AMERICAN UNIVERSITY OF BEIRUT
MAROUN SEMAAN FACULTY OF ENGINEERING & ARCHITECTURE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
EECE 350 - Computer Networks

Network Programming Project

AUB Registrar

Due: Apr. 25 midnight

Project Guidelines

- Students are required to work in groups of three.
- Each group is required to work on the project on their own. Any copying will be counted as cheating and treated accordingly.
- Group members are expected to put balanced effort on the development of the different requirements of the project.
- Designate one member of the group to submit the project deliverables (code and report) to Moodle.

Deliverables

- Source code of the project with documentation.
- Up to 6-page report and two appendices, as per the following:
  - Cover page listing the group members (name and ID number) and the workload distribution (in percentage of individual effort to total effort).
  - Description of the system architecture and the protocol used between the communicating entities.
  - A tabular presentation of all the project features indicating the ones that are successfully implemented.
  - Description of the implementation of the different functionalities.
  - One appendix that includes snapshots of the application depicting the main features.
  - One appendix that includes a table showing a breakdown of the project tasks. Indicate next to each task the name of the group member who was mainly responsible for its implementation.
- You are expected to perform a demo to present the application and all supported features.
- You should be ready to answer any question related to the implementation of the project.

General Description

In this project, you are required to design and implement **AUB Registrar**, an online platform to allow ECE students to register their courses for the upcoming semester.
AUB Registrar is a client server platform made of the following elements:

1. A student's client to allow students to register their courses
2. An admin client to allow a registrar admin to manage courses – only one admin
3. A server that manages the registration process

Each element is described below

I. Client – Student

1. The client program connects to the server's URL and port when the *clientStudent* Python code is launched.
2. When the client code is launched, the user should be prompted to enter his username and password.
3. In case of wrong credentials, the user should be prompted that his credentials are wrong and he/she has to enter them again.
4. In case of successful authentication, the user should get a list of his registered courses. A message "No courses registered yet" should be displayed in case no courses were already registered.
5. "List courses" command should list the courses available for registration alongside with the Maximum capacity of each course, the remaining seats and the schedule of the course.
6. Register <course name> (where <course name> should be replaced with the course name the student wants to register), should allow the student to register a course in case of:
    a. The course is not full
    b. The course schedule doesn't overlap with another course registered by the user.
    c. The number of registered courses is less than 5. The student is allowed to register in a maximum of 5 courses.
7. Withdraw <course name>, allows the student to withdraw from a course. This should update the remaining seats of the course.

II. Client - Admin

1. The client program connects to the server's URL and port when the clientAdmin Python code is launched.
2. When the client code is launched, the user should be prompted to enter his

username and password.

3. In case of wrong credentials, the user should be prompted that his credentials are wrong and he/she has to enter them again.
4. In case of successful login, the client displays the list of courses the admin provided to the students.
5. “Create course” command allows the admin to add a course. The course should have a name, schedule and maximum capacity.
6. “Update course” command allows the admin to only increase the capacity of the course.
7. “Add student” allows the admin to add a student to the system. A student should have a name, username and password.

III. Server

1. The server takes as a command line argument the port number on which it would be listening.
2. The server serves multiple clients simultaneously.
3. The admin user is created from within the code. No need to create it dynamically.

This platform uses Python as the programming language and is based on a simple yet effective design. For the purpose of your application, you need to decide whether TCP or UDP is to be used as the underlying transport layer protocol. You should ensure that the communication is reliable between the clients and the server. Please note that you may select any protocol as long as you have enough reasons to justify your choice. The aim is to develop the basic functionality using a client-server architecture.