UEK 223

Thema: Documentation

Dokumentinformationen

Dateiname: Documentation-uek-223.docx

Speicherdatum: 05.11.2021

Autoreninformationen

Autor: Dominic Ritzmann

E-Mail: ritzmann@noseryoung.com

Inhaltsverzeichnis

[Abbildungsverzeichnis 2](#_Toc87537643)

[Tabellenverzeichnis 2](#_Toc87537644)

[Änderungsgeschichte 2](#_Toc87537645)

[1 Einleitung 3](#_Toc87537646)

[1.1 Sinn und Zweck 3](#_Toc87537647)

[1.2 Referenzdokumente 3](#_Toc87537648)

[1.3 Abkürzungen 3](#_Toc87537649)

[2 Hauptauftrag 4](#_Toc87537650)

[2.1 Funktionale Anforderungen 4](#_Toc87537651)

[2.2 Nicht funktionale Anforderungen 4](#_Toc87537652)

[3 Arbeitsaufteilung 4](#_Toc87537653)

Abbildungsverzeichnis

**Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.**

Tabellenverzeichnis

[Tabelle 1 Versionen 2](#_Toc449701436)

[Tabelle 2 Abkürzungen 3](#_Toc449701437)

Änderungsgeschichte

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Datum** | **Autor** | **Details** |
| 1.0 | 05.11.2021 | Ritzmann | Dokument erstellt |
| 1.1 | 10.11.2021 | Ritzmann | Hauptauftrag, Funktionale Anforderungen, nicht funktionale Anforderungen + Arbeitsaufteilung geschrieben |

Tabelle 1 Versionen

# Einleitung

## Sinn und Zweck

Das vorliegende Dokument beschreibt die Arbeitsaufteilung und den Ablauf des ÜK 223.

## Referenzdokumente

1. …
2. …

## Abkürzungen

|  |  |
| --- | --- |
| **Abkürzung** | **Beschreibung** |
| NYP | **N**oser **Y**oung **P**rofessionals |

Tabelle 2 Abkürzungen

# Hauptauftrag

Unser Auftrag war es, eine Multiuser Applikation zu erstellen, nach bestimmten Voraussetzungen. Wir mussten ein Blog Post Model erstellen, welches Informationen von Blogeinträgen enthält. Jeder Eintrag hat einen Author oder eine Authorin. Nur der oder die Author/in oder ein Administrator kann den Blogpost bearbeiten oder löschen. Wir mussten Endpoints erstellen, um die CRUD Operationen sicherzustellen. Die GET Methode soll Pagination und eine Sorting Funktion haben. Auch Unautorisierte Benutzer sollen Blogs lesen können.

## Funktionale Anforderungen

### User, Rollen & Autoritäten

Um die funktionalen Anforderungen zu erfüllen, müssen wir mehrere User, Rollen und Autoritäten erstellen. Die User, Rollen und Autoritäten müssen in der Datenbank abgespeichert werden können. Ausserdem sollen neue erstellt, gelöscht oder verändert werden. Wie viele Rollen und Autoritäten wir einbauen ist unsere Entscheidung.

### Security

Beim Login Versuch sollen die Daten des Users authentifiziert und autorisiert werden mithilfe von Tests. Jeder REST Endpoint ist nur durch die entsprechende Autorität erreichbar.

## Nicht funktionale Anforderungen

### Implementation

Die Daten werden in einer PostgresSQL Datenbank abgespeichert und das OR Mapping wird mit JPA gemacht.

### Testing

Mindestens ein Endpoint wird ausführlich getestet. Dazu gehört die implementierte Funktionalität wird mit Component Tests in Postman getestet. Die Endpoints werden mit mehreren Usern welche unterschiedlichen Rollen haben getestet. Es wird mindestens ein Erfolgsfall und ein Error Fall gestestet. Dazu werden Use Cases nach UML Standard geschrieben.

### Multiuserfähigkeit

### Alle ACID Prinzipien wurden eingehalten.

### Dokumentation

Es wurde ausführlich und verständlich dokumentiert. Zusätzlich ist ein Readme File im Github Repository welches die wichtigsten Informationen zum Projekt liefert, sowie eine Setup Anleitung. Alle Endpoints, welche implementiert wurden, sind beschrieben worden. Ein Domänenmodell, Class diagram, Sequence diagram wurden erstellt und Usecases wurde für die getesteten Endpoints geschrieben.

# Arbeitsaufteilung

Um unsere Arbeit aufzuteilen, haben wir in Github die Issues Funktion genutzt. Mit dieser konnten wir immer sehen, was noch gemacht werden muss, wer an was arbeitet oder gearbeitet hat und was schon gemacht wurde oder noch reviewed werden muss.

# Tag 1 – 05.11.2021

# Tag 2 – 10-11.2021

# Tag 3 – 11.11.2021