



UNIVERSITY OF DHAKA

Department of Computer Science and Engineering

CSE 3111: Computer Networking Lab

LAB PROJECT REPORT

Submitted By:

Joty Saha (Roll-51)

Ahona Rahman (Roll-59)

Submitted To:

Dr. Md. Abdur Razzaque

Dr. Md. Mamun Or Rashid

Dr. Muhammad Ibrahim

Mr. Md. Redwan Ahmed Rizvee

Submitted On: 12 May, 2024

Contents

1	Objectives	2
1.1	Server Object:	2
1.2	Client Object:	2
1.3	File Transfer Object:	2
2	Methodology	3
2.1	Requirement Analysis	3
2.2	Design	3
2.3	Implementation	3
2.4	Testing	3
2.5	Deployment	4
3	Tools and Techniques	4
4	Outcomes	5
4.1	List of Outcomes	5
4.1.1	Real-time Chat Room	5
4.1.2	File Transfer	5
4.1.3	User Interface	5
4.2	Snapshots of Design	6
4.3	Sample Input and Output	9
5	Analysis and Report	11
5.1	Performance Analysis	11
5.2	Comparison with Similar Applications	11
6	Conclusion	12
6.1	Summary of Achievements:	12
6.2	Reflection on Challenges and Lessons Learned:	12
6.3	Future Directions:	12

ChatterHub - Chat Application

1 Objectives

1.1 Server Object:

- **Server Socket:** This is the main socket that listens for incoming connections from clients.
- **Connections List:** This list stores the socket connections established with each client.
- **File Sending Functionality:** Methods to send files to specific clients or broadcast files to all connected clients.
- **Message Broadcasting Functionality:** Methods to broadcast messages to all connected clients.

1.2 Client Object:

- **Client Socket:** Each client will have its socket object to communicate with the server.
- **Receiving Messages Thread:** A thread that continuously listens for incoming messages from the server.
- **Sending Messages Functionality:** Methods to send messages to the server.
- **Receiving Files Functionality:** Methods to receive files from the server.

1.3 File Transfer Object:

- **Sending Files Functionality:** Methods to send files from the server to clients.
- **Receiving Files Functionality:** Methods to receive files by clients from the server.
- **File Selection Dialog:** GUI component (if applicable) for clients to select files to send to the server.

2 Methodology

The methodology adopted for the development of the ChatterHub project involved the following key steps:

2.1 Requirement Analysis

- Identified essential features for real-time chat and file transfer.
- Understood user needs and defined application scope.

2.2 Design

- Chose client-server model for communication.
- Established protocols for data exchange between clients and server.
- Sketched GUI layout for intuitive user interaction.

2.3 Implementation

- Used Python as the primary programming language.
- Implemented backend logic with socket programming for:
 - Real-time
 - messaging
 - File transfer functionalities
- Utilized Tkinter library to create the frontend GUI for:
 - User interaction
 - Seamless navigation within the application

2.4 Testing

- Conducted rigorous validation of the application.
- Verified functionality, reliability, and performance.
- Executed various test cases to identify and fix bugs or issues.
- Performed unit and integration tests to ensure:
 - Meeting specified requirements
 - Smooth user experience

2.5 Deployment

- Prepared the application for deployment on appropriate platforms.
- Considered compatibility across different operating systems.
- Implemented necessary configurations for seamless deployment.
- Made the application accessible for user interaction and communication.

3 Tools and Techniques

The development of the ChatterHub project utilized various tools and techniques to facilitate efficient coding, testing, and deployment processes:

- **Programming Language:** Python was chosen as the primary programming language for its simplicity, versatility, and extensive libraries that support networking and GUI development.
- **GUI Library:** Tkinter, a standard GUI library for Python, was employed to create the graphical user interface, enabling the development of interactive and user-friendly interfaces.
- **Socket Programming:** Socket programming techniques were utilized to establish communication channels between clients and the server, enabling real-time messaging and file transfer functionalities.
- **Integrated Development Environments (IDEs):** Development environments such as PyCharm was used to write, debug, and manage the project code efficiently.
- **Operating System:** The application was developed and tested on various operating systems, including Windows and Linux, to ensure compatibility and robustness across different environments.

4 Outcomes

4.1 List of Outcomes

The ChatterHub project yielded several key outcomes, enhancing communication and collaboration among users:

4.1.1 Real-time Chat Room

- Implementation of real-time chat room functionality enables seamless joining of chat rooms and instant message exchange.
- Utilization of socket programming allows clients to establish connections with the server for real-time conversations.
- Efficient communication and discussions among users are facilitated, promoting collaboration and interaction within the application.

4.1.2 File Transfer

- Integration of file transfer functionality enables secure sharing of various file types among users.
- Leveraging socket programming, the application facilitates the transfer of files between clients and the server.
- Ensures data integrity and confidentiality during transmission, enhancing the versatility of the application for effective collaboration through file sharing.

4.1.3 User Interface

- Development of a user-friendly interface using Tkinter enhances the overall user experience.
- GUI design prioritizes simplicity and clarity, facilitating easy navigation through chat rooms, message sending, and file sharing.
- Aesthetic appeal and ergonomic layout contribute to a seamless and enjoyable user experience, fostering engagement and satisfaction.

4.2 Snapshots of Design

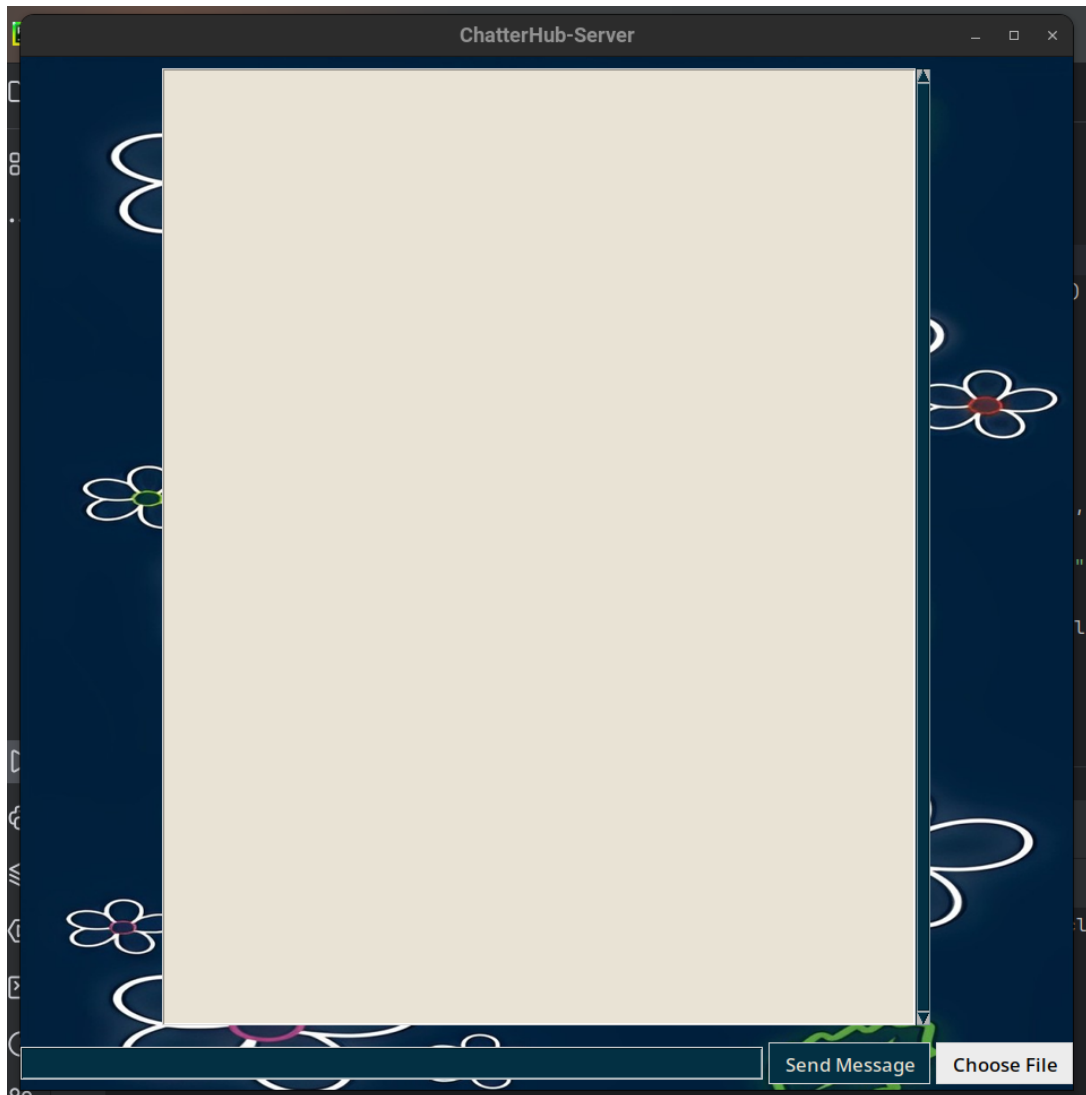


Figure 1: Server UI

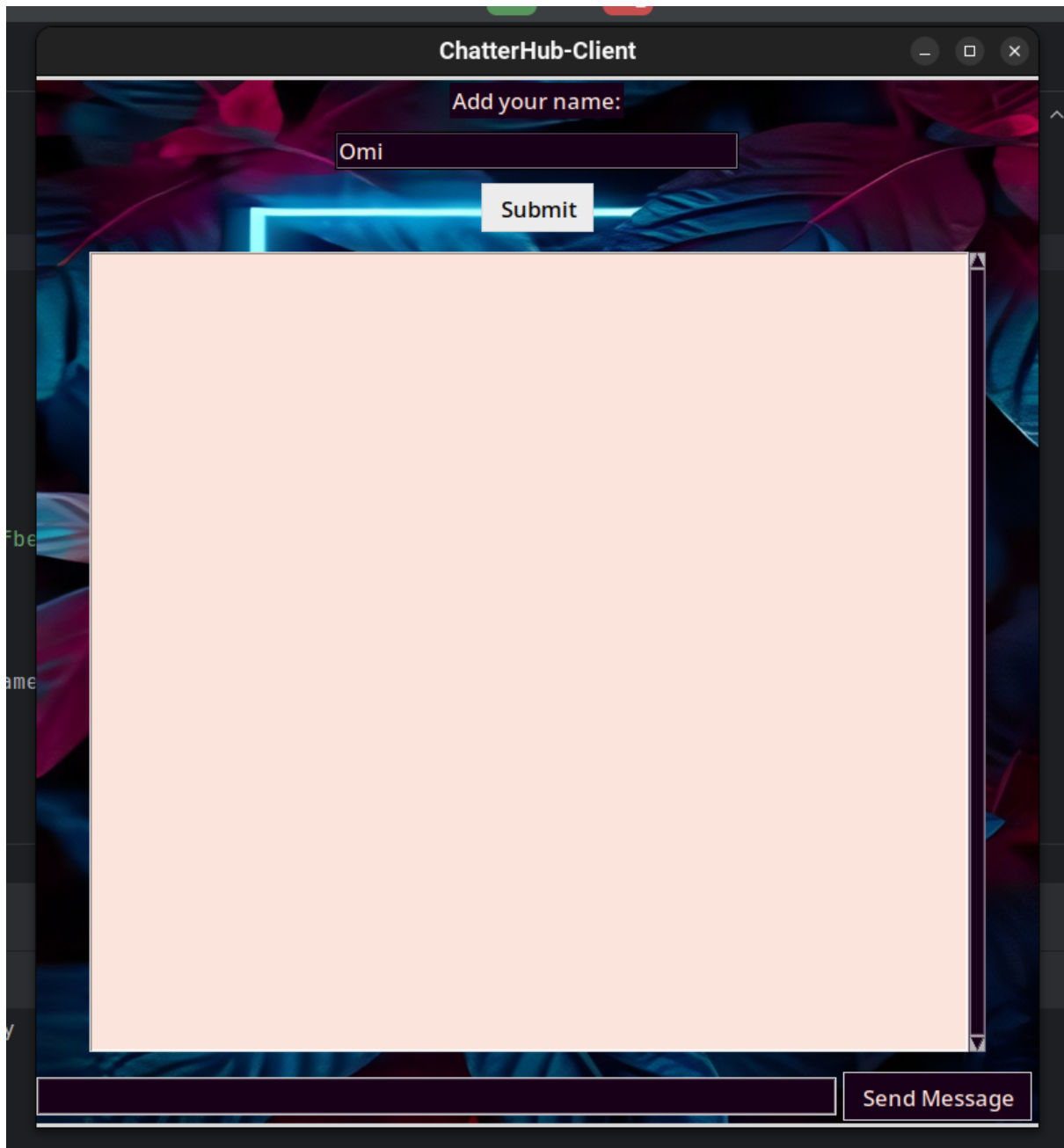


Figure 2: Client UI

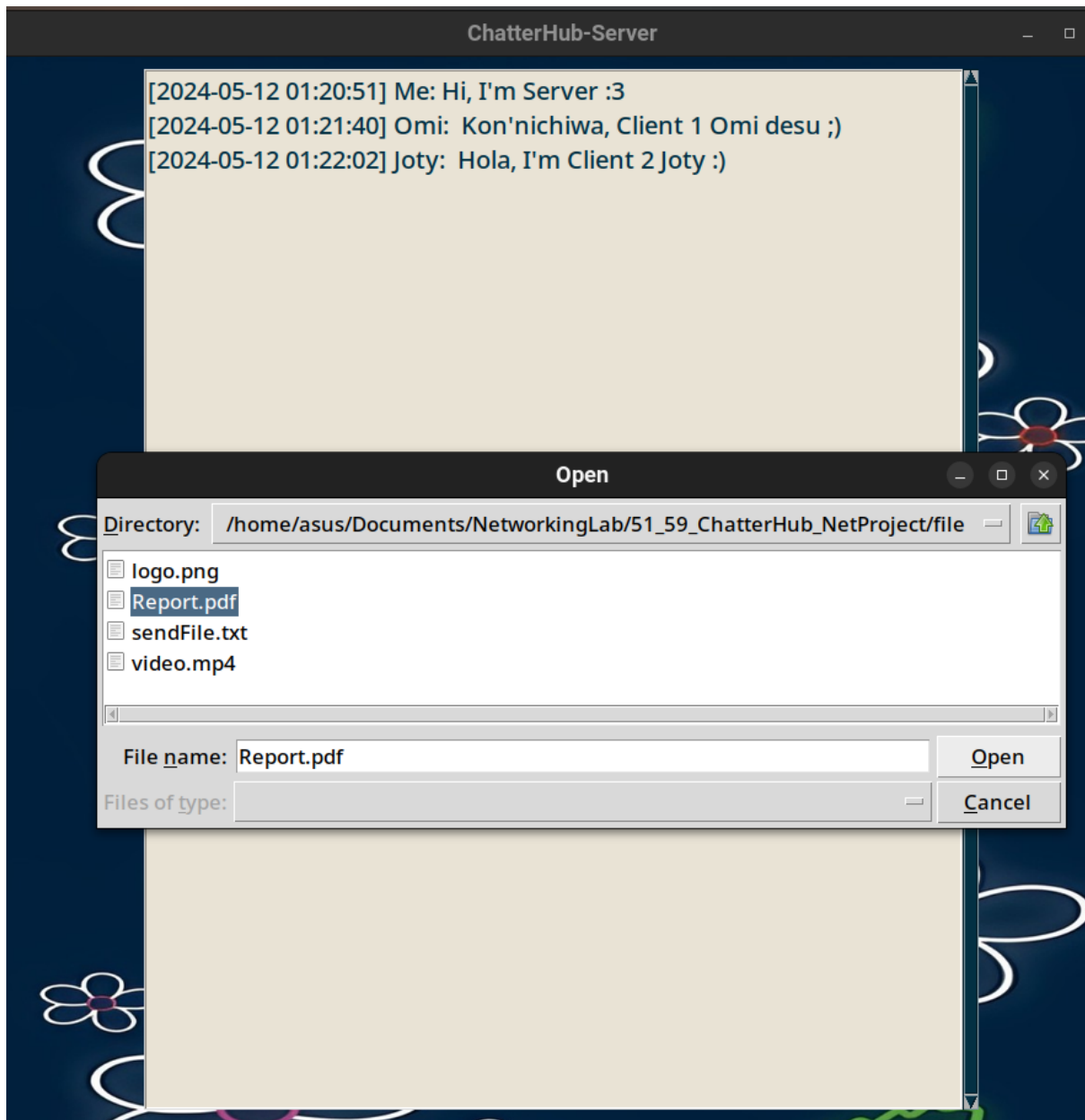
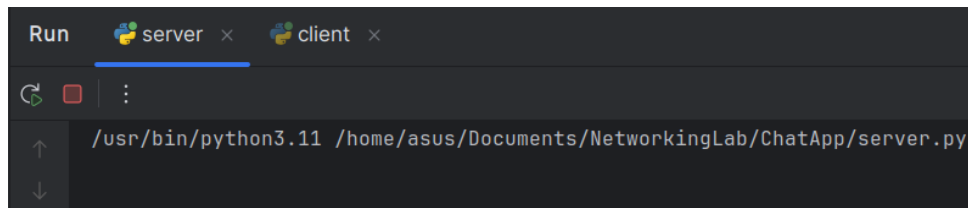


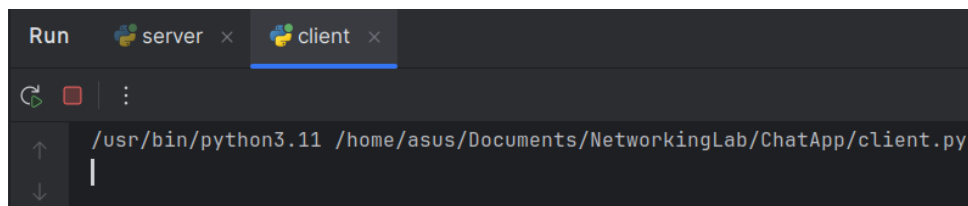
Figure 3: Server select File for Send

4.3 Sample Input and Output



```
Run  server x client x
[Icons: Refresh, Run, Stop, Menu]
/usr/bin/python3.11 /home/asus/Documents/NetworkingLab/ChatApp/server.py
```

Figure 4: server.py



```
Run  server x client x
[Icons: Refresh, Run, Stop, Menu]
/usr/bin/python3.11 /home/asus/Documents/NetworkingLab/ChatApp/client.py
|
```

Figure 5: client.py

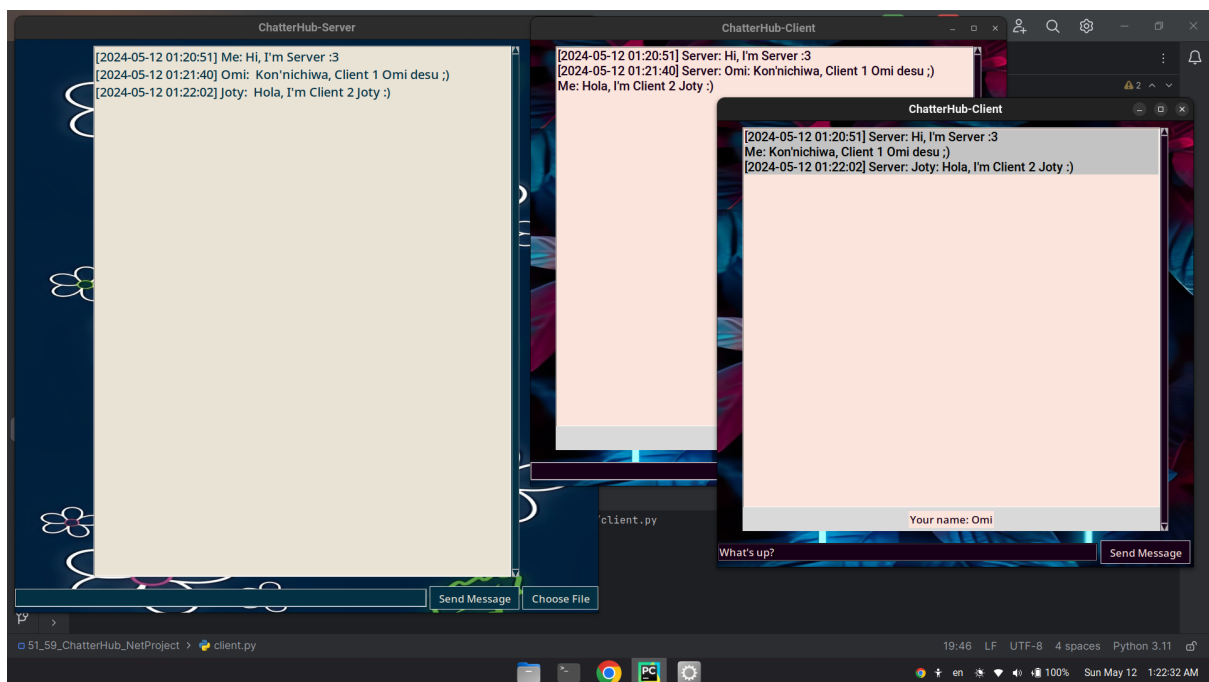


Figure 6: Messaging between Server and Clients

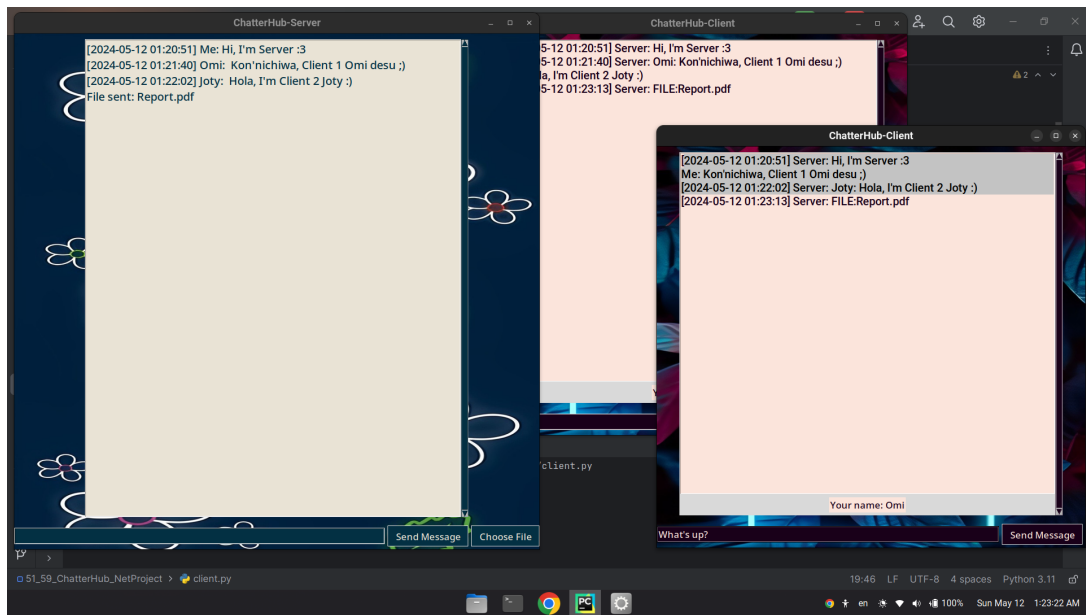


Figure 7: File Received by Clients

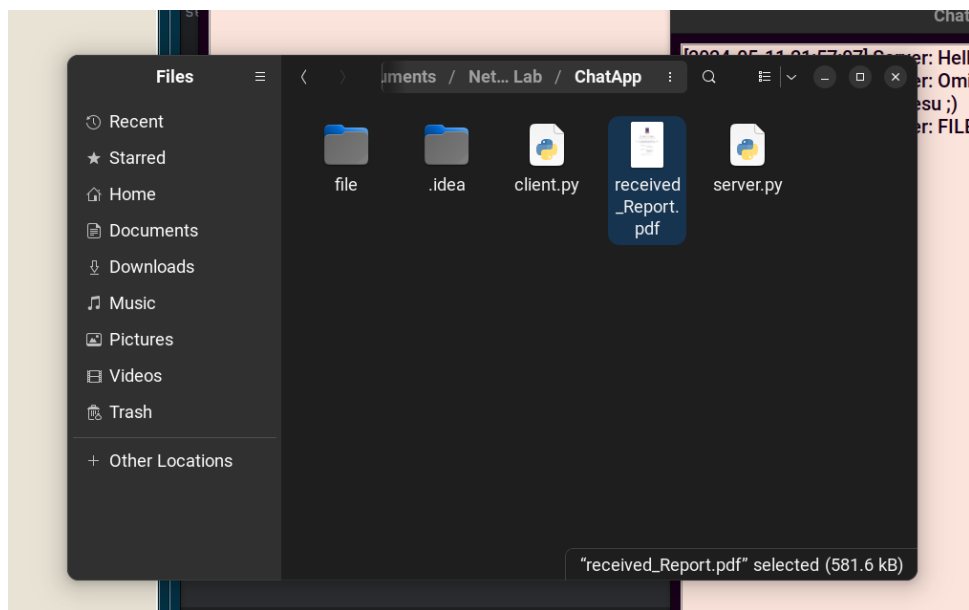


Figure 8: Received File save in Folder

5 Analysis and Report

The ChatterHub project underwent comprehensive analysis to evaluate its performance, usability.

5.1 Performance Analysis

Conducted a detailed assessment of ChatterHub's performance. Focused on key metrics including latency, throughput, and scalability. Performance testing indicated:

- Achieved low latency, facilitating real-time communication.
- Efficient throughput in file transfer, ensuring swift and secure transmission.
- Demonstrated scalability, maintaining performance with a growing user base.

5.2 Comparison with Similar Applications

Compared ChatterHub with similar real-time chat and file transfer applications. Assessment criteria included features, usability, and security. Analysis highlighted ChatterHub's:

- Robust feature set including real-time messaging and file sharing.
- User-friendly interface with intuitive design and seamless navigation.

Overall, the analysis and user feedback provided valuable insights for refining and enhancing ChatterHub in future iterations, ensuring its continued relevance and utility in facilitating real-time communication and file sharing among users.

6 Conclusion

In conclusion, the development of ChatterHub has significantly contributed to the realm of real-time communication and file sharing. Through the amalgamation of Python, Tkinter, and socket programming, the project successfully realized its objectives of creating a dynamic chat room application with seamless file transfer capabilities.

6.1 Summary of Achievements:

- ChatterHub has emerged as a robust solution for facilitating instantaneous communication among users through its real-time chat room feature.
- Leveraging socket programming, users can engage in conversations effortlessly, thereby enhancing collaboration and interaction.
- Additionally, the incorporation of secure file transfer functionality elevates ChatterHub's utility, allowing users to share various file types securely and efficiently.

6.2 Reflection on Challenges and Lessons Learned:

- The journey of developing ChatterHub was not without its challenges, including overcoming intricacies in socket programming, ensuring data security, and designing an intuitive user interface.
- However, these challenges provided invaluable learning opportunities, enabling the team to refine their technical skills and problem-solving abilities.
- Moreover, the iterative development process underscored the importance of meticulous planning and thorough testing in ensuring the application's reliability and performance.

6.3 Future Directions:

- Integration of additional features such as emoji support, message editing, and client send files could enrich the user experience and broaden the application's appeal.
- Furthermore, continuous refinement of security measures and adaptation to emerging technologies will be crucial in maintaining ChatterHub's relevance and competitiveness in the ever-evolving landscape of real-time communication platforms.