**Batch: A3 Roll No.: 16010121045**

**Experiment No. 06**

|  |
| --- |
| **TITLE:** **Develop and Demonstrate the use of Form Handling and Validation in PHP** |

**AIM:** To develop web forms using PHP form and Validation.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected Outcome of Experiment:** Integrate server side pages using php.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. .

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Problem Statement: Description of the application implemented with output**:

To design and implement a blog site that allows multiple users to create and manage

their own blogs, while also providing functionality for users to view, search, and

comment on other blogs. The site will support CRUD operations for each entity, along

with the support for querying of blogs by title or content, and all blogs and comments

associated with a specific user and all comments associated with a specific blog.

Register.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

<meta *charset*="UTF-8">

<meta *http-equiv*="X-UA-Compatible" *content*="IE=edge">

<meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

<title>Daily Blogs</title>

*<!-- Stylesheet -->*

<link *rel*="stylesheet" *href*="../styles/index.css">

*<!-- Font Awesome -->*

<link *rel*="stylesheet" *href*="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.1/css/all.min.css" *integrity*="sha512-MV7K8+y+gLIBoVD59lQIYicR65iaqukzvf/nwasF0nqhPay5w/9lJmVM2hMDcnK1OnMGCdVK+iQrJ7lzPJQd1w==" *crossorigin*="anonymous" *referrerpolicy*="no-referrer" />

</head>

<body>

<div>

<div *class*="heading">

<h1>Register</h1>

</div>

<img *class*="waves" *src*="../images/waves.png" *alt*="Waves">

<div *class*="form">

<div *class*="greeting">

<h2>Welcome</h2>

<p>Let’s Register you in quickly</p>

</div>

<form *method*="POST" *action*="success.php">

<div *class*="input-field">

<i *class*="fa-solid fa-user"></i>

<label *for*="name" *class*="form\_\_label">Name</label>

<input *type*="text" *name*="name" *placeholder*="Enter Your Name" *required*>

</div>

<div *class*="input-field">

<i *class*="fa-regular fa-envelope fa-lg"></i>

<label *for*="email" *class*="form\_\_label">Email</label>

<input *type*="email" *name*="email" *placeholder*="Enter Your Email" *required*>

</div>

<div *class*="input-field">

<i *class*="fa-solid fa-lock fa-lg"></i>

<label *for*="password" *class*="form\_\_label">Password</label>

<input *type*="password" *name*="password" *placeholder*="Enter your password" *required*>

</div>

<div *class*="input-field">

<i *class*="fa-solid fa-lock fa-lg"></i>

<label *for*="password" *class*="form\_\_label">Confirm Password</label>

<input *type*="password" *name*="confirmpassword" *placeholder*="Confirm your password" *required*>

</div>

<button *id*="register" *class*="btn">Register</button>

<div *class*="option">

Already have an account?

<br>

Login

<a *href*="index.php">here</a>

</div>

</form>

</div>

</div>

<script *src*="../scripts/Validation.js"></script>

</body>

</html>

Validation.js

document.getElementById('register').addEventListener('click', validation);

function checkEmail(email) {

var re = /*^*\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+*$*/;

*if* (re.test(email)) *return* true;

*else* {

alert('Please enter a valid email!');

*return* false;

}

}

function checkPassword(password) {

var re = /*^*(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z])[0-9a-zA-Z]{8,}*$*/;

*if* (re.test(password)) *return* true;

*else* {

alert('Please enter a strong password!');

*return* false;

}

}

function checkPasswordMatch() {

var password = document.getElementsByName('password')[0].value;

var confirmPassword = document.getElementsByName('confirmpassword')[0].value;

*if* (password !== confirmPassword) {

alert('Passwords do not match.');

*return* false;

}

*return* true;

}

function validation(event) {

var email = document.getElementsByName('email')[0].value;

var password = document.getElementsByName('password')[0].value;

*if* (checkEmail(email) && checkPassword(password) && checkPasswordMatch()) {

alert('Account Created Successfully');

*return* true;

} *else* {

event.preventDefault();

*return* false;

}

}

**PHP Basic Concepts Learned With Syntax**

Authentication.php

<?php

*//connect to database*

*include*('../scripts/connection.php');

*//start the session*

session\_start();

*//get data from form*

$email = $\_POST["email"];

$pass = $\_POST["password"];

*//to prevent from mysqli injection*

$email = stripcslashes($email);

$pass = stripcslashes($pass);

$email = mysqli\_real\_escape\_string($con, $email);

$pass = mysqli\_real\_escape\_string($con, $pass);

*//queries to get the password and the userid*

$query = "SELECT pass FROM users WHERE email='$email'";

$query2 = "SELECT userid FROM users WHERE email='$email'";

$result = $conn->query($query);

$result2 = $conn->query($query2);

*//check if the email exists*

*if* ($result->num\_rows > 0) {

*//get the password and the userid from the result*

$row = $result->fetch\_assoc();

$row2 = $result2->fetch\_assoc();

$user\_id = $row2["userid"];

$hashed\_password = $row["pass"];

*//verify the password*

*if* (password\_verify($pass, $hashed\_password)) {

*//set the session variable*

$\_SESSION['user\_id'] = $user\_id;

*//navigate to home page*

header("Location: ../pages/home.php");

*exit*();

} *else* {

*//stay on the index page*

header("Location: ../pages/index.php");

}

} *else* {

*//stay on the index page*

header("Location: ../pages/index.php");

}

?>

<!DOCTYPE *html*>

<html *lang*="en">

<body>

//NO NEED FOR HTML HERE

</body>

</html>

Database Connection: The script connects to a MySQL database using mysqli\_connect() function in the connection.php file.

Sessions: The session\_start() function is used to start a new or resume an existing session.

Form Data Handling: The code retrieves the email and password entered in the login form using the $\_POST superglobal array.

SQL Query Execution: The code executes SQL queries to retrieve the hashed password and user id from the database based on the email entered in the login form.

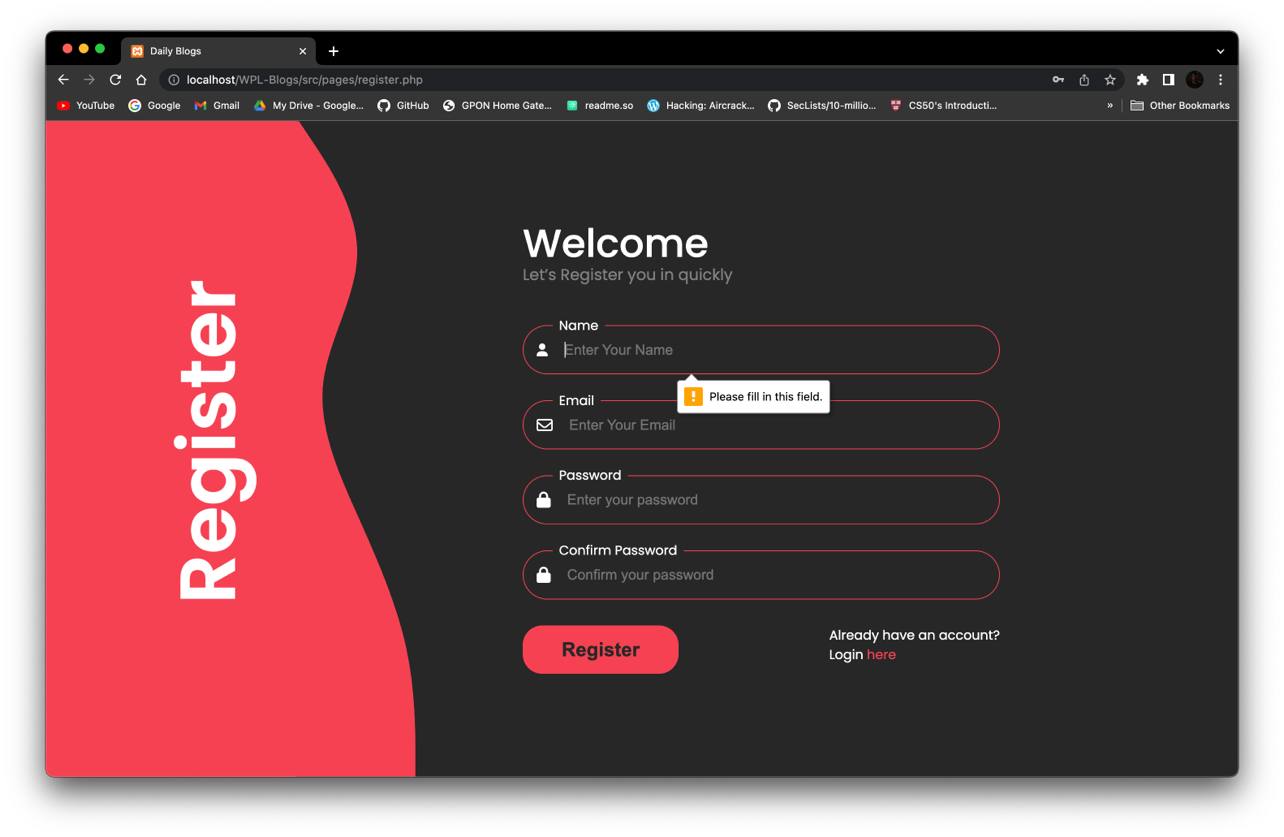
Password Verification: The code uses the password\_verify() function to compare the password entered in the login form with the hashed password retrieved from the database.

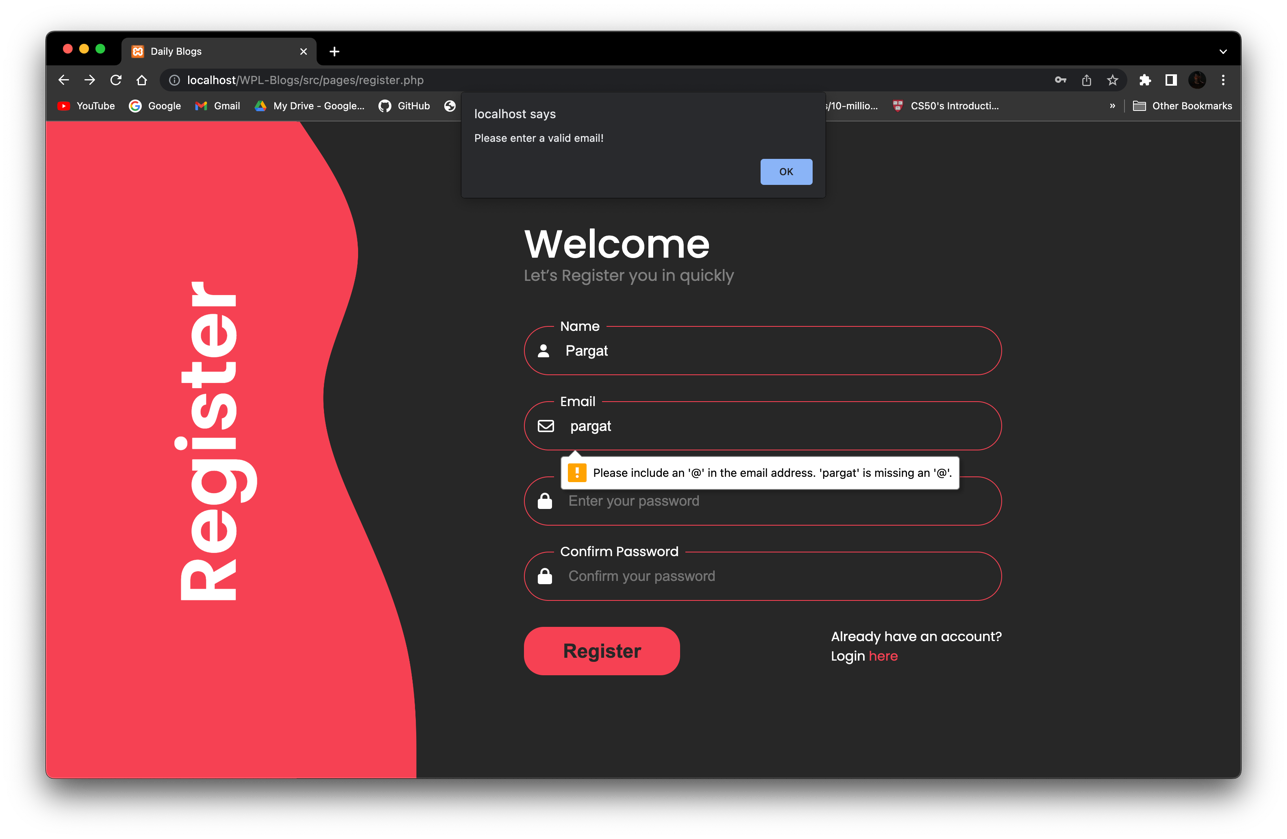
HTTP Headers: The code uses the header() function to navigate to the home page if the login is successful, or stay on the index page if the login fails.

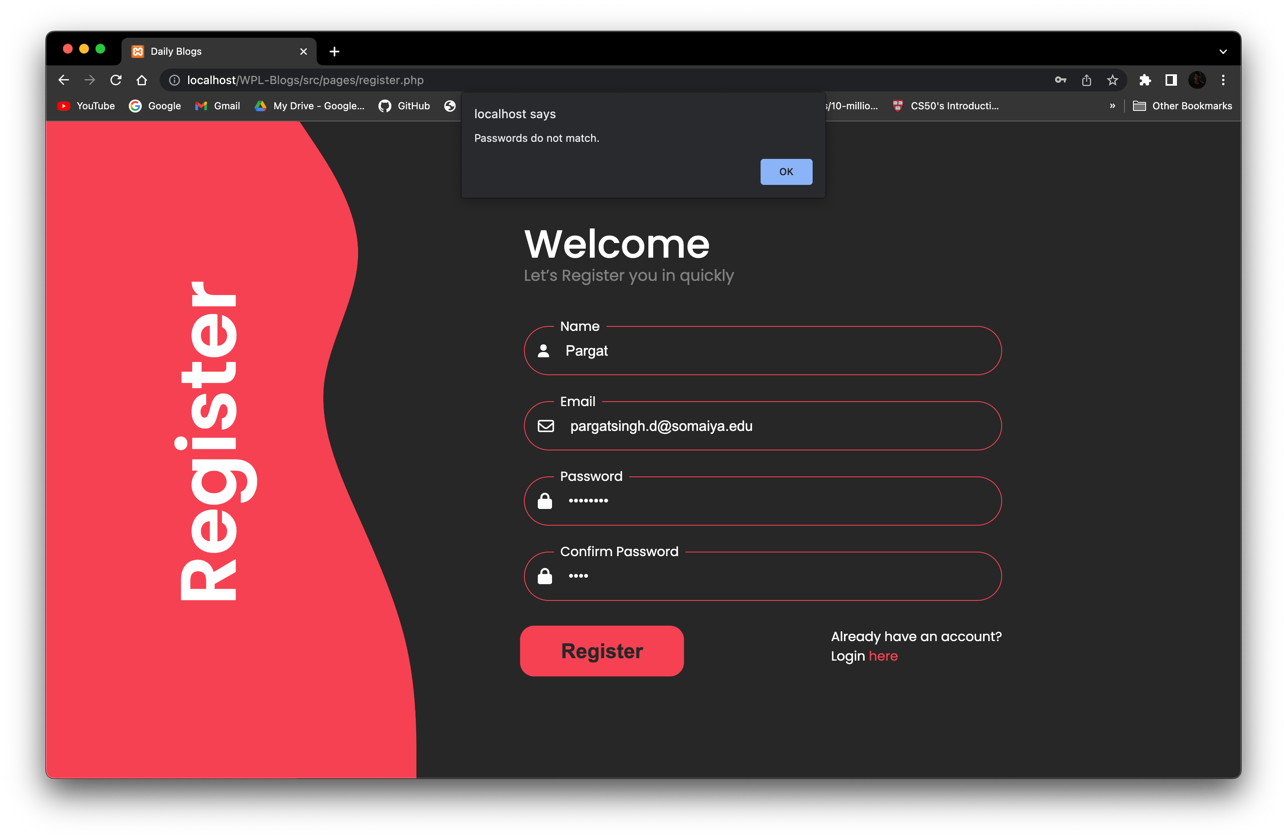
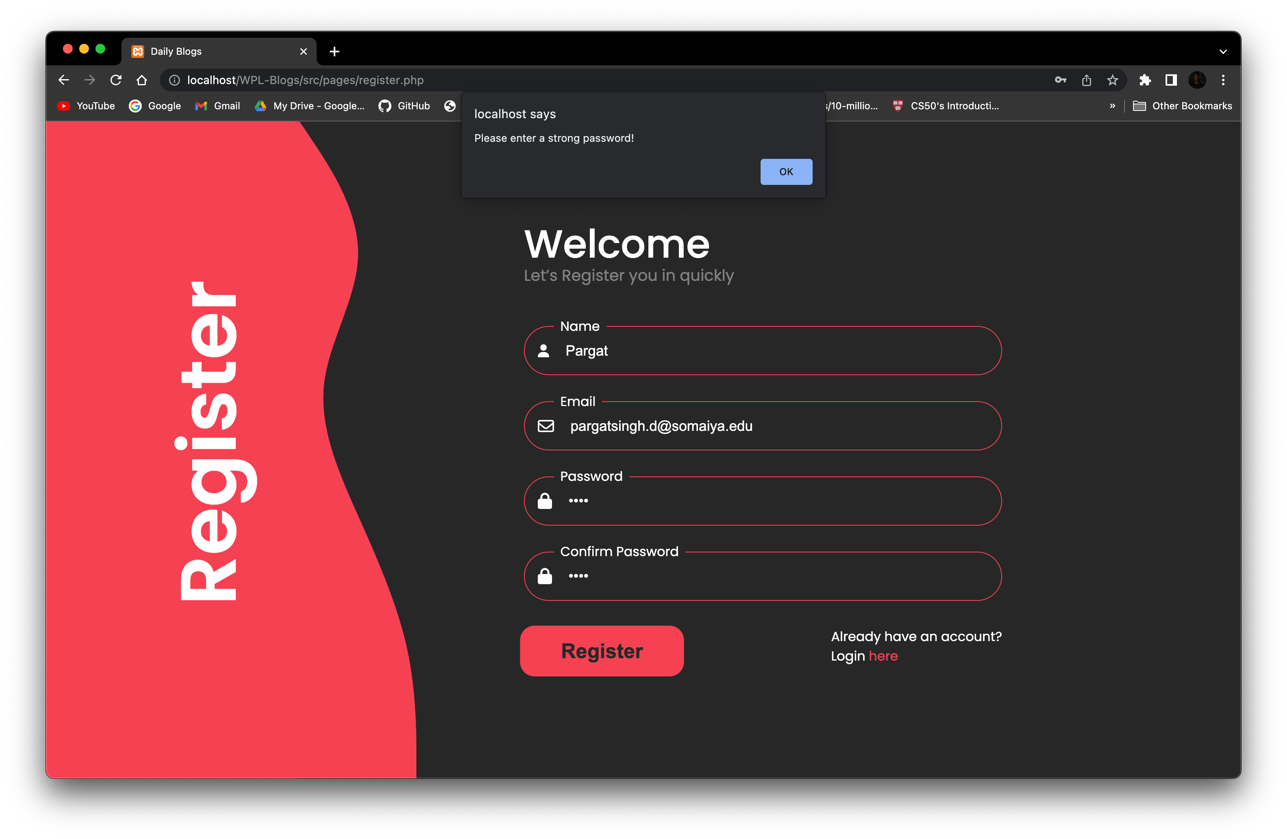
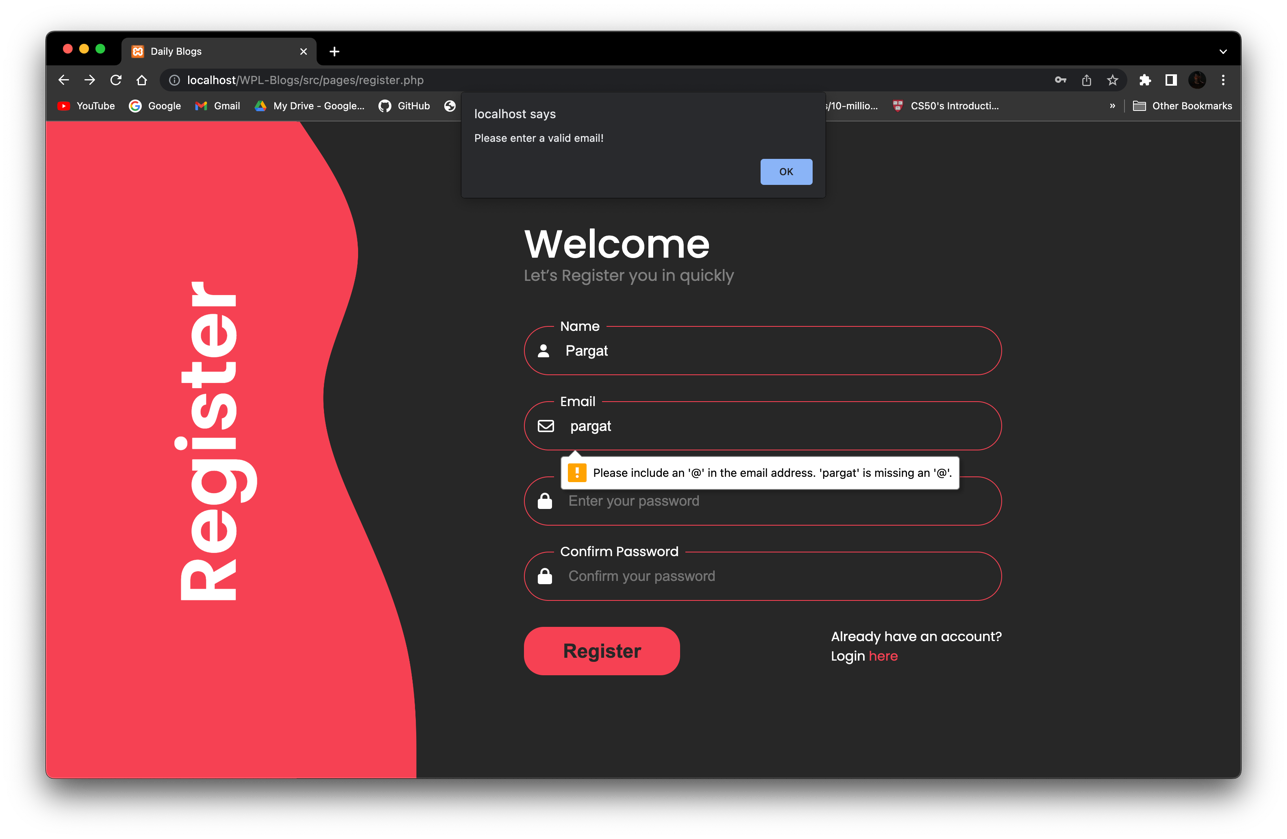
Super Global Arrays: The code uses the $\_SESSION and $\_POST superglobal arrays to store the user ID and retrieve the form data respectively.

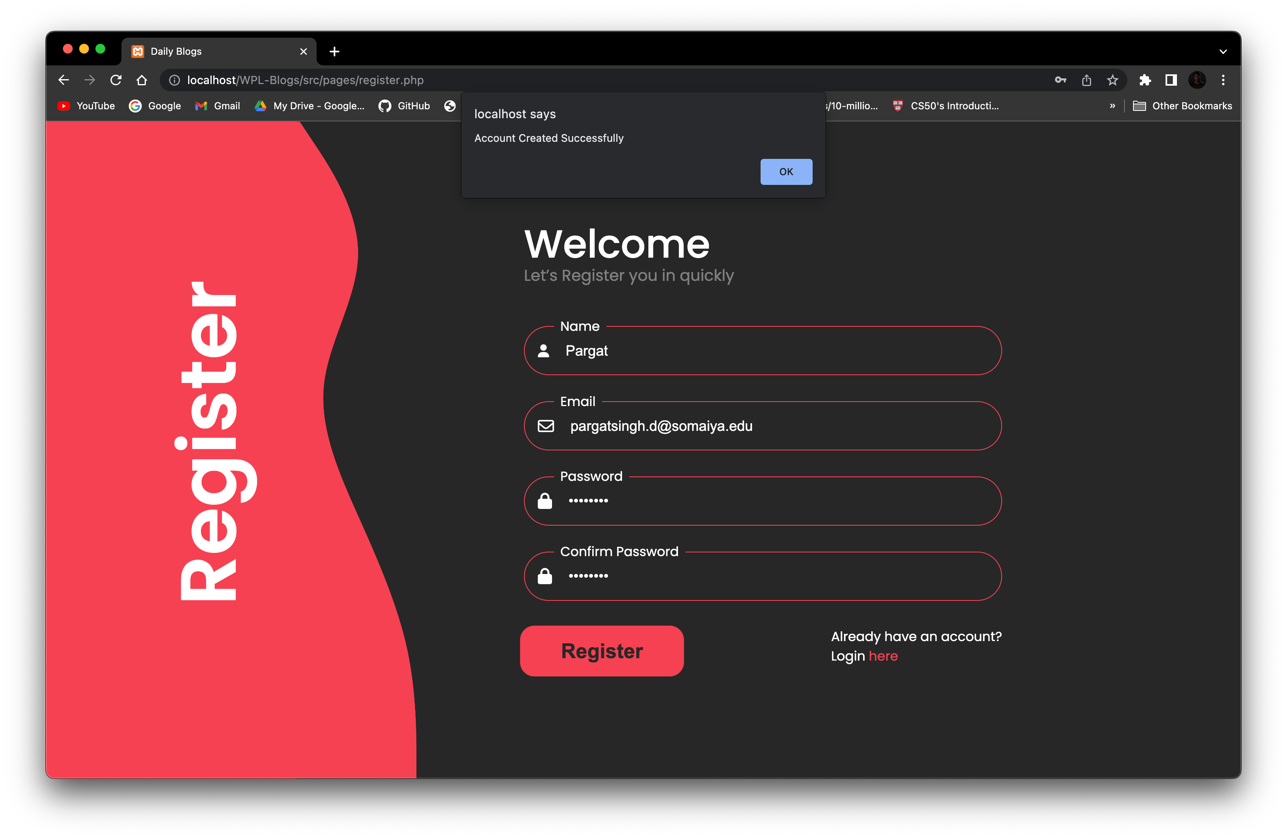
Escaping Strings: The code uses the mysqli\_real\_escape\_string() function to escape special characters in the email and password entered in the login form, in order to prevent SQL injection attacks.

**Description of the application implemented with output**:

****

****

****

****

In this experiment we implement php form validation in our sign up form and

apply it in our problem statement website.

**Post Lab Objective with Ans :**

**What is PHP Form?**

A PHP form is an HTML form that is processed by PHP on the server-side. It allows users to input data or make selections and then send that data to a server for processing or storage. PHP forms can be used for a wide variety of purposes, such as logging in to a website, registering for a service, submitting feedback, or making a purchase.

When a user submits a PHP form, the data is typically sent to a PHP script on the server, which can then process the data, validate it, and store it in a database or perform other operations based on the form data.

**What is PHP Form Validation?**

A form contains various input fields such as text box, checkbox, radio buttons,

submit button, and checklist, etc. These input fields need to be validated, which ensures

that the user has entered information in all the required fields and also validates that the

information provided by the user is valid and correct.

There is no guarantee that the information provided by the user is always correct. PHP

validates the data at the server-side, which is submitted by HTML form. You need to

validate a few things:

• Empty String

• Validate String

• Validate Numbers

• Validate Email

• Validate URL

• Input length

**List different inbuilt functions used in PHP form and validation**

In PHP, there are various inbuilt functions available for handling form data and validating user inputs. Some of the commonly used functions are:

$\_GET - used to retrieve form data sent using the GET method

$\_POST - used to retrieve form data sent using the POST method

htmlspecialchars() - used to convert special characters to HTML entities to prevent XSS attacks

strip\_tags() - used to strip HTML and PHP tags from a string

filter\_var() - used to filter a variable with a specified filter

preg\_match() - used for regular expression pattern matching

empty() - used to check if a variable is empty or not

isset() - used to check if a variable is set and not null

strlen() - used to get the length of a string

is\_numeric() - used to check if a value is numeric or not

ctype\_alpha() - used to check if a string contains only alphabetic characters

ctype\_digit() - used to check if a string contains only digits

password\_hash() - used to hash a password

password\_verify() - used to verify a password against a hash

These functions are just a few examples of the various built-in functions available in PHP for handling form data and validating user inputs.