**BTN415- Data Communications Programming**

**Lab #5**

**Title:** Web Server and Client for Inventory Management

# Objective: The goal of this lab is to create a simple web server and client application that manages an inventory of products. The server will handle user logins, inventory management, purchase requests, and user logouts, while the client will interact with the server via a basic text interface.

# Tasks:

**Server Process:**

1. **User Management**:
   * The server must maintain a hardcoded list of users with their corresponding user IDs and passwords.
   * When a user logs in, the server should authenticate the provided credentials and track logged-in users. Users can only interact with the inventory after logging in.
   * A user can log out, and the server must track their logout and update the status accordingly.
2. **Inventory Management**:
   * The server should maintain an inventory list of products, where each item is uniquely identified by a UPC code (represented as an integer).
   * The inventory list will track the quantity of each item. The quantity is fixed and will not be restocked.
   * Upon a purchase request, the server must check if the item is in stock and reduce the quantity accordingly. If the item is out of stock, the server should reject the purchase request.
   * The server must respond with a message indicating whether the purchase was successful or failed due to insufficient stock.
3. **Request Handling**:
   * **Login Request**: The client sends a request with a user ID and password. The server validates the credentials and logs the user in if the credentials are correct.
   * **Purchase Request**: A client sends a request to purchase an item. The server responds with an acknowledgment if the purchase is successful, or a failure message if the item is out of stock.
   * **Inventory Request**: The client sends a request to view the current inventory. The server responds with a list of items and their quantities, but only for logged-in users.
   * **Logout Request**: The client can log out, which removes the user's login status from the server.
4. **Persistence**:
   * Both user login statuses and inventory data must be persisted to disk. This ensures that the server can restore the data on restart.

**Client Process:**

1. **Interface**:
   * The client provides a simple, text-based interface with the following options:
     1. **Login**: The user enters their credentials to log in.
     2. **View Current Server Inventory**: The user can view the current inventory only if logged in.
     3. **View Local Inventory**: This keeps track of the items the user has purchased during the session.
     4. **Purchase Items**: The user can purchase items if they are logged in. Successful purchases are added to the local inventory.
     5. **Logout**: The user can log out, which removes their login session.
2. **Communication with Server**:
   * The client opens a socket to communicate with the server.
   * The client sends requests to the server based on the menu options selected by the user.
   * After sending each request, the client will display the response received from the server.
   * The client will store a local inventory file to keep track of items purchased during the session. This file must be saved to disk for recovery upon restart.

**Simple Protocol:**

You will need to define a basic protocol that outlines the valid interactions between the client and the server. This should include:

* How the client will send login credentials, item purchase requests, and other commands.
* The format of requests and responses.
* How errors (like invalid login or out-of-stock items) will be communicated between the client and server.

**Protocol Documentation**:  
Create a separate file to document this protocol, including the following:

* Structure of messages exchanged between client and server (e.g., login request format, purchase request format).
* Explanation of server responses (e.g., success, failure, inventory update).
* Error handling mechanisms (e.g., invalid login, purchase failure).

**Deliverables:**

1. **C++ Source Code**: Submit the C++ code for both the server and client applications.
2. **Protocol Documentation**: Submit the documentation describing the interaction protocol between the client and server.
3. **User Manual**: Provide a step-by-step guide explaining how to run both the server and the client applications, along with instructions on how to test the features.

**Grading Criteria:**

* Correct implementation of server features (login, inventory management, purchase handling).
* Correct implementation of client features (interface, sending requests, displaying responses).
* Proper handling of socket communication and data persistence.
* Clear and accurate protocol documentation.