**Project Proposal: Chatroom Application for Database Systems**

**Project Title: WeChatBox - A Real-Time Chatroom Application**

**Team Members:**

* **Waleed Ahmed**

**Introduction:**

The **WeChatBox** project aims to develop a real-time chatroom application that uses database systems for storing, managing, and retrieving users for authentication (login) and message data. This application is designed for scalability, offering features like user authentication and message persistence while ensuring data consistency and security.

**Objectives:**

1. **Core Objectives:**
   * Build a real-time chatroom application with CRUD operations for messages, users, and chatrooms.
   * Implement efficient database design to handle concurrent users and large volumes of data.
   * Ensure data integrity and security with proper schema design and validation.
2. **Learning Objectives:**
   * Apply database normalization techniques to improve performance.
   * Utilize advanced SQL concepts such as indexing, stored procedures, and transactions.

**Scope:**

The application will include the following features:

* **User Authentication:** Secure login and signup system with hashed passwords.
* **Real-Time Messaging:** Users can exchange messages within chatrooms in real-time.
* **Chatrooms:** Creation and management of public and private chatrooms.
* **Message History:** Store messages for later retrieval.
* **Search Functionality:** Allow users to search chatrooms or message history.
* **Moderation Tools:** Admins can manage chatroom members and delete inappropriate messages.

**Technical Requirements:**

1. **Frontend:**
   * Framework: React.js
   * Styling: Tailwind CSS
   * Real-Time Updates: WebSockets (Socket.IO)
2. **Backend:**
   * Framework: Node.js with Express.js
   * APIs: RESTful endpoints for non-real-time operations.
3. **Database:**
   * **Primary DB:** MongoDB for Database operations
4. **Hosting:**
   * Backend: Vercel or AWS
   * Database: MongoDB for Cloud server hosting
5. **Additional Tools:**
   * Bcrypt for password hashing.
   * Postman for API testing.

**Database Schema Overview:**

1. **Users Table/Collection**
   * user\_id (Primary Key)
   * username
   * email
   * password\_hash
   * created\_at
2. **Chatrooms Table/Collection**
   * chatroom\_id (Primary Key)
   * chatroom\_name
   * is\_private
   * created\_by
   * created\_at
3. **Messages Table/Collection**
   * message\_id (Primary Key)
   * chatroom\_id (Foreign Key)
   * user\_id (Foreign Key)
   * message\_text
   * timestamp
4. **Roles and Permissions (Optional):**
   * Assign roles (e.g., Admin, Moderator, User) to manage access control.