

Submission Worksheet

[CLICK TO GRADE](#)

<https://learn.ethereallab.app/assignment/IT202-452-M2024/it202-module-6-milestone-1-2024/grade/sha38>

IT202-452-M2024 - [IT202] Module 6 Milestone 1 2024

Submissions:

Submission Selection

1 Submission [active] 7/9/2024 12:44:37 AM

Instructions

^ COLLAPSE ^

Overview Video: <https://youtu.be/V7oHa8KKtss>

Prereqs:

- Go through each lesson from "Project Setup and SQL" to "User Login Enhancement" and follow the branching names while gathering the code
- Merge each into Milestone1 branch
- Create a Project board on GitHub (if you haven't yet)
 - Add each major item from the proposal doc as an Issue item
 - Invite the grader(s) and myself as collaborators on the board (they're separate from your repository)
 - ☐ See Canvas announcements for the Usernames or check your collab list on your repo
- Mark the related GitHub Issues items as "done"
- Implement your own custom CSS (something much different than the default "ugly" CSS given as an example)
- Consider styling all forms/inputs, data output, navigation, etc
- Implement JavaScript validation on Register, Login, and Profile (include "[Client]" in the output messages to differentiate between server-side validations)

Instructions:

1. Make sure you're in Milestone1 with the latest changes pulled
2. Ensure Milestone1 has been deployed to heroku dev
3. Gather the requested evidence and fill in the explanations per each prompt
4. Save the submission and generate the output PDF
5. Put the output PDF into your local repository folder

6. add/commit/push it to GitHub
7. Merge Milestone1 into dev
8. Locally checkout dev and pull the changes
9. Create and merge a pull request from dev to prod to deploy Milestone1 to prod
10. Upload this output PDF to Canvas

Branch name: Milestone1

Tasks: 25 **Points:** 10.00



User Registration (2 pts.)

^COLLAPSE ^



EXPAND

Task #1 - Points: 1

Text: Screenshot of form on website page



EXPAND

Task #2 - Points: 1

Text: Screenshot of the form code



EXPAND

Task #3 - Points: 1

Text: Screenshot of the client-side and server-side validation code



EXPAND

Task #4 - Points: 1

Text: Screenshot of the Users table with a valid user entry



EXPAND

Task #5 - Points: 1

Text: Explain the registration logic in a step-by-step manner from when the page loads to when the data is saved to the DB



EXPAND

Task #6 - Points: 1

Text: Include pull request links related to this feature



User Login (2 pts.)

^COLLAPSE ^



Task #1 - Points: 1

Text: Screenshot of form on website page



Task #2 - Points: 1

Text: Screenshot of the form code



Task #3 - Points: 1

Text: Explain the login logic step-by-step from when the page loads to when the data is fetched from the DB and stored in the session



Task #4 - Points: 1

Text: Include pull request links related to this feature



User Logout (1 pt.)



Task #1 - Points: 1

Text: Capture the following screenshots



Task #2 - Points: 1

Text: Include pull request links related to this feature



Basic Security Rules and Roles (2 pts.)



Task #1 - Points: 1

Text: Authentication Screenshots



Task #2 - Points: 1

Text: Authorization Screenshots



Task #3 - Points: 1

Text: Screenshots of UserRoles and Roles Tables

EXPAND

Task #4 - Points: 1

Text: Explain how Roles and UserRoles tables work in conjunction with the Users table

EXPAND

Task #5 - Points: 1

Text: Include pull request links related to this feature

COLLAPSE

User Profile (2 pts.)

COLLAPSE

Task #1 - Points: 1

Text: View Profile Website Page

Details:

Thoughtful CSS should be applied to all parts (must differ from the "ugly" CSS given via the lessons).

The Heroku dev URL must be present in all screenshots of the site.

#1) Show the profile form correctly populated on page load (username, email) Form should have the following fields: username, email, current password, new password, confirm password (or similar)



Caption (required) ✓

Describe/highlight what's being shown

Showing the profile form correctly populated on page load

COLLAPSE

Task #2 - Points: 1

Text: Explain the logic step-by-step of how the data is loaded and populated when the profile page is visited

i Details:

Don't just show code, translate things to plain English in concise steps.

May be worthwhile using a list.

Response:

Firstly we have a form where the user can input new updated information. We include the navigation and authentication check. `Partials/nav.php`, is included to provide navigation elements. We then check if user is logged in using `is_logged_in()`, if not it'll redirect user to the login page. We then handle form submission, when a form is submitted, data is retrieved from the form fields, email and username, which then prepares an SQL statement to update the users table with new email and username. If there's a duplicate entry, then it'll inform the user about the issue. We fetch the updated user data to update the session variables. Now for password, we check if all the forms are filled and validate the current password against the database. We then update the password using `password_hash` if validation is successful.

^COLLAPSE ^

Task #3 - Points: 1

Text: Edit Profile Website Page

i Details:

Thoughtful CSS should be applied to all parts (must differ from the "ugly" CSS given via the lessons).

The Heroku dev URL must be present in all screenshots of the site.

#1) (Two Screenshots) Demonstrate with before and after of a username change (including success message)



Screenshot of the 'Edit Profile' form before a username change. The form includes fields for Email, Username, Password, Current Password, New Password, and Confirm Password, along with an 'Update Profile' button. The URL in the browser is `sha3bde-2477c26b7a2.herokuapp.com/Profile.php`.

Screenshot of the 'Profile' page after a successful username change. The profile information is displayed, including the new username. The URL in the browser is `sha3bde-2477c26b7a2.herokuapp.com/Profile.php`.

Caption (required) ✓

Describe/highlight what's being shown

#2) Demonstrate the success message of updating password



The screenshot shows a web browser window with the URL `sha38-dev-2a73c56eb3a2.herokuapp.com/Project/profile.php`. The page has a navigation bar with links: Home, Profile, Create Role, List Roles, Assign Roles, and Logout. Below the navigation bar, there is a green banner that says "Profile saved". Below that, there is a section titled "Password reset" with the following form fields: Email (sha38@ghs.edu), Username (sha38), Password Reset (Current Password), New Password, and Confirm Password. There is an "Update Profile" button at the bottom of the form.

Caption (required) ✓

Describe/highlight what's being shown

Demonstrates the success message of updating password

#3) Demonstrate all JavaScript user-friendly validation messages [can be combined] (email format, username format, password format, new password matching confirm password)



The screenshot shows the same web browser window as in the previous image. The page displays several red error messages at the top: "[Client] Username is already taken" (twice), "[Client] New password must be at least 8 characters long", and "[Client] New passwords do not match". Below these, there is a yellow banner that says "[Client] Password and confirm password must match". The form fields and the "Update Profile" button are visible at the bottom.

Caption (required) ✓

Describe/highlight what's being shown

Demonstrates all JavaScript user-friendly validation messages

#4) Demonstrate PHP user-friendly validation message (desired email is already in use)



A screenshot of a web browser displaying a profile update page. The browser's address bar shows the URL: sha38-dev-2a72c56eb3a2.herokuapp.com/Project/profile.php. The page has a dark header with navigation links: Home, Profile, Create Role, List Roles, Assign Roles, and Logout. A yellow banner at the top contains the message: "The chosen email is not available." Below this, the form includes input fields for Email (sha38@npt.edu), Username (sha40), Password Reset, Current Password, New Password, and Confirm Password. An "Update Profile" button is at the bottom of the form.

Caption (required) ✓

Describe/highlight what's being shown

Demonstrates PHP user-friendly validation message

#5) Demonstrate PHP user-friendly validation message (Desired username is already in use)



A screenshot of a web browser displaying a profile update page. The browser's address bar shows the URL: sha38-dev-2a72c56eb3a2.herokuapp.com/Project/profile.php. The page has a dark header with navigation links: Home, Profile, Create Role, List Roles, Assign Roles, and Logout. A yellow banner at the top contains the message: "The chosen username is not available." Below this, the form includes input fields for Email (sha38@npt.edu), Username (sha40), Password Reset, Current Password, New Password, and Confirm Password. An "Update Profile" button is at the bottom of the form.

Caption (required) ✓

Describe/highlight what's being shown

Demonstrates PHP user-friendly validation message

#6) Demonstrate PHP user-friendly validation message (Current password doesn't match what's in the DB)



A screenshot of a web browser displaying a profile update page. The browser's address bar shows the URL: sha38-dev-2a72c56eb3a2.herokuapp.com/Project/profile.php. The page has a dark header with navigation links: Home, Profile, Create Role, List Roles, Assign Roles, and Logout. A green banner at the top contains the message: "Profile saved". Below this, a yellow banner contains the message: "Current password is invalid". The form includes input fields for Email (sha38@npt.edu), Username (sha40), Password Reset, Current Password, New Password, and Confirm Password. An "Update Profile" button is at the bottom of the form.

Caption (required) ✓*Describe/highlight what's being shown*

Demonstrates PHP user-friendly validation message

^COLLAPSE ^

Task #4 - Points: 1

Text: Explain the logic step-by-step of how the data is checked and saved for the following scenarios

Details:

Don't just show code, translate things to plain English

#1) Updating Username/Email**Explanation (required)***Explain in concise steps how this logically works*

PREVIEW RESPONSE


First, it checks to make sure only authenticated users can access the profile page, then checks to see if a form has been submitted. If a form has been submitted, it retrieves the submitted email and username using `se` function. We then prepare SQL parameters and a PDO statement for updating the users table in the database. We then execute the update query whose primary purpose is to update the database, if successful, it'll flash a successful message indicating that the profile update was


#2) Updating password**Explanation (required)***Explain in concise steps how this logically works*

PREVIEW RESPONSE

First, it checks to make sure only authenticated users can access the profile page, then checks to see if a form has been submitted. If a form has been submitted, it retrieves the submitted email and username using `se` function. We then validate the current password against the hashed password stored in the database. If valid, it updates the current password to the new hashed password, if invalid, the appropriate flash message will be displayed.

the profile update was saved. If error occurs, it informs the user that chosen email or username is not available. After updating database successfully, it retrieves the updated user data from users table and updates the \$_SESSION array with new values to reflect changes made.

 **Task #5 - Points: 1**
Text: Include pull request links related to this feature

 **Details:**
Should end in /pull/#

URL #1
<https://github.com/ShahadatAnadil/sha38-it202-452/pull/25>

 **Misc (1 pt.)**


 **Task #1 - Points: 1**
Text: Screenshot of wakatime

 **Task #2 - Points: 1**
Text: Screenshot of your project board from GitHub (tasks should be in the proper column)

 **Task #3 - Points: 1**
Text: Provide a direct link to the project board on GitHub

End of Assignment