# Project Report
## Computer Science Department

# COMPUTER NETWORKS

Submitted by

**Asfand Yar**

Lecturer Department of Computer Sciences Faculty of
ICT, BUITEMS, Quetta.

Lab Instructor

Lecturer, Department of Computer Science,
Faculty of ICT, BUITEMS, Quetta.

**Session Spring – 2024**

**BALOCHISTAN UNIVERSITY OF IT, ENGINEERING & MANAGEMENT SCIENCES, QUETTA.**

# CERTIFICATE

This is certified that **Asfand Yar** and _____ bearing CMS ID _____ and _____ has successfully completed the Project Report of Computer Network in his **4th semester** of **(BS) Computer Science**, **Spring 2024** under the supervision of his/her lab instructor _____

**BALOCHISTAN UNIVERSITY OF IT, ENGINEERING & MANAGEMENT SCIENCES, QUETTA.**

# Table of Contents

Table of Contents 3

**1. Introduction** 4

**2. Project Objectives** 4

**3. Methodology** 5

3.1 Development Approach 5

**4. Algorithms and Protocols** 6

4.1 Sharing Screen Algorithm 6

4.2 File Transfer Protocol 6

4.3 TCP/IP (Transmission Control Protocol/ Internet Protocol) 6

**5. Implementation Detail** 7

5.1 User Interface 7

5.2 Network Communication: 7

5.3 Password Management: 8

**6. System Architecture** 8

6.1 Client-Side Components 8

6.2 Server-Side Components 9

**Conclusion** 10

# 1. Introduction

With the world getting more and more connected, access to a computer from a location other than its actual location is a requirement that has continuously gained prominence. Remote desktop applications allow users control over their systems, troubleshooting of issues, and copying of files, all for convenience and enhanced productivity from a different geographical location. This project aims to develop a remote desktop application that will allow the user to control, from a different device, the running of a computer over a network. It provides screen sharing, mouse and keyboard control, secure file transfer, and all other features that provide a comprehensive tool in remote system management.

The graphical user interface of the Remote Desktop Application will be user-friendly and easily workable, developed with the help of Java Swing. Also, the application concerns secure communication between the client and the server; thus, it provides a robust solution to access remote desktops. This report documents the objectives, methodology, system architecture, algorithms, and implementation details of the project, portraying the technical aspects and challenges faced during its development process.

# 2. Project Objectives

The Remote Desktop Application project seeks to achieve the following major goals:

- **Remote Control:** Develop an application that can allow any remote user to have control over the computer's mouse and keyboard in order to access the entire system as though physically accessible.
- **Screen Sharing:** Real-time screen sharing of the remote computer's screen on your client device while providing minimal latency and great image transfer quality.
- **Secure File Transfer:** Enable file transfer securely between the client and the server, making it possible for users to send and receive files securely from the network.
- **User Authentication:** Integrate password protection to ensure that the remote desktop can only be accessed and operated by authorized users.
- **User-Friendly Interface:** A graphical user interface that must be user-friendly and visually appealing to access features and functions easily.
- **Scalability and Robustness:** Develop a scalable and solidly designed application, including features to handle many connections at once, and to work efficiently under varying network conditions.
- **Network Communication:** Develop efficient protocols of network communication between a client and a server for the smooth exchange of data with least latency and maintaining the integrity of the data.

# 3. Methodology

## 3.1 Development Approach

The Remote Desktop Application followed a structured and iterative strategy of development that allowed adequate design, implementation, and testing of functionality. The major aspects of the development approach are outlined below:

- **Technology Stack**
  - **Java Swing**: The GUI of the application was developed using Java Swing. Java Swing is a part of Java Foundation Classes (JFC). It has a very extensive set of components to create an interactive and beautiful user interface.
  - **Java Standard Libraries:** The standard libraries of Java were used to perform network communications; the standard libraries enable reliable data transfer between the client and the server.

- **Iterative Development**
  - The methodology used was iterative development form, where the application would be developed incrementally through multiple iterations. Continual testing and refinement of each functionality has gone in line-by-line, which assists in catching all the bugs at an early stage of development itself.
  - In this model of iteration, each iteration clearly had a defined set of features to address. The first iteration took care of the most central features that included network communication and the GUI basic elements. Advanced features, like screen sharing, remote control, and file transfer, were dealt with later.

- **Requirement Analysis End**
  - The development process began with Project Requirements followed by a complete analysis. Key requirements identified at that period included remote control, sharing screens, secure file transfer, and authentication of users. Eventually, it formed the base through which designing the application architecture and functionality would follow.

- **Design and Implementation:**
  - **User Interface Design:** The user interface shall be made easy to use, having simplicity in view. Buttons for setting the server password, initialization of connecting to the desktop and files transfer were conspicuously placed.
  - **Network Communication:** Network communication was achieved by using Socket and Server Socket classes in Java that facilitated the client and the server to communicate reliably and exchange data between themselves.
  - **Security Measures:** Password protection was thereby added to ensure secure access to the remote desktop. Further security measures, like data encryption, can always be incorporated in future versions.

# 4. Algorithms and Protocols

The Remote Desktop Application employs several key algorithms and protocols, which are the components driving features like screen sharing and file transfer. We present here a description of the algorithms and protocols used in the application implementation.

## 4.1 Sharing Screen Algorithm

This very impressive screen-sharing feature displays the server's screen real-time to the client. This process is actually done by a server through capturing its screen, then compressing the captured images, transmitting them across the network, and finally displaying them on the client's interface. The process is shown below:

- **Capture Screen:** This server captures the current screen at regular intervals. It is normally done with the aid of a Robot class in Java enabling the capturing of screen images as Buffered Image objects.
- **Compress Image:** The captured screen image is then compressed, which minimizes the amount of data in network transmission. This can be done by using any of the many algorithms for image compression, usually normally based on either JPEG or PNG encoding. Compression reduces the size of the image file at the cost of possibly reducing image quality.
- **Send Image**: The compressed image is then transmitted over the network to the client. Once again, Java network libraries—in particular, the Socket and output stream classes—are used to establish a connection and transmit the data.
- **Display Image:** The receiving end uses the received image data through the Input Stream class and reassembles the image. The image is then displayed on the client user interface with the aid of Java Swing components such as JPanel or JLabel.

## 4.2 File Transfer Protocol

- The file transfer protocol provides a safe, efficient way of transferring files between the client and the server. The protocol provides for both the sending and receiving of files, hence ensuring files are always reliably transmitted and accurately reconstructed.

## 4.3 TCP/IP (Transmission Control Protocol/ Internet Protocol)

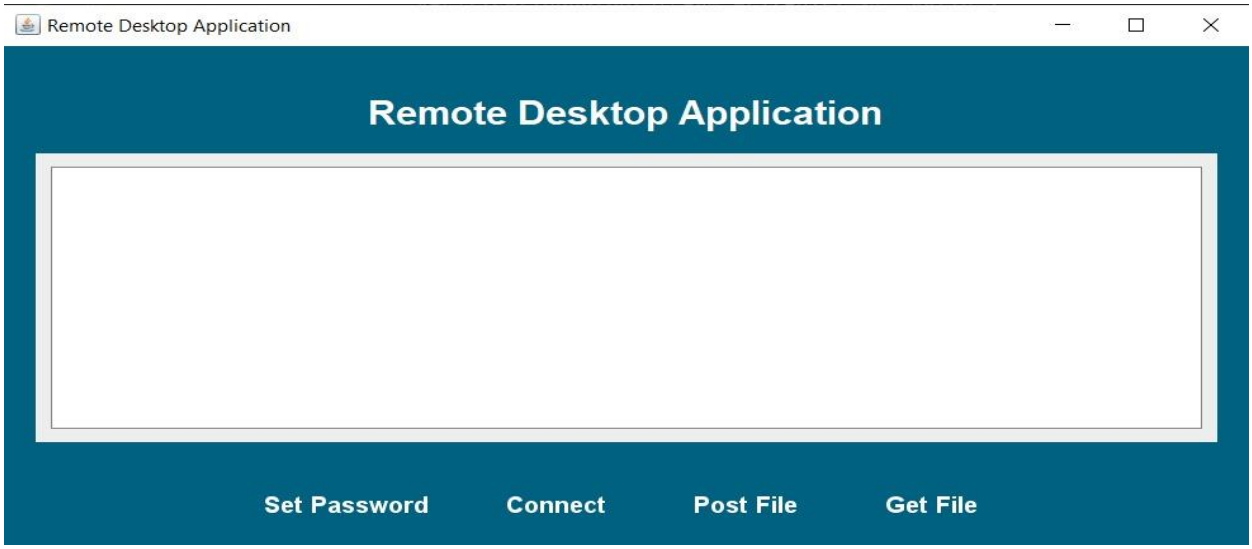
- **Overview:** TCP/IP is the set of basic communications protocols intertwined with devices on a network and connecting them to the Internet. It describes end-to-end data communication and details exactly how data must be packetized, addressed, transmitted, routed, and received.

- **Implementation Details:** The application will utilize Java's Socket class to create a TCP connection through which data is exchanged. It will utilize the Server Socket on the server side for listening to clients for a request to connect. After the establishment of the connection, the data will be sent over the TCP link using input and output streams.

# 5. Implementation Detail

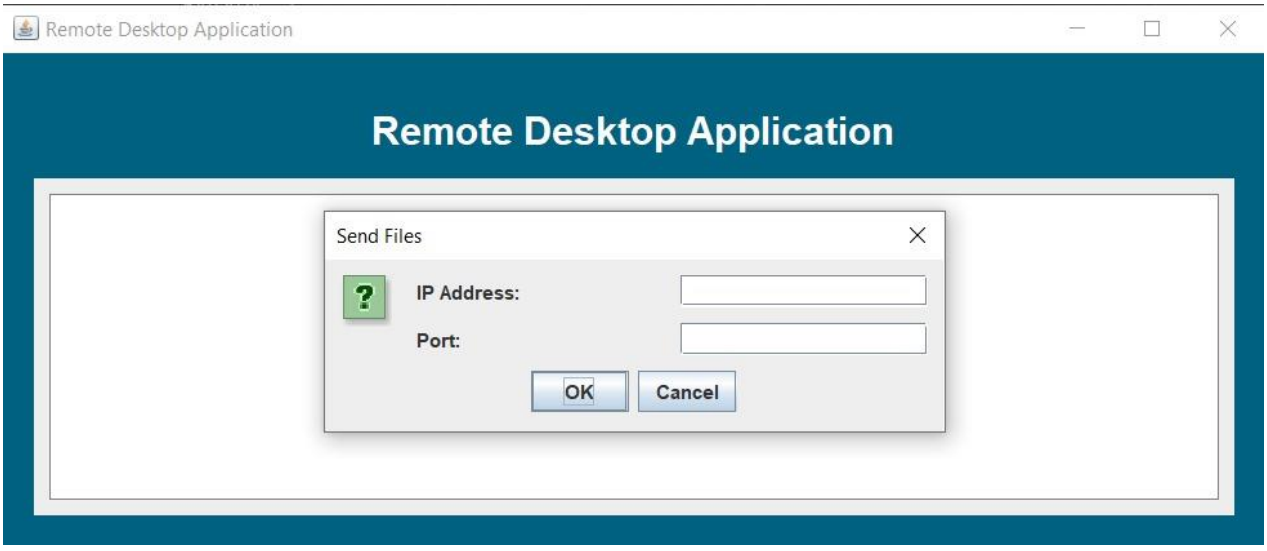
## 5.1 User Interface

The user interface is a JFrame with buttons for different functionalities. The buttons, in turn, are styled with respective Action Listeners that capture events of user interaction.



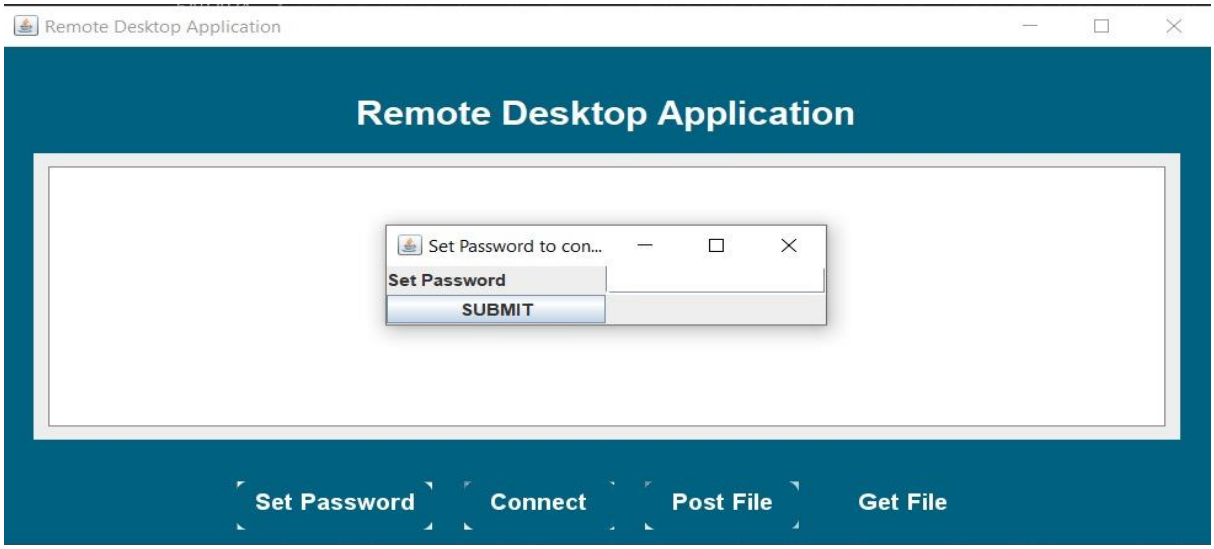

## 5.2 Network Communication:

Communication across the network in Java has been implemented using Socket and Server Socket classes. A client is connected to a server using an IP address and a port.



### 5.3 Password Management:

The server password is set using a dialog box and is stored encrypted. It will be required when the client connects before access is provided.




## 6. System Architecture

The Remote Desktop Application is designed for fast interaction between the client and the server to support strong, secure remote access. There are two major components in the architecture: the Client and the Server. Each of the components is responsible and has different functionalities, which are explained below.

### 6.1 Client-Side Components

The client-side components form the user-friendly interface and hence, all the actions against the server. Major client-side components are as follows:

1. **User Interface**
   - **Description:** The UI is developed using Java Swing that provides the users with a user-friendly, more intuitive, and aesthetically pleasing layout. Further, this component includes buttons for setting the server password, connecting to the desktop, sending files, and receiving files.
   - **Functionality:** The UI empowers the user to do a lot of different activities: connect to the remote desktop, initiate file transfers, and configure secure access. Buttons and input fields are thoughtfully designed to make these tasks easy and handy.

**2. Network Communication:**

- **Description:** This network communication module is responsible for establishing a connection and maintaining it between the client and the server. It also sends and receives data, like screen captures and file transfer data.
- **Drawback:** The component will use java's Socket class for connecting to the server and the class Output Stream and the class Input Stream for sending and receiving the data. All data shall be sent and received reliably and efficiently, where TCP/IP protocols will deal with the communication properly.

## 6.2 Server-Side Components

The server-side components are developed to handle the core functionality that enables remote desktop access and file transfer capabilities. The main components on the server side include the following:

**1. Password Management**

**Description**: Through the password management module, one could set a password for and verify it against the server. This is necessary for preventing unauthorized access to the server.
**Functionality:** This component enables a password to be set on the server and checks all the incoming connections against this password. This ensures that secure access is achieved to the remote desktop application and avoids unauthorized access.

**2. Screen Sharing**

**Description:** The screen sharing component captures the server's screen and then forwards the same to the client in real time. This in turns lets the client look and interact with the server's desktop like one sitting in front of the server's screen.
**Functionality:** The server clicks the screen periodically using the Robot class. These clicked images are then compressed and sent across to the client using Socket connections. The screen sharing component at the server makes sure that these images are transferred with very less latency and high fidelity.

**3. File Transfer**

**Description:** The program for file transfer should support file transfer from a server and reception at a client, and vice versa.
**Functionality:** Listing on an incoming file transfer request, the server manages the process of file transfer by using Server Socket for incoming connections and Socket for data transmission across them. The files are read in chunks and written to make a program for efficient and reliable file transfer.

## Conclusion

In summary, the development and implementation of the Remote Desktop Application reflect one huge step forward, which enables facilitation of remote access and collaboration features within the same. The application is designed to provide a robust platform with real-time screen sharing, secure file transfer, and easy user interfaces to improve remote desktop management and productivity in different use cases.

The entire development process paid special attention to such application features as reliability, security, and user-friendliness. In this regard, the features of applying a Java programming language and following the best practices in software design make the application very user-friendly for both technical support professionals and in a collaborative work environment.