Full Project Explanation – Web Application with MySQL, PHP, and MAMP

1. Introduction

I created a website using MySQL, PHP, HTML, and CSS. I used MAMP on my Mac to run the project locally. The purpose of the project was to manage a school system with different users such as admin, teachers, and students. Each user has their own dashboard and can perform different tasks depending on their role. The website allows teachers to manage classes, students, assignments, and attendance. Students can view their own grades and assignments. The admin can manage users, create accounts, and manage classes.

2. How I Set Up the Database

The database was made using MySQL. I created several tables to store and organize data. These tables include Users, Teachers, Students, Classes, Assignments, Grades, and Attendance. The Users table stores login details and roles (such as teacher, student, or admin). The Teachers and Students tables are linked to the Users table using IDs such as teacher_id or student_id.

Each teacher can have multiple classes, and each class can have many students. I used foreign keys to link these tables. For example, the Classes table has a teacher_id so it's easy to know which teacher teaches which class. The Attendance table stores class_id, student_id, date, and the status like Present or Absent. This setup keeps the database organized and makes it easy to manage all the information in a connected way.

3. How the Web App Interacts with the Database

To work with the database, I used PHP and MySQL. The PHP code inserts new records, updates data, deletes records, or shows data from the database depending on what the user does.

For example, when a teacher marks attendance, the system checks if attendance already exists for that student on that date. If it does, the record is updated. If not, a new record is added. I used HTML forms to collect user input, such as selecting a class or entering grades. These forms use the POST or GET method to send data to the server. The PHP script receives this data and interacts with the database.

4. Displaying Information on Web Pages

On the web pages, PHP is used to show data from the database. I used loops to display lists of students, assignments, or grades. There are also filters, like choosing a date or entering a student's name, so teachers can quickly find specific records. The website updates the information based on the user's actions, but without JavaScript, it works by submitting forms and loading pages after each action.

5. Setting Up the Local Server with MAMP (Mac)

I used MAMP on my Mac to create a local server. MAMP includes Apache (the web server), PHP (to run the code), and MySQL (the database). After installing MAMP, I placed all my project files (HTML, PHP, CSS) into the `htdocs` folder. Then I opened MAMP and started the Apache and MySQL servers.

To see the website, I opened a browser and typed: `http://localhost:8888/schoolvibes2/School-Management/`. I used phpMyAdmin to create and manage my database. It has a simple visual interface that lets me create tables, insert records, and run SQL queries. This local setup helped me test everything safely on my own computer before publishing the site online.

6. Features and User Dashboards

This web application shows different content on the pages depending on what the user does and the data in the database. When users log in, they are taken to their own dashboard based on their role:

- **Admin Dashboard:** The admin can manage all users, view all data, and update or delete records. The admin can also assign roles and manage classes to keep the system running smoothly.
- **Teacher Dashboard:** Teachers only see their own classes and students. They can take attendance, give grades, upload assignments, and search for student records.
- **Student Dashboard:** Students only see their own assignments, grades, and attendance records. They cannot edit or add data, only view it.

I used PHP sessions to keep track of which user is logged in and what they are allowed to see. The PHP code checks the user's role and displays the correct content. This helps secure the site and personalize the user experience.

7. Frontend: HTML and CSS

I used HTML to create the structure of the web pages and CSS to style them. The design is simple, clean, and easy to use for both teachers and students. CSS helps make the interface visually appealing and organized. Since I did not use JavaScript, all actions like submitting forms or filtering data require page reloads, which the PHP code handles smoothly.

8. Handling User Data Safely

When a user logs in, the system checks their username and password using PHP and prepared statements. If the login is correct, they are sent to the correct dashboard. The system only

shows data related to that user. For security, I always used prepared statements when handling user inputs. This prevents attackers from inserting harmful data into the database. I also ensured that all data sent from forms is cleaned before updating the database.

9. IPO Test Cases

Test Case #	Test Action	Expected Output	Actual Output	Pass/Fail
TC1	Login with correct teacher credentials	Redirect to teacher dashboard	Redirected to teacher dashboard	Pass
TC2	Login with wrong password	Show error message: "Invalid username or password"	Error message displayed	Pass
TC3	Teacher views their assigned classes	List of classes shown	Correct classes displayed	Pass
TC4	Teacher selects class and date for attendance	Student list displayed with dropdown for status	Correct student list and dropdown shown	Pass
TC5	Teacher submits attendance	Attendance saved, message: "Attendance saved successfully."	Attendance saved, success message shown	Pass
TC6	Teacher filters attendance history by date	Table shows filtered records	Correct filtered records shown	Pass
TC7	Teacher filters attendance by student name	Table shows matching student records	Correct records for searched name shown	Pass
TC8	Teacher views list of assignments they created	Table lists assignments with class name, title, deadline	Assignment list shown accurately	Pass
TC9	Teacher accesses grades page for an assignment	List of students and grade input fields shown	Correct student list with grade inputs shown	Pass
TC10	Teacher submits grades for an assignment	Grades saved, confirmation message shown	Grades saved in database, message shown	Pass

TC11	Student logs in and views their grades	Table showing class, assignment title, and grade	Accurate grade details shown	Pass
TC12	Admin adds new user (teacher or student)	User added to database; confirmation message	User added successfully	Pass
TC13	Attempt SQL injection in login form	Login fails; no access given	Login failed; input sanitized	Pass
TC14	Teacher logs out	Redirect to login page;	Redirected, session cleared	Pass
TC15	Teacher tries to submit attendance without selecting class	Error message or form not submitted	No submission; user prompted to select class	Pass
TC16	Teacher submits attendance twice for the same student/date	Status updated; no duplicate entry	Attendance updated, no duplicates	Pass
TC17	Teacher tries to access student page directly via URL	Access denied or redirected	Access denied; redirected to login or dashboard	Pass
TC18	Admin deletes a class	Class removed; related data (e.g., attendance) also removed	Class deleted; data handled correctly	Pass
TC19	Student submits assignment (file upload or text)	Assignment recorded; confirmation shown	Submission saved and shown in student portal	Pass