1. **About the Project**

The 5G Data Subscriber Management System is a project developed as part of a research initiative under Prof. Guanhua at Binghamton University. Initially built using Rust and MongoDB, the project was later recreated using Angular, MySQL, and APIs to enhance database performance and scalability.

The system is designed to manage subscriber data for a 5G testbed, allowing users to perform various operations like adding, viewing, and updating subscriber details. It includes functionalities to manually add subscriber data or generate random subscriber entries. Additionally, the project was designed to facilitate SQL injection simulations, enabling academic experiments related to cybersecurity.

Key features of the project include:

- A user-friendly Angular frontend for managing subscriber data.
- MySQL as the backend database for efficient storage and querying of data.
- Support for generating random subscriber data for testing purposes.
- The inclusion of SQL injection vulnerability simulations for educational purposes, allowing students to test and understand SQL injection attacks.

2. How to Perform the SQL Injection Attack

The 5G Data Subscriber Management System includes a simulated SQL injection vulnerability for educational purposes. Here's how you can perform an SQL injection attack that specifically targets the Password field:

Login Page Attack:

- Navigate to the Login Page of the system.
- In the Username field, enter any valid or invalid username (e.g., testuser).
- In the Password field, inject the following SQL code:

  Password :   ' OR '1'='1

  This injection works because the **Password** field does not properly validate input. The `OR '1'='1` clause always returns true, bypassing authentication and allowing unauthorized access to the system without knowing the actual password.

Important Note: This simulation should be carried out only in a controlled, academic setting to learn the dangers of SQL injection and how to prevent such vulnerabilities in real-world applications. The purpose is to demonstrate how unchecked inputs can lead to security breaches.

Mitigation Strategies:

To prevent SQL injection attacks, it is crucial to use prepared statements (parameterized queries) and thoroughly validate all user inputs. Additionally, use error handling techniques that do not reveal sensitive database information.

3. **How to Implement the Database on XAMPP**

To implement the 5G Data Subscriber Management System's database using XAMPP and MySQL, follow these steps:

Install XAMPP:

- Download and install XAMPP from the Apache Friends website.
- Open the XAMPP control panel and start the **Apache** and **MySQL** services.

Open the XAMPP Shell:

- In the XAMPP control panel, click on the **Shell** button on the right-hand side of the screen.
- A command-line window will open. In this window, type the following command to access the MySQL shell:

```
mysql -u root -p
```

- Press Enter. If prompted for a password, leave it blank (or enter the MySQL root password if you've set one).

Create the Database:

- In the MySQL shell, type the following command to create a new database named Guanhua:

```
CREATE DATABASE Guanhua;
```

- After executing the command, the database will be created.

Use the Database:

- Switch to the **Guanhua** database to start creating tables by typing:
  ```
  USE Guanhua;
  ```

Create the Necessary Tables:

- Now, create the tables required for the project by typing the following SQL commands in sequence:

Users Table (for login credentials):

```
CREATE TABLE users ( username VARCHAR(255) DEFAULT NULL, password
VARCHAR(255) DEFAULT NULL );
```

Subscribers Table (for storing subscriber data):

```
CREATE TABLE subscribers ( id INT AUTO_INCREMENT PRIMARY KEY,
supi VARCHAR(255) NOT NULL );
```

Security Table (for storing security data):

```
CREATE TABLE security ( id INT AUTO_INCREMENT PRIMARY KEY, subscriber_id
INT NOT NULL, `key` VARCHAR(255) NOT NULL DEFAULT
'fec86ba6eb707ed08905757b1bb44b8f', opc VARCHAR(255) NOT NULL DEFAULT
'c42449363bbad02b66d16bc975d77cc1', sqn_hn INT NOT NULL DEFAULT 33,
FOREIGN KEY (subscriber_id) REFERENCES subscribers(id) );
```

AM Data Table (for storing 5G network configurations):

```
CREATE TABLE am_data ( id INT AUTO_INCREMENT PRIMARY KEY, subscriber_id
INT NOT NULL, uplink VARCHAR(255) NOT NULL DEFAULT '104857600', downlink
VARCHAR(255) NOT NULL DEFAULT '104857600', supported_features
VARCHAR(255) NOT NULL DEFAULT 'SMS', sst INT NOT NULL DEFAULT 1, sd
VARCHAR(255) NOT NULL DEFAULT '000001', FOREIGN KEY (subscriber_id)
REFERENCES subscribers(id) );
```

SMF Selection Data Table (for storing Session Management Function data with default values):

```
CREATE TABLE smf_sel_data ( id INT AUTO_INCREMENT PRIMARY KEY,
subscriber_id INT NOT NULL, dnn VARCHAR(255) NOT NULL DEFAULT 'internet',
default_dnn_indicator BOOLEAN NOT NULL DEFAULT TRUE, FOREIGN KEY
(subscriber_id) REFERENCES subscribers(id) );
```

SM Data Table (for storing additional 5G network configurations):

```
CREATE TABLE sm_data ( id INT AUTO_INCREMENT PRIMARY KEY, subscriber_id
INT NOT NULL, sst INT NOT NULL DEFAULT 1, sd VARCHAR(255) NOT NULL
DEFAULT '000001', internet_pdu_session_type VARCHAR(255) NOT NULL DEFAULT
'IPV4', internet_ssc_mode VARCHAR(255) NOT NULL DEFAULT 'SSC_MODE_1',
session_uplink VARCHAR(255) NOT NULL DEFAULT '81920000', session_downlink
VARCHAR(255) NOT NULL DEFAULT '81920000', FOREIGN KEY (subscriber_id)
REFERENCES subscribers(id) );
```

Configure Database Connection in Angular Project:

- Update the **environment configuration** in your Angular project's backend to connect to the MySQL database running on XAMPP.
- Ensure the connection string points to `localhost:3306`, with the correct database credentials (username, password).

Example connection string (Node.js/Express backend):

```
const mysql = require('mysql');

const connection = mysql.createConnection({

    host: 'localhost',

    user: 'root',

    password: '', // Add your MySQL password here if applicable

    database: 'Guanhua'

});
```

By following these steps, you will have a fully configured MySQL database running on **XAMPP**, which is connected to your Angular project for managing 5G subscriber data.

4. **How to Run the Backend Code and Prerequisites**

Before running the backend code for the 5G Data Subscriber Management System, ensure you have the following environment and dependencies set up.

Prerequisites

1. **Node.js**:
   - Download and install **Node.js** from the official site: Node.js Download.
   - Ensure you have at least version 14 or above. You can check your Node.js version by running the following command in your terminal or command prompt:
     ```
     node -v
     ```
2. **NPM (Node Package Manager)**:
   - NPM is automatically installed with Node.js. You can verify if it's installed by running:
     ```
     npm -v
     ```
3. **MySQL**:
   - Ensure MySQL is running using **XAMPP** or another local MySQL server. You should have already set up the `Guanhua` database as explained in the previous section.

Backend Code Setup

1. **Navigate to the Backend Directory**:
   - Open a terminal and navigate to the `backend` folder where your backend code is located:
     ```
     cd <path_to_backend_directory>
     ```
2. **Install Dependencies**:
   - Once inside the `backend` folder, run the following command to install all the required dependencies listed in the `package.json` file:
     ```
     npm install
     ```
   - This command will install all the necessary Node.js packages, such as Express, MySQL libraries, and any other modules required for the backend.
3. **MySQL Database Connection**:

- In the `server.js` file, you’ve already written the MySQL connection code. Ensure that the connection details (host, user, password, and database name) match the settings of your XAMPP MySQL server.

  Example code from `server.js`:

  ```
  const mysql = require('mysql');
  const connection = mysql.createConnection({
      host: 'localhost',
      user: 'root',
      password: '',  // Use your MySQL password if
  applicable
      database: 'Guanhua'
  });

  connection.connect((err) => {
      if (err) {
          console.error('Error connecting to MySQL: ' +
  err.stack);
          return;
      }
      console.log('Connected to MySQL as ID ' +
  connection.threadId);
  });
  ```

4. **Run the Backend Server**:
    - To start the backend server, run the following command:
      `node server.js`
    - This will start the backend on the default port configured in `server.js` (commonly `http://localhost:3000/` or `http://localhost:8080/`). You should see output in the terminal indicating that the server is running.
5. **Test the Backend**:
    - Open any web browser (such as **Chrome**, **Firefox**, **Edge**, etc.).

- Type the following URL in the browser's address bar to see the subscriber data:
  `http://localhost:3000/api/subscribers`
- This will display all the data in the **subscribers** table directly on the web.

By following these steps, you will have the backend up and running, fully connected to the MySQL database. Make sure the backend is running before launching the Angular frontend to ensure the entire system works seamlessly.

5. **How to Run the Angular Project (simple-form-project) with Bootstrap and Additional Dependencies.**

Before running the Angular project **simple-form-project**, ensure that the following tools, dependencies, and configurations are properly set up on the system.

Prerequisites

1. **Node.js**:
   - Install **Node.js** from Node.js Download. Ensure you are using version 14 or above.
2. **Angular CLI**:
   - Install Angular CLI globally by running:
     npm install -g @angular/cli
   - Verify the installation by running:
     ng version
3. **NPM**:
   - NPM is installed with Node.js. Confirm it by running:
     npm -v
4. **Additional Dependencies**:
   - When setting up the backend and frontend, the following packages were installed, and you need to install them if you're setting up the project on a new system:
     - **Express**: To handle server-side routing.
     - **MySQL**: To connect to the MySQL database.
     - **CORS**: To enable cross-origin requests.
     - **Body-Parser**: To handle request bodies.

You can install these packages by running:
npm install express mysql cors body-parser

Running the Angular Project

1. **Navigate to the Project Directory**:
    - Open a terminal and navigate to the project folder named simple-form-project:
      ```
      cd <path_to_project_directory>/simple-form-project
      ```
2. **Install Project Dependencies**:
    - Inside the simple-form-project folder, install all necessary dependencies by running:
      ```
      npm install
      ```
    - This will install all required packages for Angular, Express, MySQL, CORS, and Body-Parser.
3. **Install Bootstrap** (if needed):
    - If Bootstrap is not already installed, you can install it by running:
      ```
      npm install bootstrap
      ```
    - After installation, ensure that Bootstrap is included in the **angular.json** file under the **styles** section:
      ```
      "styles": [
       "node_modules/bootstrap/dist/css/bootstrap.min.css",
       "src/styles.css"
      ]
      ```
4. **Run the Angular Application**:
    - To run the application, use the following command:
      ```
      ng serve
      ```
    - The Angular app will compile and start running on the default port, typically at http://localhost:4200/.
5. **Open the Application in a Browser**:
    - Open any browser and navigate to:
      http://localhost:4200/
    - You should now see the frontend of the 5G Data Subscriber Management System, with Bootstrap styling applied.

Ensuring Backend is Running

- Ensure the backend is running by following the instructions in **Section 4**.
- The Angular frontend will make API calls to the backend, so both the frontend and backend should be running simultaneously.

By following these steps, the **Angular frontend** with **Bootstrap** and the additional **Express, MySQL, CORS, and Body-Parser** dependencies will be running, connected to the backend and database. Ensure all dependencies are installed properly for full functionality.