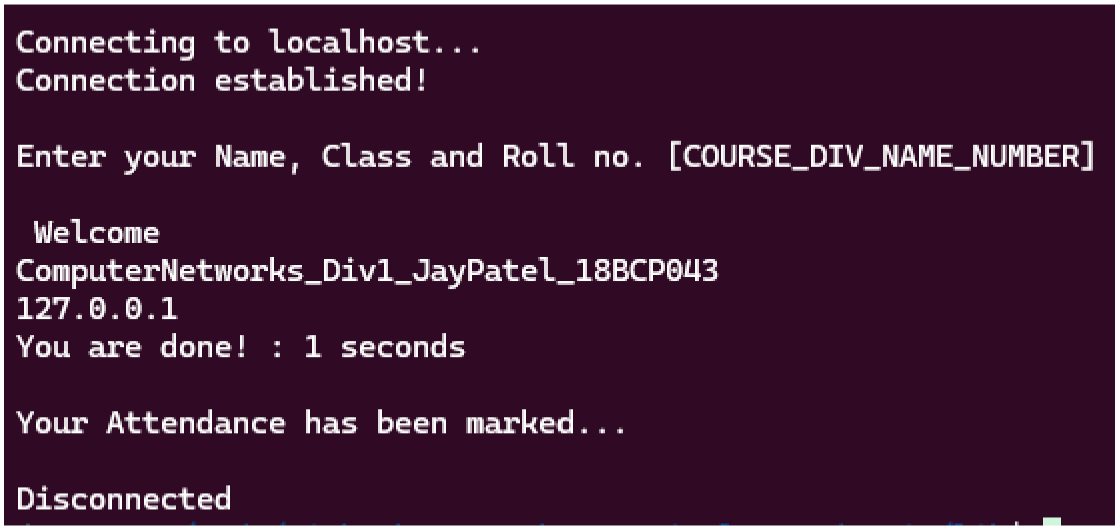
# Network Attendance Tracker

Network Attendance Tracker is a basic client-server system for attendance management over a network. The client portion allows users to input their personal details such as name, class, and roll number. After establishing a connection to the server, it sends this information, waits for a brief period, and then marks the user as "present" in the attendance record maintained by the server. The server side listens for incoming connections, receives attendance data from clients, parses and stores it in a CSV file.

Additionally, it broadcasts the arrival of new clients to all connected clients. This project facilitates remote attendance marking, making it convenient for users to log their presence without physical attendance registers. It leverages network communication to streamline the attendance recording process, enhancing efficiency and accessibility.

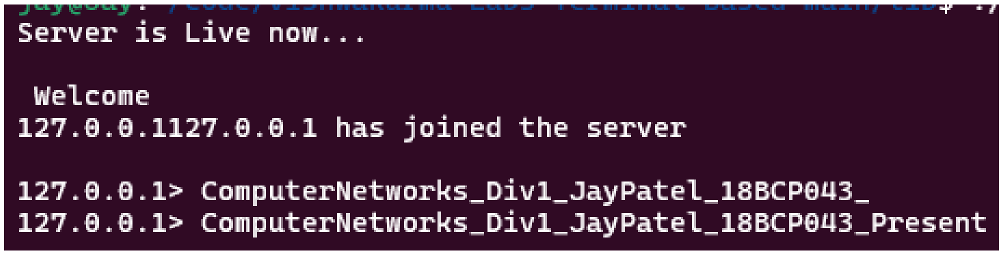
This project was created to understand the concepts of network programming, socket communication, and how both client and server use TCP/IP sockets for reliable communication over the network.

The below screenshot represents Client-side output, so here student will come and enter their information in the form of 'COURSE_DIV_NAME_NUMBER' and once entered, student has to complete entire class (maybe 1 hour long – till which our timer will run) and once class is completed, the attendance will be marked to server.

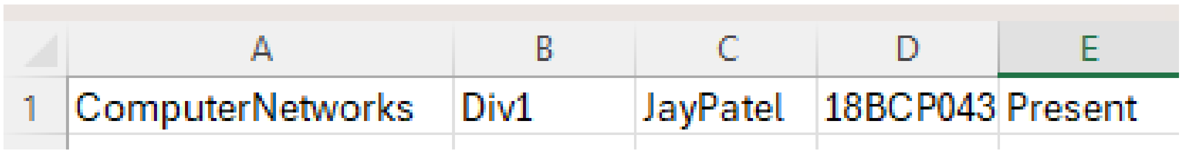


```
Connecting to localhost...
Connection established!

Enter your Name, Class and Roll no. [COURSE_DIV_NAME_NUMBER]

 Welcome
ComputerNetworks_Div1_JayPatel_18BCP043
127.0.0.1
You are done! : 1 seconds

Your Attendance has been marked...

Disconnected
```

The screenshot below shows the output when we run server-side code and it says when Jay Patel entered the class and marks present after waiting for 1 hour (class timing) and once present is marked, it will be stored into .CSV file as shown.



```
Server is Live now...

 Welcome
127.0.0.1127.0.0.1 has joined the server

127.0.0.1> ComputerNetworks_Div1_JayPatel_18BCP043_
127.0.0.1> ComputerNetworks_Div1_JayPatel_18BCP043_Present
```



| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | ComputerNetworks | Div1 | JayPatel | 18BCP043 | Present |

To better understand what is happening on Client-side and server-side,

Client-side:

- Socket Creation: Creates a socket to establish a connection with the server. (Endpoint for communication between client and server)
- Connection Establishment: Connects to the server using the server's IP address (Identifies the client and server on the network) and port number (Identifies a specific application or service on the server).
- User Input Handling: Accepts user input for name, class, and roll number.
- Timer Implementation: Sets a timer for 5 seconds using the wait_time() function.
- Sending Data: Sends user input and "Present" message to the server.
- Receiving Data: Receives and prints messages from the server.
- TCP (Transmission Control Protocol): Provides reliable, connection-oriented communication between client and server.

Server-side:

- Socket Creation: Creates a socket and binds it to a port for listening.
- Connection Acceptance: Accepts incoming connections from clients.
- Receiving Data: Receives data from clients and processes it.
- CSV Logging: Logs client information (class, division, name, roll number, attendance) into a CSV file.
- Handling Multiple Clients: Uses select() to handle multiple clients concurrently.
- Server Shutdown: Allows for server shutdown by sending a "shutdown" command.

Hence, in general, we can make a web-app based on this logic to show runtime changes made by one user. If we can imagine WhatsApp messenger, then we can see that once user A sends a message to user B, they get updated in real-time. Also, there are push notifications applied as well.

Link to code: https://github.com/jay-patel-07/Network_Attendance_Tracker