



---

# PHP & DATABASES

---

Task 3 – Evaluation and Testing



MONICA ARONG DARMNIN  
INTERACTIVE DIGITAL MEDIA

## Virtual Server Setup

The virtual server was set up locally using XAMPP. XAMPP is a web development suite of programs bundled together to simplify the installation of the various components. It includes Apache with PHP which acts as a web server and MySQL, which acts like a database server. This allows the web pages to be hosted and accessed locally through the browser. The project files were stored in the htdocs directory, and the server was started through the XAMPP control panel, launching MySQL and Apache. XAMPP was installed by downloading the installer from <https://www.apachefriends.org/download.html>.

Once the server was running, I accessed the web browser at <http://localhost:8080/> followed by the folder name of my repository project.

## Database Set Up

A new database called 2025-schoolweb was created using PhpMyAdmin. PhpMyAdmin is also included with XAMPP to enable the management of the databases. The databases was used to store all data required for the school management system in a structured and effective way.

Multiple tables were created to represent different components of the system. This stores relevant data such as users, roles, units, courses, assignments, submissions, files, and grades.

The database was designed using techniques to reduce data redundancy and to improve consistency. Constraints such as NOT NULL and AUTO\_INCREMENT enforced valid entry, while using primary and foreign keys improved query performance. Some foreign keys were created as CASCADE ON DELETE so that tables are cleared once the parent is deleted. Each table was designed with its appropriate fields and data types, such as integers for primary keys and text-based fields for names, descriptions and other contents.

Primary keys were used to uniquely identify each record, and relationships between tables were established using foreign keys where required. This relational structure ensures data integrity and reduces duplication allows the system to retrieve and manage related data efficiently.

## Techniques Used to Build a Dynamic Web Application

PHP was the main function to generate web pages dynamically, based on the user input and database content. An includes folder was created to keep code organised. This folder stores code functions that can be inserted into multiple PHP scripts, as well as the PHP files to control the user interaction. A functions.php file was created to handle SQL queries and to contain reusable functions. The dbh.php file (database handler) was used to manage connections and access to the database table. If a change was needed to the database connection such as password, username or database name, it is done in the dbh.php only.

To maintain user authorisation and authentication between pages, PHP sessions were utilised. This ensures that users can securely log in and maintain their session while navigating through

the site. This allowed the system to differentiate between the users role of students, lecturers and administrators, ensuring that the pages were only accessible to the relevant role.

HTML Forms were used to collect user data, such as user input, such as login details, assignment submissions and other academic and administrative details, ensuring that pages are updated automatically based on the stored data.

Conditional statements were used to dynamically display content based on user permissions. PHP include() statements were utilised to reduce code duplication and improve maintainability by utilising reusable elements like the header, the footer, and the navigation menu.

## Techniques to Manipulate Data

The application performs using CRUD (Create, Read, Update, Delete) operations using SQL queries executed through PHP. This inserts new records, updates existing data, remove record and retrieves information when required. In some select statements, some tables were joined together so that data is read in a single query. Some select statements use the ORDER BY to sort the data to display it efficiently on the screen. User input is validated to prevent invalid or malicious data from being stored in the database.

To maintain data integrity, foreign key constraints with cascading action were implemented in the database tables. For example, when deleting a course, it automatically removes any related child records where necessary, ensuring consistent and accurate data throughout the system. The username role was created with CASCADE ON UPDATE so, when the username was used as a foreign key, it would be updated automatically.

This interaction enables real-time data manipulation, which keeps the web application current and responsive to user input while preserving data consistency and data accuracy. Hashing techniques were used to securely store sensitive data, including passwords, so enhancing system security. These methods ensure reliable, secure, and real-time interaction between the database and the web application.

# TEST CASES

Test Case Type	Expected Output	Actual Output	Pass/Fail	Comments
<b>Login Page</b>				
User enters a valid login and password for the admin user.	The user should be redirected to the admin page	User is redirected to the admin page.	Pass	
User enters a valid login and password for the lecturer user.	The user should be redirected to the lecturer page	User is redirected to the lecturer page.	Pass	
User enters a valid login and password for the student user.	The user should be redirected to the student page	User is redirected to the student page.	Pass	
User enters an invalid username or password.	Shows an error message "Incorrect Login"	Error message is shown	Pass	
<b>Admin Page: Register User</b>				
Enter the new login, password, and details, and select the role(s).	The user is created with the proper details, and a success message is shown	The user is created with the proper details, and a success message is shown	Pass	
Create a new account with the same username.	The user should not be created.	User was created	Fail	Fixed with <code>if(userExists(\$conn, \$username)){ \$error = \$error."userExists=true&amp;";}</code>
Create a new user profile, but leave some fields empty.	Shows that all fields are required.	Shows that all fields are required.	Pass	
<b>Admin Page: Manage Users</b>				
User navigates to the <b>Manage Users</b> .	The <b>Manage Users</b> should show the list of users created	The <b>Manage Users</b> shows the list of users created	Pass	
User chooses a user and clicks on the edit icon.	It should navigate to the form where the user can edit the contents.	The form is shown	Pass	
User edits the contents of the selected user profile and saves.	The user profile is updated with the newly edited fields, and a success message is displayed.	The user profile is updated with the newly edited fields, and a success message is displayed.	Pass	
User chooses a user profile and clicks on delete.	It should be deleted and show an error message.	The user is deleted, and an error message is displayed.	Fail	In order for this function to pass, a confirmation alert stating "Are you sure you want to delete?" needs to be implemented before the delete action is performed.
<b>Admin Page: Enrolment Management Course</b>				
User creates a new course and adds its contents.	The course is created, and a success message is shown.	The course is created, and a success message is shown.	Pass	
User edits the course contents.	The course is edited, and a success message is shown	The success message was shown, but the course was not edited, and the values stayed the same.	Fail	<code>saveCourse()</code> was never called in the functions. It was then created and passed the <code>\$courseId</code> variable in the <code>course-inc.php</code>

User navigates to the <b>Manage Courses</b> and clicks on the delete icon.	It should be deleted and show a success message.	It should be deleted and show a success message.	<b>Fail</b>	In order for this function to pass, a confirmation alert stating “Are you sure you want to delete?” needs to be implemented before the delete action is performed.
User navigates to the <b>Manage Courses</b> and clicks on the edit icon.	It should navigate to the form where the user can edit its contents.	The form is shown	<b>Pass</b>	
User edits the contents of the selected course and saves.	The user course is updated with the new edited fields and shows a success message	The user course is updated with the new edited fields and shows a success message	<b>Pass</b>	
User created a course with empty fields.	It shows a message that required fields must be filled.	It shows a message that required fields must be filled.	<b>Pass</b>	

### Admin Page: Enrolment Management Unit

User creates a new unit, adds its contents, and links it to the particular course.	The unit is created, and a success message is shown.	The unit is created, and a success message is shown.	<b>Pass</b>	
User edits the unit contents.	The unit is edited, and a success message is shown	The unit is edited, and a success message is shown	<b>Pass</b>	
User navigates to the <b>Manage Unit</b> and clicks on the delete icon.	It should be deleted and show an error message	It should be deleted and show an error message	<b>Fail</b>	In order for this function to pass, a confirmation alert stating “Are you sure you want to delete?” needs to be implemented before the delete action is performed.
User creates a unit with empty fields.	It shows a message that there are missing fields.	It shows a message that there are missing fields.	<b>Pass</b>	
Admins should not be assigned to a unit.	The unit icon should not be displayed.	The unit icon is displayed.	<b>Fail</b>	This was fixed by adding <code>&lt;?php if(\$user\\["roleId"] != 3):?&gt;</code> to enforce that the unit icon is only displayed to the student and lecturers

### Admin Page: Class Management

User navigates to <b>Add Class</b> .	A form is shown to add a class its unit and description.	A form is shown to add a class its unit and description.	<b>Pass</b>	
User navigates, fills in the details of the class form and saves.	Details are filled, and a success message is shown	Details are filled, and a success message is shown	<b>Pass</b>	
User navigates to <b>Manage Class</b> .	A list of classes is shown.	A list of classes is shown	<b>Pass</b>	
User clicks on the edit action to edit the class fields.	A form is shown to edit the class, unit, and descriptions, and a success message is shown.	A form is shown to edit the class, unit, and descriptions, and a success message is shown	<b>Pass</b>	
User clicks on the delete action to delete a class.	The class is deleted, and success message is shown.	The class is deleted, and success message is shown.	<b>Fail</b>	In order for this function to pass, a confirmation alert stating “Are you sure you want to delete?” needs to be implemented before the delete action is performed.

### Admin Page: Timetable Management

User navigates to <b>Add Timetable.</b>	A form is shown to add the class name, unit, lecturer, room, day, start time and end time.	A form is shown to add the class name, unit, lecturer, room, day, start time and end time.	Pass	
User fills in the timetable form and clicks on save.	Details are filled, and a success message is shown.	Details are filled, and a success message is shown.	Pass	
User navigates to <b>Manage Timetables.</b>	A list of timetables is shown.	A list of timetables is shown.	Pass	
User clicks on the edit action to edit the timetable fields.	A form is shown to edit the class name, unit, lecturer, room, day, start time and end time and success message is shown	A form is shown to edit the class name, unit, lecturer, room, day, start time and end time and success message is shown	Pass	
User clicks on the delete action to delete the timetable.	The timetable is deleted, and a success message is shown.	The timetable is deleted, and a success message is shown.	Fail	In order for this function to pass, a confirmation alert stating “Are you sure you want to delete?” needs to be implemented before the delete action is performed.

## TEST CASES

Test Case Type	Expected Output	Actual Output	Pass/Fail	Comments
<b>Lecturer Page: Assignment Management</b>				
User navigates through the <b>Add Assignment.</b>	A form should be shown where the lecturer can select a course and a list of units applied to that course.	Form is shown, and the selected course shows its appropriate units	Pass	
User creates an assignment and uploads its contents.	The form is submitted, and a success message is shown	The form is submitted, and a success message is shown	Pass	
User navigates to the assignment to show the list of assignments created	It shows the list of assignments created with icons to edit or delete	It shows the list of assignments created with icons to edit or delete	Pass	
User clicks on the edit button to edit an assignment and update its contents.	The user can edit the assignment contents and save	The user can edit the assignment contents and save, however, the contents were missing.	Fail	I needed to fix the input hidden and remove the assignment Id from the functions.
<b>Lecturer Page: Grading Management</b>				
User views the grading list of assignments submitted by the students.	The assignments ready to be graded show in a list.	The assignments ready to be graded show in a list.	Pass	
User views assignment submission by clicking the grade icon.	A form is shown with their student assignment files attached.	A form is shown with their student assignment files attached.	Pass	

User enters grading and comments.	A success message would be shown that marks and comments have been sent.	A success message would be shown that the marks and comments have been sent	Pass	
User marks a different student with the same assignment task.	The list should show all students, indicating whether their mark has been graded	The list shows the new student has been graded even though no grading has been done.	Fail	In the function <code>getGrade()</code> - added <code>studentId</code> and In the <code>list-grading-collection</code> , added <code>\$grading['studentId']</code> for this issue to be fixed.
<b>Lecturer Page: Unit Management</b>				
User navigated to <b>My Units</b> .	The unit list is shown and its details.	The unit list is shown and its details.	Pass	
<b>Lecturer Page: Attendance Management</b>				
User views attendance.	The attendance shows the correct date and list with class, unit name, time and icon to mark.	The attendance shows the correct date and list with class, unit name, time and icon to mark.	Pass	
User clicks on the icon to mark attendance.	It shows a form with students of that class and status( <b>present, absent or late</b> )	It shows a form with students of that class and status( <b>present, absent or late</b> )	Pass	
User marks a student listed with a different status.	The success message is shown, and the student timetable calendar shows the unit name and status.	The success message is shown, and the student timetable calendar shows the unit name and status.	Pass	
<b>Lecturer Page: Timetable</b>				
User navigates through the timetable.	The screen will display the time, unit and room based on the correct timetable fields.	The screen displays the time, unit and room based on the correct timetable fields.	Pass	

## TEST CASES

<b>Student Page: Assignment Management</b>				
User views the assignment list created by a lecturer.	The list should show the list of assignments with details and a button to upload.	The list shows the list of assignments with details and a button to upload	Pass	
User clicks on one of the assignments by clicking on the upload icon.	A form should be shown with the details, assignment brief and a section where the student can upload files.	A form was shown with the details, assignment brief and a section where the student can upload files.	Pass	
User clicks and opens the assignment brief attached by the lecturer.	The assignment brief should be opened in a new tab.	The assignment brief was open on the same tab.	Fail	I needed to add <code>target="_blank"</code> on the <code>&lt;a&gt;</code> for it to open in a new tab.
<b>Student Page: Submission Management</b>				
User submits one assignment file and clicks on upload.	The submitted file should be attached.	The submitted file was attached.	Pass	

User clicks on submit assignment with no file attached.	Shows an error that you need to upload a file before submitting.	Shows an error that you need to upload a file before submitting.	Pass	
User submits two assignment files.	The two submitted files should be shown.	The user was only allowed to submit one file.	Fail	I needed to change the name from <b>name="assignmentFile"</b> to <b>name="assignmentFile[]"</b> to allow the user to submit more than one file.
User deletes a file from the file list.	The assignment would be deleted, and a success message would be shown	The assignment would be deleted, and a success message would be shown.	Pass	
Student Page: Units Management				
User navigated to <b>My Units</b> .	The unit list is shown and its details.	The unit list is shown and its details.	Pass	
Student Page: Records Management				
User views the <b>Grades</b> records in the graded assignment file.	The graded assignments are shown with overall marks and lecturer comments.	The graded assignments are shown with overall marks and lecturer comments	Pass	
User views <b>Attendance</b> .	The attendance displays a calendar with colours indicating the student's status: green for present, red for absent, and orange for late.	The attendance displays a calendar with colours indicating the student's status: green for present, red for absent, and orange for late.	Pass	
Student Page: Timetable				
User navigates through the <b>Timetable</b> .	The screen will display the time, unit and room based on the correct timetable fields.	The screen displays the time, unit and room based on the correct timetable fields.	Pass	

TEST CASES				
Test Case Type	Expected Output	Actual Output	Pass/Fail	Comments
Calendar Management				
User clicks on the schedule new event in the calendar.	A form is shown to schedule a new event	A form is shown to schedule a new event	Pass	
User fills in the form and selects <b>“Public Holiday Event.”</b>	The calendar should show the description and display with a light green background colour, and a success message should be shown.	The calendar shows the description and is displayed with a light green background colour, and a success message is shown.	Pass	

User fills in the form and selects <b>“Semester Break Event.”</b>	The calendar should show the description and display with a light blue background colour, and a success message should be shown.	The calendar shows the description and is displayed with a light blue background colour, and a success message is shown.	Pass	
User fills in the form and selects <b>“School Holiday Event.”</b>	The calendar should show the description and display with a yellow background colour, and a success message should be shown.	The calendar shows the description and is displayed with a yellow background colour, and a success message is shown.	Pass	
User goes to the lecturer's login and checks the calendar.	The calendar should show an overview of all events in light red colour.	The calendar shows the overview of all events in light red colour.	Pass	
User goes to the student login and checks the calendar.	The calendar should show an overview of all events and assignment due dates in a light red colour.	The calendar shows all events and assignment due dates in light red colour.	Pass	

Message Management				
User clicks on the buttons inbox, outbox, favourites and archives.	The correct component should be shown.	The correct component is shown.	Pass	
User creates a message and sends it to themselves	The message would be shown in their outbox	The message is shown on their outbox.	Pass	
User creates a message and sends it to a recipient that exist.	The message would be sent to the user and show a success message.	The message is sent to the user and shows a success message.	Pass	
User that receives the message goes into their inbox.	The message is shown in their inbox.	The message is shown in their inbox.	Pass	
User sends a message to a recipient that doesn't exist.	Show an error that the recipient doesn't exist.	The message is sent even though the user doesn't exist.	Fail	Check recipient existence before sending and block the message if the user does not exist.
User sends a message to a recipient with a file attached.	The file should be attached to the message	The file wasn't attached to the message	Fail	Needed to fix the message-inc for the file to be an attachment.