

**upload\_file(sock, filepath)**

**📌 Purpose:**

Uploads a file from the client to the server in chunks and shows progress.

**📥 Parameters:**

* sock: the socket object used to send data to the server
* filepath: the path of the file the client wants to upload

**🧠 What it does:**

1. Checks if the file exists.
2. Extracts filename and filesize.
3. Sends the command:

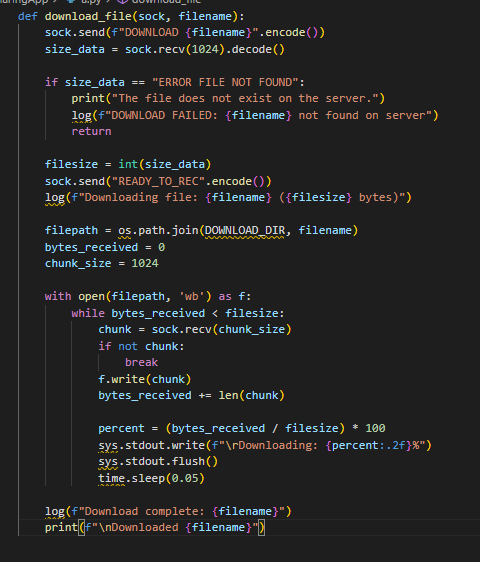
nginx

CopyEdit

UPLOAD filename filesize

1. Waits for the server to reply with "READY\_REC".
2. Reads the file in chunks (1024 bytes) and sends each chunk.
3. Displays an upload progress bar in the terminal.
4. Logs success or failure.

The upload\_file function is responsible for sending a file from the client to the server over a TCP socket. It first checks if the specified file exists locally and then retrieves the filename and its size. The client informs the server about the file to be uploaded by sending a command in the format UPLOAD <filename> <filesize>. Once the server acknowledges readiness with "READY\_REC", the file is read in binary mode and transmitted in chunks (1024 bytes each). During the transmission, the client displays a real-time upload progress percentage and logs the result. This function ensures a smooth and trackable upload process with visual and file-based logging for transparency.

------------------------------------------------------------------------------------------------------------------  


**✅ download\_file(sock, filename)**

**📌 Purpose:**

Downloads a file from the server to the client and shows download progress.

**📥 Parameters:**

* sock: the socket object to receive data
* filename: the name of the file you want to download from the server

**🧠 What it does:**

1. Sends the command:

nginx

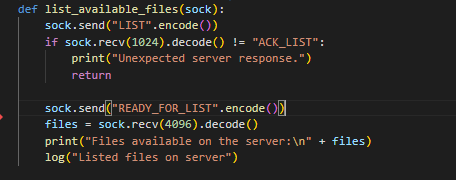
CopyEdit

DOWNLOAD filename

1. Waits for server to respond with the file size or "ERROR FILE NOT FOUND".
2. Sends back "READY\_TO\_REC" to tell server it's ready.
3. Receives the file chunk by chunk and saves it in the received/ folder.
4. Displays a download progress bar.
5. Logs the download completion or failure.

The download\_file function enables the client to retrieve a file from the server. Upon issuing the DOWNLOAD <filename> command, the client awaits a response that indicates either the file size or an error if the file does not exist. If the file is available, the client acknowledges with "READY\_TO\_REC" and begins receiving the file in chunks. Each chunk is written to a new file in the received/ directory. The function also provides a live download progress indicator in the terminal and logs the transfer details for reference. This ensures file integrity during transfer and a user-friendly experience.

------------------------------------------------------------------------------------------------------------------



**list\_available\_files(sock)**

**📌 Purpose:**

Requests a list of available files on the server.

**📥 Parameters:**

* sock: the socket object used to communicate

**🧠 What it does:**

1. Sends the command:

nginx

CopyEdit

LIST

1. Waits for the server to send "ACK\_LIST".
2. Sends "READY\_FOR\_LIST" as confirmation.
3. Receives and prints the list of files.
4. Logs the request.

The list\_available\_files function allows the client to request and view the list of files currently stored on the server. The client begins by sending the LIST command, after which it waits for an acknowledgment "ACK\_LIST" from the server. Once acknowledged, the client responds with "READY\_FOR\_LIST" to proceed. The server then sends the list of filenames, which the client prints to the screen. This function provides an easy way for users to explore what files are available for download, while also logging the interaction for traceability.

------------------------------------------------------------------------------------------------------------------

****

**log(message)**

**📌 Purpose:**

Writes log messages to logs\_client/client\_log.txt.

**📥 Parameters:**

* message: a string to log

**🧠 What it does:**

* Opens the log file in append mode.
* Adds a timestamp and the message.
* Used for tracking client actions and errors.

The log function is a utility used throughout the client application to record events and operations. It appends each message, along with a timestamp, to a local log file stored in the logs\_client/ directory. This logging mechanism is essential for debugging, monitoring file transfers, and maintaining a historical record of client activity. It ensures that both successful and failed operations are captured for later review.

------------------------------------------------------------------------------------------------------------------

**A screen shot of a computer program

AI-generated content may be incorrect.**

**✅ main()**

**📌 Purpose:**

Starts the client app and listens for user commands.

**🧠 What it does:**

1. Connects to the server using socket.connect().
2. Shows user command options:
   * LIST
   * UPLOAD <filepath>
   * DOWNLOAD <filename>
   * EXIT
3. Parses input and calls the appropriate function.
4. Runs in a loop until the user types EXIT.
5. Closes the socket and logs disconnect.

The main function serves as the core of the client application, managing server connection and user interactions. It starts by establishing a TCP connection with the server using the predefined IP and port. Once connected, the client enters a loop where it continuously prompts the user for commands. Depending on the user's input, it delegates tasks to the corresponding functions such as uploading, downloading, listing files, or exiting the session. This function ensures persistent communication with the server, clean input handling, and graceful termination, making it the backbone of the client’s functionality.