A Course Based Project Report on

Chat Application using TCP & UDP in Python

Submitted to the

Department of CSE-(CyS, DS) and AI&DS

in partial fulfilment of the requirements for the completion of course
COMPUTER NETWORKS AND ETHICAL HACKING LABORATORY
(22PC2CY201)

BACHELOR OF TECHNOLOGY

IN

Department of CSE-(CyS, DS) and AI&DS

Submitted by

23071A067C1 – KISHORE

23071A67C2 – SHAFIYA

23071A67C3– SHAIK SAMREEN

23071A67C4 – SOHAN JWALIYA KASA

23071A67C5– SYED NASEERA

Under the guidance of

Mrs. G. USHA RANI

(Course Instructor)

Assistant Professor, Department of CSE-(CYS,DS) AND AI&DS
VNRVJIET

Department of CSE-(CyS, DS) and AI&DS

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY

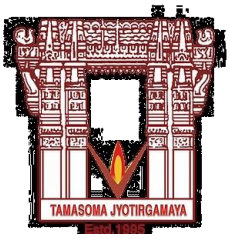
An Autonomous Institute, NAAC Accredited with 'A++' Grade, NBA
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

November 2025

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade, NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses, Approved by AICTE, New Delhi, Affiliated to JNTUH, Recognized as "College with Potential for Excellence" by UGC, ISO 9001:2015 Certified, QS I GUAGE Diamond Rated
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet(SO), Hyderabad-500090, TS, India

Department of CSE-(CyS, DS) and AI&DS



CERTIFICATE

This is to certify that the project report entitled "**Chat Application using TCP & UDP in Python"** is a bonafide work done under our supervision and is being submitted by **Mr. Kishore (23071A067C1),Ms. Shariya(23071A67C2), Ms. Shaik Samreen (23071A67C3), Mr. Sohan Jwaliya Kasa (23071A67C4), Ms. Syed Naseera(23071A67C5)** in partial fulfilment for the award of the degree of **Bachelor of Technology** in **CSE-(CyS, DS) and AI&DS**, of the VNRVJIET, Hyderabad during the academic year 2025-2026.

Mrs. G. Usha Rani
Assistant Professor
Dept of CSE-(CyS, DS) and AI&DS

Dr. T. Sunil Kumar
Professor & HOD
Dept of CSE-(CyS, DS)and AI&DS

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade, NBA

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

Department of CSE-(CyS, DS) and AI&DS

DECLARATION

We declare that the course based project work entitled **"Chat Application using TCP & UDP in Python"** submitted in the Department of **CSE-(CyS, DS) and AI&DS**, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology in CSE-(CyS, DS) and AI&DS** is a bonafide record of our own work carried out under the supervision of **Mrs.G.Usha Rani, Assistant Professor, Department of CSE-(CyS, DS) and AI&DS , VNRVJIET.** Also, we declare that the matter embodied in this thesis has not been submitted by us in full or in any part there of for the award of any degree/diploma of any other institution or university previously.

Place: Hyderabad.

23071A067C1 – KISHORE

23071A67C2 – SHAFIYA

23071A67C3– SHAIK SAMREEN

23071A67C4– SOHAN JWALIYA KASA

23071A67C5 – SYED NASEERA

ACKNOWLEDGEMENT

We express our deep sense of gratitude to our beloved President, Sri. D. Suresh Babu, VNR Vignana Jyothi Institute of Engineering & Technology for the valuable guidance and for permitting us to carry out this project.

With immense pleasure, we record our deep sense of gratitude to our beloved Principal, Dr. C.D Naidu, for permitting us to carry out this project.

We express our deep sense of gratitude to our beloved Professor Dr.T.SUNIL KUMAR, Professor and Head, Department of CSE-(CyS, DS) and AI&DS , VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad- 500090 for the valuable guidance and suggestions, keen interest and through encouragement extended throughout the period of project work.

We take immense pleasure to express our deep sense of gratitude to our beloved Guide, Mrs. G. USHA RANI, Assistant Professor in CSE-(CyS, DS) and AI&DS, VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad, for her valuable suggestions and rare insights, for constant source of encouragement and inspiration throughout my project work.

We express our thanks to all those who contributed for the successful completion of our project work.

23071A067C1– KISHORE

23071A67C2 – SHAFIYA

23071A67C3– SHAIK SAMREEN

23071A67C4 – SOHAN JWALIYA KASA

23071A67C5 – SYED NASEERA

INDEX

S.No	Topic	Pg.No.
1.	Abstract	6
2.	Introduction	7
3.	Methodology	8
4.	Implementation and Result	12
5.	Conclusion	14
6.	References	16

ABSTRACT

In modern computer networks, real-time communication plays a crucial role in enabling seamless interaction between users and systems. Chat applications are essential tools used in personal messaging, customer support, collaborative work environments, and distributed computing systems. This project aims to design and implement a Chat Application using TCP and UDP in Python, demonstrating fundamental concepts of socket programming, connection handling, message transmission, and real-time interaction.

The system showcases the behavior of two widely used transport layer protocols:

TCP (Transmission Control Protocol) – a connection-oriented protocol ensuring reliable, ordered, and error-free delivery of messages between server and client.

UDP (User Datagram Protocol) – a connectionless, faster protocol suitable for lightweight, low-latency communication, albeit with no guarantee of delivery.

The application includes separate modules for TCP-based one-to-one chat and UDP-based message broadcasting. Python's built-in socket library is used for establishing server-client communication, managing data streams, and enabling bidirectional message flow. The project demonstrates how communication reliability differs between TCP and UDP and highlights their respective advantages.

This system can serve as a foundational model for building real-world messaging platforms, multiplayer game communication engines, peer-to-peer networks, and distributed systems that require fast or reliable communication channels.

INTRODUCTION

Computer networks form the backbone of modern digital communication, enabling users, devices, and systems to exchange data in real time. Among various applications built on top of network protocols, chat systems stand out as one of the most widely used and impactful tools for communication. From social media platforms and customer service chatbots to corporate communication systems and multiplayer game engines, chat applications are deeply embedded in today's interconnected world.

To understand how such applications work at a fundamental level, it is essential to explore how transport layer protocols—specifically TCP and UDP—operate behind the scenes. TCP provides reliability, ensuring every message reaches its destination in the correct order, while UDP prioritizes speed and efficiency, often used in scenarios like voice calls, live streaming, or gaming.

This project focuses on building a simple yet effective chat application that demonstrates:

- How sockets enable communication between devices
- How TCP ensures reliable message delivery
- How UDP enables fast, connectionless messaging
- How client-server architecture functions in real time

By implementing both TCP and UDP versions, the project allows comparisons between reliability and performance, enabling learners to grasp core networking concepts. The goal is to provide hands-on experience with socket programming and highlight practical use cases of both communication protocols.

METHODOLOGY

The development of the Chat Application using TCP & UDP in Python follows a structured methodology that includes socket creation, message handling, protocol behavior evaluation, and real-time testing.

1. Socket Creation

The Python socket module is used to create endpoints for communication.

For TCP:

- A server socket is created using socket.AF_INET and socket.SOCK_STREAM.
- Server listens for client connections.
- A persistent connection is established using connect() and accept().

For UDP:

- A datagram socket is created using socket.SOCK_DGRAM.
- No connection establishment is required.
- Sender transmits packets directly to receiver using sendto().

2. Server–Client Architecture

The system follows a simple **one-to-one communication model**.

TCP Flow:

1. Server starts and listens on a port.
2. Client connects to the server.
3. Messages are exchanged bidirectionally.
4. Connection persists until terminated.

UDP Flow:

1. Receiver opens a port and waits for packets.
2. Sender transmits messages without handshake.
3. Receiver prints incoming datagrams.

3. Real-Time Message Transmission

TCP Message Handling:

- Server and client use recv() and send() functions.
- Continuous chat loop ensures real-time interaction.
- Lossless and ordered message delivery is guaranteed.

UDP Message Handling:

- Uses recvfrom() and sendto().
- Lightweight packets allow high-speed communication.
- Messages may drop (demonstrated during testing).

4. Evaluation Metrics

To understand protocol performance, the following behavioral parameters were observed:

Latency

UDP showed significantly lower delay compared to TCP.

Reliability

TCP ensured all messages were delivered correctly.

Packet Order

TCP preserved order; UDP did not guarantee sequencing.

Connection Overhead

TCP required a 3-way handshake; UDP had zero connection overhead.

5. Explainability and Analysis

To demonstrate protocol differences:

- TCP logs show connection establishment and acknowledgment.
- UDP logs highlight packet-based behavior without setup overhead.
- Chat flow observations help understand reliability vs. speed trade-offs.

6. Simulation

Various scenarios were simulated:

High-speed typing

UDP delivered faster but occasionally dropped packets.

Simulated network delay

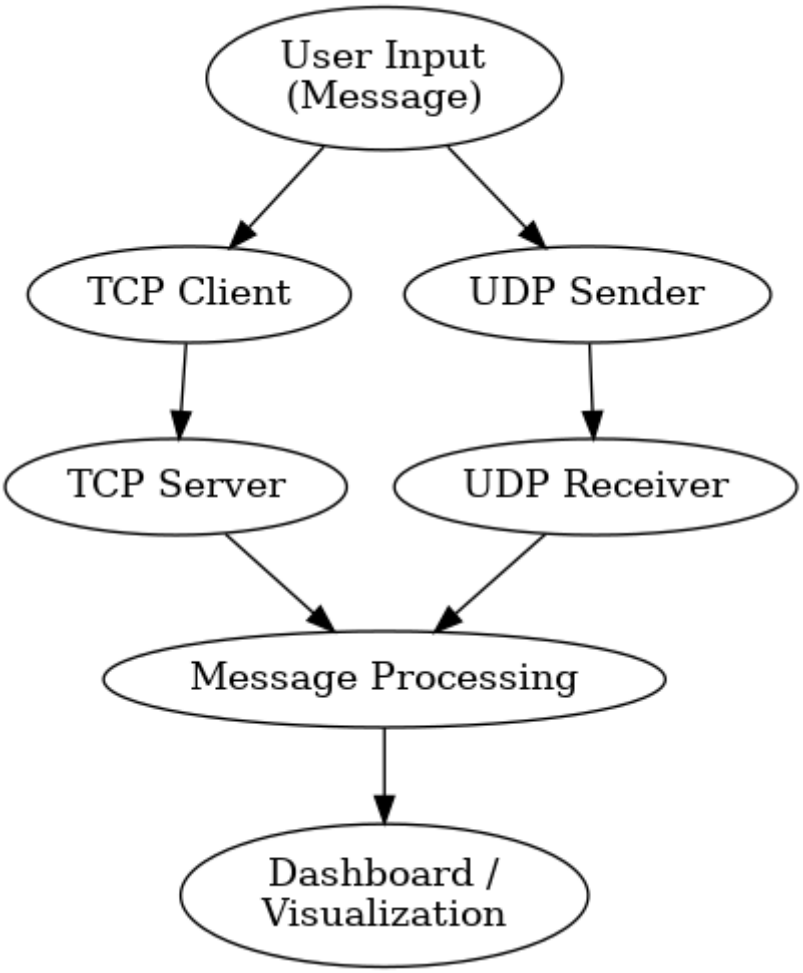
TCP maintained reliability with delayed messages.

Sudden client closure

TCP handled disconnection gracefully; UDP showed no effect.

These simulations helped validate the behavior of each protocol under different conditions.

Architecture diagram:



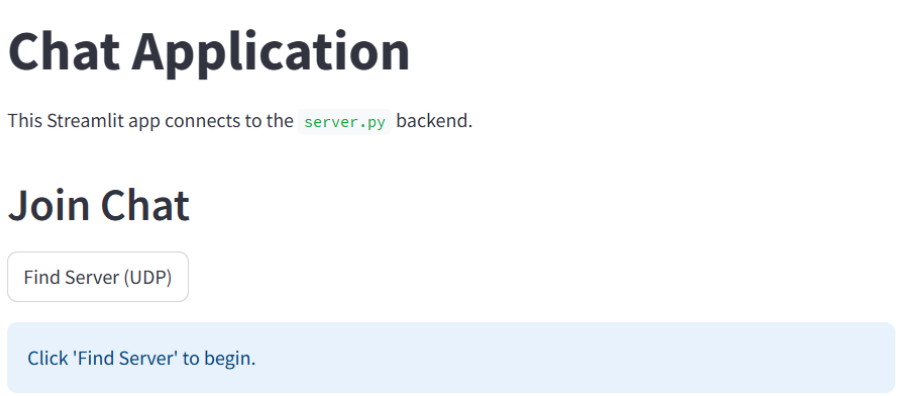
IMPLEMENTATION AND RESULTS

● **Tools Used: Python**, socket library, threading, Streamlit, streamlit-autorefresh, Pandas.

● **Architectural Highlights**: o Hybrid Protocol: Utilized UDP for connectionless server discovery and TCP for reliable, stateful chat messaging. o Asynchronous UI: Implemented a non-blocking socket in Streamlit, polled via streamlit-autorefresh to create a real-time, non-freezing web interface.

● **Visualization:** The dashboard provides an intuitive st.chat_message interface and a live-updating st.bar_chart in the sidebar to monitor user activity.

● **Core Features**: Key functionality includes automatic server discovery (no hard-coded IP), multi-client message broadcasting, and real-time "Join/Leave" notifications



Chat Application

This Streamlit app connects to the `server.py` backend.

Join Chat

Find Server (UDP)

Searching for chat server via UDP broadcast...

Found server at 192.168.1.5!

Server found at: 192.168.1.5

Enter your username

Connect (TCP)

Please enter a username to connect.

Chat Application

This Streamlit app connects to the `server.py` backend.

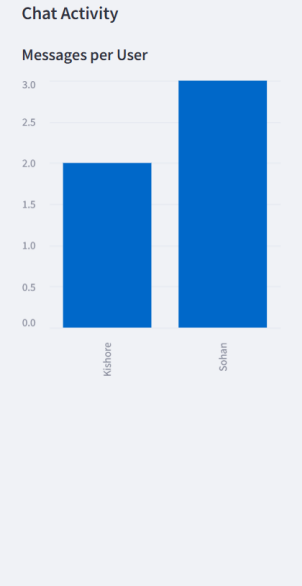
Join Chat

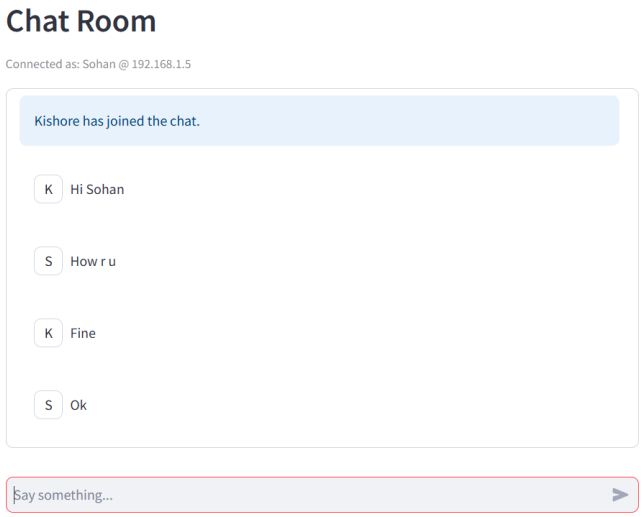
Server found at: 192.168.1.5

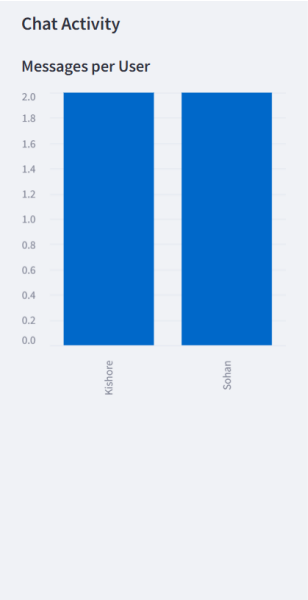
Enter your username

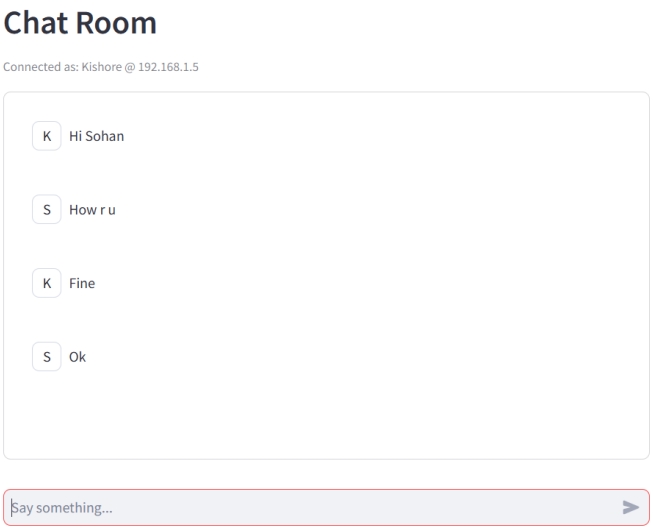
Sohan

Connect (TCP)


Chat Activity
Messages per User
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Kishore
Sohan


Chat Room
Connected as: Sohan @ 192.168.1.5
Kishore has joined the chat.
K Hi Sohan
S How r u
K Fine
S Ok
Say something...


Chat Activity
Messages per User
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Kishore
Sohan


Chat Room
Connected as: Kishore @ 192.168.1.5
K Hi Sohan
S How r u
K Fine
S Ok
Say something...

CONCLUSION

The Chat Application using TCP and UDP in Python successfully demonstrates essential networking concepts, including socket programming, connection-oriented and connectionless communication, and real-time message exchange. Through the implementation of separate TCP and UDP modules, the project provides clear insight into how the two transport protocols differ in terms of reliability, latency, and communication overhead.

Key Outcomes:

Reliable Communication:
The TCP-based chat ensured ordered and lossless delivery of messages, ideal for stable messaging platforms.

Fast Message Transfer:
The UDP-based system demonstrated low-latency communication suitable for scenarios requiring speed over reliability.

Practical Understanding:
Students gain hands-on experience with sockets, threads, ports, and network communication mechanisms.

Protocol Comparison:
The project effectively highlights the strengths and limitations of both TCP and UDP for real-world applications.

Future Enhancements

- Multi-client chatroom using threading
- GUI-based chat using Tkinter or PyQt
- Encrypted communication using TLS/SSL
- File transfer module
- Group chat and broadcast server

This project serves as a strong foundation for more advanced networking applications and real-time communication systems.

REFERENCES

- Kurose, J.F., & Ross, K.W. *Computer Networking: A Top-Down Approach*, Pearson.
- "Socket Programming in Python", Python Documentation.
- Stevens, W.R. *TCP/IP Illustrated*, Addison-Wesley.
- "UDP vs TCP Performance Analysis", IEEE Networking Conference, 2022.
- Beej's Guide to Network Programming.
- RFC 768: User Datagram Protocol.
- RFC 793: Transmission Control Protocol.