Raza Ghulam, Skyler Mayfield

TW-Mailer Description

Client and server architecture

The TW-Mailer is an object-oriented application.

- **Client class:** Contains a client socket object and is initialized with a port and an IP address
- **Server class:** Contains a server socket and a connection object. It is initialized with a port and a directory name, which is used to persist incoming and outgoing messages as well as meta data
- **Socket class:** The socket class consists of 2 subclasses: Client socket and Server socket
- **Connection class:** Accepts a connection and keeps track of the connection status

To start the server, a server object is initialized. Subsequently a server socket is created by initiating a socket instance and assigning the port on which the server should listen. After the socket is bound to an address, the server starts listening for connections. Afterwards, a connection instance is initiated to accept the connection.

To start a client, a client object is initialized. Analogously a client socket is created with the specified port and IP address. Thereafter the client socket is connected to the server.

Once the connection is established, data is sent to the server. After processing the data, the server responds with the desired data and “ok” if the client request was processed successfully or “err” if it was unsuccessful.

Saving received data is done by creating 1 directory for each sender and 1 file for each message.

User commands

- SEND: client sends a message to the server
- LIST: lists all messages of a specific user
- READ: display a specific message of a specific user
- DEL: removes a specific message
- QUIT: logout the client
- HELP: detailed description of each command

Used technologies

Operating system: Linux

Programming language: C++

Development strategy and needed adaptions

Before everything else we researched on the internet as well as the course documents, to get familiar with sockets and client-server application.

Afterwards, we started developing the server up to the point, where it would wait for a connection from the client. Next, we created the client and successfully established a connection. Then we started implementing the commands on the client side and sent back the same message from the server to ensure its correctness. When this was done, we started developing the data processing and file handling on the server.