Assignment: School Management System (SMS)

Objective:
Develop a PHP web application to manage a primary school system following the outlined rules and specifications. The project will require a strong understanding of Object-Oriented Programming (OOP), the Model-View-Controller (MVC) architecture, and effective collaboration within a team.

Project Requirements

System Features

Your application must include the following functionalities:

1. **User and User Role Management**
    - Administer roles for:
        - Teachers (manage scores only)
        - Office Administrators (manage all other functionalities)
    - Allow CRUD (Create, Read, Update, Delete) operations for users and their roles.
2. **Grade Management**
    - Predefined grades (one to six) with no possibility of modification to names.
3. **Class Management**
    - Each grade should have between 1 and 6 classes.
    - Classes are uniquely named and cannot switch grades.
4. **Student Management**
    - Allow placement of students into specific classes.
    - A student must belong to only one class at a time.
5. **Subject Management**

- Subjects are shared across all classes in the same grade.

6. **Score Management**

   - Teachers should manage scores for students:
     - Scores are recorded for each term (three terms per year).
     - Scores must be tied to subjects.

7. **School Year and Term Management**

   - Define and manage school years and terms to ensure scores are properly recorded.

Reports

1. **Student Report Card**
   - Generate a report card for a specific student for a selected term.
2. **Average Performance Report**
   - Calculate and display the average performance of students by grade for specific subjects.

Technical Requirements

1. **Development Framework**

   - Implement in pure PHP using OOP principles.
   - Use the MVC architecture for organization.
   - The application must run on XAMPP and should not rely on external frameworks.

2. **Data Validation and Security**

   - Input validation and sanitation on both front-end and back-end.
   - Prevent SQL injection, XSS, and other common security vulnerabilities.

3. **User Interface**

   - Design a user-friendly interface for seamless navigation.
   - Separate access and permissions based on user roles.

Grading Rubric

Criteria	**Weight**
Application and correctness of OOP	15%
Application and correctness of MVC	15%
User input validation and sanitation	15%
User-friendliness and UI design	15%
Working features	40%

Group Composition

- **Team Size:** 8 members.
- Collaboration and teamwork are essential for success.

Submission Requirements

Submit the following in a compressed file named **sms.zip**:

1. **README.txt**
   - Detailed instructions for setting up and running the application on XAMPP.
2. **database.sql**
   - SQL dump of a well-populated database (include data for users, grades, classes, students, subjects, scores, etc.).
3. **Link.txt**
   - A YouTube link to a video demonstration of the application. Only features shown in the video will be marked.
4. **Sms.zip**
   - A folder containing the complete PHP application.
5. **Contribution.pdf**
   - A signed and dated group contribution report specifying each member's contributions.

Penalties

- **Late Submission:** A 25% reduction per day will be applied to late submissions. No exceptions or extensions will be granted.

Recommendations for Success

1. **Divide Responsibilities:**
   - Assign group members to specific modules (e.g., user management, scores, reports, etc.).
2. **Use Version Control:**
   - Employ Git to manage code changes collaboratively.

3. **Thorough Testing:**
   - Test all features for correctness, security, and usability before submission.
4. **Presentation:**
   - Plan a clear and concise presentation video focusing on key features.