

Submission Worksheet

CLICK TO GRADE

<https://learn.ethereallab.app/assignment/IT202-008-S2024/it202-milestone-1-2024/grade/dsp82>

IT202-008-S2024 - [IT202] Milestone 1 2024

Submissions:

Submission Selection

1 Submission [active] 4/2/2024 2:49:53 PM

Instructions

[^ COLLAPSE ^](#)

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

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, Logout, and Profile (include "[Client]" in the output messages to differentiate between server-side validations)

Instructions:

Make sure you're in Milestone1 with the latest changes pulled

Ensure Milestone1 has been deployed to heroku dev

Gather the requested evidence and fill in the explanations per each prompt

Save the submission and generate the output PDF

Put the output PDF into your local repository folder

add/commit/push it to GitHub

Merge Milestone1 into dev

Locally checkout dev and pull the changes

Create and merge a pull request from dev to prod to deploy Milestone1 to prod

Upload this output PDF to Canvas

Branch name: Milestone1

Tasks: 26 **Points:** 10.00



User Registration (2 pts.)

[^ COLLAPSE ^](#)

● COLLAPSE

Task #1 - Points: 1

Text: Screenshot of form on website page

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Heroku dev url should be present in the address bar
<input type="checkbox"/> #2	1	Should have thoughtful CSS applied
<input type="checkbox"/> #3	1	Demonstrate JavaScript validation for each field [can be combined] (email validation (format), username validation (format), password validation (format), password and confirm password matching, and each field being required)
<input type="checkbox"/> #4	1	Demonstrate email already in use message (message text doesn't need to be exact, but should be clear)
<input type="checkbox"/> #5	1	Demonstrate username already in use message (message text doesn't need to be exact, but should be clear)
<input type="checkbox"/> #6	1	Demonstrate user-friendly message of new account being created

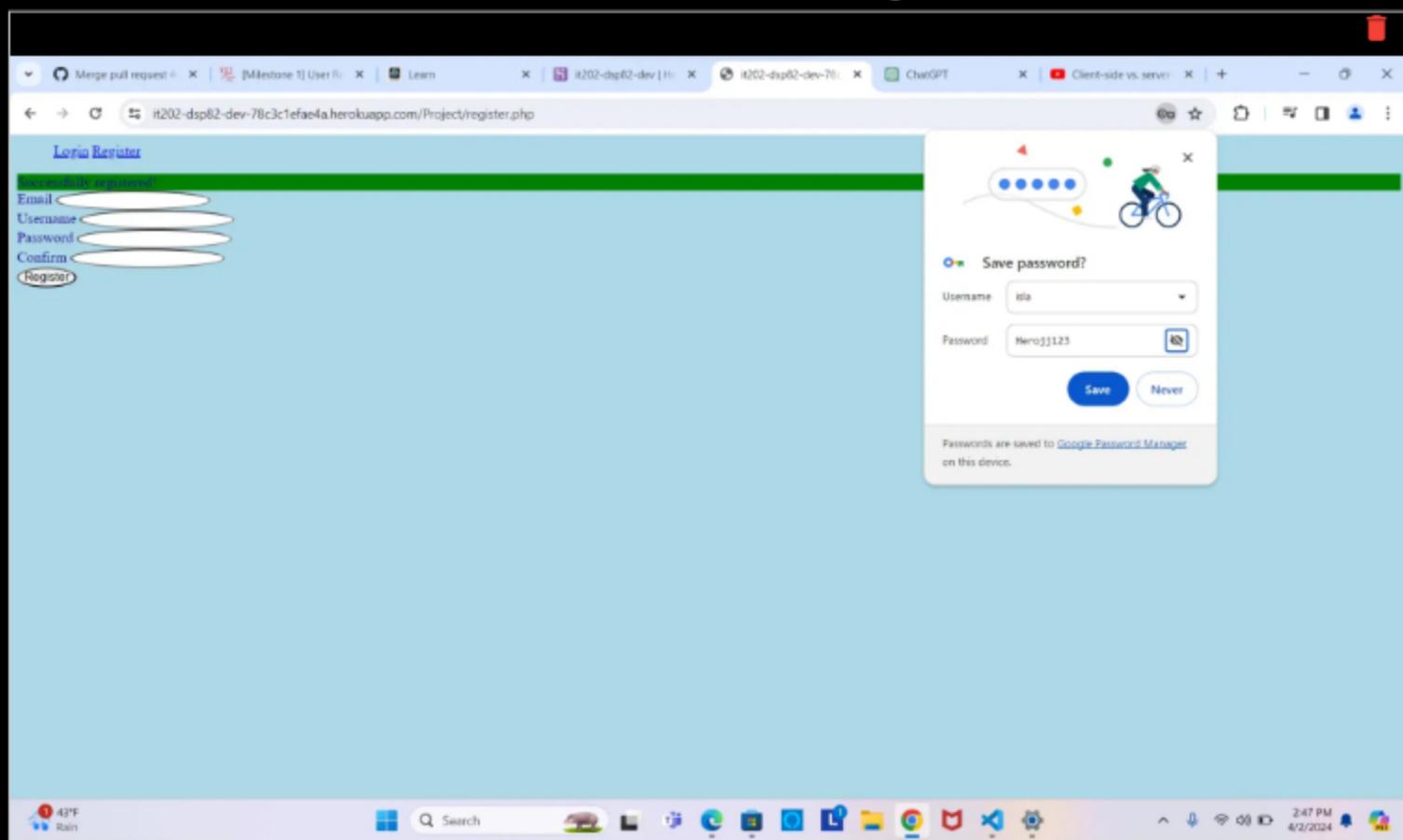
Task Screenshots:

Gallery Style: Large View

Small

Medium

Large



User registration

Checklist Items (4)

#1 Heroku dev url should be present in the address bar

#2 Should have thoughtful CSS applied

#3 Demonstrate JavaScript validation for each field [can be combined] (email validation (format), username validation (format), password validation (format), password and confirm password matching, and each field being required)

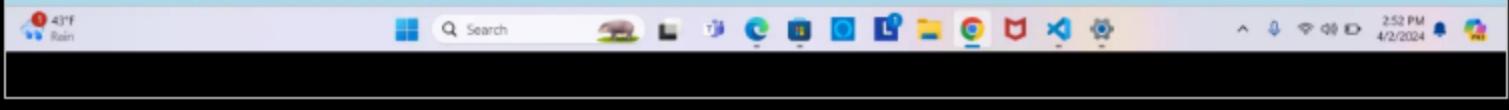
#6 Demonstrate user-friendly message of new account being created

The screenshot shows a Microsoft Edge browser window with multiple tabs open. The active tab is 'it202-dsp82-dev-78c3c1efae4a.herokuapp.com/Project/register.php'. The page displays a 'Login Register' form. An error message 'The chosen email is not available.' is highlighted in yellow above the email input field, which contains 'd@trussey_90@yahoo.com'. Below the form, a message 'Email in use' is displayed in a black box. The browser's taskbar at the bottom shows various pinned icons and the date/time '4/3/2024 2:51 PM'.

Checklist Items (1)

#4 Demonstrate email already in use message (message text doesn't need to be exact, but should be clear)

The screenshot shows a Microsoft Edge browser window with multiple tabs open. The active tab is 'it202-dsp82-dev-78c3c1efae4a.herokuapp.com/Project/register.php'. The page displays a 'Login Register' form. An error message 'The chosen username is not available.' is highlighted in yellow above the username input field, which contains 'd@trussey_90'. Below the form, a message 'Email in use' is displayed in a black box. The browser's taskbar at the bottom shows various pinned icons and the date/time '4/3/2024 2:51 PM'.



Username in use

Checklist Items (1)

#5 Demonstrate username already in use message (message text doesn't need to be exact, but should be clear)

Task #2 - Points: 1

Text: Screenshot of the form code

Details:

Should have appropriate input types for the field

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

```
</form>
<script>
//dsp82 4/2/2024
function validate(form) {
    var email = form.email.value;
    var username = form.username.value;
    var password = form.password.value;
    var confirm = form.confirm.value;

    var emailRegex = /^[^@\s]+@[^\s]+\.[^\s]+$/;
    var usernameRegex = /^[a-zA-Z0-9_-]{3,16}$/;
    var passwordRegex = /^(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}$/;

    if (!emailRegex.test(email)) {
        alert("Please enter a valid email address.");
        return false;
    }

    if (!usernameRegex.test(username)) {
        alert("Username must only contain 3-16 characters a-z, 0-9, _, or -");
        return false;
    }

    if (!passwordRegex.test(password)) {
        alert("Password must be at least 8 characters long and contain at least one digit, one lowercase letter, and one uppercase letter.");
        return false;
    }

    if (password !== confirm) {
        alert("Passwords do not match.");
        return false;
    }
}
```

```
    return true;
}
</script>
```

Form validation code in register.php

Task #3 - Points: 1

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

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Show the JavaScript validations (include any extra files related)
<input checked="" type="checkbox"/> #2	1	Show the PHP validations (include any lib content)
<input checked="" type="checkbox"/> #3	1	Include ucid/date code comments in all screenshots (one per screenshot is sufficient)

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

```
</form>
<script>
//dsp82 4/2/2024
function validate(form) {
    var email = form.email.value;
    var username = form.username.value;
    var password = form.password.value;
    var confirm = form.confirm.value;

    var emailRegex = /^[^@\s]+@[^\s]+\.\w+$/;
    var usernameRegex = /^[a-zA-Z0-9_-]{3,16}$/;
    var passwordRegex = /^(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}$/;

    if (!emailRegex.test(email)) {
        alert("Please enter a valid email address.");
        return false;
    }

    if (!usernameRegex.test(username)) {
        alert("Username must only contain 3-16 characters a-z, 0-9, _, or -");
        return false;
    }

    if (!passwordRegex.test(password)) {
        alert("Password must be at least 8 characters long and contain at least one digit, one lowercase letter, and one uppercase letter.");
        return false;
    }

    if (password !== confirm) {
        alert("Passwords do not match.");
        return false;
    }

    return true;
}
</script>
```

Javascript validation

Checklist Items (0)

```
<script>
//dsp82 4/2/2024
//TDDO02: add form validation
```

```

//TODO 2: add pre code
if (isset($_POST["email"]) && isset($_POST["password"]) && isset($_POST["confirm"]) && isset($_POST["username"])) {
    $email = se($_POST, "email", "", false);
    $password = se($_POST, "password", "", false);
    $confirm = se($_POST, "confirm", "", false);
    $username = se($_POST, "username", "", false);
    //TODO 3
    $hasError = false;
    if (empty($email)) {
        flash("Email must not be empty", "danger");
        $hasError = true;
    }
    //sanitize
    $email = sanitize_email($email);
    //validate
    if (!is_valid_email($email)) {
        flash("Invalid email address", "danger");
        $hasError = true;
    }
    if (!is_valid_username($username)) {
        flash("Username must only contain 3-16 characters a-z, 0-9, _, or -", "danger");
        $hasError = true;
    }
    if (empty($password)) {
        flash("password must not be empty", "danger");
        $hasError = true;
    }
    if (empty($confirm)) {
        flash("Confirm password must not be empty", "danger");
        $hasError = true;
    }
    if (!is_valid_password($password)) {
        flash("Password too short", "danger");
        $hasError = true;
    }
}

```

php validation1

Checklist Items (0)

```

ic_html > Project > register.php
if (isset($_POST["email"]) && isset($_POST["password"]) && isset($_POST["confirm"]) && isset($_POST["username"])) {
    if (empty($password)) {
        flash("password must not be empty", "danger");
        $hasError = true;
    }
    if (empty($confirm)) {
        flash("Confirm password must not be empty", "danger");
        $hasError = true;
    }
    if (!is_valid_password($password)) {
        flash("Password too short", "danger");
        $hasError = true;
    }
    if (
        strlen($password) > 8 && $password !== $confirm
    ) {
        flash("Passwords must match", "danger");
        $hasError = true;
    }
    if (!$hasError) {
        //TODO 4
        $hash = password_hash($password, PASSWORD_BCRYPT);
        $db = getDB();
        $stmt = $db->prepare("INSERT INTO Users (email, password, username) VALUES(:email, :password, :username)");
        try {
            $stmt->execute([":email" => $email, ":password" => $hash, ":username" => $username]);
            flash("Successfully registered!", "success");
        } catch (Exception $e) {
            users_check_duplicate($e->errorInfo);
        }
    }
}
?>
<?php
require(__DIR__ . "/../../partials/flash.php");
?>

```

php validation2

Checklist Items (0)

Task #4 - Points: 1

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

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Password should be hashed
#2	1	Should have email, password, username (unique), created, modified, and id fields
#3	1	Ensure left panel or database name is present (should contain your ucid)

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

The screenshot shows a MySQL Workbench interface with the following details:

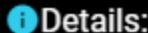
- Left Panel (DATABASE):** Shows the database structure with tables like 'db', 'dp02', '112k', 'Tables (0)', 'Roles (1)', 'UserRoles', 'Views', 'Procedures', and 'Functions'. The 'Users' table is selected.
- Central Area (Properties):** Shows the query: `SELECT * FROM 'Users' LIMIT 100`. The results pane displays 9 rows of data from the 'Users' table.
- Results Table Headers:** The columns are labeled: id (int), email (varchar(100)), password (varchar(60)), created (timestamp), modified (timestamp), and username (varchar(30)).
- Sample Data:** The data includes rows with IDs 1 through 9, corresponding email addresses like 'dhruvjay_95@yahoo.com' and 'dpli2@nj.edu', and various hashed passwords.

Users table

Checklist Items (0)

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



Details:

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

Response:

Registration process begins with user filling out form, which will go through client side validation and then server side. The server side php script sanitizes and validates data, ensuring all data meets the required criteria. Any errors prompt an error message, while successful validation leads to the data being stored in the database as a secured hash. Successful registration allows users to login with said credentials.

Task #6 - Points: 1

Text: Include pull request links related to this feature



Details:

Should end in /pull/#

URL #1

<https://github.com/dsp82njit/dsp82-IT202-008/pull/35>

URL #2

<https://github.com/dsp82njit/dsp82-IT202-008/pull/40>

URL #3

<https://github.com/dsp82njit/dsp82-IT202-008/pull/44>

User Login (2 pts.)



Task #1 - Points: 1

Text: Screenshot of form on website page

Checklist

*The checkboxes are for your own tracking

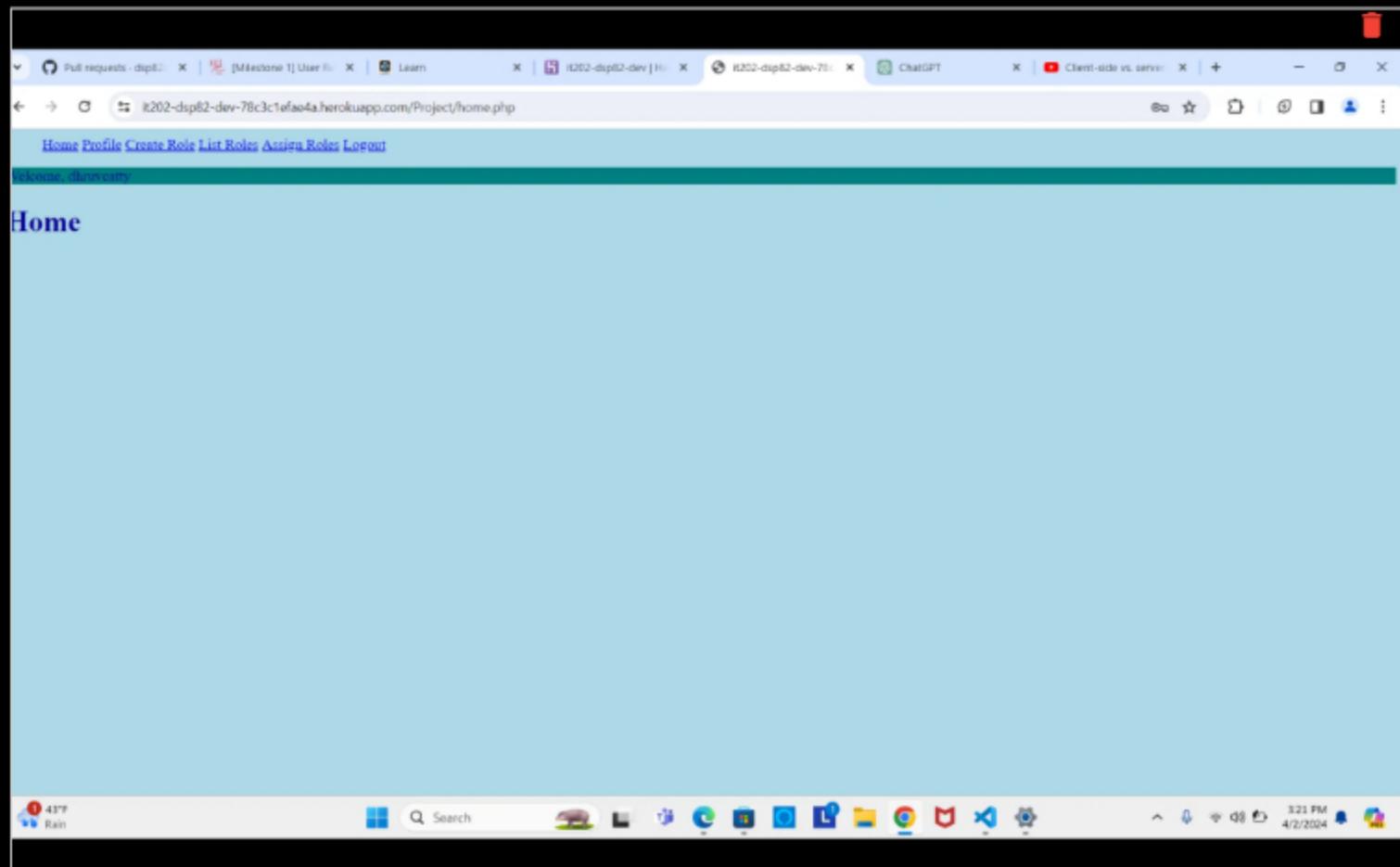
#	Points	Details
<input type="checkbox"/> #1	1	Heroku dev url should be present in the address bar
<input type="checkbox"/> #2	1	Should have thoughtful CSS applied
<input type="checkbox"/> #3	1	Include correct data where username is used to login
<input type="checkbox"/> #4	1	Include correct data where email is used to login
<input type="checkbox"/> #5	1	Show success login message
<input type="checkbox"/> #6	1	Demonstrate JavaScript validation for each field [can be combined] (username format, email format, password format, and each field being required)
<input type="checkbox"/> #7	1	Demonstrate user-friendly message of when an account doesn't exist
<input type="checkbox"/> #8	1	Demonstrate user-friendly message of when password doesn't match what's in the DB

#9	1	Demonstrate successful login message and the destination page (i.e., home or some landing/dashboard page)
#10	1	Demonstrate session data being set (captured from server logs)

Task Screenshots:

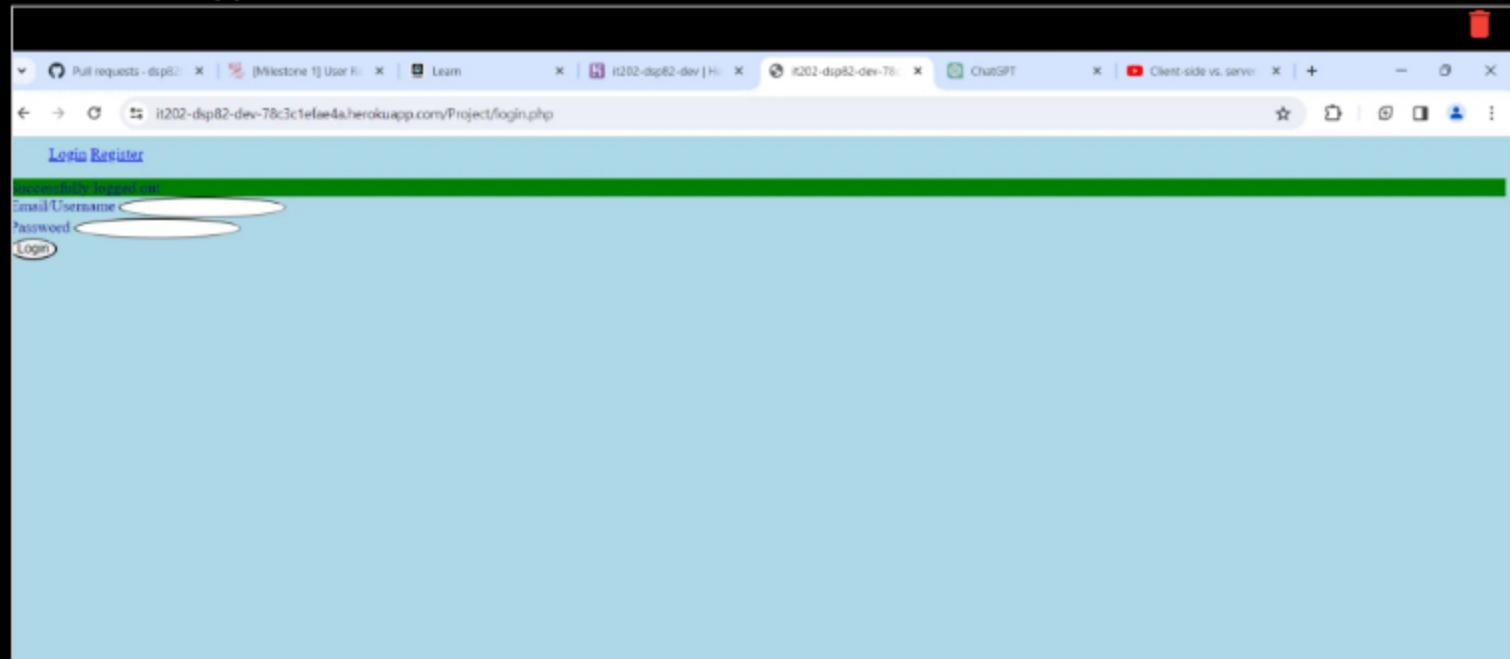
Gallery Style: Large View

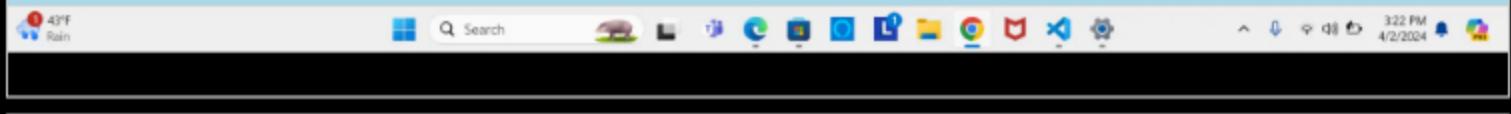
Small Medium Large



Successful login

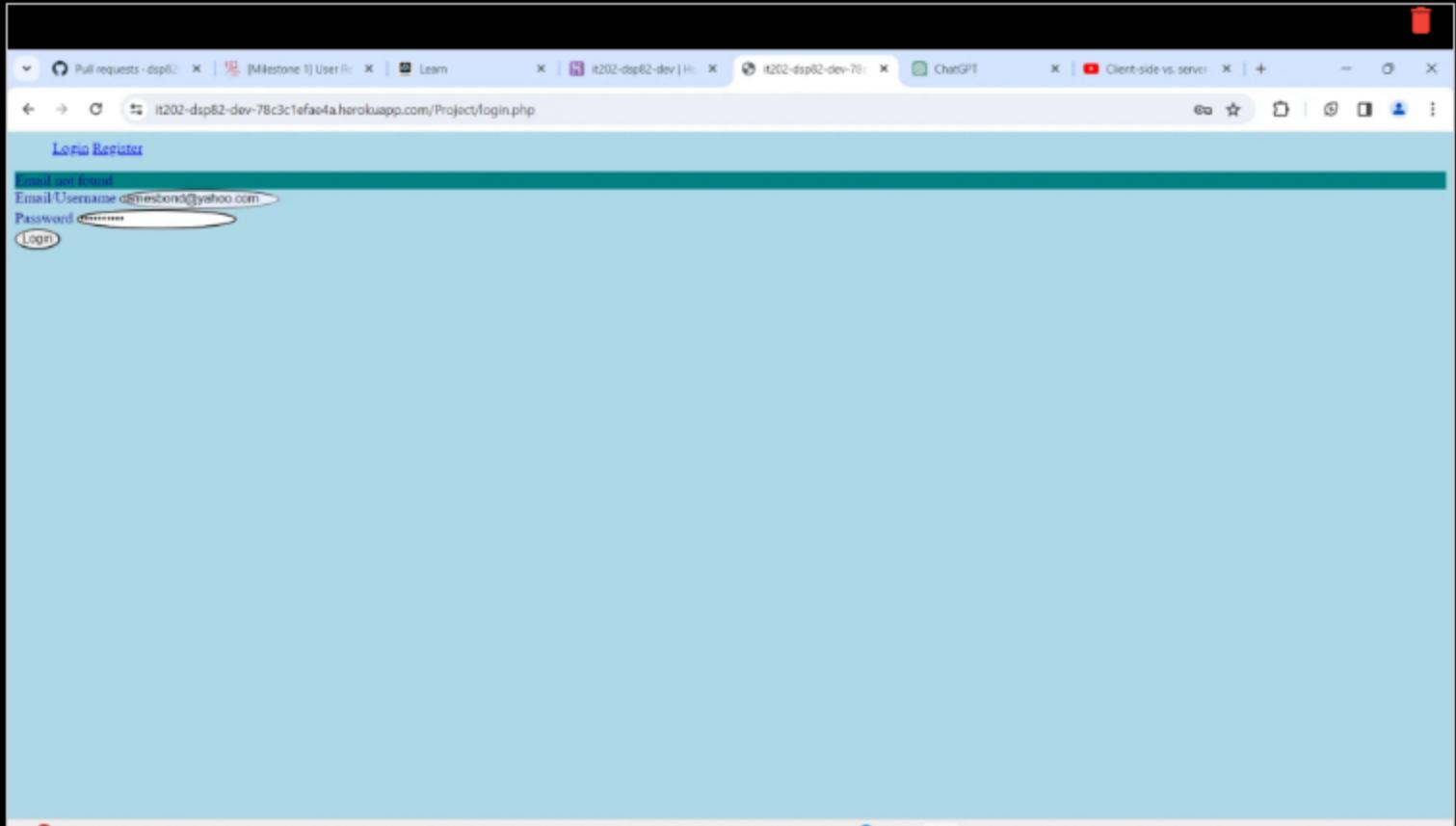
Checklist Items (0)





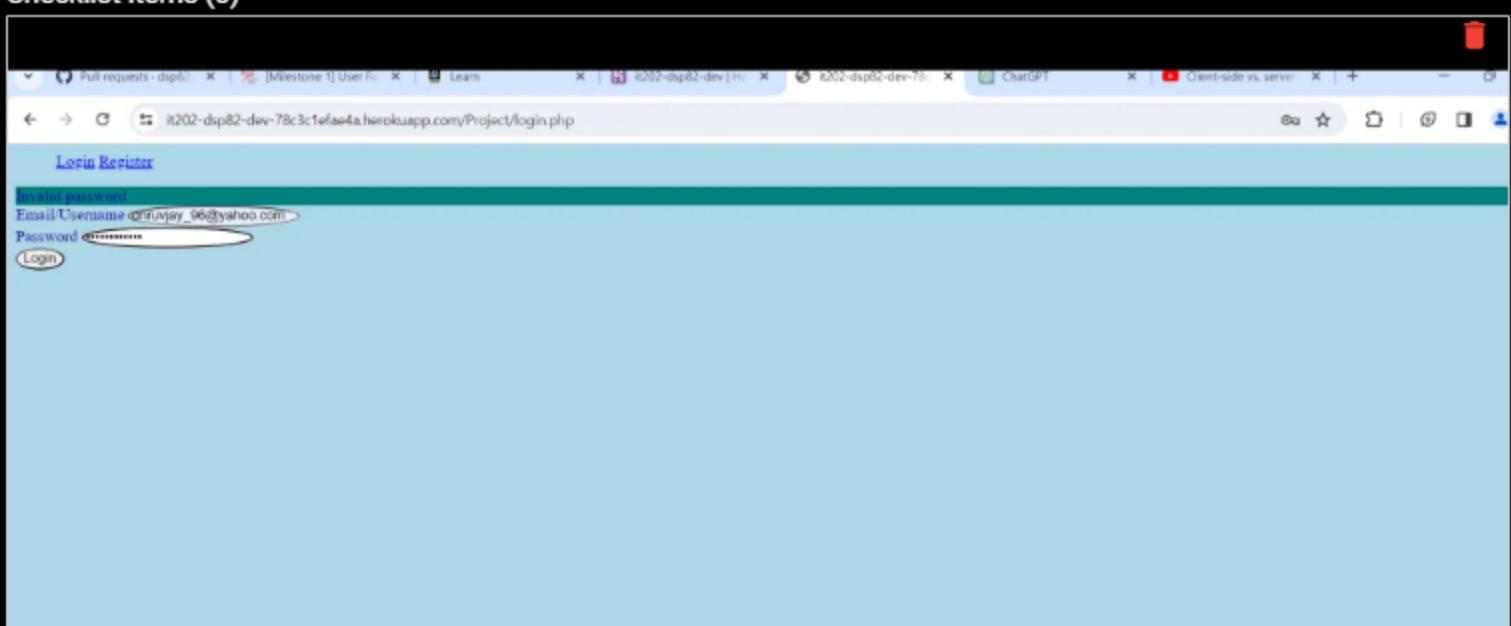
Login page

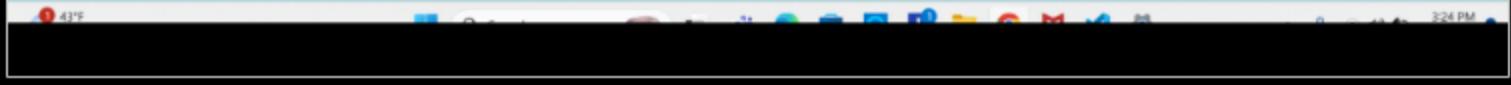
Checklist Items (0)



Email not found

Checklist Items (0)





Invalid password

Checklist Items (0)

Task #2 - Points: 1

Text: Screenshot of the form code

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Should have proper input types for the fields (note in the caption if you're using two fields for Username/Email or a combined field)
<input checked="" type="checkbox"/> #2	1	Show JavaScript validations (include any extra files related)
<input checked="" type="checkbox"/> #3	1	Show PHP validations (include any lib content)
<input checked="" type="checkbox"/> #4	1	Include ucid/date code comments in all screenshots (one per screenshot is sufficient)

Task Screenshots:

Gallery Style: Large View

Small Medium Large

```
ublic_html > Project > login.php
14  </form>
15  <script>
16      function validate(form) {
17          // Get the values of the email/username and password fields
18          var emailOrUsername = form.email.value.trim();
19          var password = form.password.value.trim();
20
21          // Check if email/username field is empty
22          if (emailOrUsername === "") {
23              alert("Email/Username must not be empty", "danger");
24              return false;
25          }
26
27          // Check if password field is empty
28          if (password === "") {
29              alert("Password must not be empty", "danger");
30              return false;
31          }
32
33          // Check if password meets the minimum length requirement
34          if (password.length < 8) {
35              alert("Password must be at least 8 characters long", "danger");
36              return false;
37          }
38
39          // All validations passed, return true to submit the form
40          return true;
41      }
42
43      // Function to display flash messages
44      function flash(message, type) {
45          // Create a new div element for the flash message
46          var flashDiv = document.createElement("div");
47
```

```
47
48     // Set the class attribute for styling based on the message type
49     flashDiv.className = "flash-message " + type;
50
```

Javcascript validation part 1

Checklist Items (0)

```
<script>
    function validate(form) {
        // Check if password meets the minimum length requirement
        if (password.length < 8) {
            alert("Password must be at least 8 characters long", "danger");
            return false;
        }

        // All validations passed, return true to submit the form
        return true;
    }

    // Function to display flash messages
    function flash(message, type) {
        // Create a new div element for the flash message
        var flashDiv = document.createElement("div");

        // Set the class attribute for styling based on the message type
        flashDiv.className = "flash-message " + type;

        // Create a text node with the message text
        var textNode = document.createTextNode(message);

        // Append the text node to the flash div
        flashDiv.appendChild(textNode);

        // Get the container element where flash messages should be displayed
        var container = document.getElementById("flash-container");

        // Append the flash div to the container
        container.appendChild(flashDiv);
    }
    //dsp82 4/2/2024
</script>
<?php
```

JAvascript validation part2

Checklist Items (0)

```
<script>
    //dsp82 4/2/2024
</script>
<?php
//TODO 2: add PHP Code
if (isset($_POST["email"]) && isset($_POST["password"])) {
    $email = se($_POST, "email", "", false);
    $password = se($_POST, "password", "", false);

//TODO 3
$hasError = false;
if (empty($email)) {
    flash("Email must not be empty");
    $hasError = true;
}
if (str_contains($email, "@")) {
    //sanitize
    //$email = filter_var($email, FILTER_SANITIZE_EMAIL);
    $email = sanitize_email($email);
    //validate
    /*if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        flash("Invalid email address");
        $hasError = true;
    }*/
    if (!is_valid_email($email)) {
        flash("Invalid email address");
        $hasError = true;
    }
} else {
    if (!is_valid_username($email)) {
        flash("Invalid username");
        $hasError = true;
    }
}
```

```
    }
}

if (empty($password)) {
    flash("password must not be empty");
    $hasError = true;
}
```

PHP validation part 1

Checklist Items (0)

```
if (isset($_POST["email"]) && isset($_POST["password"])) {
    $hasError = false;
    if (empty($email)) {
        flash("Email must not be empty");
        $hasError = true;
    }
    if (str_contains($email, "@")) {
        //sanitize
        //email = filter_var($email, FILTER_SANITIZE_EMAIL);
        $email = sanitize_email($email);
        //validate
        //if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        //    flash("Invalid email address");
        //    $hasError = true;
        //}
        if (!is_valid_email($email)) {
            flash("Invalid email address");
            $hasError = true;
        }
    } else {
        if (!is_valid_username($email)) {
            flash("Invalid username");
            $hasError = true;
        }
    }
    if (empty($password)) {
        flash("password must not be empty");
        $hasError = true;
    }
    if (!is_valid_password($password)) {
        flash("Password too short");
        $hasError = true;
    }
}
if (!$hasError) {
    //flash("Welcome, $email");
    //TODO 4
    $db = getDB();
}
```

12 mins db.ethereallab.app dsp82 En 63, Col 21 Spaces: 4 UTF-8 CRLF PHP 3:29 PM

PHP validation part 2

Checklist Items (0)

```
public_html > Project > login.php
67  if (isset($_POST["email"]) && isset($_POST["password"])) {
104      if (!$hasError) {
108          $stmt = $db->prepare("SELECT id, email, username, password from Users
109          where email = :email or username = :email");
110          try {
111              $r = $stmt->execute([":email" => $email]);
112              if ($r) {
113                  $user = $stmt->fetch(PDO::FETCH_ASSOC);
114                  if ($user) {
115                      $hash = $user["password"];
116                      unset($user["password"]);
117                      if (password_verify($password, $hash)) {
118                          //flash("welcome $email");
119                          $_SESSION["user"] = $user; //sets our session data from db
120                          //lookup potential roles
121                          $stmt = $db->prepare("SELECT Roles.name FROM Roles
122                          JOIN UserRoles on Roles.id = UserRoles.role_id
123                          where UserRoles.user_id = :user_id and Roles.is_active = 1 and UserRoles.is_active = 1");
124                          $stmt->execute([":user_id" => $user["id"]]);
125                          $roles = $stmt->fetchAll(PDO::FETCH_ASSOC); //fetch all since we'll want multiple
126                          //save roles or empty array
127                          if ($roles) {
128                              $_SESSION["user"]["roles"] = $roles; //at least 1 role
129                          } else {
130                              $_SESSION["user"]["roles"] = []; //no roles
131                          }
132                          flash("Welcome, " . get_username());
133                          die(header("Location: home.php"));
134                      } else {
135                          flash("invalid password");
136                      }
137                  }
138              }
139          }
140      }
141  }
```

```
136
137     }
138     } else {
139     }
140   }
141 } catch (Exception $e) {
142   flash("<pre>" . var_export($e, true) . "</pre>");

```

2 hrs 12 mins db.etherrealab.app dsp82

Ln 63, Col 21 Spaces: 4 UTF-8 CRLF PHP

PHP validation part3

Checklist Items (0)

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

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Don't just show code, translate things to plain English
<input checked="" type="checkbox"/> #2	1	Explain how the session works and why/how it's used

Response:

The login process begins with the user filling out the forms on the login page. The server-side validation and database will authenticate the user. If this proves successful the user's data is stored in a session, while granting access to a protected area of the website. Error handling will make sure any issues are accounted for.

Task #4 - Points: 1

Text: Include pull request links related to this feature

Details:

Should end in /pull/#

URL #1

<https://github.com/dsp82njit/dsp82-IT202-008/pull/42>

URL #2

<https://github.com/dsp82njit/dsp82-IT202-008/pull/37>

URL #3

<https://github.com/dsp82njit/dsp82-IT202-008/pull/45>

URL #4

<https://github.com/dsp82njit/dsp82-IT202-008/pull/34>

User Logout (1 pt.)

COLLAPSE

 COLLAPSE

Task #1 - Points: 1

Text: Capture the following screenshots

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Screenshot of the navigation when logged in (site)
<input checked="" type="checkbox"/> #2	1	Screenshot of the redirect to login with the user-friendly logged-out message (site)
<input checked="" type="checkbox"/> #3	1	Screenshot of the logout-related code showing the session is destroyed (code). Ensure ucid/date comment is present.

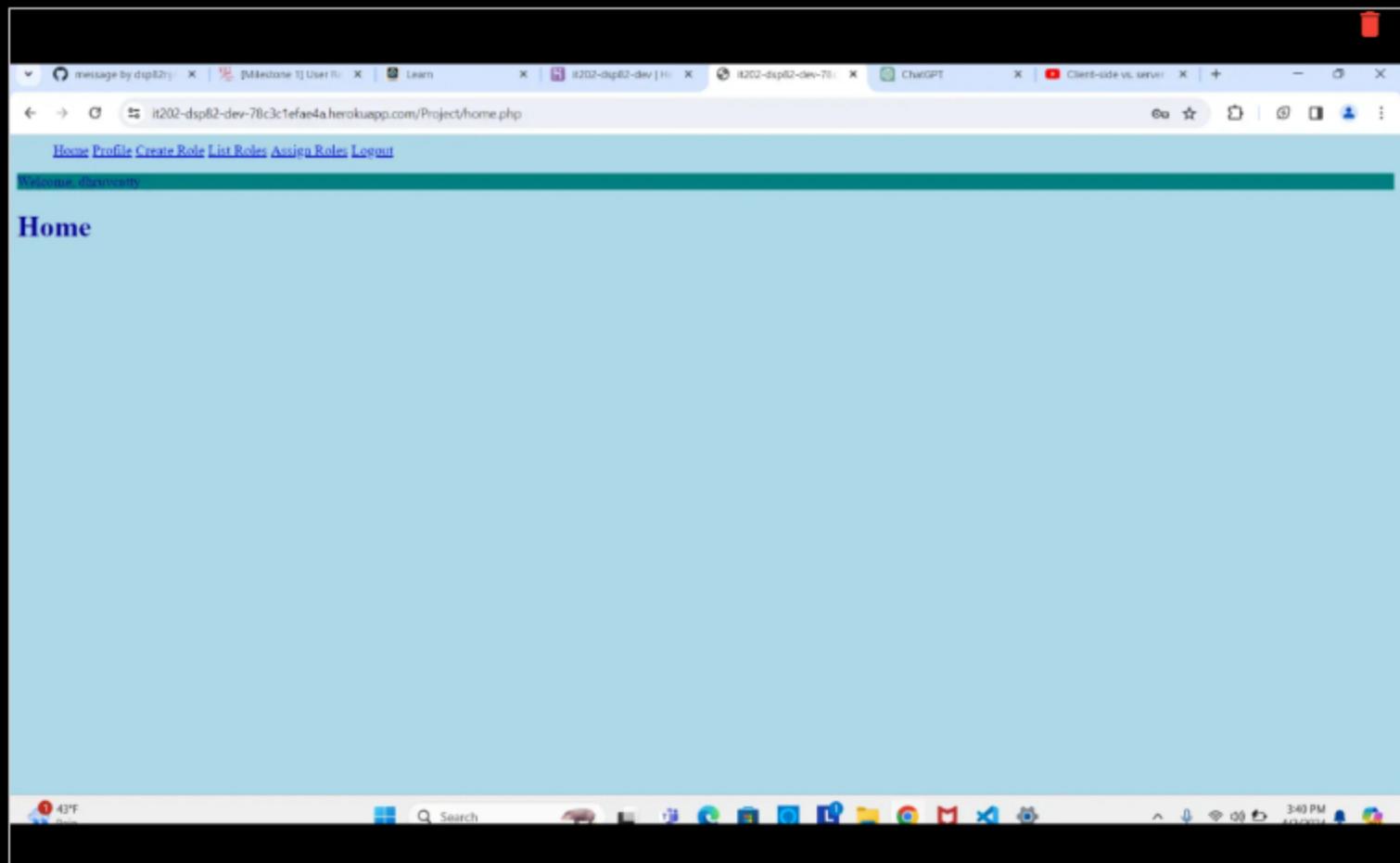
Task Screenshots:

Gallery Style: Large View

Small

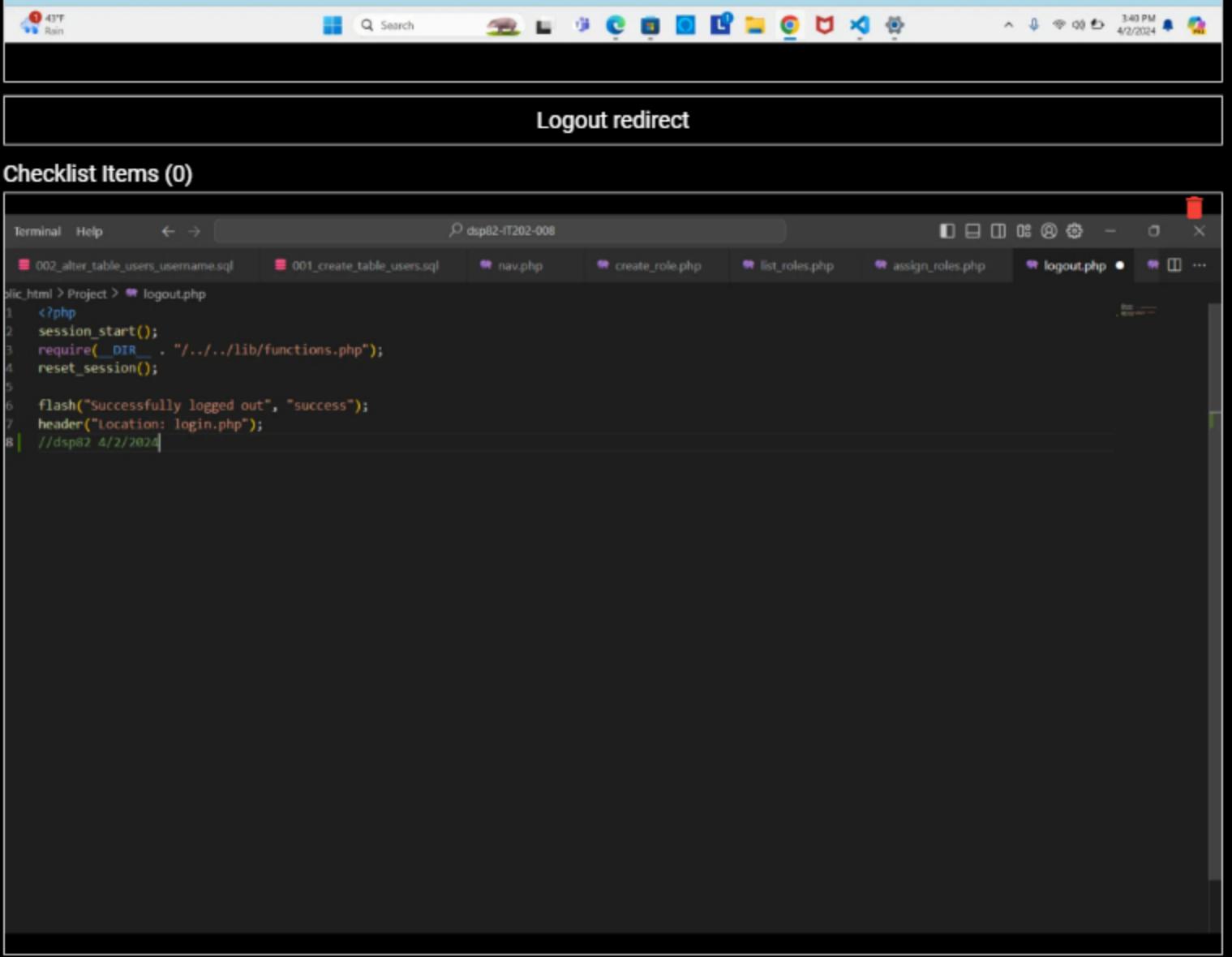
Medium

Large



Checklist Items (0)

A screenshot of a Microsoft Edge browser window. The address bar shows the URL 'it202-dsp82-dev-70c3c1efae4a.herokuapp.com/Project/login.php'. The page displays a login form with fields for 'Email/Username' and 'Password', and a 'Login' button. The entire form area is highlighted with a thick green border.



The screenshot shows a Windows desktop environment. At the top is the taskbar with various icons for search, file explorer, and other applications. The system tray shows the date and time as 4/2/2024 at 3:40 PM. In the center, a browser window is open with a title bar that says "Logout redirect". Below the browser is a code editor window titled "logout.php". The code editor shows the following PHP script:

```
<?php
session_start();
require(__DIR__ . "/../../lib/functions.php");
reset_session();

flash("Successfully logged out", "success");
header("location: login.php");
//dsp82 4/2/2024
```

Logout code

Checklist Items (0)

Task #2 - Points: 1

Text: Include pull request links related to this feature

i Details:

Should end in /pull/#

URL #1

<https://github.com/dsp82njit/dsp82-IT202-008/pull/50>

URL #2

<https://github.com/dsp82njit/dsp82-IT202-008/pull/47>

URL #3

<https://github.com/dsp82njit/dsp82-IT202-008/pull/36>

Basic Security Rules and Roles (2 pts.)

[^COLLAPSE ^](#)

Task #1 - Points: 1

Text: Authentication Screenshots

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Screenshot of the function that checks if a user is logged in
<input checked="" type="checkbox"/> #2	1	Screenshot of the login check function being used (i.e., profile likely). Also caption what pages it's used on
<input checked="" type="checkbox"/> #3	1	Include ucid/date code comments in all screenshots (one per screenshot is sufficient)
<input checked="" type="checkbox"/> #4	1	Demonstrate the user-friendly message of trying to manually access a login-protected page while being logged out

Task Screenshots:

Gallery Style: Large View

Small Medium Large



```

login.php M • index.php 002_alter_table_users_username.sql 001_create_table_users.sql nav.php create_role.php list_roles.php assign Td W ...
ablic_html > Project > login.php
15 <script>
16 //dsp82 4/2/2024
17 </script>
18 <?php
19 //TODO 2: add PHP Code
20 if (isset($_POST["email"]) && isset($_POST["password"])) {
21     $email = se($_POST, "email", "", false);
22     $password = se($_POST, "password", "", false);
23
24     //TODO 3
25     $hasError = false;
26     if (empty($email)) {
27         flash("Email must not be empty");
28         $hasError = true;
29     }
30     if (str_contains($email, "@")) {
31         //sanitize
32         //$email = filter_var($email, FILTER_SANITIZE_EMAIL);
33         $email = sanitize_email($email);
34         //validate
35     }
36 }
37
38 if ($hasError) {
39     //display error messages
40 }
41 else {
42     //process login
43 }
44
45 //redirect to dashboard
46 header("Location: dashboard.php");
47
48 //close connection
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81

```

```
82     //if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
83     //    flash("Invalid email address");
84     //    dsp82 4/2/2024 $hasError = true;
85   }/*
86   if (!is_valid_email($email)) {
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

bash + ⌂ ⌂ ... ⌂ ×

9738@thruwjay_96 MINGW64 ~/dsp82-IT202-008 (Milestone1)

Login check function

Checklist Items (0)

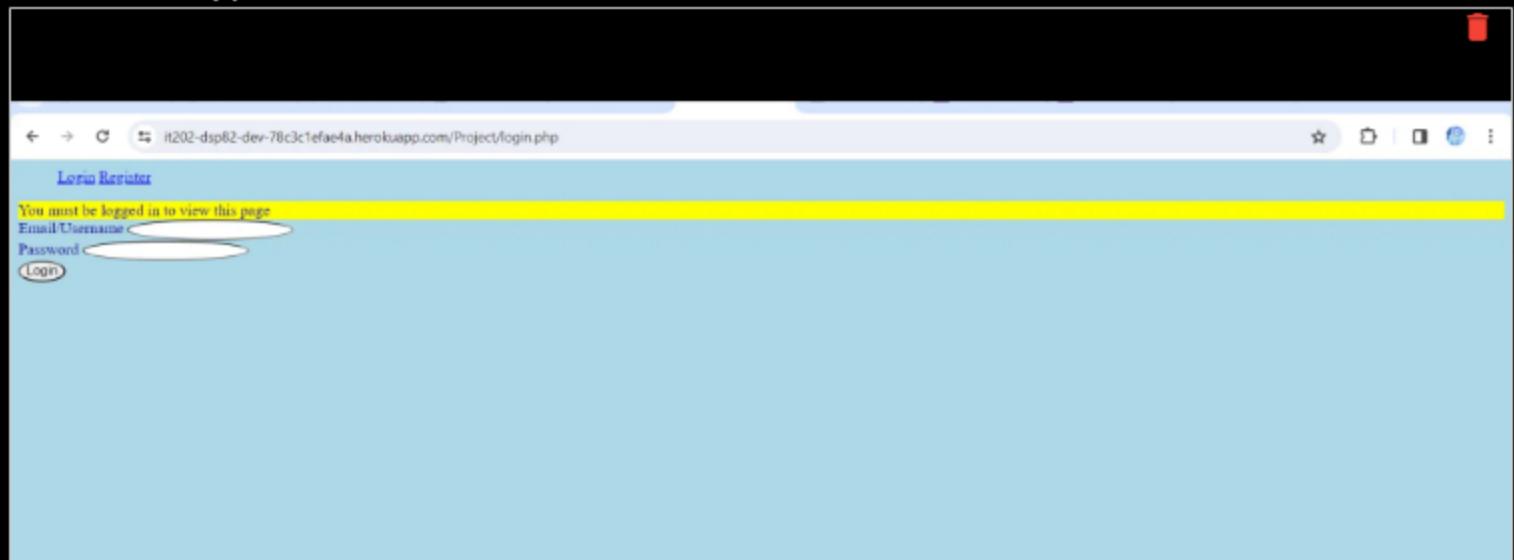
```
//dsp82 4/2/2024
?>

<!-- include css and js files -->
<link rel="stylesheet" href=<?php echo get_url('styles.css'); ?>>
<script src=<?php echo get_url('helpers.js'); ?>></script>
<nav>
<ul>
    <?php if (is_logged_in()) : ?>
        <li><a href=<?php echo get_url('home.php'); ?>>Home</a></li>
        <li><a href=<?php echo get_url('profile.php'); ?>>Profile</a></li>
    <?php endif; ?>
    <?php if (!is_logged_in()) : ?>
        <li><a href=<?php echo get_url('login.php'); ?>>Login</a></li>
        <li><a href=<?php echo get_url('register.php'); ?>>Register</a></li>
    <?php endif; ?>
    <?php if (has_role("Admin")) : ?>
        <li><a href=<?php echo get_url('admin/create_role.php'); ?>>Create Role</a></li>
        <li><a href=<?php echo get_url('admin/list_roles.php'); ?>>List Roles</a></li>
        <li><a href=<?php echo get_url('admin/assign_roles.php'); ?>>Assign Roles</a></li>
    <?php endif; ?>
    <?php if (is_logged_in()) : ?>
        <li><a href=<?php echo get_url('logout.php'); ?>>Logout</a></li>
    <?php endif; ?>
</ul>
```

EWS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Function that checks if user is logged in

Checklist Items (0)



Tried to access Profile.php but no access was granted

Checklist Items (0)

Task #2 - Points: 1

Text: Authorization Screenshots

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Screenshot of the function that checks for a specific role
<input type="checkbox"/> #2	1	Screenshot of the role check function being used. Also caption what pages it's used on
<input type="checkbox"/> #3	1	Include uid/date code comments in all screenshots (one per screenshot is sufficient)
<input type="checkbox"/> #4	1	Demonstrate the user-friendly message of trying to manually access a role-protected page while being logged out

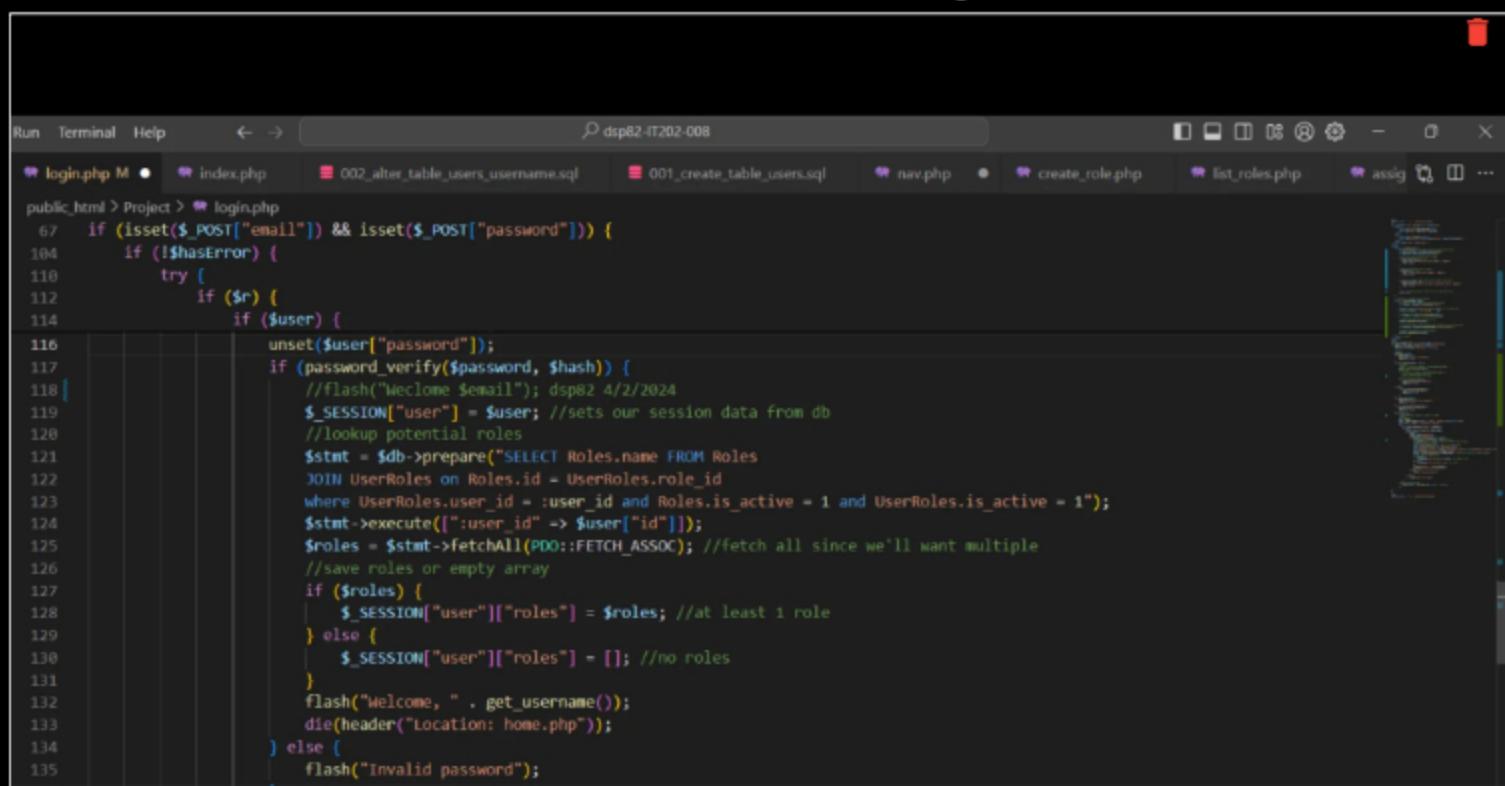
Task Screenshots:

Gallery Style: Large View

Small

Medium

Large



The screenshot shows a terminal window with several tabs open. The active tab contains PHP code for a login process. The code includes user input validation, password hashing, session management, and role assignment logic. The code is annotated with line numbers from 67 to 135.

```
public_html > Project > login.php
67 if (isset($_POST["email"]) && isset($_POST["password"])) {
104 if (!hasError) {
110     try {
112         if ($r) {
114             if ($user) {
116                 unset($user["password"]);
117                 if (password_verify($password, $hash)) {
118                     //flash("Welcome $email"); dsp82 4/2/2024
119                     $_SESSION["user"] = $user; //sets our session data from db
120                     //lookup potential roles
121                     $stmt = $db->prepare("SELECT Roles.name FROM Roles
122 JOIN UserRoles ON Roles.id = UserRoles.role_id
123 WHERE UserRoles.user_id = :user_id AND Roles.is_active = 1 AND UserRoles.is_active = 1");
124                     $stmt->execute([":user_id" => $user["id"]]);
125                     $roles = $stmt->fetchAll(PDO::FETCH_ASSOC); //fetch all since we'll want multiple
126                     //save roles or empty array
127                     if ($roles) {
128                         $_SESSION["user"]["roles"] = $roles; //at least 1 role
129                     } else {
130                         $_SESSION["user"]["roles"] = []; //no roles
131                     }
132                     flash("Welcome, " . get_username());
133                     die(header("Location: home.php"));
134                 } else {
135                     flash("Invalid password");
136                 }
137             }
138         }
139     }
140 }
141 }
```

Role check function being used

Checklist Items (0)



Shows user has no access to create_roles.php while being logged out

Checklist Items (0)

```
//dsp82 4/2/2024
?>
| <!-- include css and js files -->
| <link rel="stylesheet" href="=php echo get_url('styles.css'); ?&gt;"&gt;
| &lt;script src="<?=php echo get_url('helpers.js'); ?&gt;"&gt;&lt;/script&gt;
| &lt;nav&gt;
|   &lt;ul&gt;
|     &lt;?php if (is_logged_in()) : ?&gt;
|       &lt;li&gt;&lt;a href="<?=php echo get_url('home.php'); ?&gt;"&gt;Home&lt;/a&gt;&lt;/li&gt;
|       &lt;li&gt;&lt;a href="<?=php echo get_url('profile.php'); ?&gt;"&gt;Profile&lt;/a&gt;&lt;/li&gt;
|     &lt;?php endif; ?&gt;
|     &lt;?php if (!is_logged_in()) : ?&gt;
|       &lt;li&gt;&lt;a href="<?=php echo get_url('login.php'); ?&gt;"&gt;Login&lt;/a&gt;&lt;/li&gt;
|       &lt;li&gt;&lt;a href="<?=php echo get_url('register.php'); ?&gt;"&gt;Register&lt;/a&gt;&lt;/li&gt;
|     &lt;?php endif; ?&gt;
|     &lt;?php if (has_role("Admin")) : ?&gt;
|       &lt;li&gt;&lt;a href="<?=php echo get_url('admin/create_role.php'); ?&gt;"&gt;Create Role&lt;/a&gt;&lt;/li&gt;
|       &lt;li&gt;&lt;a href="<?=php echo get_url('admin/list_roles.php'); ?&gt;"&gt;List Roles&lt;/a&gt;&lt;/li&gt;
|       &lt;li&gt;&lt;a href="<?=php echo get_url('admin/assign_roles.php'); ?&gt;"&gt;Assign Roles&lt;/a&gt;&lt;/li&gt;
|     &lt;?php endif; ?&gt;</pre
```

```

<?php if (is_logged_in()) : ?>
    <li><a href=<?php echo get_url('logout.php'); ?>>Logout</a></li>
<?php endif; ?>

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Checks for user having the role "admin"

Checklist Items (0)

Task #3 - Points: 1

Text: Screenshots of UserRoles and Roles Tables

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	At least one valid and enabled User->Role reference (UserRoles table)
<input checked="" type="checkbox"/> #2	1	UserRoles Table should have id, user_id, role_id, is_active, created, and modified columns
<input checked="" type="checkbox"/> #3	1	Roles Table should have id, name, description, is_active, modified, and created columns
<input checked="" type="checkbox"/> #4	1	At least one valid and enabled Role (Roles table)
<input checked="" type="checkbox"/> #5	1	Ensure left panel or database name is present in each table screenshot (should contain your ucid)

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

The screenshot shows the MySQL Workbench interface with the UserRoles table selected. The table has the following data:

	id	user_id	role_id	is_active	created	modified
1	7	1	-1	1	2024-04-02 16:06:55	2024-04-02 16:06:55
2	8	5	1	0	2024-04-02 16:34:03	2024-04-02 16:37:47

At the bottom, there is a terminal window showing the command: 1973@lhuvcjay_96 MINGW64 ~/dsp82-IT202-008 [Milestone1].

A status bar at the bottom right indicates: You have Windows Subsystem for Linux (WSL) installed on your system. Do you want to install the recommended 'WSL' extension from Microsoft for it? with options to 'Install' or 'Show Recommendations'.



User Roles

Checklist Items (0)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
19730@dhruvjay_96: ~ /dsp82-IT202-008 (Milestone1)
$
```

You have Windows Subsystem for Linux (WSL) installed on your system. Do you want to install the recommended 'WSL' extension from Microsoft for it?

Install Show Recommendations

Roles table

Checklist Items (0)

Task #4 - Points: 1

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

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	What's the purpose of the UserRoles table?
<input checked="" type="checkbox"/> #2	1	How does Roles.is_active differ from UserRoles.is_active?

Response:

Roles and user roles tables are vital components of user management in an application. The roles table defines the different roles for each user and provides them with their permissions. The user roles table establishes a relationship between a user and their role. The two tables work hand in hand for managing users and their roles within an application.

Task #5 - Points: 1

Text: Include pull request links related to this feature

Details:

Should end in /pull/#

URL #1

<https://github.com/dsp82njit/dsp82-IT202-008/pull/48>

User Profile (2 pts.)

^COLLAPSE ^

Task #1 - Points: 1

Text: View Profile Website Page

Checklist

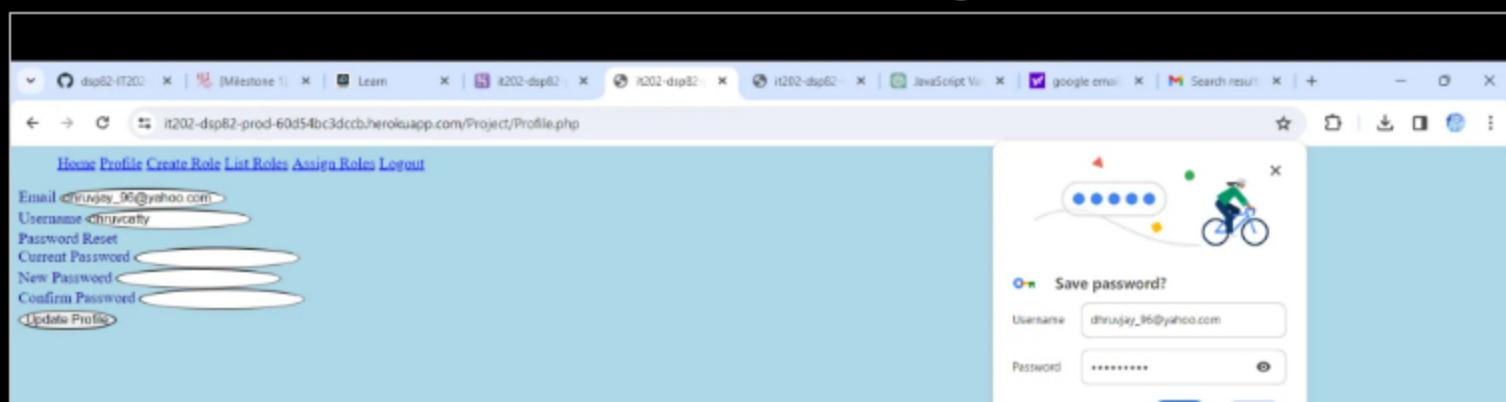
*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Heroku dev url should be present in the address bar
<input type="checkbox"/> #2	1	Should have thoughtful CSS applied
<input type="checkbox"/> #3	1	Show the profile form correctly populated on page load (username, email)
<input type="checkbox"/> #4	1	Should have the following fields: username, email, current password, new password, confirm password (or similar)

Task Screenshots:

Gallery Style: Large View

Small Medium Large



Passwords are saved to [Google Password Manager](#) on this device.



Profile page

Checklist Items (0)

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

Details:

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

Response:

When the profile page is visited, user data is fetched from the session and populates the form fields. The user can view this and modify their profile. The server-side validation manages submissions. This includes validation, database updates, and feedback for successful updates.

Task #3 - Points: 1

Text: Edit Profile Website Page

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Heroku dev url should be present in the address bar
<input checked="" type="checkbox"/> #2	1	Should have thoughtful CSS applied
<input checked="" type="checkbox"/> #3	1	Demonstrate with before and after of a username change (including success message)
<input checked="" type="checkbox"/> #4	1	Demonstrate with a before and after of an email change (including success message)

#5	1	Demonstrate the success message of updating password
#6	1	Demonstrate JavaScript user-friendly validation messages (email format, username format, password format, new password matching confirm password)
#7	1	Demonstrate PHP user-friendly validation messages (desired email is already in use, desired username is already in use, current password doesn't match what's in the DB)

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Home Profile Create Role List Roles Assign Roles Logout

Email

Username

Password Reset

Current Password

New Password

Confirm Password

before password change

Checklist Items (0)

Email saved

Password reset

Email

Username

Password Reset

Current Password

New Password

Confirm Password

after password is updated as well as username

Checklist Items (0)

← → C it202-dsp82-dev-78c3c1efae4a.herokuapp.com/Project/Profile.php

[Home](#) [Profile](#) [Create Role](#) [List Roles](#) [Assign Roles](#) [Logout](#)

Email (highlighted with a red oval)

Username (highlighted with a red oval)

Password Reset

Current Password

New Password

Confirm Password

Before username is changed

Checklist Items (0)

← → C it202-dsp82-dev-78c3c1efae4a.herokuapp.com/Project/Profile.php

[Home](#) [Profile](#) [Create Role](#) [List Roles](#) [Assign Roles](#) [Logout](#)

Current password is invalid

Email (highlighted with a red oval)

Username (highlighted with a red oval)

Password Reset

Current Password

New Password

Confirm Password

Password does not match DB

Checklist Items (0)

The chosen username is not available.

Email

Username

Password Reset

Current Password

New Password

Confirm Password

username already in use

Checklist Items (0)

New passwords don't match

Email

Username

Password Reset

Current Password

New Password

Confirm Password

NEw password dont match

Checklist Items (0)

A screenshot of a web browser window showing a profile update form. The URL in the address bar is `it202-dsp82-dev-78c3c1efae4a.herokuapp.com/Project/Profile.php`. The page title is "Profile". Below the title, there are several navigation links: Home, Profile, Create Role, List Roles, Assign Roles, and Logout. A red error message at the top states "Username must only contain 3-16 characters a-z, 0-9, _, or -". The form fields are as follows:

- Email: `chruvjay_96@yahoo.com`
- Username: `islam`
- Password Reset: (empty input field)
- Current Password: (empty input field)
- New Password: (empty input field)
- Confirm Password: (empty input field)
-

invalid username format

Checklist Items (0)

A screenshot of a web browser window showing a profile update form. The URL in the address bar is `it202-dsp82-dev-78c3c1efae4a.herokuapp.com/Project/Profile.php`. The page title is "Profile". Below the title, there are several navigation links: Home, Profile, Create Role, List Roles, Assign Roles, and Logout. A red error message at the top states "Username must only contain 3-16 characters a-z, 0-9, _, or -". The form fields are as follows:

- Email: `chruvjay_96@yahoo.com`
- Username: (empty input field)
- New Password: (empty input field)
- Confirm Password: (empty input field)
-

A tooltip for the Email field provides additional information: "!" is used at a wrong position in 'yahoo.com.'.

Wrong email format

Checklist Items (0)

Task #4 - Points: 1

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

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input checked="" type="checkbox"/> #1	1	Updating Username/Email
<input checked="" type="checkbox"/> #2	1	Updating password
<input checked="" type="checkbox"/> #3	1	Don't just show code, translate things to plain English

Response:

For updating email and username, the submitted data is retrieved and validated, ensuring it meets the criteria like non-emptiness and proper format. If validated the respective field in the database is updated. For updating password, the user enters a new password and matching password along with their old password. If the criteria is met for the old password, the new password is updated and checked to see if it meets the required criteria. If validated the new password is updated and hashed in the database.

Task #5 - Points: 1

Text: Include pull request links related to this feature

 Details:

Should end in /pull/#

<https://github.com/dsp82njit/dsp82-IT202-008/pull/42>

URL #2

<https://github.com/dsp82njit/dsp82-IT202-008/pull/41>

URL #3

<https://github.com/dsp82njit/dsp82-IT202-008/pull/39>

Misc (1 pt.)

[^COLLAPSE ^](#)

Task #1 - Points: 1

Text: Screenshot of wakatime

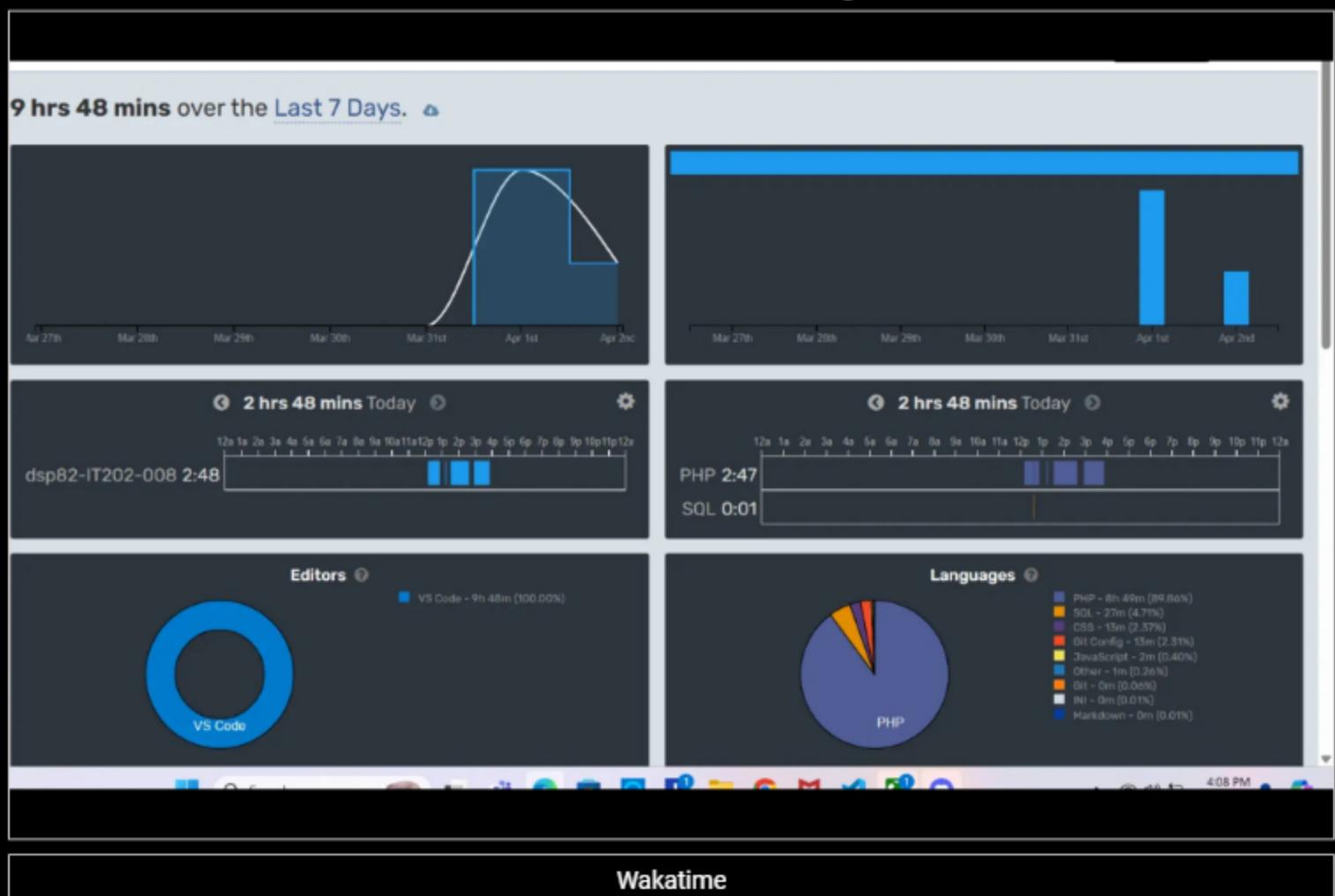
Task Screenshots:

Gallery Style: Large View

Small

Medium

Large



Task #2 - Points: 1

Text: Screenshot of your project board from GitHub (tasks should be in the proper column)

Task Screenshots:

Small

Medium

Large

Files

main

Go to file

lib

partials

public_html

Project

Challenge.html

README.md

index.php

problem1.php

problem2.php

problem3.php

test_db.php

.gitignore

.htaccess

Profile

README.md

composer.json

composer.lock

dsp82_generic-git-readings-via-L...

dsp82-IT202-008 / public_html /

dsp82njit message

69ffbe4 · 3 hours ago · History

Name	Last commit message	Last commit date
..		
Project	message	3 hours ago
Challenge.html	message	2 months ago
README.md	Save all changes	2 months ago
index.php	Saying hi	2 months ago
problem1.php	save changes	2 months ago
problem2.php	save changes	2 months ago
problem3.php	save changes	2 months ago
test_db.php	Save all changes	2 months ago

README.md

Put files here that you want to be publicly accessible via the url

These are the user facing files and typically include/require other files from lib or partials

These will follow the domain name after the initial /

43°F Light rain

Search

423 PM 4/2/2024

Project Board

Task #3 - Points: 1

Text: Provide a direct link to the project board on GitHub

URL #1

https://github.com/dsp82njit/dsp82-IT202-008/tree/main/public_html/Project

Task #4 - Points: 1

Text: Provide a direct link to the login page from your prod instance

URL #1

<https://it202-dsp82-prod-60d54bc3dccb.herokuapp.com/Project/login.php>

