ÜK 223 - Noser Young

Dokumentation Gruppe 2

«OurSpace» Fullstack Blogging Projekt

Autoren:
Jan Ludwig
Leon Probst
Lynn Ruchti
Thomas Stern

Abgabedatum: 16.01.2025

Inhaltsverzeichnis

Einleitung:

Im Rahmen des üK 223 wurden wir mit der Erstellung der Social-Media-Plattform “OurSpace” beauftragt. Das Kernziel dieses Projektes ist die Konzeption und Implementierung einer Fullstack-Komponente, die es Benutzern ermöglicht, Blog-Beiträge zu erstellen, zu verwalten und zu lesen.

Diese Dokumentation beschreibt die technische Umsetzung des Moduls «Blog Posts» (Aufgabe 5.2). Die Applikation basiert auf einer Spring Boot Backend-Architektur mit PostgreSQL-Datenbankanbindung und einem React Frontend. Ein besonderer Fokus liegt dabei auf der Multi-User-Fähigkeit, der Einhaltung strenger Sicherheitsstandards durch rollenbasierte Zugriffsrechte (JWT) sowie einer umfassenden Qualitätssicherung durch automatisierte Tests.

Die folgenden Kapitel erläutern die Architektur, die definierten Use Cases sowie die angewandte Teststrategie, um die Anforderungen an Funktionalität und Sicherheit zu erfüllen.

Testing Strategie:

Testing Tool

Layer	Tool	Zweck
API Integration	Postman	REST API Endpoints, Workflows
Frontend Unit	Cypress	User Flows, Role-Based Testing

Testing Endpoints

HTTP Method	Endpoint	Beschreibung	Expected Result
GET	/blogpost	Alle Blog Posts abrufen	200 + All BlogPost
GET	/blogpost/{id}	Einzelnen Blog Post abrufen	200 + BlogPost
POST	/blogpost	Neuen Blog Post erstellen	201 Created + BlogPost
PUT	/blogpost/{id}	Blog Post aktualisieren	200 + BlogPost
DELETE	/blogpost/{id}	Blog Post löschen	204 No Content

Test scenarios

GET /blogpost (UC4 - Public Read)

Scenario Type	Condition / Input	Expected Result & Status	Tools
Success	Public call, no specific params	200 OK + List (Default size 5)	Postman
Success	Default sort behavior	200 OK + Sorted by Date	Postman
Edge Case	Database is empty	200 OK + Empty Content []	Postman

POST /blogpost (UC1 - Create)

Scenario Type	Condition / Input	Expected Result & Status	Tools
Success	User with BLOGPOST_CREATE, Valid input	201 Created + Saved Object	Cypress
Validation	Title length < 5 characters	400 Bad Request / Error Msg	Cypress
Validation	Content length < 20 characters	400 Bad Request / Error Msg	Cypress
Validation	Category is missing/null	400 Bad Request / Error Msg	Cypress
Auth	Request without JWT Token	401 Unauthorized	Cypress
Permission	User without BLOGPOST_CREATE right	403 Forbidden	Cypress

PUT /blogpost/{id} (UC2 / UC5 - Edit)

Scenario Type	Condition / Input	Expected Result & Status	Tools
Success	Author edits own post	200 OK + Updated Object	Postman
Success	Admin edits any post (..._ANY)	200 OK + Updated Object	Postman
Auth	User A tries to edit User B's post	403 Forbidden	Postman
Auth	Request without JWT Token	401 Unauthorized	Postman
Validation	Invalid input (e.g. Title < 5)	400 Bad Request	Postman
Edge Case	ID does not exist	404 Not Found	Postman

DELETE /blogpost/{id} (UC3 / UC5 - Delete)

Scenario Type	Condition / Input	Expected Result & Status	Tools
Success	Author deletes own post	204 No Content	Postman
Success	Admin deletes any post (..._ANY)	204 No Content	Postman
Auth	User A tries to delete User B's post	403 Forbidden	Postman
Edge Case	ID does not exist	404 Not Found	Postman

Use Case Definition:

UC1: User erstellt neuen Blogpost

Actor	User/System
Beschreibung	Ein User möchte einen neuen Blogpost erstellen.
Precondition	- User ist eingeloggt - User hat die Erlaubnis neue Blogposts zu erstellen
Postcondition	- Blogpost ist validiert - Autor ist zugewiesen - Blogpost ist gespeichert/erstellt - Blogpost ist veröffentlicht und lesbar für andere
Normaler Durchlauf	1. User navigiert zu “neuen Blogpost erstellen” 2. System zeigt Blogpost-Erstellungs Formular an 3. System teilt Autor zu (User = Autor) 4. User gibt Content ein (Titel, Text, Kategorie) 5. System validiert Blogpost 6. User submitted Blogpost 7. System speichert Blogpost 8. System gibt Erfolgsmeldung aus 9. System navigiert zur Homepage
Alternative Durchlauf	4. User bricht Erstellungsprozess ab: System führt Rollback aus und navigiert zur Homepage 5. Validation fehlgeschlagen: System gibt Fehlermeldung aus und navigiert zu 4.
Exception	1. User ist nicht berechtigt: System navigiert zur Homepage 5/7. Systemfehler beim Validieren/Speichern: System gibt Fehlermeldung aus und navigiert zu 4.

UC2: User bearbeitet eigenen Blogpost

Actor	User/System
Beschreibung	User bearbeitet einen eigenen, bereits erstellten Blogpost.
Precondition	- User ist eingeloggt - User ist der Autor des Blogpost - Blogpost ist bereits erstellt
Postcondition	- Blogpost ist aktualisiert/gespeichert - Blogpost ist validiert
Normaler Durchlauf	1. User navigiert zu “Blogpost bearbeiten” 2. System zeigt Blogpost-Bearbeitungs Formular mit allen Infos des Blogposts an 3. User bearbeitet Content (Titel, Text, Kategorie) 4. System validiert Blogpost 5. User speichert bearbeiteten Blogpost 6. System speichert Blogpost 7. System gibt Erfolgsmeldung aus 8. System navigiert zur Homepage
Alternative Durchlauf	3. User bricht Bearbeitung ab: System führt Rollback aus und navigiert zur Homepage 4. Validation fehlgeschlagen: System gibt Fehlermeldung aus und navigiert zu 3.
Exception	1. User ist nicht berechtigt: System navigiert zur Homepage 4/6. Systemfehler beim Validieren/Speichern: System gibt Fehlermeldung aus und navigiert zu 3.

UC3: User lösch eigenen Blogpost

Actor	User/System
Beschreibung	User löscht einen eigenen, bereits erstellten Blogpost.
Precondition	- User ist eingeloggt - User ist der Autor des Blogpost - Blogpost ist bereits erstellt
Postcondition	- Blogpost ist gelöscht - Löschung ist validiert
Normaler Durchlauf	1. User navigiert zu "Blogpost löschen” 2. User bestätigt Löschung mit einem Button 3. System validiert Löschung 4. System löscht Blogpost 5. System gibt Erfolgsmeldung aus 6. System navigiert zur Homepage

Alternative Durchlauf	2. User bricht Löschung ab: System navigiert zur Homepage
Exception	1. User ist nicht berechtigt: System navigiert zur Homepage 3/4. Systemfehler beim Validieren/Löschen: System gibt Fehlermeldung aus und navigiert zu 2.

UC4: Anonyme User lesen (anzeigen) Blogposts mit Pagination und Sortierung

Actor	Anonymer User/System
Beschreibung	Ein anonymer User schaut sich veröffentlichte Blogposts mit Pagination und Sortierung an.
Precondition	- User ist ein anonym, also nicht eingeloggt - System ist erreichbar - Anonymer User hat Homepage geöffnet
Postcondition	- Blogposts werden angezeigt - Ausgewählte Sortierung wird vorübergehend gemerkt
Normaler Durchlauf	1. Anonymer User navigiert zu “Blogpost Liste” 2. System listet 5 Blogposts auf (standard Sortierung nach Datum von neu zu alt) 3. Anonymer User schaut sich die aufgelisteten Blogposts an
Alternative Durchlauf	3. Anonymer User navigiert zur nächsten Page: System lädt die nächsten 5 Blogposts (Sortierung aktiv) 2. 3. Anonymer User verändert Sortierung: System lädt die gesamte Liste mit ausgewählter Sortierung neu und navigiert zur ersten Seite 2. 3. Anonymer User verlässt “Blogpost Liste”: System/Browser lädt nächste Seite
Exception	2. Es existieren keine Blogpost: System gibt Fehlermeldung aus und navigiert zur Homepage 2. Systemfehler beim Laden der Liste: System gibt Fehlermeldung aus und lädt die Liste nochmals. Nach erneutem auftreten navigiert zur Homepage

UC5.1: Admin bearbeitet beliebige Blogposts

Actor	Admin/System
Beschreibung	Admin bearbeitet einen beliebigen Blogpost.
Precondition	- User ist eingeloggt als Admin - Blogpost ist bereits erstellt

Postcondition	- Blogpost ist aktualisiert/gespeichert - Blogpost ist validiert
Normaler Durchlauf	1. Admin navigiert zu “Blogpost bearbeiten” 2. System zeigt Blogpost-Bearbeitungs Formular mit allen Infos des Blogposts an 3. Admin bearbeitet Content (Titel, Text, Kategorie) 4. Admin speichert bearbeiteten Blogpost 5. System validiert Blogpost 6. System speichert Blogpost 7. System gibt Erfolgsmeldung aus 8. System navigiert zum Admin Portal
Alternativer Durchlauf	3. Admin bricht Bearbeitung ab: System führt Rollback aus und navigiert zum Admin Portal 5. Validation fehlgeschlagen: System gibt Fehlermeldung aus und navigiert zu 3.
Exception	5/6. Systemfehler beim Validieren/Speichern: System gibt Fehlermeldung aus und navigiert zu 3.

UC5.2: Admin löscht beliebige Blogposts

Actor	Admin/System
Beschreibung	Admin löscht einen beliebigen Blogpost.
Precondition	- User ist eingeloggt als Admin - Blogpost ist bereits erstellt
Postcondition	- Blogpost ist gelöscht - Löschung ist validiert
Normaler Durchlauf	1. Admin navigiert zu "Blogpost löschen” 2. Admin bestätigt Löschung mit einem Button 3. System validiert Löschung 4. System löscht Blogpost 5. System gibt Erfolgsmeldung aus 6. System navigiert zum Admin Portal
Alternativer Durchlauf	2. Admin bricht Löschung ab: System navigiert zum Admin Portal
Exception	3/4. Systemfehler beim Validieren/Löschen: System gibt Fehlermeldung aus und navigiert zu 2.

Zusammenfassung:

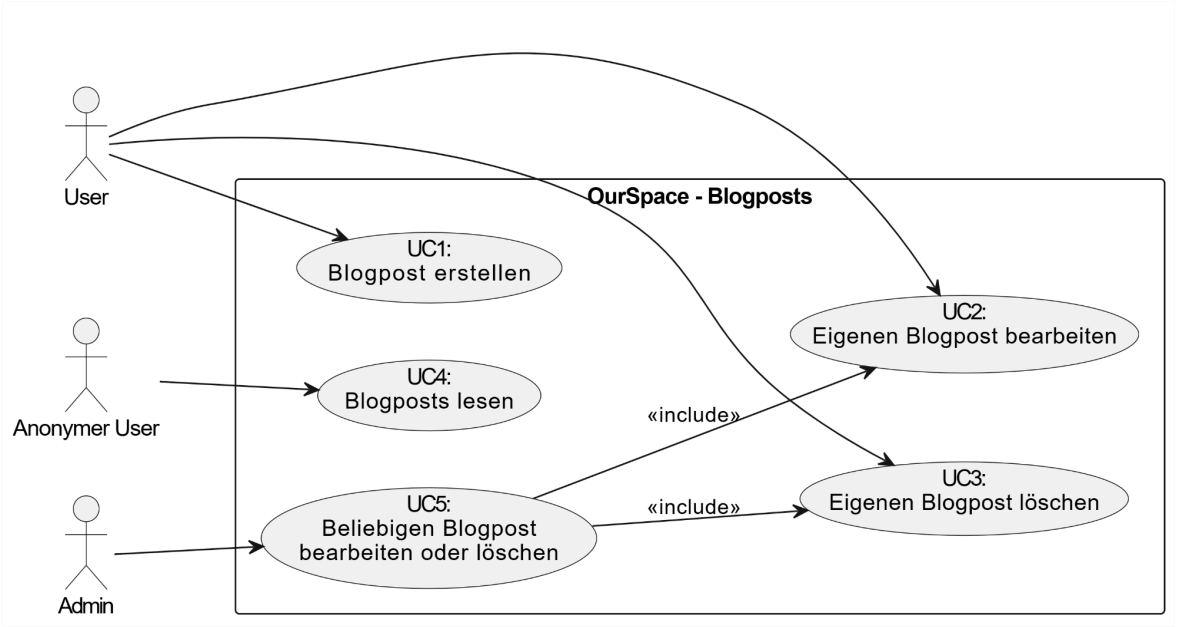
Im Rahmen des üK 223 entwickelten wir eine Fullstack Applikation der Social-Media-Plattform OurSpace. Der Fokus lag auf der Implementierung der Blogpost-Funktionalität (5.2) mit React, Spring Boot und PostgreSQL. Ziel war eine sichere und multiuserfähige Applikation mit klarer Rollen- und Rechteverwaltung.

Benutzer können eigene Blogposts erstellen, bearbeiten und löschen, während anonyme Nutzer öffentliche Blogposts mit Pagination und Sortierung lesen können. Administratoren besitzen erweiterte Rechte für das Bearbeiten und Löschen beliebiger Blogposts. Die Authentifizierung erfolgt über JWT, die Autorisierung rollen und rechte basiert.

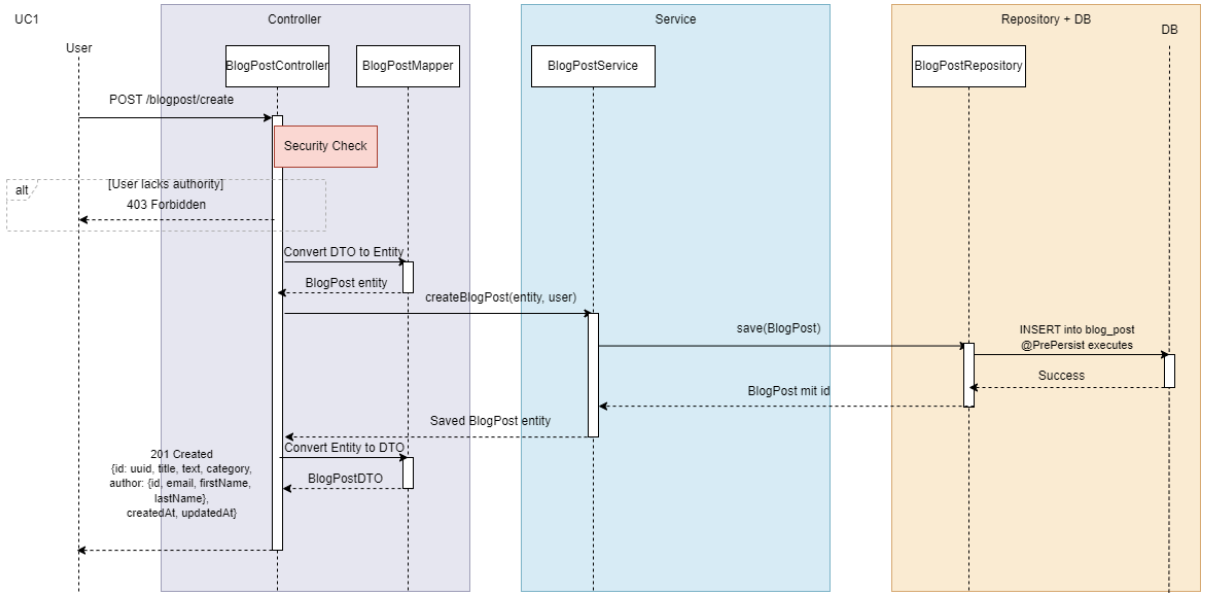
Die Qualitätssicherung erfolgt gemäss der Testing-Pyramide mit Postman und Cypress, wobei sowohl positive als auch negative Szenarien getestet wurden. Die Dokumentation umfasst Use-Case-Beschreibungen, Diagramme (Unten Angehängt) sowie ein Progress Journal. Damit erfüllt das Projekt alle Anforderungen des üK 223.

Diagramme und Anhang:

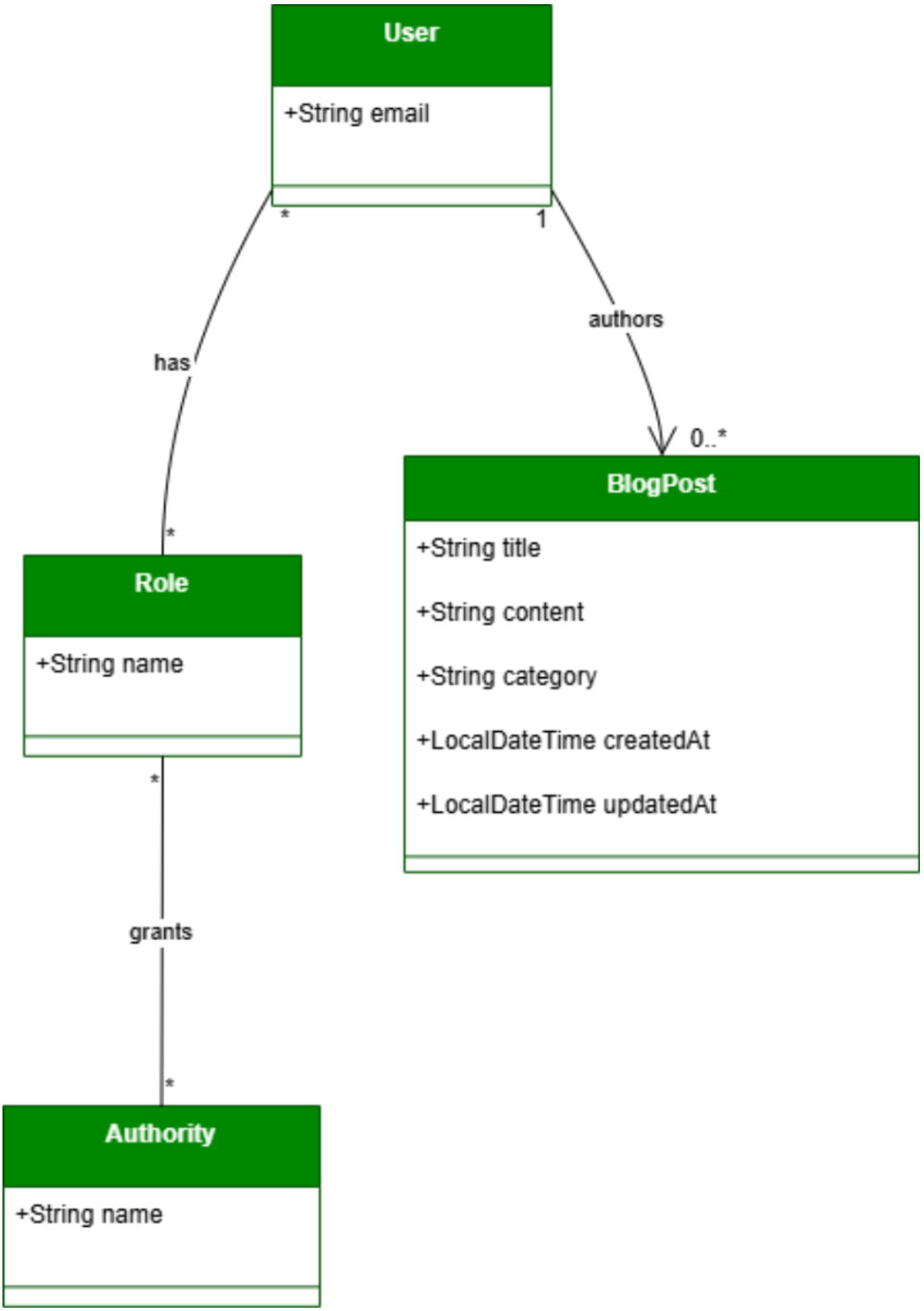
Use Case Diagram:



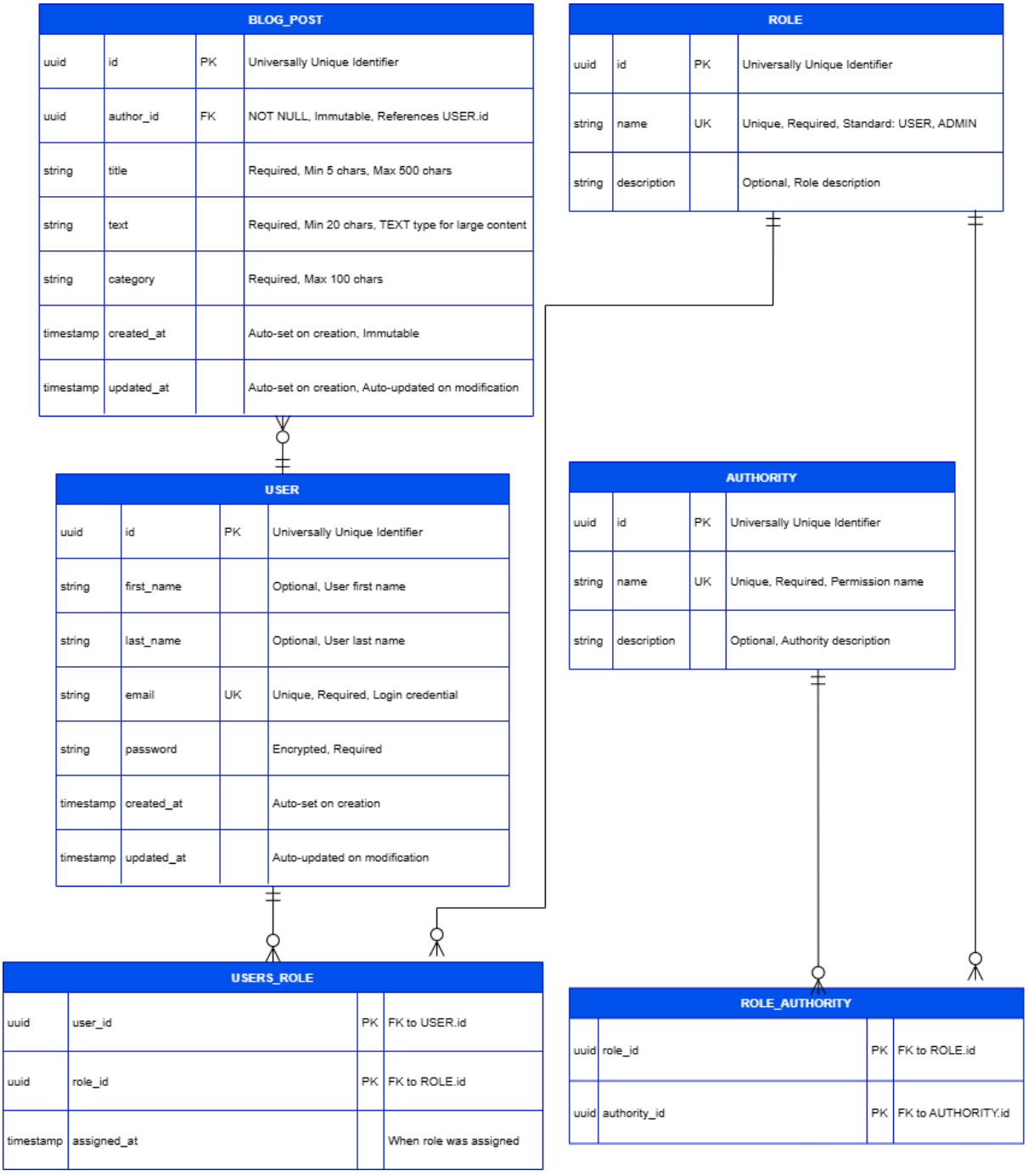
Sequence Diagram:



Domain Model:



ERD:


BLOG_POST
uuid id PK Universally Unique Identifier
uuid author_id FK NOT NULL, Immutable, References USER.id
string title Required, Min 5 chars, Max 500 chars
string text Required, Min 20 chars, TEXT type for large content
string category Required, Max 100 chars
timestamp created_at Auto-set on creation, Immutable
timestamp updated_at Auto-set on creation, Auto-updated on modification
ROLE
uuid id PK Universally Unique Identifier
string name UK Unique, Required, Standard: USER, ADMIN
string description Optional, Role description
USER
uuid id PK Universally Unique Identifier
string first_name Optional, User first name
string last_name Optional, User last name
string email UK Unique, Required, Login credential
string password Encrypted, Required
timestamp created_at Auto-set on creation
timestamp updated_at Auto-updated on modification
AUTHORITY
uuid id PK Universally Unique Identifier
string name UK Unique, Required, Permission name
string description Optional, Authority description
USERS_ROLE
uuid user_id PK FK to USER.id
uuid role_id PK FK to ROLE.id
timestamp assigned_at When role was assigned
ROLE_AUTHORITY
uuid role_id PK FK to ROLE.id
uuid authority_id PK FK to AUTHORITY.id

Lo-Fi Website Mockup