CSC3002F NETWORKS ASSIGNMENT 1 2022 SOCKET PROGRAMMING

Tshegofatso Mokaa
MKXTSH013@myuct.ac.za

Happiness Mkhwanazi
MKWHAP002@myuct.ac.za

Mkhululi Ncube
NCBMKH005@myuct.ac.za

TABLE OF CONTENT

INTRODUCTION

The goal of this project is to investigate network applications, especially the usage of UDP in networking. There are procedures in place since things may go wrong during the data transfer, such as data corruption because networks are a huge region.

We will develop a Python-based client-server chat application that uses a User Datagram Protocol at the transport layer. UDP is a communication protocol where there is no connection between the client and server. It provides unreliable data transfer of a group of bytes called datagrams between the client and server. The sender explicitly attaches the IP destination address and port number to each packet and the receiver extracts the sender IP address and port number from the received packets.

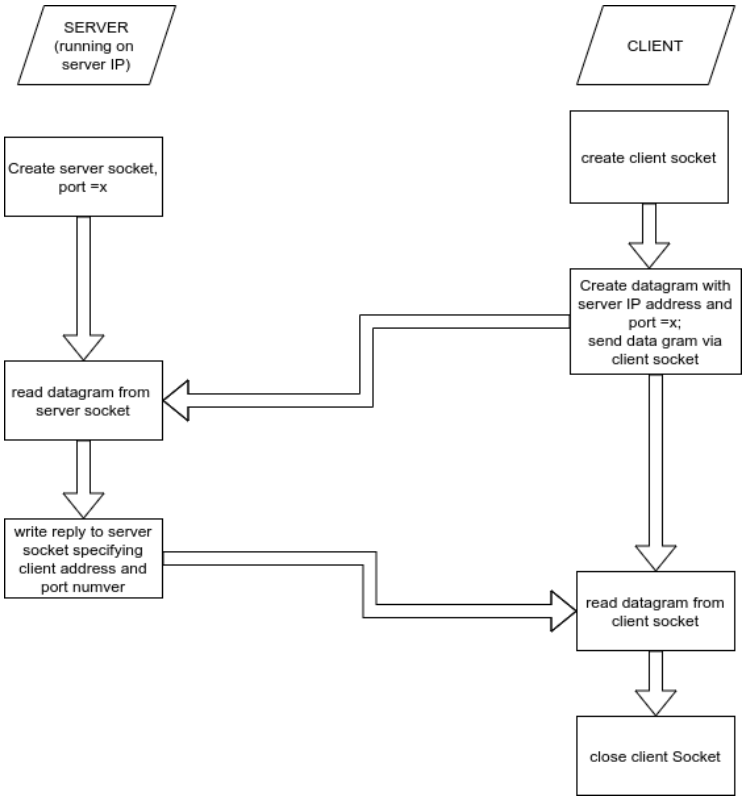


Fig. 1 : client/server socket interaction

DESIGN SPECIFICATIONS

CLASSES AND METHODS IMPLEMENTATIONS

Server.py

The server.py class is what we used as the server for the application. The class contains methods to send data to groups if a user is a participant in a certain group, if not it will inform the user that there are no chats for them. StoreGroupData() method to store group information such as ID, the user who created the group, and group name. The validate_people method is used to validate if the user information is correct so the data can be stored. Every time a user log in they are allocated unique authentication tokens used to distinguish different users.

Client.py

The Client.py class is what all the clients that want to connect to the application will be running to try and connect with the server and other clients. It contains methods such as group_request() used to store information about the group and send data to multiple clients that are parts of the group, the showChatMessages() is responsible for displaying the chat messages of the groups when a user is within the group, the SendMessage() is responsible for send messages amongst members of the group and there is a trace to show if the message was successfully sent within. The RegisterUser() and LoginUser() methods are used by clients to either register when it's their first time connecting to the application or login in when they're already connected to the application and the server is up and running.

SECURITY

Since User Datagram Protocol is a connectionless and unreliable protocol, we had to implement an application that ensured that all packets lost and corrupt are recovered.. As an additional feature we used an authentication token for user to be able to access their messages. The authentication token is unique for each to ensure that only the that user is able to access messages.

SEQUENCE DIAGRAM

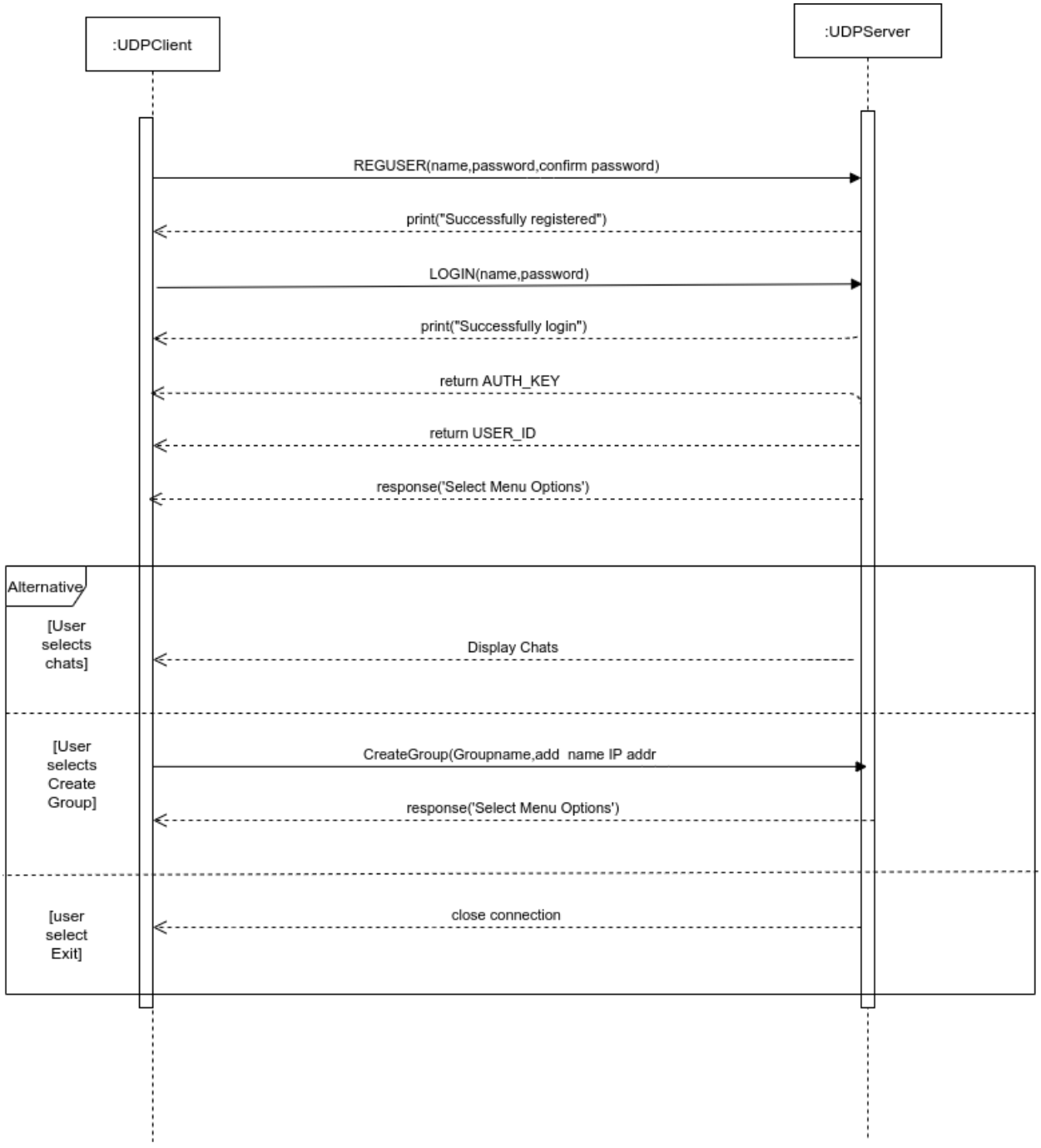


Fig. 2 : Sequence diagram for the chat application

APPLICATION FUNCTIONALITIES

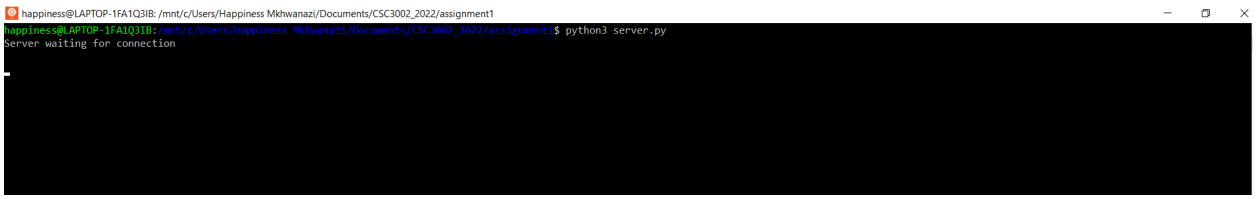
The application works by sending data in a form of a JSON string between clients and a server. We specifically used Json string because we faced difficulties while trying to use objects with UDP. The Json string is divided into three categories which are type, sender and data. Type refers to the type of services such as registering/logging in, the sender is the is IP address of the client and the data will contain user information such as name and password.

After every request a client sends the server will respond to the client. That is, if a user request to login, after the user has successfully logged in the server will send a message to inform the user if the log in activity was successful or not.

Below are the visual application functionalities and descriptions in sections :

Firstly the server should be up and running before the clients. Once the server is ready, the clients can now connect by running the Client.py.

Fig. 3 : Screenshot of the server waiting for connection



When the user connects, they are asked to either register or login in. Each user needs to register every time a new connection is formed (meaning if the server crashes or closed when the server is run again the client will need to re-register). Once the user has registered they will be prompted to enter their details to login in. Once they have successfully logged in they will be allocated a unique authentication token and a user ID.

The data gets stored in server.py as shown below

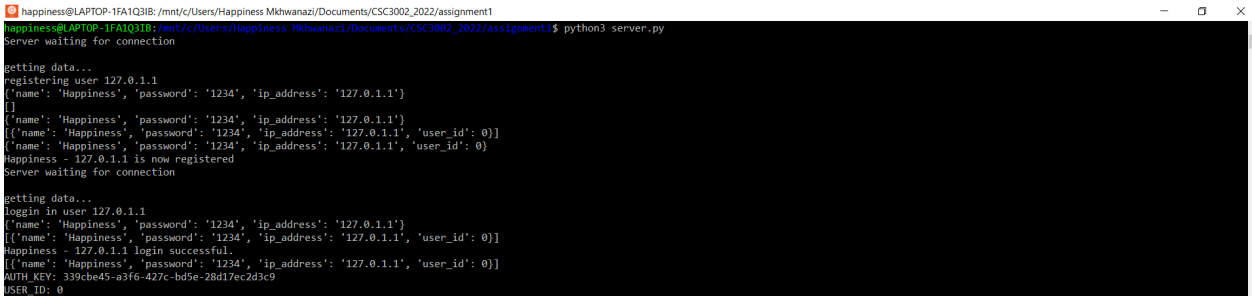
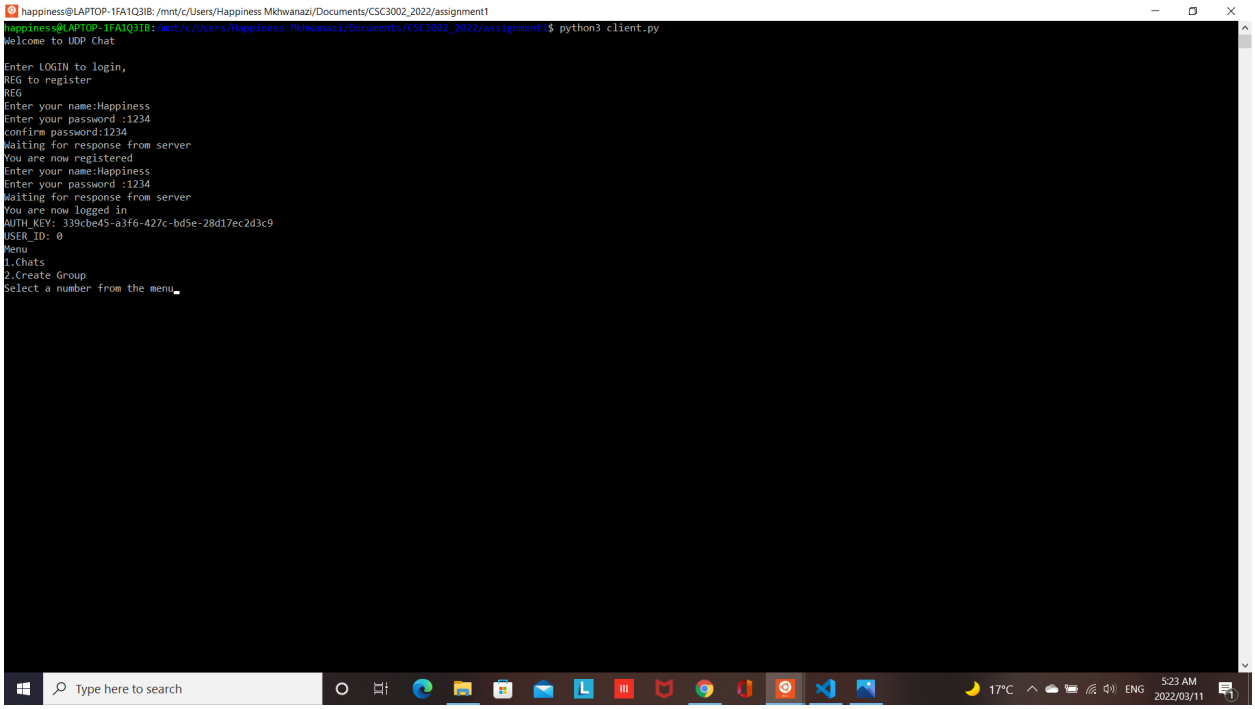


Fig.4 : Screenshot of the data that gets stored after registering & logging in

Fig.5 : Screenshot of user registering and logging in



Once the user has logged in they will be prompted to select from the menu whether they would like to access their chats or created a new group. If the user selects chats then all the user chats will be displayed and if the user chooses to create a group they will need to provide the group name, names of the group members and their IP address.

Fig.6 : Displaying menu & prompting user to view chats or create group

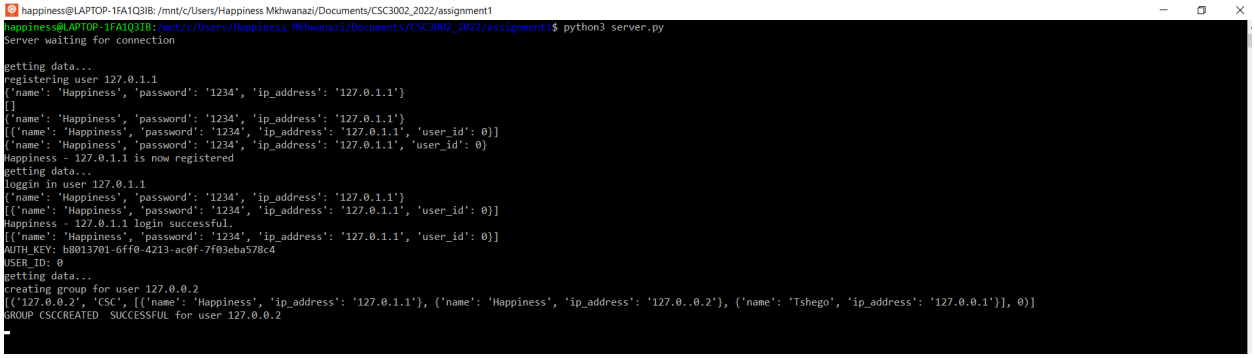


Fig.7 : Group successfully created and users' details stored.

The group will be successfully created, the users that have been invited to the group will be able to view the group under their chats and once they select the specified group they will be able to view the messages from all members of the group.

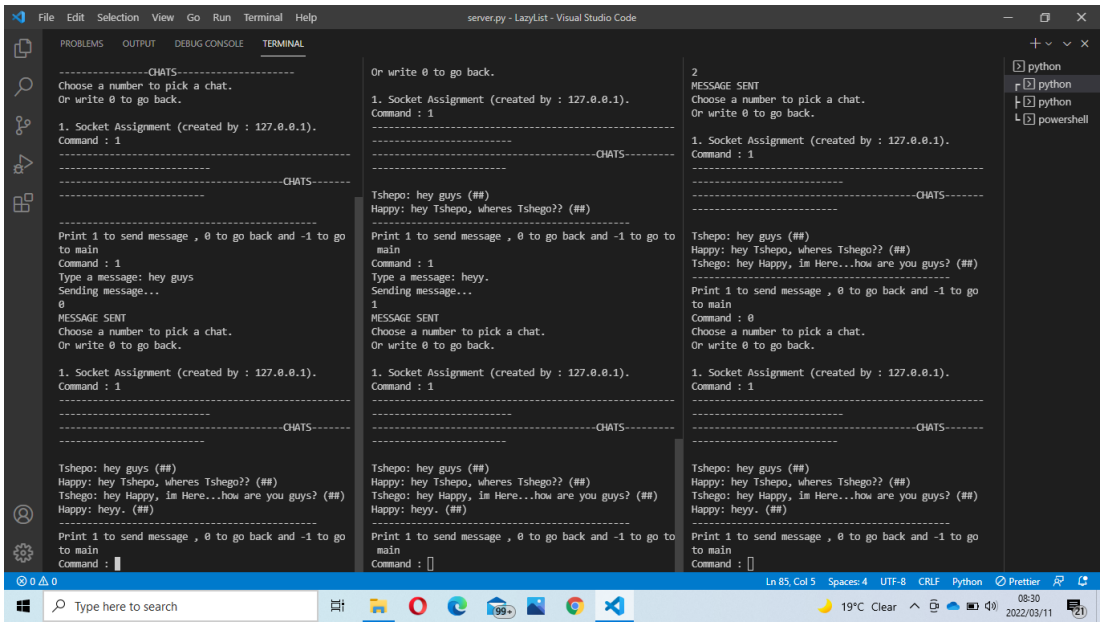


Fig. 8 The group messages

INSTRUCTIONS TO EXECUTE THE APPLICATION

- Run the application on the terminal with all files in one directory
- Run the server first [python3 Server.py]
- Then run the client [python3 Client.py]
- To run the application on your machine, change the host and or port numbers in the server and client program

CONCLUSION

In conclusion, the goal of if the assignment was to develop a client-server application that uses UDP at application layer. The application was successfully developed with additional features such as group chat and security, since UDP is unreliable and data can be lost or corrupted during transmissions.

REFERENCES

Chavula, J 2022. Application layer slides; page 1-69