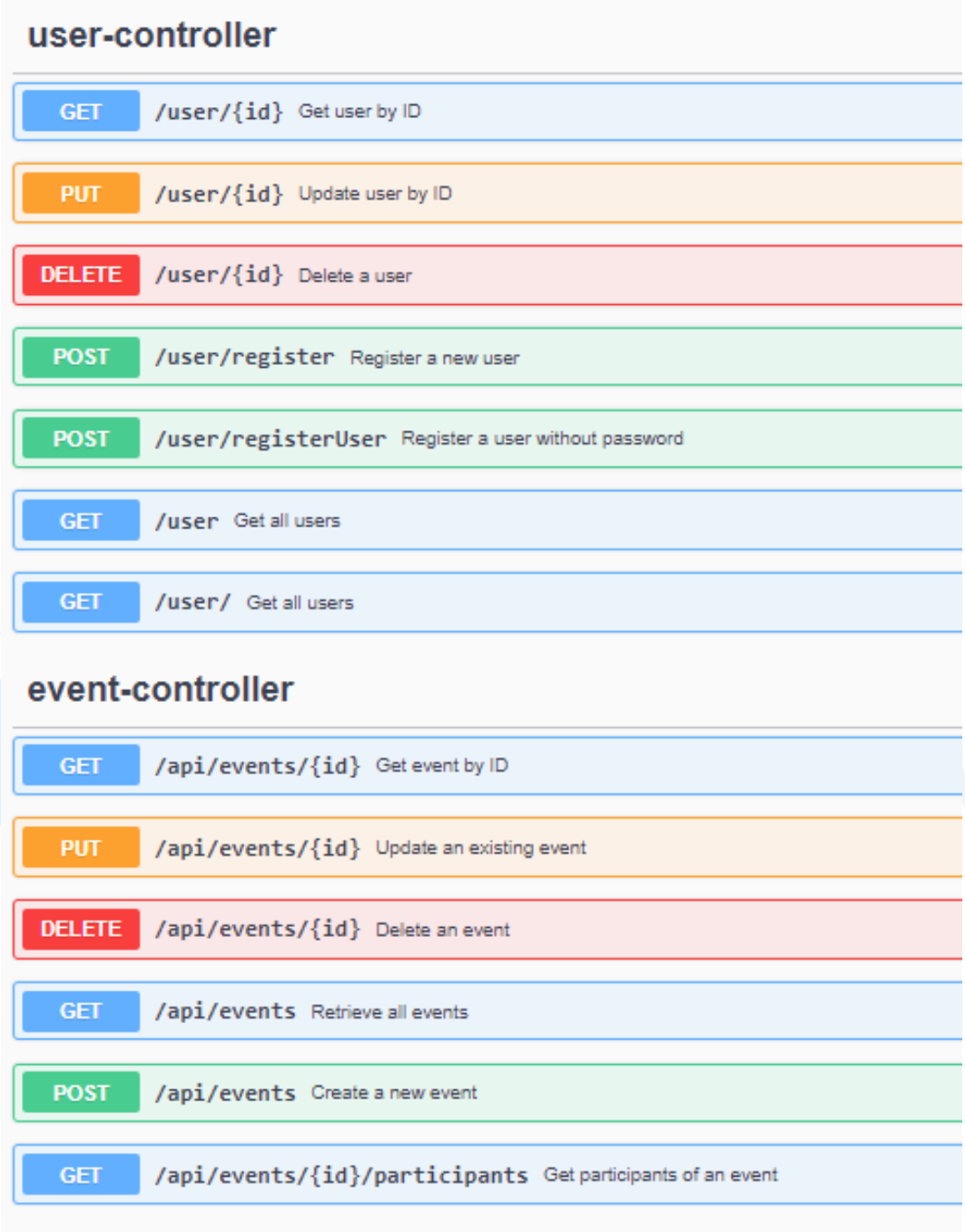
7.2.1 Testfälle Frontend

Um das Frontend zu Test haben wir uns für Cypress entschieden.

Rollen	User	Admin
Descripton	Der Benutzer meldet sich für eine Veranstaltung	Der Admin versucht sich bei der Veranstaltung anzumelden
Preconditions	• User muss eingeloggt sein • Darf nicht Eventgrösse überschreiten	• Der Admin muss eingeloggt sein • Muss Admin Rechte haben
Normal Course	• Benutzer meldet sich an • Benutzer drückt auf Button «teilnehmen» • Bekommt eine Meldung	• Admin meldet sich an • Admin versucht sich für ein Veran. Anzumelden • Kann sich nicht anmelden
Alternative Courses	• Wenn Anzahl Teilnehmer überschritten wird, kommt eine Fehlermeldung	• Der Admin ist nicht in der Lage sich anzumelden, falls es dazu kommt, sollte es eine Fehlermeldung kommen
Exceptions	Keine	

Die Tests befinden sich im frontend git Repository:
https://github.com/SairamVijayakumar19/PLJ2023_uek223_team4_event_frontend

8 Swagger Dokumentation



Mit dem folgenden Link können Sie zur Swagger Dokumentation gelangen: http://localhost:8080/myapi/swagger-ui/index.html#/