Ethan Moore's **Project Report** **10/12/2025 16:07pm**

Requirement by brief:

You are required to submit a two-page project report (in PDF format) that includes the following:

- *Basic information: Your name, student ID, hyperlink to your system's entry page, and a test user's credentials (username and password).*
- *Source code link: A GitHub (or other platform) link to your source code (if used).*
- *Functionality: A list of the required functions implemented and the corresponding source files.*
- *Security features: A list of the implemented security features with corresponding source files.*
- *Additional notes: Any other information relevant to running or evaluating your website.*

Avoid including screenshots in the report, and ensure the report is clear and concise to avoid penalties for missing or unclear information.

- *Basic information: Your name, student ID, hyperlink to your system's entry page, and a test user's credentials (username and password).*

Name: Ethan John Moore

Student ID: 250287637

Hyperlink to my projects entry page: *Assuming that you have done the following steps:*

1. *Downloaded a zipped copy of my php files & folders*
2. *Extracted them, and placed them in the following directory: C:\xampp\htdocs\*
3. *Confirm that the path C:\xampp\htdocs\projectmanager contains all of the php files and folder as per my submission*
4. *Created & Imported the SQL database with the name* ***'projectdb'.***

The link will work by entering this in your **web browser:** http://localhost/projectmanager/

A test user's credentials (username and password): Assuming that you have done the following step:

1. Inserted the database with name **'projectdb'** or ran the query to populate ALL tables

An account you will be able to long into in plaintext values will be:

Username	Password
jakehenry	Password123

Initially to set the user up, I accessed my **'generate_hash.php'** file which is located in **'PhpProject\projectmanager\tools\ generate_hash.php'** & I entered my plaintext password in the PHP file, I then opened the hyperlink: http://localhost/projectmanager/tools/generate_hash.php. I then copied the hashed value, and inserted the user into the database manually to test manual insertion, but it can be done by registering as well.

- *Source code link: A GitHub (or other platform) link to your source code (if used).*

Uploaded a copy of my files to: https://github.com/itzcha11/projectmanager

- *Functionality: A list of the required functions implemented and the corresponding source files.*

Area	Assignment Requirement	What's been done
Database	You are provided with an SQL script, `aproject.sql`, to create two tables (`users` and `projects`) in a MySQL database:	I used MySQL and created users and projects tables using the provided SQL script. Foreign key uid links correctly. I inserted multiple test users and projects for demonstration. I used PHP with HTML, CSS & JavaScript on the client-side. Database name: **projectdb.sql**
	You can extend the database with additional tables and fields if necessary. Populate the tables with test data either manually or automatically for demonstration purposes.	I have manually populated some data in the database for users/projects.
	Your server-side implementation should incorporate any technologies	2 examples below:

	covered in the module, such as PHP (Vanilla or Object-Oriented), Laravel framework, or Node.JS, in combination with HTML and other front-end technologies.	**Register.php ->** Done using PHP Vanilla, SQL + PDO, HTML, CSS Styles, JavaScript for validation, Sessions for tracking **Styles.css ->** css
Functional requirements for public users	• View a list of all projects (display title, starting date, and short description).	The below files contain the code logic for viewing the projects. **project_view.php** **project_edit.php** **project_add.php** **index.php**
	• View project details (including end date, phase, and user email).	This is visible in the following file: **project_view.php**
	• Search projects by title or start date.	Logic for this is in the following file: **index.php**
	• Register as a new user.	It is in: **register.php**
Functional requirements for registered users	• Log in to the system.	There is a log in form with verification against user in database, you can log in if your user is created there. **login.php**
	• Add a new project.	You can only add a new project if you are signed in **project_add.php**
	• Update their own projects.	Only users that created the project can amend it. **project_edit.php**
	• Log out of the system.	Users can log out **logout.php**
Security measures	• Authentication (user login)	**`login.php`** (password_verify)
	• Authorisation (ensuring users can only modify their own projects)	**`project_edit.php, project_add.php`** (ownership checks lines 30-36, AND in queries line 77)
	• Form validation (validating input on both client and server-side)	**`register.php`**, **`project_add.php`**, **`project_edit.php.`** Client-side JS in all forms + Server-side in all POST handlers
	• SQL/HTML injection protection	All DB queries . Prepared SQL statements in code + sanitizing output. **`index.php`**, **`project_view.php`**, `project_edit.php`, etc **htmlspecialchars()** on all output
	• Password hashing	`password_hash()` in **register.php**, `password_verify()` in **login.php**
	• Cross-Site Request Forgery (CSRF) prevention	**header.php** - Token generation (lines 4-6) **login.php** - Token in form + validation (lines 10-12, line 42) **register.php** - Token in form + validation (lines 13-15, line 77) **project_add.php** - Token in form + validation (lines 14-16, line 58) **project_edit.php** - Token in BOTH forms + validation for both update AND delete (lines 44-46, 64-66, lines 106, 115) **logout.php** - Token in URL + validation (lines 4-6) **logout link in header.php** - Token passed in URL (line 31)
	• Proper use of comments.	Comments in key logic + files separated by function

Good coding practice		All files have this.. Example: **register.php**
	• Consistent and logical naming conventions.	Consistent file + variable naming Example: **register.php**
	• Efficient and modular code.	Functions used for DB queries and validation Example: **register.php**
User interface	• All web pages should be easy to navigate, with appropriate text size, font, and colour.	Header menu visible on all pages, easy to navigate. All files.
	• Use descriptive names for links and buttons.	Very logically developed. All files.
	• Ensure a consistent layout across pages.	Same header on each page. All files.
	• You are free to use any front-end technologies (HTML, CSS, JavaScript, and libraries) to enhance the UI.	Php, SQL, HTML, CSS, JavaScript, Sessions have been used in file: **register.php**
Project report	• Basic information: Your name, student ID, hyperlink to your system's entry page, and a test user's credentials (username and password).	This document.
	• Source code link: A GitHub (or other platform) link to your source code (if used).	https://github.com/itzcha11/projectmanager
	• Functionality: A list of the required functions implemented and the corresponding source files.	This document.
	• Security features: A list of the implemented security features with corresponding source files.	
	• Additional notes: Any other information relevant to running or evaluating your website.	

- *Security features: A list of the implemented security features with corresponding source files.*

Security measures	• Authentication (user login)	**login.php** (password_verify)
	• Authorisation (ensuring users can only modify their own projects)	**project_edit.php, project_add.php** (ownership checks lines 30-36, AND in queries line 77)
	• Form validation (validating input on both client and server-side)	**register.php, project_add.php, project_edit.php.** Client-side JS in all forms + Server-side in all POST handlers
	• SQL/HTML injection protection	All DB queries . Prepared SQL statements in code + sanitizing output. **index.php, project_view.php,** project_edit.php, etc **htmlspecialchars()** on all output
	• Password hashing	password_hash() in **register.php,** password_verify() in **login.php**
	• Cross-Site Request Forgery (CSRF) prevention	**header.php** - Token generation **login.php** - Token in form + validation **register.php** - Token in form + validation **project_add.php** - Token in form + validation **project_edit.php** - Token in BOTH forms + validation for both update AND delete **logout.php** - Token in URL + validation **logout link in header.php** - Token passed in URL

- *Additional notes: Any other information relevant to running or evaluating your website.*

The following steps are how I get it working from scratch.

1. go to -> C:\xampp\htdocs\phpProject\database\

2. Go to XAMPP control panel, start Apache & MySQL.

3. Open your Web Browser, go to 'http://localhost/phpmyadmin'.

4. Click 'New' in the top left, enter name for db as **'projectdb'** & click create

5. Go to 'Import' tab.

6. Click 'Browse' and select the following SQL db file, 'phpProject\database\projectdb.sql' and click 'import' at the bottom of the screen.

7. Confirm the queries ran successfully, and the 'Projects' and 'Users' tables have been populated with some data.

8. Navigate to 'http://localhost/projectmanager/'.

Then you will then have access to the local website, which is linked with the database!

For information purposes, below is a short explanation of what the files do in the following location.

Database:

PhpProject\database\ -> contains **projectdb.sql,** which is the database you need to import, used to populate the user and project tables.

Recording:

PhpProject\recording\ -> contains a recording in which I talk through the web functionality.

Project Files:

PhpProject\projectmanager\ -> Contains the main project files and subfolders..

Subfolders:

PhpProject\projectmanager\css\ -> this contains styles.css which is used to control the page layout and 'beautify' it.

PhpProject\projectmanager\includes\ -> this contains footer.php & header.php, which are the headers and footers.

PhpProject\projectmanager\sql\ -> this contains the sql file 'aproject.sql' and a readme.txt, which explains the purpose, this file was used to create the users & projects table initially, provided by the tutor.

PhpProject\projectmanager\tools\ -> this folder contains the following files, generate_hash.php, reset_password.php & readme.txt (just explains what they do), but are simple tools I created whilst developping the website initially, as I ran into some problems, however, 1 generates a hash value of a plaintext hardcoded password, and the other resets the password.

PHP Files (in projectmanager root) PhpProject\projectmanager\:

Config.php -> contains db connection settings / connecting using PDO

Index.php -> handles search request for titles/dates, acts as a ‘homepage’ contains project titles, start dates, short description & view button.

Login.php -> allows users to log into the website using credentials from user in database and prompts error if applicable.

Logout.php -> logic to handle users loging out.

Project_add.php -> code to allow users that are only logged in to be able to add a project.

Project_edit.php -> code to allow owners of projects to edit their own projects, and logic for it.

Project_view.php -> contains code for users to be able to view more project information such as end date/phase.

Register.php -> code to allow users to register and save to db, allowing them to log in.