

# Submission Worksheet

**CLICK TO GRADE**

<https://learn.ethereallab.app/assignment/IT202-008-S2024/it202-milestone-1-2024/grade/jn426>

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

## Submissions:

Submission Selection

1 Submission [active] 4/1/2024 10:34:35 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 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 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 checked="" type="checkbox"/> #4	1	Demonstrate email already in use message (message text doesn't need to be exact, but should be clear)
<input checked="" type="checkbox"/> #5	1	Demonstrate username already in use message (message text doesn't need to be exact, but should be clear)
<input checked="" type="checkbox"/> #6	1	Demonstrate user-friendly message of new account being created

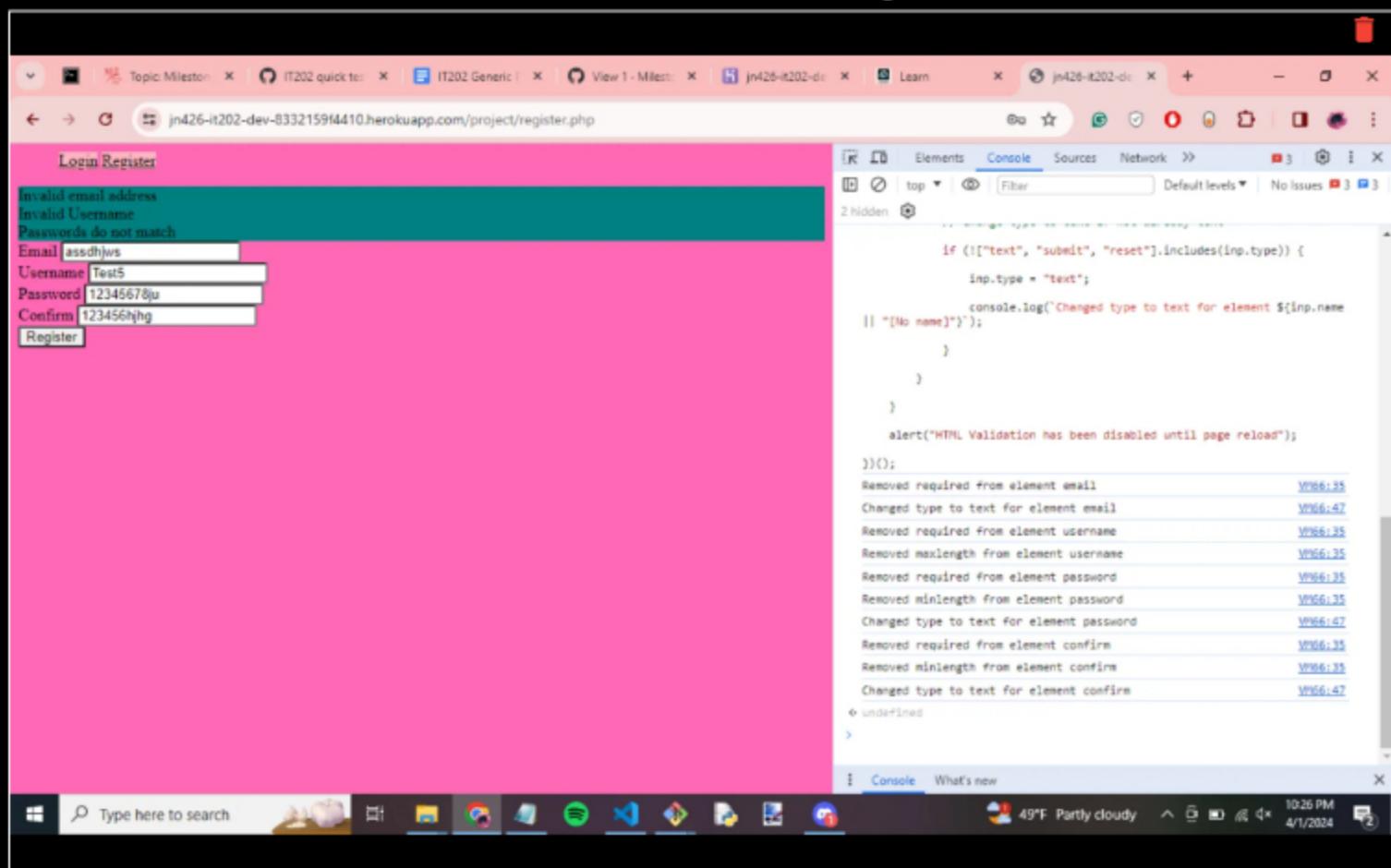
### Task Screenshots:

#### Gallery Style: Large View

Small

Medium

Large



JS Validation for #3

Checklist Items (3)

#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)

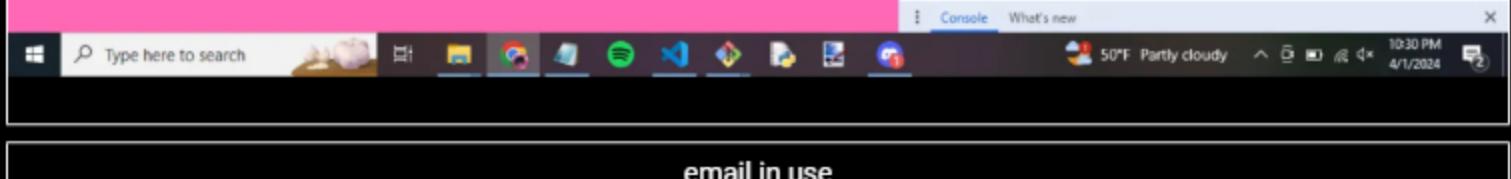
The screenshot shows a web browser window with multiple tabs open at the top. The active tab is for a registration page at [jn426-it202-dev-8332159f4410.herokuapp.com/project/register.php](http://jn426-it202-dev-8332159f4410.herokuapp.com/project/register.php). The page has a pink header with 'Login Register' buttons. Below is a form with four input fields: 'Email', 'Username', 'Password', and 'Confirm'. A red error message 'Email is required' is displayed above the first field. The browser's developer tools are open, showing the 'Console' tab which is currently empty. At the bottom of the screen, the Windows taskbar is visible with various pinned icons and the system tray showing the date and time.

user friendly msg of acc being made

Checklist Items (1)

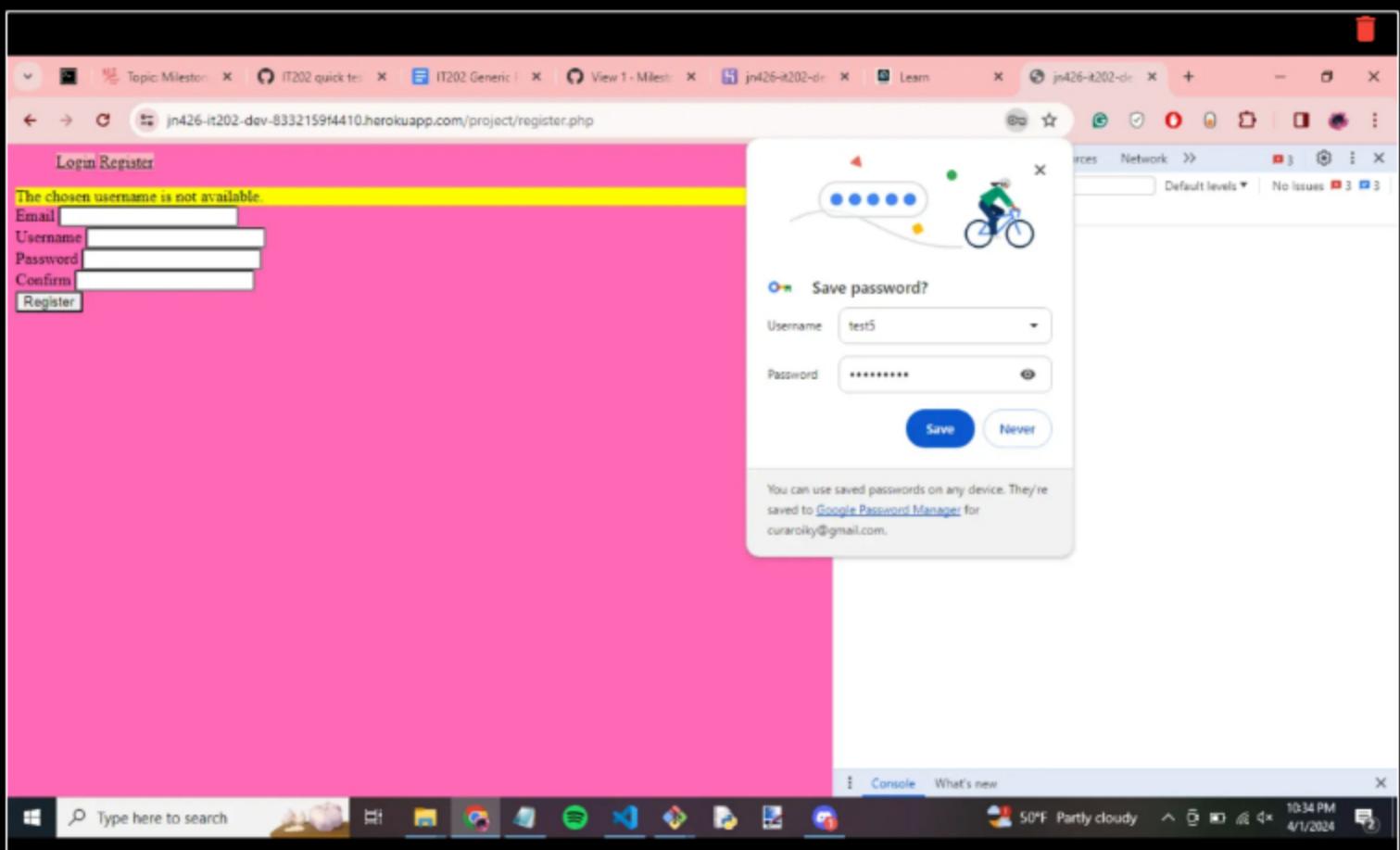
#6 Demonstrate user-friendly message of new account being created

This screenshot shows the same registration page as the previous one, but with a different error message. The 'Email' field now contains the text 'The chosen email is not available.' This message is displayed in a yellow box above the 'Email' input field. The rest of the interface and developer tools are identical to the previous screenshot.



## Checklist Items (1)

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



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

The screenshot shows a code editor interface with a dark theme. The left sidebar (Explorer) lists files like login.php, profile.php, register.php, and helpers.js. The main editor area displays PHP code for a registration form. The code includes client-side validation using JavaScript (validate.js) and server-side validation using PHP. The terminal tab shows a command-line session with a bash prompt. The bottom status bar provides system information such as the date and time.

```
public_html > project > register.php > validate
You, 3 hours ago | 1 author (You)
1 <?php
2 require(__DIR__ . "/../../partials/nav.php");
3 reset_session();
4 ?>
5 <form onsubmit="return validate(this)" method="POST">
6   <div>
7     <label for="email">Email</label>
8     <input type="email" name="email" required />
9   </div>
10  <div>
11    <label for="username">Username</label>
12    <input type="text" name="username" required maxlength="30" />
13  </div>
14  <div>
15    <label for="pw">Password</label>
16    <input type="password" id="pw" name="password" required minlength="8" />
17  </div>
18  <div>
19    <label for="confirm">Confirm</label>
20    <input type="password" name="confirm" required minlength="8" />
21  </div>
22  <input type="submit" value="Register" />
23 </form>
24 <script>
25   function validate(form) {
26     ...
27   }
28</script>
```

### register form

#### 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

The screenshot shows the VS Code interface with the file `helpers.js` open in the center editor tab. The code implements several validation functions:

```
function flash(message = "", color = "info") {
    let innerDiv = document.createElement("div");
    //apply the CSS (these are bootstrap classes which we'll learn later)
    innerDiv.className = `alert alert-${color}`;
    //set the content
    innerDiv.innerText = message;

    outerDiv.appendChild(innerDiv);
    //add the element to the DOM (if we don't it merely exists in memory)
    flash.appendChild(outerDiv);
}

function isValidEmail(email) {
    return /^[^@\s]+@[^\s]+\.[^\s]+$/i.test(email.trim());
}

function isValidUsername(username) {
    return /^[a-zA-Z0-9_-]{3,16}$/.test(username);
}

function isValidPassword(password) {
    return password.length >= 8;
}
```

The status bar at the bottom indicates the file was last modified 2 weeks ago, has 28 lines, and is in JavaScript mode.

## Extra file for JS validation

### Checklist Items (1)

#1 Show the JavaScript validations (include any extra files related)

The screenshot shows the VS Code interface with the file `register.php` open in the center editor tab. The code contains a validation function:

```
<script>
function validate(form) {
    let username = form.username.value.trim();
    let password = form.password.value;
    let confirm = form.confirm.value;
    let isValid = true;

    if (!isValidEmail(email)) {
        flash("Invalid email address");
        isValid = false;
    }

    // Validate username
    if (!isValidUsername(username)) {
        flash("Invalid Username");
        isValid = false;
    }

    // Validate password
    if (!isValidPassword(password)) {
        flash("Invalid Password");
        isValid = false;
    }

    //confirm
    if (password !== confirm) {
        flash("Passwords do not match");
        isValid = false;
    }

    return isValid;
}
```

The status bar at the bottom indicates the file was last modified 3 hours ago, has 58 lines, and is in PHP mode.

## Checklist Items (1)

### #1 Show the JavaScript validations (include any extra files related)

The screenshot shows the VS Code interface with the title bar "JN426-IT202-008". The left sidebar displays a file tree for a project named "JN426-IT202-008" under ".vscode" and "lib". The "lib" folder contains several PHP files: config.php, db.php, duplicate\_user\_details.php, flash\_messages.php, functions.php, get\_url.php, README.md, reset\_session.php, safer\_echo.php, sanitizers.php, and user\_helpers.php. The "user\_helpers.php" file is currently open in the editor. The code in the editor is as follows:

```
1 <?php
2 /**
3  * Passing $redirect as true will auto redirect a logged out user to the $destination.
4  * The destination defaults to login.php
5  */
6
7 function is_logged_in($redirect = false, $destination = "login.php")
8 {
9     $logged_in = isset($_SESSION["user"]);
10    if ($logged_in && !isloggedin) {
11        // If this triggers, the calling script won't receive a reply since die()/exit() terminates it
12        // Ready "You must be logged in to view this page", "warning"
13        die(header("Location: $destination"));
14    } <- #19-18 if ($redirect && !$logged_in)
15    return $logged_in;
16 } <- #18 Function is_logged_in($redirect = false, $destination = "logi...
17 function has_role($role)
18 {
19     if (is_logged_in() && isset($_SESSION["user"]["roles"])) {
20         foreach ($_SESSION["user"]["roles"] as $r) {
21             if ($r["name"] === $role) {
22                 return true;
23             }
24         } // You, last month + Helper Functions <- #20-24 foreach ($_SESSION["user"]["roles"] as $r)
25     } <- #19-25 if (is_logged_in() && isset($_SESSION["user"]["roles"]))
26     return false;
27 } <- #18-27 Function has_role($role)
28 function get_username()
29 {
30     if (is_logged_in()) { // We need to check for login first because "user" key may not exist
31         return $_SESSION["user"], "username", "", false;
32     }
33     return "";
34 } <- #20-34 Function get_username()
35 function get_user_email()
36 {
37     if (is_logged_in()) { // We need to check for login first because "user" key may not exist
38         return $_SESSION["user"], "email", "", false;
39     }
40     return "";
41 } <- #35-41 Function get_user_email()
42 function get_user_id()
43 {
44     if (is_logged_in()) { // We need to check for login first because "user" key may not exist
45         return $_SESSION["user"], "id", false, false;
46     }
47     return false;
48 } <- #41-48 Function get_user_id()
```

The status bar at the bottom shows "You, last month + Helper functions" and "10:47 PM 4/1/2024".

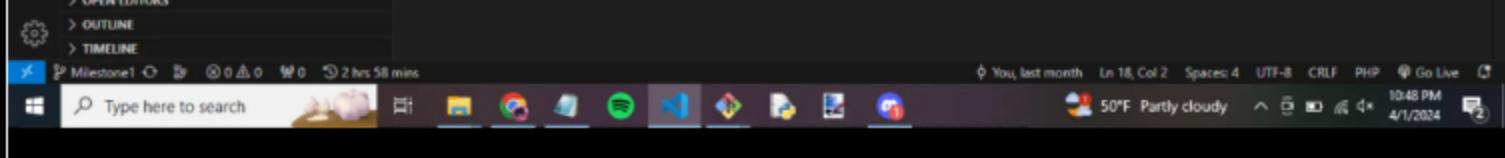
## User helpers (lib file1)

## Checklist Items (1)

### #2 Show the PHP validations (include any lib content)

The screenshot shows the VS Code interface with the title bar "JN426-IT202-008". The left sidebar displays a file tree for a project named "JN426-IT202-008" under ".vscode" and "lib". The "lib" folder contains several PHP files: config.php, db.php, duplicate\_user\_details.php, flash\_messages.php, functions.php, get\_url.php, README.md, reset\_session.php, safer\_echo.php, sanitizers.php, and user\_helpers.php. The "sanitizers.php" file is currently open in the editor. The code in the editor is as follows:

```
1 <?php
2
3 function sanitize_email($email = "") {
4     return filter_var(trim($email), FILTER_SANITIZE_EMAIL);
5 }
6
7 function is_valid_email($email = "") {
8     return filter_var(trim($email), FILTER_VALIDATE_EMAIL);
9 }
10
11 function is_valid_username($username) {
12     return preg_match('/^[\w-]{3,16}$/', $username);
13 }
14
15 function is_valid_password($password) {
16     return strlen($password) >= 8;
17 } // You, last month + Helper functions
```



## lib file 2

### Checklist Items (1)

#2 Show the PHP validations (include any lib content)

```
register.php
public_html > project > register.php > script > validate

58 <?php
59 // add PHP Code
60 if (isset($_POST["email"]) && isset($_POST["password"]) && !empty($_POST["email"])) {
61     $email = se($_POST, "email", "", false);
62     $password = se($_POST, "password", "", false);
63     $confirm = se($_POST, "confirm", "", false);
64     $username = se($_POST, "username", "", false);
65 }
66 //1000
67 $hasError = false;
68 if (empty($email)) {
69     flash("Email must not be empty", "danger");
70     $hasError = true;
71 }
72 //sanitize
73 $email = sanitize_email($email);
74 //validate
75 if (!is_valid_email($email)) {
76     flash("Invalid email address", "danger");
77     $hasError = true;
78 }
79 if (!preg_match('/^([a-zA-Z0-9_-]{3,16})$/i', $username)) {
80     flash("Username must only contain 3-16 characters a-z, A-Z, 0-9, _-");
81     $hasError = true;
82 }
83 if (empty($password)) {
84     flash("Password must not be empty", "danger");
85     $hasError = true;
86 }
87 if (empty($confirm)) {
88     flash("Confirm password must not be empty", "danger");
89     $hasError = true;
90 }
91 if (strlen($password) < 8) {
92     flash("Password too short", "danger");
93     $hasError = true;
94 }
95 if (
96     strlen($password) > 0 && $password !== $confirm
97 ) {
98     flash("Passwords must match", "danger");
99     $hasError = true;
100 }
101 if (!$hasError) {
102     //1000
103     $hash = password_hash($password, PASSWORD_BCRYPT);
104     $db = getDB();
105     $stmt = $db->prepare("INSERT INTO Users (email, password)");
106     try {
107         $stmt->execute([":email" => $email, ":password" => $hash]);
108         flash("Successfully registered!", "success");
109     } catch (PDOException $e) {
110         users_check_duplicate($e->errorInfo);
111     }
112 }
113 } <- #102-113 if (!$hasError)
114 } <- #102-114 if (isset($_POST["email"]) && isset($_POST["password"]))
115 >>
116 <?php
117 require(__DIR__ . "/../../partials/flash.php");
```

## php register

### Checklist Items (1)

#2 Show the PHP validations (include any lib content)

### 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
<input checked="" type="checkbox"/>	#1 1	Password should be hashed
<input checked="" type="checkbox"/>	#2 1	Should have email, password, username (unique), created, modified, and id fields
<input checked="" type="checkbox"/>	#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 the MySQL Workbench interface. On the left is the database browser tree, which includes the schema 'IT202Server', the table 'Users', and its columns: id, email, password, created, modified, and username. The main pane displays the results of the query: 'SELECT \* FROM `Users` LIMIT 100'. The results table has 9 rows of data. The bottom of the screen shows the Windows taskbar with various icons and the system tray.

	id	email	password	created	modified	username
1	1	anothaonebiatch@gmail.com	\$2y\$10\$eWjC1jNZp5ho	2024-02-24 07:32:26	2024-02-24 07:32:26	anothaonebiatch
3	3	curaroiky@gmail.com	\$2y\$10\$eRZNZBhw8mp	2024-02-24 07:32:48	2024-02-24 07:32:48	curaroiky
5	5	mother@father.son	\$2y\$10\$yw4fN4zZqjxn1	2024-03-19 01:11:35	2024-03-19 01:11:35	mother
6	6	test1@yahoo.com	\$2y\$10\$9Ei7NwgAuNm	2024-03-19 02:35:08	2024-03-19 02:35:08	test1
7	7	test2@yahoo.com	\$2y\$10\$3idgZ/Jt7lsyY9/	2024-03-19 03:13:36	2024-03-19 03:13:36	test2
8	8	test3@test.com	\$2y\$10\$VGiq1hpO8XgC	2024-04-01 22:37:28	2024-04-01 22:37:28	test3
9	9	test3@gmail.com	\$2y\$10\$2FyCe8Hk55TQ	2024-04-02 02:29:18	2024-04-02 02:29:18	test5

### users

#### 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:

1. Page Load: HTML form with JavaScript validation and PHP handling.

2. Form Submission:

- JavaScript validates email, username, password, and confirmation.

- If validation fails, error messages are shown.

- The data is then sent to the server via HTTP POST.

- If validation passes, data is submitted to the server.

### 3. Server-side Validation (PHP):

- PHP validates and sanitizes form data.
- Checks for empty fields, valid email format, and username constraints.
- Password and confirmation are checked for match.
- Flash messages display errors or success.

### 3. Database Interaction:

- If validation passes, password is hashed.
- Database connection established, user data inserted.
- Success or error messages flashed accordingly.

### 4. Final Result:

- Success message displayed if registration is successful.
- Error messages prompt user to correct input.

#### Task #6 - Points: 1

**Text:** Include pull request links related to this feature

#### ⓘ Details:

Should end in /pull/#

#### URL #1

<https://github.com/jennatnguyen/JN426-IT202-008/pull/24>

#### 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



▼ EXPAND ▼

## Task #4 - Points: 1

**Text:** Include pull request links related to this feature



User Logout (1 pt.)

▲ 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



login

## Checklist Items (0)

A screenshot of a web browser window. The address bar shows the URL: `jn426-it202-dev-8332159f4410.herokuapp.com/project/login.php`. The page content is a login form with fields for Email/Username and Password, and a Login button. A green banner at the top says "Successfully logged out". The browser has multiple tabs open in the background.

log out

## Checklist Items (0)

A screenshot of a code editor (VS Code) showing a file named `logout.php` in the center pane. The code contains PHP script to log the user out. The left sidebar shows a project structure with files like `index.php`, `login.php`, and `register.php`. The bottom status bar shows the terminal command `Jenina@DESKTOP-CSIBM4Q9 MINION64 /c/JN426-IT202-008 (Milestone1)`.

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

flash("Successfully logged out", "success");
header("Location: login.php");
```

## 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/jennatnguyen/JN426-IT202-008/pull/38>

## 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

```

File Edit Selection View Go Run ...
EXPLORER JN426-IT202-008 .vscode lib README.md
login.php user_helpers.php profile.php register.php Users
lib > user_helpers.php > has_role
You, 2 weeks ago | 1 author (You)
<?php
/**
 * Passing $redirect as true will auto redirect a logged out user to the $destination.
 * The destination defaults to login.php
 */
function is_logged_in($redirect = false, $destination = "login.php")
{
    $isLoggedIn = isset($_SESSION["user"]);
    if ($redirect && !$isLoggedIn) {
        //if this triggers, the calling script won't receive a reply since die()/exit() terminates it
        flash("You must be logged in to view this page", "warning");
        die(header("Location: $destination"));
    } <- #10-14 if ($redirect && !$isLoggedIn)
}

```

```
15     return $isloggedin;
16 } <- #8-16 function is_logged_in($redirect = false, $destination = "logi...
17 function has_role($role)
18 {
19     if (is_logged_in() && isset($_SESSION["user"]["roles"])) {
20         foreach ($_SESSION["user"]["roles"] as $r) {
21             if ($r["name"] === $role) {
22                 return true;
23             }
24         }
25     }
26 } <- #19-25 if (is_logged_in() && isset($_SESSION["user"]["roles"]))
27 } <- #18-27 function has_role($role)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

Jenna@DESKTOP-CSBHQ4Q MINGW64 /c/JN426-IT202-008 (Milestone1)

48°F Mostly clear 12:02 AM 4/2/2024

checks if user is logged in

## Checklist Items (1)

### #1 Screenshot of the function that checks if a user is logged in

```
1 <?php
2 require_once(__DIR__ . "/../../partials/nav.php");
3 is_logged_in(true); You, 2 weeks ago * housekeeping
4 ?>
5 <?php
6 if (isset($_POST["save"])) {
7     $email = se($_POST, "email", null, false);
8     $username = se($_POST, "username", null, false);
9     $hasError = false;
10    //sanitize
11    $email = sanitize_email($email);
12    //validate
13    if (!is_valid_email($email)) {
14        flash("Invalid email address", "danger");
15        $hasError = true;
16    }
17    if (!is_valid_username($username)) {
18        flash("Username must only contain 3-16 characters a-z, 0-9, _, or -, "danger");
19        $hasError = true;
20    }
21    if (!$hasError) {
22        $params = [":email" => $email, ":username" => $username, ":id" => get_user_id()];
23        $db = getDB();
24        $stmt = $db->prepare("UPDATE Users set email = :email, username = :username where id = :id");
25        try {
26            $stmt->execute($params);
27            flash("Profile saved", "success");
        }
```

EXPLORER login.php user\_helpers.php profile.php register.php Users

File Edit Selection View Go Run ...

JN426-IT202-008

lib reset\_session.php safer\_echo.php sanitizers.php user\_helpers.php

partials public\_html assignment2 M6 project admin sql helpers.js home.php index.php login.php logout.php

profile.php register.php styles.css challenge.html index.php README.md test\_db.php

.gitignore

OPEN EDITORS

OUTLINE

TIMELINE

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

Jenna@DESKTOP-CSBHQ4Q MINGW64 /c/JN426-IT202-008 (Milestone1)

48°F Mostly clear 12:03 AM 4/2/2024

function in profile

## Checklist Items (0)

```
1 <?php
2 require(__DIR__ . "/../../partials/nav.php");
3 ?>
4 <h1>Home</h1>
5 <?php
6 if (is_logged_in(true)) {
7     //Comment this out if you don't want to see the execution confirmation
```

EXPLORER login.php home.php profile.php register.php Users

File Edit Selection View Go Run ...

JN426-IT202-008

lib reset\_session.php safer\_echo.php sanitizers.php user\_helpers.php

partials public\_html

home.php

profile.php register.php

Users

Type to search

is\_log Aa ab ? of 1 ↑ ↓ ⌂ ⌂

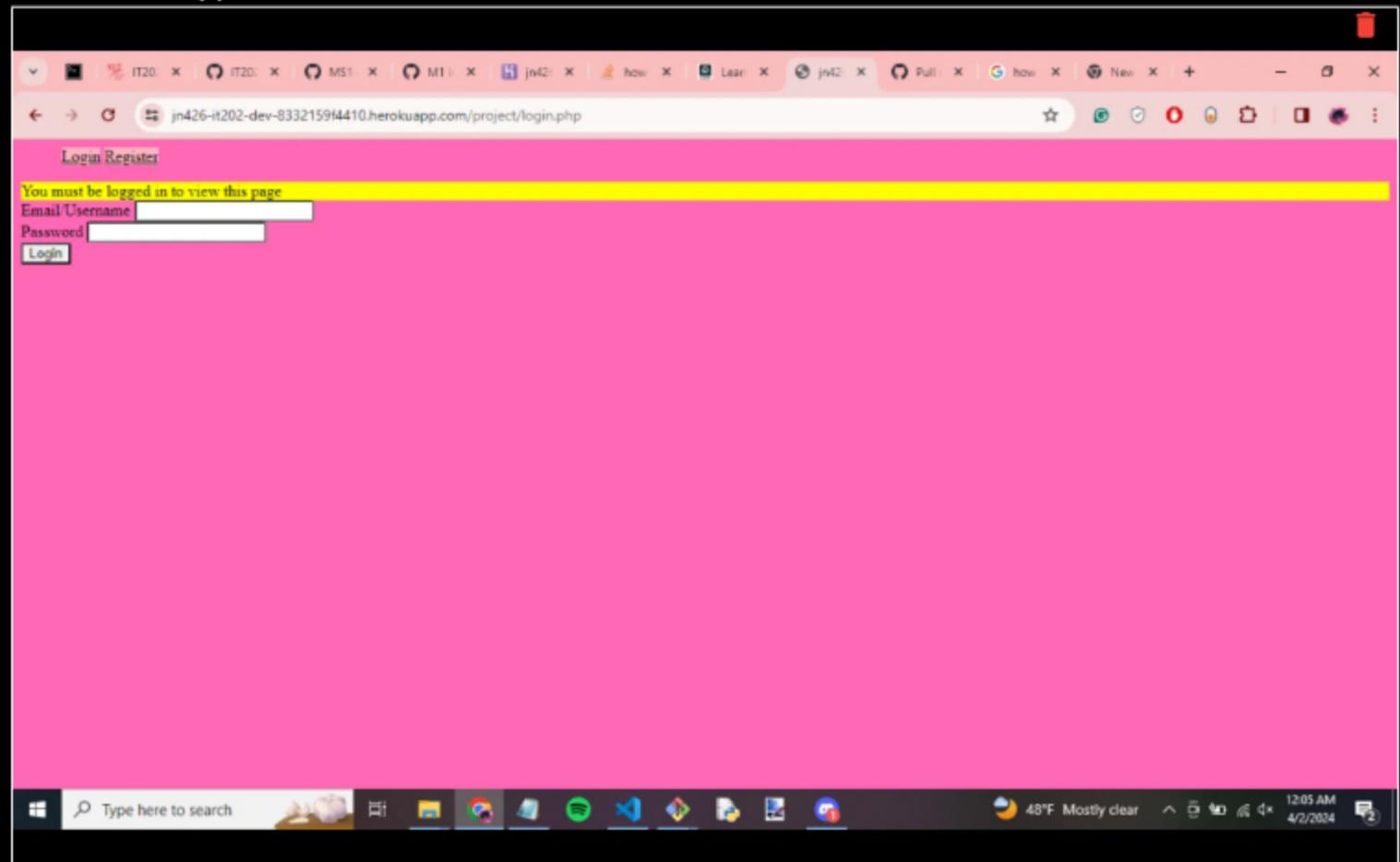
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

Jenna@DESKTOP-CSBHQ4Q MINGW64 /c/JN426-IT202-008 (Milestone1)

A screenshot of the Visual Studio Code interface. The top menu bar includes 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', 'PORTS', and 'GITLENS'. The 'TERMINAL' tab is selected, showing the command line output: 'Denna@DESKTOP-CSBHQ4Q MINGW64 /c/JN426-IT202-008 (Milestone1) \$'. Below the terminal is a status bar with 'You last month', 'Ln 14, Col 3', 'Spaces: 4', 'UTF-8', 'CRLF', 'PHP', 'Go Live', and a refresh icon. The bottom of the screen features a row of icons for file operations like Open, Save, Find, and others.

### **function in home**

## Checklist Items (0)



user friendly msd

## Checklist Items (0)

Task #2 - Points: 1

Text: Authorization Screenshots

#	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

```

File Edit Selection View Go Run ...
EXPLORER JN426-IT202-008 .vscode lib login.php user_helpers.php profile.php register.php Users
lib > user_helpers.php > has_role
    function is_logged_in($redirect = false, $destination = "login.php")
        flash("You must be logged in to view this page", "warning");
        die(header("Location: $destination"));
    } <- #10-14 if ($redirect && !$isLoggedIn)
        return $isLoggedIn;
    } <- #8-16 function is_logged_in($redirect = false, $destination = "log...
function has_role($role)
{
    You, last month (February 24th, 2024 3:26 AM)

    Helper functions
    dd71dbe < > ⌂ ⌂ ⌂ ⌂ ⌂ Connect to GitHub... | Team... | ...
    + }
    Changes added in dd71dbe | 
}
You, last month + Helper functions <- #18-27 function has_role($role)
function get_username()
{
    if (is_logged_in()) { //we need to check for login first because "user" key may not exist
        return se($_SESSION["user"], "username", "", false);
    }
    return "";
} <- #29-34 function get_username()
function get_user_email()
{
    if (is_logged_in()) { //we need to check for login first because "user" key may not exist
        return se($_SESSION["user"], "email", null, false);
    }
}

```

## has roles function

## Checklist Items (0)

```

File Edit Selection View Go Run ...
EXPLORER JN426-IT202-008 .vscode public_html project admin assign_roles.php Users
public_html > project > admin > assign_roles.php > ...
> has_role 1 of 1
You, 2 weeks ago | 1 author (You)
<?php
//note we need to go up 1 more directory
require(__DIR__ . "/../../../../partials/nav.php");
if (!has_role("Admin")) {
    You, 2 weeks ago + user roles
    flash("You don't have permission to view this page", "warning");
    die(header("Location: $BASE_PATH" . "/home.php"));
}
//attempt to apply
if (isset($_POST["users"]) && isset($_POST["roles"])) {
    $user_ids = $_POST["users"]; //se() doesn't like arrays so we'll just do this
    $role_ids = $_POST["roles"]; //se() doesn't like arrays so we'll just do this
    if (empty($user_ids) || empty($role_ids)) {
        You, 2 weeks ago + user roles
        flash("You must select users and roles to assign", "warning");
        die(header("Location: $BASE_PATH" . "/assign_roles.php"));
    }
    foreach ($user_ids as $user_id) {
        foreach ($role_ids as $role_id) {
            You, 2 weeks ago + user roles
            $query = "INSERT INTO user_roles (user_id, role_id) VALUES ($user_id, $role_id)";
            $result = mysqli_query($conn, $query);
            if (!$result) {
                You, 2 weeks ago + user roles
                flash("An error occurred while assigning roles", "error");
                die(header("Location: $BASE_PATH" . "/assign_roles.php"));
            }
        }
    }
    You, 2 weeks ago + user roles
    flash("Roles assigned successfully", "success");
    die(header("Location: $BASE_PATH" . "/home.php"));
}

```

```

    assign_roles.php
    create_role.php
    list_roles.php
    > sql
    helpers.js
    home.php
    index.php
    login.php
    logout.php
    profile.php
    register.php
    # styles.css
    > OPEN EDITORS
    > OUTLINE
    > TIMELINE
    M Milestone1 O 0 0 0 0 0 4 mins IT202Server jn426 You, 2 weeks ago Ln 5, Col 14 (8 selected) Spaces: 4 UTF-8 CRLF PHP Go Live
    Type here to search 48°F Partly cloudy 12:31 AM 4/2/2024

```

## hasroles in assign\_roles

### Checklist Items (0)

```

    File Edit Selection View Go Run ...
    JN426-IT202-008
    EXPLORER ... login.php profile.php register.php create_role.php Users
    lib get_url.php README.md reset_session.php safer_echo.php sanitizers.php user_helpers.php
    partials
    public_html assignment2 M6
    project admin assign_roles.php create_role.php list_roles.php > sql helpers.js home.php index.php login.php logout.php profile.php register.php # styles.css
    > OPEN EDITORS
    > OUTLINE
    > TIMELINE
    M Milestone1 O 0 0 0 0 0 4 mins IT202Server jn426 You, 2 weeks ago Ln 46, Col 3 Spaces: 4 UTF-8 CRLF PHP Go Live
    Type here to search 48°F Partly cloudy 12:34 AM 4/2/2024

```

## has role in create\_role

### Checklist Items (0)

```

    File Edit Selection View Go Run ...
    JN426-IT202-008
    EXPLORER ... login.php profile.php register.php list_roles.php Users
    lib get_url.php README.md reset_session.php safer_echo.php sanitizers.php user_helpers.php
    partials
    public_html assignment2 M6
    project admin assign_roles.php create_role.php list_roles.php > sql helpers.js home.php index.php login.php logout.php profile.php register.php # styles.css
    > OPEN EDITORS
    > OUTLINE
    > TIMELINE
    M Milestone1 O 0 0 0 0 0 4 mins IT202Server jn426 You, 2 weeks ago Ln 5, Col 14 (8 selected) Spaces: 4 UTF-8 CRLF PHP Go Live
    Type here to search 48°F Partly cloudy 12:31 AM 4/2/2024

```

```
try {
    $stmt->execute([":rid" => $role_id]);
    flash("Updated Role", "success");
} catch (PDOException $e) {
    flash(var_export($e->errorInfo, true), "danger");
}

$query = "SELECT id, name, description, is_active FROM Roles";
$params = null;
if (isset($_POST["role"])) {
    $search = $_POST["role"];
    $query .= " WHERE name LIKE :role";
    $params = [":role" => "%$search%"];
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

Jenna@DESKTOP-CS8H4Q9 MINGW64 /c/JW426-IT202-008 (Milestone1)

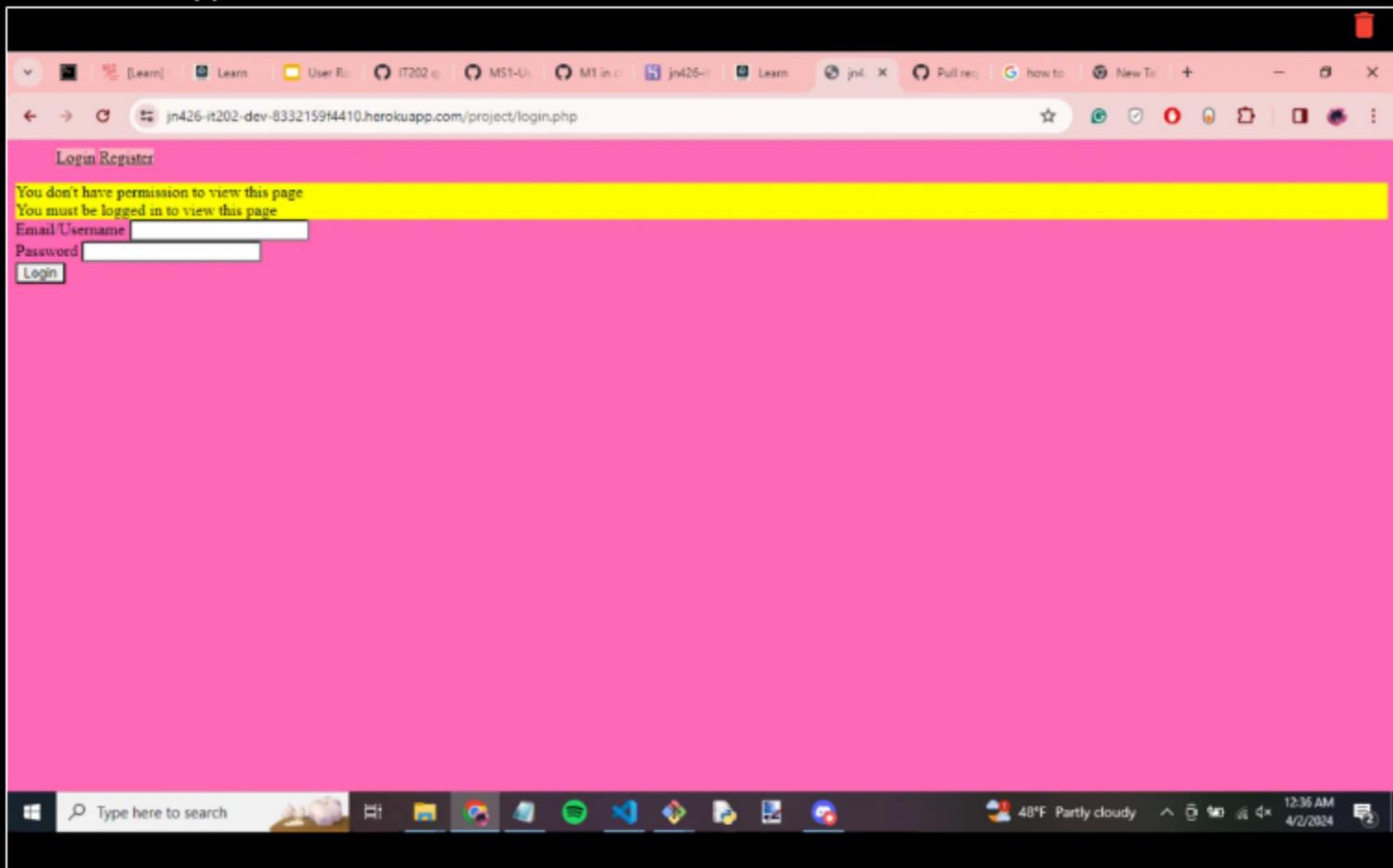
File Milestone1 Open in Explorer Run 0 ▾ 0 4 mins IT202Server Jn426 You, 2 weeks ago Ln 90, Col 3 Spaces 4 UTF-8 CRLF PHP Go Live

Type here to search

48°F Partly cloudy 12:35 AM 4/2/2024

has role in list role

### Checklist Items (0)



user friendly message

### Checklist Items (0)

#### Task #3 - Points: 1

Text: Screenshots of UserRoles and Roles Tables

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
1	1	Completed

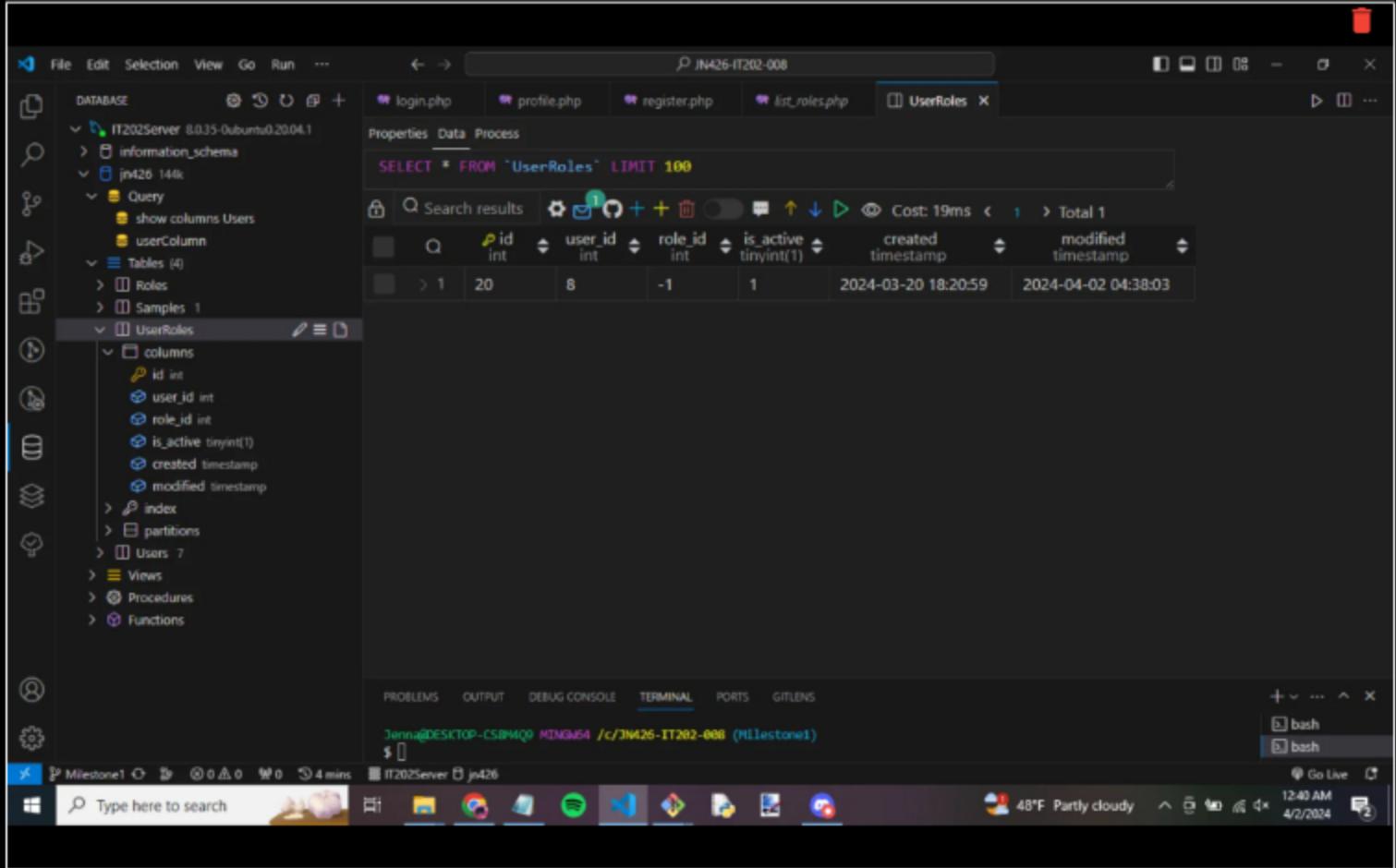
#1	1	At least one valid and enabled User->Role reference (UserRoles table)
#2	1	UserRoles Table should have id, user_id, role_id, is_active, created, and modified columns
#3	1	Roles Table should have id, name, description, is_active, modified, and created columns
#4	1	At least one valid and enabled Role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

### Task Screenshots:

**Gallery Style: Large View**

---

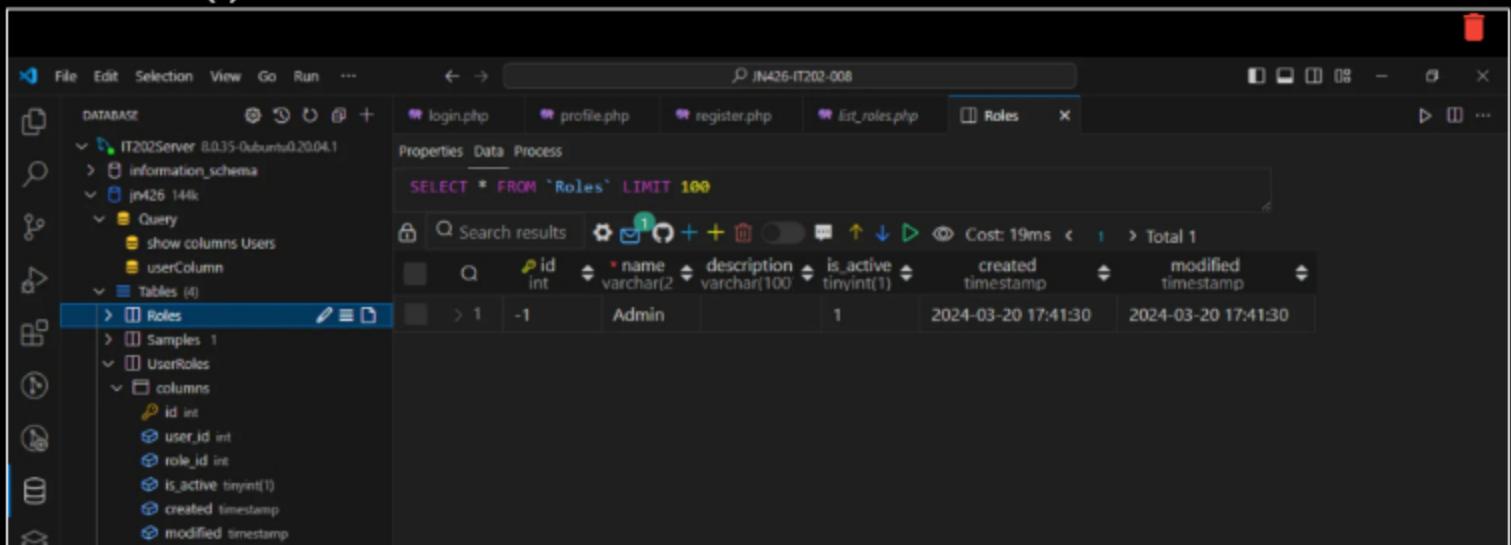
Small      Medium      Large



The screenshot shows a large view of a database interface. On the left is a tree view of the database structure under 'IT202Server'. The 'UserRoles' table is selected. In the center, there's a SQL query editor with the command 'SELECT \* FROM `UserRoles` LIMIT 100'. Below it is a results grid showing one row of data. At the bottom, there's a terminal window showing a bash session on 'DESKTOP-CSBHQ4Q'.

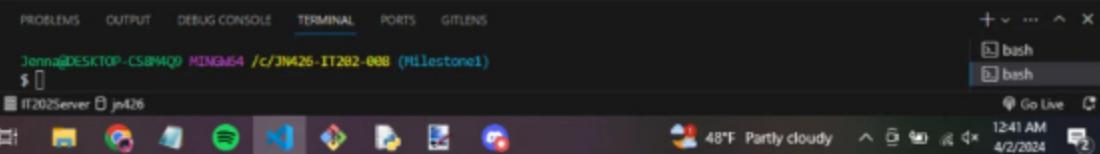
### UserRoles table checklist 1 and 2

#### Checklist Items (0)



The screenshot shows a database interface with the 'Roles' table selected. The results grid displays one row of data: id=1, name='Admin', description='', is\_active=1, created='2024-03-20 17:41:30', modified='2024-03-20 17:41:30'.

> index  
> partitions  
> Users ?  
> Views  
> Procedures  
> Functions



### Roles table 3 & 4

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:

The purpose of the user roles table is to establish a many to many relationship. A role can have many users and a user can have many roles. It allows for certain users to have different permissions. Roles.is\_active is a global on and off switch for the role, and UserRoles.is\_active is a user level active switch for the role.

#### Task #5 - Points: 1

**Text:** Include pull request links related to this feature

#### ● Details:

Should end in /pull/#

URL #1

<https://github.com/jennatnguyen/JN426-IT202-008/pull/39>

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)

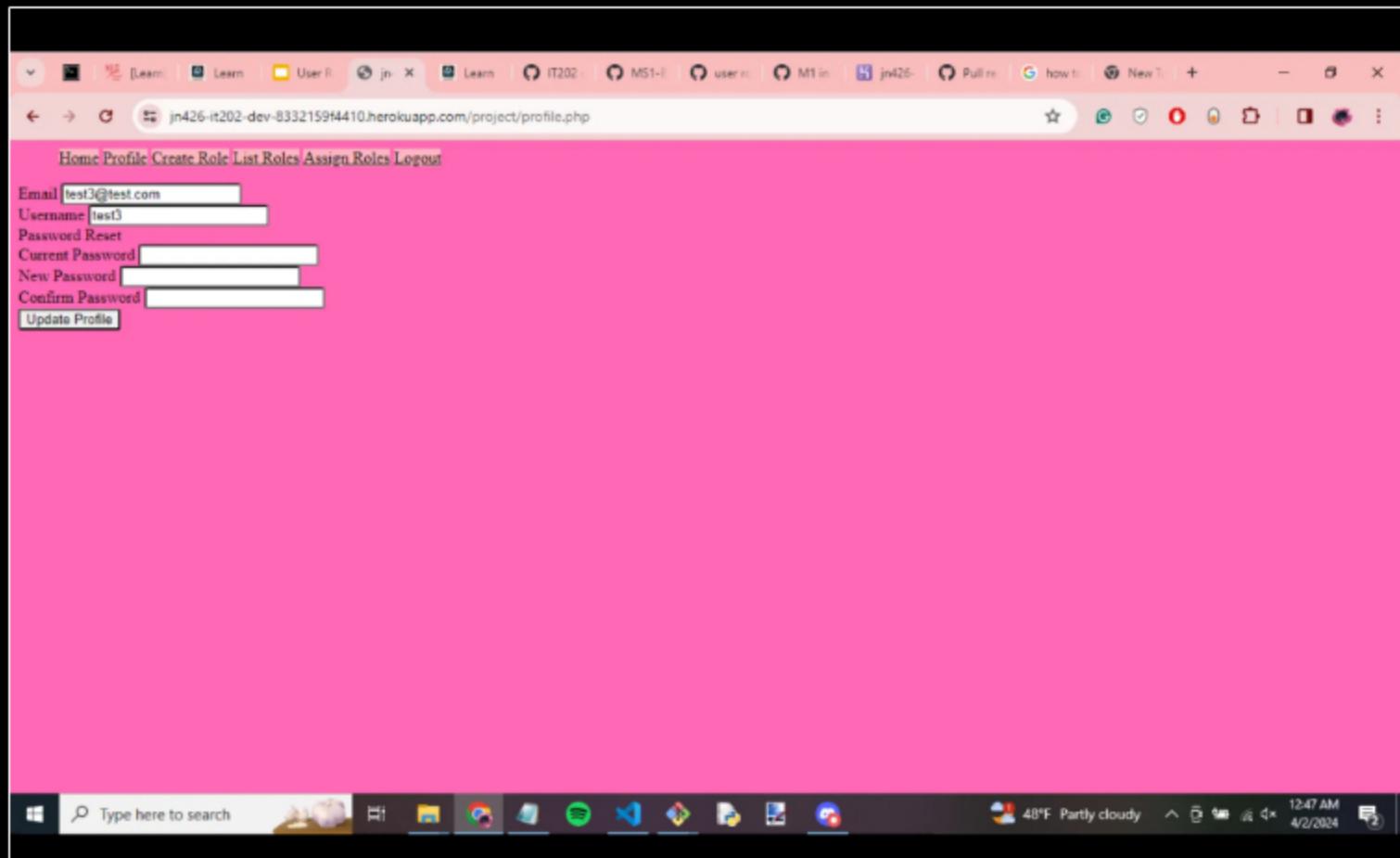
### Task Screenshots:

Gallery Style: Large View

Small

Medium

Large



profile

### 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:

1. Require files and check login status: The code includes necessary files and ensures the user is logged in.
2. Handle form submission: If the form is submitted, it updates the user's email, username, and password if provided, after validating the inputs.
3. Load user data: It retrieves the current email and username of the logged-in user to pre-fill the form fields.
4. Client-side validation: JavaScript validates form inputs before submission to provide instant feedback to the user.
5. Display flash messages: Any feedback or error messages are shown to the user after form submission or client-side validation.

### Task #3 - Points: 1

Text: Edit Profile Website Page

#### Checklist

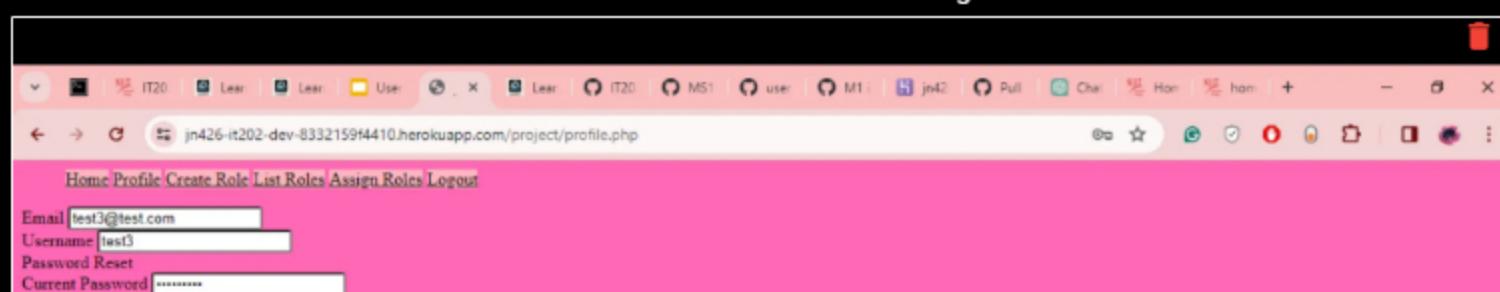
\*The checkboxes are for your own tracking

#	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)
<input checked="" type="checkbox"/> #5	1	Demonstrate the success message of updating password
<input checked="" type="checkbox"/> #6	1	Demonstrate JavaScript user-friendly validation messages (email format, username format, password format, new password matching confirm password)
<input checked="" type="checkbox"/> #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



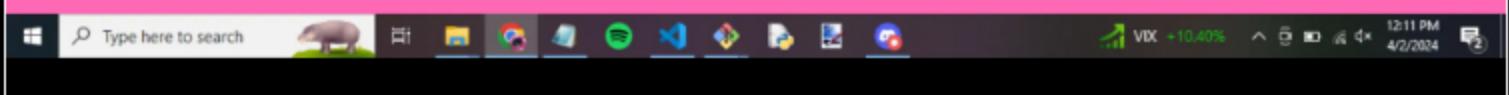
New Password   
Confirm Password



### before username/email change

#### Checklist Items (0)

A screenshot of a web browser window. The address bar shows the URL "jn426-it202-dev-8332159f4410.herokuapp.com/project/profile.php". The page content includes a navigation bar with links like Home, Profile, Create Role, List Roles, Assign Roles, and Logout. Below the navigation is a green success message bar stating "Profile saved" and "Password reset". There are input fields for Email (test3@test.com), Username (test20), and Password (with three masked password fields). A "Update Profile" button is at the bottom.



### Username changed

#### Checklist Items (0)

A screenshot of a web browser window, identical to the one above it, showing the profile edit page. The green success message bar now only displays "Profile saved". The rest of the page content, including the input fields and the "Update Profile" button, remains the same.

Email: test20@test.com  
Username: test20  
Password Reset  
Current Password:   
New Password:   
Confirm Password:



email changed

## Checklist Items (0)

Home Profile Create Role List Roles Assign Roles Logout

Invalid email address  
Invalid username  
New passwords don't match  
Invalid email address  
Invalid username  
New passwords don't match  
Invalid email address  
Invalid username  
Invalid Password  
Email: test20  
Username: Test5  
Password Reset  
Current Password: people123  
New Password:   
Confirm Password:

Console tab in developer tools showing validation logs:

```
inp.removeAttribute(attr);
console.log('Removed ${attr} from element ${inp.name
|| "[No name]"}');

// Change type to text if not already text
if (!["text", "submit", "reset"].includes(inp.type)) {
    inp.type = "text";
    console.log('Changed type to text for element ${inp.name
|| "[No name]"}');
}

}
}

alert("HTML Validation has been disabled until page reload");
})());
Changed type to text for element email
Changed type to text for element currentPassword
Changed type to text for element newPassword
Changed type to text for element confirmPassword
< undefined
>
```

Type here to search  12:12 PM 4/2/2024

JS validation messages

## Checklist Items (0)

IT201 Learn Learn User j Learn IT201 MS1 user M1 in jn420 Pull Hom home +

Home Profile Create Role List Roles Assign Roles Logout

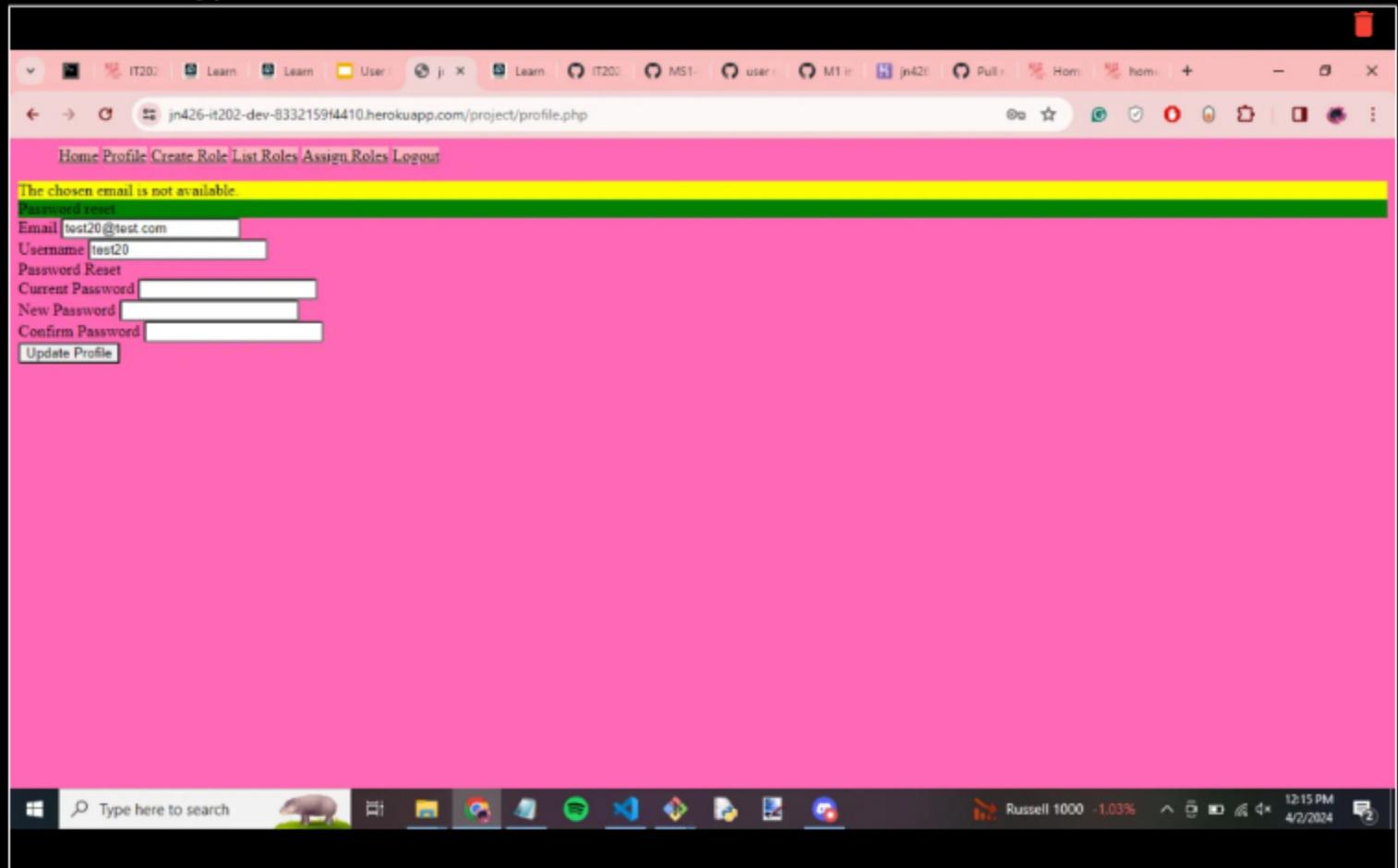
Profile saved  
Password reset  
New passwords don't match

Email: curaroiky@gmail.com  
Username: test5  
Password Reset  
Current Password: .....  
New Password: .....  
Confirm Password: .....



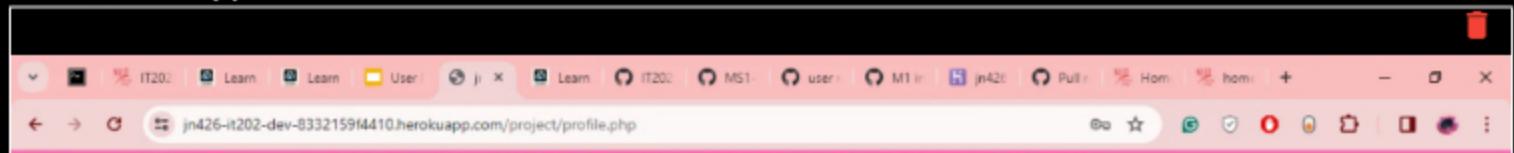
## php validation passwords dont match

### Checklist Items (0)



## php validation chosen email not available

### Checklist Items (0)



The chosen username is not available.

Username	test20
Password Reset	
Current Password	
New Password	
Confirm Password	
<input type="button" value="Update Profile"/>	



php username not available

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

## Response:

## Updating Username/Email:

1. Form Submission: When the user submits the form by clicking the "Update Profile" button, the PHP code first checks if the form was submitted

2. Data Retrieval: It retrieves the entered email and username from the POST data submitted by the form.

## 3. Data Sanitization and Validation:

The email is sanitized and checked for validity using `sanitize_email()` and `is_valid_email()` functions respectively.

The username is checked for validity using `is_valid_username()` function.

## 3. Error Handling:

If the email or username is invalid, it sets a flash message indicating the error.

If the email or username is invalid, it sets a flash message indicating the error.

If there are no errors, it proceeds to update the user's profile.

#### 4. Database Update:

It prepares an SQL statement to update the user's email and username in the database.

It then executes the SQL statement with the provided parameters.

#### 5. Session Update:

After successfully updating the profile in the database, it updates the session data with the new email and username.

### Updating Password:

1. Form Submission: Similar to the previous scenario, the form submission is detected.

#### 2. Data Retrieval:

It retrieves the current password, new password, and confirm password from the form.

#### 3. Validation:

It checks if all password fields are filled.

If they are, it proceeds to further validation.

If not, it sets a flash message to prompt the user to fill all password fields.

#### 4. Error Handling:

If the new passwords don't match, it sets a flash message indicating the mismatch.

If the new password is invalid, it sets a flash message.

If the current password is incorrect, it sets a warning flash message.

#### 5. Database Update:

If all validation passes, it checks the current password's validity by querying the database.

If the current password is correct, it updates the password in the database with the new hashed password.

**Flash Message Display:** Flash messages are displayed to inform the user about the success or failure of the password update.

6. JavaScript Validation: The JavaScript function validate() provides client-side validation for email, username, and password fields, ensuring a smoother user experience.

 Task #5 - Points: 1

 Text: Include pull request links related to this feature

#### Details:

Should end in /pull/#

## URL #1

<https://github.com/jennatnguyen/JN426-IT202-008/pull/33>

Misc (1 pt.)

[^COLLAPSE ^](#)

### Task #1 - Points: 1

Text: Screenshot of wakatime

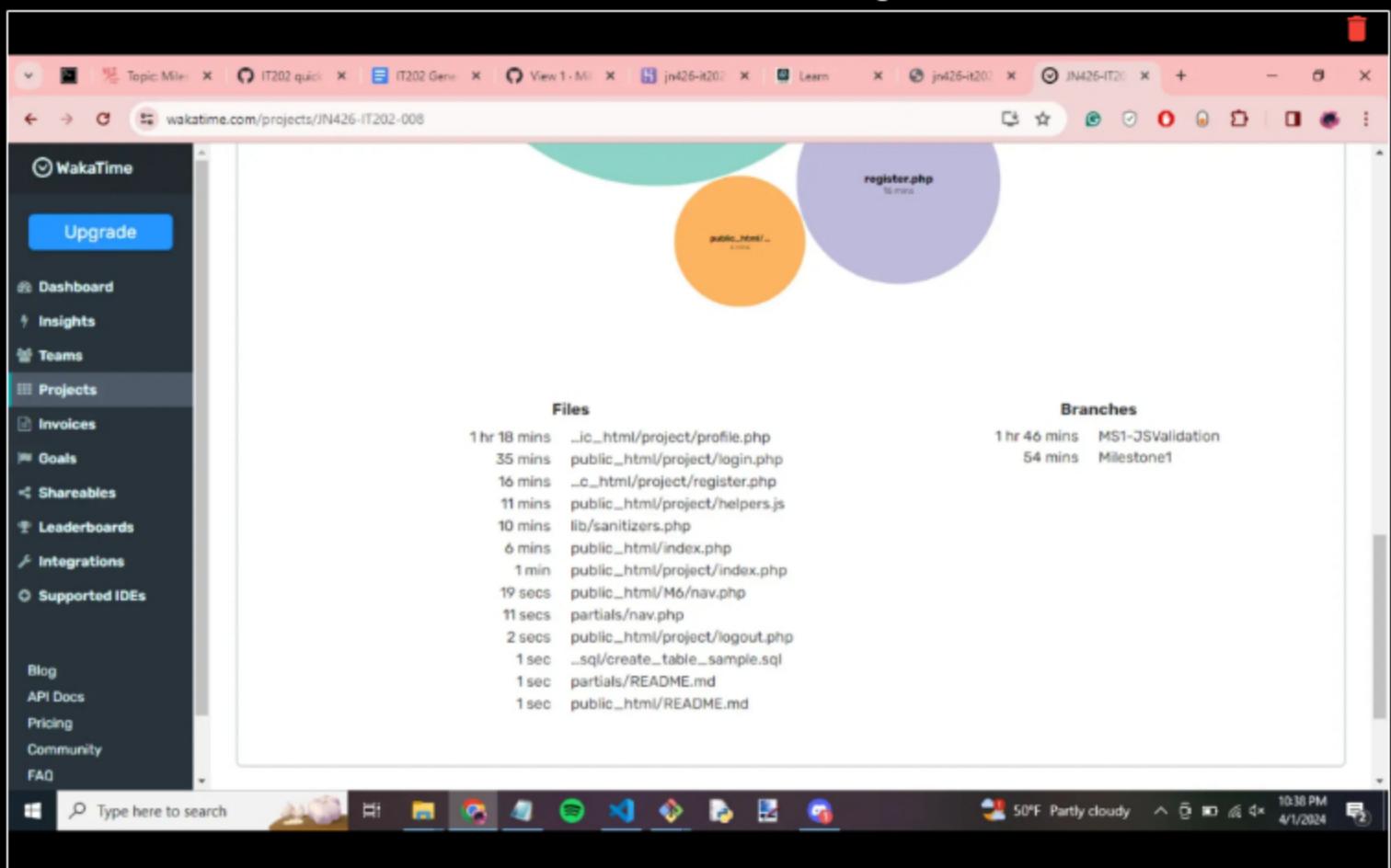
Task Screenshots:

#### Gallery Style: Large View

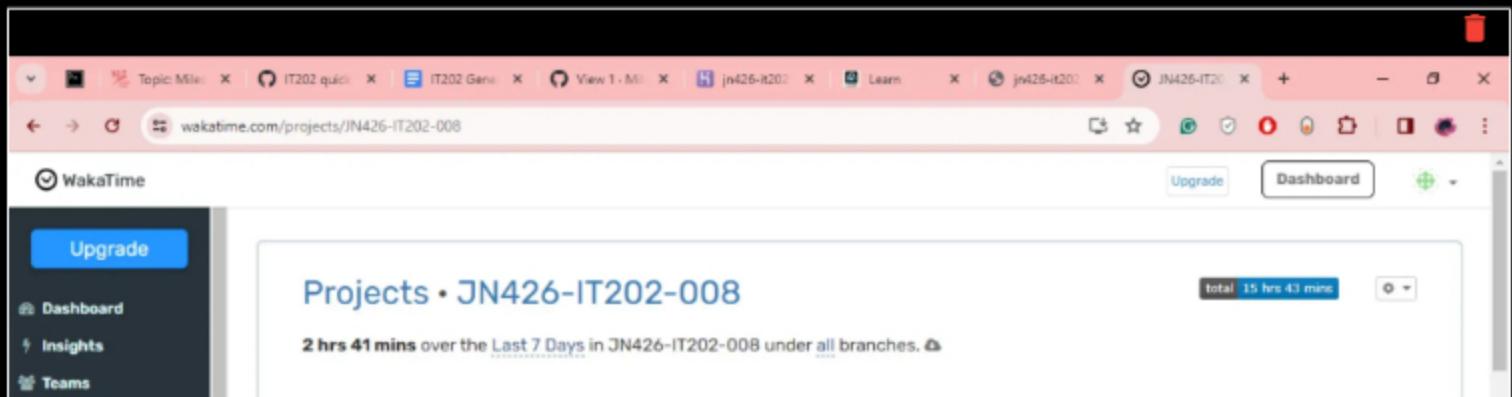
Small

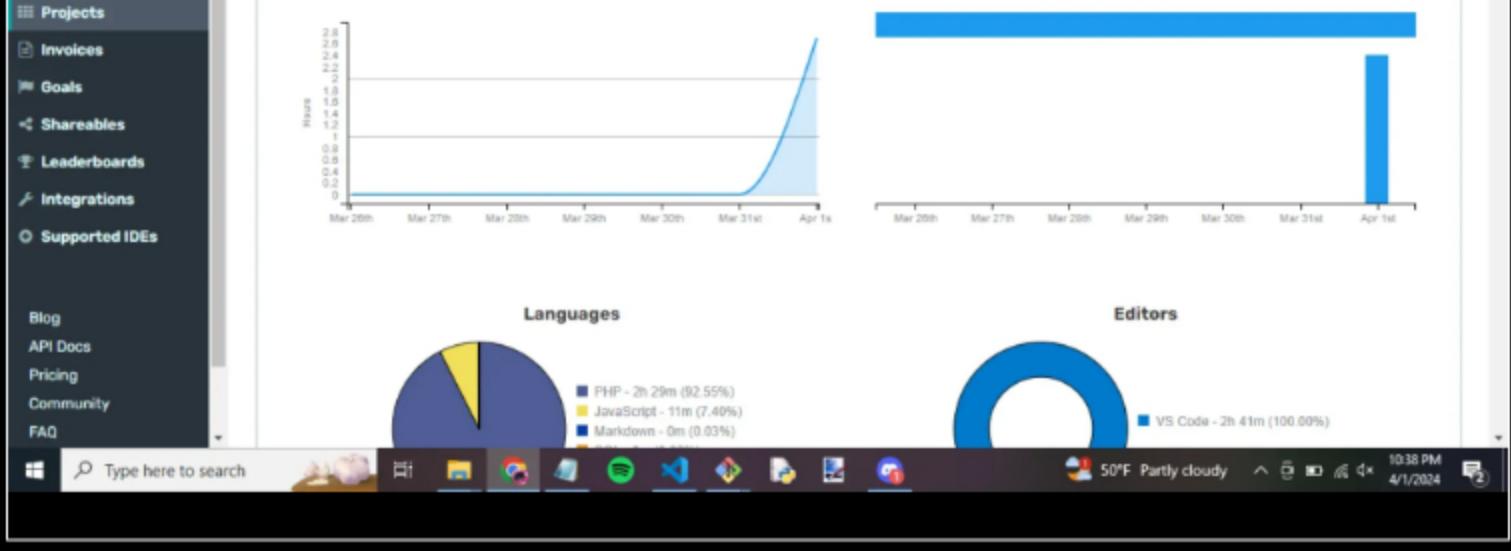
Medium

Large



#### wakatime from this week





wakatime showing total (15hrs)

### Task #2 - Points: 1

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

#### Task Screenshots:

##### Gallery Style: Large View

Small

Medium

Large

Milestones - JN426

Add status update

View 1 + New view

Filter by keyword or by field

Discard Save

Todo	In Progress	Done
0	0	9
This item hasn't been started	This is actively being worked on	This has been completed
+ Add item	+ Add item	+ Add item

JN426-IT202-008 #43  
MS1-User will be able to register a new account

JN426-IT202-008 #46  
MS1-User will be able to login to their account given they enter the correct credentials

JN426-IT202-008 #47  
MS1-User will be able to logout

JN426-IT202-008 #48

Type here to search

50°F Partly cloudy 10:36 PM 4/1/2024

proj board



[^COLLAPSE ^](#)

### Task #3 - Points: 1

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

**URL #1**

<https://github.com/users/jennatnguyen/projects/1>



[^COLLAPSE ^](#)

### Task #4 - Points: 1

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

**URL #1**

<https://jn426-it202-prod-a94514875a20.herokuapp.com/project/login.php>

End of Assignment