PROJECT REPORT

RealTime-Multiplayer Quiz Based on Python v3.7 using Sockets

*** INSTRUCTIONS ***

(For Running Everything On Same Computer)

1. Run server.py file in python 3.7 using following command.

```
$ python3 server.py
```

2. Open 3 more Terminal Tabs and Run client.py file in python 3.7 in each terminal using following command.

```
$ python3 client.py
```

3. Server script is already hardcoded to use 'localhost IP' and Port Number 9999.
4. You are ready to play the Game.

*** GAME DESCRIPTION ***

There are three players in this game competing against each other to answer some questions. For every question, each player has time limit of 10 seconds to press buzzer ('ENTER' Key). The First one to press Buzzer would be asked by host for the answer input. The scores Would be given according to the answer input. If no one presses the buzzer till 10 seconds, Host will proceed to next question.

Scores of all the players are displayed after each question.
Final Result is displayed at the end.
Buzzer -----> "ENTER" key

Scoring Scheme :

Correct Answer = 1 pt
Wrong Answer = - 0.5 pt

Assumptions:

1) After 50 questions game will end automatically, If no one reaches at or above 5 points.
2) There is no time limit to answer the question after buzzer is pressed.
3) **Answer to the question is available on server's side.**
4) Answer to a question is only the option number i.e. 1,2,3 or 4.

*** PROJECT DESCRIPTION***

Modules Used :

Server Side :	Client Side :
• socket • select • sys • time • random	• socket • select • sys • time • os • termios

Pre Requisites for Networking Part :

Socket Programming with TCP

Socket :- A socket is one end point of a two way communication link between two programs running on the network. A socket is bound to a Port number so that the TCP layer can identify the Application that the data is destined to be sent to. An endpoint is combination of IP Address and Port number.

TCP (Transmission Control Protocol) – It is a standard that defines how to establish and maintain a network conversation through which application programs can exchange data.

Direct And Reverse Connections :-

1. Direct Connection – Directly connecting multiple clients.
2. Reverse Connection – Connecting multiple clients through a single server.

IP Address

IP Address is a unique string of numbers that identifies each computer using the Internet Protocol to communicate over the network.

1. Public and Private IP

Public IP : It is the IP address which is assigned to your device by internet and can be used to connect with anyone globally.

Private IP : It is assigned to your device by your Router and can be used to connect with devices over that LAN network only.

2. Static and Dynamic IP:

Static IP: It is the IP address which does not change over a period of time. Generally used for servers and websites.

Dynamic IP : It is the IP address which changes time to time. Generally for common devices connected to internet

Port Number : -

Port number is the address to the application in your device. It is used to identify which application has to recieve the data that has been sent.

Some Basic Functions used in Socket Programming :-

socket.socket ( ) - Creates a socket and sets the connection to given type - TCP or UDP
socket.setsockopt ( ) - Allows to reuse the Address and Ports
socket.bind ( ) - Binds the Socket to Given Host Address and Port Number
socket.listen( ) - Listens for Multiple Connections of Clients connecting to same port and address
socket.accept( ) - Accept the Client's Connection and returns Client Object and Address of Client
socket.connect( ) - Connects the socket to given Address and Port Number of Server
socket.send ( ) - Sends data to a connection object
socket.recv ( ) - Recieves data from a connection object
socket.close( ) - Closes the socket

select.select( ) - It blocks the Program, until it reads any input from given list of socket objects, till a given time limit. It returns the list of objects that responded in order of their response time.

Working of Project:

The project is divided into two phases -

1. Server Phase
2. Client Phase

Server creates a socket using *socket ( )* function in *socket module* and bind it to the Host IP Address and a Port Number using *bind ( )* function. Server waits for connection from clients. When it accepts 3 clients, it calls *start_quiz ( )* function to start game. It then asks questions to the clients one by one.

The connection established is through the TCP/IP layer.

Client also creates a socket to connect to server Host IP and Port Number using *connect ( )* function. It then sends and recieves the data and acknowledgements to and from server respectively.

Question-Answer Part with Buzzer Functionality :

Server Side :

This part uses *select ( )* function of *Select module*. It blocks the program until it get any response from client side with a time limit of 10 seconds.

If *select ( )* gets any response, it returns a list in which first index is client object of the first response it got. Server then do further operations with that client object.

If it does not get any response i.e Recieves an Empty list, It continues the program.

Client Side :

Using *select( )* function of *Select module*, It blocks the program until it get any response from its terminal input or server (as acknowledgment that other player has buzzed).

If the first response is from server, it waits till the player answers the question. Any input given by user in this time is flushed out before It gets next question.

If the first response is from its terminal input, it sends that input to the server.

Server then calls *end_quiz ( )* function to display final results after any player reaches on or over 5 pts. Server then closes all the connections using *close ( )* function.