# Chat Application using TUI

## Project Report

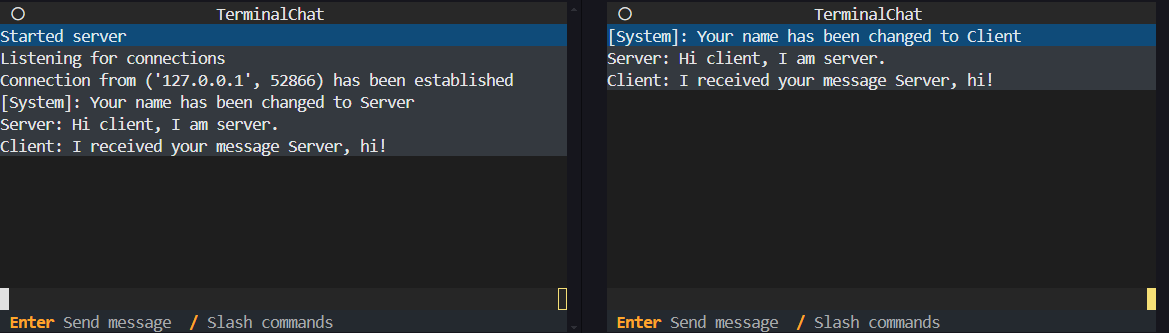
#### Enis ISKENDER

Introduction

This project is an example of how to create an application with multiple threads. Current code only connects to local machine. The network architecture used is an user acting as the MCU (multipoint control unit). This can be changed to dedicated server acting as the SFU (selective forwarding unit) for better performance. This program uses Textual for the user interface, standard Python sockets for networking and standard queue for communication between threads.

Architecture and Solutions

Most UI frameworks do not grant access to modifying UI elements from other threads. This project uses Queue, a producer-consumer system for communication between threads. Network thread produces a message when packets are received from the socket, and the UI thread consumes the messages from the queue. The network architecture used, MCU, means all clients connect to a central server, and there is no control of what type of messages to send/receive. Since an user acts as the MCU, they could be put under under heavy network and CPU load if many clients connects to the server.



Final Thoughts

It’s quite easy to work with sockets in Python. If this project were to be improved, a change in network architecture would be necessary. Protobuffers could also be used in order to supply messages on sockets with more information, such as who sent the message.