

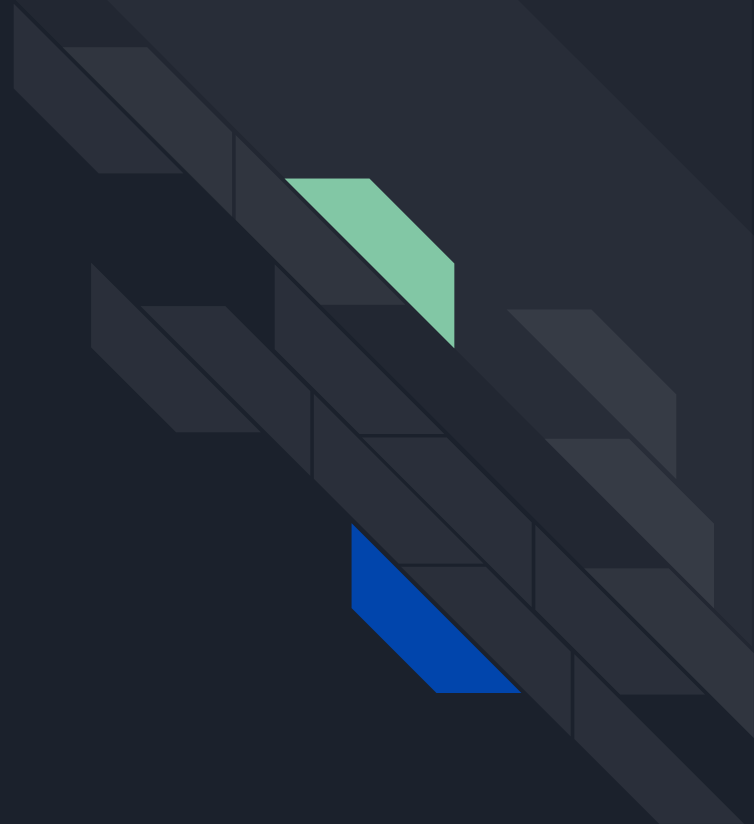


# David Moore's Website

Author: David Moore  
Date: 9/7/2024

// FLATIRON SCHOOL

# LAMP Stack





# LAMP STACK Information

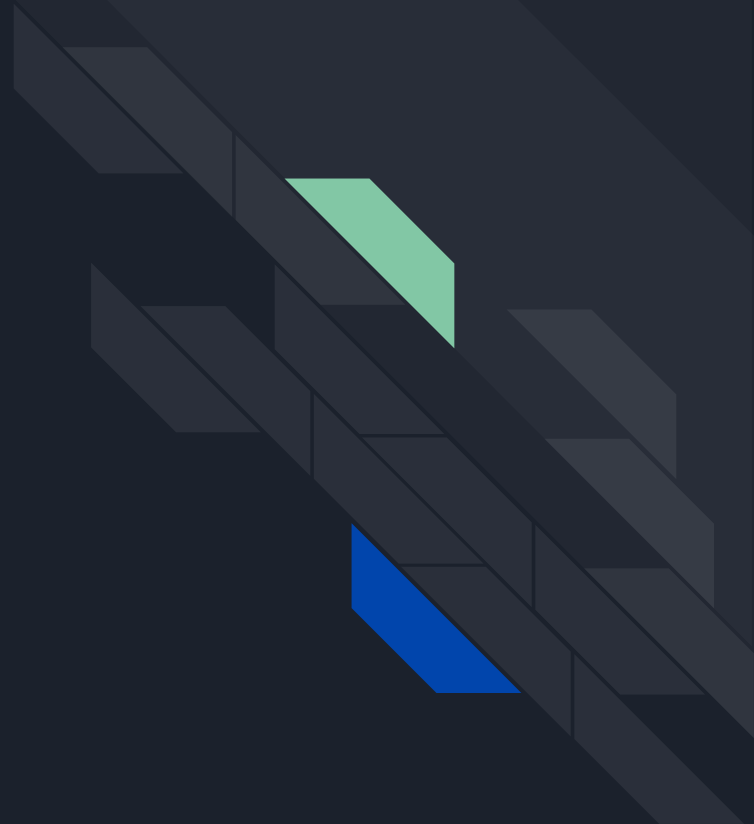
## Description:

The LAMP STACK, powerful open-source platform used to build dynamic and interactive web applications. Linux is the core operating system, Apache serves web content, MySQL or MariaDB handle database management, and PHP for processing server-side logic. This toolkit works seamlessly to create a flexible and scalable environment for web developers, providing all the essentials needed to run modern websites for both startups and large enterprises.

## System Specs:

- **Operating System:** Fedora Linux, 40 (Workstation Edition)
- **Apache Version:** Apache, Version 2.4.62
- **Database:** MariaDB, Version 10.11.8
- **PHP:** PHP, Version 8.3.10

Index.php





# Index.php Information

## Description:

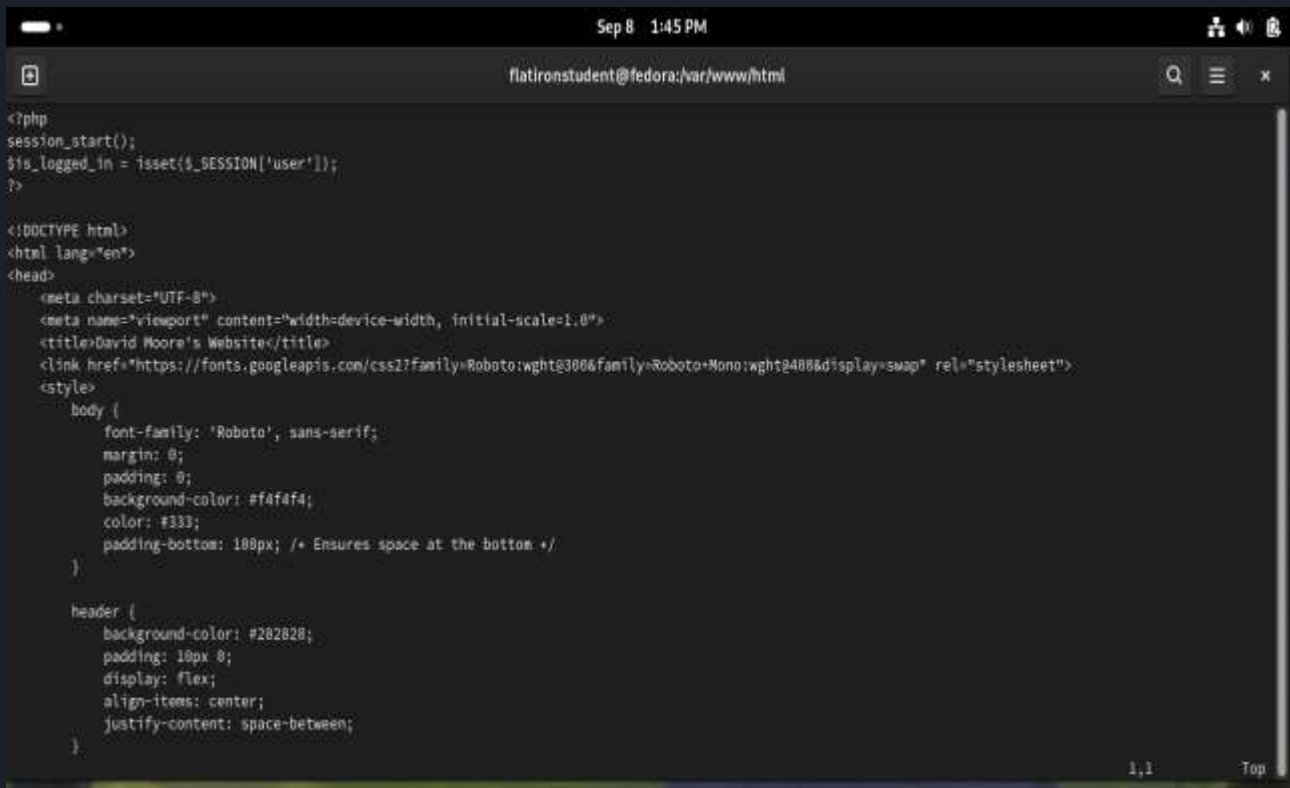
The index page, or homepage, is the primary gateway to the website, offering visitors a 1<sup>st</sup> glance into what the site is all about. Usually containing an introduction, key navigation links, and highlights on important sections to help users find relevant content quickly. The main purpose of an index page is to capture user attention, guide them through the site, and provide a clear sense of the website's identity and structure.

## Key Points about the Page:

- Simple Design
- Welcome Message and "About Me" section
- Navigation Links

# Index.php Code Screenshot

- `Session_start();` This initiates a PHP session and checks if a 'user' session is set, storing the result in `$is_logged_in` to determine whether the user is logged in or not.
- Links to Google Fonts (Roboto and Roboto Mono), which are used for styling the text on the webpage.
- The header is styled with a dark background color (`#282828`), and its content is aligned using flexbox to center items vertically and distribute them horizontally. Padding is added for spacing.



```

<?php
session_start();
$is_logged_in = isset($_SESSION['user']);
?>

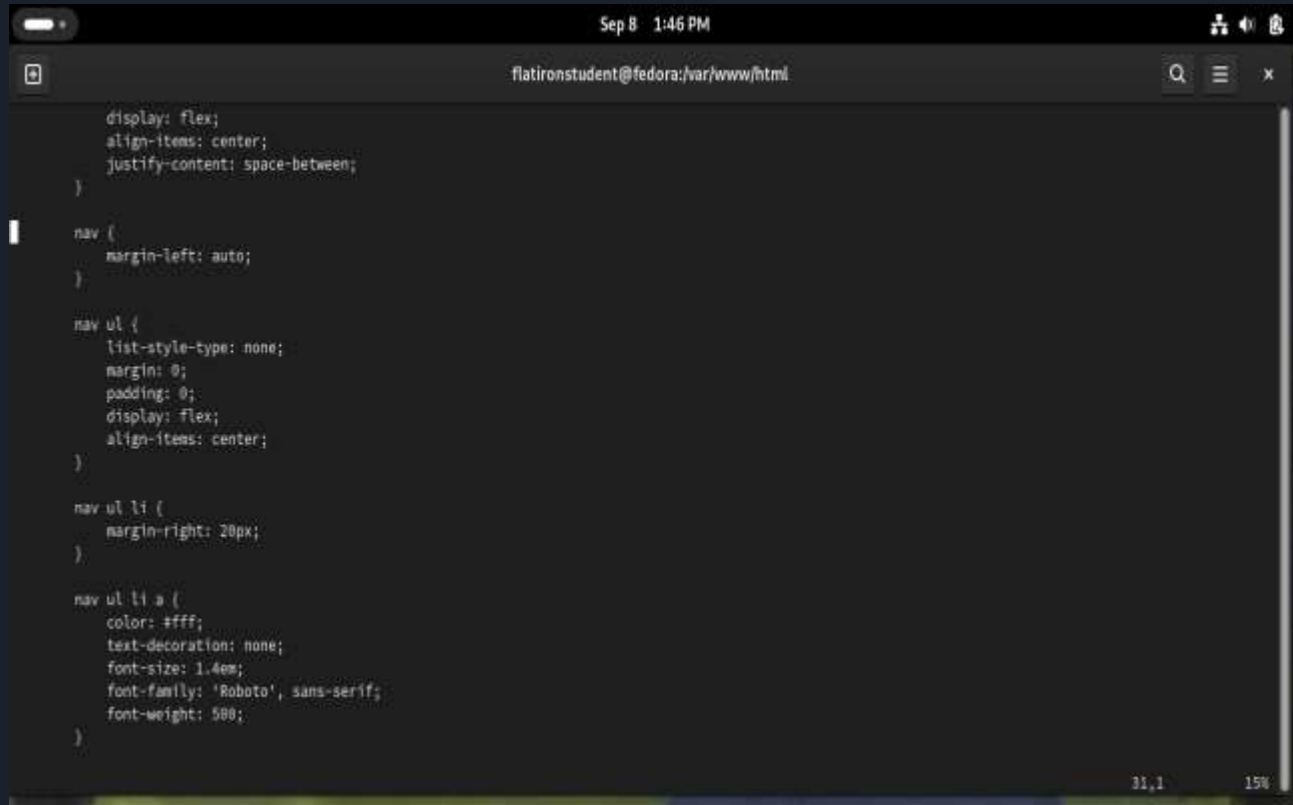
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>David Moore's Website</title>
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300&family=Roboto+Mono:wght@400&display=swap" rel="stylesheet">
    <style>
        body {
            font-family: 'Roboto', sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
            color: #333;
            padding-bottom: 100px; /* Ensures space at the bottom */
        }

        header {
            background-color: #282828;
            padding: 10px 0;
            display: flex;
            align-items: center;
            justify-content: space-between;
        }
    </style>

```

# Index.php Code Screenshot

- nav Element Styling: The navigation (nav) section is set to align to the right (margin-left: auto), pushing the navigation elements to the far right of the header.
- Navigation List Styling (nav ul)
- List Item (li) Styling
- Navigation Link Styling (nav ul li a)

A screenshot of a code editor window with a dark theme. The title bar shows 'Sep 8 1:46 PM' and system icons. The address bar shows 'flatironstudent@fedora:/var/www/html'. The code is CSS for a navigation bar. It includes a flex container for the header, a right-aligned navigation container, a list-style-type: none for the navigation ul, and specific styling for the list items and links (white color, Roboto font, 1.4em size, 500 weight).

```
display: flex;
align-items: center;
justify-content: space-between;
}

nav {
margin-left: auto;
}

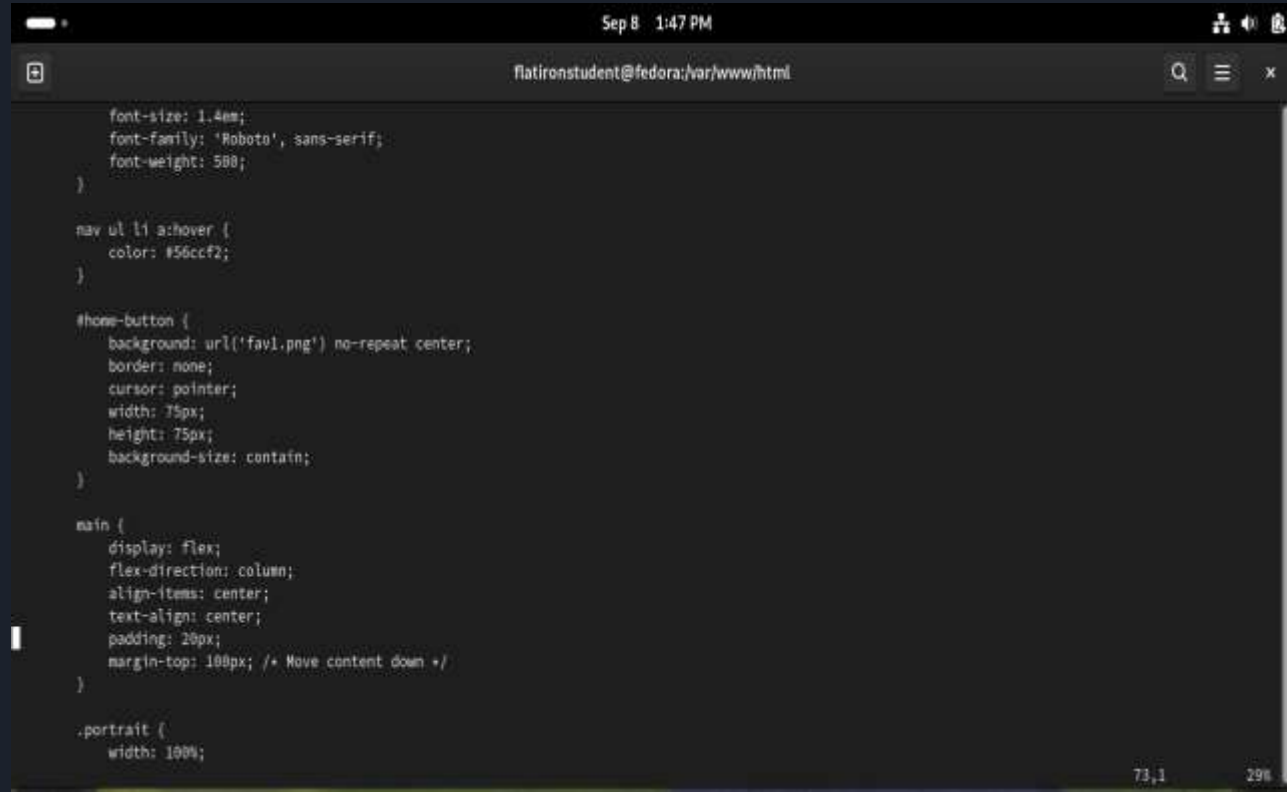
nav ul {
list-style-type: none;
margin: 0;
padding: 0;
display: flex;
align-items: center;
}

nav ul li {
margin-right: 20px;
}

nav ul li a {
color: #fff;
text-decoration: none;
font-size: 1.4em;
font-family: 'Roboto', sans-serif;
font-weight: 500;
}
```

# Index.php Code Screenshot

- Hover Effect for Links (nav ul li a:hover)
- Home Button Styling (#home-button)
- Main Content Styling (main)
- Portrait Image Styling (.portrait):



```
font-size: 1.4em;
font-family: 'Roboto', sans-serif;
font-weight: 500;
}

nav ul li a:hover {
  color: #56ccf2;
}

#home-button {
  background: url('fav1.png') no-repeat center;
  border: none;
  cursor: pointer;
  width: 75px;
  height: 75px;
  background-size: contain;
}

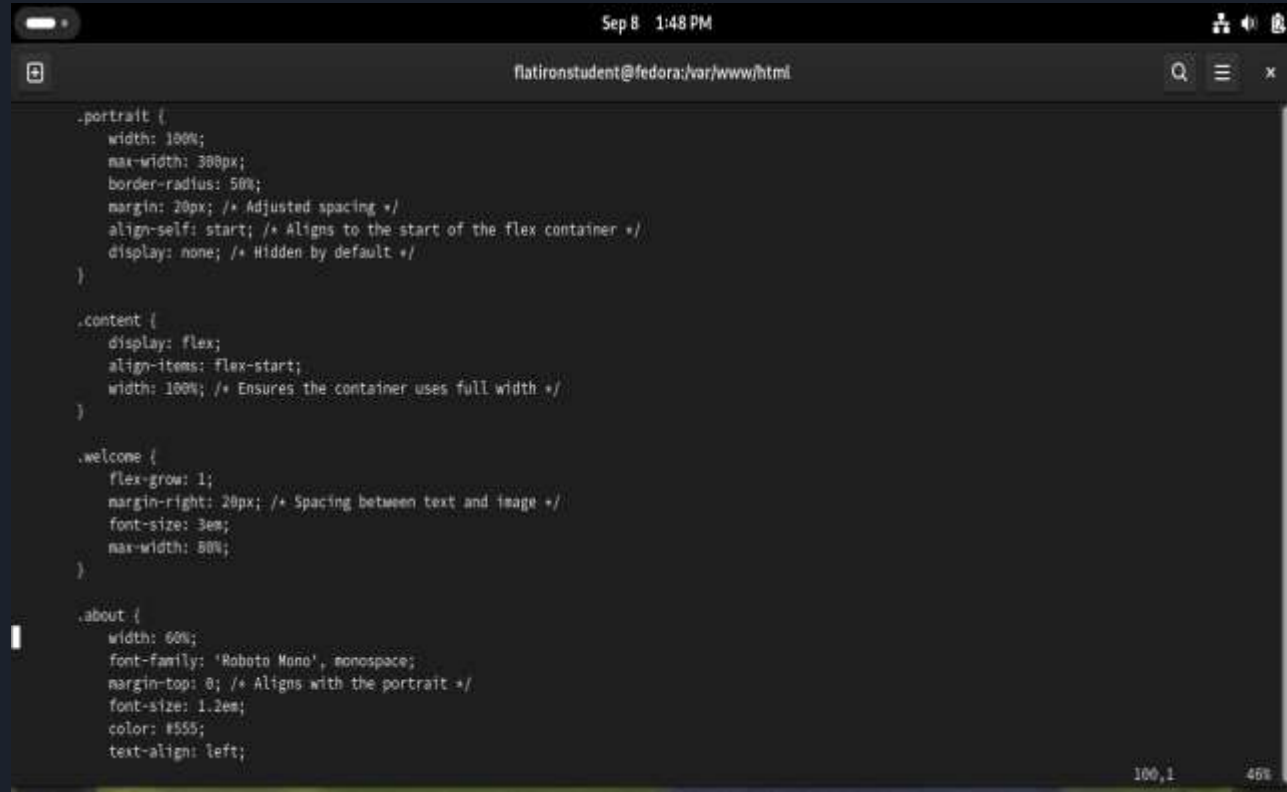
main {
  display: flex;
  flex-direction: column;
  align-items: center;
  text-align: center;
  padding: 20px;
  margin-top: 100px; /* Move content down */
}

.portrait {
  width: 100%;
```



# Index.php Code Screenshot

- Portrait Styling (.portrait): The portrait image is set to a maximum width of 300px, with a circular shape (border-radius: 50%), and includes margins for spacing. It is aligned to the start of its flex container and is hidden by default (display: none).
- Continued class styling for main homepage components.



```

Sep 8 1:48 PM
flatironstudent@fedora:~/var/www/html

.portrait {
  width: 100%;
  max-width: 300px;
  border-radius: 50%;
  margin: 20px; /* Adjusted spacing */
  align-self: start; /* Aligns to the start of the flex container */
  display: none; /* Hidden by default */
}

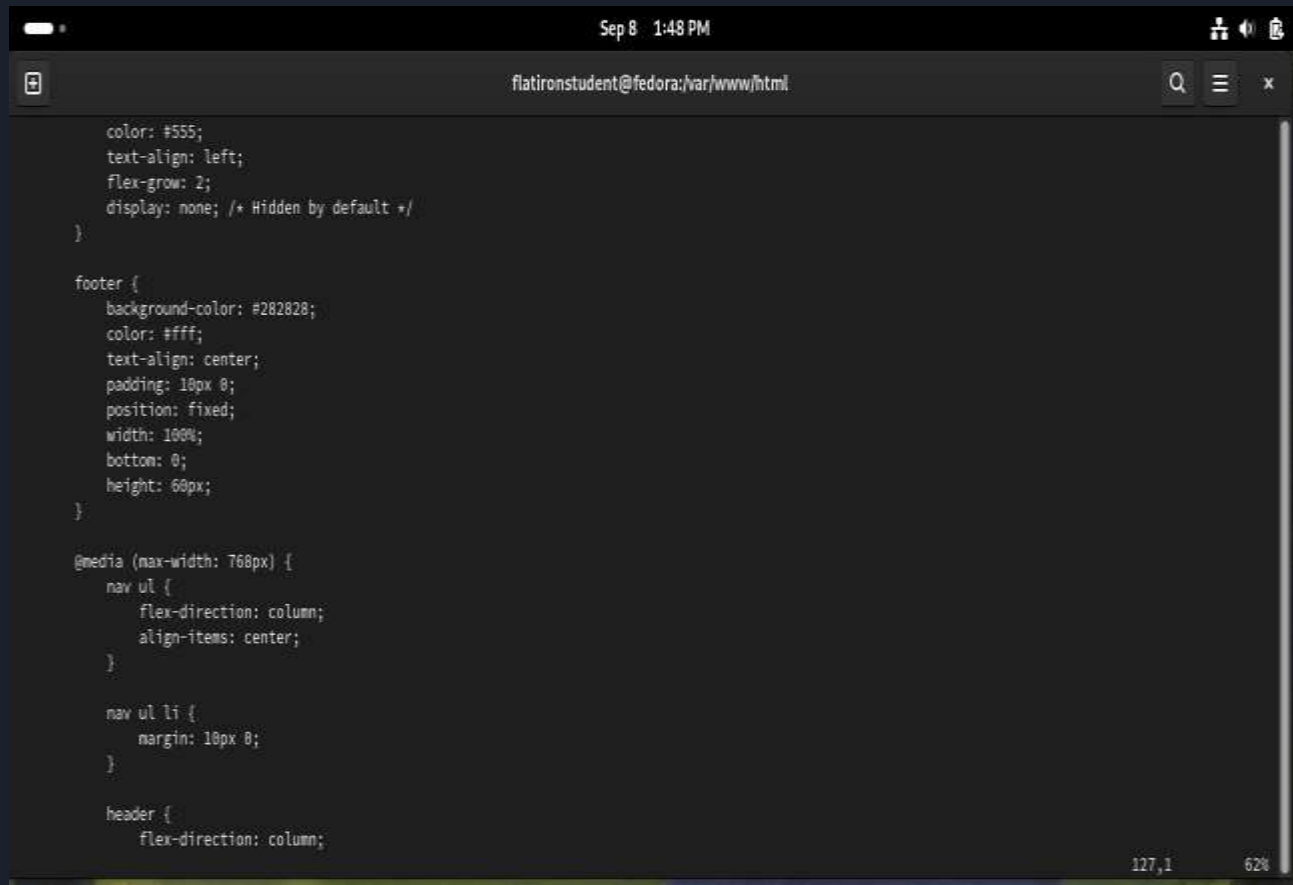
.content {
  display: flex;
  align-items: flex-start;
  width: 100%; /* Ensures the container uses full width */
}

.welcome {
  flex-grow: 1;
  margin-right: 20px; /* Spacing between text and image */
  font-size: 3em;
  max-width: 80%;
}

.about {
  width: 60%;
  font-family: 'Roboto Mono', monospace;
  margin-top: 0; /* Aligns with the portrait */
  font-size: 1.2em;
  color: #555;
  text-align: left;
}
100,1 45%
```

# Index.php Code Screenshot

- Footer Styling
- Media Query for Responsive Design: This media query is activated when the screen width is 768px or smaller (@media (max-width: 768px)), adjusting the layout for smaller devices. It changes the navigation list (nav ul) to a vertical column layout (flex-direction: column) and centers its items.
- Responsive List Item Styling (nav ul li)
- Responsive Header Layout (header)



```
color: #555;
text-align: left;
flex-grow: 2;
display: none; /* Hidden by default */
}

footer {
  background-color: #282828;
  color: #fff;
  text-align: center;
  padding: 10px 0;
  position: fixed;
  width: 100%;
  bottom: 0;
  height: 60px;
}

@media (max-width: 768px) {
  nav ul {
    flex-direction: column;
    align-items: center;
  }

  nav ul li {
    margin: 10px 0;
  }

  header {
    flex-direction: column;
  }
```

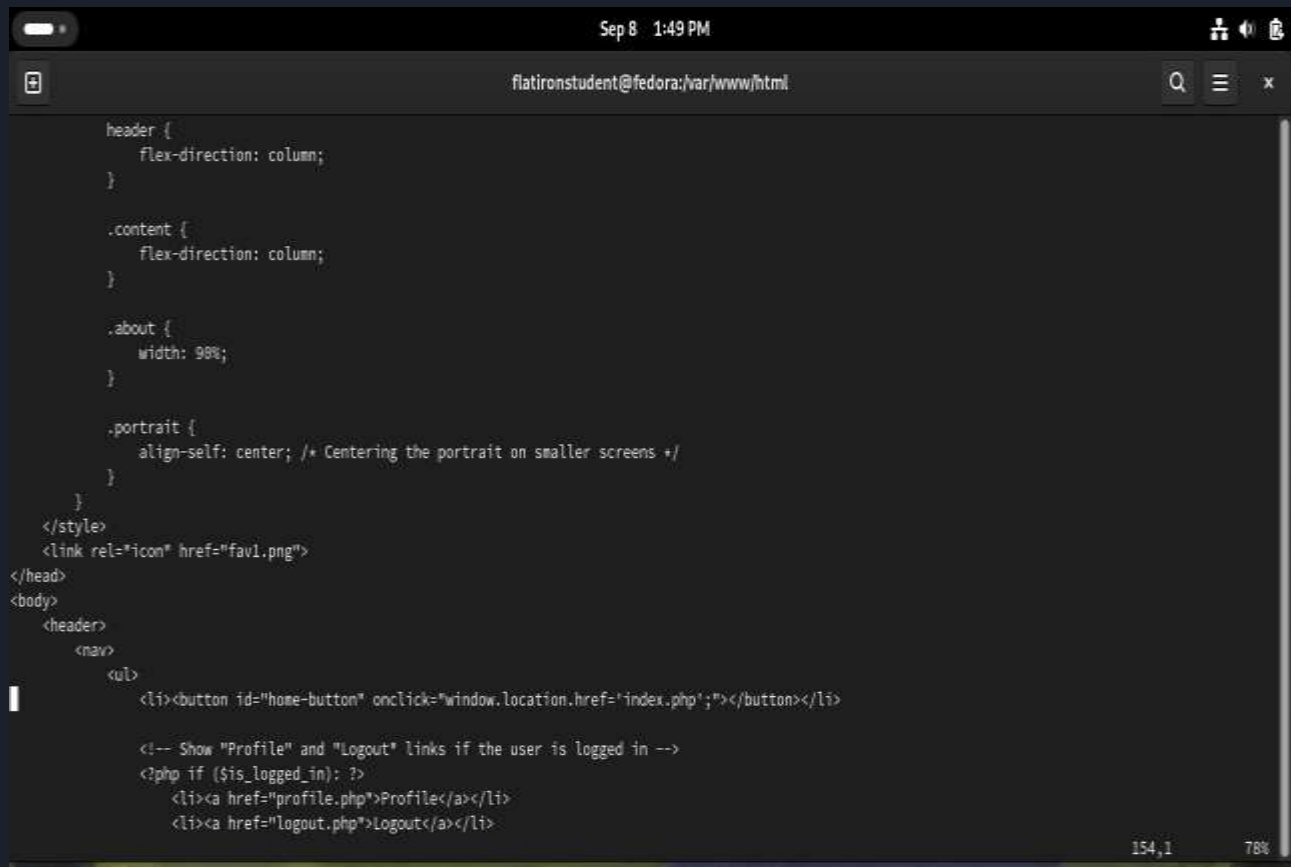
Sep 8 1:48 PM

flatironstudent@fedora:/var/www/html

127,1 62%

# Index.php Code Screenshot

- Styling for smaller screens continued.
- Favicon Link: The <link> tag defines the favicon of the webpage, linking to the image fav1.png to display in the browser tab.
- Home Button (#home-button): The home button is a clickable button that triggers a JavaScript onclick event, redirecting the user to the index.php page when clicked.
- Conditional Profile and Logout Links: PHP conditional statements (<?php if (\$is\_logged\_in): ?>) dynamically generate "Profile" and "Logout" links if the user is logged in, enhancing user experience based on session status.



```
header {
    flex-direction: column;
}

.content {
    flex-direction: column;
}

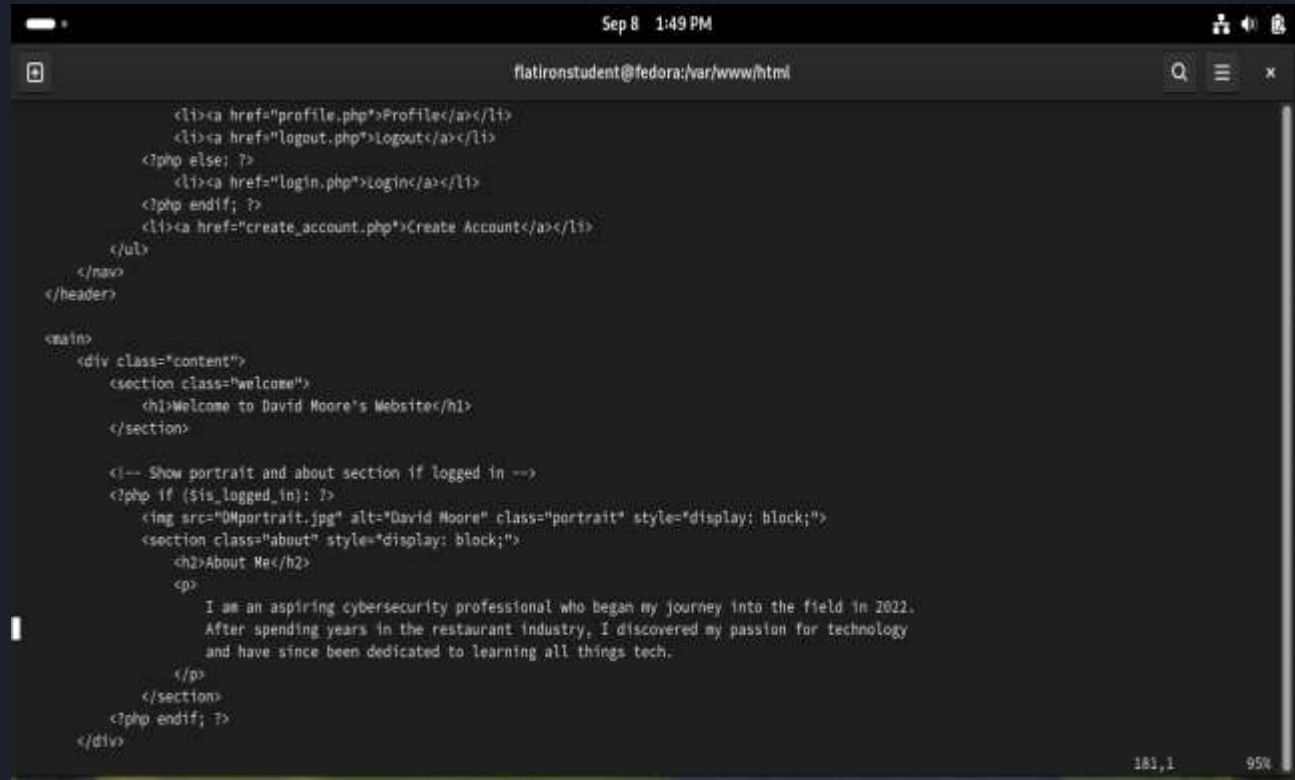
.about {
    width: 98%;
}

.portrait {
    align-self: center; /* Centering the portrait on smaller screens */
}
</style>
<link rel="icon" href="fav1.png">
</head>
<body>
    <header>
        <nav>
            <ul>
                <li><button id="home-button" onclick="window.location.href='index.php';"></button></li>

                <!-- Show "Profile" and "Logout" links if the user is logged in -->
                <?php if ($is_logged_in): ?>
                    <li><a href="profile.php">Profile</a></li>
                    <li><a href="logout.php">Logout</a></li>
                </?php>
            </ul>
        </nav>
    </header>
</body>
</html>
```

# Index.php Code Screenshot

- Profile and Logout Links: If the user is logged in, the "Profile" and "Logout" links are displayed, directing the user to their profile page and allowing them to log out.
- Login Link: If the user is not logged in, the "Login" link is displayed, allowing the user to navigate to the login page.
- Create Account Link: A "Create Account" link is always visible, directing new users to the account creation page.
- Main Welcome Section: Inside the <main> tag, the welcome message ("Welcome to David Moore's Website") is prominently displayed inside the <h1> tag
- Conditional Display of Portrait and About Section: If the user is logged in, their portrait image (DMportrait.jpg) is shown, along with an "About Me" section that provides a very brief "biography" of myself.



```
Sep 8 1:49 PM
flatironstudent@fedora:/var/www/html

<li><a href="profile.php">Profile</a></li>
<li><a href="logout.php">Logout</a></li>
<?php else: ?>
<li><a href="login.php">Login</a></li>
<?php endif; ?>
<li><a href="create_account.php">Create Account</a></li>
</ul>
</nav>
</header>

<main>
<div class="content">
<section class="welcome">
<h1>Welcome to David Moore's Website</h1>
</section>

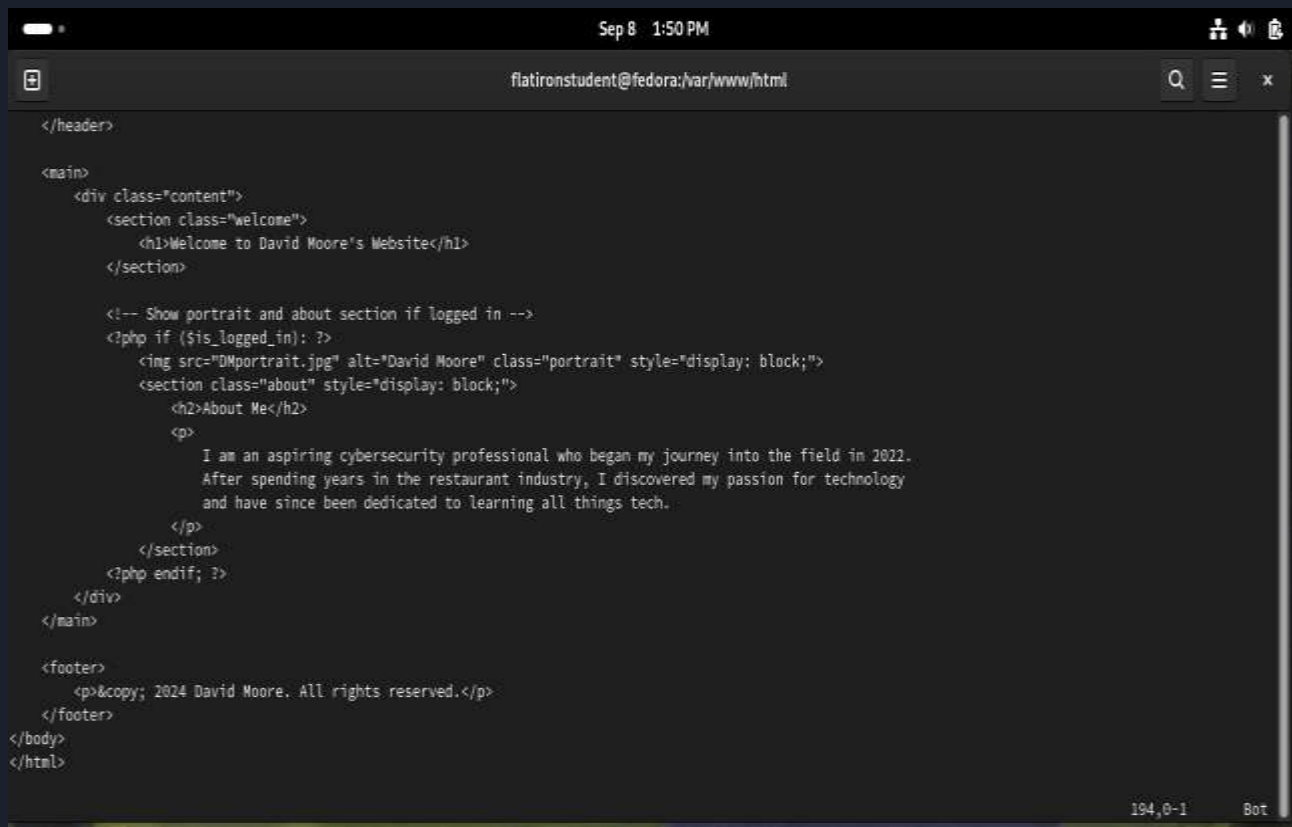
<!-- Show portrait and about section if logged in -->
<?php if ($is_logged_in): ?>

<section class="about" style="display: block;">
<h2>About Me</h2>
<p>
I am an aspiring cybersecurity professional who began my journey into the field in 2022.
After spending years in the restaurant industry, I discovered my passion for technology
and have since been dedicated to learning all things tech.
</p>
</section>
<?php endif; ?>
</div>
```

181,1 95%

# Index.php Code Screenshot

- Main Content Wrapper (<div class="content">): This <div> serves as the container for the main content of the webpage, holding the welcome message and any additional content displayed to the user.
- Footer Section (<footer> Styling



```

</header>

<main>
  <div class="content">
    <section class="welcome">
      <h1>Welcome to David Moore's Website</h1>
    </section>

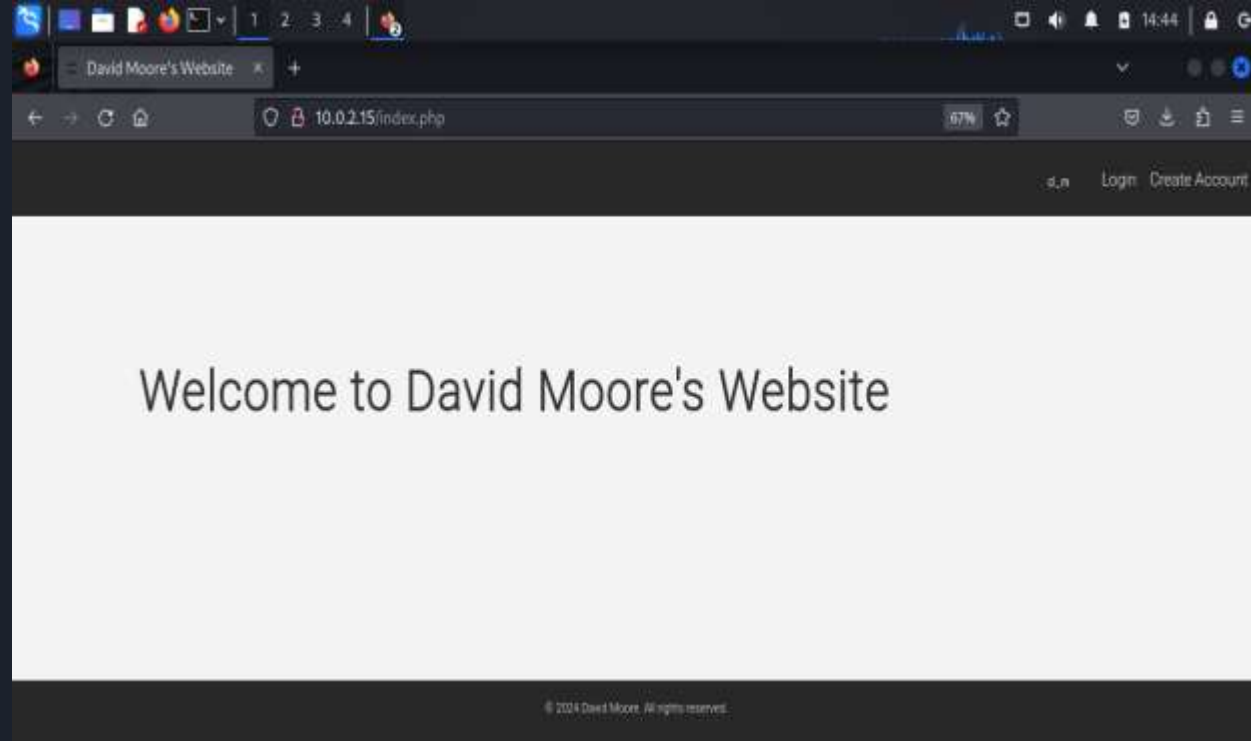
    <!-- Show portrait and about section if logged in -->
    <?php if ($is_logged_in): ?>
      
      <section class="about" style="display: block;">
        <h2>About Me</h2>
        <p>
          I am an aspiring cybersecurity professional who began my journey into the field in 2022.
          After spending years in the restaurant industry, I discovered my passion for technology
          and have since been dedicated to learning all things tech.
        </p>
      </section>
    <?php endif; ?>
  </div>
</main>

<footer>
  <p>&copy; 2024 David Moore. All rights reserved.</p>
</footer>
</body>
</html>

```

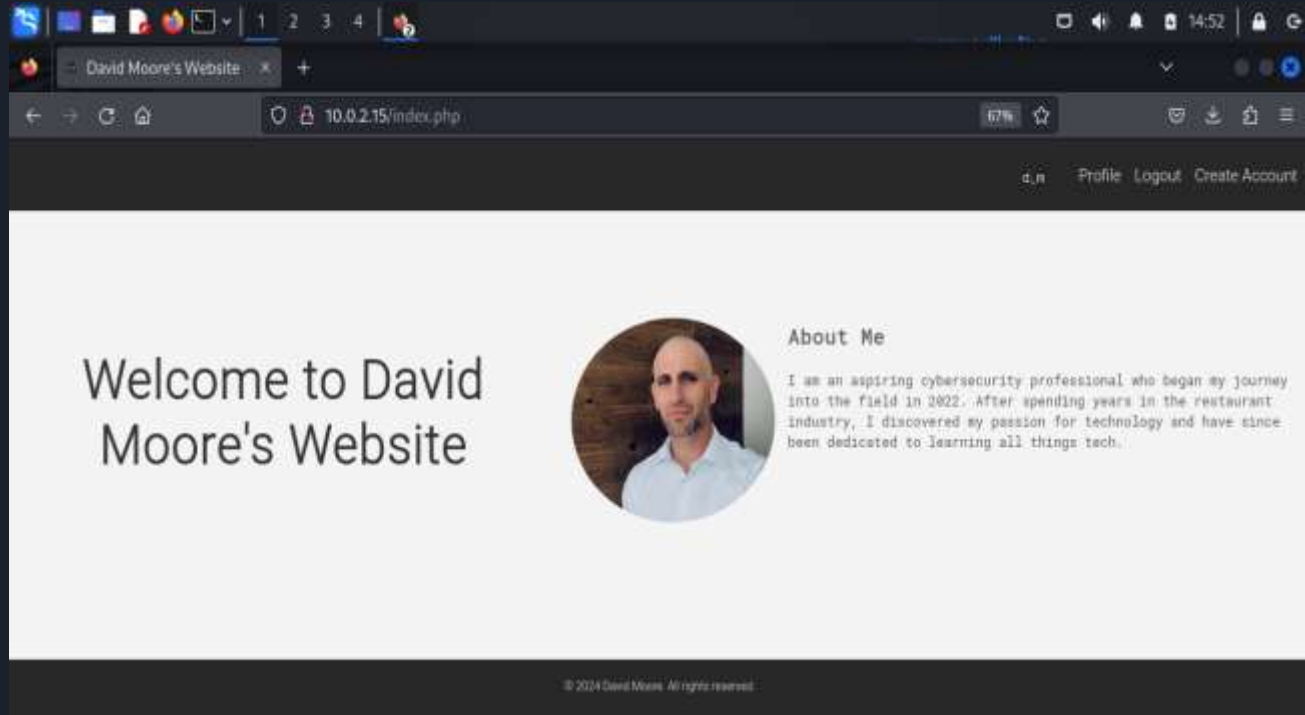
# PHP Webpage Screenshot

- Default display of index.php
- Welcome Message
- Header Bar with navigation links to "Login" and "Create Account"
- Footer Bar with Copyright.

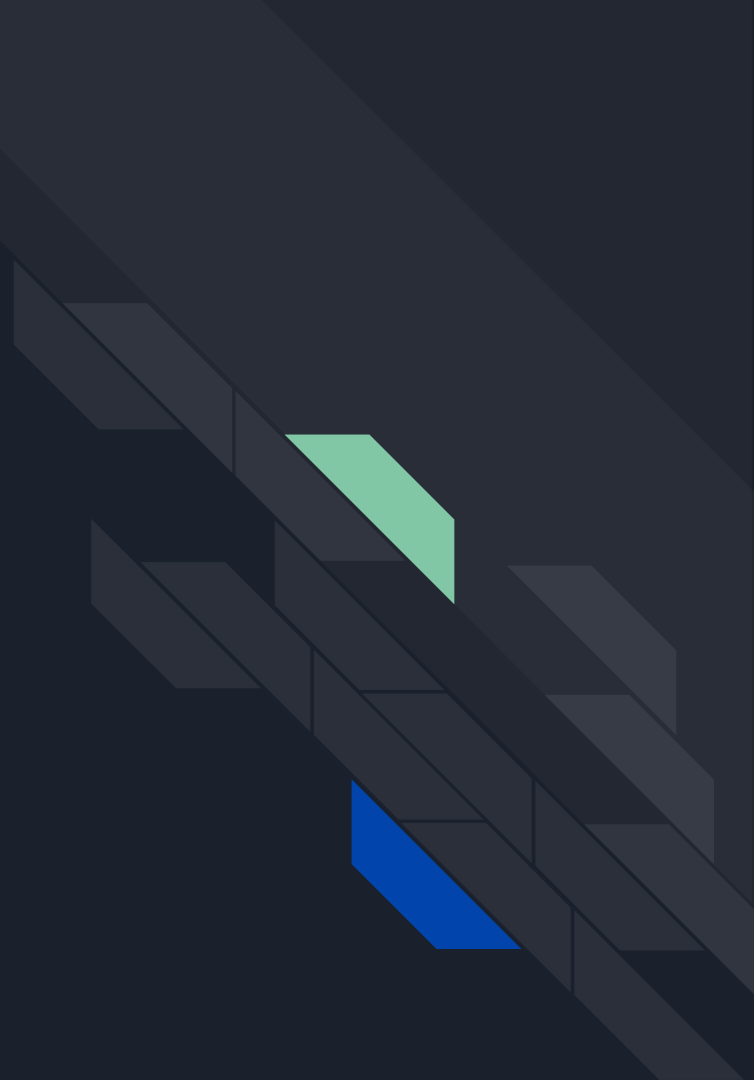


# PHP Webpage Screenshot

- Logged-in display of index.php
- Welcome Message
- Photo and "About Me" Section
- Header Bar with navigation links to "Login" and "Create Account", now includes the "Profile" link.
- Footer Bar with Copyright.



# Database Information







# Database Information

## Description:

The MariaDB database is important for my website because it helps store and manage user data, like account information and login details. By using MariaDB, my website can allow users to create accounts, log in, and access personalized content. It also helps make sure the data is stored securely, which is paramount in cybersecurity. Having a database like MariaDB makes it easier to manage and protect user information, while also making the website more flexible for future updates.

## Key Points about the Page:

- **User Data Management:** The MariaDB page is designed to store and manage user information securely, such as usernames, passwords (encrypted), and any other account-related data within the users table.
- **Data Retrieval and Interaction:** The page integrates with MariaDB to handle dynamic queries, allowing the website to retrieve, update, or delete user data as needed.
- **Database Integration for Scalability:** By using MariaDB, the page ensures that the website can scale efficiently with more users while maintaining data integrity and security, making it adaptable for future improvements and features.

# Database Screenshot

```
Sep 8 3:00 PM
flatironstudent@fedora:/var/www/html

flatironstudent@fedora:/var/www/html$ mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 554
Server version: 10.11.8-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> USE users;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [users]>
MariaDB [users]> describe people;
```

Field	Type	Null	Key	Default	Extra
userid	int(13)	NO	PRI	NULL	auto_increment
username	varchar(31)	NO	UNI	NULL	
fname	varchar(31)	NO		NULL	
lname	varchar(31)	NO		NULL	
pass	varchar(255)	NO		NULL	

```
5 rows in set (0.002 sec)

MariaDB [users]>
```

```
Sep 8 3:03 PM
flatironstudent@fedora:/var/www/html

MariaDB [users]> describe people;
```

Field	Type	Null	Key	Default	Extra
userid	int(13)	NO	PRI	NULL	auto_increment
username	varchar(31)	NO	UNI	NULL	
fname	varchar(31)	NO		NULL	
lname	varchar(31)	NO		NULL	
pass	varchar(255)	NO		NULL	

```
5 rows in set (0.002 sec)

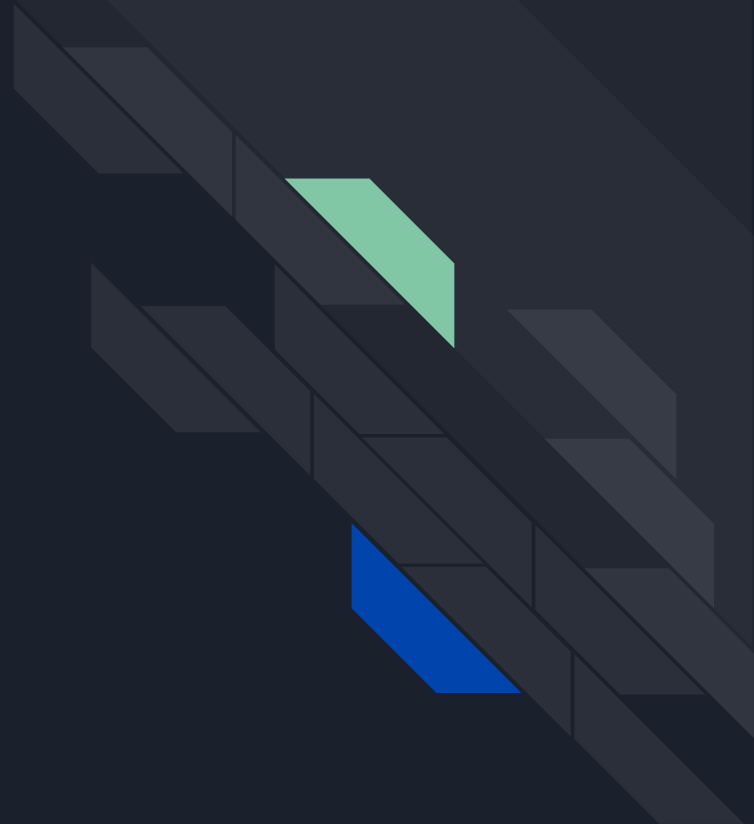
MariaDB [users]> SELECT * FROM users;
ERROR 1146 (42S02): Table 'users.users' doesn't exist
MariaDB [users]> SELECT * FROM people;
```

userid	username	fname	lname	pass
8	davidm	d	m	\$2y\$10\$Uk168uvrAuKpN4EkqhZ3BurbtxNt5SELj6LF93HSe1hjuHxMDHG10
9	Test	Test	Tester	\$2y\$10\$FG4ntWyXfUujQ10rn3QTae6wqYp8Mzz9M/Ec1pxu3bapu7fQJz10K

```
2 rows in set (0.000 sec)

MariaDB [users]>
```

phptest.php





# phptest.php Information

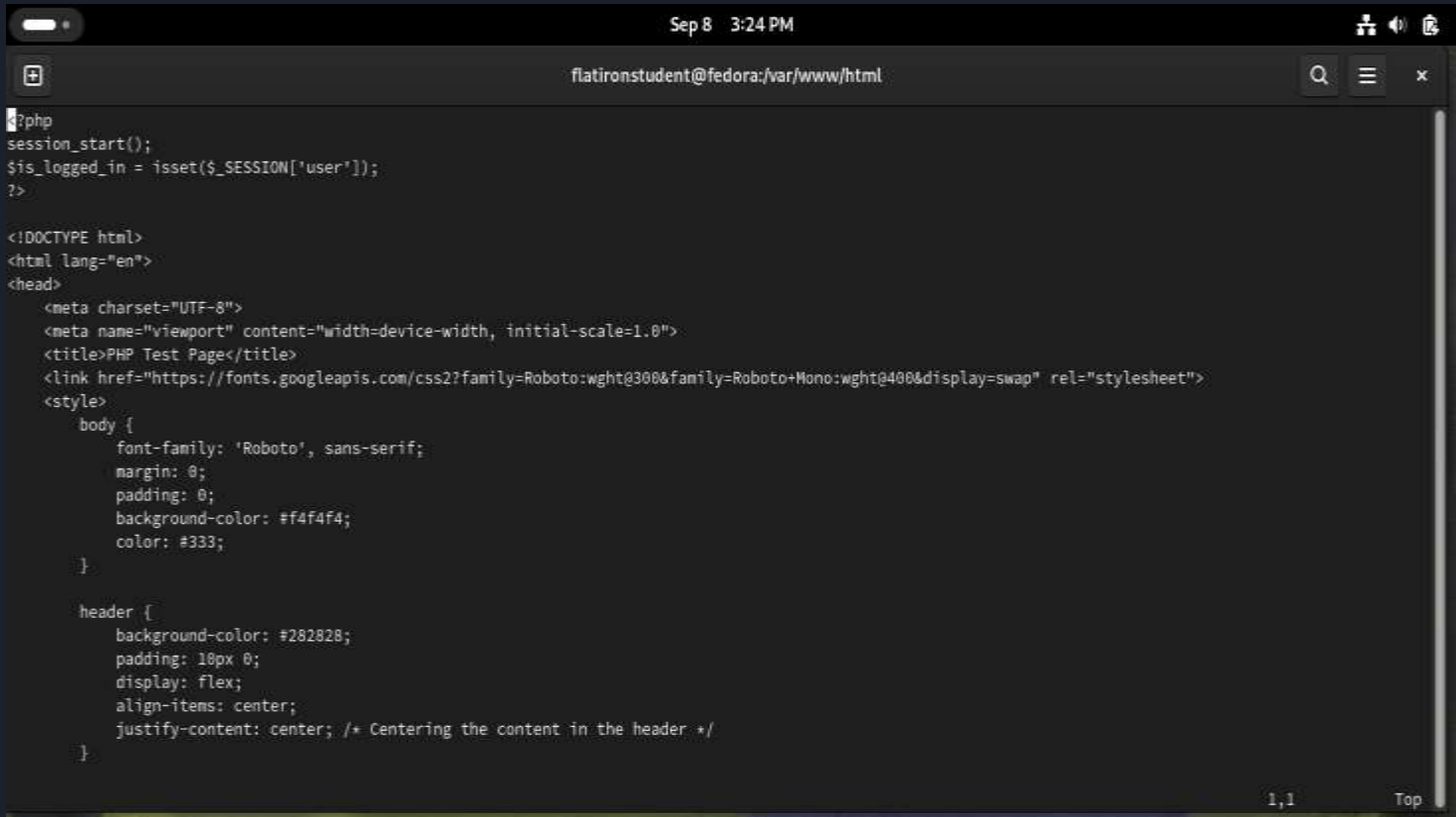
## Description:

The purpose of a `phptest.php` page is to display important information about the PHP configuration on your server. When you access this page, it runs the `phpinfo()` function, which outputs details such as the PHP version, server environment, loaded extensions, and configuration settings. This page is typically used for troubleshooting and verifying that PHP is properly installed and configured, ensuring that your server can support the PHP-based functionality of your website.

## Key Points about the Page:

- **Troubleshooting Tool:** It serves as a useful tool for diagnosing server issues, ensuring that PHP is correctly installed and configured to support your website's functionality.
- **Server Environment Information:** The page displays essential server environment details, including loaded modules and system paths, which can help in debugging and optimizing the website's performance.

# phptest.php Code Screenshot



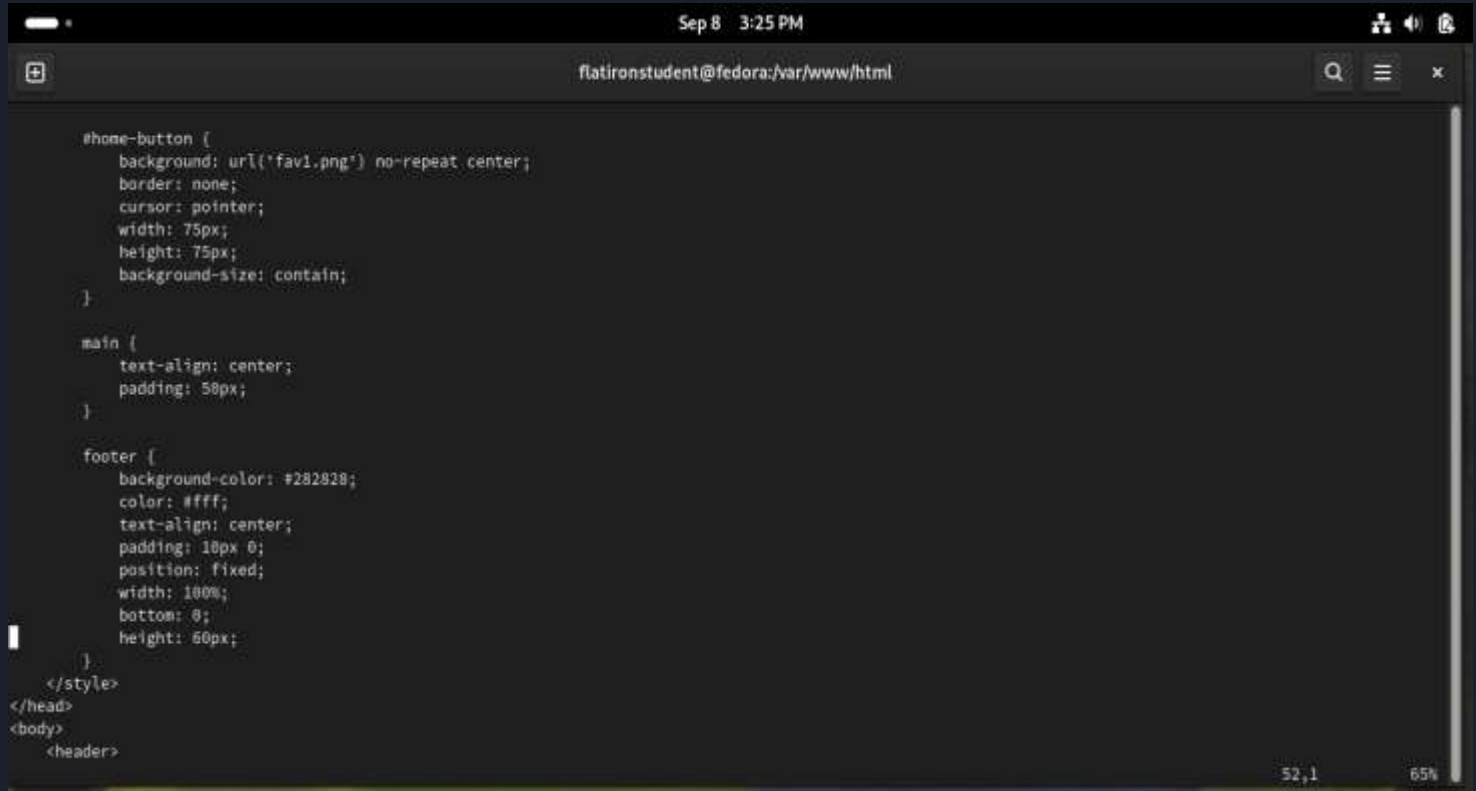
```
?php
session_start();
$is_logged_in = isset($_SESSION['user']);
?>

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>PHP Test Page</title>
  <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300&family=Roboto+Mono:wght@400&display=swap" rel="stylesheet">
  <style>
    body {
      font-family: 'Roboto', sans-serif;
      margin: 0;
      padding: 0;
      background-color: #f4f4f4;
      color: #333;
    }

    header {
      background-color: #282828;
      padding: 10px 0;
      display: flex;
      align-items: center;
      justify-content: center; /* Centering the content in the header */
    }
```

1,1 Top

# phptest.php Code Screenshot

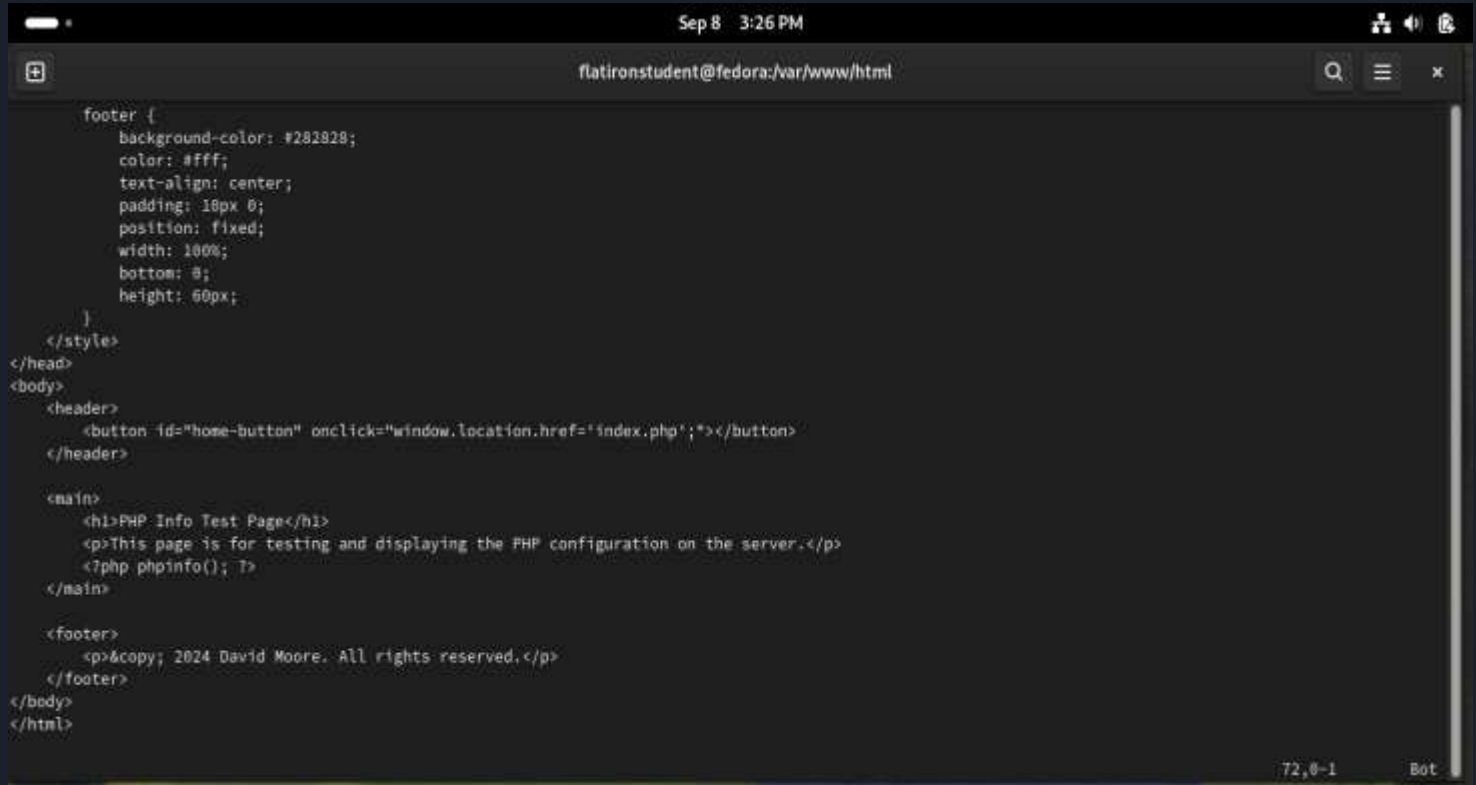


The screenshot shows a code editor window with a dark theme. The title bar at the top indicates the date and time as 'Sep 8 3:25 PM'. The address bar shows the file path 'flatironstudent@fedora:/var/www/html'. The code is written in a light gray font on a dark background. It includes CSS rules for a home button, a main container, and a footer, followed by HTML tags for the head, body, and header sections. The footer CSS rule is partially visible at the bottom of the code block.

```
#home-button {  
  background: url('fav1.png') no-repeat center;  
  border: none;  
  cursor: pointer;  
  width: 75px;  
  height: 75px;  
  background-size: contain;  
}  
  
main {  
  text-align: center;  
  padding: 58px;  
}  
  
footer {  
  background-color: #282828;  
  color: #fff;  
  text-align: center;  
  padding: 18px 0;  
  position: fixed;  
  width: 100%;  
  bottom: 0;  
  height: 60px;  
}  
  
</style>  
</head>  
<body>  
  <header>
```

52,1 65%

# phptest.php Code Screenshot



The screenshot shows a code editor window with a dark theme. The title bar at the top indicates the date and time as 'Sep 8 3:26 PM'. The address bar shows the file path 'flatironstudent@fedora:/var/www/html'. The code is written in HTML and CSS, defining a footer and the main body of a web page. The footer has a fixed position at the bottom with a blue background and white text. The main body contains a header with a button, a main section with a heading and a paragraph, and a footer with a copyright notice.

```

    footer {
      background-color: #282828;
      color: #fff;
      text-align: center;
      padding: 10px 0;
      position: fixed;
      width: 100%;
      bottom: 0;
      height: 60px;
    }
  </style>
</head>
<body>
  <header>
    <button id="home-button" onclick="window.location.href='index.php';"></button>
  </header>

  <main>
    <h1>PHP Info Test Page</h1>
    <p>This page is for testing and displaying the PHP configuration on the server.</p>
    <?php phpinfo(); ?>
  </main>

  <footer>
    <p>&copy; 2024 David Moore. All rights reserved.</p>
  </footer>
</body>
</html>

```

72,0-1 Bot

# php test.php Code Screenshot



**PHP Info Test Page**

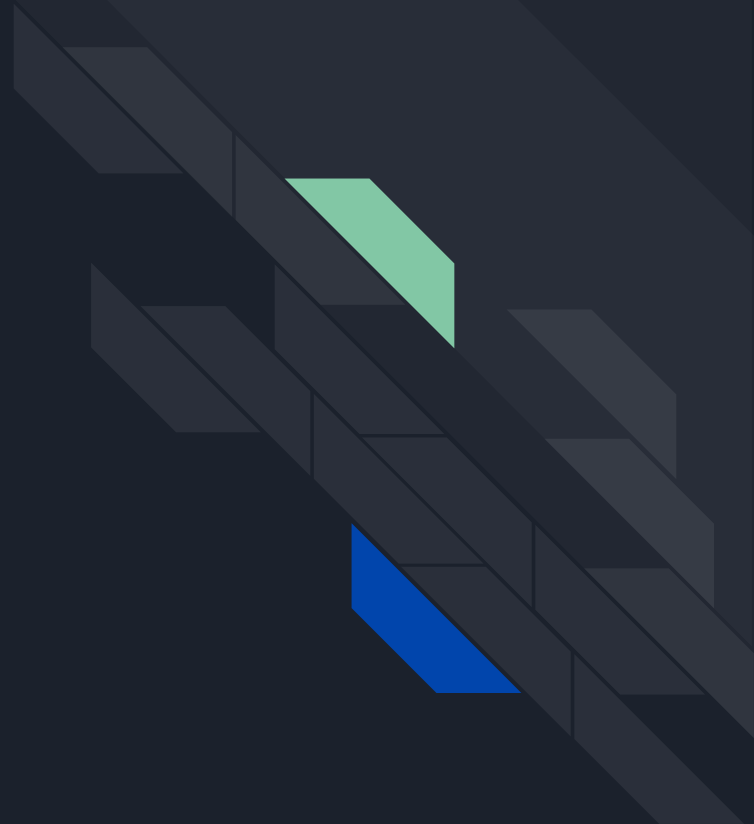
This page is for testing and displaying the PHP configuration on the server.

PHP Version 8.3.10	
System	Linux fedora 6.10.6-200.fc40.x86_64 #1 SMP PREEMPT_DYNAMIC Mon Aug 19 14:09:30 UTC 2024 x86_64
Build Date	Jul 30 2024 11:44:17
Build System	Fedora release 40 (Forty)
Build Provider	Fedora Project
Compiler	gcc (GCC) 14.1.1 20240701 (Red Hat 14.1.1-1)
Architecture	x86_64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/10-opcache.ini, /etc/php.d/20-redis.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-dom.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-ffi.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-gdlib.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-gmp.ini, /etc/php.d/20-gnupg.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-imagick.ini, /etc/php.d/20-intl.ini, /etc/php.d/20-ldap.ini, /etc/php.d/20-openssl.ini, /etc/php.d/20-pdo_mysql.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sysvmsg.ini, /etc/php.d/20-sysvsem.ini, /etc/php.d/20-sysvshm.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/20-xml.ini, /etc/php.d/20-xmlrpc.ini, /etc/php.d/20-zip.ini, /etc/php.d/20-zlib.ini

© 2024 David Moore. All rights reserved.



Connect.php page





# Connect Page Information

## Description:

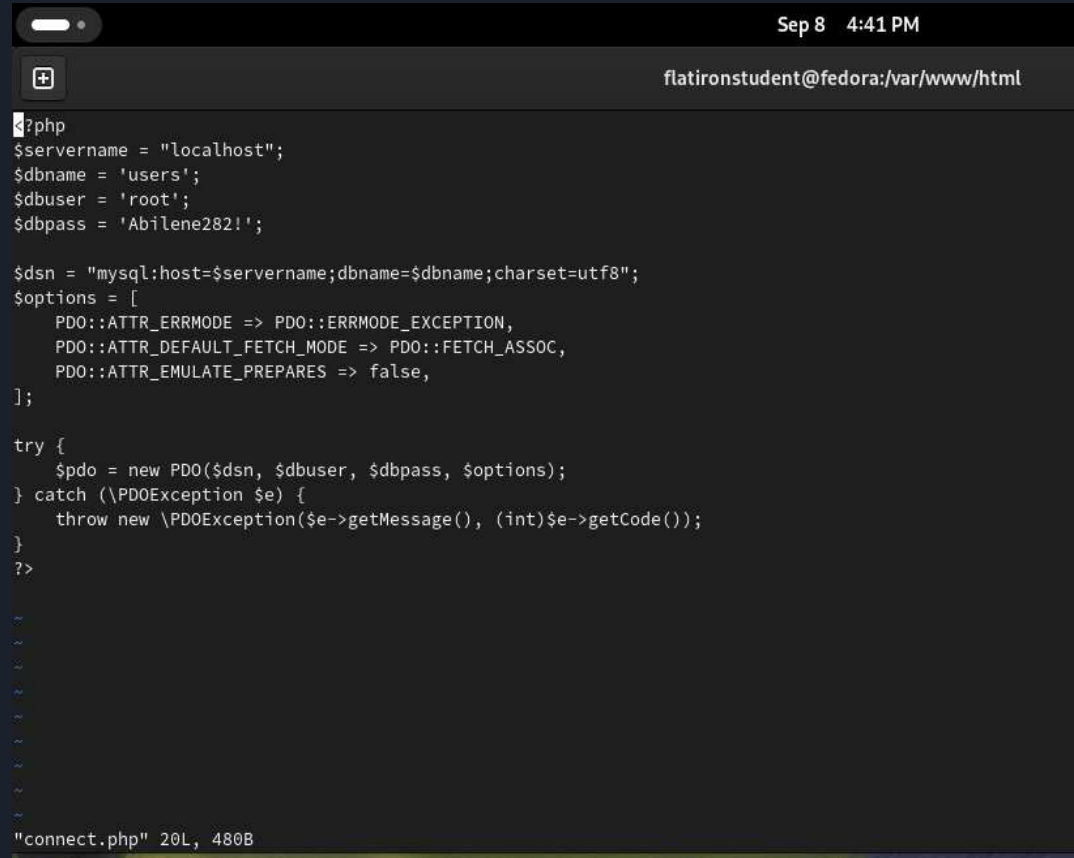
The connect.php page is designed to establish a secure connection to a MariaDB or MySQL database. It defines the necessary credentials such as the server name, database name, username, and password, and then uses PHP Data Objects (PDO) to connect to the database. The page also includes error handling to catch and display connection errors if the connection fails. This setup is important because it allows the website to interact with the database securely and efficiently, enabling features like user authentication or data retrieval.

## Key Points about the Page:

- **Error Handling:** It includes robust error handling through PDOException, ensuring that any issues during the connection process are caught and can be addressed appropriately.
- **Credential Storage:** The page stores the server name, database name, username, and password in variables, which are then used to authenticate the connection, allowing the website to communicate with the database.

# PHP Code Screenshot

- Variables Initialization. Several variables are initialized to store the database connection parameters.
- The \$dsn variable defines the Data Source Name, which contains the information required to connect to the database.
- Connection Options (\$options)
- Try-Catch Block for Connection Handling



The screenshot shows a terminal window with a dark background. The title bar at the top right displays 'Sep 8 4:41 PM'. Below the title bar, the terminal shows the command prompt 'flatironstudent@fedora:/var/www/html'. The code being executed is a PHP script for database connection. It starts with a shebang line '#?php'. The code initializes four variables: \$servername to 'localhost', \$dbname to 'users', \$dbuser to 'root', and \$dbpass to 'Abilene282!'. It then constructs a Data Source Name (\$dsn) as 'mysql:host=\$servername;dbname=\$dbname;charset=utf8;'. An array of options is defined, including PDO::ATTR\_ERRMODE to PDO::ERRMODE\_EXCEPTION, PDO::ATTR\_DEFAULT\_FETCH\_MODE to PDO::FETCH\_ASSOC, and PDO::ATTR\_EMULATE\_PREPARES to false. A try-catch block is used to handle the connection. Inside the try block, a new PDO object is created with the \$dsn, \$dbuser, \$dbpass, and \$options. The catch block catches any PDOException and throws a new PDOException with the message and code from the caught exception. The code ends with a closing tag '?' and a prompt '>'. At the bottom of the terminal, it shows the file name and size: '"connect.php" 20L, 480B'.

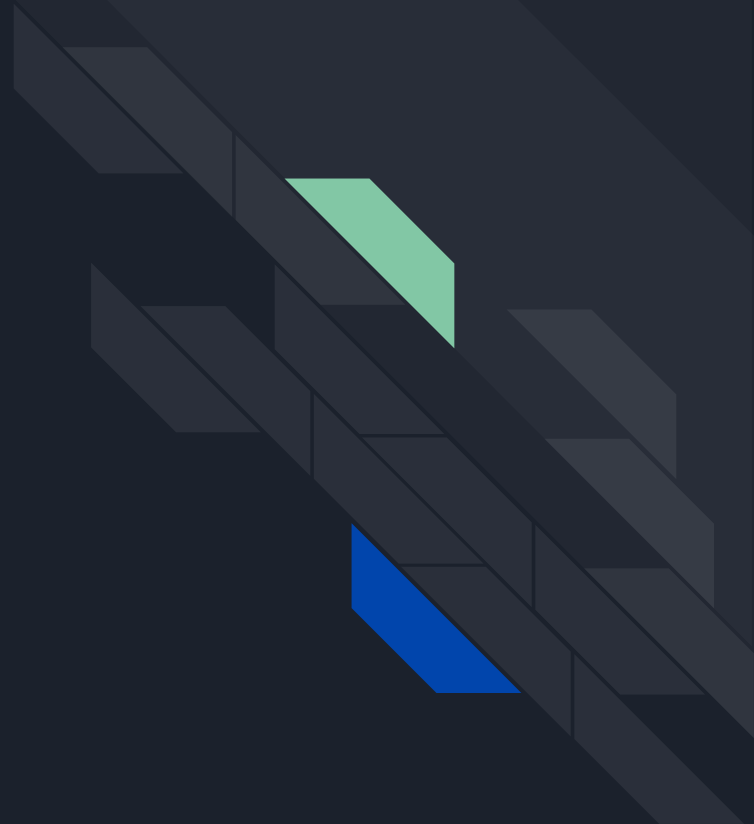
```
#?php
$servername = "localhost";
$dbname = 'users';
$dbuser = 'root';
$dbpass = 'Abilene282!';

$dsn = "mysql:host=$servername;dbname=$dbname;charset=utf8";
$options = [
    PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::ATTR_EMULATE_PREPARES => false,
];

try {
    $pdo = new PDO($dsn, $dbuser, $dbpass, $options);
} catch (\PDOException $e) {
    throw new \PDOException($e->getMessage(), (int)$e->getCode());
}
?>
```

"connect.php" 20L, 480B

# Account Creation Web Page





# Account Creation Page Information

## Description:

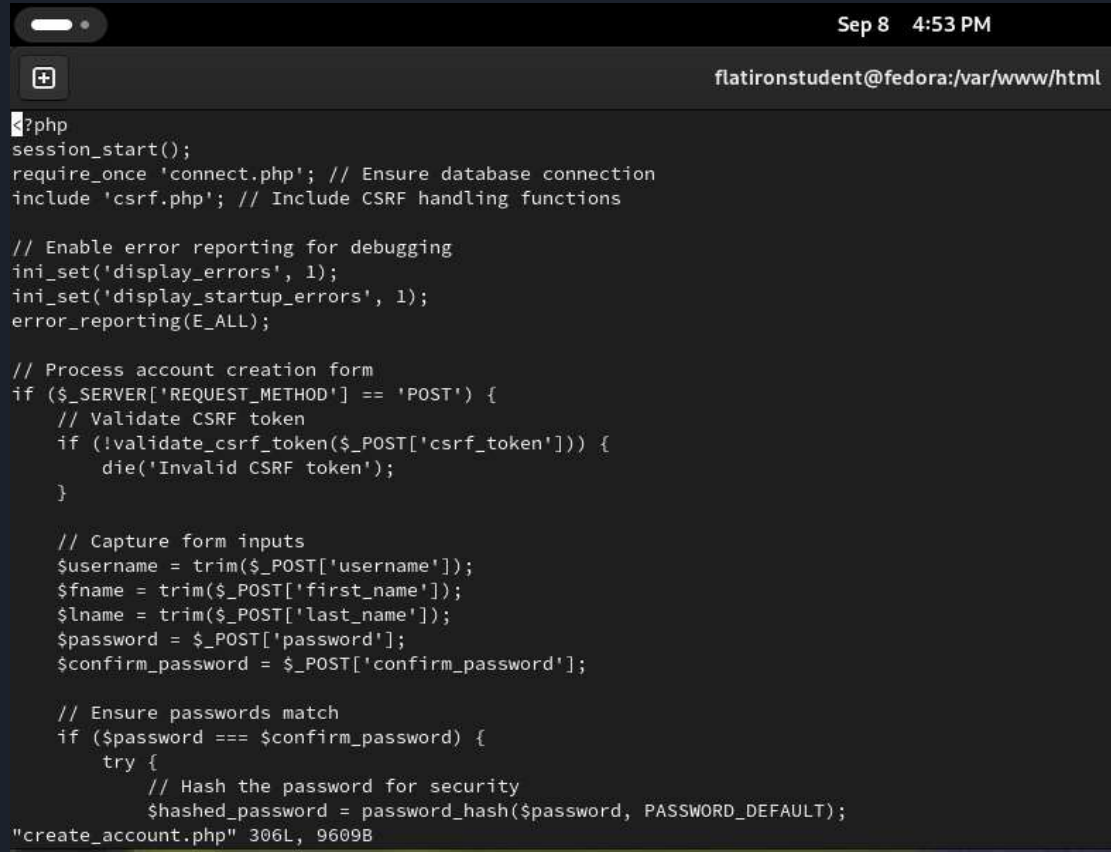
The `create_account.php` page allows users to create a new account by submitting their username, first name, last name, and password through a form. It includes security features such as CSRF token validation to protect against cross-site request forgery and hashes passwords for secure storage. After successfully creating an account, users are redirected to the login page.

## Key Points about the Page:

- **Account Creation Form**
- **Security Features:** It implements security measures like CSRF token validation and password hashing to protect user data and prevent unauthorized submissions.
- **Database Integration:** The page interacts with the database, inserting new user information into the `people` table upon successful form submission.

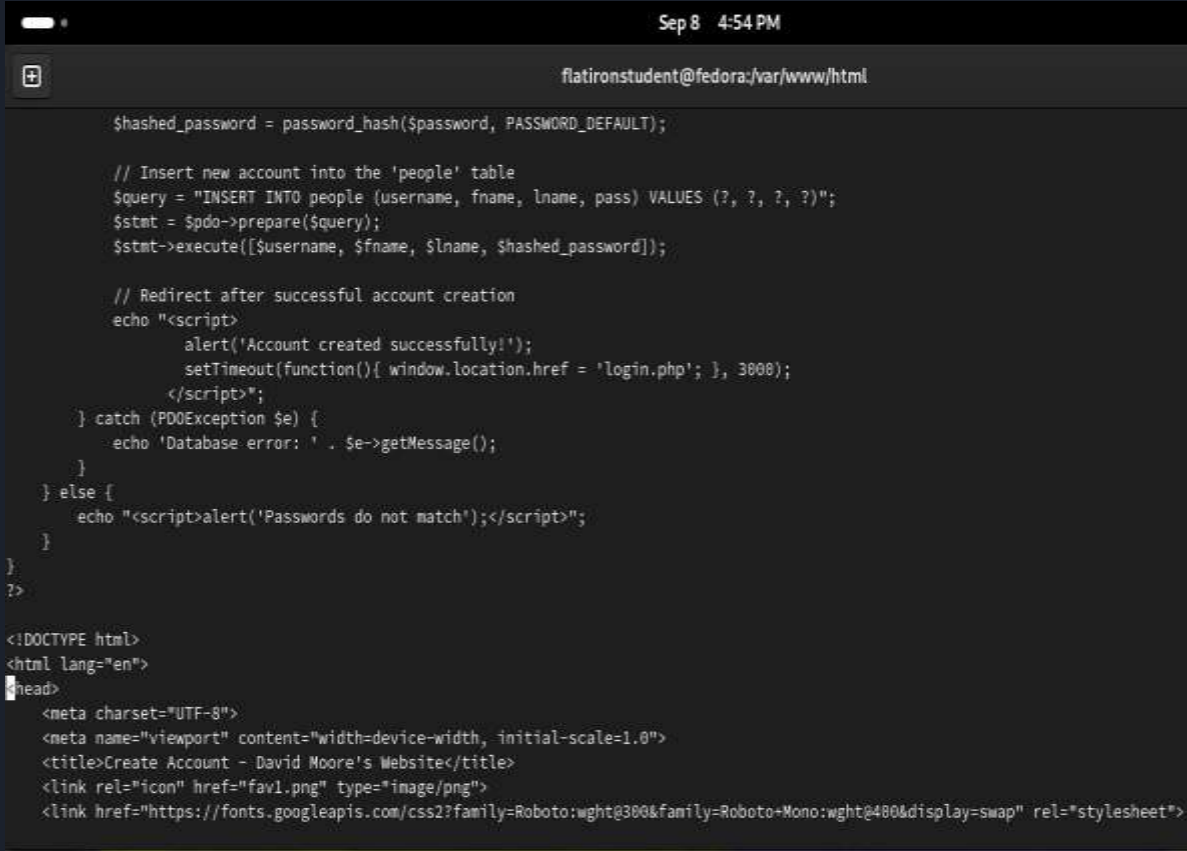
# PHP Code Screenshot

- The code starts by initiating a session with `session_start()` and including necessary files like `connect.php` for database connection and `csrf.php` for CSRF protection. It enables error reporting for debugging, then checks if the form has been submitted using the POST method. Upon submission, it validates the CSRF token to ensure the request is legitimate. User inputs such as username, first name, last name, and password are captured and sanitized. It verifies that the password and confirmation match, and if so, the password is securely hashed using `password_hash()` before being stored in the database.

A screenshot of a code editor window. The title bar at the top right shows the date and time as 'Sep 8 4:53 PM'. Below the title bar, the editor's path is 'flatironstudent@fedora:/var/www/html'. The code is in a dark-themed editor. It starts with a PHP opening tag and includes `session_start()`, `require_once 'connect.php'`, and `include 'csrf.php'`. It then enables error reporting with `ini_set('display_errors', 1)` and `error_reporting(E_ALL)`. The main logic is in a block for POST requests, where it validates a CSRF token. If valid, it captures form inputs (username, first name, last name, password, and confirm password), checks if the password and confirm password match, and if they do, it hashes the password using `password_hash()` and stores it in a database table named 'create\_account.php'.

# PHP Code Screenshot

- Password Hashing (`$hashed_password = password_hash()`)
- SQL Query Preparation and Execution (`$stmt = $pdo->prepare($query);`)
- Success Redirection and Alert (`echo "<script>..."`)
- Error Handling (`catch (PDOException $e)`)
- Password Mismatch Alert (`else { echo "<script>alert(...)"`)



Sep 8 4:54 PM  
flatironstudent@fedora:var/www/html

```
$hashed_password = password_hash($password, PASSWORD_DEFAULT);

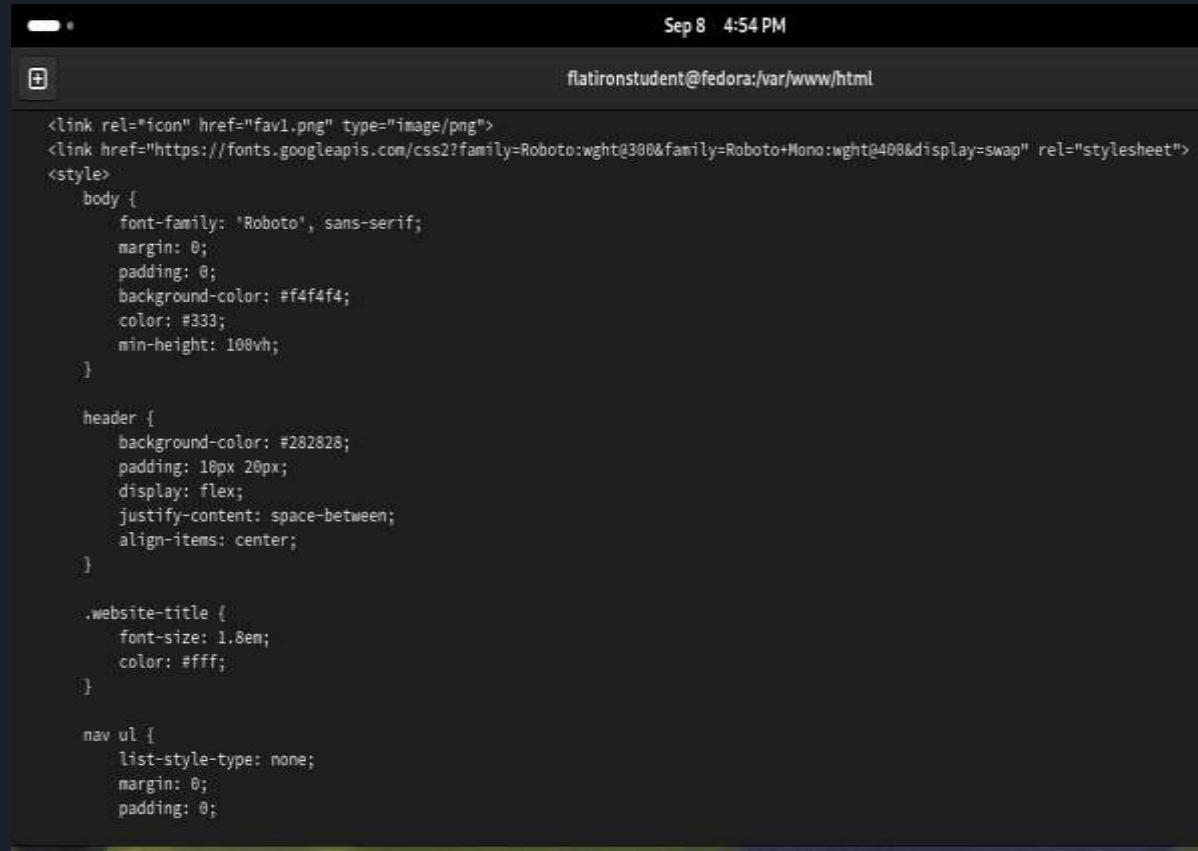
// Insert new account into the 'people' table
$query = "INSERT INTO people (username, fname, lname, pass) VALUES (?, ?, ?, ?)";
$stmt = $pdo->prepare($query);
$stmt->execute([$username, $fname, $lname, $hashed_password]);

// Redirect after successful account creation
echo "<script>
    alert('Account created successfully!');
    setTimeout(function(){ window.location.href = 'login.php'; }, 3000);
</script>";
} catch (PDOException $e) {
    echo 'Database error: ' . $e->getMessage();
}
} else {
    echo "<script>alert('Passwords do not match!');</script>";
}
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Create Account - David Moore's Website</title>
    <link rel="icon" href="fav1.png" type="image/png">
    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght=300&family=Roboto Mono:wght=400&display=swap" rel="stylesheet">
```

# PHP Code Screenshot

- body Styling
- header Styling
- .website-title Class
- nav ul Styling



The screenshot shows a code editor window with a dark theme. The title bar at the top indicates the date and time as 'Sep 8 4:54 PM'. Below the title bar, the file path is displayed as 'flatironstudent@fedora:/var/www/html'. The code content is as follows:

```
<link rel="icon" href="fav1.png" type="image/png">
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300&family=Roboto+Mono:wght@400&display=swap" rel="stylesheet">
<style>
  body {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    color: #333;
    min-height: 100vh;
  }

  header {
    background-color: #282828;
    padding: 10px 20px;
    display: flex;
    justify-content: space-between;
    align-items: center;
  }

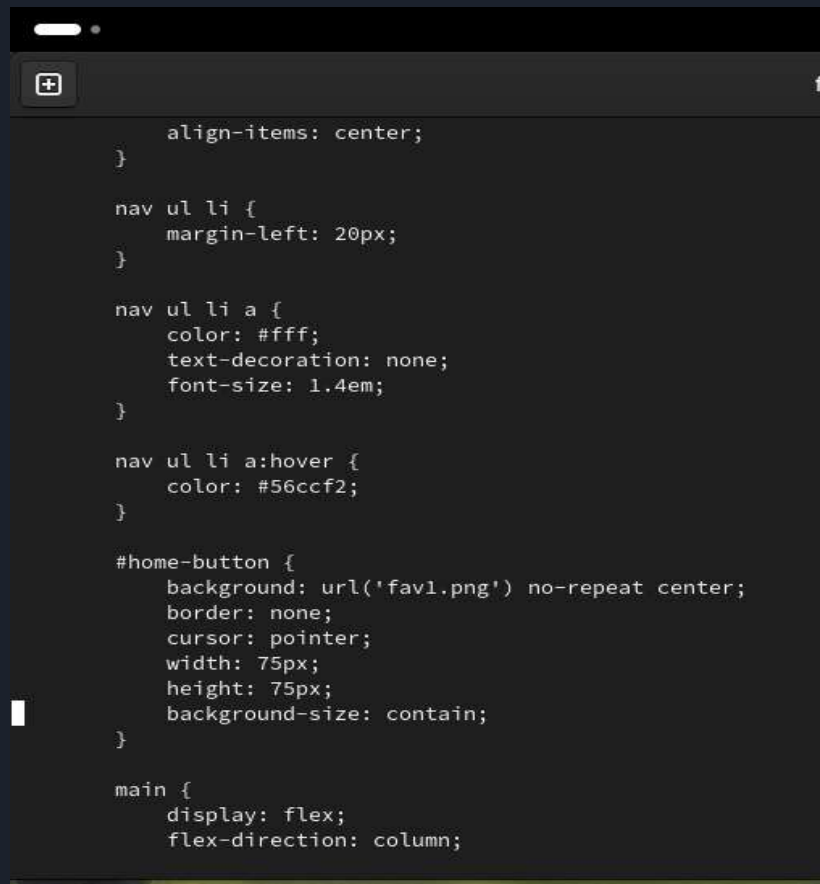
  .website-title {
    font-size: 1.8em;
    color: #fff;
  }

  nav ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
```



# PHP Code Screenshot

- nav ul li Styling
- nav ul li a Styling
- nav ul li a:hover Styling
- #home-button Styling
- main Styling



```
    align-items: center;
}

nav ul li {
    margin-left: 20px;
}

nav ul li a {
    color: #fff;
    text-decoration: none;
    font-size: 1.4em;
}

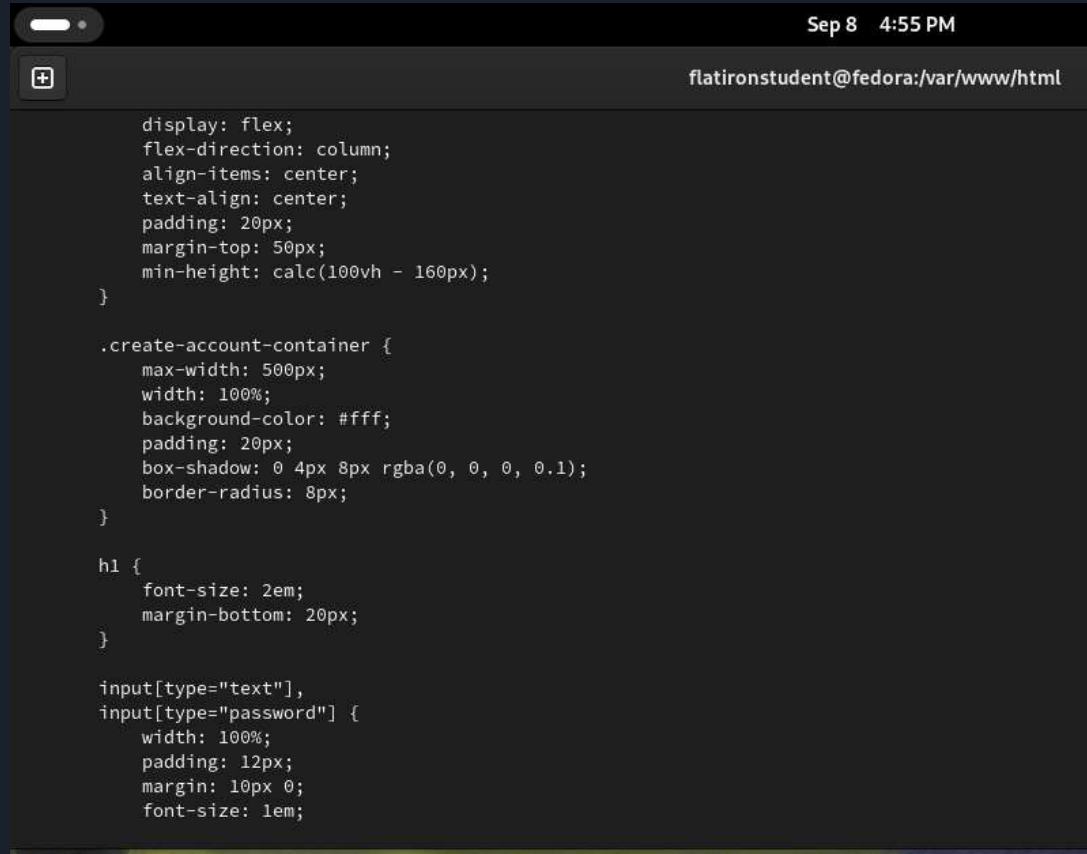
nav ul li a:hover {
    color: #56ccf2;
}

#home-button {
    background: url('fav1.png') no-repeat center;
    border: none;
    cursor: pointer;
    width: 75px;
    height: 75px;
    background-size: contain;
}

main {
    display: flex;
    flex-direction: column;
```

# PHP Code Screenshot

- `.create-account-container` Styling
- `h1` Styling
- Input Fields Styling (`input[type="text"]`, `input[type="password"]`)



The screenshot shows a code editor window with a dark theme. The title bar at the top right displays 'Sep 8 4:55 PM' and the file path 'flatironstudent@fedora:/var/www/html'. The code is CSS, defining styles for a container, a heading, and input fields.

```
display: flex;
flex-direction: column;
align-items: center;
text-align: center;
padding: 20px;
margin-top: 50px;
min-height: calc(100vh - 160px);
}

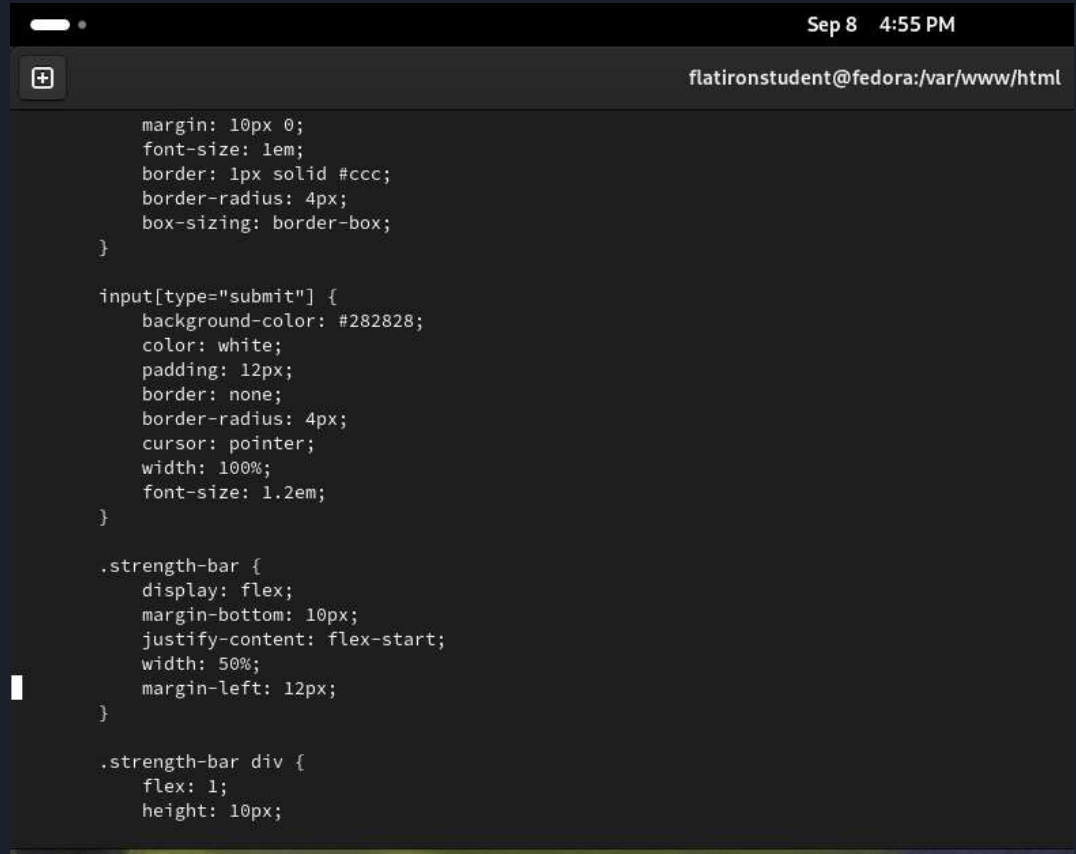
.create-account-container {
  max-width: 500px;
  width: 100%;
  background-color: #fff;
  padding: 20px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  border-radius: 8px;
}

h1 {
  font-size: 2em;
  margin-bottom: 20px;
}

input[type="text"],
input[type="password"] {
  width: 100%;
  padding: 12px;
  margin: 10px 0;
  font-size: 1em;
}
```

# PHP Code Screenshot

- Input Fields (input[type="text"]) Styling
- Submit Button (input[type="submit"]) Styling
- Password Strength Bar (.strength-bar) Styling
- Password Strength Bar Divs (.strength-bar div)



The screenshot shows a terminal window with a dark background. The title bar at the top right indicates the date and time as 'Sep 8 4:55 PM'. Below the title bar, the terminal shows the username and path 'flatironstudent@fedora:/var/www/html'. The main content of the terminal is a CSS code snippet. The code defines styles for a general input field, a submit button, a password strength bar, and its internal divs. The submit button is styled with a dark background and white text. The password strength bar is styled with a flex display and a width of 50%. The internal divs of the strength bar are styled with a flex display and a height of 10px.

```
margin: 10px 0;
font-size: 1em;
border: 1px solid #ccc;
border-radius: 4px;
box-sizing: border-box;
}

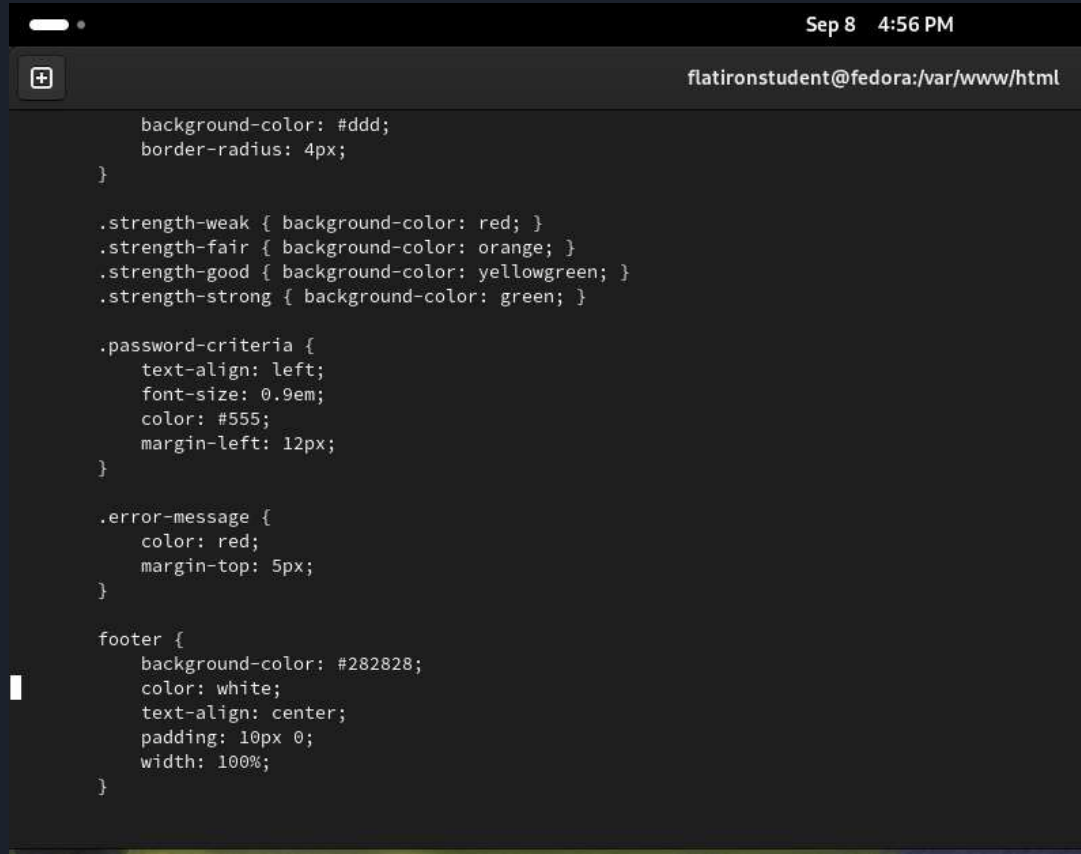
input[type="submit"] {
  background-color: #282828;
  color: white;
  padding: 12px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  width: 100%;
  font-size: 1.2em;
}

.strength-bar {
  display: flex;
  margin-bottom: 10px;
  justify-content: flex-start;
  width: 50%;
  margin-left: 12px;
}

.strength-bar div {
  flex: 1;
  height: 10px;
```

# PHP Code Screenshot

- Password Strength Indicators (.strength-weak, .strength-fair, .strength-good, .strength-strong) Each class changes the background color of the password strength bar based on the strength of the password, ranging from red (weak) to green (strong).
- Password Criteria (.password-criteria) Styles the password criteria section with left-aligned text, smaller font size (0.9em), and a light color to guide users on password requirements.
- Error Message (.error-message) Styles error messages with a red color and a small top margin to clearly indicate validation issues to the user.
- Footer Styling



```
Sep 8 4:56 PM
flatironstudent@fedora:/var/www/html

background-color: #ddd;
border-radius: 4px;
}

.strength-weak { background-color: red; }
.strength-fair { background-color: orange; }
.strength-good { background-color: yellowgreen; }
.strength-strong { background-color: green; }

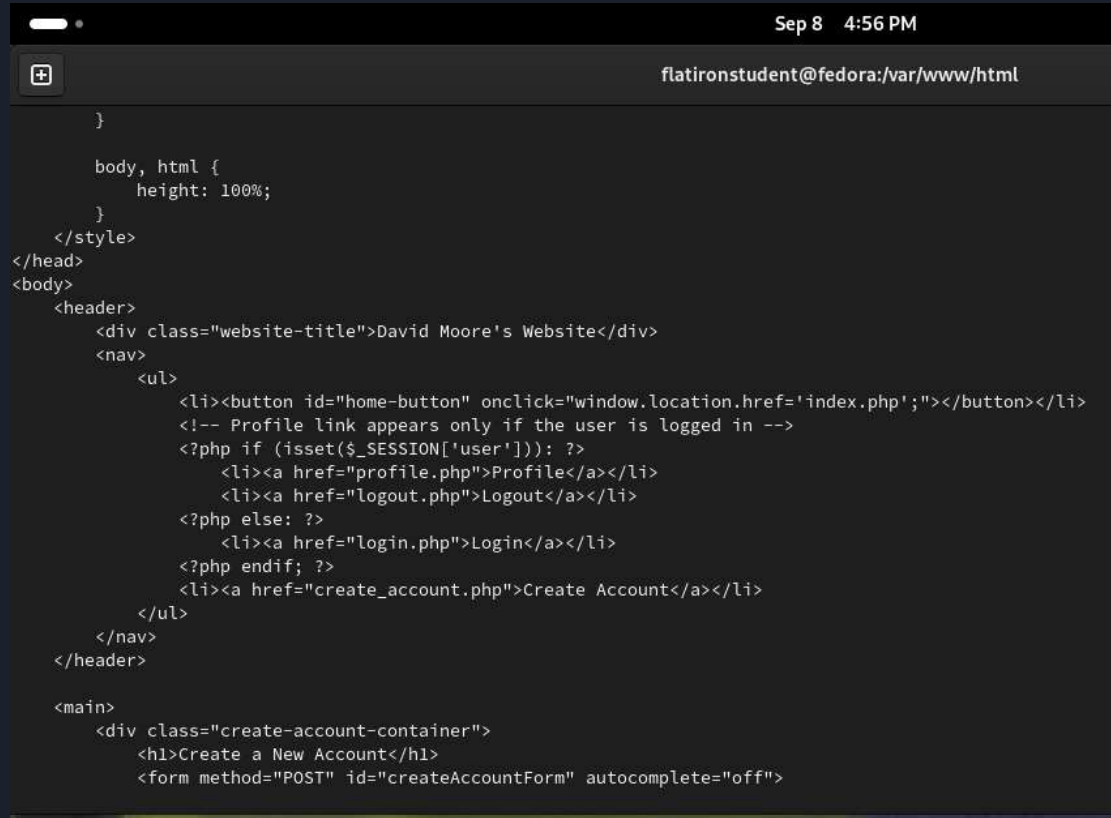
.password-criteria {
  text-align: left;
  font-size: 0.9em;
  color: #555;
  margin-left: 12px;
}

.error-message {
  color: red;
  margin-top: 5px;
}

footer {
  background-color: #282828;
  color: white;
  text-align: center;
  padding: 10px 0;
  width: 100%;
}
```

# PHP Code Screenshot

- PHP Session Check (<?php if (isset(\$\_SESSION['user'])): ?>)Checks if a user session is active (i.e., the user is logged in), and if so, displays the "Profile" and "Logout" links.
- Conditional Links (<?php else: ?>)If the user is not logged in, it shows the "Login" link instead of "Profile" and "Logout."
- Navigation List (<ul><li>)Provides a navigation menu with options like "Profile," "Login," "Logout," and "Create Account," depending on the user's session status.

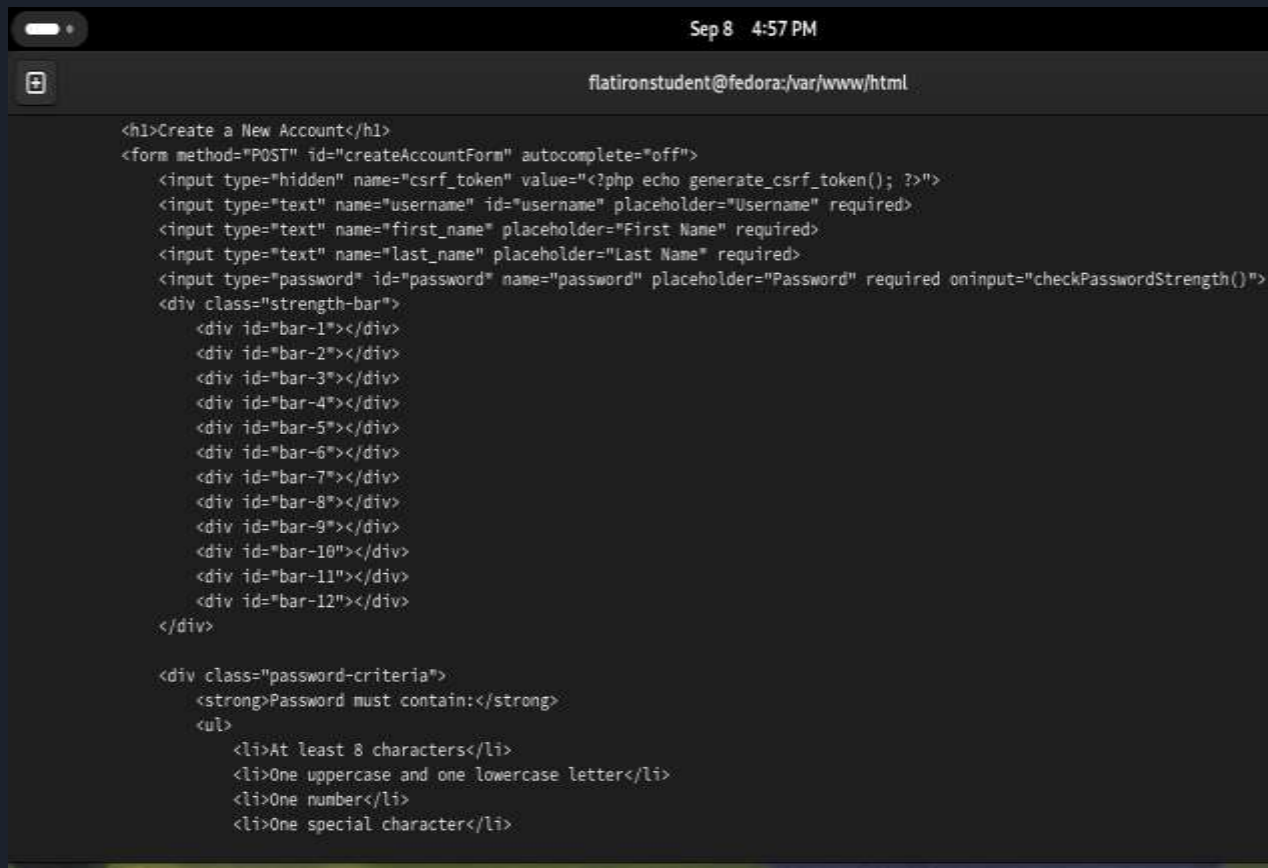


The screenshot shows a code editor window with a dark theme. The title bar at the top right displays "Sep 8 4:56 PM" and the file path "flatironstudent@fedora:/var/www/html". The code is PHP mixed with HTML, defining the structure of a website's header and main content area. It includes a navigation menu that conditionally displays "Profile" and "Logout" links for logged-in users, or a "Login" link for others. A "Create Account" link is also present. The main content area contains a form for creating a new account.

```
}  
  
body, html {  
    height: 100%;  
}  
  
</style>  
</head>  
<body>  
    <header>  
        <div class="website-title">David Moore's Website</div>  
        <nav>  
            <ul>  
                <li><button id="home-button" onclick="window.location.href='index.php';"></button></li>  
                <!-- Profile link appears only if the user is logged in -->  
                <?php if (isset($_SESSION['user'])): ?>  
                    <li><a href="profile.php">Profile</a></li>  
                    <li><a href="logout.php">Logout</a></li>  
                <?php else: ?>  
                    <li><a href="login.php">Login</a></li>  
                <?php endif; ?>  
                <li><a href="create_account.php">Create Account</a></li>  
            </ul>  
        </nav>  
    </header>  
  
    <main>  
        <div class="create-account-container">  
            <h1>Create a New Account</h1>  
            <form method="POST" id="createAccountForm" autocomplete="off">
```

# PHP Code Screenshot

- Text Input Fields
- Password Input Field (<input type="password" ...>) A password input field that triggers the checkPasswordStrength() function to dynamically evaluate the strength of the entered password, with the required attribute to ensure the field is completed.
- Strength Bar (<div class="strength-bar">) A visual strength bar consisting of multiple divs that update in color based on the password strength, providing real-time feedback to the user.
- Password Criteria (<div class="password-criteria">) Displays a list of password requirements (e.g., minimum length, character types) to guide users in creating a secure password.



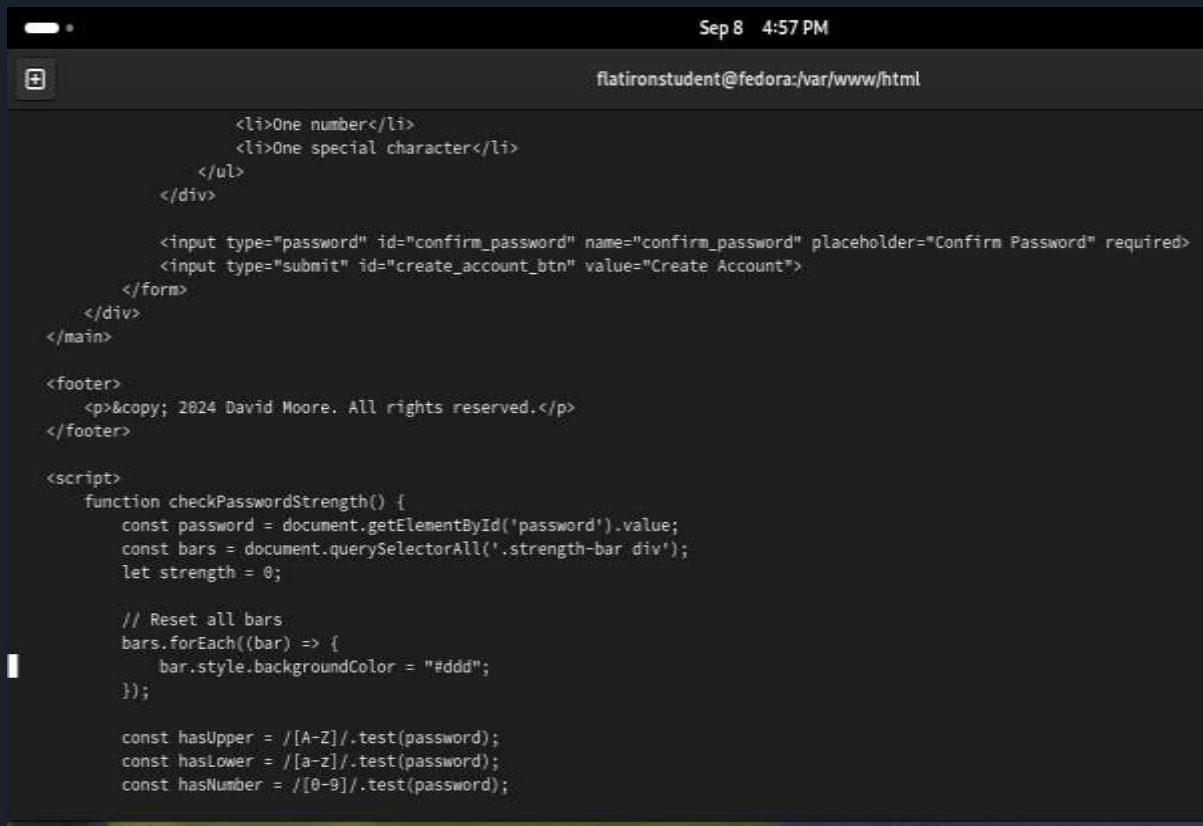
```
Sep 8 4:57 PM
flatironstudent@fedora:/var/www/html

<h1>Create a New Account</h1>
<form method="POST" id="createAccountForm" autocomplete="off">
  <input type="hidden" name="csrf_token" value="<?php echo generate_csrf_token(); ?>">
  <input type="text" name="username" id="username" placeholder="Username" required>
  <input type="text" name="first_name" placeholder="First Name" required>
  <input type="text" name="last_name" placeholder="Last Name" required>
  <input type="password" id="password" name="password" placeholder="Password" required oninput="checkPasswordStrength()">
  <div class="strength-bar">
    <div id="bar-1"></div>
    <div id="bar-2"></div>
    <div id="bar-3"></div>
    <div id="bar-4"></div>
    <div id="bar-5"></div>
    <div id="bar-6"></div>
    <div id="bar-7"></div>
    <div id="bar-8"></div>
    <div id="bar-9"></div>
    <div id="bar-10"></div>
    <div id="bar-11"></div>
    <div id="bar-12"></div>
  </div>

  <div class="password-criteria">
    <strong>Password must contain:</strong>
    <ul>
      <li>At least 8 characters</li>
      <li>One uppercase and one lowercase letter</li>
      <li>One number</li>
      <li>One special character</li>
    </ul>
  </div>
</form>
```

# PHP Code Screenshot

- Password Confirmation Field
- Submit Button
- Footer
- checkPasswordStrength()  
JavaScript Function
- Password Strength Bar ResetResets the background color of all strength bar divs to a neutral color (#ddd) each time the password changes, ensuring the strength indicator is updated in real-time.



```
Sep 8 4:57 PM
flatironstudent@fedora:/var/www/html

<li>One number</li>
<li>One special character</li>
</ul>
</div>

<input type="password" id="confirm_password" name="confirm_password" placeholder="Confirm Password" required>
<input type="submit" id="create_account_btn" value="Create Account">
</form>
</div>
</main>

<footer>
<p>&copy; 2024 David Moore. All rights reserved.</p>
</footer>

<script>
function checkPasswordStrength() {
  const password = document.getElementById('password').value;
  const bars = document.querySelectorAll('.strength-bar div');
  let strength = 0;

  // Reset all bars
  bars.forEach((bar) => {
    bar.style.backgroundColor = "#ddd";
  });

  const hasUpper = /[A-Z]/.test(password);
  const hasLower = /[a-z]/.test(password);
  const hasNumber = /[0-9]/.test(password);
```

# PHP Code Screenshot

- Password Criteria Checks (const hasUpper, hasLower, hasNumber, hasSpecial)
- Password Strength CalculationThe strength of the password is determined by incrementing the strength variable based on whether the password meets certain criteria (length, upper/lower case, numbers, special characters).
- Strength Bar Color Update (Loop)A loop iterates over the strength bar divs and updates their color: red for weak passwords, yellow-green for moderate strength, and green for strong passwords.
- Additional Green Bars for Strong PasswordsIf the password achieves the highest strength score (5), two additional bars in the strength bar are colored green, signaling a very strong password.

```
Sep 8 4:58 PM
flatironstudent@fedora:/var/www/html

const hasLower = /[a-z]/.test(password);
const hasNumber = /[0-9]/.test(password);
const hasSpecial = /[!@#$%^&*(),.?":{}|<>]/.test(password);

if (password.length >= 8) strength++;
if (hasUpper) strength++;
if (hasLower) strength++;
if (hasNumber) strength++;
if (hasSpecial) strength++;

for (let i = 0; i < strength * 2; i++) {
  if (strength < 4) {
    bars[i].style.backgroundColor = "red";
  } else if (strength === 4) {
    bars[i].style.backgroundColor = "yellowgreen";
  } else if (strength === 5) {
    bars[i].style.backgroundColor = "green";
  }
}

if (strength === 5) {
  bars[10].style.backgroundColor = "green";
  bars[11].style.backgroundColor = "green";
}
}
</script>
</body>
</html>
```



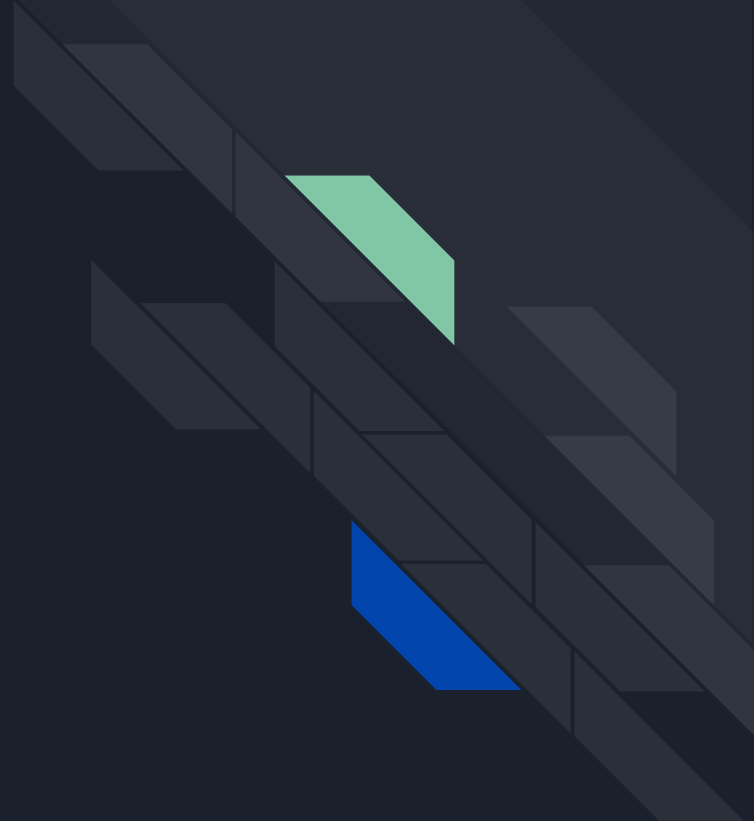
# PHP Webpage Screenshot

The screenshot shows a web browser window with the address bar displaying `10.0.2.15/create_account.php`. The page title is "David Moore's Website". The form is titled "Create a New Account" and contains the following elements:

- Username field: `davidm`
- First Name field: `David`
- Last Name field: `Moore`
- Password field: `*****`
- Progress bar: A green progress bar showing 67% completion.
- Password requirements: "Password must contain:"
  - At least 8 characters
  - One uppercase and one lowercase letter
  - One number
  - One special character
- Confirm Password field: `*****`
- Submit Button: "Create Account"

- Header
- Account Creation Form
- Password Strength Bar
- Password Criteria
- Submit Button

Login Webpage





# Login Page Information

## Description:

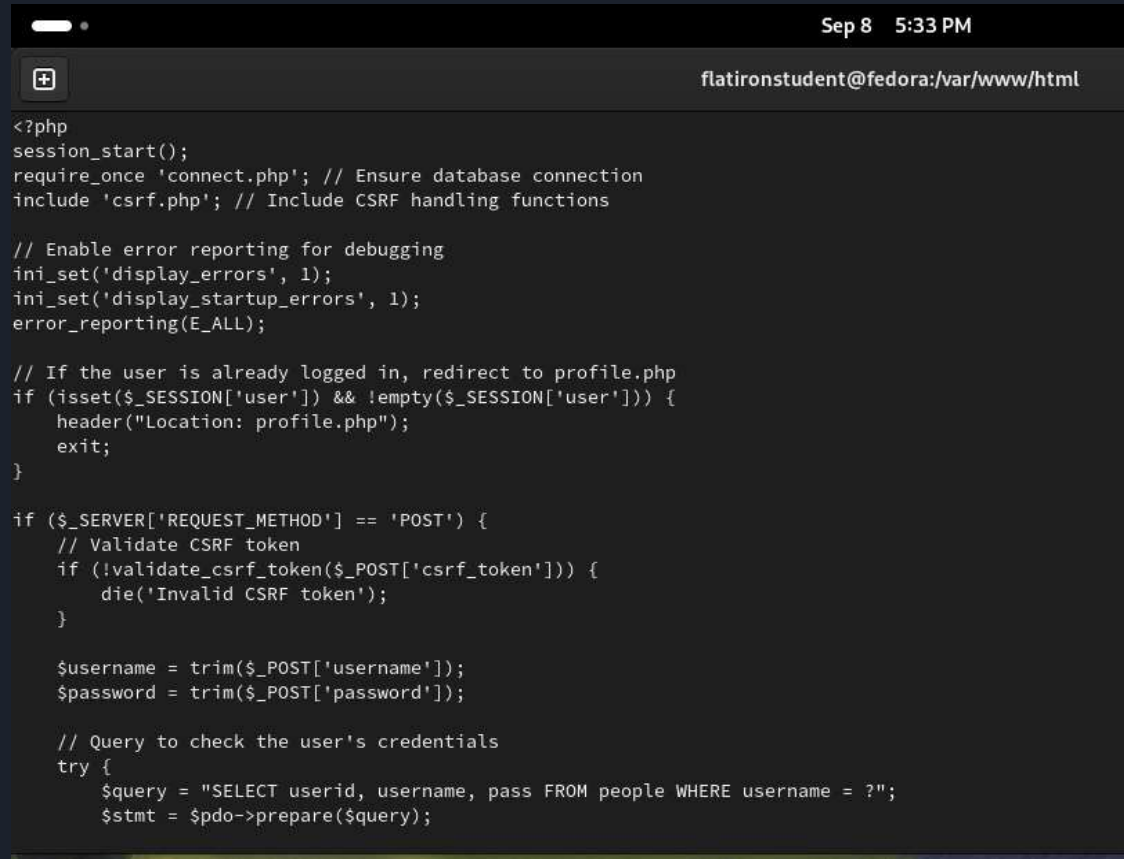
The login.php page is used to let users log in to the website by entering their username and password. It has some security features like a CSRF token to make sure the form submission is safe and checks the password against what's stored in the database. If the user is already logged in, the page will automatically take them to their profile. If the username or password is wrong, an error message will show up. This page helps keep user accounts secure while letting them access their profiles.

## Key Points about the Page:

- User Authentication
- Security Measures
- Error Handling

# PHP Code Screenshot

- Error Reporting Configuration (ini\_set() and error\_reporting())
- Session Check and Redirect (if (isset(\$\_SESSION['user']))) Checks if a user is already logged in by verifying the session and redirects them to the profile page if true.
- Form Submission Handling (if (\$\_SERVER['REQUEST\_METHOD'] == 'POST')) Handles the login form submission and processes user credentials for authentication if the form is submitted via POST.
- CSRF Token Validation
- User Input Capture (\$username and \$password)
- Database Query for Credentials Prepares and executes a query to check the entered username in the database, ensuring the user's credentials are valid.



```
<?php
session_start();
require_once 'connect.php'; // Ensure database connection
include 'csrf.php'; // Include CSRF handling functions

// Enable error reporting for debugging
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);

// If the user is already logged in, redirect to profile.php
if (isset($_SESSION['user']) && !empty($_SESSION['user'])) {
    header("Location: profile.php");
    exit;
}

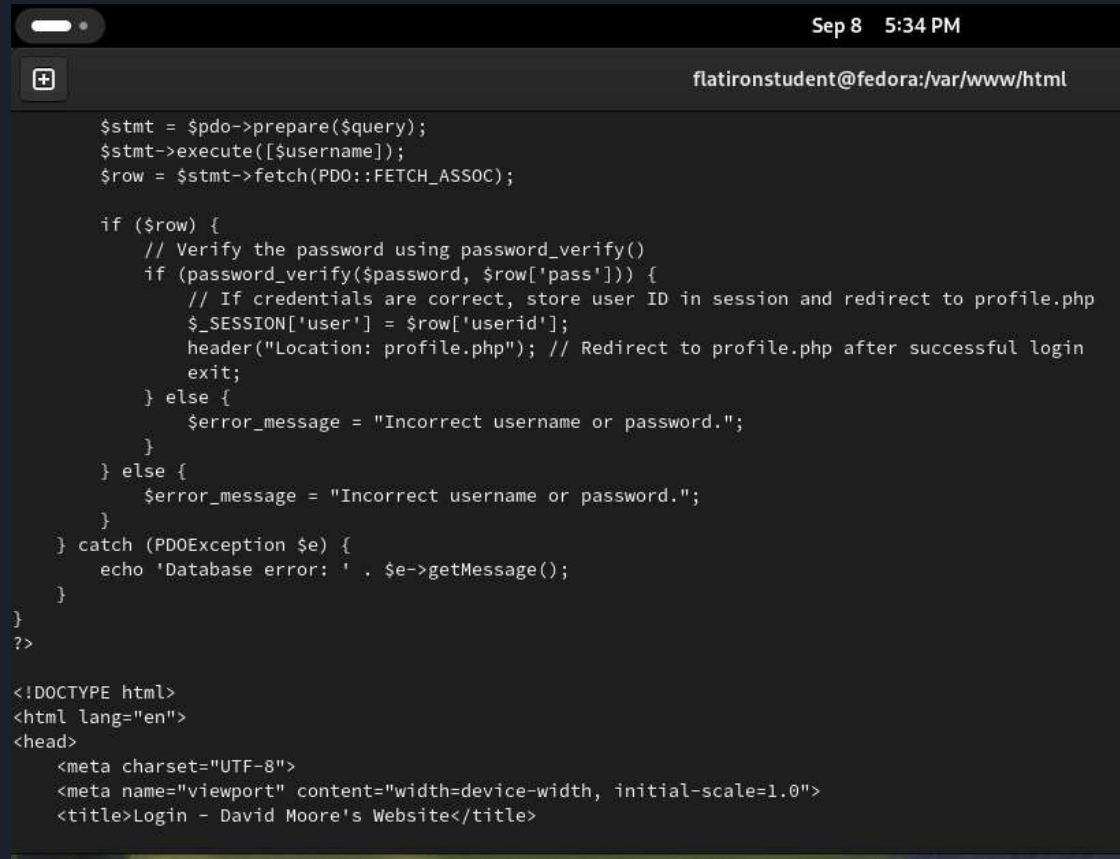
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Validate CSRF token
    if (!validate_csrf_token($_POST['csrf_token'])) {
        die('Invalid CSRF token');
    }

    $username = trim($_POST['username']);
    $password = trim($_POST['password']);

    // Query to check the user's credentials
    try {
        $query = "SELECT userid, username, pass FROM people WHERE username = ?";
        $stmt = $pdo->prepare($query);
```

# PHP Code Screenshot

- Database Query Execution (\$stmt->execute([\$username]));
- Fetch User Data (\$row = \$stmt->fetch(PDO::FETCH\_ASSOC));
- Password Verification (if (password\_verify(\$password, \$row['pass'])))
- Session Setup and Redirection (\$\_SESSION['user'] = \$row['userid'];)
- Error Handling (\$error\_message = "Incorrect username or password.")
- Exception Handling (catch (PDOException \$e))



```
Sep 8 5:34 PM
flatironstudent@fedora:/var/www/html

$stmt = $pdo->prepare($query);
$stmt->execute([$username]);
$row = $stmt->fetch(PDO::FETCH_ASSOC);

if ($row) {
    // Verify the password using password_verify()
    if (password_verify($password, $row['pass'])) {
        // If credentials are correct, store user ID in session and redirect to profile.php
        $_SESSION['user'] = $row['userid'];
        header("Location: profile.php"); // Redirect to profile.php after successful login
        exit;
    } else {
        $error_message = "Incorrect username or password.";
    }
} else {
    $error_message = "Incorrect username or password.";
}

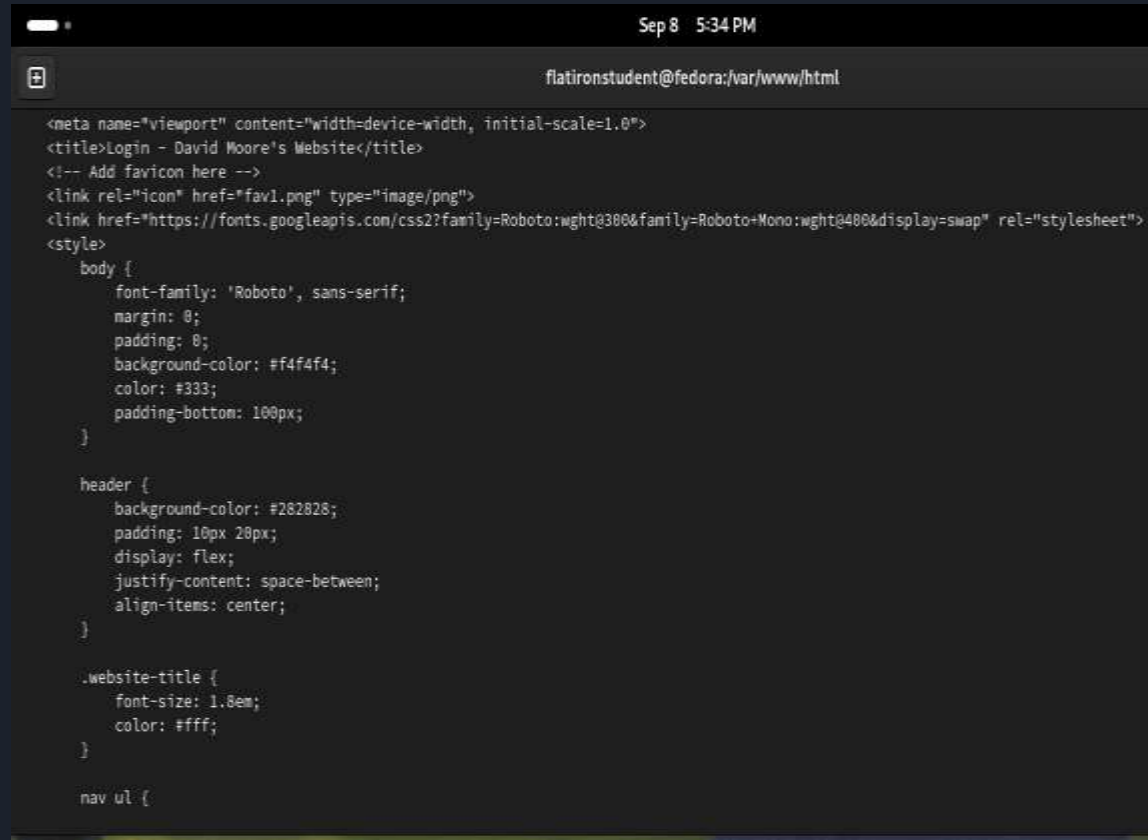
} catch (PDOException $e) {
    echo 'Database error: ' . $e->getMessage();
}

}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login - David Moore's Website</title>
```

# PHP Code Screenshot

- Meta Viewport Tag (<meta name="viewport">)Ensures the page is responsive by setting the viewport width to match the device's width and scaling it properly for mobile devices.
- Favicon (<link rel="icon">)
- Body Styling
- Header Styling
- Website Title Styling
- Navigation List Styling



The screenshot shows a code editor window with a dark theme. The title bar indicates the date and time as 'Sep 8 5:34 PM'. The address bar shows the file path 'flatironstudent@fedora:/var/www/html'. The code is a mix of HTML and CSS. The HTML part includes a meta viewport tag, a title 'Login - David Moore's Website', a comment for a favicon, a link to a favicon, and a link to a Google Fonts stylesheet. The CSS part defines styles for the body, header, website title, and navigation list.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Login - David Moore's Website</title>
<!-- Add favicon here -->
<link rel="icon" href="fav1.png" type="image/png">
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300&family=Roboto+Mono:wght@400&display=swap" rel="stylesheet">
<style>
  body {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    color: #333;
    padding-bottom: 100px;
  }

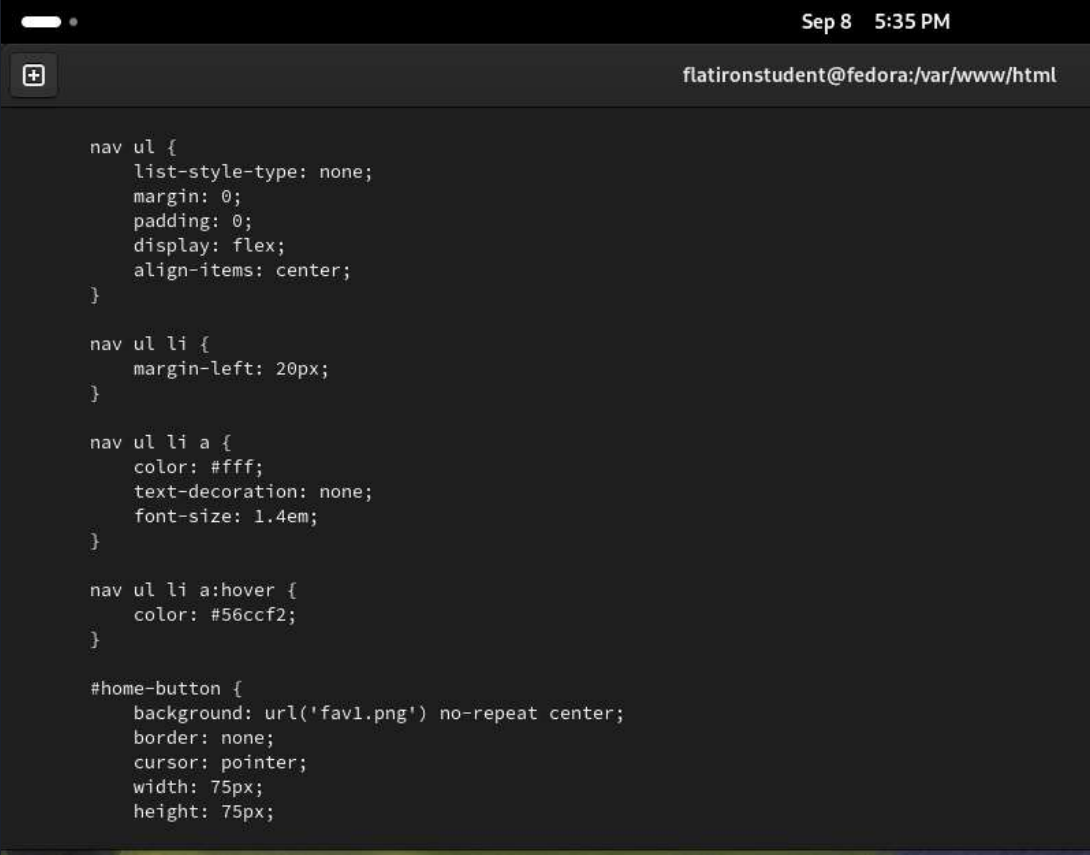
  header {
    background-color: #282828;
    padding: 10px 20px;
    display: flex;
    justify-content: space-between;
    align-items: center;
  }

  .website-title {
    font-size: 1.8em;
    color: #fff;
  }

  nav ul {
```

# PHP Code Screenshot

- Nav link Styling
- Home Button Styling



The screenshot shows a terminal window with a dark background. The title bar at the top right indicates the date and time as 'Sep 8 5:35 PM'. Below the title bar, the user's email 'flatironstudent@fedora:' and the current directory '/var/www/html' are displayed. The main area of the terminal contains CSS code for styling navigation links and a home button. The code is as follows:

```
nav ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  display: flex;
  align-items: center;
}

nav ul li {
  margin-left: 20px;
}

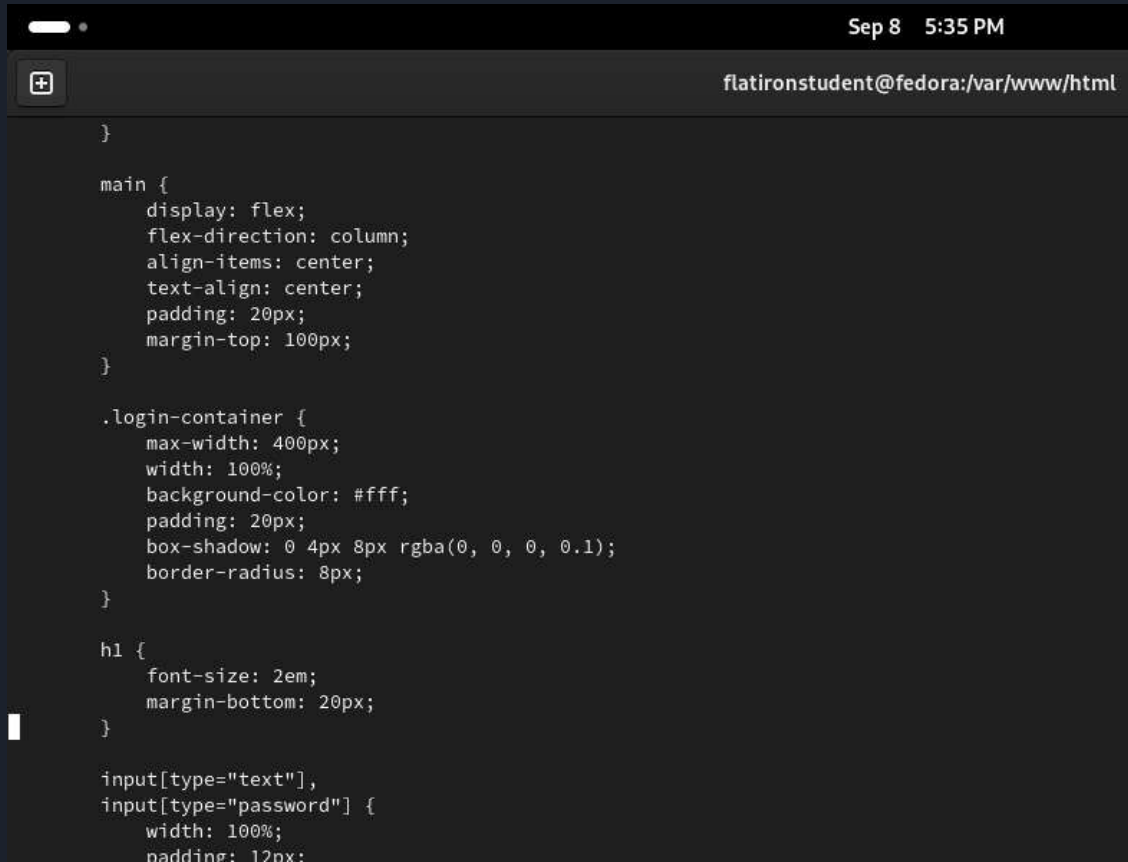
nav ul li a {
  color: #fff;
  text-decoration: none;
  font-size: 1.4em;
}

nav ul li a:hover {
  color: #56ccf2;
}

#home-button {
  background: url('fav1.png') no-repeat center;
  border: none;
  cursor: pointer;
  width: 75px;
  height: 75px;
}
```

# PHP Code Screenshot

- Class Styling continued



The screenshot shows a code editor window with a dark theme. The title bar at the top right indicates the date and time as 'Sep 8 5:35 PM'. Below the title bar, the file path 'flatironstudent@fedora:/var/www/html' is visible. The code is written in a light gray font on a dark background. It defines a 'main' container with flexbox properties and a '.login-container' with a white background, padding, and a box shadow. Below these, 'h1' and form input elements are styled.

```
}

main {
  display: flex;
  flex-direction: column;
  align-items: center;
  text-align: center;
  padding: 20px;
  margin-top: 100px;
}

.login-container {
  max-width: 400px;
  width: 100%;
  background-color: #fff;
  padding: 20px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  border-radius: 8px;
}

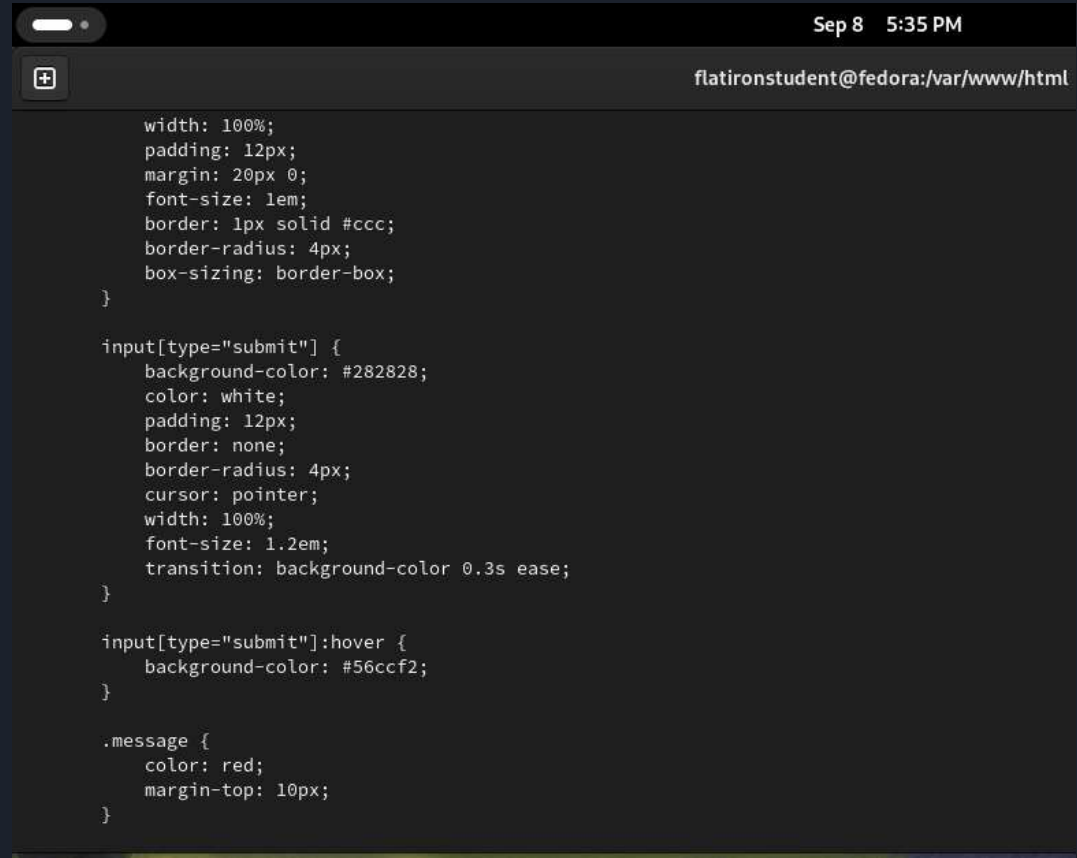
h1 {
  font-size: 2em;
  margin-bottom: 20px;
}

input[type="text"],
input[type="password"] {
  width: 100%;
  padding: 12px;
```



# PHP Code Screenshot

- Text Input Fields Styling
- Submit Button Styling
- Submit Button Hover Effect
- Error Message Styling



The screenshot shows a code editor window with a dark theme. The title bar at the top right displays 'Sep 8 5:35 PM' and the file path 'flatironstudent@fedora:/var/www/html'. The code is CSS, defining styles for text input fields, submit buttons, and error messages. The code is as follows:

```
width: 100%;
padding: 12px;
margin: 20px 0;
font-size: 1em;
border: 1px solid #ccc;
border-radius: 4px;
box-sizing: border-box;
}

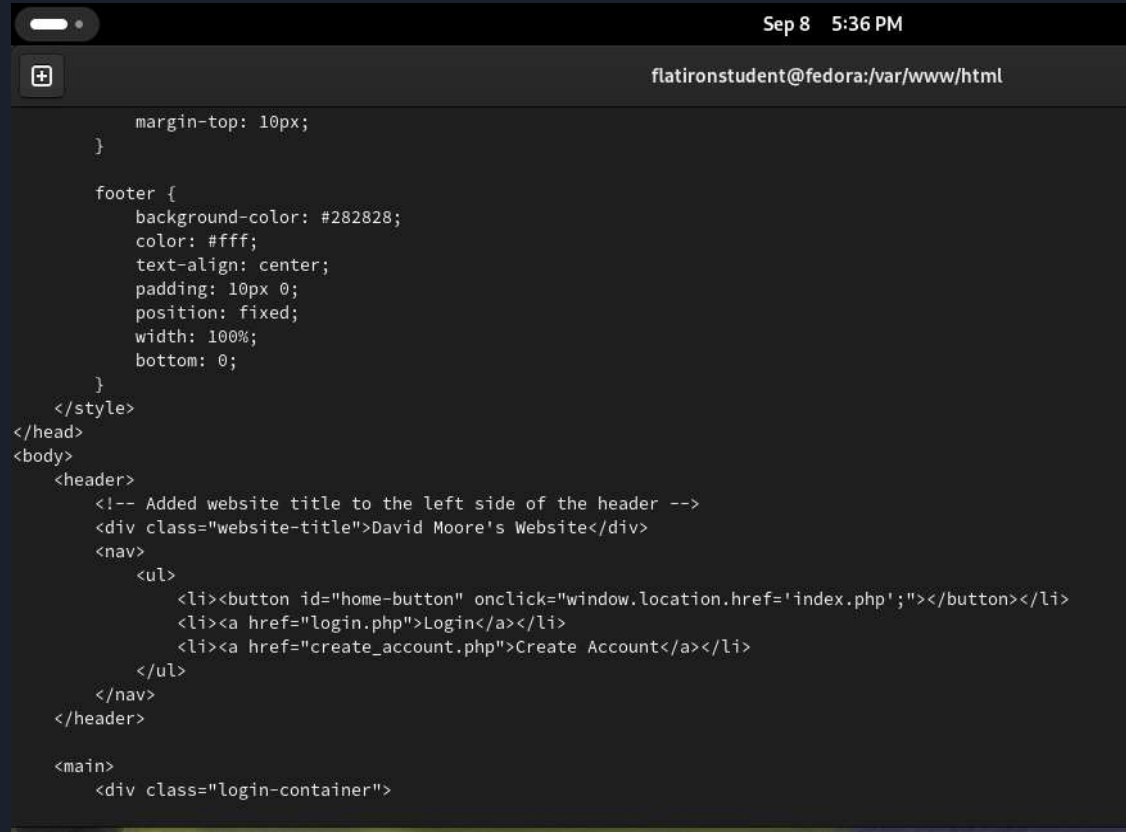
input[type="submit"] {
  background-color: #282828;
  color: white;
  padding: 12px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  width: 100%;
  font-size: 1.2em;
  transition: background-color 0.3s ease;
}

input[type="submit"]:hover {
  background-color: #56ccf2;
}

.message {
  color: red;
  margin-top: 10px;
}
```

# PHP Code Screenshot

- Website Title
- Navigation Menu
- Home Button
- Login Container (<div class="login-container">)A container that holds the login form elements, styled for a clean and organized appearance.



The screenshot shows a code editor window with a dark theme. The title bar at the top indicates the date and time as 'Sep 8 5:36 PM'. Below the title bar, the file path is displayed as 'flatironstudent@fedora:/var/www/html'. The code is written in PHP and HTML, defining the header and navigation menu. It includes CSS styles for a footer and a navigation menu with a home button and a login link. The code is as follows:

```
margin-top: 10px;
}

footer {
    background-color: #282828;
    color: #fff;
    text-align: center;
    padding: 10px 0;
    position: fixed;
    width: 100%;
    bottom: 0;
}
</style>
</head>
<body>
<header>
    <!-- Added website title to the left side of the header -->
    <div class="website-title">David Moore's Website</div>
    <nav>
        <ul>
            <li><button id="home-button" onclick="window.location.href='index.php';"></button></li>
            <li><a href="login.php">Login</a></li>
            <li><a href="create_account.php">Create Account</a></li>
        </ul>
    </nav>
</header>

<main>
    <div class="login-container">
```

# PHP Code Screenshot

- Error Message Handling (<?php if (isset(\$error\_message)): ?>)
- Login Form (<form action="login.php" method="POST">)
- CSRF Token (<input type="hidden" name="csrf\_token" ...>)
- Input Fields for Username and Password
- Submit Button

```
Sep 8 5:36 PM
flatironstudent@fedora:/var/www/html

<li><a href="login.php">Login</a></li>
<li><a href="create_account.php">Create Account</a></li>
</ul>
</nav>
</header>

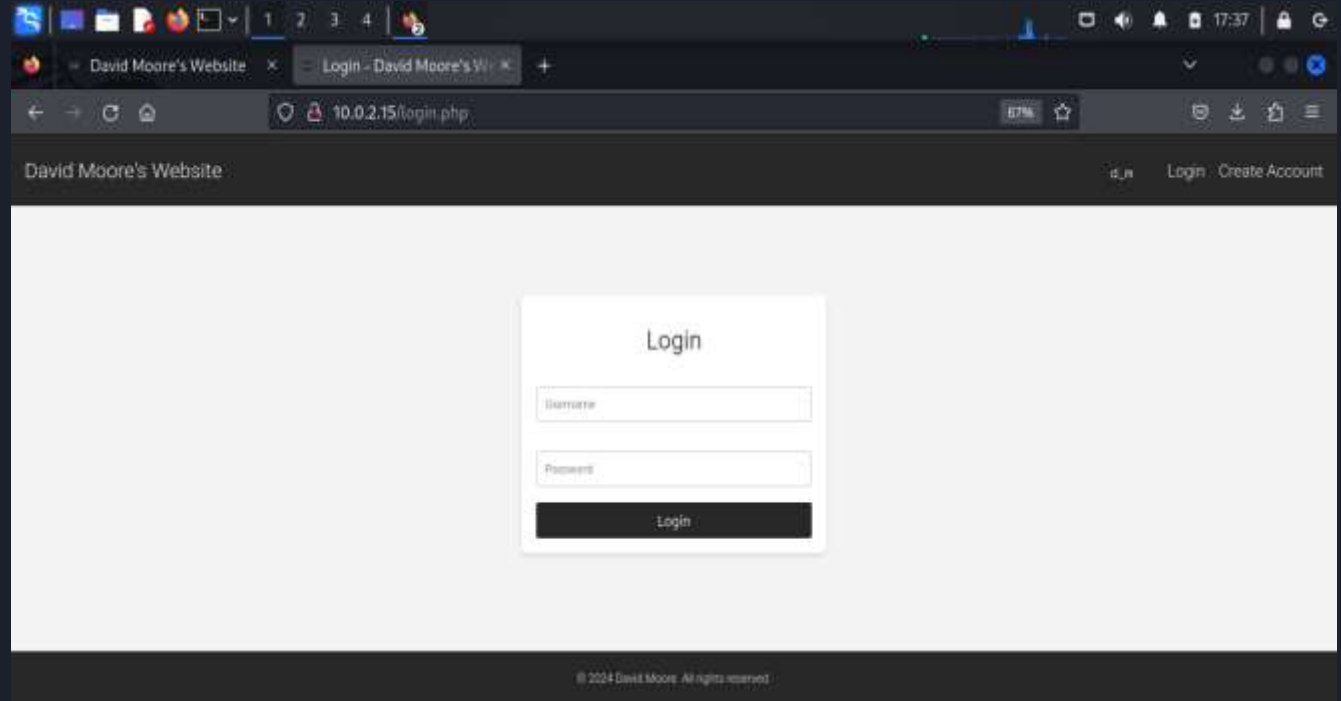
<main>
  <div class="login-container">
    <h1>Login</h1>
    <!-- Display any error messages -->
    <?php if (isset($error_message)): ?>
      <p class="message"><?php echo $error_message; ?></p>
    <?php endif; ?>

    <form action="login.php" method="POST">
      <input type="hidden" name="csrf_token" value="<?php echo generate_csrf_token(); ?>">
      <input type="text" name="username" placeholder="Username" required>
      <input type="password" name="password" placeholder="Password" required>
      <input type="submit" name="login" value="Login">
    </form>
  </div>
</main>

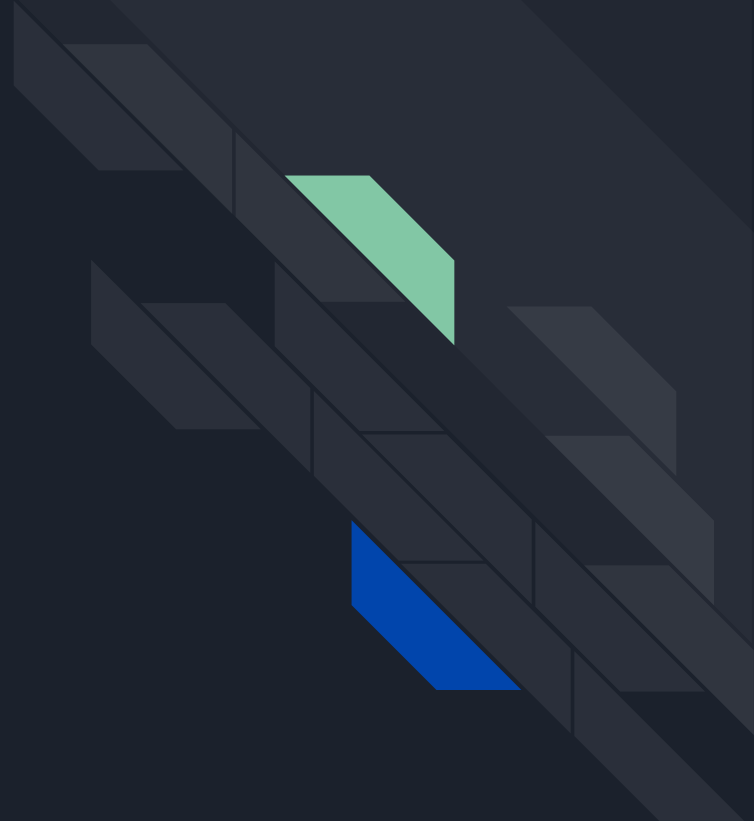
<footer>
  <p>&copy; 2024 David Moore. All rights reserved.</p>
</footer>
</body>
</html>
```

# PHP Webpage Screenshot

- Login container with input fields
- Header Bar with navigation links to keep the display uniform with the other pages



Profile.php page





# Profile Page Information

## Description:

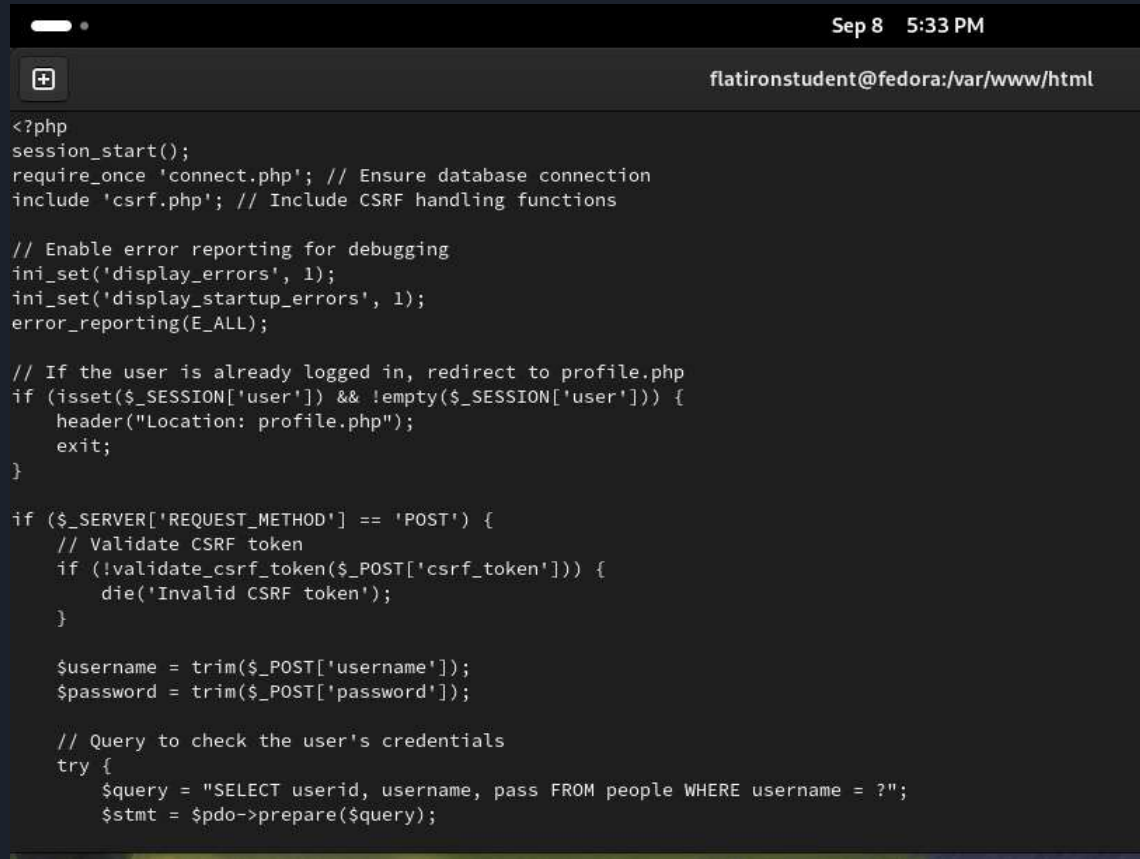
The profile.php page displays a user's profile information after they have logged in. It checks if a user session exists, and if not, redirects them to the login page. Once verified, the page retrieves and displays the user's first name, last name, and username from the database. The purpose of the page is to provide users with a personalized view of their profile and allow them to log out or navigate back to the homepage. Additionally, it includes security features like session regeneration to protect the user's session.

## Key Points about the Page:

- **User Authentication:** The page checks if a user is logged in by verifying the session, and redirects to the login page if no session exists.
- **Profile Information Display:** It retrieves and displays the logged-in user's profile details, such as their username, first name, and last name, from the database.
- **Security Measures:** The page includes session security by regenerating the session ID upon loading to prevent session hijacking.

# PHP Code Screenshot

- Session Check (if (!isset(\$\_SESSION['user']))), checks if a user session exists; if not, the user is redirected to the login page to prevent unauthorized access.
- Session Regeneration (session\_regenerate\_id(true);), regenerates the session ID to provide additional security by mitigating session fixation attacks.
- Fetching User Profile Information, prepares and executes an SQL query to fetch the logged-in user's profile information (username, first name, and last name) from the database using their session ID.
- User Validation (if (!\$userRow)), if no user data is found in the database, the user is logged out, which helps handle cases where the user session might be invalid.



```

Sep 8 5:33 PM
flatiironstudent@fedora:/var/www/html

<?php
session_start();
require_once 'connect.php'; // Ensure database connection
include 'csrf.php'; // Include CSRF handling functions

// Enable error reporting for debugging
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);

// If the user is already logged in, redirect to profile.php
if (isset($_SESSION['user']) && !empty($_SESSION['user'])) {
    header("Location: profile.php");
    exit;
}

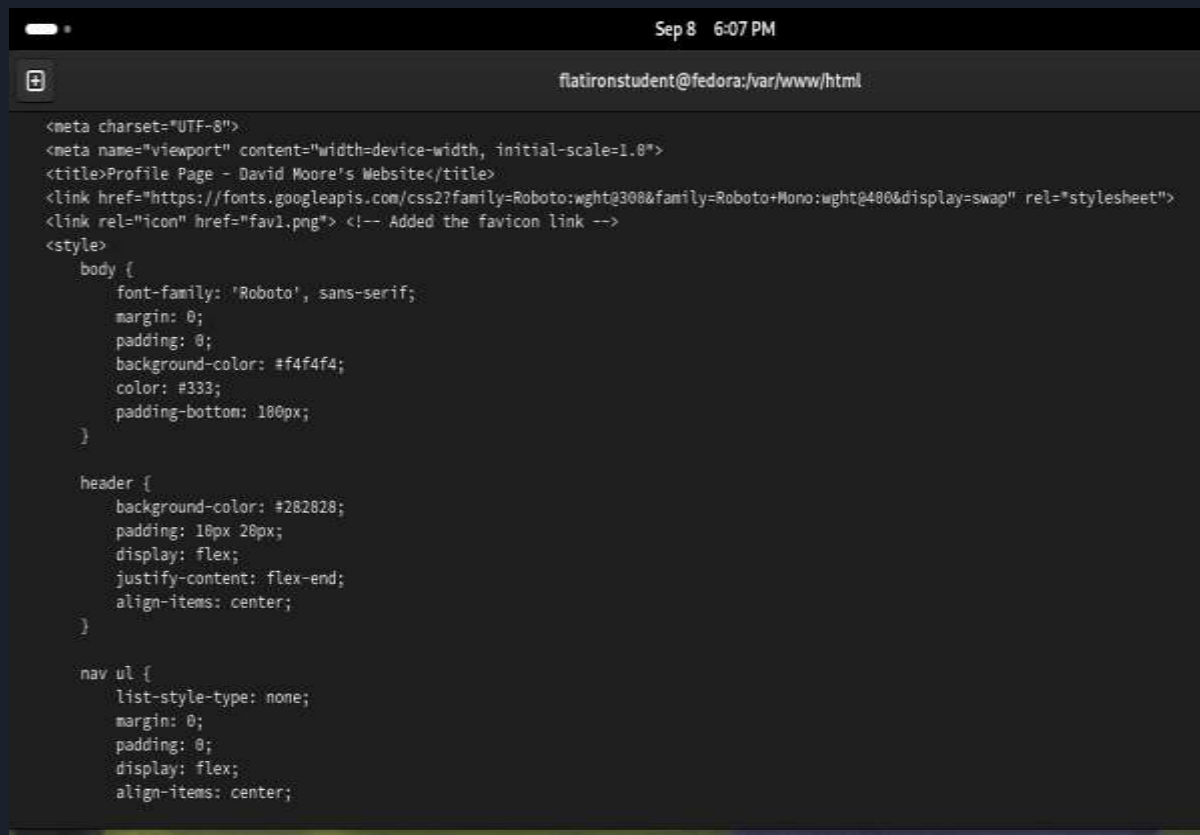
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Validate CSRF token
    if (!validate_csrf_token($_POST['csrf_token'])) {
        die('Invalid CSRF token');
    }

    $username = trim($_POST['username']);
    $password = trim($_POST['password']);

    // Query to check the user's credentials
    try {
        $query = "SELECT userid, username, pass FROM people WHERE username = ?";
        $stmt = $pdo->prepare($query);
    }
}
```

# PHP Code Screenshot

- Meta Viewport Tag to ensure responsiveness
- CSS continued



```

Sep 8 6:07 PM
flatironstudent@fedora:/var/www/html

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Profile Page - David Moore's Website</title>
<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300&family=Roboto+Mono:wght@400&display=swap" rel="stylesheet">
<link rel="icon" href="fav1.png"> <!-- Added the favicon link -->
<style>
  body {
    font-family: 'Roboto', sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    color: #333;
    padding-bottom: 100px;
  }

  header {
    background-color: #282828;
    padding: 10px 20px;
    display: flex;
    justify-content: flex-end;
    align-items: center;
  }

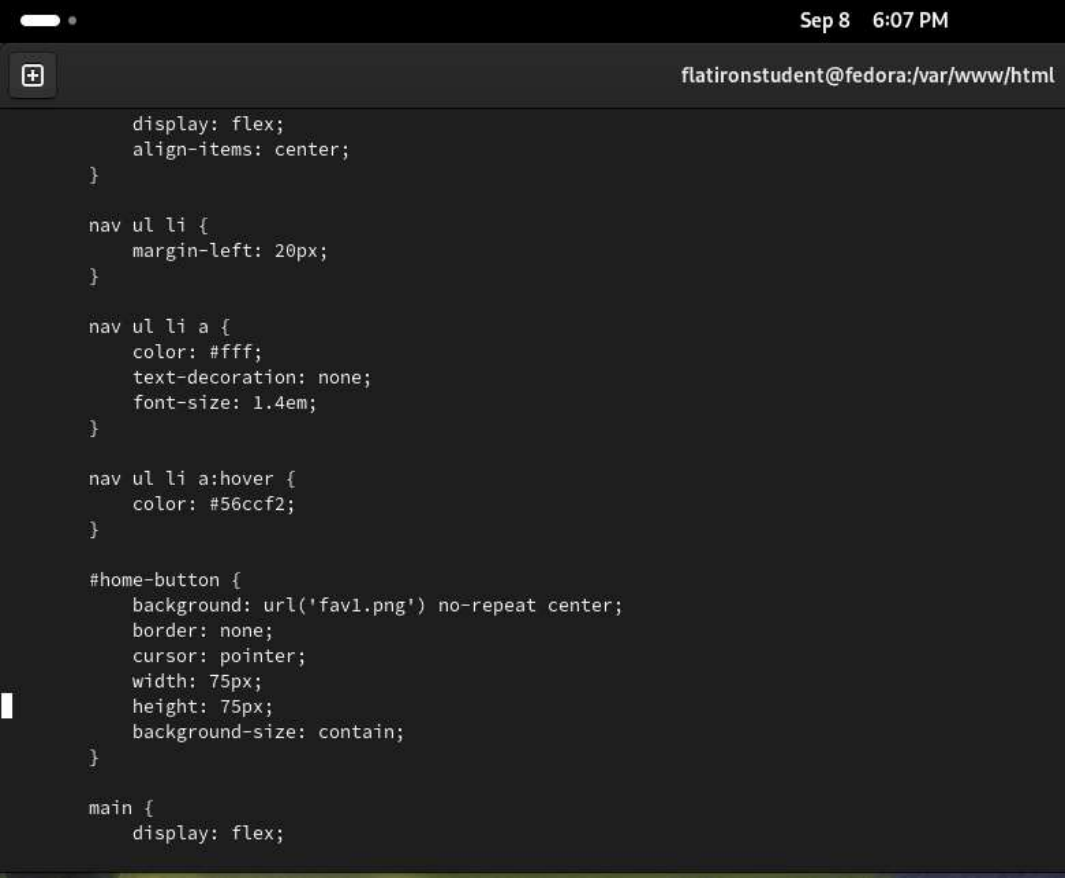
  nav ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    display: flex;
    align-items: center;
  }

```



# PHP Code Screenshot

- Navigation Styling
- Home Button Styling
- Main Section Styling



The screenshot shows a code editor window with a dark theme. The title bar at the top right displays 'Sep 8 6:07 PM' and the file path 'flatironstudent@fedora:/var/www/html'. The code is CSS, defining styles for a navigation bar and a home button. The navigation bar uses flexbox for layout. The home button is a square with a background image and a pointer cursor. The main section is also defined with flexbox.

```
display: flex;
align-items: center;
}

nav ul li {
margin-left: 20px;
}

nav ul li a {
color: #fff;
text-decoration: none;
font-size: 1.4em;
}

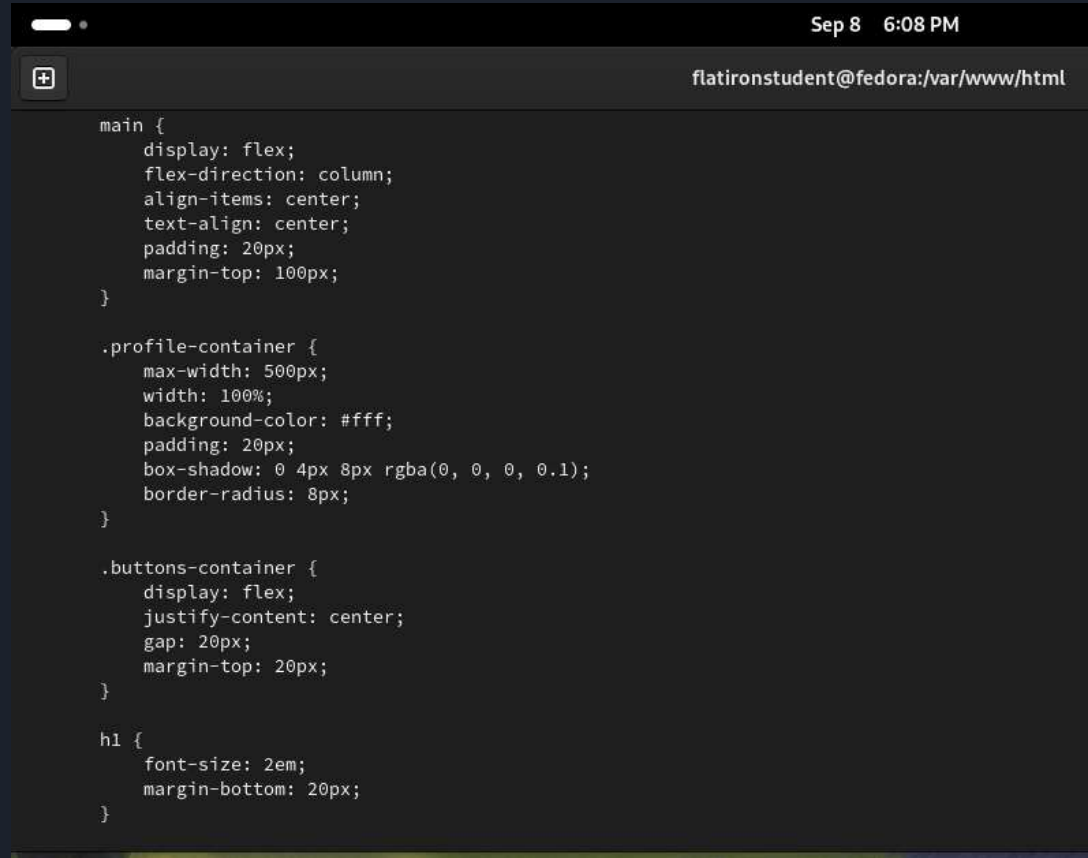
nav ul li a:hover {
color: #56ccf2;
}

#home-button {
background: url('fav1.png') no-repeat center;
border: none;
cursor: pointer;
width: 75px;
height: 75px;
background-size: contain;
}

main {
display: flex;
```

# PHP Code Screenshot

- Profile Container Styling
- Buttons Container Styling
- Heading Styling



The screenshot shows a code editor window with a dark theme. The title bar at the top right displays 'Sep 8 6:08 PM' and the file path 'flatironstudent@fedora:/var/www/html'. The code is CSS, defining styles for a main container, a profile container, a buttons container, and a heading.

```
main {
  display: flex;
  flex-direction: column;
  align-items: center;
  text-align: center;
  padding: 20px;
  margin-top: 100px;
}

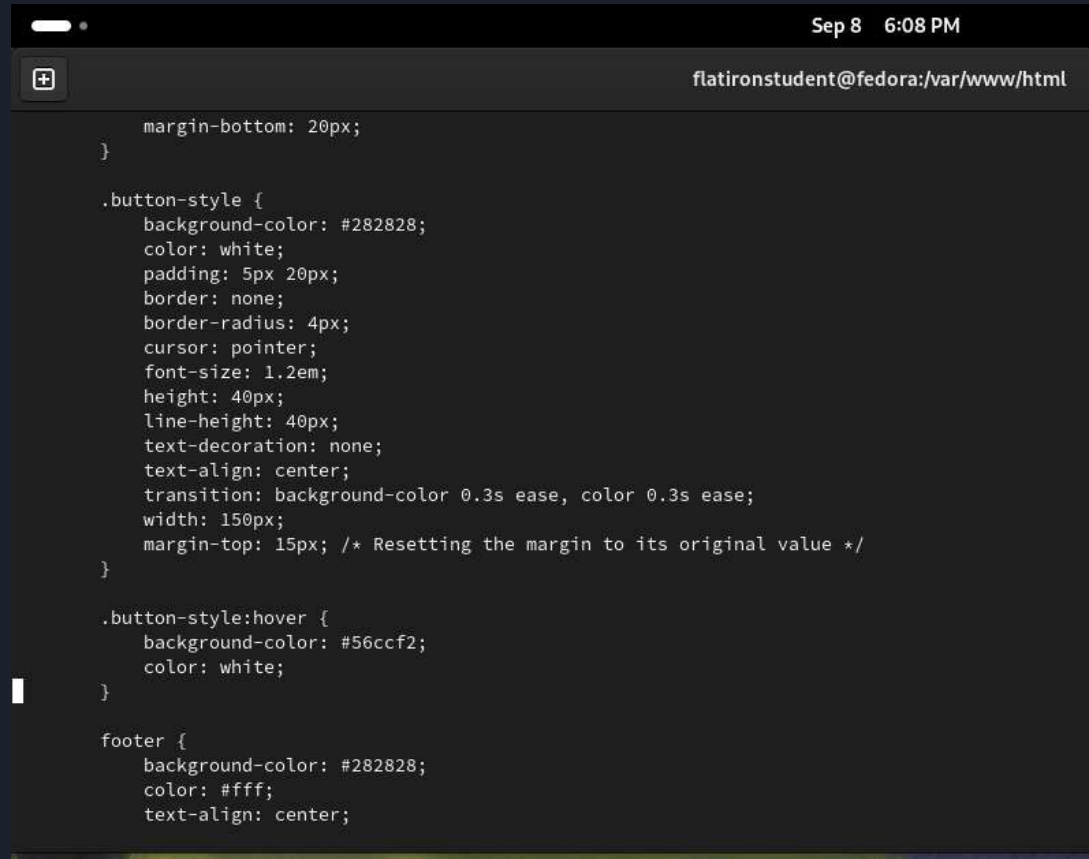
.profile-container {
  max-width: 500px;
  width: 100%;
  background-color: #fff;
  padding: 20px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  border-radius: 8px;
}

.buttons-container {
  display: flex;
  justify-content: center;
  gap: 20px;
  margin-top: 20px;
}

h1 {
  font-size: 2em;
  margin-bottom: 20px;
}
```

# PHP Code Screenshot

- Button Styling
- Footer Styling



The screenshot shows a code editor window with a dark theme. The title bar at the top right indicates the date and time as 'Sep 8 6:08 PM'. Below the title bar, the user's email 'flatironstudent@fedora:/var/www/html' is visible. The code is written in a light gray font on a dark background. It includes CSS rules for a button and a footer. The button rule is a class selector '.button-style' with various properties like background-color, color, padding, border, border-radius, cursor, font-size, height, line-height, text-decoration, text-align, transition, width, and margin-top. A comment is present for the margin-top property. The footer rule is a class selector 'footer' with background-color, color, and text-align properties.

```
margin-bottom: 20px;
}

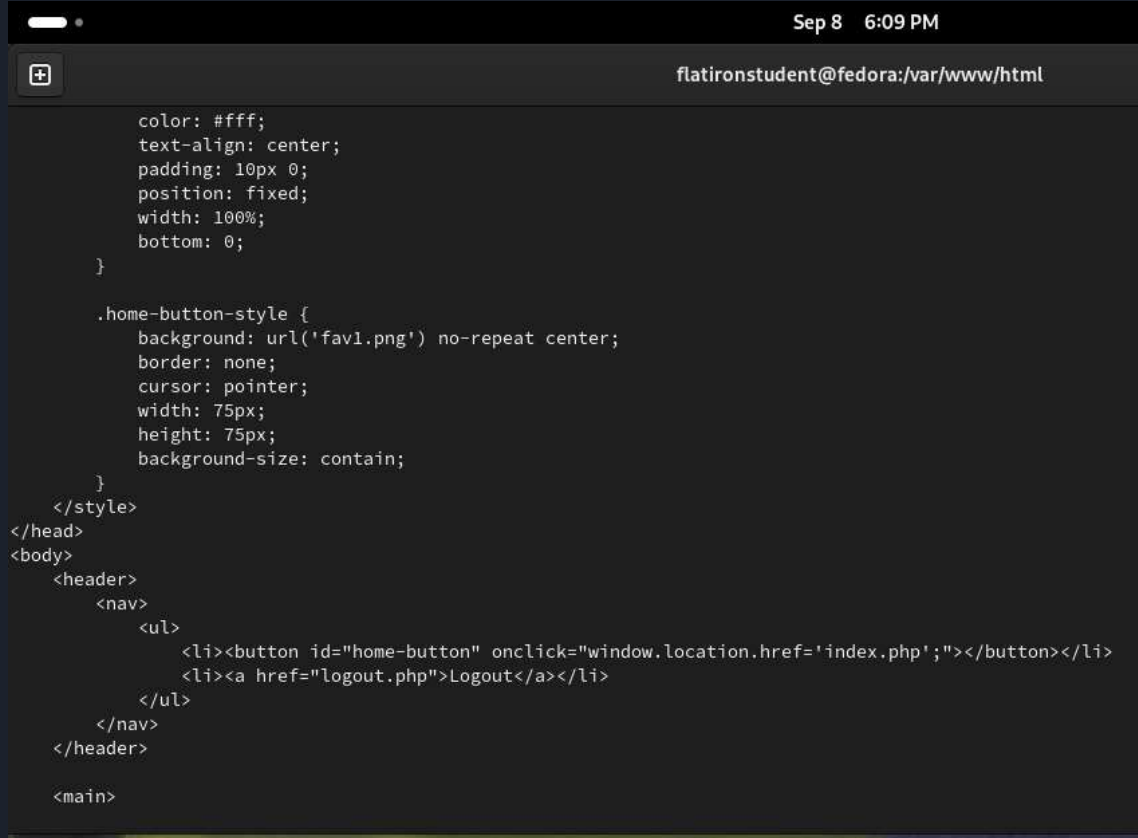
.button-style {
  background-color: #282828;
  color: white;
  padding: 5px 20px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size: 1.2em;
  height: 40px;
  line-height: 40px;
  text-decoration: none;
  text-align: center;
  transition: background-color 0.3s ease, color 0.3s ease;
  width: 150px;
  margin-top: 15px; /* Resetting the margin to its original value */
}

.button-style:hover {
  background-color: #56ccf2;
  color: white;
}

footer {
  background-color: #282828;
  color: #fff;
  text-align: center;
```

# PHP Code Screenshot

- Home Button Styling
- Logout Link (<a href="logout.php">Logout</a>), A clickable link that allows the user to log out of their account, ensuring easy navigation and session management.



```
color: #fff;
text-align: center;
padding: 10px 0;
position: fixed;
width: 100%;
bottom: 0;
}

.home-button-style {
background: url('fav1.png') no-repeat center;
border: none;
cursor: pointer;
width: 75px;
height: 75px;
background-size: contain;
}
</style>
</head>
<body>
  <header>
    <nav>
      <ul>
        <li><button id="home-button" onclick="window.location.href='index.php';"></button></li>
        <li><a href="logout.php">Logout</a></li>
      </ul>
    </nav>
  </header>

  <main>
```

# PHP Code Screenshot

- Header and Navigation
- Profile Container
- Greeting Heading (<h1>Welcome, <?php echo htmlspecialchars(\$userRow['fname']); ?>, outputs a personalized greeting with the user's first name, securely rendering the value to prevent XSS attacks using htmlspecialchars())
- User Information Display
- Buttons Container
- Footer

```

Sep 8 6:09 PM
flatironstudent@fedora:/var/www/html

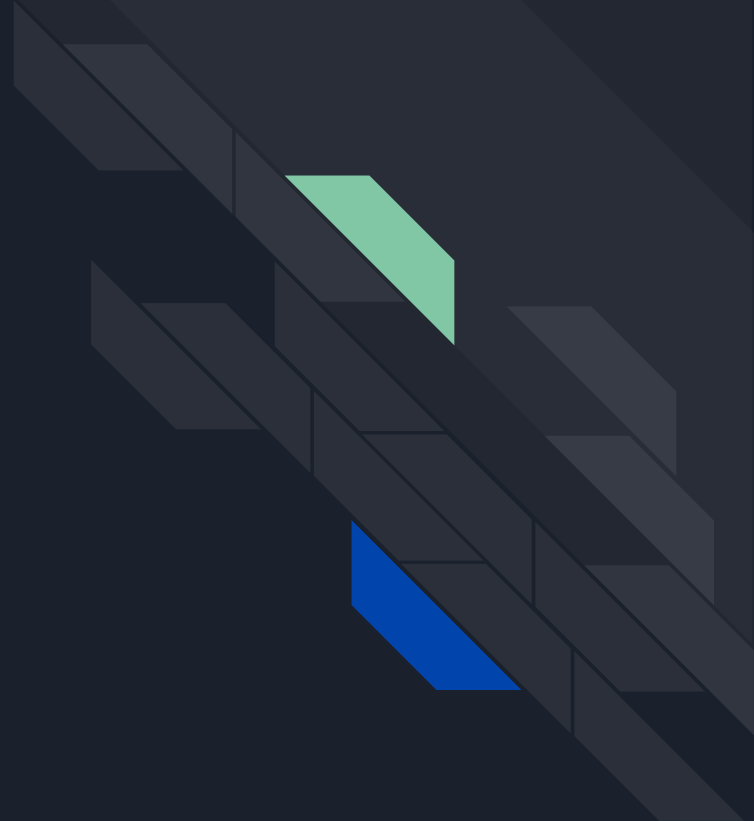
<header>
  <nav>
    <ul>
      <li><button id="home-button" onclick="window.location.href='index.php';"></button></li>
      <li><a href="logout.php">Logout</a></li>
    </ul>
  </nav>
</header>

<main>
  <div class="profile-container">
    <h1>Welcome, <?php echo htmlspecialchars($userRow['fname']); ?>!</h1>
    <p>Username: <?php echo htmlspecialchars($userRow['username']); ?></p>
    <p>First Name: <?php echo htmlspecialchars($userRow['fname']); ?></p>
    <p>Last Name: <?php echo htmlspecialchars($userRow['lname']); ?></p>

    <div class="buttons-container">
      <button class="home-button-style" onclick="window.location.href='index.php';"></button>
      <a href="logout.php" class="button-style">Logout</a>
    </div>
  </div>
</main>

<footer>
  <p>&copy; 2024 David Moore. All rights reserved.</p>
</footer>
</body>
</html>
```

Logout.php page





# Logout Page Information

## Description:

The `logout.php` page is responsible for securely logging the user out of their session. It begins by starting the session to ensure session data is accessible, then destroys all session data using `session_destroy()`. After clearing the session, it redirects the user to the login page (`login.php`) to allow them to log back in or remain logged out. This script ensures that all user data is properly cleared, and the user is redirected to a secure state after logging out.

## Key Points about the Page:

- **Session Termination:** The page uses `session_destroy()` to completely terminate the user's session, ensuring all session data is removed.
- **Redirection to Login:** After destroying the session, the user is immediately redirected to the login page, guiding them to a secure login state.
- **Security:** The page ensures that once the session is destroyed, the script execution stops, preventing any further actions after logging out.

# PHP Code Screenshot

- `session_start()`; Initiates or resumes the current session, ensuring session data is accessible before proceeding with the logout process.
- `session_unset()`; Unsets all session variables, clearing any data stored in the session without destroying the session itself.
- `session_destroy()`; Destroys the session, completely removing any session data, and ending the session for the user.
- `header("Location: login.php");` Redirects the user to the login page after the session is destroyed, ensuring they cannot access restricted pages without logging back in.
- `exit()`; Stops the script execution immediately after the redirection to ensure no further code is run after the session is destroyed.



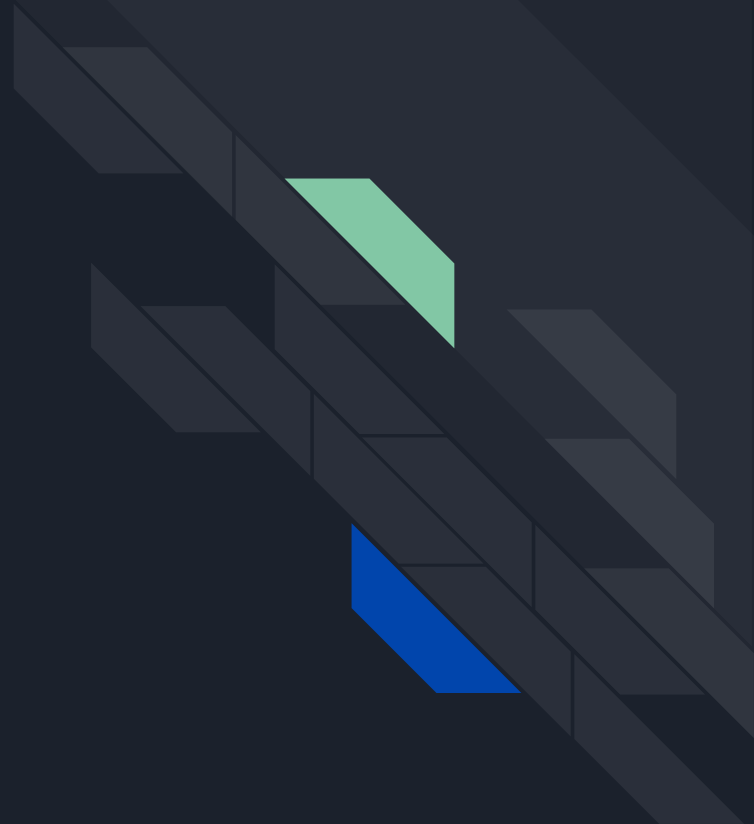
The screenshot shows a terminal window with a dark background. The title bar at the top indicates the date and time as 'Sep 8 6:37 PM' and the user as 'flatironstudent@fedora: /var/www/html'. The terminal content displays the following PHP code:

```
#!/usr/bin/php
session_start(); // Start the session
session_destroy(); // Destroy all session data
header("Location: login.php"); // Redirect to the login page
exit(); // Ensure the script stops executing after redirection
?>
```

At the bottom of the terminal, a status bar shows the file path '"logout.php" 7L, 219B' and the cursor position '1,1'.



# 404 Error Page





# 404 Error Page Page Information

## Description:

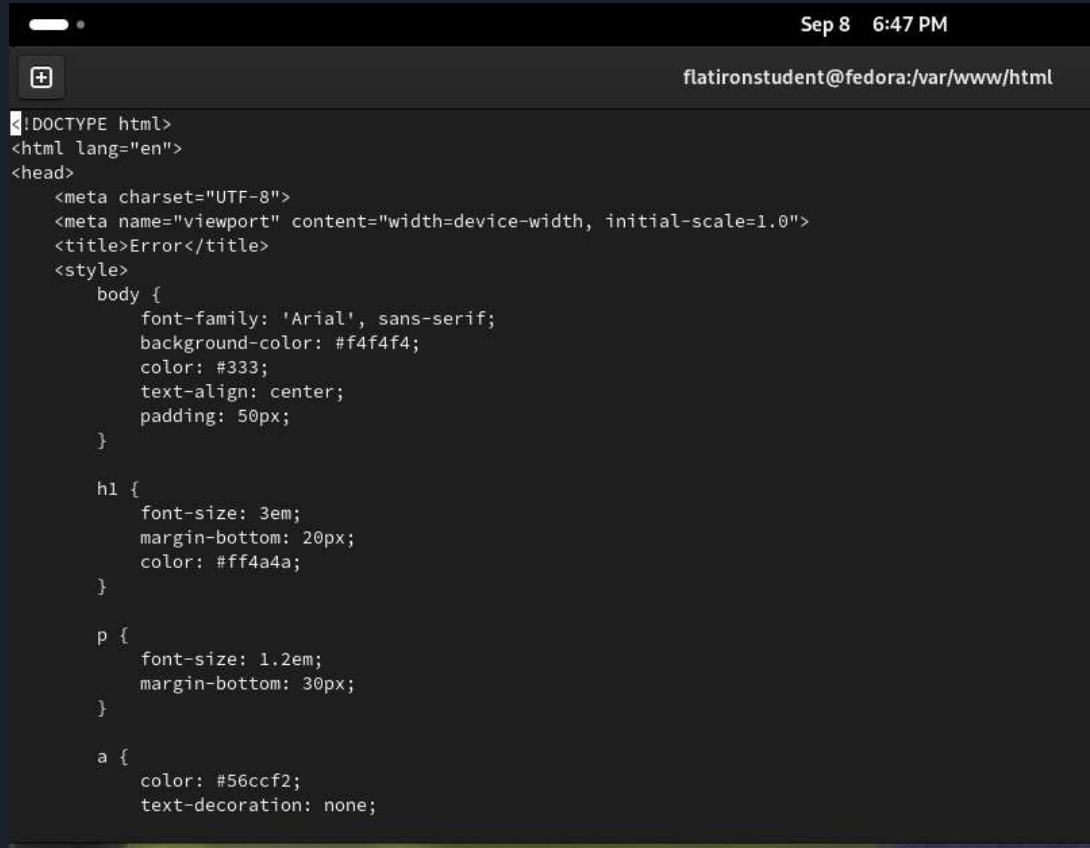
The 404.php page is a custom error page designed to handle "Page Not Found" (404) errors when users try to access a page that doesn't exist. It displays a clear "Error 404" message and provides a link for users to return to the homepage. The page is styled with a clean and simple design, using a light background and a red error message to signal the issue. The purpose of the page is to guide users back to the main site gracefully when they encounter an error, rather than displaying a generic server error message.

## Key Points about the Page:

- Error Handling
- User Navigation
- Simple and Clean Design

# PHP Code Screenshot

- Meta Viewport Tag
- Title Tag
- Body Styling
- Heading
- Paragraph Styling
- Link Styling



```
!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Error</title>
  <style>
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f4f4f4;
      color: #333;
      text-align: center;
      padding: 50px;
    }

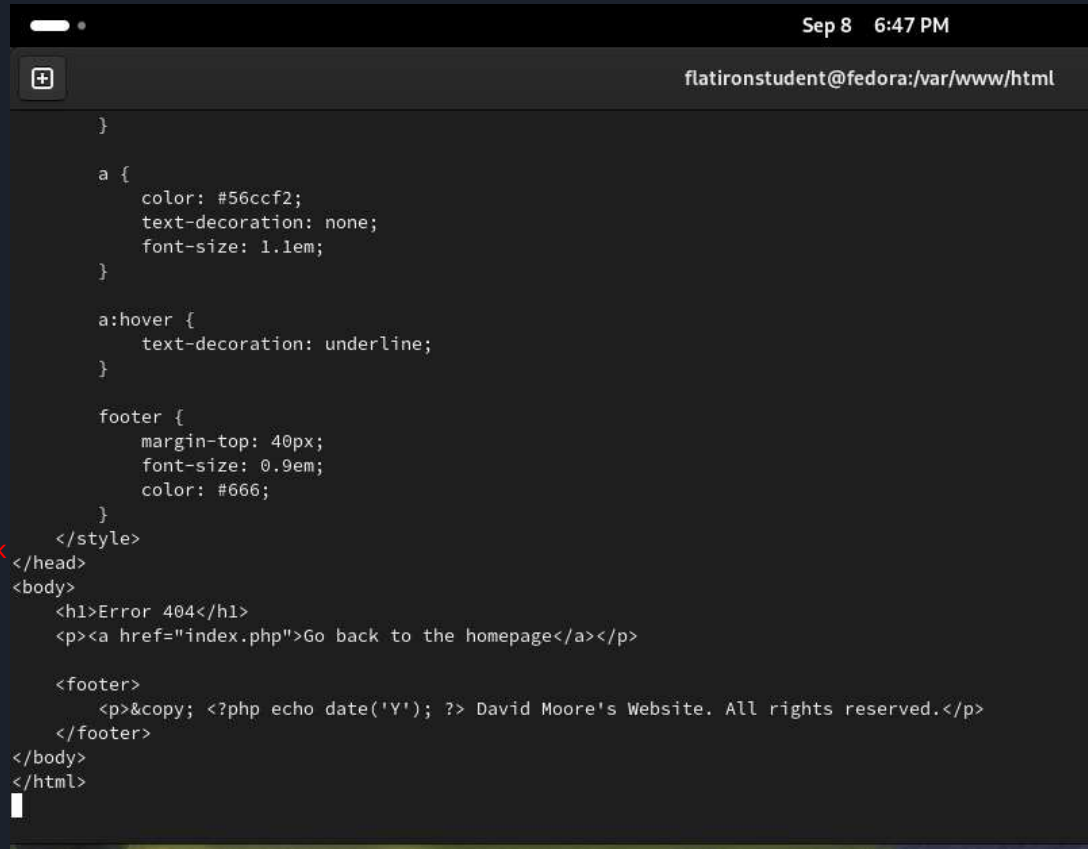
    h1 {
      font-size: 3em;
      margin-bottom: 20px;
      color: #ff4a4a;
    }

    p {
      font-size: 1.2em;
      margin-bottom: 30px;
    }

    a {
      color: #56ccf2;
      text-decoration: none;
    }
  </style>
</head>
<body>
  <h1>Error</h1>
  <p>Something went wrong. Please try again later.</p>
  <a href="#"></a>
</body>
</html>
```

# PHP Code Screenshot

- Link
- Link Hover Effect
- Footer Styling
- Error Message (<h1>Error 404</h1>), Displays a large, bold "Error 404" message at the top of the page, making it clear that the requested page was not found.
- Back to Homepage Link (<a href="index.php">...), Provides a clickable link that redirects the user back to the homepage, allowing them to navigate away from the error page.
- Dynamic Copyright in Footer (<?php echo date('Y'); ?>)



The screenshot shows a code editor window with a dark theme. The title bar at the top right displays 'Sep 8 6:47 PM' and the file path 'flatironstudent@fedora:/var/www/html'. The code is a mix of CSS and HTML. The CSS part defines styles for a link 'a' (color: #56ccf2, no decoration, 1.1em font size) and its hover state (underline decoration). It also styles a 'footer' with a top margin of 40px, 0.9em font size, and #666 color. The HTML part includes a head section with the CSS, a body section with a large 'Error 404' heading and a 'Go back to the homepage' link, and a footer section containing a dynamic copyright notice using PHP's date function.

```
}

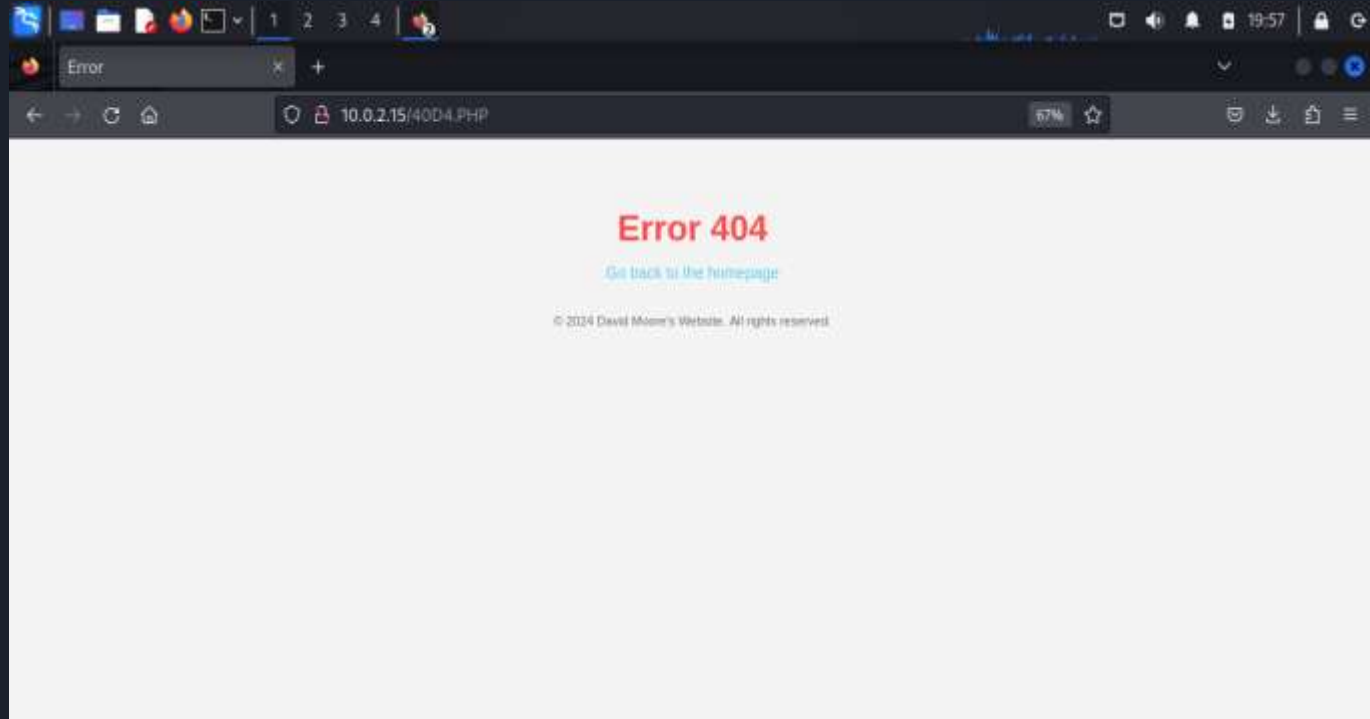
a {
  color: #56ccf2;
  text-decoration: none;
  font-size: 1.1em;
}

a:hover {
  text-decoration: underline;
}

footer {
  margin-top: 40px;
  font-size: 0.9em;
  color: #666;
}
}
</style>
</head>
<body>
  <h1>Error 404</h1>
  <p><a href="index.php">Go back to the homepage</a></p>

  <footer>
    <p>&copy; <?php echo date('Y'); ?> David Moore's Website. All rights reserved.</p>
  </footer>
</body>
</html>
```

# PHP Webpage Screenshot



- Error message
- Link back to homepage