



# UNIVERSITY OF DHAKA

Department of Computer Science and Engineering

## CSE-3111 : Computer Networking Lab

### **Project Report: ConvoCorner**

#### **Submitted By:**

Fahim Shakil

Roll No : 06

Md. Mahmudul Hasan

Roll No : 10

#### **Submitted On :**

11 May, 2024

#### **Submitted To :**

Dr. Md. Abdur Razzaque

Dr. Md. Mamun Or Rashid

Dr. Muhammad Ibrahim

Md. Redwan Ahmed Rizvee

# 1 Project Title

ConvoCorner

## 2 Objectives

Our primary objective with the "ConvoCorner" project is to create an engaging and dynamic platform where users can connect with each other through chat rooms. We aim to facilitate seamless communication and collaboration among users. By providing a user-friendly interface and robust features, we aim to create meaningful interactions and build a vibrant community within the platform.

One of our key motivations behind this project is to address the need for virtual spaces where individuals can engage in real-time conversations, exchange ideas, and form connections irrespective of geographical barriers. In today's fast-paced world, where digital communication plays a vital role in our daily lives, we believe in the power of technology to bring people together and bridge gaps between them.

Through this project, we aspire to not only provide a space for casual conversations but also facilitate meaningful interactions that lead to knowledge sharing, collaboration, and community building. We are committed to creating a welcoming environment where users feel empowered to express themselves.

In summary, our objectives revolve around creating a user-centric chat room platform that promotes communication, collaboration, and community engagement. We are driven by the belief that fostering connections and facilitating meaningful interactions can enhance the overall user experience and contribute to building a stronger sense of community in the digital realm.

## 3 Methodology

- Our project adopts a three-tier architecture design approach.
- The frontend communicates with the backend using WebSocket and HTTP protocols.

- Backend clients transmit requests to the server via sockets for processing.
- Upon receiving requests, the server accesses the database and manipulates data using SQL statements.
- After processing, the server sends responses back to the client.
- Client-server communication is facilitated through TCP New Reno for reliable data transfer.
- Upon receiving responses, the client caches them if needed.
- Finally, the client returns the responses to the frontend via WebSocket.

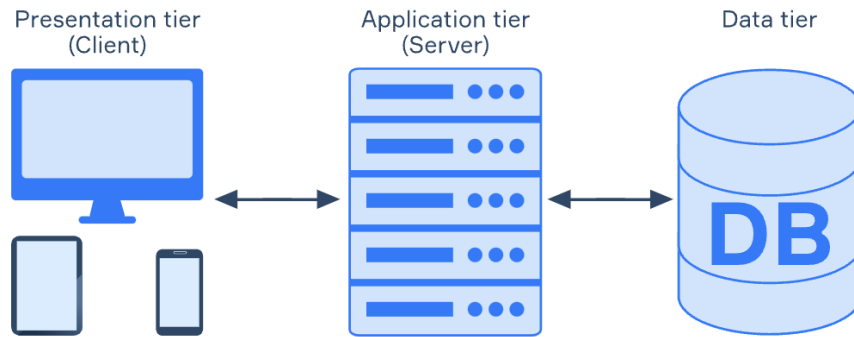


Figure 1: Three Tier Architecture

## 4 Tools and Techniques

### 4.1 Networking Tools

#### 4.1.1 TCP/IP Protocol Suite

We used the TCP/IP protocol suite for reliable and efficient data transmission over the Internet. TCP (Transmission Control Protocol) ensures reliable delivery of data packets by handling packet loss, retransmission, and flow control. IP (Internet Protocol) provides addressing and routing capabilities to enable communication between devices across different networks.

#### **4.1.2 WebSockets**

We used WebSockets for real-time, bidirectional communication between clients and the server. WebSockets enable persistent connections that will allow instant transmission of messages and updates between users within the chat room. This facilitates a more interactive and responsive chat experience compared to traditional HTTP polling or long-polling methods.

#### **4.1.3 Web Caching**

In our project, web caching has been instrumental in optimizing performance and scalability. By storing frequently accessed web resources closer to users, we reduce load times and network latency, enhancing the overall user experience. This technique also alleviates strain on backend servers by serving cached content, contributing to cost savings and improved scalability.

#### **4.1.4 HTTP Protocol**

We utilized the HTTP protocol for client-server communication over the web. HTTP requests are used to retrieve resources or perform actions on the server, while HTTP responses contain the requested data or status information. Implement RESTful APIs to define endpoints for various functionalities such as user authentication, message handling, and chat room management.

### **4.2 Technology Stack**

#### **4.2.1 Design**

Utilizing Tailwind CSS for streamlined and responsive design implementation, ensuring a visually appealing and user-friendly interface.

#### **4.2.2 Frontend**

Employing ReactJS for dynamic and interactive frontend development, enabling efficient rendering and seamless user experiences.

#### **4.2.3 Backend**

Leveraging FastAPI and Python for robust backend development, facilitating rapid API creation with high performance and scalability.

#### **4.2.4 Database**

Utilizing PostgreSQL as the database management system, ensuring data integrity, reliability, and efficient storage and retrieval of application data.

## **5 Outcomes**

### **5.1 List of Outcomes**

#### **5.1.1 User Authentication**

Secure user authentication and registration functionality are implemented, allowing users to create accounts and log in.

#### **5.1.2 Chat Room Creation and Join**

Users are able to create or join new chat rooms based on their interests, topics, or specific purposes, fostering diverse and engaging conversations.

#### **5.1.3 Real-Time Chat**

Real-time messaging functionality is implemented using WebSocket, enabling instant communication between users within the same chat room. It supports text-based messages and image attachments.

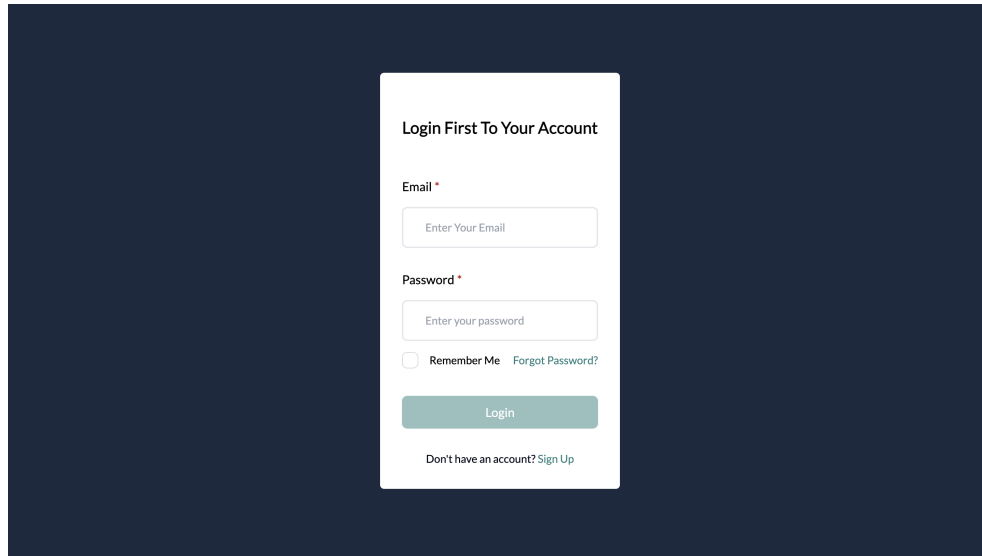
#### **5.1.4 Message History**

Message history are stored and retrieved within each chat room, enabling users to view past conversations and scroll through previous messages, facilitating continuity and reference.

#### **5.1.5 File Sharing**

Users are able to share images of any formate or other files such as text documents within chat rooms by uploading them directly from their devices. Users could view and download them as per their need.

## 5.2 Snapshot of Design



The login screen features a dark blue background with a white login form centered. The form is titled "Login First To Your Account". It includes input fields for "Email" and "Password", both marked with a red asterisk. Below the password field, there is a "Remember Me" checkbox and a "Forgot Password?" link. A green "Login" button is positioned below the form. At the bottom, a link says "Don't have an account? Sign Up".

Login First To Your Account

Email \*

Enter Your Email

Password \*

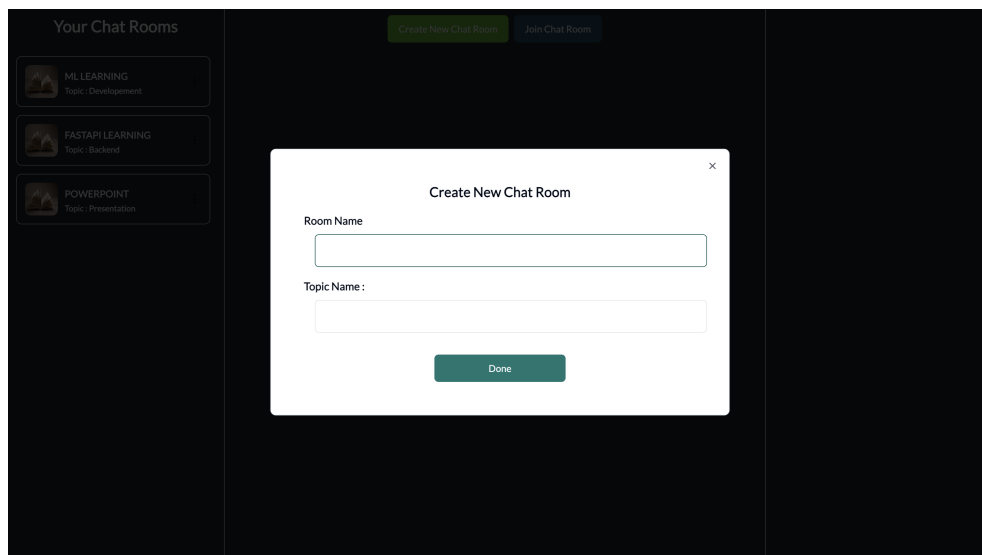
Enter your password

☐ Remember Me [Forgot Password?](#)

Login

Don't have an account? [Sign Up](#)

Figure 2: Login Screen



The "New Chat Room Create Screen" shows a dark interface. On the left, a sidebar titled "Your Chat Rooms" lists three rooms: "ML LEARNING" (Topic: Development), "FASTAPI LEARNING" (Topic: Backend), and "POWERPOINT" (Topic: Presentation). The main area has two buttons at the top: "Create New Chat Room" (green) and "Join Chat Room" (dark blue). A modal window titled "Create New Chat Room" is open, containing input fields for "Room Name" and "Topic Name :", and a green "Done" button.

Your Chat Rooms

- ML LEARNING  
Topic: Development
- FASTAPI LEARNING  
Topic: Backend
- POWERPOINT  
Topic: Presentation

Create New Chat Room Join Chat Room

Create New Chat Room

Room Name

Topic Name :

Done

Figure 3: New Chat Room Create Screen

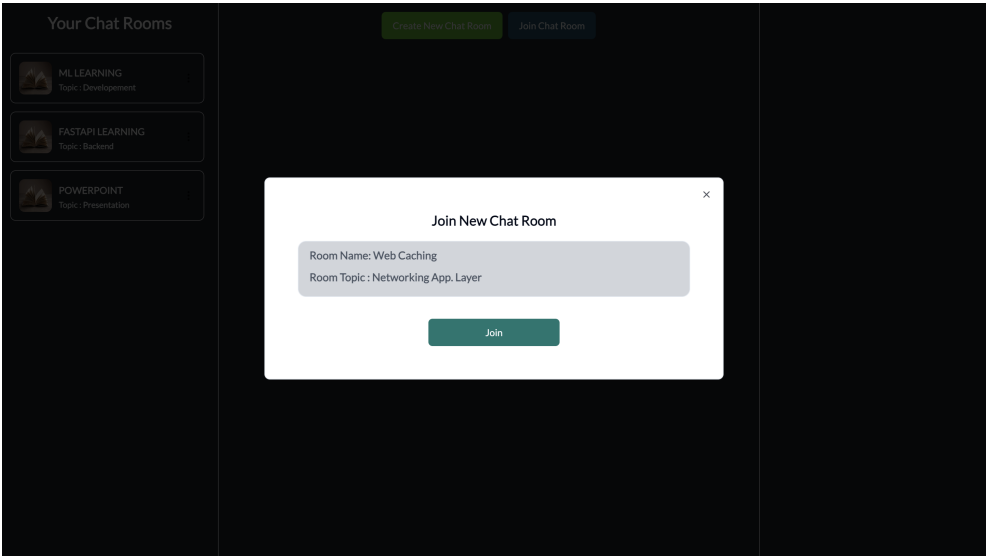


Figure 4: New Chat Room Join Screen

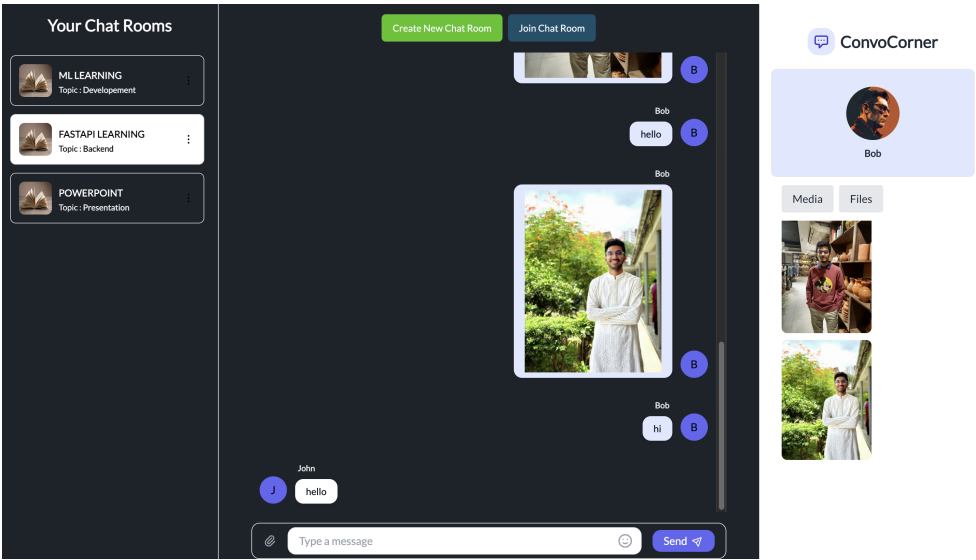


Figure 5: Messaging Screen

## 6 Analysis

- The adoption of TCP New Reno contributed to reliable and efficient data transmission over the network. By employing congestion control mechanisms, TCP New Reno optimized throughput while minimizing packet loss and network congestion. Performance metrics such as throughput, round-trip time, and congestion window size can be analyzed to evaluate the effectiveness of TCP New Reno in handling data traffic within our system.
- The implementation of web caching significantly enhanced the performance and scalability of our platform. By caching frequently accessed web resources, we reduced latency and alleviated the load on backend servers, leading to faster response times and improved user experience.
- WebSockets played a pivotal role in enabling real-time communication between users within chat rooms. By establishing persistent, bidirectional communication channels, WebSockets facilitated instant message delivery and reduced latency, enhancing the responsiveness and interactivity of our platform.
- For HTTP protocols used in our API, we observed robust and versatile communication capabilities, supporting a wide range of client-server interactions. HTTP's request-response model provided a standardized and efficient mechanism for transmitting data over the web, enabling seamless integration with various client applications and platforms.



```

Sequence no 8241152 is transferred to the upper layer

Seq no 8249344 has arrived
Seq no 8249344 is received
Sent ack 8257536
Seq no 8257536 has arrived
Seq no 8257536 is received
Sent ack 8265728

Sequence no 8249344 is transferred to the upper layer
Sequence no 8257536 is transferred to the upper layer

Seq no 8265728 has arrived
Seq no 8265728 is received
Sent ack 8273920

Sequence no 8265728 is transferred to the upper layer

Seq no 8273920 has arrived
Seq no 8273920 is received
Sent ack 8282112

```

Figure 6: Client receiving data using TCP

```

CWND: 10.235728619753006 MSS
SSTHRESH: 2.284942095178838 MSS
Seq no 8282112 is sent

Received ack 8282112

[CONGESTION AVOIDANCE]

CWND: 10.333425621808159 MSS
SSTHRESH: 2.284942095178838 MSS
Received ack 8290304

[CONGESTION AVOIDANCE]

CWND: 10.430198951060131 MSS
SSTHRESH: 2.284942095178838 MSS
Sent file successfully

Execution time: 8859.71188545227 ms
Packet loss rate: 6.642066420664207 %
Estimated Round Trip: 49.97793078693854 ms

```

Figure 7: Server sending data using TCP

```

Seq no 8282112 has arrived
Seq no 8282112 is received
Sent ack 8290304

Sequence no 8282112 is transferred to the upper layer

Successfully received file
Execution time: 10973.809719085693 ms

```

Figure 8: Execution time without caching

```

INFO: connection closed
INFO: ('127.0.0.1', 55105) - "WebSocket /messages/ws/13?user_id=3" [accepted]
INFO: connection open
Execution time: 29.504060745239258 ms
Execution time: 22.18794822692871 ms
INFO: 127.0.0.1:55108 - "GET /messages/13 HTTP/1.1" 200 OK
INFO: 127.0.0.1:55109 - "GET /messages/images/13 HTTP/1.1" 200 OK

```

Figure 9: Execution time with caching

## 7 Conclusion

In conclusion, our project has successfully implemented a dynamic chat room platform, leveraging a comprehensive technology stack including TCP New Reno, web caching, WebSockets, and HTTP protocols. With a focus on seamless communication and collaboration, our platform has facilitated meaningful interactions and community engagement, transcending geographical barriers and fostering global connections.

The applications of our project are diverse, ranging from casual conversations to professional collaboration, educational endeavors, and social networking. By providing users with a user-friendly interface and robust features, we have created a space where individuals can exchange ideas, share knowledge, and form meaningful connections irrespective of their physical location.

Looking ahead, we have ambitious plans to further enhance and expand the capabilities of our platform. This includes integrating advanced features such as enhanced moderation tools, voice and video chat functionalities, and community-driven feature development. Additionally, we aim to explore opportunities for strategic partnerships, continuous optimization, and user-driven innovation to ensure the con-

tinued growth and relevance of our platform in the ever-evolving landscape of digital communication.

In essence, our project embodies the power of technology to bring people together, facilitate collaboration, and build vibrant communities in the digital realm. We are committed to pushing the boundaries of what is possible, empowering users to connect, communicate, and thrive in a dynamic and interconnected world.