Full Site Implementation – Assignment Guidelines

This assignment requires you to build and deploy a fully functional PHP + MySQL web application on the school's student server. Your system must demonstrate CRUD operations, secure coding practices, search functionality, and optional advanced features such as Ajax and template engines.

1. Assignment Objective

The goal is to design and implement a complete dynamic website that interacts with a MySQL database. Students must apply backend development concepts, database handling, PHP programming, and secure coding practices. The final product should resemble a real-world, fully working web application.

2. Core Requirements

Your project must include the following components:

• Use PHP (backend) and MySQL (database) for all application logic.
• Host the website on the school's student server.
• Implement full CRUD functionality:
  – Create new records
  – Read and display records
  – Update existing records
  – Delete records
• Include a functional search feature. Advanced searches (multiple criteria) earn extra points.
• Provide protection against common security vulnerabilities:
  – SQL Injection (through prepared statements)
  – XSS (through proper output escaping)
  – CSRF protection is optional but recommended
• Use Ajax for at least one useful feature:
  – Autocomplete search
  – Live form validation
  – Fetching data without page reload
• (Optional) Use a template engine such as Twig or Smarty to separate presentation from logic.

3. Suggested System Structure

Below is a recommended folder structure for your project:

```
project_root/
├── config/
│   └── db.php
├── public/
```

```
│   ├── index.php
│   ├── add.php
│   ├── edit.php
│   ├── delete.php
│   └── search.php
├── templates/ (if using template engine)
├── assets/
│   ├── css/
│   └── js/
└── includes/
    ├── header.php
    ├── footer.php
    └── functions.php
```

4. Step-by-Step Implementation Guidelines

4.1 Database Setup

1. Identify the main data entity (books, movies, students, products, etc.).
2. Design a MySQL table with appropriate fields.
3. Create the table using phpMyAdmin or SQL scripts.
4. Test database connectivity using a simple PHP connection script.

4.2 CRUD Functionality

Your website must allow users to:

- Add new entries through a form (use POST method).
- View all entries in a table layout.
- Edit entries using pre-filled forms.
- Delete entries with confirmation prompts.

All database operations must use prepared statements to prevent SQL injection.

4.3 Search Feature

The search page should allow users to query the database. Examples:

- Simple Search: Search by title or keyword.
- Advanced Search: Search by multiple criteria such as category, year, price range, etc.
- Ajax Search: Provide autocomplete as the user types.

4.4 Security Requirements

You must implement basic security measures:

- SQL Injection Prevention: Use prepared statements.
- XSS Prevention: Escape all output using htmlspecialchars().
- Form validation on both client and server side.
- Validate file uploads if your project includes them.

4.5 Ajax Integration

Your website must include at least one useful Ajax feature:

• Autocomplete search bar
• Live validation (e.g., check if email or username exists)
• Loading records without page refresh

Use JavaScript Fetch API or XMLHttpRequest.

4.6 Using a Template Engine (Optional)

For additional points, integrate a template engine like Twig or Smarty to separate UI from PHP logic.

5. Example Project Ideas

Click here for project list

6. Submission Requirements

Your submission must include:

• A working website hosted on the student server
• A zip file of your full project directory
• A SQL file containing your database structure and sample data
• A readme document with:
  – Login credentials
  – Setup instructions
  – List of features implemented
  – Known issues (if any)