Design Document

TopicHive

Project Part II CSCI 6004

Introduction to Web Development

Submitted to –

Professor Rahul Simha

Prepared by:

Shyama Arunachalam

Link To the Demo Video :
https://drive.google.com/file/d/1l7CKYOkZspqSavEqg83_rr89z3AlL1tk/view?usp=sharing

1. Introduction

a. Purpose

The purpose of this document is to provide a detailed design and implementation overview of TopicHive, a platform for chatroom-based communication and administration. This document serves as a blueprint for understanding the system's architecture, components, workflows, and database interactions. It outlines the functionality and features of TopicHive, including user management, chatroom administration, and role-based permissions.

b. Scope

This document focuses on key aspects of the system:

1. User authentication and role management (admin and regular users).
2. Chatroom creation, deletion, and participation workflows.
3. Integration of features such as banning users, upgrading users to admin, and automatically managing user-to-chatroom mappings.
4. Database design, including tables for users, chatrooms, and their relationships.

c. Overview (Project Planning and Setup)

TopicHive is a feature-rich platform designed to facilitate dynamic, chatroom-based interactions among users while providing robust administrative capabilities. The system enables users to create and participate in chatrooms, and administrators to manage users and chatrooms effectively.

The system is built on a modular architecture that integrates:

Frontend: A user-friendly interface built using AngularJS, providing seamless interaction for both regular users and administrators.

Backend: A Java-based servlet framework that processes user requests, manages database interactions, and enforces business rules.

Database: An H2 relational database to store user profiles, chatroom data, and user-to-chatroom mappings with defined relationships and constraints.

Key Features

User Management:

User authentication via email and password.

Differentiation between regular users and admins based on roles.

Admin functionalities such as banning users and upgrading users to admins.

Chatroom Management:

Chatroom creation with name and description fields.

Automatic addition of all admin users to newly created chatrooms.

Chatroom deletion, which cascades to related user mappings.

Tracking Multiple Simultaneous Users:

We use sessionStorage.setItem() function in javascript to store the unique identifying user (Email, in our case). This data is tied to the page session and is not shared across tabs or windows. Thus, multiple users can be logged in from different windows (or different computers altogether) and content that is specific to that user (such as communities the user is a part of) are displayed for each session.

Role-Based Functionalities:

1. Admin-only actions:

View all users.

Manage chatrooms (create, delete).

Ban/unban users.

Upgrade regular users to admins.

2. Regular user actions:

Join and leave chatrooms.

Participate in discussions.

Database Integrity:

Well-defined relationships between USERS, CHATROOMS, and USERTOCHATROOM tables.

Enforced constraints to prevent data inconsistencies (e.g., cascading deletions).

4. System Architecture

a. Architectural Design

Figure 1:

Architecture diagram

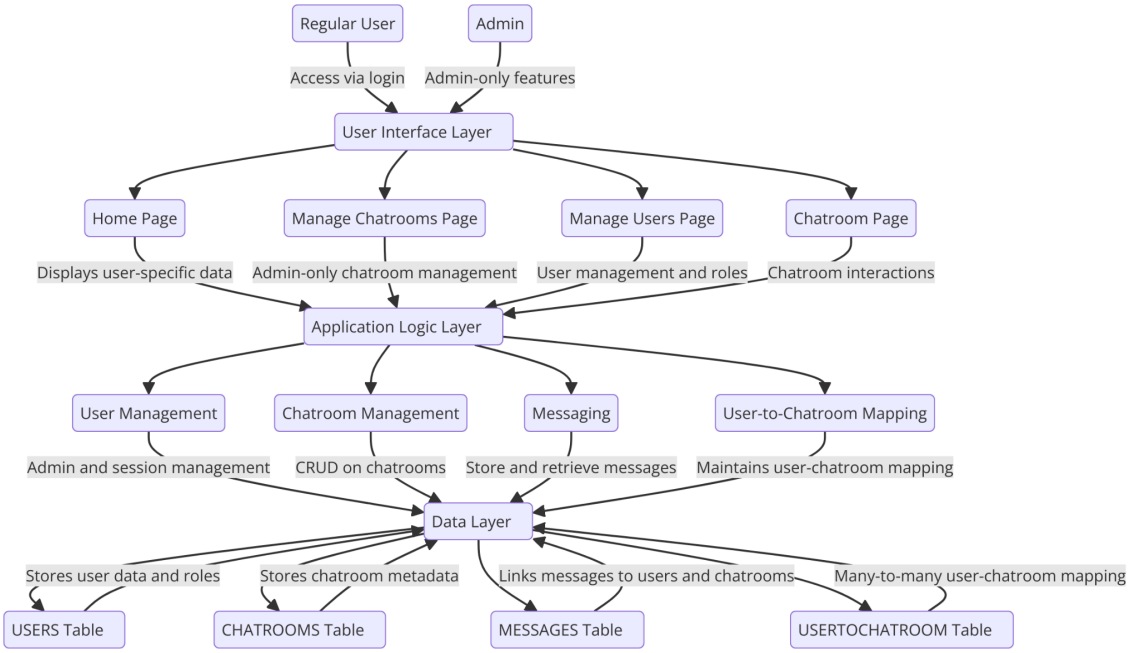


Figure denoting system architecture (3 layers – Presentation, Application, and Data layer)

1. User Interface Layer

The **User Interface Layer/Presentation Layer** handles user interaction and provides the graphical interface for the TopicHive platform. This includes the following components:

Responsibilities:

- Display data from the Application Layer (e.g., chatrooms, messages, user profiles).
- Capture user input (e.g., login credentials, chatroom creation details, messages).
- Send user actions to the Application Layer via HTTP requests.
- Role-based UI customization (e.g., admin vs. regular user).

Technologies:

- **HTML/JavaScript:** Markup and styling for user interfaces.
- **AngularJS:** Framework for dynamic UI rendering and interaction.

Pages and Features:

1. **Home Page:**
    - Displays user-specific chatrooms and admin-only options (e.g., "View All Users," "Manage Chatrooms").
2. **Manage Chatrooms Page:**
    - Allows admins to view, create, and delete chatrooms.
3. **Manage Users Page:**
    - Provides options to ban/unban users and upgrade users to admins.
4. **Chatroom Page:**
    - Displays messages in a chatroom and allows users to send messages.

2. Application Logic Layer (Tool)

The **Application Layer** processes business logic, orchestrates workflows, and bridges the Presentation and Data layers. It is implemented using Java Servlets.

Responsibilities:

- Authenticate users and manage sessions.
- Enforce role-based access control (e.g., admin-only functionalities).
- Perform CRUD operations on chatrooms, users, and messages.
- Handle cascading updates or deletions to maintain database integrity.
- Format and return data to the Presentation Layer (e.g., JSON responses).

Key Actions (Servlet Methods):

1. **User Management:**
    - `checkAdmin`: Determines if a user is an admin.
    - `banUser`: Sets `IS_BLOCKED = TRUE` for a user.
    - `removeBan`: Sets `IS_BLOCKED = FALSE` for a user.
    - `upgradeToAdmin`: Sets `IS_ADMIN = TRUE` for a user and adds them to all chatrooms.
2. **Chatroom Management:**
    - `getAllChatrooms`: Fetches all chatrooms from the database.

    - `createChatroom`: Inserts a new chatroom into the `CHATROOMS` table and adds all admin users to it.
    - `deleteChatroom`: Deletes a chatroom and cascades deletions to related `USERTOCHATROOM` entries.
3. **Messaging:**
    - `retrieveChats`: Fetches all messages for a chatroom.
    - `sendMessage`: Adds a new message to the `MESSAGES` table.
4. **User-to-Chatroom Mapping:**
    - `addToAllChatrooms`: Automatically adds users to chatrooms based on specific conditions (e.g., new admin since admins need to be added to all chatrooms, new chatroom).

3. Data Layer (Database)

The **Data Layer** manages persistent storage and ensures data consistency through relational tables and constraints.

Responsibilities:

- Store and retrieve user, chatroom, and message data.
- Maintain referential integrity using foreign key constraints.
- Enforce cascading deletions and updates.

Schemas:

USERS Table:

- Stores user data, roles (IS_ADMIN), and account status (IS_BLOCKED).
- Used for authentication and role-based access control.

CHATROOMS Table:

- Stores metadata for chatrooms (name and description).
- Referenced by the USERTOCHATROOM and MESSAGES tables.

MESSAGES Table:

- Stores messages sent in chatrooms, linking each message to a chatroom (`CHATROOM_ID`) and sender (`EMAIL`).
- Supports message deletion with the `IS_DELETED` flag.

USERTOCHATROOM Table:

- Maintains a many-to-many relationship between users and chatrooms.
- Automatically updated when chatrooms are deleted (`ON DELETE CASCADE`).

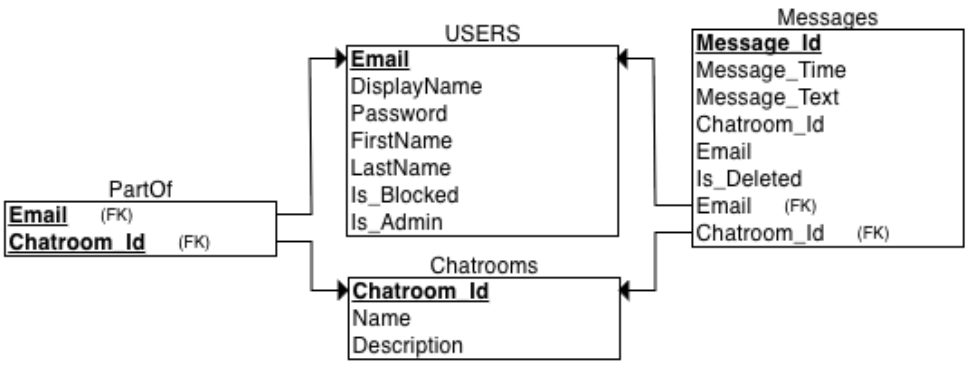
Constraints and Relationships:

- `CHATROOMS.ID` ↔ `MESSAGES.CHATROOM_ID` (One-to-Many).
- `USERS.EMAIL` ↔ `MESSAGES.EMAIL` (One-to-Many).
- `USERS.EMAIL` ↔ `USERTOCHATROOM.USER_EMAIL` (Many-to-Many).
- `CHATROOMS.ID` ↔ `USERTOCHATROOM.CHATROOM_ID` (Many-to-Many).

5. Data Design

Figure 2:

Database table attributes - Entity Relationship Diagram



ERD represents different tables and attributes along with their relationship by utilizing foreign keys.

5.1 Data Description

The TopicHive platform utilizes four relational tables to manage users, chatrooms, messages, and user-to-chatroom mappings. These tables are designed with a set of attributes and constraints to enable functionalities such as user authentication, role management, chatroom administration, and messaging. Each table is interlinked through foreign key relationships to maintain data consistency and integrity across the system.

The detailed structure and purpose of each table are outlined in the Data Dictionary section. The relationships between these tables are illustrated in the Entity Relationship Diagram (ERD) provided earlier (refer Figure 1), which demonstrates the logical connections between entities and their attributes.

USERS Table:

This table stores information about registered users, including their email, display name, password, and roles (admin or regular user). It also tracks whether a user is blocked from accessing the platform.

CHATROOMS Table:

This table contains metadata for all chatrooms, such as their names and descriptions. Each chatroom is uniquely identified by its ID.

MESSAGES Table:

This table records all messages exchanged within chatrooms. Each message is linked to a chatroom (CHATROOM_ID) and the sender (EMAIL). It also includes a timestamp and a flag to indicate whether a message is deleted.

USERTOCHATROOM Table:

This junction table establishes a many-to-many relationship between users and chatrooms, specifying which users are members of which chatrooms.

5.2 Data Dictionary

1. **USERS Table**

Column Name	**Data Type**	**Description**
EMAIL	VARCHAR(50) (PK)	Unique identifier for each user. Used as the primary key.
DISPLAYNAME	VARCHAR(20)	The display name chosen by the user, shown in chatrooms and user lists.
PASSWORD	VARCHAR(12)	The user's encrypted password for authentication.
FIRSTNAME	VARCHAR(25)	The user's first name.
LASTNAME	VARCHAR(25)	The user's last name.

IS_BLOCKED	BOOLEAN	Indicates whether the user's account is blocked (TRUE = blocked, FALSE = active).
IS_ADMIN	BOOLEAN	Indicates if the user has admin privileges (TRUE = admin, FALSE = regular user).

2. **CHATROOMS Table**

Column Name	**Data Type**	**Description**
ID	INT (PK)	Unique identifier for each chatroom. Used as the primary key.
NAME	VARCHAR (100)	The name of the chatroom, displayed to users in the chatroom list.
DESCRIPTION	VARCHAR (300)	A brief description of the chatroom, providing context or purpose for its use.

3. **MESSAGES Table**

Column Name	**Data Type**	**Description**
MESSAGE_ID	INT (PK)	Unique identifier for each message. Used as the primary key.
MESSAGE_TIME	DATETIME	Timestamp indicating when the message was sent. Default value is the current timestamp.
MESSAGE_TEXT	VARCHAR(300)	The text content of the message.

CHATROOM_ID	INT (FK)	Foreign key referencing the ID column in the CHATROOMS table. Indicates the chatroom the message belongs to.
EMAIL	VARCHAR(50) (FK)	Foreign key referencing the EMAIL column in the USERS table. Indicates the sender of the message.
IS_DELETED	BOOLEAN	Flag indicating if the message is deleted (TRUE = deleted, FALSE = active).

4. USERTOCHATROOM Table

Column Name	Data Type	Description
USER_EMAIL	VARCHAR(50) (FK)	Foreign key referencing the EMAIL column in the USERS table. Indicates the user in the chatroom.
CHATROOM_ID	INT (FK)	Foreign key referencing the ID column in the CHATROOMS table. Identifies the associated chatroom.

7. Human Interface Design (Initial Screen Sketches)

In this section, we show the initial sketches of various screens we wish to implement in TopicHive along with operations that might modify the database from each screen.

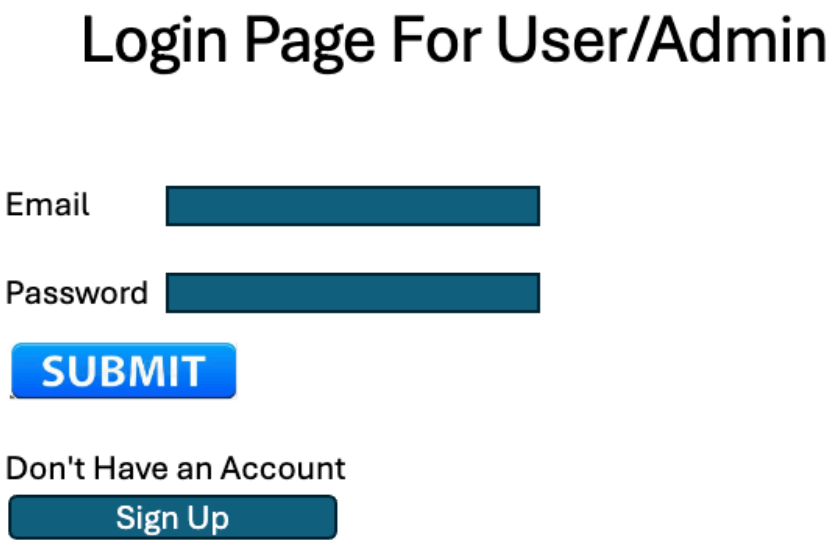
7.1 Screen Images

i. Login Page

The Login Page serves as the primary entry point to the TopicHive platform for both regular users and admin users. It provides a simple and intuitive interface for authentication, ensuring role-based access once credentials are validated. There are no direct implications to the database from this screen. However, the unique user identifier (email) is set using sessionStorage.setItem() in javascript to track and enable multiple users logged in simultaneously.

Figure 3:

Login page For User/Admin



ii. Sign-up Page

The Sign-up page is for new users to create an account on TopicHive and will consist of a form to enter user details such as email, password, display-name etc. Implications to the database include -

- Insert new row into the Users table with the user info.

Figure 4:

Sign Up page For User

iii. Home Page for Admin

The Home Page for Admin serves as the central dashboard for administrators of the TopicHive platform. It provides access to advanced functionalities for managing users and chatrooms while also enabling participation in chatrooms as a regular user. There are no direct implications to the database from this page.

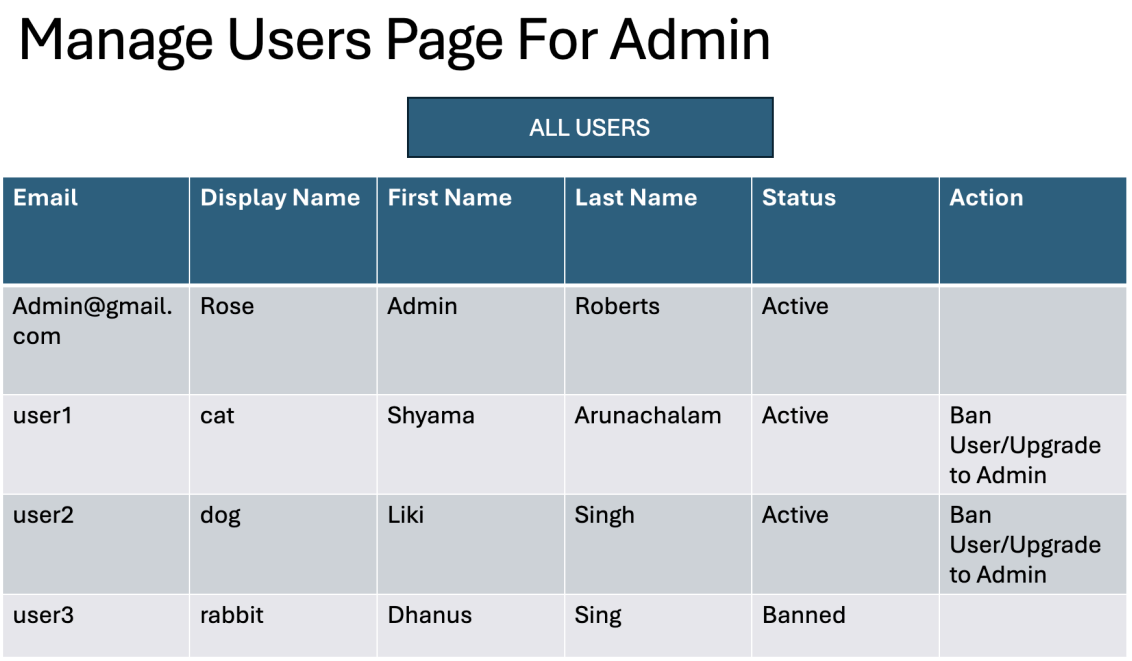
Figure 5:
Home Page for Admin

iv. Manage Users Page for Admin

The Manage Users page is a dedicated interface for admin users to oversee and control the platform's user base. It provides the ability to view user details, enforce platform rules, and manage roles efficiently. Implications to the database include -

- Ban user - Modifies the Users table to set is_Blocked column to true for a particular user
- Remove Ban - Modifies the Users table to set is_Blocked column to false for a particular user
- Upgrade to Admin - Modifies the Users table to set is_Admin column to true for a particular user. Inserts into the UserToChatroom table to add user to all the chatrooms

Figure 6:
Manage Users



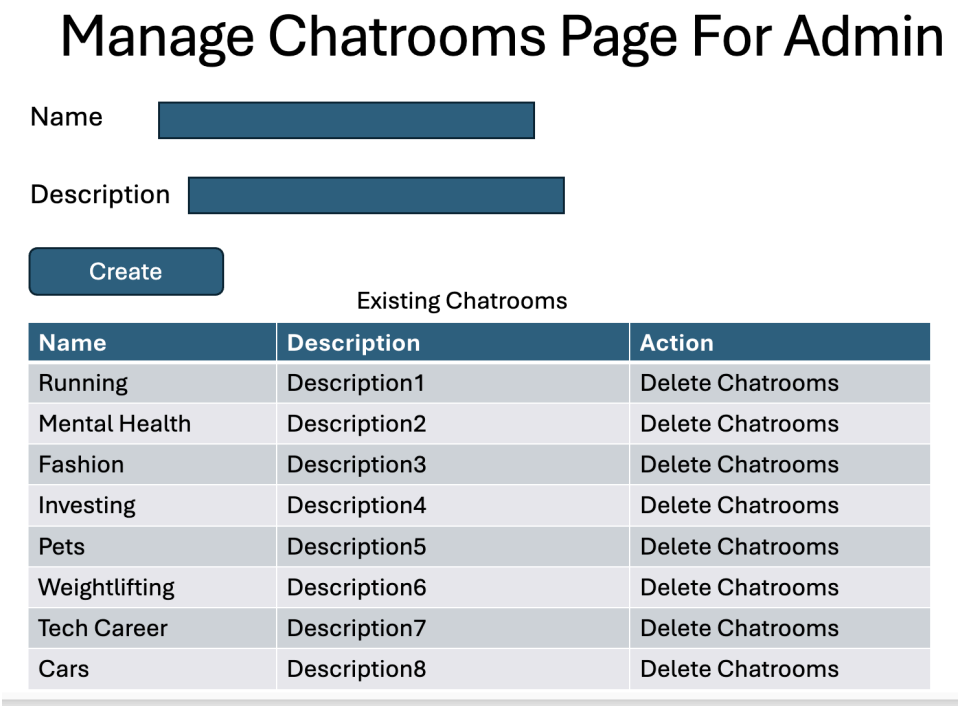
v. Manage Chatrooms Page for Admin

The Manage Chatrooms Page provides admin users with a dedicated interface to oversee and manage all chatrooms on the TopicHive platform. Admins can create new chatrooms, view existing ones, and delete chatrooms as needed, with changes cascading across related entities to maintain data consistency. Implications to the database include -

- Create New Chatroom - Inserts a new row in the Chatroom table with the chatroom id,name, and description. Inserts into the UserToChatroom table to admins to the newly-created chatroom
- Delete Chatroom - Deletes a row in the Chatroom table for the corresponding chatroom. Deleted into the UserToChatroom table to admins to the newly-created chatroom

Figure 7:

Manage Chatrooms



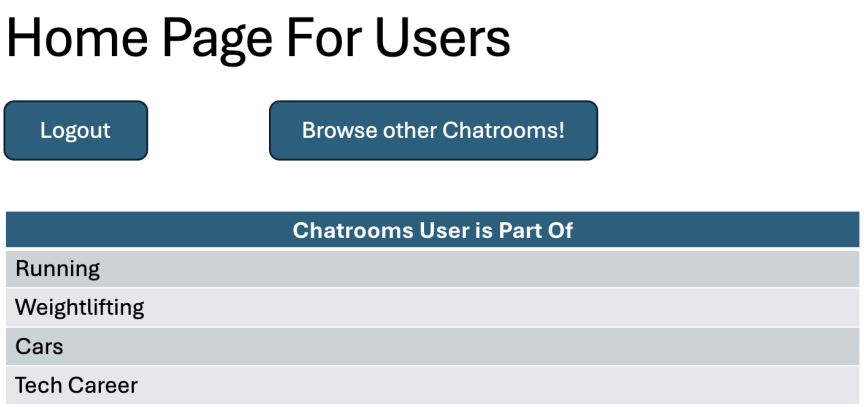
vi. Home Page for User

The Home Page for Users is the central hub for regular users on the TopicHive platform. It allows users to view the chatrooms they are members of, join new chatrooms, leave existing chatrooms, and interact with the platform's features tailored to their role. Implications to the database include -

- Leave Chatroom - Modifies the UserToChatroom table to remove the user from the chatroom

Figure 8:

Home Page for User



vii. Browse Other Chatrooms Page for User

The Browse Other Chatrooms page allows users to discover and join chatrooms they are not currently a member of. It serves as an exploration interface for users to expand their participation within the TopicHive platform. Implications to the database include -

- Join Chatroom - Modifies the UserToChatroom table to add user to the chatroom

Figure 9:

Browse Other Chatrooms

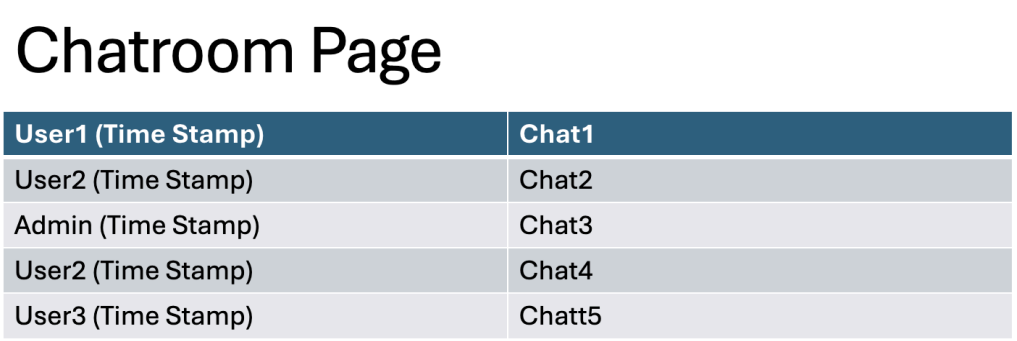
viii. Chatroom Page

Main page that displays messages in the various chatrooms. Allows user to read previous messages and type in a new message. Implications to the database include -

- Send Message - Insert new message (along with timestamp chatroom/user info etc.) into the messages table

Figure 10:

Chatroom Page



8. Timeline

Phase	Estimated Duration	Actual Duration
Project SetUp (Design Document and Database Creation)	12 hours	12-15 hours
Core Features (Home and Chat)	5-6 hours	12 hours
Admin Features	5-6 hours	10 hours
Users Features	5-6 hours	6-7 hours
Testing and Debugging	2-3 hours	2 hours
Instruction and Test Document	3-4 hours	3-4 hours