Course:	**INFO6016 - Network Programming, Fall 2023**
Project # 1:	**Authentication Service**
Due Date:	**November 21, 2023, 11:59pm (4 weeks)**
Weight:	**15%**

Note: This project can be done in groups.

Description and Purpose

Create an authentication service that provides a way for your chat server to authenticate users. This authentication service should be reusable. You must use C++.

Details

Source Control (3 marks):

Project must be done in git, commits must be short and sweet **(3 marks)**

Authentication Protocol (7 marks):

1. Create an authentication protocol that uses Google Protocol Buffers as its serialization and deserialization method (2)
2. Must use the same .proto files on the 'Authentication Server' and the 'Authentication Client' (2)
3. Must implement a protocol similar to the one below (3)

Here is an example protocol that you may use as a reference (in pseudo code)

Note: *requestId* should be tied to the connected ChatClient to keep track of who made what request.

```
message CreateAccountWeb {
      long requestId;
      string email;
      string plaintextPassword;
```

```
}

message CreateAccountWebSuccess {
      long requestId;
      long userId;
}

message CreateAccountWebFailure {
      long requestId;
      enum reason {
            ACCOUNT_ALREADY_EXISTS;
            INVALID_PASSWORD;
            INTERNAL_SERVER_ERROR;
      }
}

message AuthenticateWeb {
      long requestId;
      string email;
      string plaintextPassword;
}

message AuthenticateWebSuccess {
      long requestId;
      long userId;
      string creationDate;
}

message AuthenticateWebFailure {
      long requestId;
      enum reason {
            INVALID_CREDENTIALS;
            INTERNAL_SERVER_ERROR;
      }
}
```

Authentication Server Database (6 marks)

1. create a table `web_auth` (1)
    a. id BIGINT AUTO_INCREMENT;
    b. email VARCHAR(255);
    c. salt CHAR(64);
    d. hashed_password CHAR(64);
    e. userId BIGINT;
2. Create a table `user` (1)

    a. id BIGINT AUTO_INCREMENT;
    b. last_login TIMESTAMP;
    c. creation_date DATETIME;
3. Export your SQL schema, and provide the .sql file (3)

ID should be your primary key, add indexes to the appropriate columns (1)

Authentication Service (17 marks):
NOTE: *This is a separate server that should be able to be reused by another program.*
NOTE: *This is **NOT** the CHAT SERVER*

Must use TCP (1 mark)

Must use length prefix header for serialization **(3 marks)**

- Must be able to create a new account **(8 marks total)**
    a. Must use SHA256 as the hash algorithm. (2)
    b. Must use a randomized salt for EACH password. (2)
    c. Must add this account to the MySQL database. (2)
    d. Must respond with a failure reason on failure. (1)
    e. Must respond with “success” on success. (1)

- Must be able to authenticate an account **(5 marks total)**
    a. Must hash the plaintext with SHA256 (1)
    b. Must compare this hash to the database hash properly (1)
    c. Must respond with a failure reason on failure (1)
    d. Must respond with “success” on success (1)
    e. Must update the `last_login` column in the `user` table in MySQL (1)

Authentication Client (7 marks)
Note: *This is ALSO the Chat Server*
Note: *This is NOT the chat client*

1. Must use TCP (1)
2. Must use length prefix header for serialization (3)
3. Must connect to the authentication service (1)
4. Should be able to create a new account (1)
5. Should be able to authenticate a user (1)

Chat Client (6 marks)

1. Should be able to send a command: REGISTER email password (1)

2. If registration was successful, it should tell the client: “Registration successful” (1)
3. If the registration failed, it should tell the client the reason for the failure (1)
4. Should be able to send a command: AUTHENTICATE email password (1)
5. If authentication was successful, it should say “Authentication successful, account created on [DATE IN DATABASE]” (1)
6. If authentication failed, it should tell the client the reason for the failure (1)

Important note: Do not forget to include all dependencies within your project, you might want to use your project directory for includes and libs.
i.e.
$(ProjectDir)includes
$(ProjectDir)lib

Plagiarism

- While you may freely “borrow” mine (or anyone else’s) code, ***but*** your code should be “sufficiently” different from mine.

- In other words, you *cannot* simply use an existing game engine (or part of a game engine) to complete this assignment; it should be either completely new or **significantly modified**.

Project Submission

The following are **required** for submitting your project:

- ReadMe.txt
  - Describe how to build your project.
  - Describe how to run your project.
  - Describe the user input options.

- Video.mp4
  - Upload a video of you demoing your project.
  - You may include the video with your project submission or upload to YouTube and provide a link.
- Git
  - If you prefer to submit your project through Github you may. You just need to send me a link to your github repository.

Project Corrections

If any corrections or changes are necessary they will be posted to the course web site and you will be notified of any changes in class. It is your responsibility to check the site periodically for changes to the project. Additional resources relating to the project may also be posted.