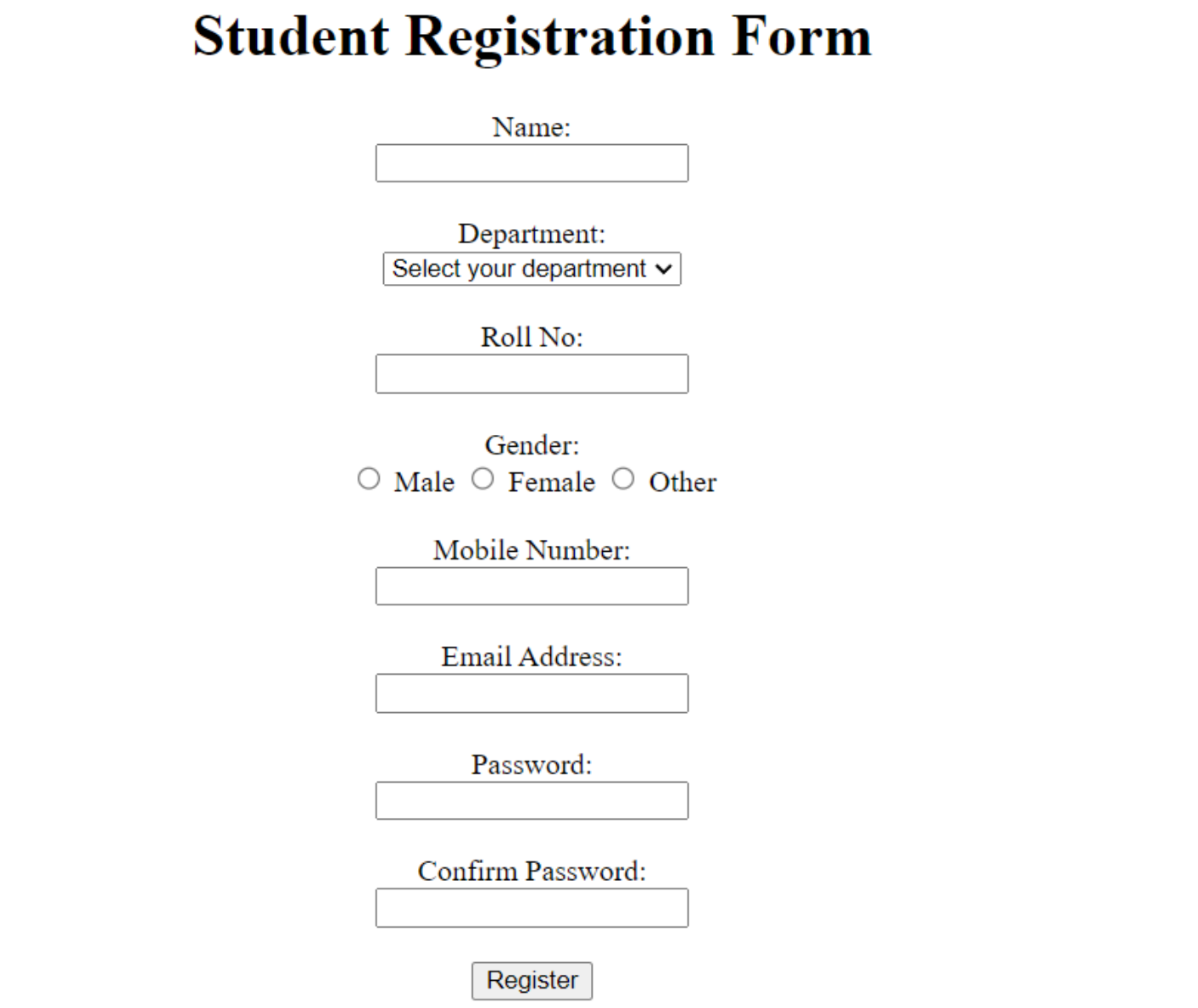
EXPERIMENT NO. 1

Title: Create html pages for website like login, registration and about us pages.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Registration Form</title>
</head>
<body>
    <center>
        <h1>Student Registration Form</h1>
        <form action="#" method="post">
            <div>
                <label for="first-name">Name:</label><br>
                <input type="text" id="first-name" name="first-name" required>
            </div>
            <br>
            <div>
                <label for="department">Department:</label><br>
                <select id="department" name="department" required>
                    <option value="" disabled selected>Select your department</option>
                    <option value="Computer-Science">Computer Science</option>
                    <option value="Electrical">Electrical</option>
                    <option value="Mechanical">Mechanical</option>
                    <option value="Civil">Civil</option>
                </select>
            </div>
            <br>
            <div>
                <label for="roll-no">Roll No:</label><br>
                <input type="text" id="roll-no" name="roll-no" required>
            </div>
            <br>
            <div>
                <label>Gender:</label><br>
                <input type="radio" id="male" name="gender" value="male">
                <label for="male">Male</label>
                <input type="radio" id="female" name="gender" value="female">
                <label for="female">Female</label>
                <input type="radio" id="other" name="gender" value="other">
                <label for="other">Other</label>
            </div>
            <br>
            <div>
                <label for="mobile-number">Mobile Number:</label><br>
                <input type="tel" id="mobile-number" name="mobile-number" required>
            </div>
            <br>
            <div>
                <label for="email">Email Address:</label><br>
                <input type="email" id="email" name="email" required>
            </div>
            <br>
            <div>
```

```
            <label for="password">Password:</label><br>
            <input type="password" id="password" name="password" required>
        </div>
        <br>
        <div>
            <label for="password">Confirm Password:</label><br>
            <input type="password" id="password" name="password" required>
        </div>
        <br>
        <div>
            <button type="submit">Register</button>
        </div>
    </form>
</center>
</body>
</html>
```

OUTPUT



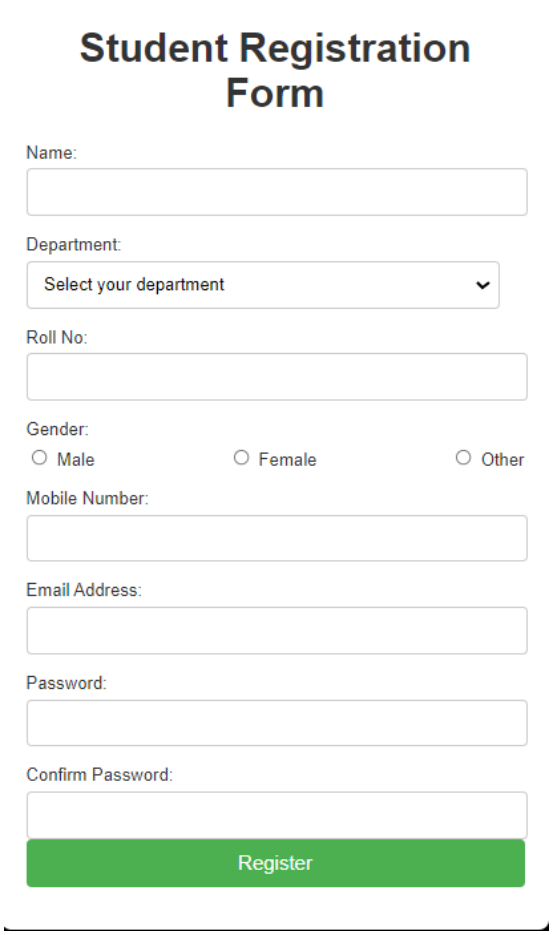
Experiment No. 2

Title : Design web pages using CSS.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Student Registration Form</title>

<style>
body {font-family: Arial, sans-serif;margin: 0;padding: 0;display: flex;justify-content:
center;align-items: center;height: 100vh;background-color: #000000;}
form {width: 100%;max-width: 400px;background: #fff;padding: 20px;border-radius: 8px;
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);}
h1 {text-align: center;color: #333;}
.form-group {margin-bottom: 15px;}
label {display: block;font-size: 14px;margin-bottom: 5px;color: #333;}
input, select {width: 95%;padding: 10px;font-size: 14px;border: 1px solid #ccc;borderradius:
4px;}
input[type="radio"] {width: auto;}
.form-group-inline {display: flex;justify-content: space-between;align-items: center;}
.form-group-inline label {margin: 0;}
button {width: 100%;padding: 10px;background-color: #4CAF50;color: white;font-size:
16px;border: none;border-radius: 4px;cursor: pointer;}
button:hover {background-color: #45a049;}
@media (max-width: 500px) {
form {padding: 15px;}
input, select, button {font-size: 13px;}}
</style>

</head>
<body>
<form action="#" method="post">
<h1>Student Registration Form</h1>
<div class="form-group">
<label for="first-name">Name:</label>
<input type="text" id="first-name" name="first-name" required>
</div>
<div class="form-group">
<label for="department">Department:</label>
<select id="department" name="department" required>
<option value="" disabled selected>Select your department</option>
<option value="Computer-Science">Computer Science</option>
<option value="Electrical">Electrical</option>
<option value="Mechanical">Mechanical</option>
<option value="Civil">Civil</option>
</select>
</div>
<div class="form-group">
<label for="roll-no">Roll No:</label>
<input type="text" id="roll-no" name="roll-no" required>
</div>
<div class="form-group">
<label>Gender:</label>
<div class="form-group-inline"><label for="male"><input type="radio" id="male" name="gender"
```

```
value="male">
Male</label>
<label for="female"><input type="radio" id="female" name="gender" value="female"
> Female</label>
<label for="other"><input type="radio" id="other" name="gender" value="other">
Other</label>
</div>
</div>
<div class="form-group">
<label for="mobile-number">Mobile Number:</label>
<input type="tel" id="mobile-number" name="mobile-number" required>
</div>
<div class="form-group">
<label for="email">Email Address:</label>
<input type="email" id="email" name="email" required>
</div>
<div class="form-group">
<label for="password">Password:</label>
<input type="password" id="password" name="password" required>
</div>
<div class="form-group"></div>
<label for="password">Confirm Password:</label>
<input type="password" id="password" name="password" required>
</div>
<div class="form-group">
<button type="submit">Register</button>
</div>
</html>
```

OUTPUT



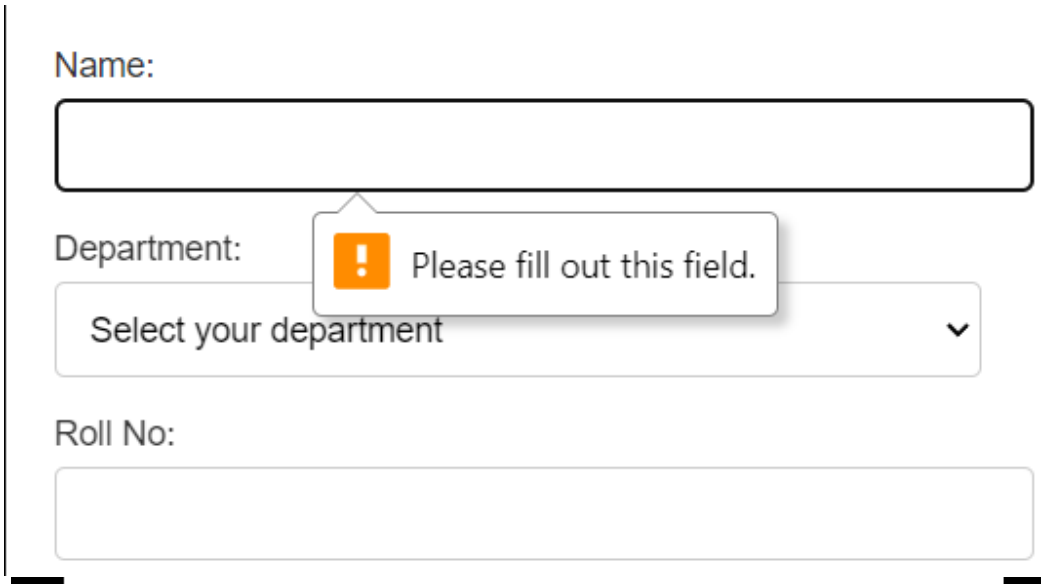
Experiment No. 3

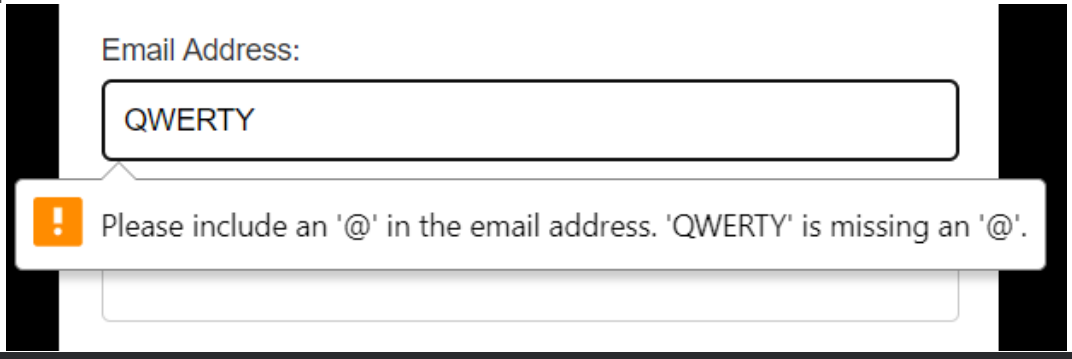
Title: Write a program demonstrating javascript functions and different validations

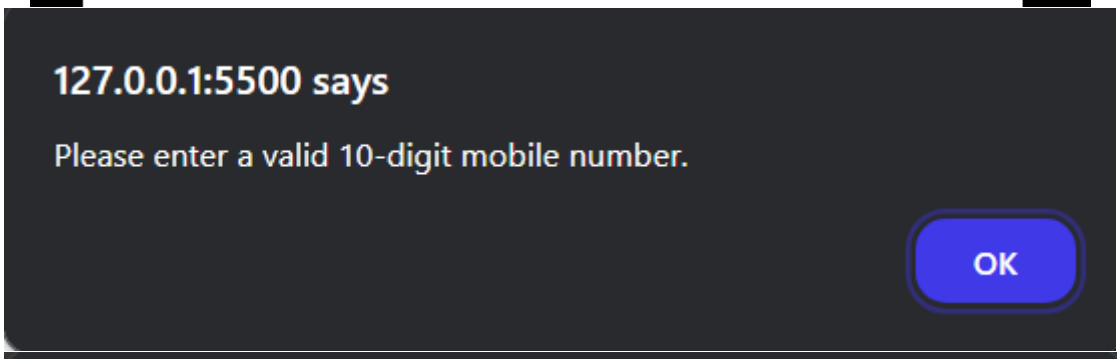
```
<!DOCTYPE html>
<html lang=”en”>
<head>
    <meta charset=”UTF-8”>
    <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>
    <title>Student Registration Form</title>
    <style>
        body {font-family: Arial, sans-serif;margin: 0;padding: 0;display: flex;justify-content:
center;align-items: center;height: 100vh;background-color: #000000;}
        form {width: 100%;max-width: 400px;background: #fff;padding: 20px;border-radius: 8px;box-
shadow: 0 4px 8px rgba(0, 0, 0, 0.1);}
        h1 {text-align: center;color: #333;}
        .form-group {margin-bottom: 15px;}
        label {display: block;font-size: 14px;margin-bottom: 5px;color: #333;}
        input, select {width: 95%;padding: 10px;font-size: 14px;border: 1px solid #ccc;border-
radius: 4px;}
        input[type=”radio”] {width: auto;}
        .form-group-inline {display: flex;justify-content: space-between;align-items: center;}
        .form-group-inline label {margin: 0;}
        button {width: 100%;padding: 10px;background-color: #4CAF50;color: white;font-size:
16px;border: none;border-radius: 4px;cursor: pointer;}
        button:hover {background-color: #45a049;}
        @media (max-width: 500px) {
            form {padding: 15px;}
            input, select, button {font-size: 13px;}}
    </style>
    <script>
        function validateForm() {
            // Get form elements
            var firstName = document.getElementById(‘first-name’).value;
            var department = document.getElementById(‘department’).value;
            var rollNo = document.getElementById(‘roll-no’).value;
            var gender = document.querySelector(‘input[name=”gender”]:checked’);
            var mobileNumber = document.getElementById(‘mobile-number’).value;
            var email = document.getElementById(‘email’).value;
            var password = document.getElementById(‘password’).value;
            var confirmPassword = document.getElementById(‘confirm-password’).value;
            if (!firstName || !department || !rollNo || !mobileNumber || !email || !password ||
!confirmPassword) {
                alert(“All fields must be filled out.”);
                return false;}
            if (!gender) {
                alert(“Please select a gender.”);
                return false;}
            var mobilePattern = /^[0-9]{10}$/;
            if (!mobilePattern.test(mobileNumber)) {
                alert(“Please enter a valid 10-digit mobile number.”);
                return false;}
            var emailPattern = /^[a-Za-Z0-9._-]+@[a-Za-Z0-9.-]+\.[a-Za-Z]{2,6}$/;
            if (!emailPattern.test(email)) {
                alert(“Please enter a valid email address.”);
                return false;}
            if (password !== confirmPassword) {
                alert(“Passwords do not match.”);
                return false;}
            return true;
        }
    </script>
</head>
<body>
    <form onsubmit=”return validateForm()” method=”post”>
        <h1>Student Registration Form</h1>

        <div class=”form-group”>
```

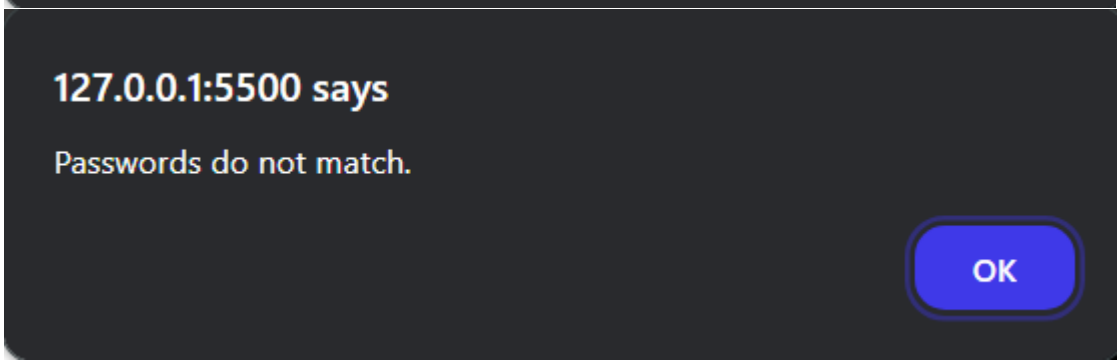
```
            <label for="first-name">Name:</label>
            <input type="text" id="first-name" name="first-name" required>
        </div>

        <div class="form-group">
            <label for="department">Department:</label>
            <select id="department" name="department" required>
                <option value="" disabled selected>Select your department</option>
                <option value="Computer-Science">Computer Science</option>
                <option value="Electrical">Electrical</option>
                <option value="Mechanical">Mechanical</option>
                <option value="Civil">Civil</option>
            </select>
        </div>

        <div class="form-group">
            <label for="roll-no">Roll No:</label>
            <input type="text" id="roll-no" name="roll-no" required>
        </div>

        <div class="form-group">
            <label>Gender:</label>
            <div class="form-group-inline">
                <label for="male"><input type="radio" id="male" name="gender" value="male">
Male</label>
                <label for="female"><input type="radio" id="female" name="gender" value="female">
Female</label>
                <label for="other"><input type="radio" id="other" name="gender" value="other">
Other</label>
            </div>
        </div>

        <div class="form-group">
            <label for="mobile-number">Mobile Number:</label>
            <input type="tel" id="mobile-number" name="mobile-number" required>
        </div>

        <div class="form-group">
            <label for="email">Email Address:</label>
            <input type="email" id="email" name="email" required>
        </div>

        <div class="form-group">
            <label for="password">Password:</label>
            <input type="password" id="password" name="password" required>
        </div>

        <div class="form-group">
            <label for="confirm-password">Confirm Password:</label>
            <input type="password" id="confirm-password" name="confirm-password" required>
        </div>

        <div class="form-group">
            <button type="submit">Register</button>
        </div>
    </form>
</body>
</html>
```

OUTPUT


Name:
Department:
Please fill out this field.
Select your department
Roll No:


Email Address:
QWERTY
Please include an '@' in the email address. 'QWERTY' is missing an '@'.


127.0.0.1:5500 says
Please enter a valid 10-digit mobile number.
OK


127.0.0.1:5500 says
Passwords do not match.
OK

Experiment No. 4

Title: Write a program to read and write HTML contents using JQuery.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>The jQuery Example</title>
   <!-- jQuery Library -->
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
   <script type="text/javascript">
      $(document).ready(function() {
         // Change background color of all <p> elements
         $("p").css("background-color", "yellow");

         // Reading HTML content of an element
         let content = $("#myid").html();
         console.log("Content of #myid:", content);

         // Writing HTML content to an element
         $(".myclass").html("Updated paragraph content using jQuery.");
      });
   </script>
</head>
<body>
   <div>
      <!-- Paragraph with class "myclass" -->
      <p class="myclass">This is a paragraph.</p>

      <!-- Paragraph with id "myid" -->
      <p id="myid">This is second paragraph.</p>

      <!-- General Paragraph -->
      <p>This is third paragraph.</p>
   </div>
</body>
</html>
```

OUTPUT

Updated paragraph content using jQuery.

This is second paragraph.

This is third paragraph.

Experiment No. 6

Title: Create a simple testing Angular applications.

Step 1: Set Up Angular Environment

1. Install Node.js
2. Install Angular CLI:

```
npm install -g @angular/cli
```

Step 2: Create a New Angular Project

1. Create a New Project

```
ng new simple-test-app
```

2. Navigate to Your Project Directory

```
cd simple-test-app
```

Step 3: Generate a Component

1. Generate a Simple Component

```
ng generate component hello-world
```

Step 4: Modify the Component

1. Edit hello-world.component.ts: Open src/app/hello-world/hello-world.component.ts and modify it to include a simple message

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-hello-world',
  templateUrl: './hello-world.component.html',
  styleUrls: ['./hello-world.component.css']
})
export class HelloWorldComponent {
  message: string = 'Hello, World! This is a simple Angular application.';
}
```

2. Edit hello-world.component.html: Update src/app/hello-world/hello-world.component.html

```
<h1>{{ message }}</h1>
```

Step 5: Use the Component in Your App

1. Edit app.component.html: Open src/app/app.component.html and include the hello-world component

```
<h1>Welcome to Simple Test App</h1>
<app-hello-world></app-hello-world>
```

Step 6: Run the Application

1. Serve the Application: In your terminal, run

```
ng serve
```

2. Open in Browser: Open web browser and navigate to http://localhost:4200. You should see your welcome message and the "Hello, World!" message from the component.

Step 7: Testing Your Application

1. Angular comes with built-in support for unit testing with Jasmine and Karma. Test run using

```
ng test
```

OUTPUT

My Angular App

Hello, World! This is a simple Angular application.

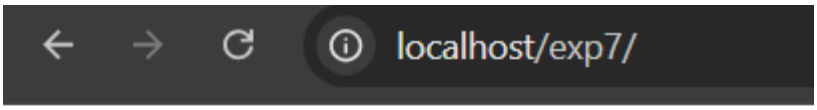
Experiment No. 7

Title: **Program based on PHP variables, Expression, arrays, control structure**

```
<?php
// Variables
$students = array(
    "Alice" => 85,
    "Bob" => 92,
    "Charlie" => 78,
    "Diana" => 90
);

// Function to determine the grade based on the score
function determineGrade($score) {
    if ($score >= 90) {
        return "A";
    } elseif ($score >= 80) {
        return "B";
    } elseif ($score >= 70) {
        return "C";
    } else {
        return "F";
    }
}

// Array to store results
$results = array();

// Control structure to iterate through the students array
foreach ($students as $name => $score) {
    // Expression to calculate the grade
    $grade = determineGrade($score);

    // Store the result in the results array
    $results[$name] = array(
        "score" => $score,
        "grade" => $grade
    );
}

// Output the results
echo "<h1>Student Grades</h1>";
echo "<table border='1'>";
echo "<tr><th>Name</th><th>Score</th><th>Grade</th></tr>";

foreach ($results as $name => $info) {
    echo "<tr>
            <td>$name</td>
            <td>{$info['score']}</td>
            <td>{$info['grade']}</td>
          </tr>";
}

echo "</table>";
```

```
?>
```

OUTPUT



Student Grades

Name	Score	Grade
Alice	85	B
Bob	92	A
Charlie	78	C
Diana	90	A

Experiment No. 8

Title : Experiment Based on basic OOP concepts in PHP

```
<?php
class Car {
// properties (attributes) with proper visibility
public $model;
public $color;
public $fuelLevel;
// Constructor to initialize the car with model and color
public function __construct($model, $color) {
$this->model = $model;
$this->color = $color;
$this->fuelLevel = 100; // Default fuel level is set to 100
}
// Public method to get the model of the car
public function getModel() {
return $this->model;
}
// Public method to get the color of the car
public function getColor() {
return $this->color;
}
// Public method to get the fuel level of the car
public function getFuelLevel() {
return $this->fuelLevel;
}
// Public method to simulate driving, reducing fuel level
public function drive() {
// Simulate driving by reducing fuel level
$this->fuelLevel -= 10;
if ($this->fuelLevel < 0) {
$this->fuelLevel = 0; // Ensure fuel level doesn't go below 0
}
}
}
// Create a new car object
$myCar = new Car("Toyota", "Blue");// Output the car details
echo "Model: " . $myCar->getModel() . "<br>";
echo "Color: " . $myCar->getColor() . "<br>";
echo "Fuel Level: " . $myCar->getFuelLevel() . "%<br>";
// Simulate driving the car
$myCar->drive();
// Output the updated fuel level after driving
echo "Updated Fuel Level after driving: " . $myCar->getFuelLevel() . "%";
?>
```

OUTPUT

```
Model: Toyota
Color: Blue
Fuel Level: 100%
Updated Fuel Level after driving: 90%
```

Experiment No. 9

Title : Experiment Based on advanced OOP concepts in PHP

```
<?php

// Abstract class for Media items
abstract class Media {
    protected $title;
    protected $author;
    const MEDIA_TYPE = 'General';
    // Constructor to initialize title and author
    public function __construct($title, $author) {
        $this->title = $title;
        $this->author = $author;
    }
    // Abstract method to get details of the media
    abstract public function getDetails();
    // Static method to get media type
    public static function getMediaType() {
        return static::MEDIA_TYPE;
    }
}
// Interface for borrowable items
interface Borrowable {
    public function borrow($borrower);
}
// Class for Book, extending Media and implementing Borrowable
class Book extends Media implements Borrowable {
    private $isbn;
    public static $totalBooks = 0;
    const MEDIA_TYPE = 'Book';

    // Constructor to initialize title, author, and ISBN
    public function __construct($title, $author, $isbn) {
        parent::__construct($title, $author);
        $this->isbn = $isbn;
        self::$totalBooks++;
    }

    // Implementation of getDetails method
    public function getDetails() {
        return "Title: {$this->title}, Author: {$this->author}, ISBN: {$this->isbn}, Type: " .
self::getMediaType();
    }

    // Implementation of borrow method
    public function borrow($borrower) {
        return "{$borrower} has borrowed the book: {$this->title}";
    }
}
// Class for Magazine, extending Media and implementing Borrowable
class Magazine extends Media implements Borrowable {
    private $issueNumber;
    public static $totalMagazines = 0;
```

```
    const MEDIA_TYPE = 'Magazine';

    // Constructor to initialize title, author, and issue number
    public function __construct($title, $author, $issueNumber) {
        parent::__construct($title, $author);
        $this->issueNumber = $issueNumber;
        self::$totalMagazines++;
    }

    // Implementation of getDetails method
    public function getDetails() {
        return "Title: {$this->title}, Author: {$this->author}, Issue: {$this->issueNumber}, Type: " .
self::getMediaType();
    }

    // Implementation of borrow method
    public function borrow($borrower) {
        return "{$borrower} has borrowed the magazine: {$this->title}";
    }
}

// Usage example
$libraryItems = [
    new Book("1984", "George Orwell", "123456789"),
    new Magazine("National Geographic", "Various", "2023-09")
];

// Loop through library items and display details
foreach ($libraryItems as $item) {
    echo $item->getDetails() . PHP_EOL;
    echo $item->borrow("John Doe") . PHP_EOL . PHP_EOL; // Added extra newline for separation
}

// Accessing static properties for total counts
echo "Total Books: " . Book::$totalBooks . PHP_EOL;
echo "Total Magazines: " . Magazine::$totalMagazines . PHP_EOL;

?>
```

OUTPUT

```
Title: 1984, Author: George Orwell, ISBN: 123456789, Type: Book John Doe has borrowed the book: 1984 Title:
National Geographic, Author: Various, Issue: 2023-09, Type: Magazine John Doe has borrowed the magazine:
National Geographic Total Books: 1 Total Magazines: 1
```

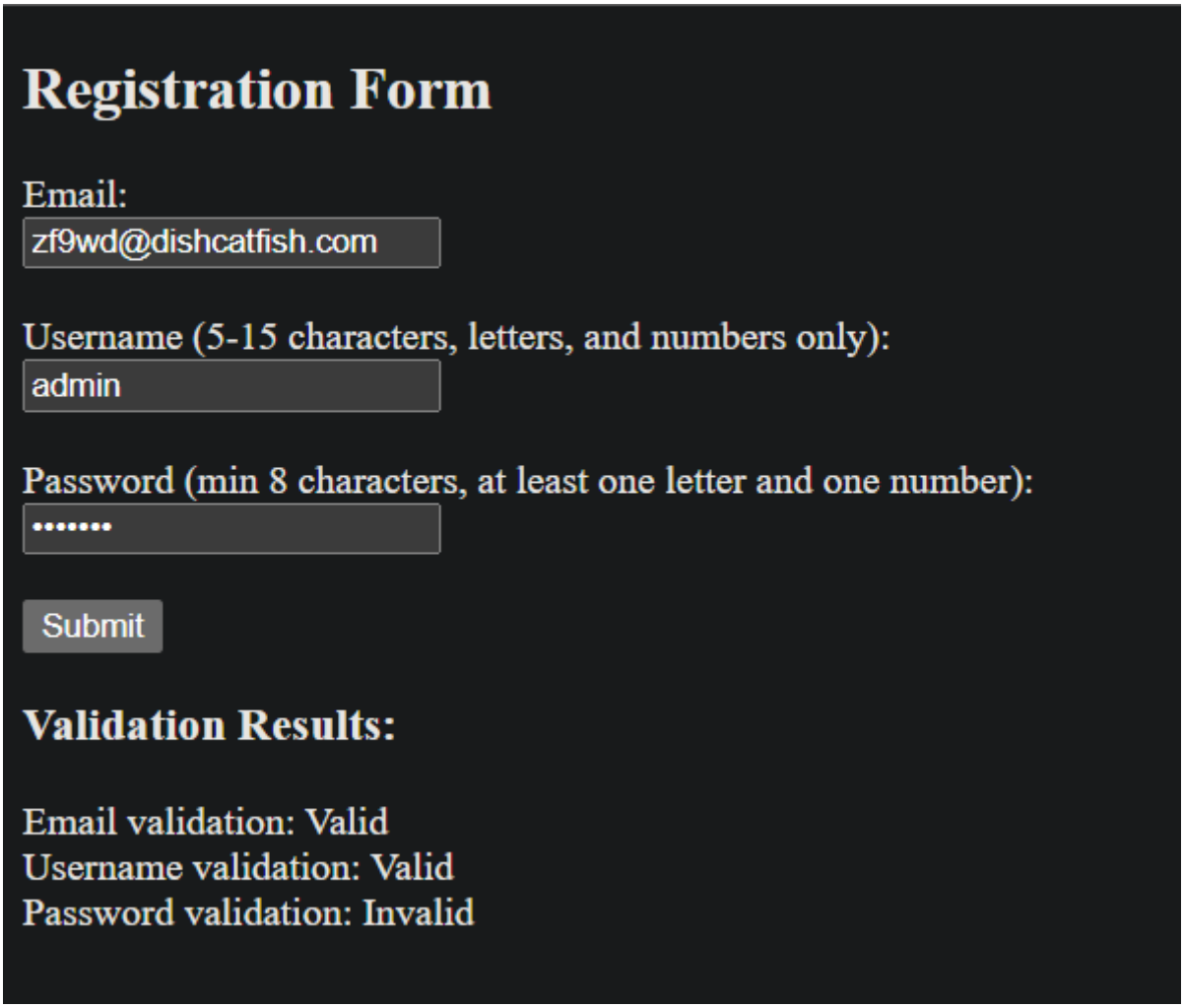
Experiment No. 10

Title: Form validation using PHP regular expressions

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Form Validation Using Regular Expressions</title>
</head>
<body>
<h2>Registration Form</h2>
<form method="POST" action="">
<label for="email">Email:</label><br>
<input type="email" id="email" name="email"
value="<?php echo isset($_POST['email']) ? htmlspecialchars($_POST['email']) : ''; ?>"
required><br><br>
<label for="username">Username (5-15 characters, letters, and numbers
only):</label><br>
<input type="text" id="username" name="username"
value="<?php echo isset($_POST['username']) ? htmlspecialchars($_POST['username']) :
''; ?>"
required><br><br>
<label for="password">Password (min 8 characters, at least one letter and one
number):</label><br>
<input type="password" id="password" name="password"
value="<?php echo isset($_POST['password']) ? htmlspecialchars($_POST['password']) :
''; ?>"
required><br><br>
<input type="submit" name="submit" value="Submit">
</form>
<?php
// Run the validation logic only after the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
// Get user inputs from form submission
$email = $_POST['email'];
$username = $_POST['username'];
$password = $_POST['password'];
// Function to validate email using regex
function validateEmail($email) {
return preg_match("/^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/", $email);
}
// Function to validate username using regex
function validateUsername($username) {
return preg_match("/^[a-zA-Z0-9]{5,15}$/", $username);
}
// Function to validate password using regex
```

```
function validatePassword($password) {
return preg_match("/^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d]{8,}$/", $password);
}// Validate the inputs
$isEmailValid = validateEmail($email);
$isUsernameValid = validateUsername($username);
$isPasswordValid = validatePassword($password);
// Output validation results after form submission
echo "<h3>Validation Results:</h3>";
echo "Email validation: " . ($isEmailValid ? "Valid" : "Invalid") . "<br>";
echo "Username validation: " . ($isUsernameValid ? "Valid" : "Invalid") . "<br>";
echo "Password validation: " . ($isPasswordValid ? "Valid" : "Invalid") . "<br>";
}
?>
</body>
</html>
```

OUTPUT



Experiment No. 11

Title: Various type of file upload using php

Form.html

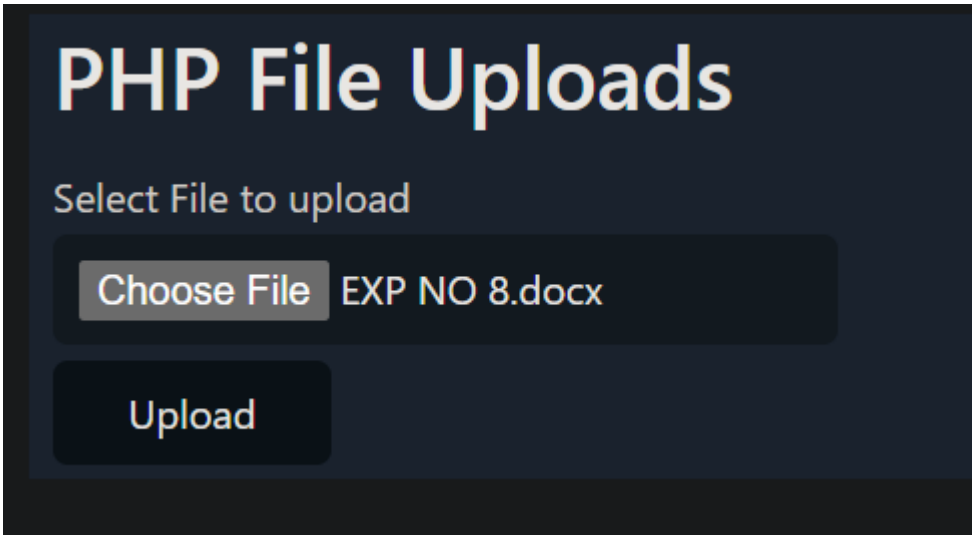
```
<!DOCTYPE html>
<html>
<head>
<title>PHP File Uploads</title>
<meta charset="UTF-8">
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/water.css@2/out/water.css">
</head>
<body>
<h1>PHP File Uploads</h1>
<form method="post" enctype="multipart/form-data" action="handleFile.php">
<label for="file1"> Select File to upload</label>
<input type="file" name="file1" id="file1" required>
<button type="submit">Upload</button>
</form>
</body>
</html>
```

Filehandle.php

```
<?php
// Check if the uploads directory exists; if not, create it
$uploadDir = __DIR__ . "/uploads/";
if (!is_dir($uploadDir)) {
mkdir($uploadDir, 0777, true);
}
// Check if a file was uploaded
$data = $_FILES["file1"];
$code = $data["error"];
if ($code !== UPLOAD_ERR_OK) {
echo "Error uploading file: " . $code;
exit("Upload Error");
}
$src = $data["tmp_name"];
$dest = $uploadDir . basename($data["name"]);
// Move the uploaded file to the destination directory
if (move_uploaded_file($src, $dest)) {
echo "File uploaded successfully.<br>",
"Size: ", $data["size"], " bytes<br>",
"Type: ", $data["type"], "<br>";
// Provide a preview or download link based on file type
$fileType = $data["type"];
$fileUrl = "uploads/" . basename($data["name"]);// Check if the uploaded file is an image
```

```
if (strpos($fileType, 'image') === 0) {
echo "<h2>File Preview:</h2>";
echo "<img src='$fileUrl' alt='File Preview' style='max-width: 500px;'><br>";
} else {
// Provide a download link for non-image files
echo "<h2>Download File:</h2>";
echo "<a href='$fileUrl' download>Click here to download the uploaded file</a><br>";
}
} else {
exit("Failed to move uploaded file.");
}
?>
```

OUTPUT



File uploaded successfully.
Size: 30059 bytes
Type: application/vnd.openxmlformats-officedocument.wordprocessingml.document

Download File:

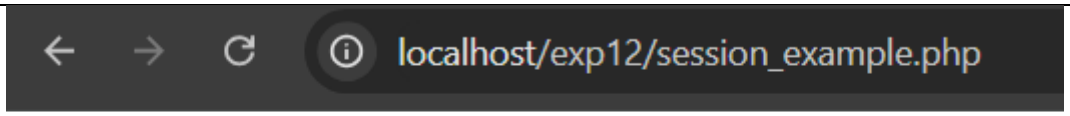
Click here to download the uploaded file

Experiment No. 12

Title : Write a program to create and handle a session, cookie in PHP

Session_example.php

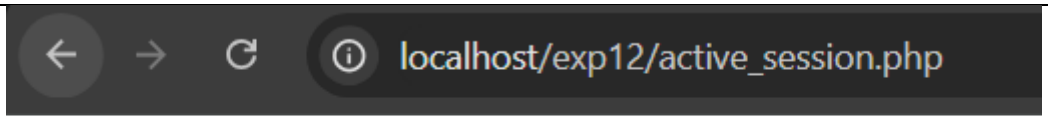
```
<?php
// Start the session session_start();
// Store session variables
$_SESSION["username"] = "JohnDoe";
$_SESSION["email"] = "johndoe@example.com"; echo "Session variables are set.";
?>
```



Session variables are set.

Active_session.php

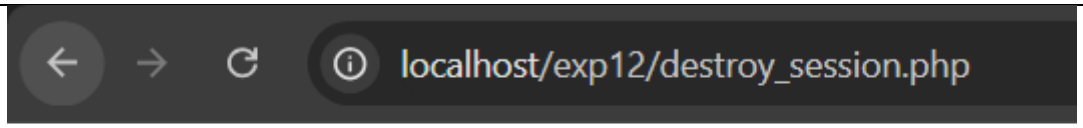
```
<?php
// Start the session session_start();
// Store session variables
$_SESSION["username"] = "JohnDoe";
$_SESSION["email"] = "johndoe@example.com"; echo "Session variables are set.";
?>
```



Session variables are set.

Destroy_session.php

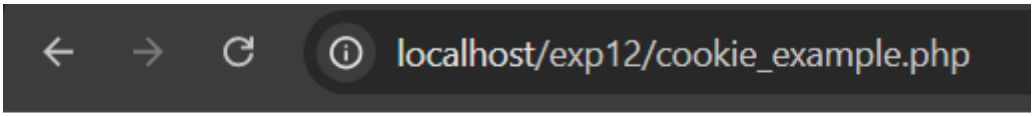
```
<?php
// Start the session session_start();
// Remove all session variables session_unset();
// Destroy the session session_destroy();
echo "Session destroyed. You have been logged out.";
?>
```



Session destroyed. You have been logged out.

Cookie_example.php

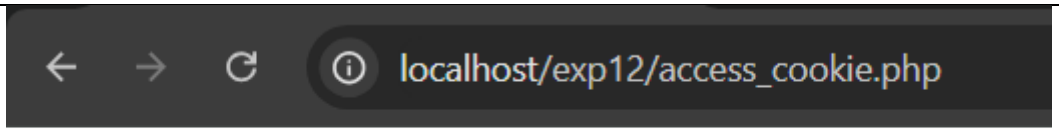
```
<?php
// Set a cookie that lasts for 1 hour
setcookie("username", "JohnDoe", time() + 3600, "/"); // "/" makes it available across the whole domain
echo "Cookie has been set.";
?>
```



Cookie has been set.

Access_cookie.php

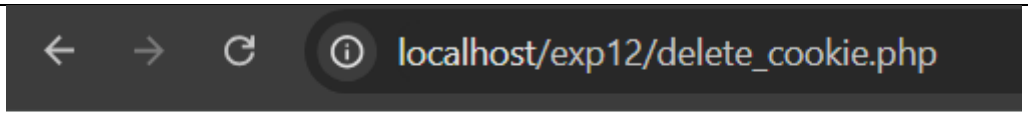
```
<?php
// Check if the cookie is set

if (isset($_COOKIE["username"])) {
echo "Welcome back, " . $_COOKIE["username"] . "!";
} else {
echo "No cookie found. Please log in.";
}
?>
```



Welcome back, JohnDoe!

Delete_cookie.php

```
<?php
// Delete the cookie by setting its expiration date to the past setcookie("username", "", time() - 3600, "/");
echo "Cookie has been deleted.";
?>
```



Cookie has been deleted.

Experiment No. 13

Title: Insert user entered data in form to MySQL database using PHP

Form.html

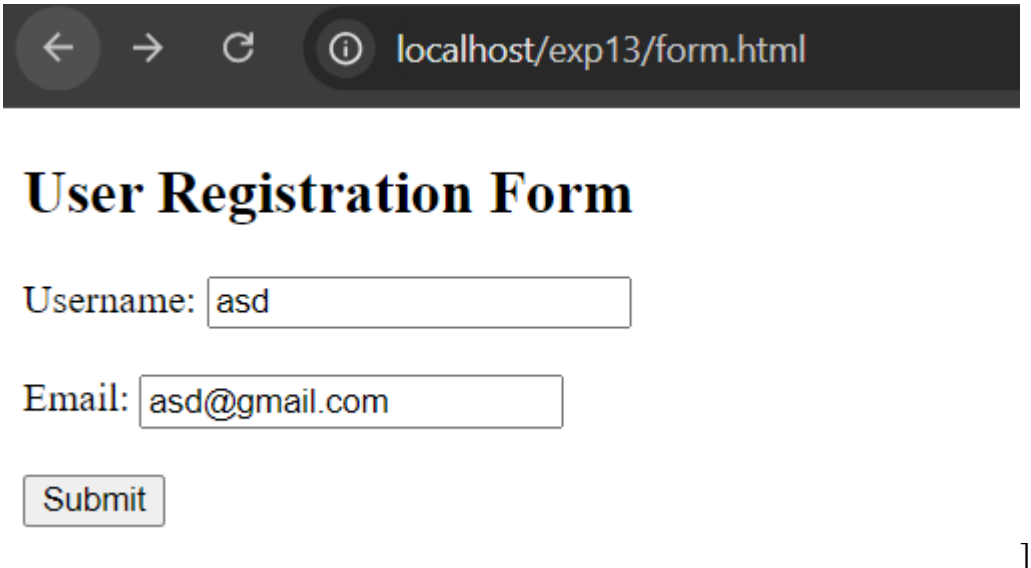
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>User Registration</title>
</head>
<body>
<h2>User Registration Form</h2>
<form action="insert.php" method="POST">
<label for="username">Username:</label>
<input type="text" id="username" name="username" required>
<br><br>
<label for="email">Email:</label>
<input type="email" id="email" name="email" required>
<br><br>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Insert.php

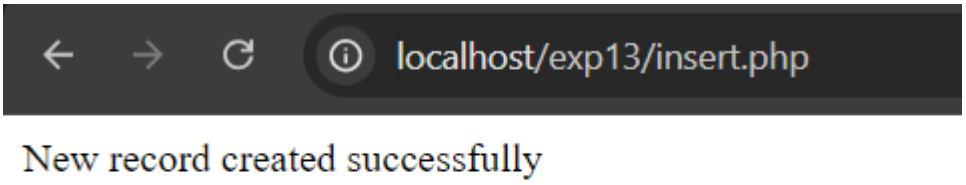
```
<?php
// Database configuration
$host = 'localhost'; // Change if your DB is hosted elsewhere
$dbname = 'WT';
$username = 'root'; // Default username for MySQL
$password = ''; // Default password (leave empty if none)
// Create a connection
$conn = new mysqli($host, $username, $password, $dbname);
// Check the connection
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}
// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
// Get user input
$user = $conn->real_escape_string($_POST['username']);
$email = $conn->real_escape_string($_POST['email']);
// Insert query
$sql = "INSERT INTO users (username, email) VALUES ('$user', '$email')";

if ($conn->query($sql) === TRUE) {
echo "New record created successfully";
} else {
echo "Error: " . $sql . "<br>" . $conn->error;
}}
// Close the connection
```

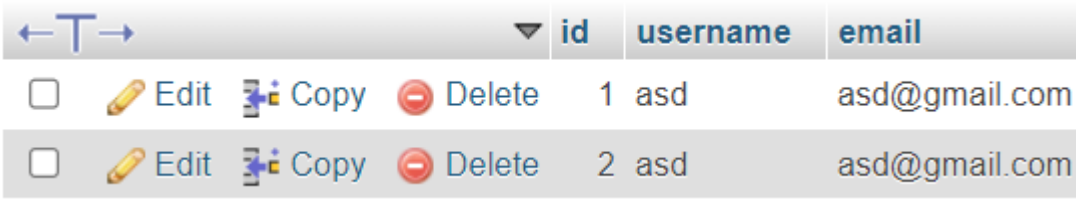
```
$conn->close();
?>
```

OUTPUT



]





		id	username	email
☐	Edit Copy Delete	1	asd	asd@gmail.com
☐	Edit Copy Delete	2	asd	asd@gmail.com

Experiment No. 14

Title : Update user data in MySQL database using PHP

Form.html

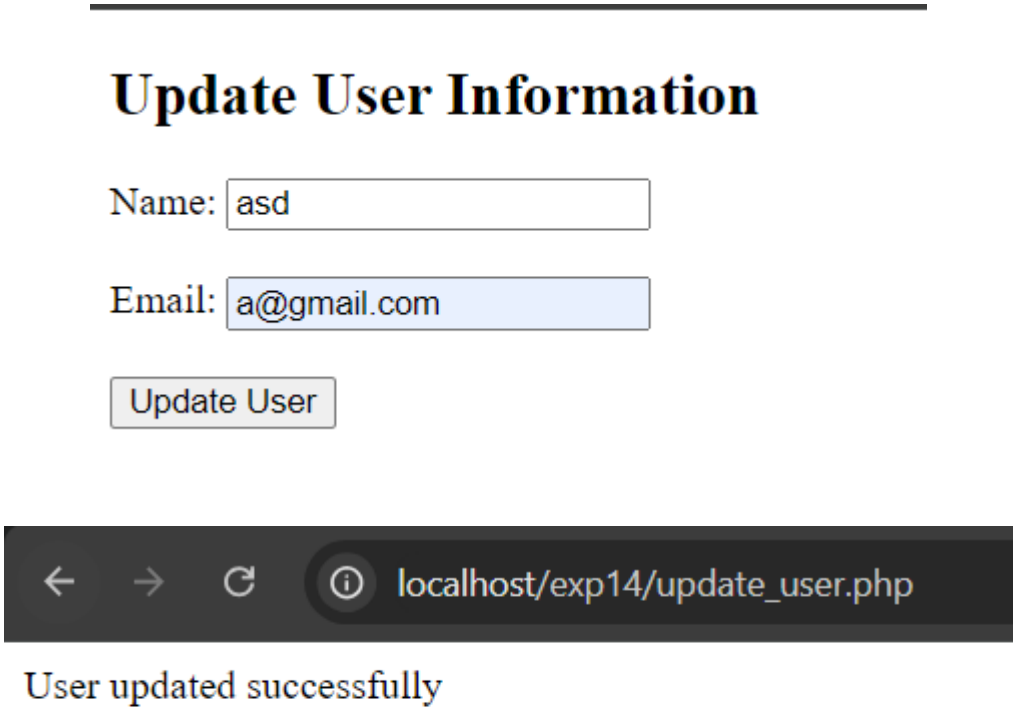
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Update User</title>
</head>
<body>
<h2>Update User Information</h2>
<form action="update_user.php" method="post">
<input type="hidden" name="id" value="1"> <!-- Assuming we want to update user with ID 1 -->
<label for="name">Name:</label>
<input type="text" id="name" name="name" required>
<br><br>
<label for="email">Email:</label>
<input type="email" id="email" name="email" required>
<br><br>
<input type="submit" value="Update User">
</form>
</body>
</html>
```

Update_user.php

```
<?php
// Database credentials
$host = 'localhost';
$username = 'root';
$password = '';
$dbname = 'WT';

// Create connection
$conn = mysqli_connect($host, $username, $password, $dbname);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Check if form data is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Get user ID, name, and email from form input
    $id = $_POST['id'];
    $name = $_POST['name'];
    $email = $_POST['email'];

    // Prepare the SQL query
    $sql = "UPDATE users SET username = '$name', email = '$email' WHERE id = $id";
```

```
    // Execute the query
    if (mysqli_query($conn, $sql)) {
        echo "User updated successfully";
    } else {
        echo "Error updating user: " . mysqli_error($conn);
    }
}

// Close the connection
mysqli_close($conn);
?>
```

OUTPUT



Experiment No. 15

Title: NodeJs program to handle the error in NodeJs

```
const fs = require('fs');

// Function to read a file
function readFile(filePath) {
    return new Promise((resolve, reject) => {
        fs.readFile(filePath, 'utf8', (err, data) => {
            if (err) {
                reject(err); // Reject the promise if there's an error
            } else {
                resolve(data); // Resolve the promise with the file data
            }
        });
    });
}

// Main function to execute the file reading
async function main() {
    const filePath = 'example.txt'; // Change this to an invalid path to test error handling
    try {
        const data = await readFile(filePath);
        console.log('File Content:', data);
    } catch (error) {
        console.error('Error reading file:', error.message);
    }
}

// Execute the main function
main();
```

OUTPUT

```
PS C:\FY\WT\EXP15> node .\errorhandling.js
Error reading file: ENOENT: no such file or directory, open 'C:\FY\WT\EXP15\example.txt'
PS C:\FY\WT\EXP15> node .\errorhandling.js
File Content:
PS C:\FY\WT\EXP15> node .\errorhandling.js
File Content: hello
PS C:\FY\WT\EXP15>
```