# CNET LAB

**Muhammad Azan Afzal**

**22I-1741**

**CY-B**

**Assignment no : 2**

## Report on Server and Client Implementation

## Overview

This report provides an explanation of the server and client code designed to handle interactions between teachers and students over a network. The primary functionality includes role identification, file management for storing student information, and command handling for adding or viewing student records.

## Server Code Explanation

The server code is responsible for accepting client connections, identifying the client's role (teacher or student), and managing interactions based on the role. The key components and functionalities are described below:

### 1. Socket Setup and Listening

The server initializes a socket using the socket() function and binds it to a specified port (8080 in this case). It then listens for incoming connections with listen() and accepts them using accept().

### 2. Handling Client Connections

Upon accepting a client connection, the server forks a new process to handle the client. This ensures that multiple clients can be handled concurrently.

### Role Identification

In the handle\_client() function, the server asks the client to specify their role (teacher or student). Based on the response, it calls either handle\_teacher() or handle\_student().

## Teacher Handling

### 1. File Creation

When a teacher connects, the server requests their name and automatically creates a file if it does not already exist.

### 2. Command Handling

The teacher can add or view student information through commands (add, view, exit):

**Add:** Prompts the teacher to enter student information and appends it to the teacher's file.

**View:** Reads and sends all student records from the teacher's file to the client.

## Student Handling

### 1. Roll Number Input

Students provide their roll number, which the server uses to search for their information across all teacher files.

### 2. Information Retrieval

The server iterates through potential teacher files, looking for records matching the student's roll number. If found, it sends the matching records to the student.

## Strategy and Functionality

The primary strategy used in this implementation is the role-based handling of client interactions, which ensures a clear separation of functionalities for teachers and students. The automatic file creation for teachers simplifies the user experience, eliminating the need for additional prompts. Concurrent client handling is achieved using the fork() system call, allowing the server to manage multiple clients simultaneously.

## Conclusion

This server-client implementation efficiently manages teacher and student interactions, ensuring robust file handling and clear communication protocols. The code design prioritizes simplicity and ease of use while maintaining functionality, making it a practical solution for managing student records over a network.