

**National University of Computer and Emerging**

**Sciences, Karachi**

**FAST School of Computing**

**CS3001-Computer Networks, Spring 2023**

**Project Report**

**Project title: File Transfer Application**



## **Group members** Muhammad Ali Khan (20P-0553) **Instructor: Mr Ali Naseer Shah**

## Contents

[Proposed project description: 3](#_Toc4595)

[Plan of Work: 3](#_Toc4596)

[Server Implementation: 4](#_Toc4597)

[Client Implementation: 5](#_Toc4598)

[Outputs: 6](#_Toc4599)

[Conclusion: 8](#_Toc4600)

[References 8](#_Toc4601)

### 

# Proposed project description:

File transfer application with multiple servers and multiple clients is a software application that allows server and clients to transfer files within each other over a network. The application should allow multiple clients to connect to multiple servers, each with their own set of files to transfer. Clients can upload documents which can be viewed by the server and vice versa.

Following will objectives be implemented in this tutorial:

•Multiple Server Support: The application should support multiple servers to allow clients to connect and access their files.

•Multiple Client Support: The application should support multiple clients, each of which can connect to any of the available servers and access the files on those servers.

* How to Code a client
* How to Code a Server
* File Management: The application should provide users with an interface to manage their files, including uploading, downloading,
* Connecting Multiple Clients to a Server

# Plan of Work:

WEEK 1 - Creating a Client Using Sockets

WEEK 2 - Creating a Server

WEEK 3 – Creating multiple clients and servers.

WEEK 4 – Sending & Receiving Information from Server

WEEK 5 - Uploading and downloading of data/files within servers and clients.

***Functionality*:** which functional features (from the itemized list at the end of the previous section) each one will contribute.

1. Client-Server Architecture: The application should be designed using a client-server architecture where the server manages the file sharing and communication between clients.
2. Connection Establishment: The app should establish a connection between the client and server when transferring files.
3. Data Transfer: The app should use protocols to transfer data between the client and server.
4. Packetization: The app should packetize the data to ensure reliable transmission over the network.
5. Protocol Specification: The app should have a protocol specification that defines the format and structure of the data packets exchanged between the client and server.
6. Concurrent Connections: The app server should be able to handle multiple concurrent connections from different clients.

# Server Implementation:

File transfer application with multiple servers and multiple clients involved designing and implementing a server-side system that enabled clients to connect to multiple servers and access their files. The server architecture was designed to support multiple servers and multiple clients [1]. A distributed architecture was used, where each server was responsible for a specific set of files, and clients could connect to any server to access the files they needed.

File storage and management: The server had a file storage system that allowed users to upload, download and manage files. The files were stored in a directory structure that was organized based on the user, date, or any other relevant criteria. The server also provided features like file search, file permissions, and versioning [2].

# Client Implementation:

File transfer application with multiple servers and multiple clients project involved designing and implementing a client-side system that enabled users to connect to multiple servers and access their files [1].

User interface: The client-side system had an intuitive user interface that allowed users to easily navigate the system, go through different department/courses e.g., Artificial Intelligence, Software Engineering Computer Networks and manage their files. The interface provided features like file upload, download, delete, as well as options to connect to different servers and manage user accounts.

Overall, implementing the client-side of a file transfer application with multiple servers and multiple clients required a user-friendly and scalable system that was designed to handle multiple connections and file transfers [3]. The system provided features like an intuitive user interface, support for different file transfer protocols, file management and scalability.

# Outputs:

* After clicking on Client.py, you will be presented with this GUI.

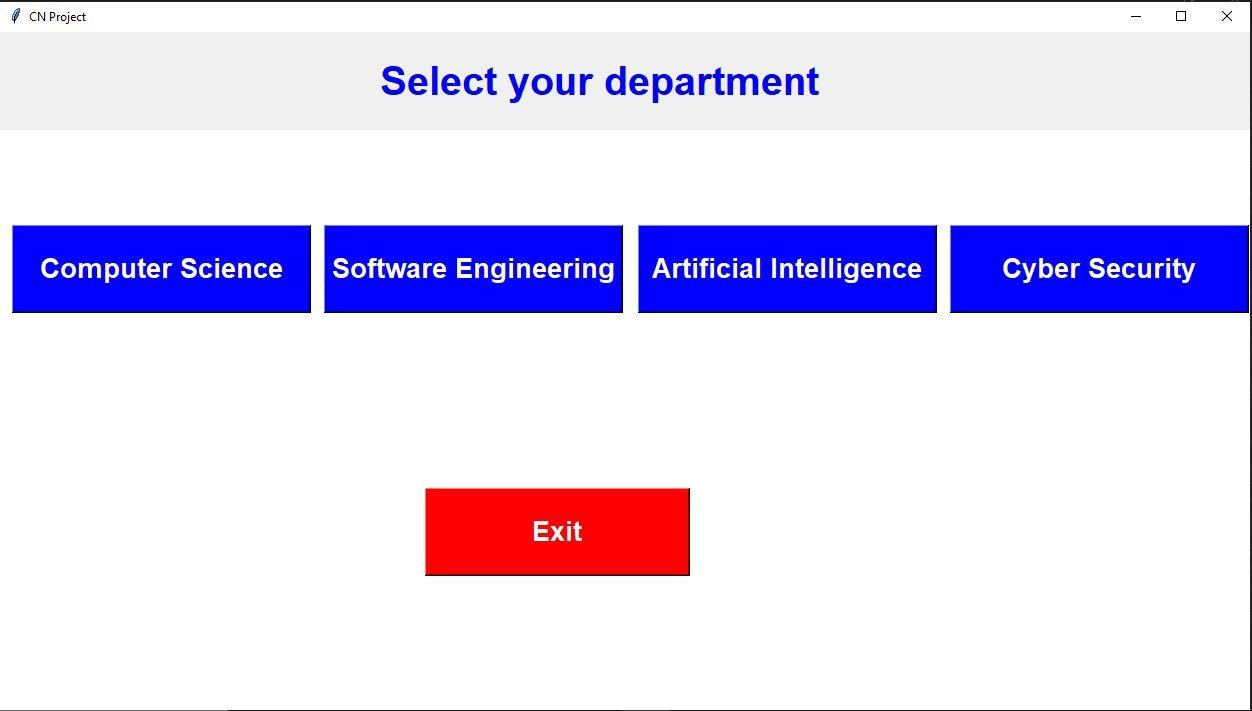


Figure 1.1

* After clicking on sever button, you will be presented with this prompt box asking for donwloading the file.

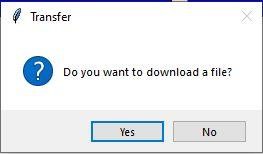


Figure 1.2

* After choosing the option ‘yes’, you will be directed to file storage of the particular server you chose to open and you will be able to access the files of that server.

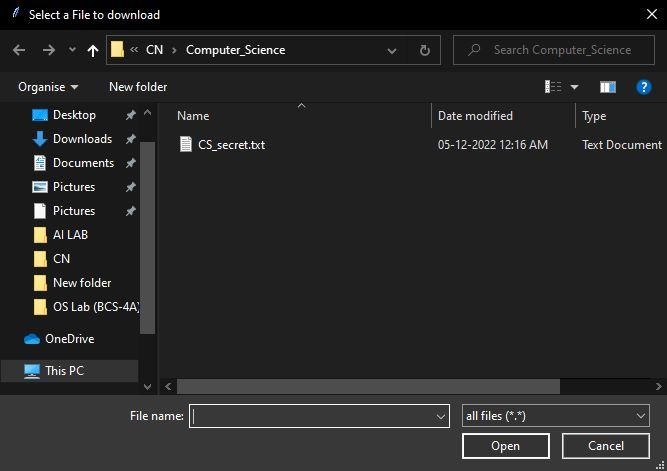


Figure 1.3

* If you choose the option ‘no’, you will be asked to upload any file from the computer to the server just like the example below.

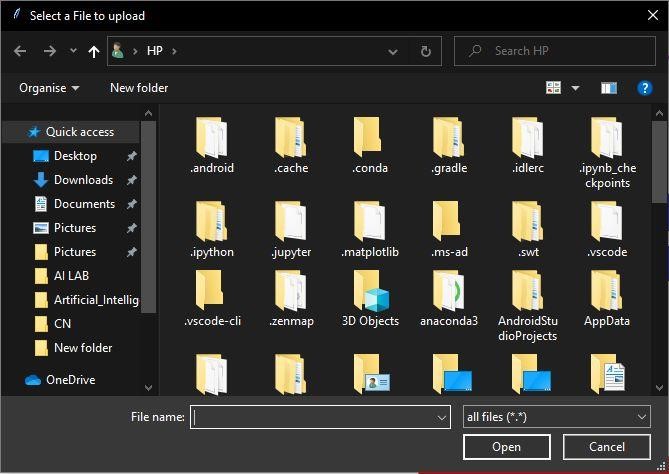


Figure 1.4

# Conclusion:

The file transfer application with multiple servers and multiple clients project was a challenging but rewarding undertaking that resulted in a functional and efficient file transfer system. The project involved designing and implementing a server-side system that enabled multiple clients to connect and access files, as well as a client-side system that allowed users to connect to multiple servers and manage their files. The project was successful in achieving its objectives, providing a functional and efficient file transfer system with multiple servers and multiple clients. The project also provided an opportunity to develop technical skills in network programming, security, and system design, as well as collaboration and communication skills through teamwork. The project could be further improved by adding more advanced features, such as support for cloud storage, real-time collaboration, and machine learning-based security mechanisms.

# References

1. K. P. M. S. D. T. M. T. a. B. W. A. Demers, ""The Bayou Architecture: Support for Data Sharing Among Mobile Users,"," *1994 First Workshop on Mobile Computing Systems and Applications,,* pp. pp. 2-7,, 1994.
2. Q. L. Z. H. J. L. F. Y. Jianping Zhang, "A multi-server information-sharing environment for cross-party collaboration on a private cloud,," *Automation in Construction,* vol. 81, pp. 180-195, 2017.
3. "The Farsite project: a retrospective," *ACM SIGOPS Operating Systems Review,* vol. 41, pp. pp. 17-26, 2007.
4. R. J. R. D. A. e. a. Dowell, "The Distributed Annotation System.," *BMC Bioinformatics 2, 7,* (2001)..