Software Engineering

Documentation of “Blog” project

Student:Kuralay Albekova, Drobotenco Denis, Arkabayev Yelnur, Aldiyar Alibayev

Student ID:w68610, w67116, w67111,w68613

Group:3LID-A-GDD

Teacher: dr inz. Maksymilian Knap

System Description: Social Media Platform for Interaction

Introduction

Our social media platform aims to provide users with a seamless experience for creating, sharing, and interacting with content. With a focus on user engagement, our system offers a range of functionalities to facilitate communication, expression, and community building. Below is a detailed analysis of the system's key functionalities:

User Registration

The user registration process is streamlined, requiring only an email address and password for account creation.

Authentication

The platform employs robust encryption protocols to securely store and authenticate user credentials during login.

Search Functionality

Users are empowered with a versatile search tool, allowing them to explore content based on keywords or hashtags.
Search results are intelligently curated, offering filtering options such as relevance, popularity, and recency.

Post Creation

Registered users can unleash their creativity by crafting engaging posts, expressing their thoughts and ideas in a text-based format.

Draft Saving

Acknowledging the importance of spontaneity and refinement, users are provided with the option to save drafts before publishing their posts.
This feature empowers users to iterate on their creations, ensuring the delivery of polished and compelling content.

Post Hiding

Recognizing the need for users to have control over their content visibility, the post hiding functionality allows users to remove their posts from public view.
This feature empowers users to manage their content according to their preferences, ensuring privacy and control over their online presence.

Post Editing

Empowering users with the ability to refine and update their posts, the post editing feature allows users to make modifications to their published content.
This functionality ensures that users can maintain the accuracy and relevance of their posts over time, enhancing the overall quality of content on the platform.

Post Interaction

A vibrant ecosystem of engagement unfolds as users express their sentiments towards posts through a diverse array of emojis.
The platform fosters dialogue and discourse through its robust commenting system, allowing users to contribute insights and perspectives.

Comment Presentation

Comments are elegantly presented beneath each post, arranged chronologically to ensure fluidity and coherence within the conversation.
This intuitive layout enhances readability and encourages active participation within the community.

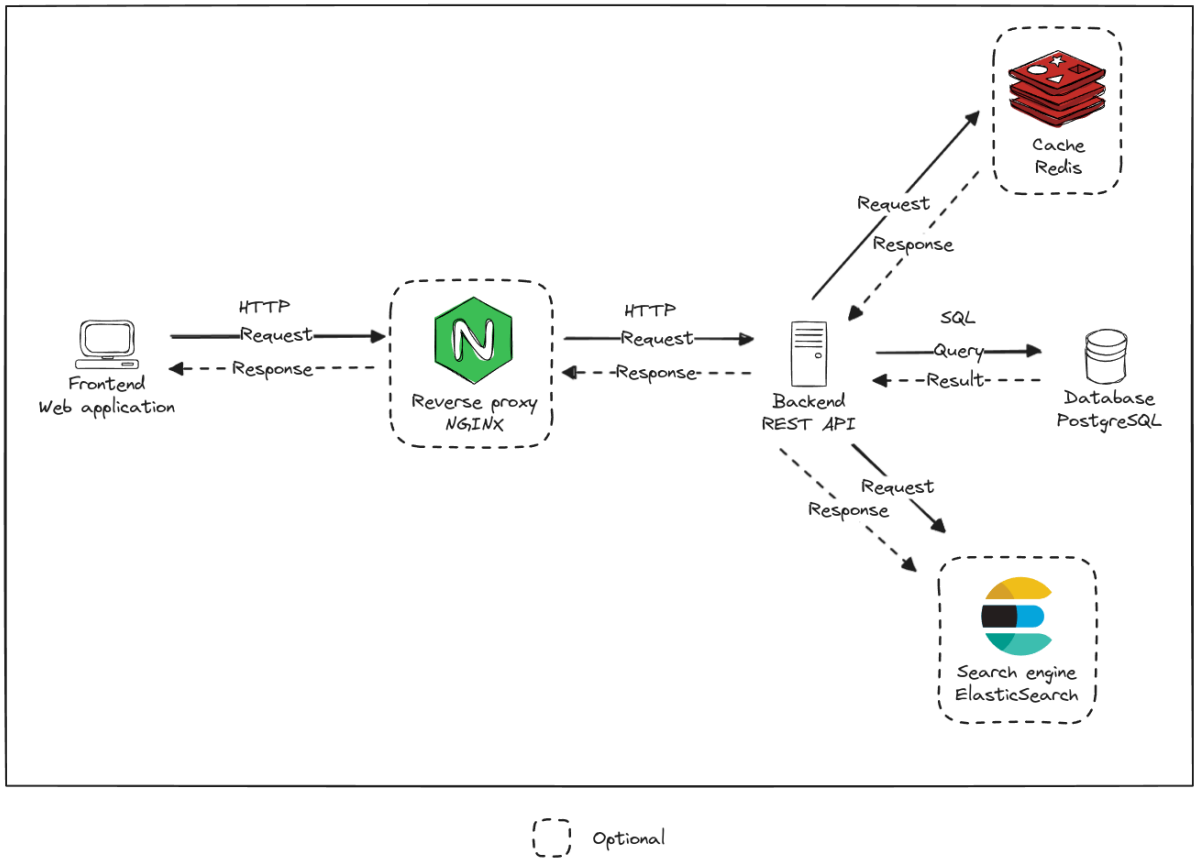
Sharing Functionality

Collaboration and dissemination of content are facilitated through the seamless sharing functionality embedded within the platform.
Users have the freedom to amplify the reach of posts by sharing them with fellow users, fostering a culture of interconnectedness and community engagement.

Architecture, user stories and diagrams

Non-functional requirements

Architecture



Technology Stack

Backend services will be developed using Node.js, providing a lightweight and efficient runtime environment for handling HTTP requests.

Frontend development will utilize Angular for building a responsive and interactive user interface, enhancing the user experience.

PostgreSQL will serve as the primary relational database management system (RDBMS) for storing structured data, ensuring data integrity and efficient querying capabilities.

Redis could be utilized as a caching layer to enhance performance and reduce latency, particularly for frequently accessed data if there is a need for that.

Amazon S3 or a similar object storage service could be used for storing multimedia content such as images and videos uploaded by users if there are features requiring it.

WebSocket protocol could be implemented for real-time communication between the client and server, enabling instant updates and notifications if there are features requiring it.

NGINX could be utilized as a reverse proxy server to terminate SSL, improve performance, reliability, and security if the project is hosted somewhere some day.

User Stories

As a new user, I want to be able to register on the platform using my email and password so that I can create an account and start using the system.

- Database
    - Users table
- Backend
    - Password encryption
    - POST /auth/register endpoint. Create user, assert uniqueness.
- Frontend
    - Registration page

As a registered user, I want to be able to create text-based posts.

- Database
    - Posts table
- Search engine
    - Document indexing
- Backend
    - POST /posts endpoint. Create post, index document.
- Frontend
    - Post creation page

As a registered user, I want to be able to list recent posts.

- Backend
    - GET /posts endpoint. List and sort results.
- Frontend
    - Post listing page.

As a registered user, I want to be able to search for posts based on keywords to discover relevant content.

- Search engine
    - Searching
- Backend
    - GET /posts endpoint. Integrate search.
- Frontend
    - Post listing page, integrate search.

As a registered user, I want to be able to view a single post.

- Backend
    - GET /posts/:id endpoint.
- Frontend
    - View post page.

As a registered user, I want to be able to react to posts using emojis to engage with other users.

- Database
    - PostReaction table
- Backend
    - POST /posts/:id/react endpoint. Persist the user reaction to the post
- Frontend
    - View post page, view and create reactions

As a registered user, I want to be able to leave comments to posts to engage with other users.

- Database
    - PostComment table
- Backend
    - POST /posts/:id/comment endpoint. Create user comment to the post
- Frontend
    - View post page, view and create comments

As a user, I want to be able to hide my posts from public view, so that I can control the visibility of my content and manage my online presence effectively.

- Database
    - Post table update
- Backend
    - POST /posts/:id/hide endpoint. Hide the post from the public listing.
- Frontend
    - View post page, “hide” button with confirmation dialog

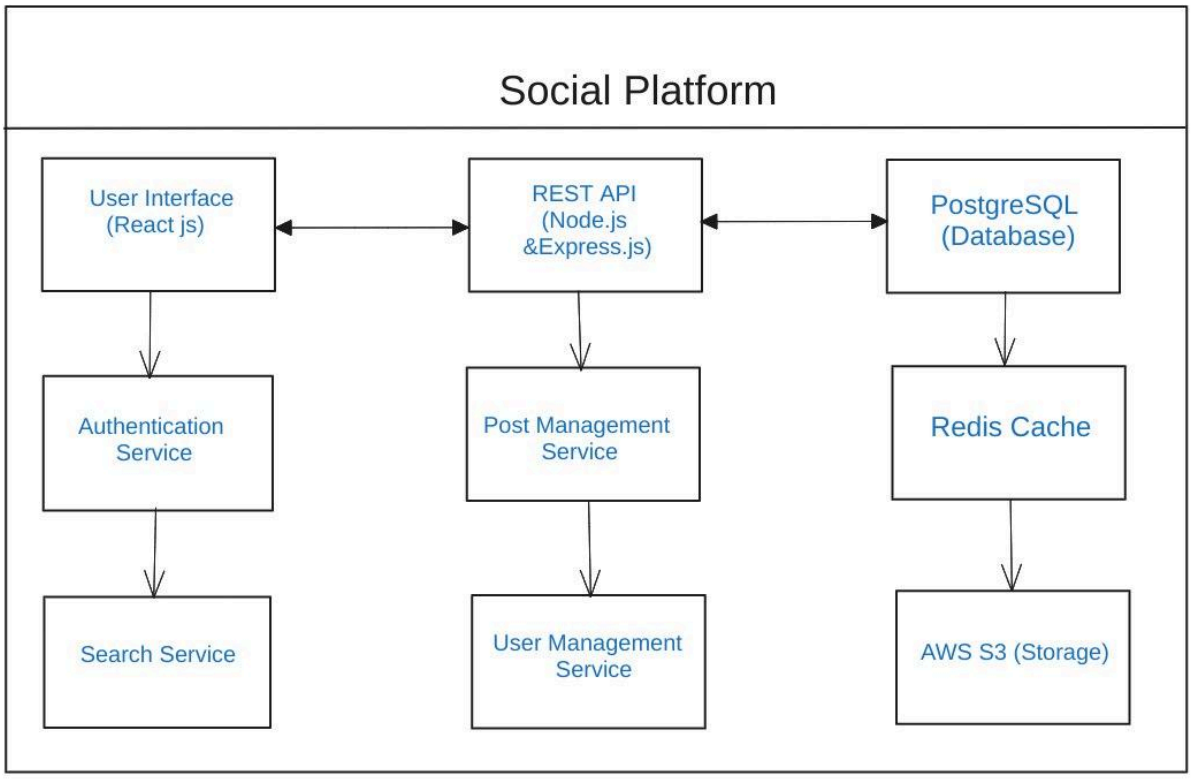
As a user, I want to be able to edit my published posts, allowing me to correct errors or update information as needed to maintain the accuracy and relevance of my content.

- Backend
    - PATCH /posts/:id endpoint. Update the post.
- Frontend
    - View post page, “edit” button with editing dialog

Component Diagram for the Analyzed System

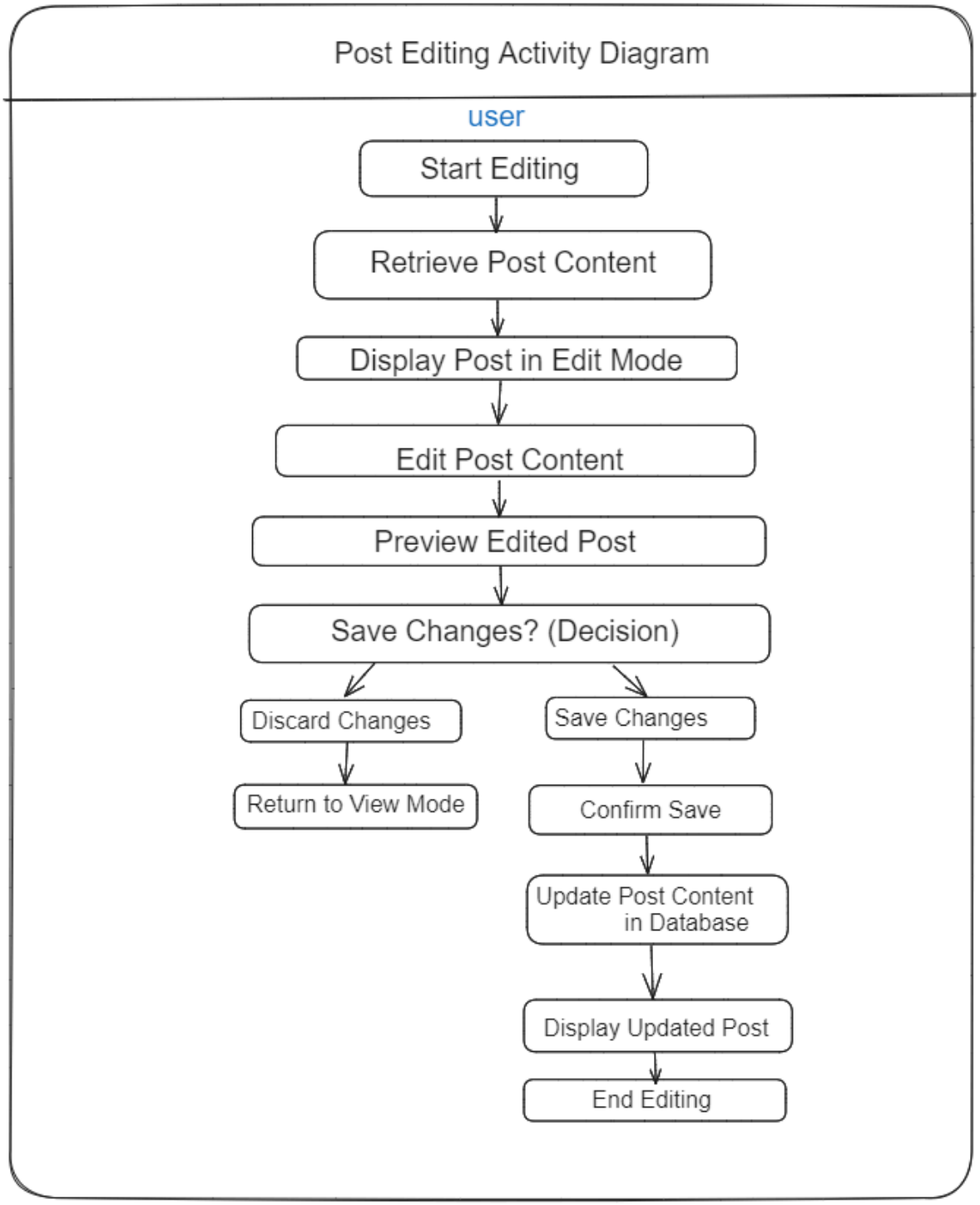
The component diagram for our social platform will include the following components:

- User Interface (React.js)
- REST API (Node.js & Express.js)
- PostgreSQL (Database)
- Redis (Cache)
- Authentication Service
- Post Management Service
- User Management Service
- AWS S3 (Storage)

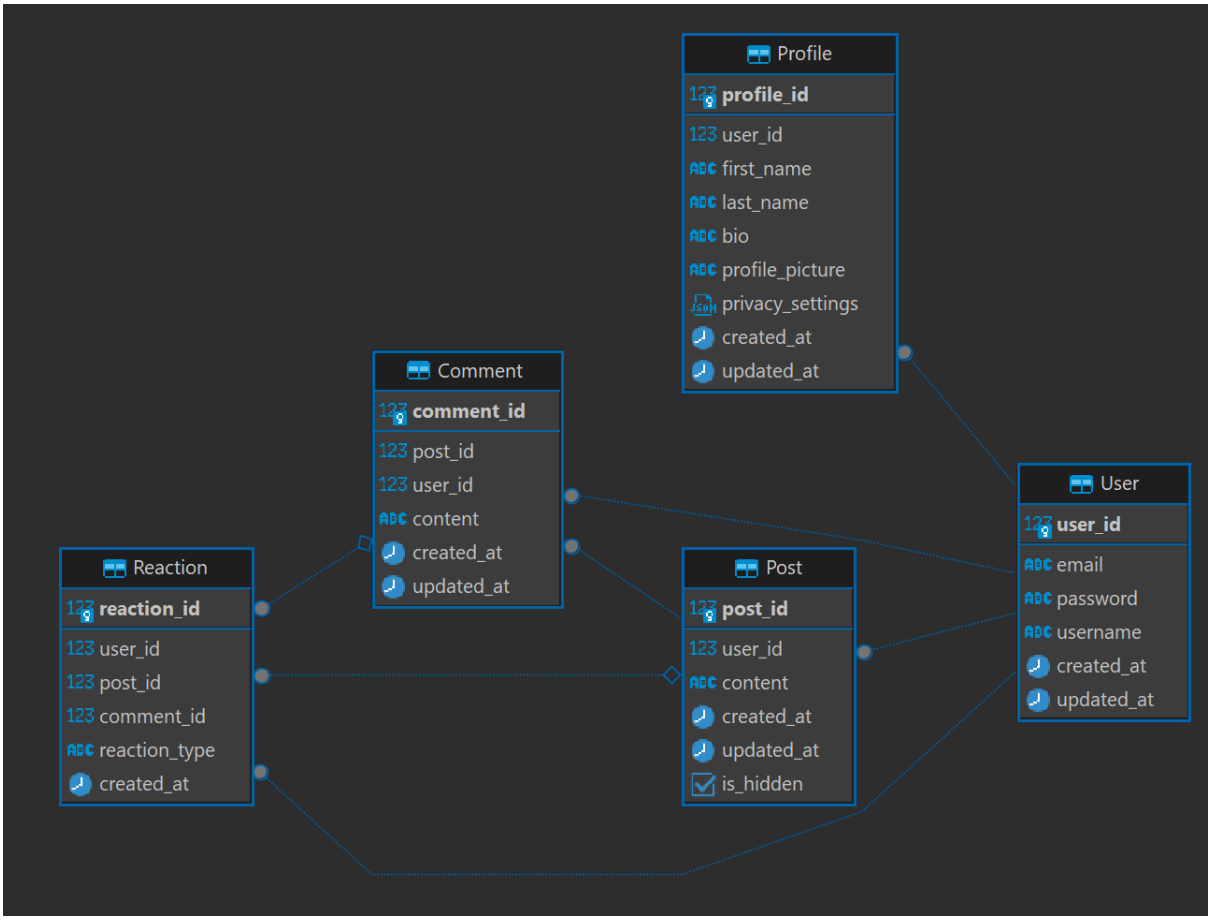


Activity Diagram for the "Post Editing" Use Case

An activity diagram for the "Post Editing" use case will outline the sequence of actions taken by a user to edit a post.



Entity Relationship Diagram



Links

Github:
https://github.com/kuralayalbekova/social-platform-project