**7/8/24 PROJECT NAME**

**Network File Sharing Server and Client**

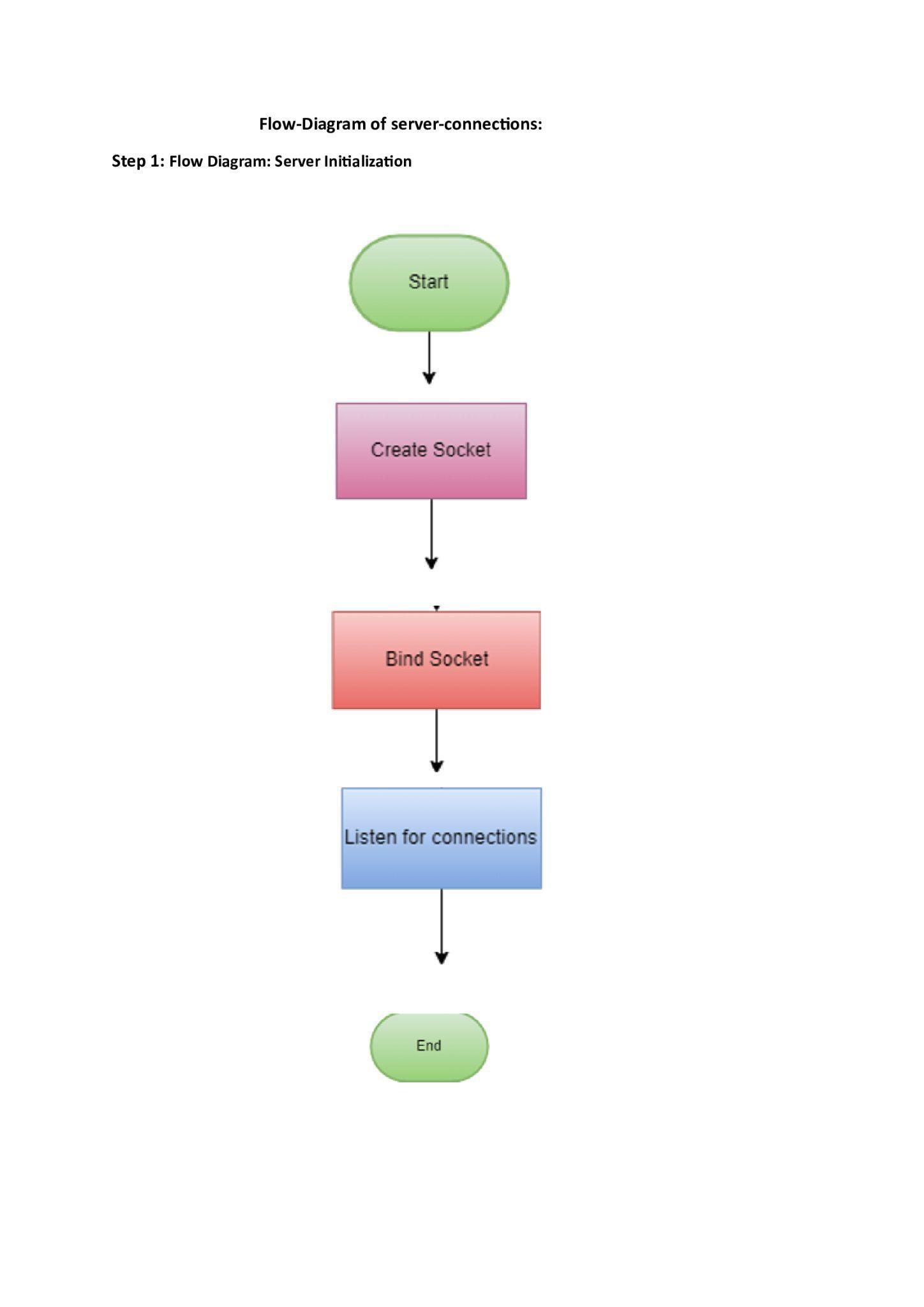
Creating a high-level and low-level design for a network file-sharing application involves outlining the system architecture, components, and interactions at different levels of detail.

**High-Level Design (HLD)**

**AIM :**  **Create a network file-sharing application with server-client architecture, enabling file transfer, listing, uploading, authentication, and encryption.**

#### **1. Architecture Overview**

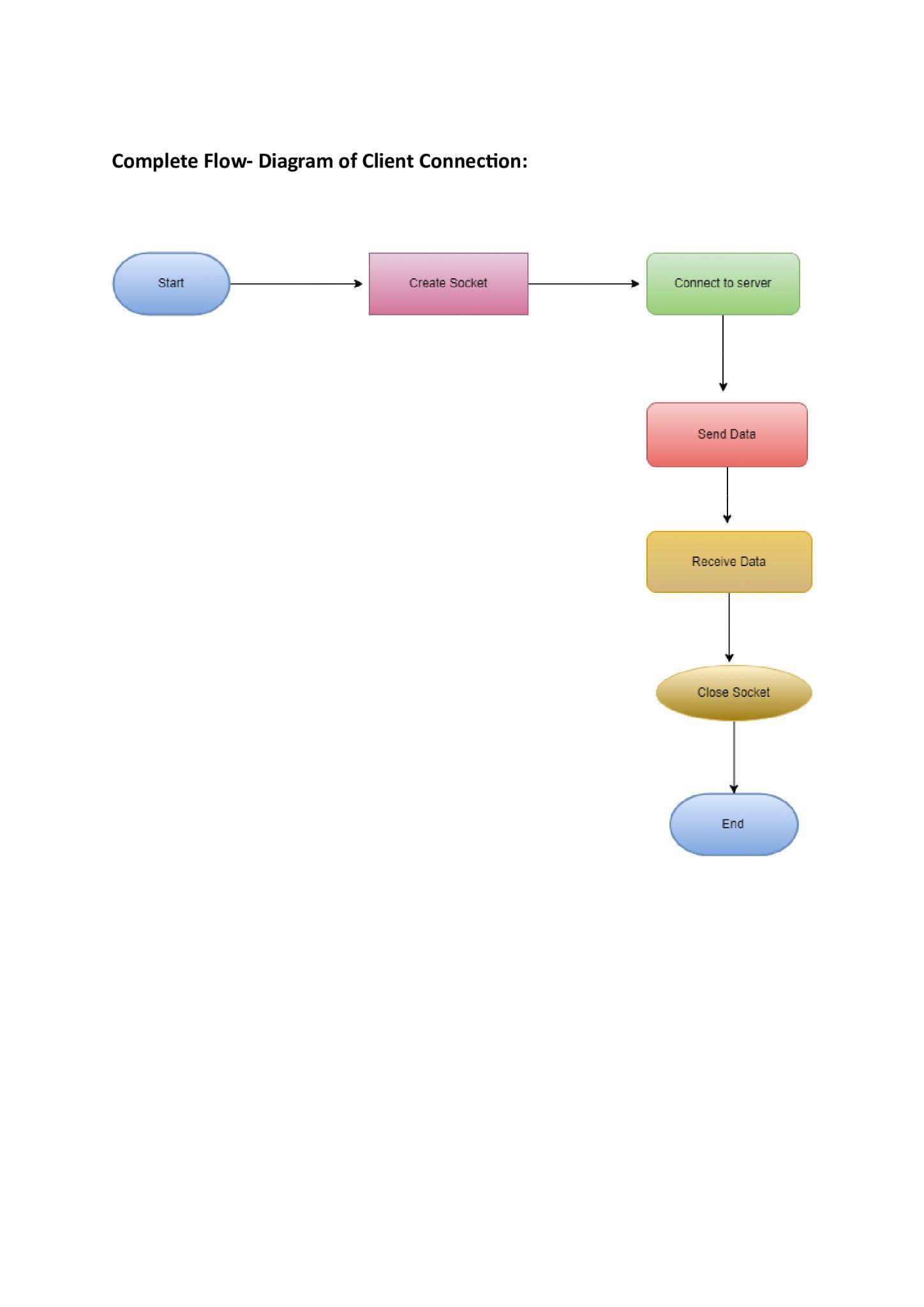
* **Client-Server Model: The system follows a client-server architecture where multiple clients can connect to a central server to user authentication, msg send, and send file .**
* **2. Data Flow Diagram**
* **1. Client Authentication:**
* **○ The client sends a password to the server for authentication.**
* **○ The server verifies the password and responds with an acknowledgment.**
* **Low-Level Design (LLD)**
* **1. Detailed Component Design**
* **1. Client Application:**
* **○ Modules:**
* **■ Network Module: Handles socket creation, connection, and communication.**
* **■ Authentication Module: Manages sending and receiving authentication data.**
* **■ File Transfer Module: Handles file msg and operations.**
* **■ User Interface Module: Provides command line interface for user interaction.**
* **2. Server Application:**
* **■ Network Module: Handles socket creation, binding, listening, and accepting connections.**
* **■ Authentication Module: Verifies client credentials.**
* **■ File Management Module: Manages file send, msg.**
* **■ Security Module: Manages SSL/TLS setup and encryption.**
* **2. Class Diagrams**
* **1. Client Classes:**
* **○ Methods:**
* **■ connectToServer()**
* **■ authenticate()**
* **■ sendFile()**
* **■ sendMessage()**
* **2. NetworkManager Class:**
* **○ Methods:**
* **■ createSocket()**
* **■ connect()**
* **■ sendData()**
* **■ receiveData()**
* **1. Server Class:**
* **○ Methods:**
* **■ start()**
* **■ acceptClient()**
* **■ handleClient()**
* **■ authenticateClient()**
* **■ receiveFile()**
* **■ sendAck()**
* **2. NetworkManager Class:**
* **○ Methods:**
* **■ createSocket()**
* **■ bindSocket()**
* **■ listenForConnections()**
* **■ acceptConnection()**
* **■ sendData()**
* **■ receiveData()**
* **Sequence Diagrams:**
* **1. Client Authentication Sequence:**
* **○ Client -> Server:**
* **connectToServer()**
* **○ Client -> Server:**
* **authenticate(username, password)**
* **○ Server -> Client: authenticationResult()**
* **2. File Send Sequence:**
* **○ Client -> Server: sendFile(filename)**
* **○ Server -> Client: sendAck() The server application diagram is designed to represent the different modules and their interactions within the server application. Each module has a specific role in handling client requests, managing files, and ensuring secure communication**
* **. CLIENT- This is the main class that orchestrates the operations of the client application. It utilizes other modules to establish a connection to the server, authenticate, transfer files.**

**CLIENT FLOW DIAGRAM**

**CLIENT- This is the main class that orchestrates the operations of the client application. It utilizes other modules to establish a connection to the server, authenticate, request file listings, and manage file transfers (upload/download).**

**The Client class uses the NetworkManager to send authentication credentials (e.g., password) to the server.**

**The server verifies the credentials and responds. The NetworkManager receives this response and passes it back to the Client.**

****

