**Boulevard Blog**

**DESIGN DOCUMENT**

**INTRODUCTION**

**Purpose**

The primary goal of Boulevard blog is to provide a platform for individuals or organizations to publish and manage content in the form of blog posts. Users can create, edit, and delete posts, share their thoughts, insights, and expertise on various topics, and engage with their audience through likes and comments.

**Scope**

The scope of the project includes the development of a full-featured blog application with user authentication, CRUD operations for blog posts and comments by users, and a responsive user interface.

**ARCHITECTURE AND OVERVIEW**

**Technology Stack**

Boulevard Blog is built upon the MERN Stack.

* React.js: Frontend
* Node.js with Express.js: Backend
* MongoDB: Database
* Tailwind CSS: Styling
* JWT: Authentication
* Redux Toolkit: State management

**Overview**

* Frontend : Developed with React.js, this blog app provides interfaces for creating blog posts, user authentication, and adding comments.
* Backend : Powered by Express.js and Node.js, the backend offers RESTful APIs for user authentication, blog posts handling, and database interactions.
* Database : MongoDB serves as the database to store user data, blog content, comments and related information.

**DATABASE DESIGN**

MongoDB database consists of the following collections:

* Users Collection: Stores user information, including name, email, password hash, and timestamps.
* Blogs Collection: Contains the content and details about the blog posts of every user.
* Comments Collection: Stores the comments added by different users with details and timestamps.

**SCHEMA MODELS**

The MongoDB database schema for user accounts, blog posts, and comments.



**FUNCTIONAL REQUIREMENTS**

**User Stories**

* Registration/Login: User can sign up or sign in.
* View blogs: User can see a list of blogs.
* Create blogs: User can create new blog posts.
* Edit/Delete post: User can modify or remove their posts.
* Add/Edit/Delete comment: User can add /edit or delete their comments.
* Logout: User can end their session.

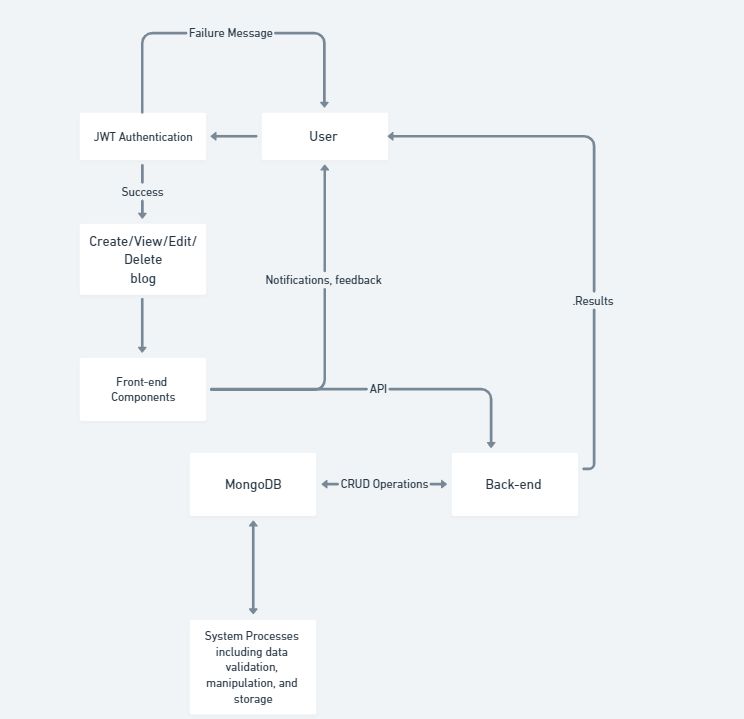
**API DOCUMENTATION**

Authentication API’s :

* /api/auth/signup : Register a new user
* /api/auth/signin : Authenticate user credentials and generate a JWT token
* /api/auth/user : Retrieve current user details

More to be added.

**DATA FLOW DIAGRAM**

****

**NON-FUNCTIONAL REQUIREMENTS**

**Performance**

The application loads blog posts and comments quickly.

Server response time should be optimized.

**Security**

User passwords are securely hashed and stored.

Authentication is implemented using JWT and bcryptjs for hashing.

**TESTING**

* Unit tests for frontend components using Jest.
* Test API endpoints to verify correct request handling and response generation.
* Used Insomnia to send HTTP requests and validate responses.
* Test edge cases, error handling, and authentication/authorization mechanisms.

**DEPLOYMENT**

Boulevard Blog can be deployed on cloud platforms like Heroku, AWS, or DigitalOcean. Frontend hosting services such as Netlify or Vercel, coupled with MongoDB Atlas for database hosting, offer a robust deployment solution.

**CONCLUSION**

The outlined specifications aim to ensure the development of a scalable and secure application that meets user expectations. Boulevard Blog aims to provide enthusiasts with a seamless and enjoyable experience for sharing thoughts online. By following the outlined design and implementation strategies, the application will deliver on its promise of simplicity, reliability, and efficiency.