ECALL Emergency Response System Project

Project Title: ECALL Emergency Response System

Project Description:

Students will develop a simulated ECALL System that detects an accident and sends emergency notifications. The system will focus on applying OOP principles, Modern C++ features, and some basic networking or file-handling techniques.

Project Requirements:

1. Class Structure:

   - Create a Vehicle class with properties such as speed, location, and acceleration.
   - Create an ECALLSystem class that manages emergency notifications.
   - Create a NotificationService class to handle interactions with emergency services.

2. Core Functionality:

   - Implement accident detection based on sudden deceleration (e.g., if the speed goes from a high value to zero in a short time).
   - Add a method in the ECALLSystem class to initiate an emergency call when an accident is detected.
   - Simulate sending data such as vehicle location, time of the accident, and other relevant information to the NotificationService.

3. OOP Concepts:

   - Use encapsulation to manage the vehicle's state (e.g., private variables for speed and location with public getters and setters).

- Use inheritance to create specific types of vehicles (e.g., Car, Truck), where each vehicle may handle accident detection slightly differently.
- Use polymorphism for handling various types of notifications (e.g., SMS, email) in NotificationService.

4. Modern C++ Features:

- Use smart pointers (std::unique_ptr or std::shared_ptr) for managing memory, particularly for objects like Vehicle and NotificationService.
- Implement lambda functions for simple event handling, such as logging messages or triggering emergency actions.
- Use std::vector or std::map to store multiple vehicles and their associated ECALL systems.

5. Error Handling:

- Implement error handling for invalid input data (e.g., negative speeds) and simulate network errors when sending notifications.

6. Data Management:

- Save accident data to a file for logging purposes. The log should include details such as vehicle ID, accident time, speed, and location.
- Optionally, allow loading this data to recreate a crash scenario for review.

7. Optional Features (for advanced students):

- Integrate basic networking capabilities using sockets to simulate the ECALL system sending information to a remote server.
- Create a mock GPS system to generate realistic vehicle location data.

- Use the Observer design pattern to notify the system's components whenever an accident is detected.

8. Data Logging:

- Save ECALL Events: Each time an accident is detected, the ECALLSystem should log the event by saving it to a file. This log file should include:
  - Date and Time of the accident.
  - Vehicle Data such as:
    - Vehicle ID
    - Speed at the time of the accident
    - Location (latitude and longitude)
    - Acceleration data
- File Format: The data should be saved in a structured format, such as JSON or CSV, to make it easy to read and process later.
- Reload Functionality: Optionally, add functionality to read and load accident events from this file, allowing the system to recreate and analyze past events.

9. Mock Data Entry through Terminal:

- Implement a terminal-based interface for the Vehicle class, allowing users to input mock data such as:
  - Speed (in km/h or mph)
  - Location (latitude and longitude)
  - Acceleration (m/s^2)
- The terminal interface should guide users with prompts, for example:

  Enter vehicle speed:

  Enter vehicle location (latitude, longitude):

Enter acceleration:

- Use the input data to simulate different driving conditions. Based on the speed and acceleration, detect whether an accident has occurred by comparing against predefined thresholds.
- Display real-time feedback after each input. For example:

```
Speed: 100 km/h, Acceleration: -9.8 m/s²
Accident detected. Initiating emergency call...
```

- Automate Accident Simulation: To further enhance the terminal interface, allow users to enter a sequence of values for speed and acceleration over time to simulate driving scenarios. The system should automatically detect when an accident threshold is met and activate the ECALL feature.