DBMS Assignment-2 Report

Chittepu Rutheesh Reddy (CS21BTECH11014)
G Harsha Vardhan Reddy (CS21BTECH11017)
Kotikalapudi Karthik (CS21BTECH11030)
Sadineni Abhinay (CS21BTECH11055)

March 5, 2023

1 Project Description

We developed a Community Question Answer (CQA) website that will allow users to interact with databases in a user-friendly manner.

2 Summary of how we built the application

The database cqadb contains 5 tables
1) **users**- stores user details like username, password, display_name
2) **tags**- stores tags and their ids
3) **posts**- stores post details like id, owner, tags, body, title etc
4) **answers**- stores answer details like id, owner, post id, body etc
5) **post**_upvotes- stores whether an user upvoted/downvoted the particular post
And 5 indices i.e. userid,username,postid,answerid,questionid These table are interconnected with each other and we referenced from table to table making good model.

We implemented authentication by creating sessionIDs using jsonwebtoken library, which will be sent after a user logs in, this will be stored in the cookie. All routes requiring authentication will verify this id before sending the response. Next, we began displaying posts based on searching using single tag,multiple tags and username, which we acquire from the back-end server through GET/POST requests. The back-end server validates the request and fetches the required data using the pg library, which connects to a PostgreSQL server through a client connection. After obtaining the required data, front-end components display the data in a user-friendly manner.

We now started creating forms for posting questions and answers using Markdown-Editor. These forms are submitted using POST requests, where the back-end verifies whether the user is logged in or not. If the user is logged in, the server connects to the database and inserts the data into the corresponding tables. We can later see the posted entities displayed by the front-end. We then added edit forms, where we first fetch the saved data from the database and then post it again. The back-end performs an update query based on the post ID instead of inserting it into the table.

3 Overall System Architecture

1. **Frontend(Client): ReactJS with TailwindCSS**
   ReactJS handles the user interface and user experience of the web application. TailwindCSS was used for styling the application. Axios is used to make HTTP requests to the backend.

2. **Backend(Server): ExpressJS with PostgreSQL**
   ExpressJS handles the server-side logic of the web application. PostgreSQL is used as the database for storing and retrieving data.

4 Programming languages used

We used Javascript for all components of frontend, backend. We used Postgresql for database.

5 Tools Used

1. **React framework:**
   React (also known as React.js or ReactJS) is an open-source front-end JavaScript library for building user interfaces based on components.

2. **Express/Node:**
   Express.js or Express, is a back end web application framework for building RESTful APIs with Node.js. It is designed for building web applications and APIs.

3. **Tailwind CSS:**
   Tailwind CSS is an open source CSS framework. The main feature of this library is that, unlike other CSS frameworks like Bootstrap, it does not provide a series of predefined classes for elements such as buttons or tables. Instead, it creates a list of "utility" CSS classes that can be used to style each element by mixing and matching.

4. **Pg library:**
   node-postgres is a collection of node.js modules for interfacing with your

PostgreSQL database. It has support for callbacks, promises, async/await, connection pooling, prepared statements, cursors, streaming results, C/C++ bindings, rich type parsing, etc.It also tries to provide guides for more advanced edge-case topics allowing you to tap into the full power of PostgreSQL from node.js.

5. **Postgres-server:**
   PostgreSQL/Postgres, is an open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance.

6 Work Distribution

1. **Abhinay:** frontend modification,middleware,some backend routes
2. **Harsha:** Database queries, database schemas
3. **Karthik:** frontend page to page integration, CSS styling
4. **Rutheesh:** frontend-backend integration,frontend templates, bug fixing, authentication, backend templates