**A PROJECT REPORT ON**

**Real-Time Chat Application**

**SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE**

**OF**

**BACHELOR OF BUSINESS ADMINISTRATION IN**

**COMPUTER APPLICATION**

**BY**

Ashwini Sunil Kamble Examination Seat No. :5941

Snehal surykant subhedar Examination Seat No.:5978

**UNDER THE GUIDANCE OF**

Prof. Watpade A.S.



## Year 2025-26

***Rayat Shikshan Sanstha's***

**S. M. JOSHI COLLEGE, PUNE HADAPSAR, PUNE 411028**

**DEPARTMENT OF COMPUTER SCIENCE AND APPLICATION 2025-26**

**CERTIFICATE**

This is to certify that the project report entitled Real Time Chat Application

**Submitted by**

Ashwini Sunil Kamble Examination Seat No.:5941

Snehal Suryakant subhedar Examination Seat No.: 5978

is a bonafide work carried out by them under the supervision of **Prof. Watpade A.S.** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University for the award of the Degree of **Bachelor of Business Administration in (Computer Application).**

This project report has not been earlier submitted to any other Institute or University for the award of any degree or diploma.

## Prof. Watpade A. S. Hon. Sangeeta Yadav

Project Guide HOD

## Internal Examiner External Examiner

**ACKNOWLEDGEMENT**

We feel great pleasure in expressing our deepest sense of gratitude and sincere thanks to our guide **Prof. Watpade A.S.** for their valuable guidance during the Project work, without which it would have been a difficult task. I have no words to express my sincere thanks for valuable guidance, extreme assistance and cooperation extended to all the Staff Members of our department.

This acknowledgement would be incomplete without expressing our special thanks to **Hon. Sangeeta Yadav,** Head of the Computer Application department for her support during the work.

We would also like to extend our heartfelt gratitude to our Principal, **Dr. S. T. Salunkhe** who provided a lot of valuable support, mostly being behind the veils of college bureaucracy.

Last but not the least we would like to thank all the Teaching, Non-teaching staff members of our Department, our Parents and our colleagues those who helped us directly or indirectly for completion of this project successfully.

Ashwini Sunil Kamble exam no.5941 Snehal Suryakant subhedar exam no.5978

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **INDEX** |  |
| **Sr.No** |  | **Chapter Name** | **PageNo.** |
| **1** |  | **Abstract** | **1** |
| **2** |  | **Introduction** | **2** |
|  | 2.1 | Motivation |  |
|  | 2.2 | Problem Statement |  |
|  | 2.3 | Purpose/Objective and Goals |  |
|  | 2.4 | Literature Survey |  |
| **3** | 2.5 | Project Scope & Limitation  **System Analysis** | **4** |
|  | 3.1 | Existing System |  |
|  | 3.2 | Scope and Limitations |  |
|  | 3.3 | Perspective and Features |  |
|  | 3.4 | Stakeholders |  |
| **4** | 3.5 | Requirement Analysis  **System Design** | **8** |
|  | 4.1 | Design Constraints |  |
|  | 4.2 | System Model: DFD |  |
|  | 4.3 | Data Model |  |
| **5** | 4.4 | User Interfaces  **Implementation Details** | **22** |
|  | 5.1 | Software Requirements |  |
|  | 5.2 | Hardware Requirements |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **6** |  | **Outputs and Report Testing** | **24** |
|  | 6.1 | Test Cases |  |
|  | 6.1.1 | Black Box Testing |  |
|  | 6.1.2 | White Box Testing |  |
| **7** |  | **Conclusion & Recommendations** | **32** |
|  | 7.1 | Conclusion |  |
|  | 7.2 | Recommendations |  |
| **8** |  | **Future Scope** | **34** |
| **9** |  | **Bibliography & References** | **36** |

# CHAPTER 1: ABSTRACT

**ABSTRACT**

The advancement of web technologies has led to a growing demand for real-time communication systems. This project presents a Real-Time Chat Application developed using the Flask framework in Python integrated with Socket.IO for WebSocket-based communication. The application allows multiple users to communicate in real-time through an intuitive and responsive web interface.

The chat interface is built using HTML, CSS and JavaScript, offering users an easy-to-use platform for exchanging messages. The application supports multiple chat rooms, displays timestamps, and provides notifications when users join or leave a conversation. Optional database integration allows for user management and message storage.

# CHAPTER 2: INTRODUCTION

**MOTIVATION:**

**1.Improving Communication Tools**:  
Existing chat tools often come with limitations such as lack of customization, dependency on third-party platforms, or expensive premium services. Building our own application allows full control over features and design.

**2.Enhancing Full-Stack Skills**:  
This project integrates **frontend** development (HTML/CSS/JS), **backend** logic (Python Flask), **real-time networking**, and **database** operations (MySQL), giving us valuable experience in full-stack development.

**3.Scalability and Practicality**:  
A real-time chat system is a core feature in many modern applications like customer support, social media, team collaboration platforms, and gaming. Understanding how it works prepares us for real-world development scenarios.

**4.Security and User Management**:  
Implementing secure user authentication and data handling encourages the adoption of **best practices** in web development, such as password hashing, session management, and data validation.

**PROBLEM STATEMENT :**

* There is a need for a real-time communication platform that enables instant messaging between users without reloading the page. Furthermore, the solution should be lightweight, scalable, and built using open-source technologies that are easy to deploy and understand, especially for educational and prototyping purposes.
* Most existing chat platforms are either commercial, lack customization options, or do not provide full control over data and privacy. Additionally, many are not optimized for specific use cases like academic discussions, team collaboration, or internal organizational messaging.

**PURPOSE / OBJECTIVE & GOALS:**

To design and develop a real-time chat application using Python Flask and Socket.IO that allows multiple users to communicate seamlessly through a web interface. The primary purpose of this project is to **design and develop a web-based real-time chat application** that enables users to communicate instantly and efficiently

**Objectives:**

* To implement a chat system using Flask and WebSockets via Flask-SocketIO.
* To allow multiple users to send and receive messages in real-time.
* To create a user-friendly interface using HTML, CSS and JavaScript.
* To support user login, message timestamps, and multiple chat rooms.
* To demonstrate full-duplex communication using open-source tools.

**Goals:**

* Build a functional and scalable real-time web chat application.
* Enhance understanding of real-time event-driven communication.
* web development and Python programming knowledge to a real-world use case.

**LITERATURE SEVREY:**

Real-time communication has become a fundamental requirement in modern web applications, especially with the growing popularity of chat-based platforms for collaboration and customer interaction. This project explores the design and implementation of a **real-time chat application** using a tech stack consisting of:

**PROJECT SCOPE & LIMITATION**

* **Scope:**

1. **User Authentication and Authorization:**

Users can register, log in, and log out securely.

Only authenticated users can access chat functionalities.

1. **One-to-One Messaging:**

Users can send and receive real-time messages to/from other individual users.

1. **Real-Time Communication:**

Utilizes WebSockets (or similar technologies) to enable instant message delivery without refreshing the page.

1. **Message Persistence:**

All messages are stored in a database for future retrieval.

**Limitations:**

1. **No End-to-End Encryption:**

Messages are stored in plain text or basic encryption on the server, which means administrators could potentially access them. Designed for small-scale use. May not handle a large number of concurrent users or messages without performance degradation.

1. **No File Sharing:**

The application may not support sending images, videos, or documents.

1. **No Push Notifications:**

Users might not receive alerts if they’re not actively using the app.

1. **Basic Moderation Tools:**

No content filtering, blocking/reporting users, or admin moderation features.

# CHAPTER 3:

# SYSTEM ANALYSIS

**SYSTEM ANALYSIS**

**Existing Systems:** Several existing real-time chat applications serve as benchmarks or legacy systems. These include:

# WhatsApp, Telegram, Facebook Messenger: Feature-rich apps with global user bases.

# Slack, Microsoft Teams, Discord: Focus on team collaboration and enterprise communication.

# Open-source platforms: Like Rocket.Chat , Mattermost, and Matrix.

**Scope and Limitations of Existing Systems:**

**Scope of Existing Systems:**

# User Registration and Authentication

# Secure registration and login mechanisms.

# User profile management including name, profile photo, and status.

# Online/offline status indicators.

# 2. Friend Management System

# Ability to search for and add friends.

# Accepting or rejecting friend requests.

# Viewing and managing friend lists.

# Blocking and unblocking users.

# 3. Real-Time Messaging

# One-to-one text messaging with real-time delivery.

# Typing indicators and message status (sent, delivered, read).

# Support for emojis, media files (images, videos), and document sharing.

# Timestamps for all messages.

**Limitations of Existing Systems:**

# Limited customizability: Many apps don’t allow customization for business-specific needs.

# Data privacy concerns: Especially with third-party data sharing.

# High resource usage: Some apps are heavy on memory and CPU.

# Dependency on internet: Offline messaging or caching often limited.

# Proprietary limitations: Most platforms are closed-source and not extendable.

* **Lack of Media Sharing:** The messaging system supports only text messages. Sending images, videos, audio, or documents is not available.

**Project Perspective and Features:**

**Project Perspective:**

The real-time chat application can be viewed from multiple perspectives to better understand its design and functionality. From the user perspective, the system provides a simple and user-friendly platform for sending messages and making calls instantly, with a focus on privacy and ease of use

**Features:**

**1. User Module**

User registration and login functionality.

Unique user identification (e.g., username or email).

Basic profile management (name, email, password).

Secure authentication to prevent unauthorized access.

**2. Friends Module**

Search users by name or user name.

Send and receive friend requests.

Accept or reject friend requests.

View list of added friends (friend list).

**3. Message Module**

One-to-one real-time text messaging. View chat history with each friend.

Store and retrieve messages from the database.

**Requirement analysis - Functional requirements, performance requirements, security requirements:**

#### **Functional Requirements:**

1. User Module: The system shall allow a new user to register with a unique username and password.

2. Friends Module: The system shall prevent users from sending multiple requests to the same user if a request is already pending.

3. Message Module: The system shall display received messages instantly without refreshing (using WebSockets or similar). The system shall store all messages exchanged between users.

**Performance Requirements:**

* Scalability: The system should be built so it can easily expand in the future to handle more users, more messages, or new features like media sharing.
* Network Efficiency: minimize message size by removing extra metadata. Implement automatic reconnection logic in JavaScript (if Socket disconnects).
* Real-Time Updates: Changes like new messages or friend requests should appear immediately on the user interface without needing to refresh the page.

**Security Requirements:**

* Access Control: Basic login systems, often without role-based permissions.
* Data Protection: Limited encryption or backup mechanisms.
* No Unauthorized Access: Users cannot view or interact with accounts or messages they are not allowed to access.

# CHAPTER 4: SYSTEM DESIGN

* 1. **Design Constraints:**

**USER Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(10) | NO | PRIMARY | NULL | AUTO\_INCREMENT |
| name | varchar(50) | YES |  | NULL |  |
| email | varchar(50) | NO | UNIQUE | NULL |  |
| password | varchar(255) | NO |  | NULL |  |
| date\_created | datetime | YES |  | current\_timestamp() |  |

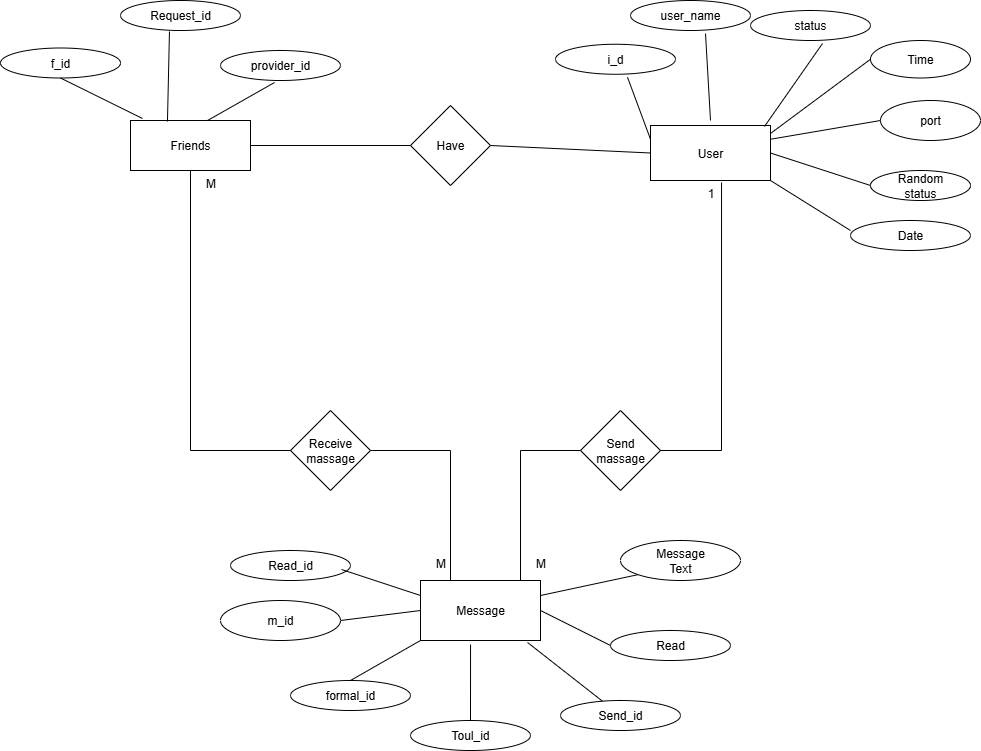
**MESSAGE Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(10) | NO | PRIMARY | NULL | AUTO\_INCREMENT |
| Sender id | int(10) | NO |  | NULL |  |
| Receiver  id | int(10) | NO |  | NULL |  |
| message | text | NO |  | NULL |  |
| timestamp | datetime | YES |  | current\_timestamp() |  |
| status | enum('sent','delivered','seen') | YES |  | 'sent' |  |

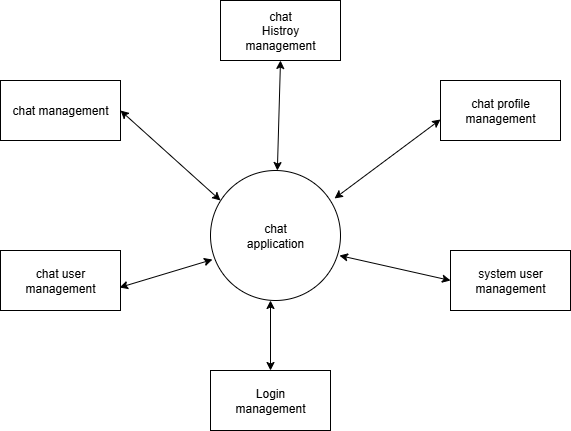
**CHATROOMS Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(10) | NO | PRIMARY | NULL | AUTO\_INCREMENT |
| room**\_**name | varchar(100) | YES |  | NULL |  |
| created**\_**by | int(10) | NO |  | NULL |  |
| created**\_**at | datetime | YES |  | current\_timestamp() |  |

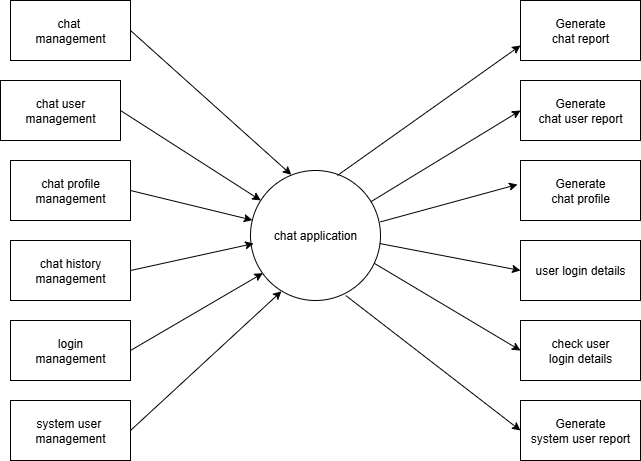
* 1. **System Model :**
     1. **ER DIAGRAM:**

****

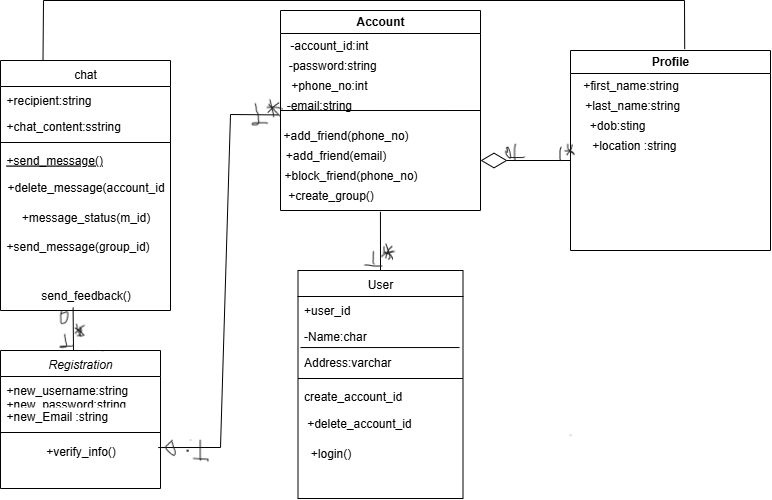
* + 1. **ZERO LEVEL DFD:**



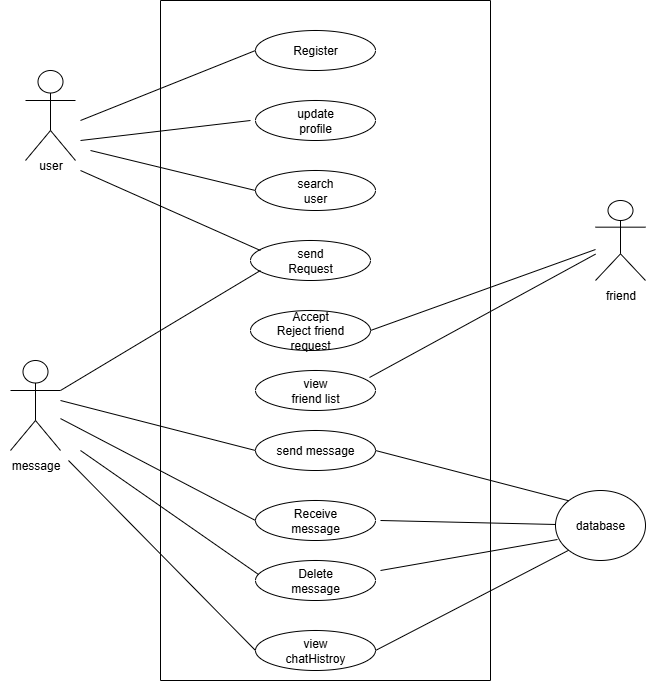
**4.2.2 ONE LEVEL DFD:**

****

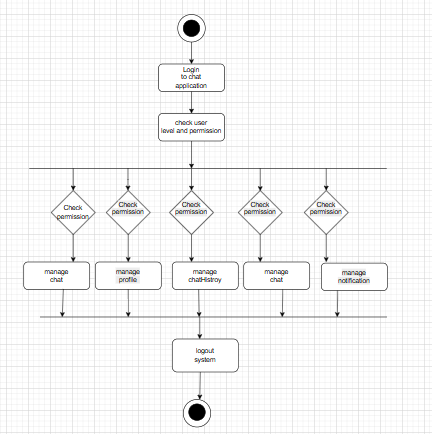
* + 1. **CLASS DIAGRAM:**



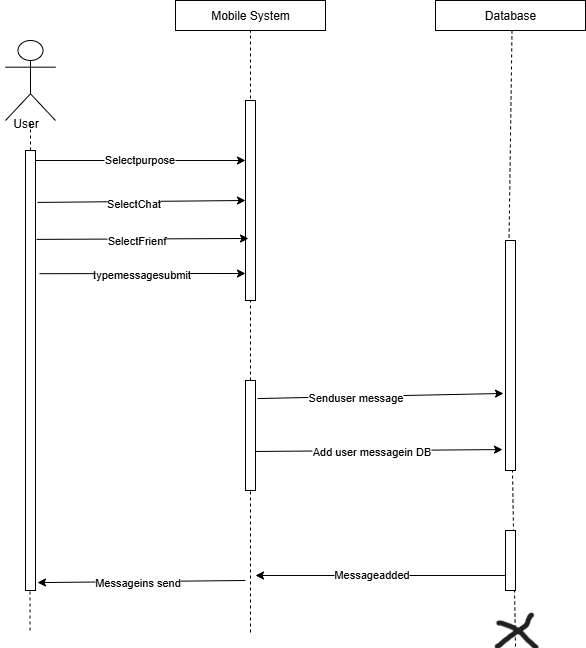
* + 1. **USE CASE DIAGRAM:**



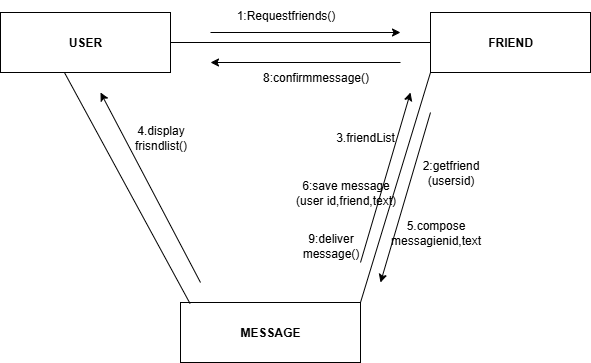
* + 1. **ACTIVITY DIAGRAM:**



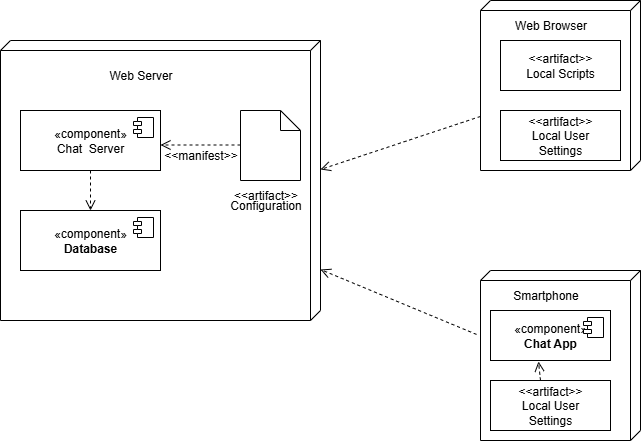
* + 1. **SEQUENCE DIAGRAM:**



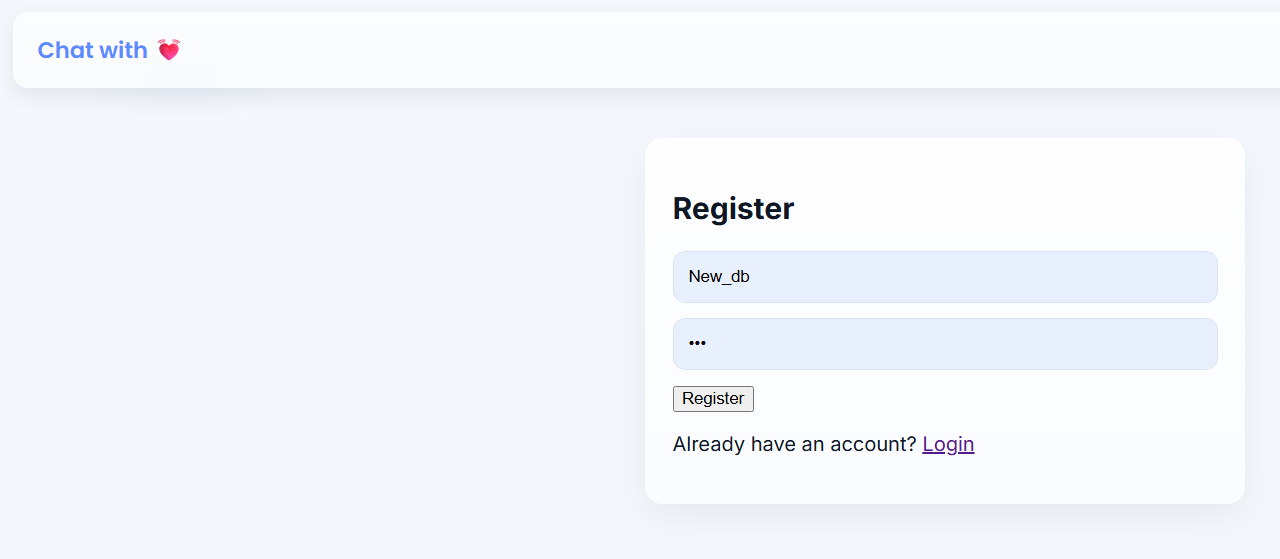
* + 1. **COLLABRATION DIAGRAM:**



**4.2.8 DEPLOYMENT DIAGRAM:**



* 1. **User Interfaces:**
     1. **Register Page:**

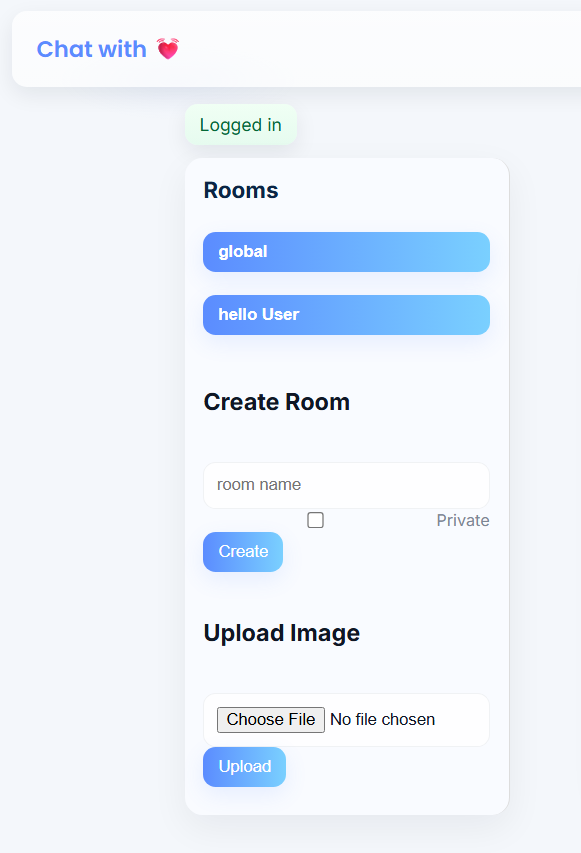
****

* + 1. **Login:**

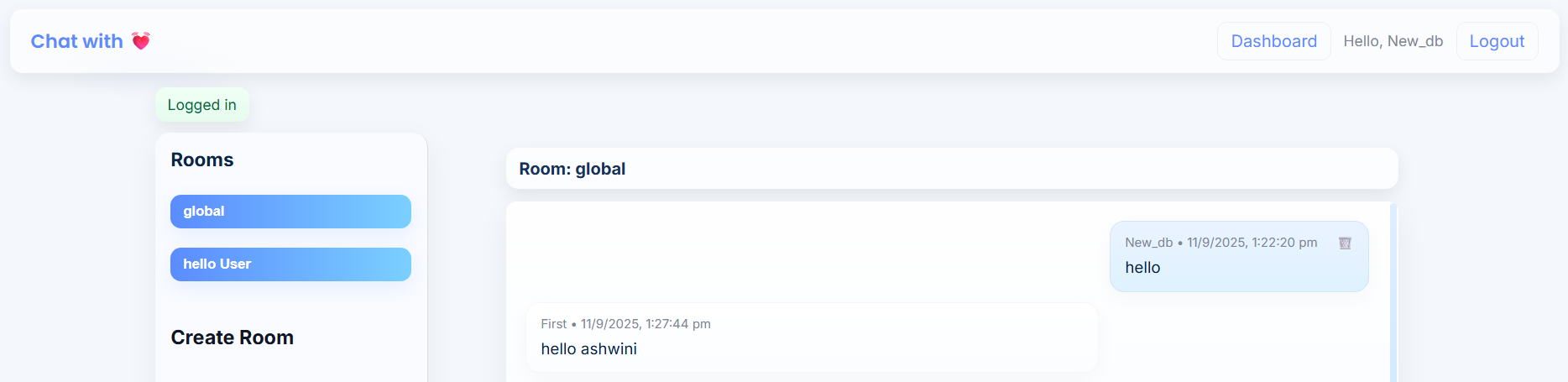
**A screenshot of a login form

AI-generated content may be incorrect.**

* + 1. **ChartRooms:**



* + 1. **Messages:**

****

# CHAPTER 5:

**IMPLEMENTATION DETAILS**

**IMPLEMENTATION DETAILS**

**Software Specifications:**

1.frontend Technology:

* HTML5
* CSS3
* JavaScript

2.Backend Technology:

* Python (Flask)

**Hardware Requirements:**

* RAM: 8GB
* Storage: SSD with 50+free space
* OS: Windows 10pro

# CHAPTER 6: OUTPUTS AND REPORTS TESTING

# 

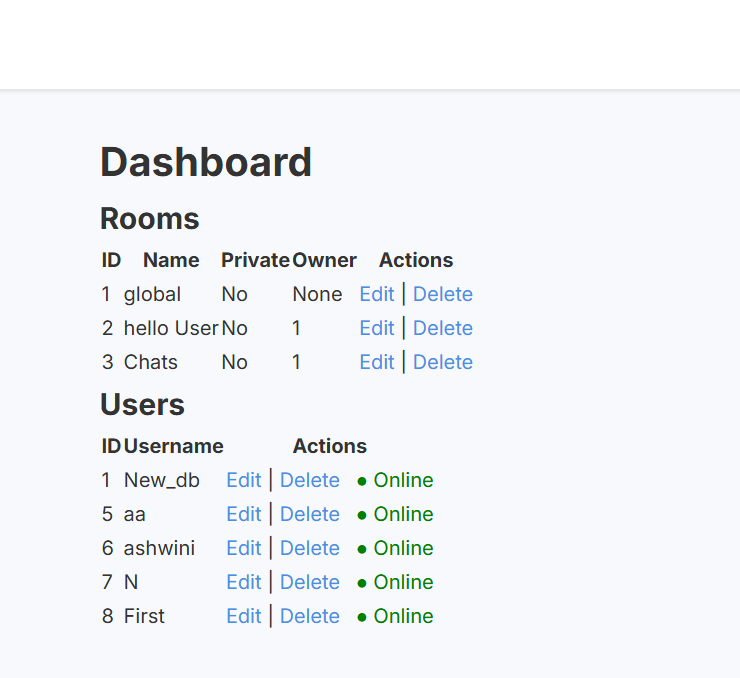
* 1. **Test Plan**

**6.1.1 BlackBox Testing:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Preconditions** | **Test Steps** | **Expected Results** | **Actual Results** | **Status** | **Remarks** |
| **BB-01** | Verify user registration | Registration page accessible | 1. Go to registration page 2. Enter valid username & password 3. Click Register | User account created successfully | User registered successfully | Pass | Data saved in DB |
| **BB-02** | Verify user login | Valid user exists | 1. Go to login page 2. Enter correct credentials 3. Click Login | User should log in successfully | User logged in | Pass | Session created |
| **BB-03** | Verify message sending | User in a chat room | 1. Enter message 2. Click Send | Message should appear instantly | Message sent & shown | Pass | Real-time update |
| **BB-04** | Verify file upload | User logged in | 1. Click Attach 2. Choose valid image 3. Send | Image sent successfully | Image displayed | Pass | File preview visible |
| **BB-05** | Verify logout | User logged in | 1. Click Logout button | User should be logged out | Redirected to login page | Pass | Session cleared |

# 

**A screenshot of a login form

AI-generated content may be incorrect.**

**WB-04:**

**A screenshot of a chat box

AI-generated content may be incorrect.**

# CHAPTER 7: CONCLUSIONAND

**RECOMMENDATIONS**

**CONCLUSION AND RECOMMENDATIONS**

The **Real Time Chat Application** project focuses on three main modules User, Friends, and Message and offers a clean, responsive interface for chatting. It proves that real-time communication systems can be implemented effectively using lightweight frameworks and open source technologies. While the application performs well under normal use, some limitations such as lack of media sharing, group chats, and advanced security features remain, leaving room for future development.

Deploying the application on a cloud platform with a scalable database will help handle more users and data efficiently. Additionally, integrating real-time notifications will keep users updated on new messages and friend requests, even when they are not actively using the app.

# CHAPTER 8: FUTURE SCOPE

### **FUTURE SCOPE**

The current chat application focuses solely on text messaging and voice/video calling, providing a simple yet effective platform for real-time communication. In the future, the system can be enhanced by introducing features such as user authentication, friend/contact management, and message history for improved usability.

**1.Real-Time Translation:**

Instant language translation for global communication, making conversations seamless across different languages.

**2.Stronger Privacy & Security:**

End-to-end encryption, blockchain-based messaging, and advanced security to protect user data and ensure trust.

#### **3.Responsive & Accessible Design:**

Fully functional on mobile, tablet, and desktop with accessibility for all users

#### **4.Searchable Chat History:**

Makes it easy to find past messages and conversations.

# CHAPTER 9: BIBLIOGRAPHY AND REFERENCES

**BIBLIOGRAPHY AND REFERENCES**

**WEB REFERENCES:**

* GeeksForGeeks
* **Adam la morre: <https://youtu.be/YDZPp0EnzEA?si=VlJy7jX-JYmP7Y5M>**