

Submission Worksheet

Submission Data

Course: IT202-450-M2025

Assignment: IT202 Milestone 1

Student: Nathanael G. (ng569)

Status: Submitted | **Worksheet Progress:** 100%

Potential Grade: 10.00/10.00 (100.00%)

Received Grade: 0.00/10.00 (0.00%)

Started: 7/7/2025 1:25:56 AM

Updated: 7/7/2025 10:13:57 PM

Grading Link: <https://learn.ethereallab.app/assignment/v3/IT202-450-M2025/it202-milestone-1/grading/ng569>

View Link: <https://learn.ethereallab.app/assignment/v3/IT202-450-M2025/it202-milestone-1/view/ng569>

Instructions

- Overview Link: <https://youtu.be/IDtqdaLwcVg>

1. Refer to Milestone1 of this doc:

<https://docs.google.com/document/d/1XE96a8DQ52Vp49XACBDTNCq0xYDt3kF29cO88EWVwfo/view>

2. Ensure you read all instructions and objectives before starting.

3. Ensure you've gone through each lesson related to this Milestone

4. Switch to the Milestone1 branch

1. git checkout Milestone1 (ensure proper starting branch)
2. git pull origin Milestone1 (ensure history is up to date)

5. Fill out the below worksheet

- Ensure there's a comment with your UCID, date, and brief summary of the snippet in each screenshot
- Ensure proper styling is applied to each page
- Ensure there are no visible technical errors; only user-friendly messages are allowed

6. Once finished, click "Submit and Export"

7. Locally add the generated PDF to a folder of your choosing inside your repository folder and move it to Github

1. git add .
2. git commit -m "adding PDF"
3. git push origin Milestone1
4. On Github merge the pull request from Milestone1 to dev
5. On Github create a pull request from dev to prod and immediately merge. (This will trigger the prod deploy to make the heroku prod links work)

8. Upload the same PDF to Canvas

9. Sync Local

10. git checkout dev

11. git pull origin dev

Section #1: (2 pts.) Feature: User Will Be Able To Register A New Account

≡ Task #1 (0.67 pts.) - Validation

Progress: 100%

Details:

- Show the relevant code snippets for each validation layer
- Show the relevant demo of each validation layer
- Briefly explain how the validation steps work at each layer

≡ Task #1 (0.33 pts.) - HTML Form/Validation

Progress: 100%

Details:

- System should allow the user to correct the error without wiping/clearing the form (for the PHP side this includes sticky-form logic to keep the username/email address entries)

Part 1:

Progress: 100%

Details:

- Show the code related to the register page's form (HTML validation)
- Show examples of each validation message (you may be able to capture this in one or few screenshots)

```
<form onsubmit="return validate(this)" method="POST">
<!-- ong569 //--> 

<div>
    <label for="email">Email</label>
    <input id="email" type="email" name="email" required />
</div>
<div>
    <label for="username">Username</label>
    <input type="text" name="username" required maxlength="30" />
</div>
<div> Your 2 selected tags are registration points! ...
    <label for="pw">Password</label>
    <input type="password" id="pw" name="password" required minlength="8" />
</div>
<div>
    <label for="confirm">Confirm</label>
    <input type="password" name="confirm" required minlength="8" />
</div>
<input type="submit" value="Register" />
</form>
```

Code for Register

The screenshot shows a web browser window with a registration form titled "Register". The "Email" input field is highlighted with a red border and contains the text "w". A validation message is displayed above the field: "Please include an '@' in the email address. 'w' is missing an '@'.". Below the form, there are three buttons: "Login", "Register", and "Logout". The browser's address bar shows the URL "http://ng569-106-60.prod-3273507b1c37herokuapp.com/project/Register.php".

Email Validation

Login Register

Register

Email: j@j.com

Username: j

Password:

Please lengthen this text to 8 characters or more (you are currently using 1 character).

Password Validation

Login Register

Register

Email: j@j.com

Username: j

Password:

Confirm:

Please lengthen this text to 8 characters or more (you are currently using 1 character).

Confirm Validation



Saved: 7/7/2025 1:44:28 AM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e, what you chose, how they solve the requirement)

Your Response:

The html validation was taken from the lectures. Using html type, the email and passwords are checked for length and formatted according to the html types.



Saved: 7/7/2025 1:44:28 AM

☰ Task #2 (0.33 pts.) - JS Validation (validate() function)

Progress: 100%

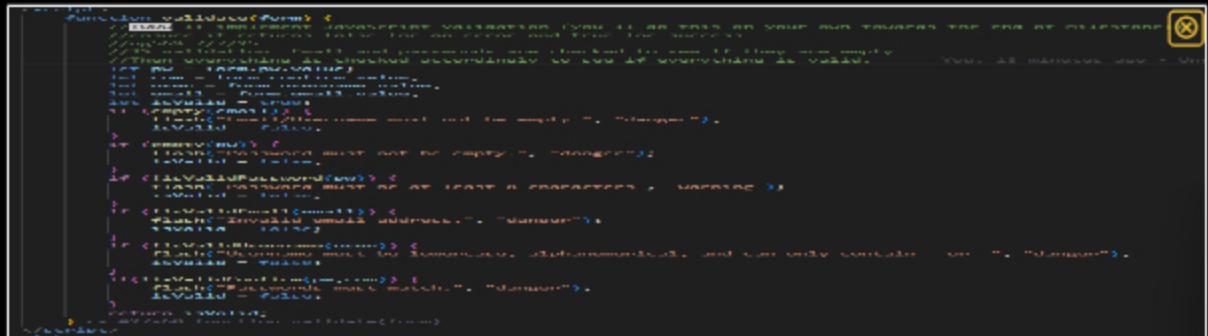
Part 1:

Progress: 100%

Details:

Details:

- Show the code related to the register page's validate() function
- Show examples of each validation message [username format, email format, password format, password doesn't match confirm](you may be able to capture this in one or few screenshots)
- Note: You may need to use dev tools to temporarily apply `novalidate` to the form tag



A screenshot of a browser's developer tools showing the DOM structure of a registration form. The form includes fields for email, password, and password confirmation, each with its own validation message displayed below it.

JS Validation Code



Heroku Validation

```
//ng569 7/7/25
//JS Validation functions
//Use simple regex to check for email and username
//To make sure no outside characters are used
function isValidPassword(pass) {
    return pass?.length >= 8;
}
function isValidEmail(pass) {
    return /^[^@]+@[^\s@]+\.\w+$/i.test(pass);
}
function isValidUsername(pass) {
    return /^[a-z-9-]{3,16}$/i.test(pass);
}
function isMatchConfigPass(newPass) {
    return origPass === newPass;
}
function empty(pass) {
    return pass?.length === 0;
}
```

JS Validation functions



Saved: 7/7/2025 2:09:46 AM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e, what you chose, how they solve the requirement).

Your Response:

I used `{regex}.test` in order to compare the string to a regex expression to prove the validation of the email and username. Then by using `string?.length` I am able to do a one-line return statement and check the length of the string for validation.



Saved: 7/7/2025 2:09:46 AM

☰ Task #3 (0.33 pts.) - PHP Validation (steps before the DB call)

Progress: 100%

☒ Part 1:

Progress: 100%

Details:

- Show the code related to the register page's PHP validation
- Show examples of each validation message [username format, email format, password format, invalid password, password doesn't match confirm, username taken, email taken] (you may be able to capture this in one or few screenshots)
- Note: You may need to use dev tools to temporarily apply `novalidate` to the form tag and temporarily override the validate() function or use the provided helper script in the instructions
- Include the outputs related to username/email already in use

The screenshot shows a registration form with several validation errors displayed below the input fields:

- Email must not be empty.
- Invalid email address.
- Username must be lowercase, alphanumeric, and can only contain - or _.
- Password must not be empty.
- Password must be at least 8 characters long.
- Passwords must match.

Heroku Validation

```
2025-07-07T14:51:25.000Z | app[web.1]: Email must not be empty.
2025-07-07T14:51:25.000Z | app[web.1]: Invalid email address.
2025-07-07T14:51:25.000Z | app[web.1]: Username must be lowercase, alphanumeric, and can only contain - or _.
2025-07-07T14:51:25.000Z | app[web.1]: Password must not be empty.
2025-07-07T14:51:25.000Z | app[web.1]: Password must be at least 8 characters long.
2025-07-07T14:51:25.000Z | app[web.1]: Passwords must match.
```

```

if (!$hasError) {
    // TODO-4: Hash password and store record in DB
    $hashed_password = password_hash($password, PASSWORD_BCRYPT);
    $db = getDB(); // available due to the "require()" of "functions.php"
    // Code for inserting user data into the database
    $stmt = $db->prepare("INSERT INTO users (email, password, username) VALUES (:email, :password, :username)");
    try {
        $stmt->execute([':email' => $email, ':password' => $hashed_password, ':username' => $username]);
        flash("Successfully registered! You can now log in.", "success");
    } catch (PDOException $e) {
        // Handle duplicate email/username
        users_check_duplicate($e);
    } catch (Exception $e) {
        flash("There was an error registering. Please try again.", "danger");
        error_log("Registration Error: " . var_export($e, true)); // log the technical error for debugging
    }
} <- #110-12/ 11 ($hasError)

```

PHP Validation Code 2

```

<?php
//ngSG9_7/7/25
//PHP Functions, using filter_var functions to sanitize and validate traditional emails system
//then uses preg_match function to check for valid username
function sanitize_email($email = "") {
    return filter_var(trim($email), FILTER_SANITIZE_EMAIL);
}
function is_valid_email($email = "") {
    return filter_var(trim($email), FILTER_VALIDATE_EMAIL);
}
function is_valid_username($username) {
    return preg_match('/^[\w\-\.\_]{3,30}$/', $username);
}
function is_valid_password($password) {
    return strlen($password) >= 8;
}
function is_valid_confirm($original, $confirm) {
    // checking not empty to avoid empty equals empty being true
    return !empty($original) && $original == $confirm;
}

```

PHP Functions



Saved: 7/7/2025 2:15:30 AM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e, what you chose, how they solve the requirement)

Your Response:

I used filter_var in order to sanitize and validate traditional emails. Then preg-match is used to ensure the username is correctly validated and put together.



Saved: 7/7/2025 2:15:30 AM

Task #2 (0.67 pts.) - Related URLs

Progress: 100%

Details:

- Include the direct link to this file from the Milestone branch (should end in .php)
- Include the heroku prod link to this file (Just grab the base prod url and

manually enter the path to the file)

- Include the related pull request link for this feature

URL #1 https://github.com/Nate-Gaw/ng569-IT202-450-Milestone1/public_html/project/register.php	  https://github.com/Nate-Gaw 
URL #2 https://ng569-it202-450-prod-3272507b1c51.herokuapp.com/project/register.php	  https://ng569-it202-450-prod-3272507b1c51.herokuapp.com/project/register.php 
URL #3 https://github.com/Nate-Gaw/ng569-IT202-450/pull/25	  https://github.com/Nate-Gaw 
URL #4 https://github.com/Nate-Gaw/ng569-IT202-450/pull/24	  https://github.com/Nate-Gaw 
URL #5 https://github.com/Nate-Gaw/ng569-IT202-450/pull/21	  https://github.com/Nate-Gaw 
URL #6 https://github.com/Nate-Gaw/ng569-IT202-450/pull/20	  https://github.com/Nate-Gaw 
URL #7 https://github.com/Nate-Gaw/ng569-IT202-450/pull/20	  https://github.com/Nate-Gaw 
URL #8 https://github.com/Nate-Gaw/ng569-IT202-450/pull/19	  https://github.com/Nate-Gaw 
URL #9 https://github.com/Nate-Gaw/ng569-IT202-450/pull/18	  https://github.com/Nate-Gaw 
URL #10 https://github.com/Nate-Gaw/ng569-IT202-450/pull/11	  https://github.com/Nate-Gaw 



Saved: 7/7/2025 5:41:19 PM

☒ Task #3 (0.67 pts.) - Database - Users table

Progress: 100%

Details:

- Add a screenshot of the database view from vs code showing a valid registered user (preferably a recent one during evidence capturing)

	<i>id</i>	<i>email</i>	<i>password</i>	<i>created</i>	<i>modified</i>	<i>username</i>
>	1	ng569@njit.edu	\$2y\$12\$FltLBhRYwb86toO7	2025-06-20 02:41:24	2025-06-20 02:41:24	ng569-d399cbe974894875f
>	4	demo@test.com	\$2y\$12\$6deU42II9SjPxXrP.	2025-06-28 01:23:50	2025-07-06 23:17:58	demoman
>	6	test@test.com	\$2y\$12\$unhxngjYwKmFY3q	2025-07-07 06:32:46	2025-07-07 06:32:46	testt

VS Code SQL DB



Saved: 7/7/2025 2:34:29 AM

Section #2: (2 pts.) Feature: User Will Be Able To Login To Their Account

Progress: 100%

≡ Task #1 (0.67 pts.) - Validation

Progress: 100%

Details:

- Show the relevant code snippets for each validation layer
- Show the relevant demo of each validation layer
- Briefly explain how the validation steps work at each layer

≡ Task #1 (0.33 pts.) - HTML Form/Validation

Progress: 100%

Details:

- System should allow the user to correct the error without wiping/clearing the form (for the PHP side this includes sticky-form logic to keep the username/email address entries)

❑ Part 1:

Progress: 100%

Details:

- Show the code related to the login page's form (HTML validation)
- Show examples of each validation message (you may be able to capture this in one or few screenshots)

```
<form onsubmit="return validate(this)" method="POST">
    <!--ng569 7/725
    HTML Form, 2 inputs 1 for email and password and 1 button to submit the form-->
    <div>
        <label for="email">Email or Username</label>
        <input id="email" type="text" name="email" required />
    </div>
    <div> You, 3 weeks ago • Added Nav and Login Files ...
        <label for="pw">Password</label>
        <input type="password" id="pw" name="password" required minlength="8" />
    </div>
    <input type="submit" value="Login" />
</form>

```

HTML Form Code

The screenshot shows a web browser window with a login form. The URL is `ng569-it202-450-prod-3272507bf1c1.herokuapp.com/project/login.php`. The form contains two input fields: "Email or Username" and "Password". Below the "Password" field is a validation message: "Please lengthen this text to 8 characters or more (you are currently using 1 character)".

Heroku HTML Validation

Saved: 7/7/2025 4:52:18 PM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e, what you chose, how they solve the requirement)

Your Response:

There is only validation for password, as even if the user doesn't input an email format, they could be using a username to login. So only password is validated to check for a min length of 8.

Saved: 7/7/2025 4:52:18 PM

☰ Task #2 (0.33 pts.) - JS Validation (validate() function)

Progress: 100%

Part 1:

Progress: 100%

Details:

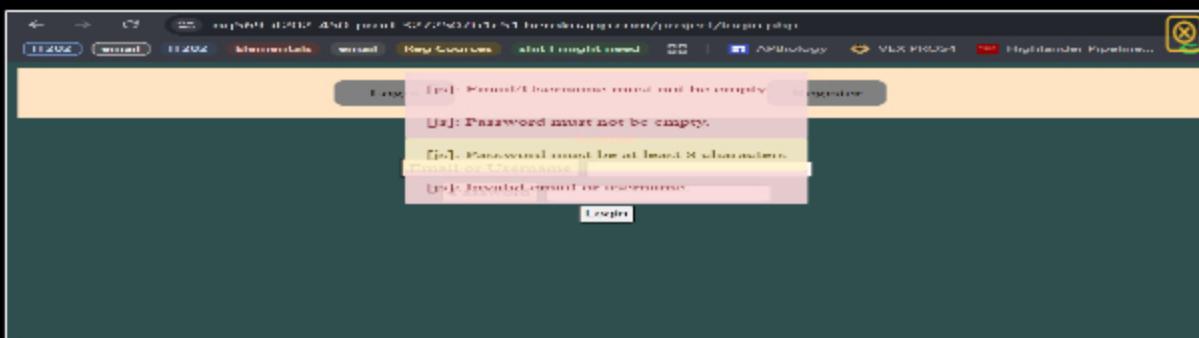
- Show the code related to the login page's validate() function

- Show examples of each validation message [username format, email format, password format] (you may be able to capture this in one or few screenshots)
- Note: You may need to use dev tools to temporarily apply `novalidate` to the form tag

```
EXCERPT:
function validateForm() {
    // This is a simple Javascript validation (you'll do this on your own towards the end of milestone 2)
    // It's just for example purposes. It does form validation and logs to console.
    // User enters the first input to see if it's a valid email or a valid username.
    // If it's a valid email, then email1 = Username1 (otherwise)
    // If it's a valid username, then email1 = Username1 (otherwise)
    // Then email1 = Username1 (otherwise)
    // Finally (password) {
    //   If both Username and Password are empty, then "dangerous"
    //   Otherwise = "safe"
    // }
    // Return isValidity
    // Returns: isValidity, validationMessages (array)
}

// Example:
```

JavaScript Code



Heroku Javascript Validation

```
//ngn9 7/7/25
//JS Validation functions
//Use simple regex to check for email and username
//To make sure no outside characters are used
function isValidPassword(pass) {
    return pass.length >= 8;
}
function isValidEmail(pass) {
    return /^[^@]+@[^\n@]+\.\w{2,}\$/.test(pass);
}
function isValidUsername(pass) {
    return /^[a-z0-9 -]{3,30}\$/.test(pass);
}
function isMatchConfirm(origPass, newPass) {
    return origPass === newPass;
}
function empty(pass) {
    return pass.length === 0;
}
```

Javascript Functions

Saved: 7/7/2025 4:59:53 PM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e, what you chose, how they solve the requirement)

Your Response:

My code checks to see if both inputs are empty. If they are then the code will flag the issue. If not then the password is checked for length <8. After the email/username input is checked to see if the input is a valid email or valid username, the site then outputs the same code if the input is neither an email nor a username. Since password, email and username requirements are all the same, the functions are reused from the registration site.



Saved: 7/7/2025 4:59:53 PM

≡ Task #3 (0.33 pts.) - PHP Validation (steps before the DB call)

Progress: 100%

Part 1:

Progress: 100%

Details:

- Show the code related to the login page's PHP validation
- Show examples of each validation message [username format, email format, password format, invalid password] (you may be able to capture this in one or few screenshots)
- Note: You may need to use dev tools to temporarily apply `novalidate` to the form tag and temporarily override the validate() function or use the provided helper script in the instructions
- Include the outputs related to user doesn't exist and php-side password mismatch



Login PHP Code



Login PHP Code 2

```
        $ambiguity = true; // ambiguous login attempt
    }
    else {
        //echo "Email and password";
        $ambiguity = true; // ambiguous login attempt
    }
    if ($ambiguity) {
        echo("Invalid login attempt. Please check your email and password." . "\n");
    }
}
catch (Exception $e) {
    //echo "There was an error logging in"; // user-friendly message
    flash("There was an error logging in. Please try again later.", "danger");
    error_log("Login Errors: " . var_export($e, true)); // log the technical errors for debugging
}
else {
    #102-140 try
}
#108-149 if (!isset($_POST["email"], $_POST["password"]))
#109-150 try
#110-151 if (!empty($email) & !empty($password))
#111-152 if (!is_email($email))
#112-153 if (!is_valid_password($password))
#113-154 if (!is_valid_confirm($original, $confirm))
#114-155 if ($original == $confirm)
#115-156 if ($password == $confirm)
#116-157 if ($password != $confirm)
```

Login PHP Code 2

The screenshot shows a login page with several validation errors displayed in red boxes:

- Email/Username must not be empty.
- Username must be lowercase, alphanumeric, and can only contain _ or -
- Password must not be empty.
- Password must be at least 8 characters long.

Heroku PHP Output

The screenshot shows a login page with a single error message displayed in a red box:

Invalid login attempt. Please check your email and password.

Heroku Login Failed Attempt

```
<?php
//ng569 7/7/25
//PHP Functions. Using filter_var functions to sanitize and validate traditional email syntax
//Then uses preg_match function to check for valid username
function is_sanitize_email($email) {
    return filter_var(trim($email), FILTER_SANITIZE_EMAIL);
}
function is_valid_email($email) {
    return filter_var(trim($email), FILTER_VALIDATE_EMAIL);
}
function is_valid_username($username) {
    return preg_match('/^([a-zA-Z0-9]+([._][a-zA-Z0-9]+){0,4})$/i', $username);
}
function is_valid_password($password) {
    return strlen($password) >= 8;
}
function is_valid_confirm($original, $confirm) {
    // checking not empty to avoid empty inputs empty being false
    return !empty($original) && $original == $confirm
}
```

PHP Functions

Saved: 7/7/2025 5:13:28 PM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e, what you chose, how they solve the requirement)

Your Response:

Since most of the validation is similar to other pages, the functions were reused from previous pages. Then SQL commands were used in order to retrieve the usernames and passwords from the User database, and potential roles from the UserRoles database in order to look up and prepare for potential roles needed inside the landing page. But if nothing fits, then an alert is issued to the user.



Saved: 7/7/2025 5:13:28 PM

☞ Task #2 (0.67 pts.) - Related URLs

Progress: 100%

Details:

- Include the direct link to this file from the Milestone branch (should end in .php)
- Include the heroku prod link to this file (Just grab the base prod url and manually enter the path to the file)
- Include the related pull request link for this feature

URL #1

https://github.com/Nate-Gaw/ng569-IT202-450-Milestone1/public_html/project/login.php



URL

<https://github.com/Nate-Gaw>



URL #2

<https://ng569-it202-450-prod-3272507b1c51.herokuapp.com/project/login.php>



URL

<https://ng569-it202-450-prod-3272507b1c51.herokuapp.com/project/login.php>



URL #3

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/25>



URL

<https://github.com/Nate-Gaw>



URL #4

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/24>



URL

<https://github.com/Nate-Gaw>



URL #5

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/23>



URL

[">https://github.com/Nate-Gaw](https://github.com/Nate-Gaw)



URL #6

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/22>



URL

<https://github.com/Nate-Gaw>



URL #7

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/21>



URL

<https://github.com/Nate-Gaw>



URL #8

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/20>



tht

<https://github.com/Nate-Gaw>



URL #9

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/19>



tht

<https://github.com/Nate-Gaw>



URL #10

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/18>



tht

<https://github.com/Nate-Gaw>



URL #11

<https://github.com/Nate-Gaw/ng569-IT202-450/pull/16>



tht

<https://github.com/Nate-Gaw>



Saved: 7/7/2025 5:29:13 PM

▣ Task #3 (0.67 pts.) - User Session Evidence

Progress: 100%

Details:

- From the heroku logs (Dashboard -> More button -> view Logs), capture
- It should show the id, username, email, and roles

```
2025-07-07T23:35:29.633307+00:00 heroku[router]: at=info method=GET path="/api/v1/auth/login" host=https://ng569-11202-450-erid-327250793121.herokuapp.com?request_id=ba140f7-10d6-4876-a797-07a5e097-02a5&user_id=5&username=nategaw&password=ng569-11202-450-erid-327250793121.herokuapp.com&remember_me=false&remember_token=<REDACTED> referer=https://herokuapp.com/projects/username%2Fng569-11202-450-erid-327250793121.herokuapp.com/login/auth?username=nategaw&password=ng569-11202-450-erid-327250793121.herokuapp.com&remember_me=false&remember_token=<REDACTED> useragent="Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5241.64 Safari/117.36" browseragent="herokuapp.com/projects/username%2Fng569-11202-450-erid-327250793121.herokuapp.com" hostport="327250793121.herokuapp.com:443" method="POST" path="/api/v1/auth/login" host=ng569-11202-450-erid-327250793121.herokuapp.com request_id=ba140f7-10d6-4876-a797-07a5e097-02a5 session_id=53353333-3333-4333-8333-333333333333" status="200" bytes="206" protocol="HTTP/1.1"
2025-07-07T23:35:29.633307+00:00 app[web.1]: {"id": "5", "username": "nategaw", "email": "nategaw@herokuapp.com", "roles": ["admin"]}
```

Heroku Logs



Saved: 7/7/2025 5:36:33 PM

Section #3: (1 pt.) Feature: User Will Be Able To Logout

Progress: 100%

≡ Task #1 (1 pt.) - Evidence

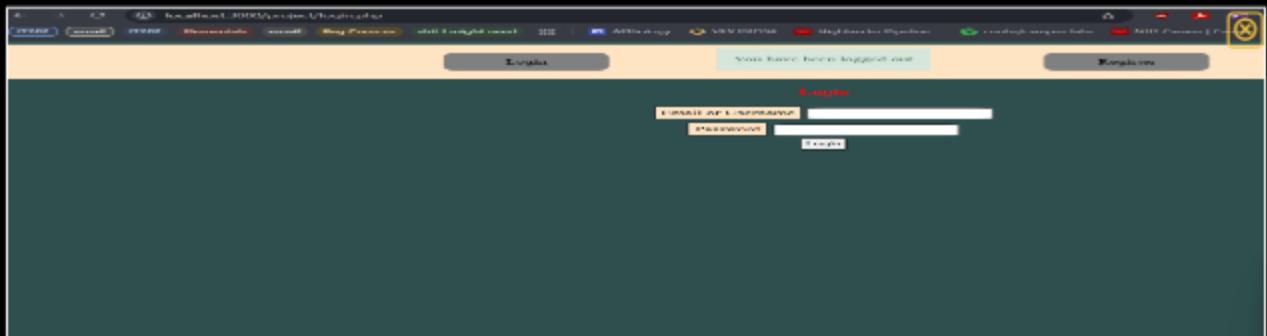
Progress: 100%

Part 1:

Progress: 100%

Details:

- Show the successful logout message
- Show the code snippet of the logout page



Logout Redirect

```
<?php
// require functions.php to pull in flash()
require(__DIR__ . "/../../lib/functions.php");
reset_session(); // clear session data and start a new session
flash("You have been logged out","success");
header("Location: $BASE_PATH/login.php"); // redirect back to login
```

Logout Code

```
<?php
function reset_session() You, 6 days ago - clean up milestone!
{
    // ensure session is started before attempting to reset
    // just because it's not active doesn't mean it doesn't exist
    if(session_status() !== PHP_SESSION_ACTIVE){
        session_start();
        error_log("Session was not active, started a new session.");
    }
    session_unset();
    session_destroy();
    session_start();
} <- #3-13 function reset_session()
```

Reset Session Function



Saved: 7/7/2025 2:45:24 AM

Part 2:

Progress: 100%

Details:

- Include the pull request url for this feature

URL #1

<https://github.com/Nate-Gaw/ng569>

IT2024507



URL

<https://github.com/Nate-Gaw/ng569>



Saved: 7/7/2025 2:45:24 AM

Part 3:

Progress: 100%

Details:

- Briefly explain how the logout process works and how the user is officially logged off

Your Response:

The reset_session function calls session functions, which destroys and starts a new session so the other session is terminated. Then the alert is issued and the user is redirected back to the login page.



Saved: 7/7/2025 2:45:24 AM

Section #4: (2 pts.) Feature: Security Rules

Progress: 100%

Task #1 (1 pt.) - Database Tables

Progress: 100%

Details:

- From the vs code mysql tool, show both the Roles and UserRoles tables

The screenshot shows the MySQL Workbench interface with the 'Properties' tab selected. A query window displays the following SQL statement:

```
SELECT * FROM `Roles` LIMIT 100;
```

The results pane shows the following data for the 'Roles' table:

	<input checked="" type="checkbox"/> id	<input checked="" type="checkbox"/> name	<input checked="" type="checkbox"/> description	<input checked="" type="checkbox"/> is_active	<input checked="" type="checkbox"/> created	<input checked="" type="checkbox"/> modified
1	1	Admin		1	2025-07-02 16:53:14	2025-07-02 16:53:14
2	4	test	test	0	2025-07-04 06:39:54	2025-07-04 06:39:54
3	5	demo	test	1	2025-07-04 06:40:01	2025-07-04 06:40:01
4	7	people	people role	1	2025-07-07 21:44:59	2025-07-07 21:44:59

Roles DB

Properties DATA Log FR Monitor

SELECT * FROM "UserRoles" LIMIT 100

	Id	User_Id	Role_Id	Is_Active	Created	Modified
1	1	1	-1	1	2025-07-07 16:56:55	2025-07-07 16:56:55
2	2	4	5	1	2025-07-07 21:46:27	2025-07-07 21:46:27
3	3	4	7	1	2025-07-07 21:46:27	2025-07-07 21:46:27

UserRoles DB

Saved: 7/7/2025 5:47:26 PM

☰ Task #2 (1 pt.) - Demo Security Rules

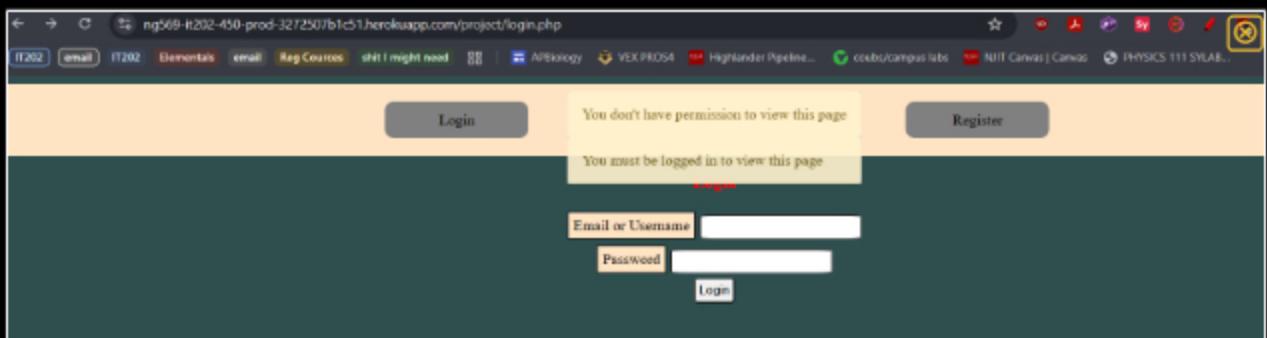
Progress: 100%

Part 1:

Progress: 100%

Details:

- Show the message related to needing to be logged in (i.e., try to manual)
- Show the message related to not having the proper permission/role (i.e.,
- Include a snippet of the login check function
- Include a snippet of the role check function



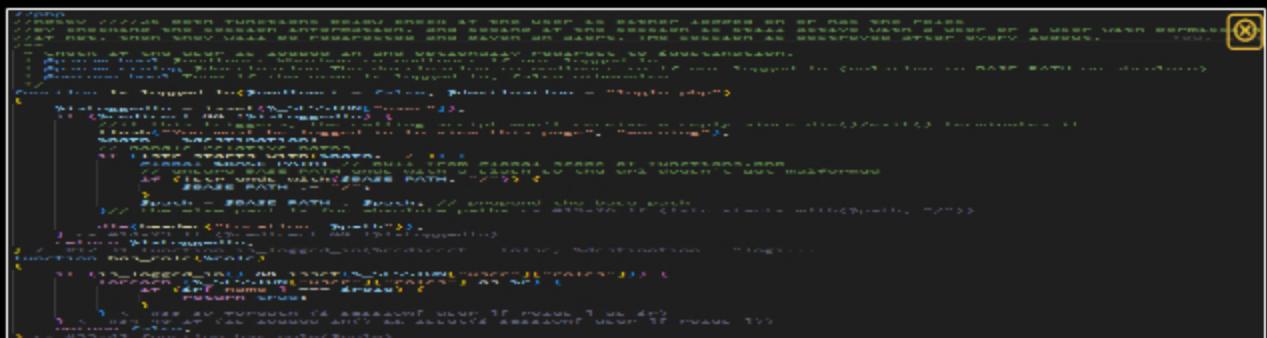
Shows both message to be needing to logged in to access page and not having proper role

```
<?php
//ng569 7/7/25 redirect if user is not logged in
require_once(__DIR__ . "/../../partials/nav.php");
if (!is_logged_in()) {
    die(header("Location: login.php"));
}
?>
```

Check if user is logged in (from Profile.php)

```
//ng569 7/7/25 checks to see if user has admin role and is allowed in
if (!has_role("Admin")) {
    flash("You don't have permission to view this page", "warning");
    die(header("Location: " . get_url("landing.php")));
}
//allow the user to...
```

Check if user has role (from assign_roles.php)



Function snippet (from user_helpers.php)



Saved: 7/7/2025 6:03:52 PM

Part 2:

Progress: 100%

Details:

- Include the pull request url for this feature

URL #1

<https://github.com/Nate-Gaw/ng569-IT202415022>

URL

<https://github.com/Nate-Gaw/ng569-IT202415022>

URL #2

<https://github.com/Nate-Gaw/ng569-IT202415021>

URL

<https://github.com/Nate-Gaw/ng569-IT202415021>

URL #3

<https://github.com/Nate-Gaw/ng569-IT20241508>

URL

<https://github.com/Nate-Gaw/ng569-IT20241508>

Saved: 7/7/2025 6:03:52 PM

Part 3:

Progress: 100%

Details:

- Briefly explain how the login check code works
- Briefly explain how the role check code works

Your Response:

Since user data is stored in `$_SESSION`, the code loops through the array to figure out if the user is still logged in and any roles that the user has. If the code finds nothing, the user is redirected to another page and given an alert. If the code does find something, but it doesn't align with the roles, the same thing happens. After every time logout is clicked, the session is destroyed and erased, so previous information will not get mixed up



Saved: 7/7/2025 6:03:52 PM

Section #5: (2 pts.) Feature: User Profile

Progress: 100%

≡ Task #1 (1 pt.) - Validation

Progress: 100%

Details:

- Show the relevant code snippets for each validation layer
- Show the relevant demo of each validation layer
- Briefly explain how the validation steps work at each layer

≡ Task #1 (0.33 pts.) - HTML Form/Validation

Progress: 100%

Details:

- System should allow the user to correct the error without wiping/clearing the form (for the PHP side this includes sticky-form logic to keep the username/email address entries)

❑ Part 1:

Progress: 100%

Details:

- Show the code related to the profile page's form (HTML validation)
- Show examples of each validation message (you may be able to capture this in one or few screenshots)

```

<?php require 'header.php'; ?>
HTML Validation email and username are required and new and confirm password is required and has min length 8.
The current password is required for check if no minlength given. In case somehow they forgotten or change their password without changing the old one.
<?php
class Form extends \Oneway\Forms {
    public $form = [
        'Email' => [
            'label' => 'Email',
            'type' => 'text',
            'value' => 'lwpip.su@sempf33.de',
            'required' => true,
            'minlength' => 10
        ],
        'Username' => [
            'label' => 'Username',
            'type' => 'text',
            'value' => 'test123',
            'required' => true,
            'minlength' => 40
        ],
        'Current Password' => [
            'label' => 'Current Password',
            'type' => 'password',
            'value' => 'currentpassword',
            'required' => true
        ],
        'New Password' => [
            'label' => 'New Password',
            'type' => 'password',
            'value' => 'newpassword',
            'required' => true,
            'minlength' => 8
        ],
        'Confirm Password' => [
            'label' => 'Confirm Password',
            'type' => 'password',
            'value' => 'confirmpassword',
            'required' => true,
            'minlength' => 8
        ]
    ];
}
</?php>
```

HTML Validation code

Profile

Email: lwpip.su@sempf33.de

Username: test123

[Password Reset](#)

Current Password: *

New Password: *

Confirm Password: *

Please lengthen this text to 8 characters or more (you are currently using 1 characters).

Heroku HTML Validation

Profile

Email: lwpip.su@sempf33.de

Username: test123

[Password Reset](#)

Current Password: *

New Password: *****

Confirm Password: *

Please lengthen this text to 8 characters or more (you are currently using 1 characters).

Heroku HTML Validation 2



Saved: 7/7/2025 10:02:21 PM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e., what you chose, how they solve the requirement)

Your Response:

The code has everything required, so things are not able to be erased or entered as empty. Passwords are then req and have min length > 8 except the current password, in case of mix up. The php validation will check to make sure the password is correct anyway.

☰ Task #2 (0.33 pts.) - JS Validation (validate() function)

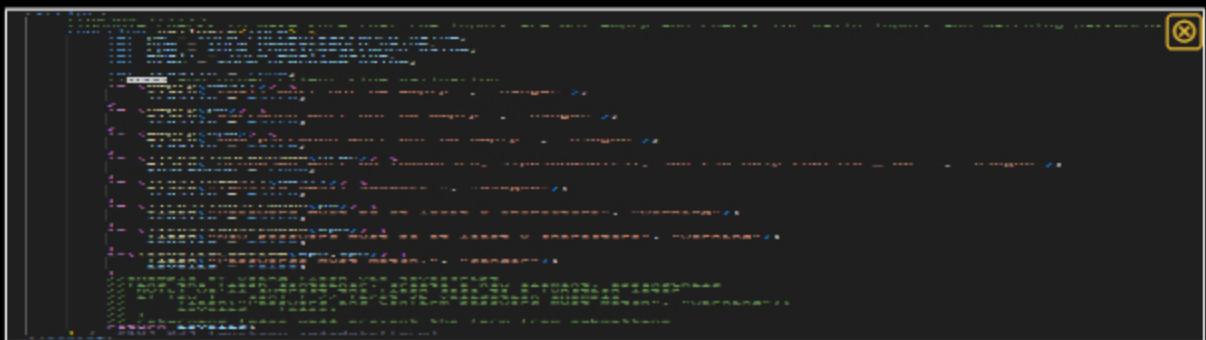
Progress: 100%

▣ Part 1:

Progress: 100%

Details:

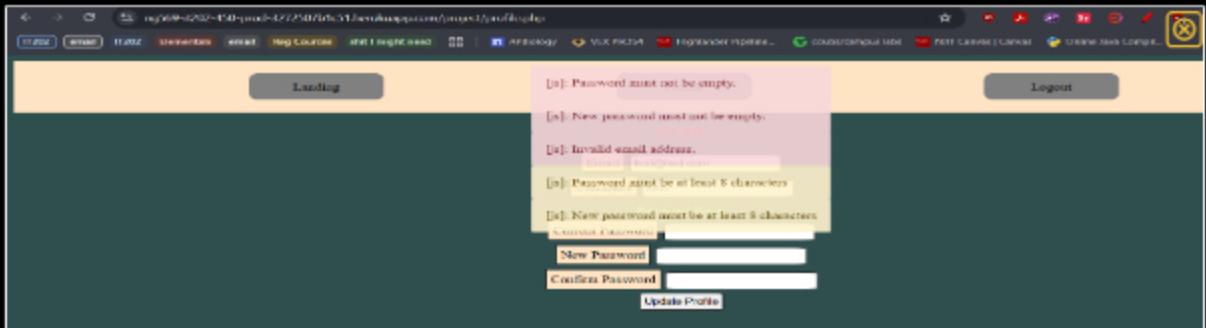
- Show the code related to the profile page's validate() function
- Show examples of each validation message [username format, email format, password format, password doesn't match confirm] (you may be able to capture this in one or few screenshots)
- Note: You may need to use dev tools to temporarily apply `novalidate` to the form tag



Javascript Code

```
//ng569 // // / 25
//JS Validation functions
//Use simple regex to check for email and username
// to make sure no outside characters are used
function isValidPassword(pass) {
    return pass.length >= 8;
}
function isValidEmail(pass) {
    return /^[^@]+@[^\n]+[^\n]+$/ . test (pass);
}
function isValidUsername(pass) {
    return /^[a-zA-Z_]{3,30}$/.test (pass);
}
function isValidConfirm(origPass, newPass) {
    return origPass == newPass;
}
function empty (pass) {
    return pass.length == 0;
}
```

JS Validation Functions



Landing

Logout

[js]: Password must not be empty.
[js]: Invalid email address.
[js]: Password must be at least 8 characters
[js]: New password must be at least 8 characters
[js]: Passwords must match.

New Password:

Confirm Password:

Update Profile

Heroku Javascript Validations 2



Saved: 7/7/2025 9:07:08 PM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e, what you chose, how they solve the requirement)

Your Response:

This code reuses the javascript functions I created, which uses regex and checks length of strings to make sure the passwords, email and username format are valid.



Saved: 7/7/2025 9:07:08 PM

☰ Task #3 (0.33 pts.) - PHP Validation (steps before the DB call)

Progress: 100%

Part 1:

Progress: 100%

Details:

- Show the code related to the profile page's PHP validation
- Show examples of each validation message [username format, email format, password format, invalid password, password doesn't match confirm, username taken, email taken] (you may be able to capture this in one or few screenshots)
- Note: You may need to use dev tools to temporarily apply `novalidate` to the form tag and temporarily override the validate() function or use the provided helper script in the instructions



```
<?php  
// This file contains validation functions for various fields.  
// It uses filter_var() function to validate email, phone number, and URL.  
  
function validate_email($email) {  
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {  
        return false;  
    }  
    return true;  
}  
  
function validate_phone($phone) {  
    if (!filter_var($phone, FILTER_VALIDATE_PHONE_NUMBER)) {  
        return false;  
    }  
    return true;  
}  
  
function validate_url($url) {  
    if (!filter_var($url, FILTER_VALIDATE_URL)) {  
        return false;  
    }  
    return true;  
}
```

PHP Validation Code

```
<?php  
// This file contains validation functions for various fields.  
// It uses filter_var() function to validate email, phone number, and URL.  
  
function validate_email($email) {  
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {  
        return false;  
    }  
    return true;  
}  
  
function validate_phone($phone) {  
    if (!filter_var($phone, FILTER_VALIDATE_PHONE_NUMBER)) {  
        return false;  
    }  
    return true;  
}  
  
function validate_url($url) {  
    if (!filter_var($url, FILTER_VALIDATE_URL)) {  
        return false;  
    }  
    return true;  
}
```

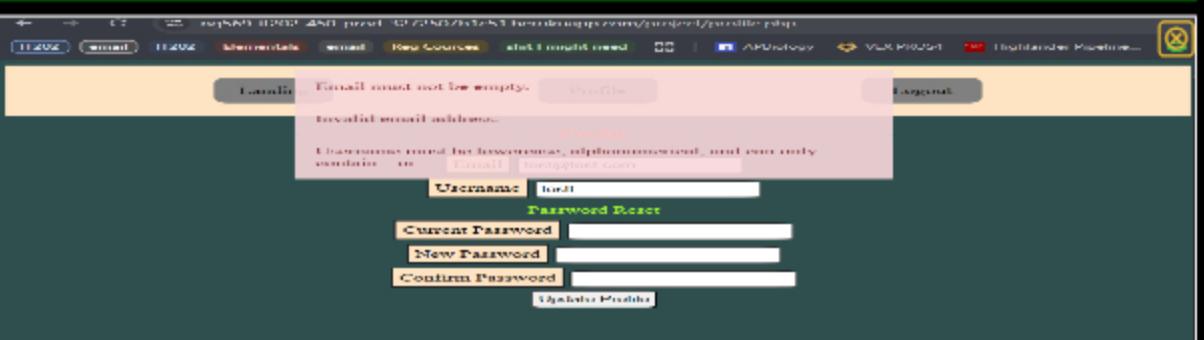
PHP Validation Code 2

```
<?php  
// This file contains validation functions for various fields.  
// It uses filter_var() function to validate email, phone number, and URL.  
  
function validate_email($email) {  
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {  
        return false;  
    }  
    return true;  
}  
  
function validate_phone($phone) {  
    if (!filter_var($phone, FILTER_VALIDATE_PHONE_NUMBER)) {  
        return false;  
    }  
    return true;  
}  
  
function validate_url($url) {  
    if (!filter_var($url, FILTER_VALIDATE_URL)) {  
        return false;  
    }  
    return true;  
}
```

PHP Validation Code 3

```
<?php  
//ng569 2/2/25  
//PHP Functions: using filter_var() functions to sanitize and validate conditional email syntax  
//Then uses preg_match() function to check for valid username  
function sanitize_email($email = "") {  
    return filter_var(trim($email), FILTER_SANITIZE_EMAIL);  
}  
function is_valid_email($email = "") {  
    return filter_var(trim($email), FILTER_VALIDATE_EMAIL);  
}  
function is_valid_username($username)  
{  
    return preg_match('/^[\w-]{3,15}(\.\w{2,3})?$/i', $username);  
}  
function is_valid_password($password)  
{  
    return strlen($password) >= 8;  
}  
function is_valid_confirm($original, $confirm)  
{  
    // checking not empty to avoid empty equals empty being true  
    return !empty($original) && $original === $confirm;  
}
```

PHP Validation Functions



The screenshot shows a web browser window with a login form. The URL is "https://ng569-2/2/25.herokuapp.com/login/password". The form has two fields: "Email" and "Password". The "Email" field is highlighted in red, indicating an error. A tooltip above the field says "Email must not be empty". Another tooltip below it says "Invalid email address...". The "Password" field is also highlighted in red, with a tooltip saying "Email must be lowercase, alphanumeric, and exactly 8 characters long". Below the form, there is a message: "Email must be lowercase, alphanumeric, and exactly 8 characters long".

Landing Now passwords don't match Logout

Profile

Email: test@test.com
Username: test

Password Reset

Current Password:
New Password:
Confirm Password:

Update Profile

Heroku PHP 2

Landing Current password is invalid Logout

Profile

Email: test@test.com
Username: test

Password Reset

Current Password:
New Password:
Confirm Password:

Update Profile

Heroku PHP 3



Saved: 7/7/2025 9:33:51 PM

Part 2:

Progress: 100%

Details:

- Briefly explain the validation step (i.e. what you chose, how they solve the requirement)

Your Response:

The php validation checks for the same thing JS does, except it actually validates it with the DB. First it checks for the email and username and adjusts them if they are valid. Then the current password is checked with the DB and then the new and confirm are cross-checked, assuming they are in a valid format. Then the new password is saved in the DB.



Saved: 7/7/2025 9:33:51 PM

Task #2 (1 pt.) - Related URLs

Progress: 100%

Details:

- Include the direct link to this file from the Milestone branch (should end in .php)
- Include the heroku prod link to this file (Just grab the base prod url and manually enter the path to the file)
- Include the related pull request link for this feature

URL #1 https://github.com/Nate-Gaw/ng569-IT202-450-Milestone1/public_html/project/profile.php   https://github.com/Nate-Gaw 

URL #2 <https://ng569-it202-450-prod-3272507b1c51.herokuapp.com/project/profile.php>   https://ng569-it202-450-prod-3272507b1c51.herokuapp.com/project/profile.php 

URL #3 <https://github.com/Nate-Gaw/ng569-IT202-450/pull/24>   https://github.com/Nate-Gaw 

URL #4 <https://github.com/Nate-Gaw/ng569-IT202-450/pull/21>   https://github.com/Nate-Gaw 

URL #5 <https://github.com/Nate-Gaw/ng569-IT202-450/pull/20>   https://github.com/Nate-Gaw 

URL #6 <https://github.com/Nate-Gaw/ng569-IT202-450/pull/25>   https://github.com/Nate-Gaw 



Saved: 7/7/2025 9:36:42 PM

Section #6: (1 pt.) Misc

Progress: 100%

☰ Task #1 (0.33 pts.) - Github Details

Progress: 100%

❑ Part 1:

Progress: 100%

Details:

From the Commits tab of the Pull Request screenshot the commit history

Commits

Milestone - All users - All time -

Commits on Jul 7, 2025

- Merge pull request #26 from Nate-Gaw/feat-M51-FinalM51Cleanup  1 minute ago
- Added comments and cleaned up some HTML Validation  1 minute ago

Commits on Jul 6, 2025

- Merge pull request #25 from Nate-Gaw/Fix-M51-JSCleanup  1 minute ago
- cleaned up and completed JS validation  1 minute ago

Commits on Jul 5, 2025

- Merge pull request #24 from Nate-Gaw/Fix-M51-CssCleanup  1 minute ago
- fixed CSS alignment & margins  1 minute ago

Commit History

Commits (4) since last update			
Merge pull request #24 from Nate-Gaw/Fix-MSI-UserLoginEnhancement	(Modified)	1 commit	1 file
● Add user authentication logic	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
Merge pull request #23 from Nate-Gaw/Fix-MSI-UserRole	(Modified)	1 commit	1 file
● Role now reflected in login logic	Comments	1	0
User Role Changes	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
Comments (4) on Jul 1, 2025			
Merge pull request #21 from Nate-Gaw/fixup-mst-styling-issue	(Modified)	1 commit	1 file
● fixed some additional issues	Comments	1	0
clean up codebase	Comments	1	0
● implemented #13 (commented last week)	Comments	1	0
Comments (4) on Jun 20, 2025			
MERGE PULL REQUEST #20 FROM NATE-GAW/1-8BT-MSI-UserProfile	(Modified)	1 commit	1 file
● User Profile and password reset	Comments	1	0
● implemented #13 (commented last week)	Comments	1	0

Commit History 2

Commits (6) since last update			
Merge pull request #19 from Nate-Gaw/Test-MSI-FetchMessages	(Modified)	1 commit	1 file
● Add new authentification service	Comments	1	0
Added Flash warnings	Comments	1	0
● implemented #13 (commented last week)	Comments	1	0
Merge pull request #18 from Nate-Gaw/Test-MSI-HelperFunctions	(Modified)	1 commit	1 file
● New class without test result	Comments	1	0
added new helper functions and conditional build in new.php	Comments	1	0
● implemented #13 (commented last week)	Comments	1	0
Comments (4) on Jun 20, 2025			
Merge pull request #17 from Nate-Gaw/Test-MSI-BasicNav-Login	(Modified)	1 commit	1 file
● New class without test result	Comments	1	0
forgot to add logout.php	Comments	1	0
● implemented #13 (commented 3 weeks ago)	Comments	1	0
Merge pull request #16 from Nate-Gaw/Test-MSI-BasicNav-Login	(Modified)	1 commit	1 file
● New class without test result	Comments	1	0
add basic navigation and login functionality	Comments	1	0
● implemented #13 (commented 3 weeks ago)	Comments	1	0

Commit History 3

Commits (10) since last update			
Added User and Login files	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
Merge pull request #14 from Plate-Gaw/Test-MSI-UserRegistrations	(Modified)	1 commit	1 file
user registrations, save data to database with useremail field	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
Merge pull request #13 from Plate-Gaw/Test-MSI-UserLogins	(Modified)	1 commit	1 file
● More user authentication & more user	Comments	1	0
added user validation	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
added login validation	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
just have them enter their email as the saving identifier	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
update user_email field name	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
mark 2 new	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
register users	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0
Merge pull request #9 from Nate-Gaw/ProblemsSolved	(Modified)	1 commit	1 file
● implemented #13 (commented 4 days ago)	Comments	1	0
configured routes with thoughts	Comments	1	0
● implemented #13 (commented 4 days ago)	Comments	1	0

Commit History 4



Part 2:

Progress: 100%

Details:

Include the link to the Pull Request (should end in `/pull/#`)

URL #1

<https://github.com/Nate-Gaw/ng569-IT20245026>



URL

<https://github.com/Nate-Gaw/ng5>



Saved: 7/7/2025 10:02:58 PM

▣ Task #2 (0.33 pts.) - WakaTime - Activity

Progress: 100%

Details:

- Visit the WakaTime.com Dashboard
- Click **Projects** and find your repository
- Capture the overall time at the top that includes the repository name
- Capture the individual time at the bottom that includes the file time
- Note: The duration isn't relevant for the grade and the visual graphs aren't necessary

Projects: ng569-IT202-450

total 26 hrs 11 mins

9 hrs 34 mins over the Last 7 Days in ng569-IT202-450 under all branches.

Wakatime total

ng569-IT202-450

total 26 hrs 11 mins

Wakatime branch and file

ng569-IT202-450

total 26 hrs 11 mins

≡ Task #3 (0.33 pts.) - Reflection

Progress: 100%

⇒ Task #1 (0.33 pts.) - What did you learn?

Progress: 100%

Details:

Briefly answer the question (at least a few decent sentences)

Your Response:

I learned a lot about web development. I never realized how much actually went to just having a user create an account and login. I always thought it was a simple click type and save. But I learned that there is a lot more running in the background to make sure that your account is safe and secure. I also learned how to program in PHP, javascript and HTML. I had done a bit of everything in my high school years, but nothing to this extent. The same goes for all of my CS classes at NJIT. Like sure, some are hard to understand, but this has been the coolest and most practical thing I have done. Furthermore, I learned about how Sessions and DB works, which I will need to know and understand for my next semester, so it's been really cool learning about all of it and getting a head start is always great.



Saved: 7/7/2025 10:08:45 PM

≡, Task #2 (0.33 pts.) - What was the easiest part of the assignment?

Progress: 100%

Details:

Briefly answer the question (at least a few decent sentences)

Your Response:

The easiest part has to be the javascript. Since it was all based on the PHP validations, it went by pretty fast and smoothly. Of course I still learned how to use regex and some functions in javascript, but since the structure was all there, it was a lot easier than understanding the PHP for sure.



Saved: 7/7/2025 10:09:55 PM

≡, Task #3 (0.33 pts.) - What was the hardest part of the assignment?

Progress: 100%

Details:

Briefly answer the question (at least a few decent sentences)

Your Response:

Hardest part of the assignment has to be understanding the PHP portion. Since the class is online, it's been hard to understand how PHP works at times. I had to

even watch a 3 hr course on SQL to understand how it all came together. But I can say it is very neat how these things work and the things you can do with PHP. But nonetheless, looking at the massive amount of code I have, I don't think I would've been able to write this within the summer by myself and especially considering I don't know much about PHP. Furthermore, this time limit made it kinda hard to sit down and really look into the PHP side of things. A lot of it I can see and understand, Im just not sure how I would've made it from scratch, which is kinda the hard part of any assignment/project. This is a really cool project in my opinion, though and I definitely see myself using this knowledge in the real world (for one of the first times in any class)



Saved: 7/7/2025 10:13:57 PM