Python TCP Chatting Project

Agenda:

Chapter 1: Introduction

Chapter 2: Software Requirements

Chapter 3: Architecture Overview

Chapter 4: Detailed Code Explanation

Chapter 5: Running the Application

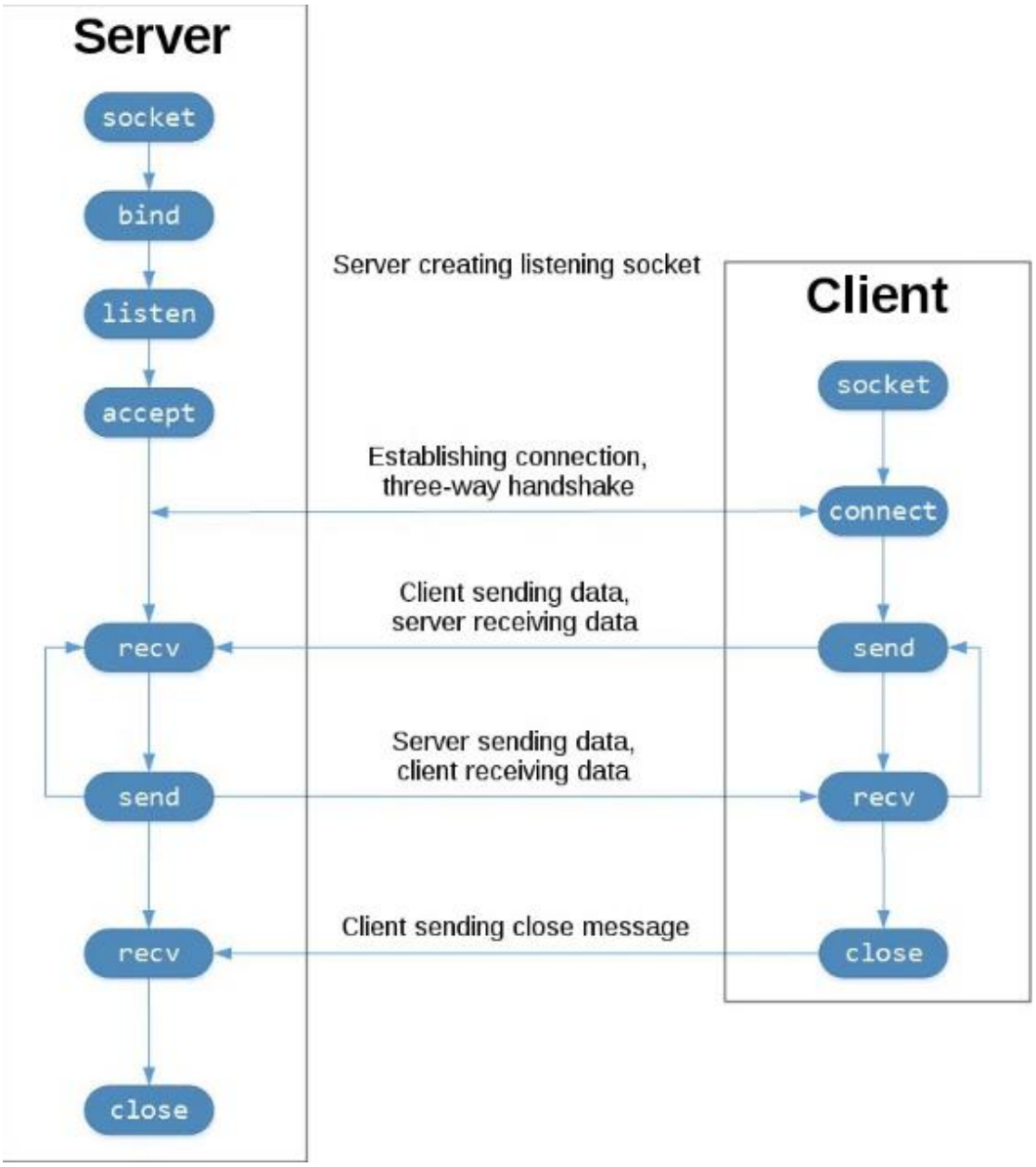
Done By :

1. Omar Abdulaal (230103273)
2. Mahmoud Usama (230103270)
3. Mariam Khallil (230100000)

Chapter 1: Introduction

How The Code Runs :

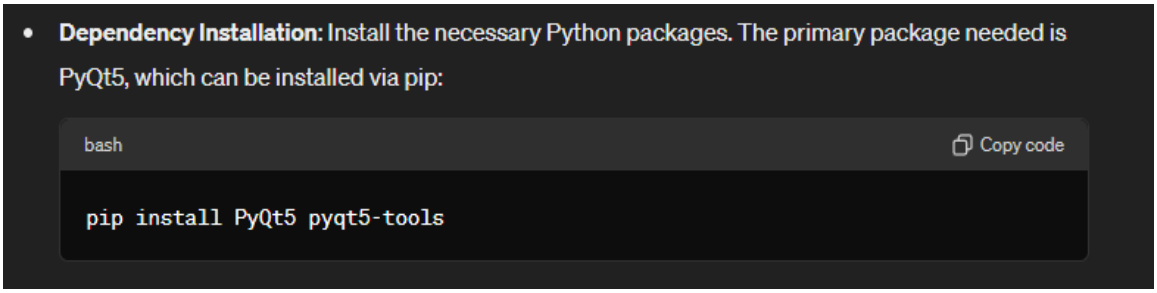
This document explains the implementation of a chat server using Python, socket and PyQt5 (GUI Library).



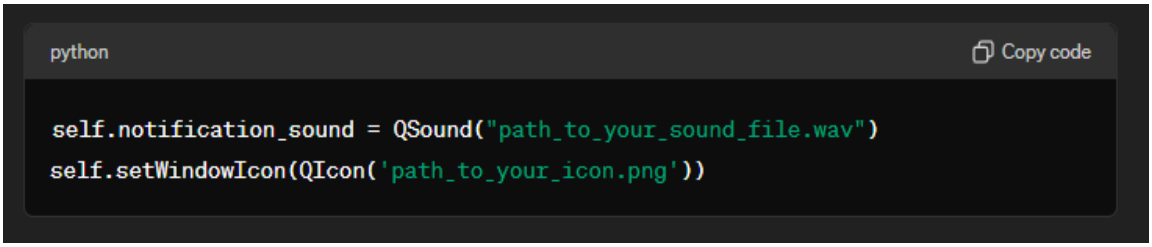
Chapter 2: Software Requirements

- Python and PyQt5 need to be installed, along with dependencies for sound playback and icons.

- **Dependency Installation:** Install the necessary Python packages. The primary package needed is PyQt5, which can be installed via pip:



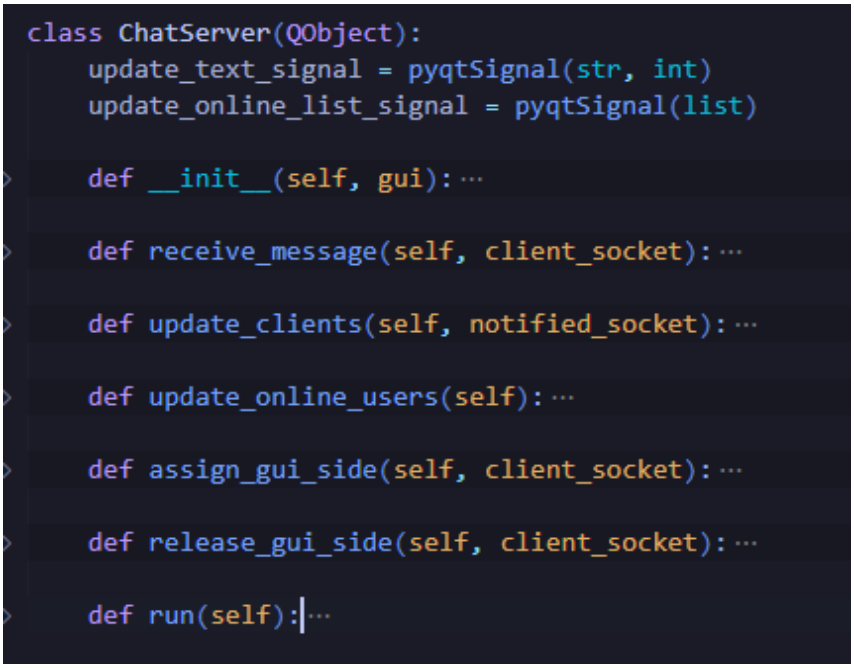
- **Sound and Icon Resources**: The application uses specific sound files and icons. Ensure the paths in the code match the actual locations on your system or adjust the code to point to the correct paths. For example:



Chapter 3: Architecture Overview

- A client-server model where the server handles multiple client connections using non-blocking sockets.

1. Server BackEnd Code:



```
class ChatServer(QObject):
    update_text_signal = pyqtSignal(str, int)
    update_online_list_signal = pyqtSignal(list)

    def __init__(self, gui):···

    def receive_message(self, client_socket):···

    def update_clients(self, notified_socket):···

    def update_online_users(self):···

    def assign_gui_side(self, client_socket):···

    def release_gui_side(self, client_socket):···

    def run(self):···
```

2. Server GUI Code:

```
class ServerGUI(QMainWindow):
    update_text_signal = pyqtSignal(str, int)
    update_online_list_signal = pyqtSignal(list)

    def __init__(self):...

    def initUI(self):...

    def update_chat_display(self, message, client_id):...

    def revert_color(self, chat_display, message, original_color):

    def update_online_list(self, users):...

    def stop_server(self):...
    def initTray(self):...
```

Chapter 4: Detailed Code Explanation

❖ From the ServerBackEnd Code:

- `def __init__(self, gui):`
  - Used to initial the backend code with the server Gui
- `def receive_message(self, client_socket):`
  - Used to receive message from the client when the server is connected
- `def update_clients(self, notified_socket):`
  - Used to update clients when they try to connect to the server
- `def assign_gui_side(self, client_socket):`
  - Used to direct the client for the chat side to improve the vision
- `def update_online_users(self):`
  - Used to refresh the panel of the online clients

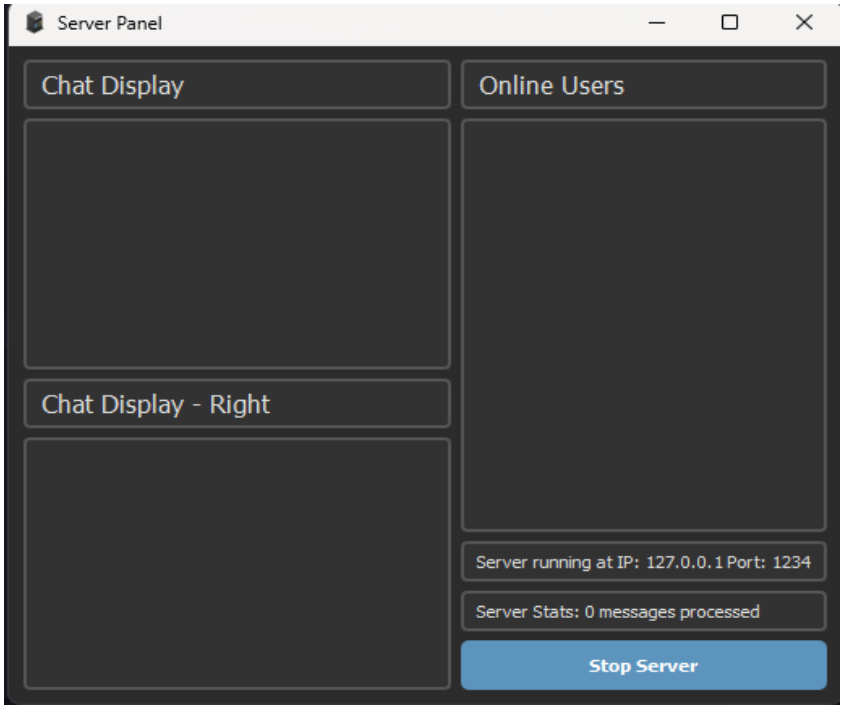
❖ From The ServerGUI Code:

- `def __init__(self):`
- `def initUI(self):`
  - Used to Initial the code in the GUI frame and make it visible
- `def revert_color(self, chat_display, message, original_color):`
  - Used to control the colors of the text to know their statues
- `def update_chat_display(self, message, client_id):`
  - Used to refresh the code to view messages that been send by the clients
- `def update_online_list(self, users)`
  - Used to update the panel of the online clients
- `def stop_server(self):`

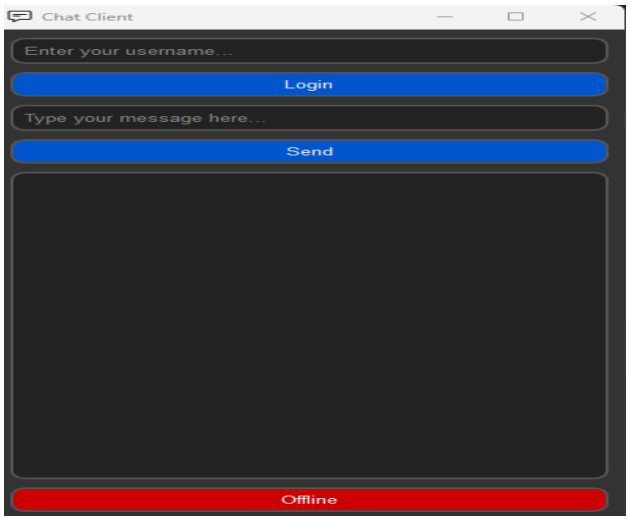
- Used to cut the server out and kick out any clients

Chapter 5: Running the Application

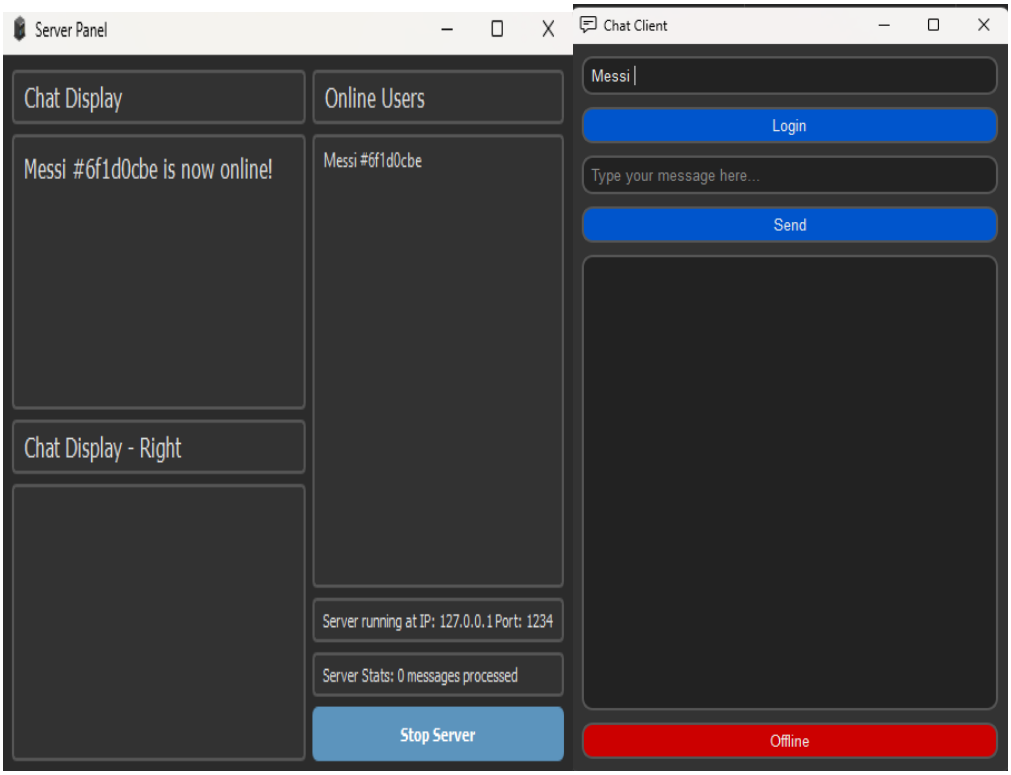
1. Run the Server: (The server running on port: 1234)



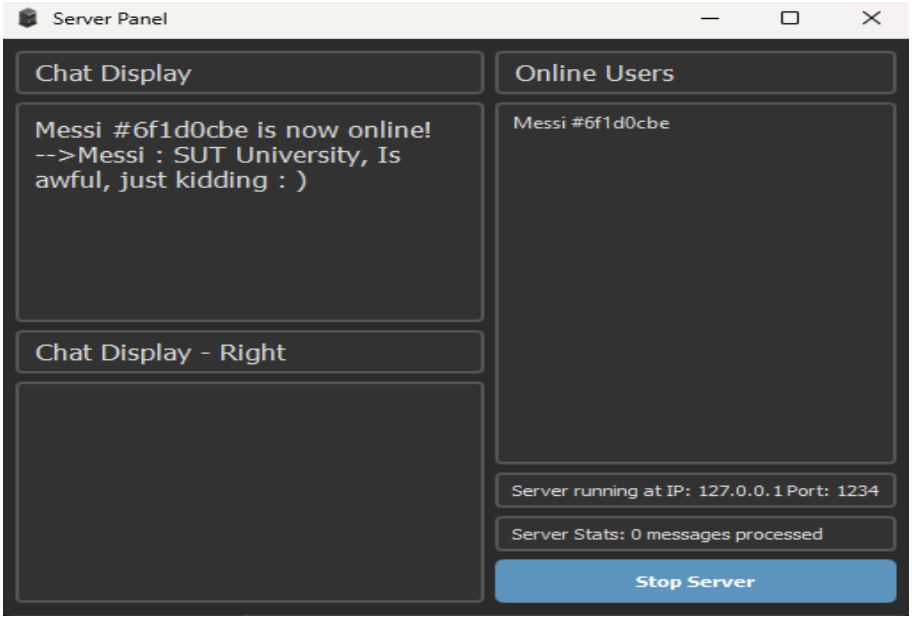
2. Run client.py and connect it the server



3. Establish connection:



4. Now it been connected, do what you want but not in illegal way ;):


Server Panel
Chat Display
Messi #6f1d0cbe is now online!
-->Messi : SUT University, Is awful, just kidding :)
Chat Display - Right
Online Users
Messi #6f1d0cbe
Server running at IP: 127.0.0.1 Port: 1234
Server Stats: 0 messages processed
Stop Server

Thank You