WEB TECHNOLOGY AND APPLICATIONS

CS422I1C

LABORATORY MANUAL

IV Semester B.E.

(Academic Year: 2023-24)

Prepared by

Ms. Amritha Mohan C K

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

SAHYADRI

College of Engineering & Management

Adyar, Mangaluru - 575007

Institute Vision

To be a premier institution in Technology and Management by fostering excellence in education, innovation, incubation and values to inspire and empower the young minds.

Institute Mission

M1: Creating an academic ambience to impart holistic education focusing onindividual growth, integrity, ethical values and social responsibility.

M2: Develop skill-based learning through industry-institution interaction to enhance competency and promote entrepreneurship.

M3: Fostering innovation and creativity through competitive environment with state-of-the-art infrastructure.

Department Vision

To be a center of excellence in Information Science and Engineering through interactive Teaching-learning process, research and innovation

Department Mission

M1: Creating competitive ambience to enhance the innovative andexperiential learning process through state-of the art infrastructure

M2: Grooming young minds through industry-institute interactions to solve societal issues and inculcate affinity towards research and entrepreneurship.

M3: Promoting team work and leadership qualities through inter-disciplinary activities in diversified areas of information science and engineering

Program Outcomes:

PO1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs):

PSO1: Exhibit competency and skills in distributed computing, information security, cyber security, data analytics, and machine learning.

PSO2: Able to provide sustainable solution to implement and validate data science projects.

WEB TECHNOLOGY AND APPLICATION

(Effective from the Academic Year 2023 - 2024)

IV SEMESTER

Course Code	**CS422I1C**	CIA Marks	50
Number of Contact Hours/Week (L: T: P: S)	3:0:2:0	SEE Marks	50
Total Hours of Pedagogy	40L + 20P	Exam Hours	03

CREDITS – 4

COURSE PREREQUISITES:

- Programming Fundamentals

COURSE OBJECTIVES:

Develop web pages using HTML syntax and semantics.

- Develop web pages using HTML syntax and semantics.
- Construct and visually format forms using HTML and CSS.
- Develop Client-Side Scripts using JavaScript to generate and display the contents dynamically.
- Develop fully functional dynamic web applications using the concepts of PHP, MySQL.
- Construct a scalable web based system using Laravel.

LABORATORY COMPONENTS

Exp. No.	Experiment Description	CO No.	Bloom's Taxonomy Level
1.	Create a simple webpage using HTML.	CO1	CL3
2.	Use frames to Include Images and Videos.	CO1	CL3
3.	Add a Cascading Style sheet for designing the web page.	CO2	CL3
4.	Design a dynamic web page with validation using JavaScript.	CO3	CL3
5.	Design an HTML having a text box and four buttons viz Factorial, Fibonacci, Prime, and Palindrome. When a button is pressed an appropriate javascript function should be called to display a. Factorial of that number b. Fibonacci series up to that number c. Prime numbers up to that number d. Is it palindrome or not	CO3	CL3
6.	Write JavaScript programs on Event Handling a. Validation of registration form b. Open a Window from the current window c. Change color of background at each click of button or refresh of a page d. Display calendar for the month and year selected from combo box e. On Mouse over event	CO3	CL3
7.	Demonstrate a simple web application using PHP, MySQL.	CO4	CL3

COURSE OUTCOMES		
Upon completion of this course, the students will be able to:		
CO No.	**Course Outcome Description**	**Bloom's Taxonomy Level**
CO1	Develop web pages using HTML syntax and semantics.	CL3
CO2	Construct and visually format forms using HTML and CSS.	CL3
CO3	Develop Client-Side Scripts using JavaScript to generate and display the contents dynamically.	CL3
CO4	Develop fully functional dynamic web applications using the concepts of PHP, MySQL.	CL3
CO5	Construct a scalable web based system using Laravel.	CL3

CO No.	**Programme Outcomes (PO)**												**Programme Specific Outcome (PSO)**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
CO1	3	3	3		2				2	2	2	2		
CO2	3	3	3		2				2	2	2	2		
CO3	3	3	3		2				2	2	2	2		
CO4	3	3	3		2				2	2	2	2		
CO5	3	3	3		2				2	2	2	2		
3: Substantial (High)					**2: Moderate (Medium)**					**1: Poor (Low)**				

List of Experiments

Expt. No.	Objective of the Experiment
1	Create a Simple Webpage Using HTML
2	Use frames to Include Images and Videos
3	Add a cascading style sheet for designing the web page
4	Design a dynamic web page with validation using JavaScript
5	Design an html having a text box and four buttons viz. factorial, Fibonacci, prime, and palindrome. when a button is pressed an appropriate JavaScript function should be called to display a. factorial of that number b. Fibonacci series up to that number c. Prime numbers up to that number d. is it palindrome or not.
6	Write JavaScript programs on Event Handling a. Validation of registration form b. Open a Window from the current window c. Change color of background at each click of button or refresh of a page d. Display calendar for the month and year selected from combo box e. On mouse over event
7	Demonstrate a simple web application using PHP,MYSQL

Experiment 01: Create a Simple Webpage Using HTML

1.1 Objective:

To create a simple webpage using HTML that displays basic text and images, demonstrating the foundational structure of an HTML document**.**

1.2 Apparatus Required:

Windows, Notepad++, Web browser

1.3 Pre-Requisite:

Basic understanding of HTML concepts such as tags, elements, and attributes.

1.4 Introduction:

HTML (HyperText Markup Language) is the standard markup language used to create web pages. It consists of a series of elements that define the structure and layout of a webpage. Learning to create a simple webpage is an essential skill for anyone interested in web development. This guide will help you set up a basic HTML document and display some text and images on your webpage.

1.5 Procedure:

```
<html>
<head>
<title>MY FIRST WEB PAGE</title>
</head>
```

```
<body style=background-color:pink>
<h3 style=text-align:center;color:red;font-size:50;>SAHYADRI COLLEGE</h3><hr>
<center><h3><b>sahyadri college of engineering and management<hr></b>
</center></h3>
<h5 style=;color:pink;font-size:20;><b>
<a href="https://sahyadri.edu.in/Home/contactUs">Contact Us</a>
<p style=;color:black;font-size:12;>Sahyadri College of Engineering & Management offer...</p>
<br>
<a href="https://sahyadri.edu.in/Home/Admission"> Admission </a>
<p style=;color:black;font-size:12;>Sahyadri College of Engineering & Management offer...</p>
<br>
</body>
</html>
```

1.6 Result:

SAHYADRI COLLEGE

sahyadri college of engineering and management

Contact Us

Sahyadri College of Engineering & Management offer ...

Admission

Sahyadri College of Engineering & Management offer ...

1.7 Pre–Experimentation Questions:

1. What key elements will you include on your webpage?
2. How will you ensure the HTML document is properly structured?
3. What is the primary purpose of this webpage?

1.8 Post-Experimentation Questions:

1. Did the webpage render correctly across different browsers?
2. What challenges did you encounter in the HTML structure?
3. How did the layout and design contribute to the overall user experience?

Experiment 02.Use frames to Include Images and Videos

2.1 Objective:

To create a webpage using HTML frames that includes images and videos.

2.2 Apparatus Required:

Windows, Notepad++, Web browser

2.3 Pre-Requisite:

Basic understanding of HTML and how frames work.

2.4 Introduction:

Frames allow you to divide a webpage into multiple sections, each capable of displaying a different Document or media. Although frames are less common today due to advancements in web technologies, they can still be useful for specific layouts

2.5 Procedure:

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Webpage with Frames</title>
</head>
<frameset cols="50%,50%">
```

```
<frame src="images_frame.html" name="images_frame">
<frame src="videos_frame.html" name="videos_frame">
<noframes>
<body>
<p>This page requires a browser that supports frames.</p>
</body>
</noframes>
</frameset>
</html>
```

Images_frame.html

```
<! DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Images Frame</title>
  </head>
<body>
<h2>Images</h2>
<img src="image1.jpg" alt="Image 1">
<img src="image2.jpg" alt="Image 2">
<! -- Add more image tags as needed -->
</body>
</html>
```

Videos_frame.html

```
<! DOCTYPE html>
<html lang="en">
```

```
<head>
<meta charset="UTF-8">
<title>Videos Frame</title>
</head>
<body>
<h2>Videos</h2>
<video width="320" height="240" controls>
<source src="video1.mp4" type="video/mp4">Your browser does not support the video tag.
</video>
</body>
</html>
```

2.6 Result:

2.7 Pre-Requisite Questions:

1. What content will you display in each frame?
2. How will you define the layout of the frames?
3. Are you familiar with the syntax for creating frames in HTML?

2.8 Post-Requisite Questions:

1. Did using frames enhance the organization of content on the page?
2. What issues did you face regarding the display of images or videos?
3. How could you improve the responsiveness of the framed layout?

Experiment 03:Add a cascading style sheet for designing the web page

3.1 Objective:

To create a visually appealing webpage using HTML frames and a separate CSS file for styling.

3.2 Apparatus Required:

Windows, Notepad++, Web browser

3.3 Pre-Requisite:

Basic understanding of HTML, CSS, and how to link stylesheets.

3.4 Introduction:

Cascading Style Sheets (CSS) are used to control the layout and appearance of web pages. By separating style from content, CSS makes it easier to manage the design of a website. In this example, we will create a CSS file to style our framed webpage that includes an image and a video.

3.5 Procedure:

styles.css

```
h1
{
color:red;
font-
family:caStellar;
font-size:22pt;
text-decoration:underline;
}
```

```
h2
{
color:blue;
font-
family:Chiller;
font-size:18pt;
text-decoration:overline;
}
p
{
color:magenta;
font-family:Trebuchet
MS;font-size:14pt;
font-style:italic;
}
```

fontstyles.html

```
<html>
<head>
<title>Usage of different font,styles and colors </title>
<link rel="stylesheet" type="text/css" href="styles.css"/>
</head>
<body>
<h1>This header is in red</h1>
<h2>This header is in blue</h2>
<p>This is normal text</p>
</body>
</html>
```

A .Set a background image for the page

bg_image.html

```
<html>
<head>
<title>Setting background image</title>
 <style type="text/css">body
 {
background-image:url("image1.jpg");background-repeat:no-repeat

}
</style>
</html></head>
<body text="#ee78a2">
<h1>Life is beautiful!!!</h1>
</body>
```

3.6 Result:

This header is in red

This header is in blue

This is normal text

Background image:

3.7 Pre-Requisite Questions:

1. What visual styles do you intend to apply?
2. How will you structure your CSS?
3. Are you familiar with CSS properties that will be important for this project?

3.8 Post-Requisite Questions:

1 .How did the CSS changes affect the overall appearance of the webpage?

2. What difficulties did you encounter while styling the elements?

Experiment 04. Design a dynamic web page

4.1 Objective:

To create a dynamic webpage that includes a form with JavaScript validation to ensure user input is correct before submission.

4.2 Apparatus Required:

Windows, Notepad++, Web browser

4.3 Pre-Requisite:

Fundamental of JavaScript, HTML, CSS, Web Browser

4.4 Introduction:

JavaScript is commonly used for client-side validation to improve user experience by providing immediate feedback. This example demonstrates a simple registration form that checks if the required fields are filled out correctly.

4.5 Procedure:

scripts.js

```
function validateForm() {
   var name = document.getElementById("name").value;
   var email = document.getElementById("email").value;
   var password = document.getElementById("password").value;

   var emailPattern = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;
```

```
    if (name == "") {
        alert("Name must be filled out");
        return false;
    }

    if (email == "" || !emailPattern.test(email)) {
        alert("Please enter a valid email address");
        return false;
    }

    if (password == "" || password.length < 6) {
        alert("Password must be at least 6 characters long");
        return false;
    }

    // Redirect only if all validations pass
    window.location.href = "success.html";
    return false; // Prevents default form submission behavior
}
```

form.html

```
<! DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Simple Validation Form</title>
<link rel="stylesheet" href="styles.css">
```

```
</head>
<body>
<h2>Simple Validation Form</h2>
<form id="myForm" onsubmit="return validateForm()">
<label for="name">Name:</label>
<input type="text" id="name" name="name"><br><br>
<label for="email">Email:</label>
<input type="text" id="email" name="email"><br><br>
<label for="password">Password:</label>
<input type="password" id="password" name="password"><br><br>
<input type="submit" value="Submit">
</form>
<script src="scripts.js"></script>
</body>
</html>
```

styless.css

```
body {
font-family: Arial, sans-serif;
}

form {
max-width: 300px;

margin: auto;
padding: 1em;
border: 1px solid#ccc;
border-radius: 1em;
```

```
}

label {
margin-top: 1em; margin-
bottom:5em;color:
#333333; display: block;
}

input[type="text"],
input[type="password"] {
width: 100%;
padding:.5em;
box-sizing: border-box;
border: 1px solid #ccc;
border-radius: 4px;
}

input[type="submit"] {
padding: 0.7em; color: #fff;
background-color: #007BFF;
border: none;
border-radius: 4px;cursor: pointer;
}

input[type="submit"]:hover
{
background-color: #0056b3;
  }
```

success.html

```
<!DOCTYPE html>
```

```
<html>
<head>
<title>Submission Successful</title>
</head>
<body>
<h1>Thank you for registering!</h1>
<p>Your submission was successful.</p>
</body>
</html>
```

4.6 Result:

4.7 Pre-Requisite Questions:

1. What specific form elements will require validation?
2. How will you implement the validation logic using JavaScript?
3. Are you familiar with handling user input and displaying error messages?

4.8 Post-Requisite Questions:

1. Did the validation function correctly prevent invalid submissions?
2. What types of user errors were most common during testing?
3. How did the dynamic features enhance user interaction with the f

Experiment 05: Design an html having a text box and four buttons viz. factorial,Fibonacci, prime, and palindrome.

5.1 Objective:

To create an interactive webpage that allows users to input a number and perform four mathematical operations: calculating the factorial, generating the Fibonacci series, finding prime numbers, and checking if the input is a palindrome.

5.2 Apparatus Required:

Windows, Notepad++, Web browser

5.3 Pre-Requisite:

Basic understanding of HTML, CSS, and JavaScript concepts.

5.4 Introduction:

In web development, JavaScript is widely used to add interactivity to web pages. This project will demonstrate how to use JavaScript to perform mathematical computations based on user input. The page will consist of a text box for user input and four buttons corresponding to different operations. Each button will invoke a JavaScript function that executes the relevant calculation and displays the result dynamically.

5.5 Procedure:

index.html

```
<! DOCTYPE html>
<html lang="en">
<head>
```

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Number Operations</title>
<script src="scripts.js"></script>
<style>
body {
font-family: Arial, sans-serif;
margin: 20px;
}
.container {
max-width: 400px;
margin: 0 auto;
text-align: center;
 }
input, button {
margin: 10px;
padding: 10px;
font-size: 16px;
 }
 .result {
  margin-top: 20px;
  font-size: 18px;
  color: #333;
 }
</style>
</head>
<body>
<div class="container">
<h2>Number Operations</h2>
<input type="text" id="number" placeholder="Enter a number">
<br>
<button onclick="calculateFactorial()">Factorial</button>
```

```
<button onclick="generateFibonacci()">Fibonacci</button>
<button onclick="listPrimes()">Prime</button>
<button onclick="checkPalindrome()">Palindrome</button>
<div class="result" id="result"></div>
</div>
</body>
</html>
```

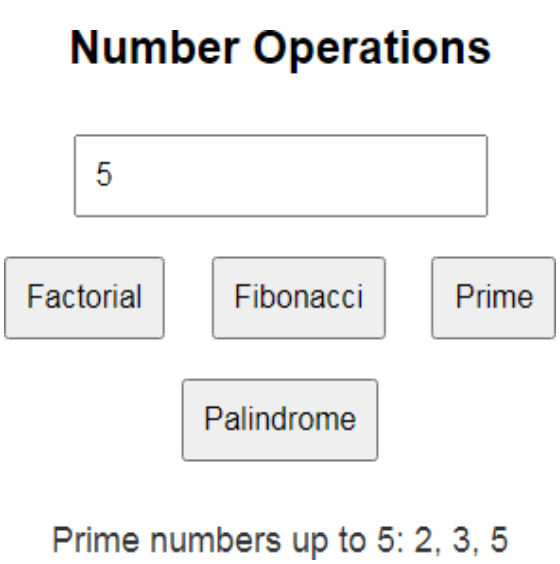
scripts.js

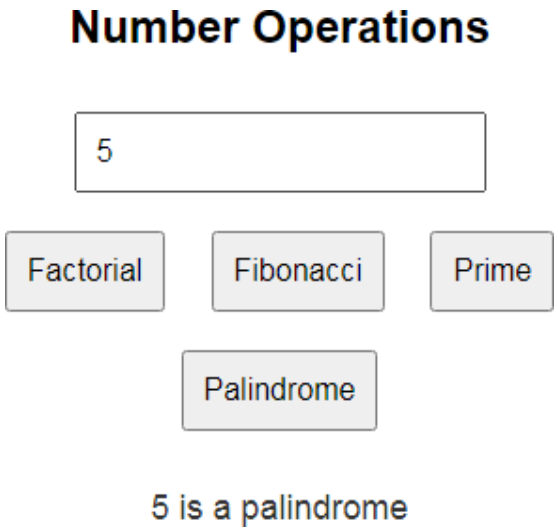
```
function calculateFactorial() {
    let number = parseInt(document.getElementById("number").value);
    if (isNaN(number) || number < 0) {
        alert("Please enter a valid non-negative number");
        return;
    }
    let factorial = 1;
    for (let i = 1; i <= number; i++) {
        factorial *= i;
    }
    document.getElementById("result").innerText = `Factorial of ${number} is
    ${factorial}`;
}

function generateFibonacci() {
    let number = parseInt(document.getElementById("number").value);
    if (isNaN(number) || number < 1) {
        alert("Please enter a valid positive number");
        return;
    }
    let fibonacci = [0, 1];
```

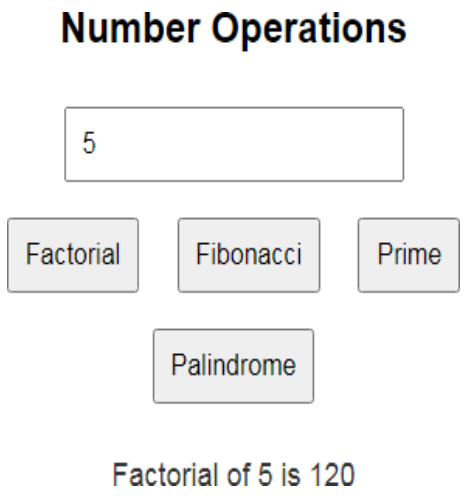
```
    for (let i = 2; i < number; i++) {
        fibonacci[i] = fibonacci[i - 1] + fibonacci[i - 2];
    }
    document.getElementById("result").innerText = `Fibonacci series up to ${number}:
    ${fibonacci.slice(0, number).join(", ")}`;
}

function listPrimes() {
    let number = parseInt(document.getElementById("number").value);
    if (isNaN(number) || number < 2) {
        alert("Please enter a valid number greater than 1");
        return;
    }
    let primes = [];
    for (let i = 2; i <= number; i++) {
        let isPrime = true;
        for (let j = 2; j * j <= i; j++) {
            if (i % j === 0) {
                isPrime = false;
                break;
            }
        }
        if (isPrime) {
            primes.push(i);
        }
    }
    document.getElementById("result").innerText = `Prime numbers up to ${number}:
    ${primes.join(", ")}`;
}
```

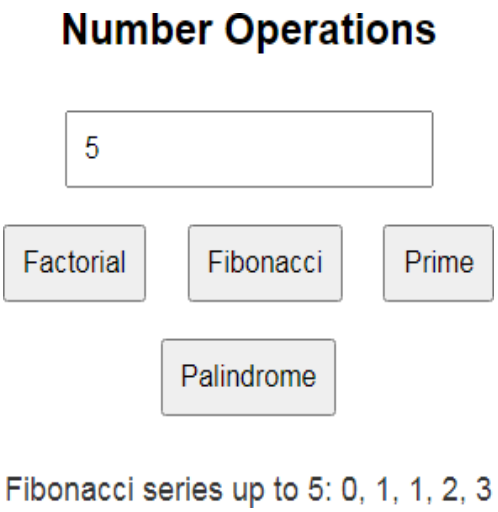
```
function checkPalindrome() {
    let number = document.getElementById("number").value;
    if (number === "") {
        alert("Please enter a valid number");
        return;
    }
    let reversed = number.split("").reverse().join("");
    let isPalindrome = (number === reversed);
    document.getElementById("result").innerText = `${number} is ${isPalindrome ? "" :
    "not "}a palindrome`;
}
```

5.6 Result:









5.7 Pre-Requisite Questions:

1. What mathematical functions will you implement in JavaScript?
2. How will you design the user interface for input and buttons?
3. Are you confident in your ability to write the necessary JavaScript function

5.8 Post-Requisite Questions:

1. Did each button perform its designated function accurately?
2. What challenges did you face when implementing the calculations?
3. How did users respond to the functionality of the buttons?

Experiment 06: Write JavaScript programs on Event Handling

6.1 Objective:

To demonstrate the practical application of JavaScript event handling through various functionalities such as form validation, opening a new window, changing background colors, displaying a calendar, and responding to mouse events.

6.2 Apparatus Required:

Windows, Notepad++, Web browser

6.3 Pre-Requisite:

- HTML structure and elements
- CSS for basic styling
- JavaScript fundamentals, including functions and events

6.4 Introduction:

JavaScript is a powerful scripting language that enables dynamic content and interactive elements on web pages. Event handling in JavaScript allows developers to respond to user actions, such as clicks, mouse movements, and keyboard inputs. This project showcases several common use cases for event handling, allowing users to interact with a webpage in meaningful ways.

6.5 Procedure:

a. Validation of registration form

scripts.js

```
function validateForm() {
```

```
    var name = document.getElementById("name").value;
    var email = document.getElementById("email").value;
    var password = document.getElementById("password").value;

    var emailPattern = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;

    if (name == "") {
        alert("Name must be filled out");
        return false;
    }

    if (email == "" || !emailPattern.test(email)) {
        alert("Please enter a valid email address");
        return false;
    }

    if (password == "" || password.length < 6) {
        alert("Password must be at least 6 characters long");
        return false;
    }

    // Redirect only if all validations pass
    window.location.href = "success.html";
    return false; // Prevents default form submission behavior
}
```

form.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Simple Validation Form</title>
<link rel="stylesheet" href="styles.css">
</head>
 <body>
```

```
<h2>Simple Validation Form</h2>
<form id="myForm" onsubmit="return validateForm()">
<label for="name">Name:</label>
<input type="text" id="name" name="name"><br><br>
  <label for="email">Email:</label>
<input type="text" id="email" name="email"><br><br>
<label for="password">Password:</label>
<input type="password" id="password" name="password"><br><br>
<input type="submit" value="Submit">
</form>
<script src="scripts.js"></script>
</body>
</html>
```

styless.css

```
 body {
 font-family: Arial, sans-serif;
}
 form {
 max-width: 300px;
margin: auto;
 padding: 1em;
 border: 1px solid #ccc;border-
radius: 1em;
}
 label {
margin-top: 1em;
margin-bottom: .5em;
color: #333333;
display: block;
}
```

```
input[type="text"],
input[type="password"]
 {
 width: 100%; padding: .5em;
 box-sizing:    border-box;
border: 1pxsolid #ccc;
 border-radius: 4px;
}
input[type="submit"]
{
padding: 0.7em;
color: #fff;
background-color: #007BFF;
border: none;
border-radius:
4px;cursor: pointer;
}
 input[type="submit"]:hover {
background-color: #0056b3;
```

6.6 Result:

Simple Validation Form

Name: smitha

Email: smitha@gmail.com

Password: ••••••••

Submit

B. Open a Window from the current window

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Open a New Window</title>
<script src="openWindow.js"></script>
</head>
<body>
<button onclick="openNewWindow()">Open New Window</button>
</body>
</html>
```

openWindow.js

```
function openNewWindow() {
window.open("https://www.example.com", "_blank", "width=800,height=600");
}
```

Result:

C. Change color of background at each click of button or refresh opage

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Change Background Color</title>
<script src="changeColor.js"></script>
</head>
<body onload="changeBackgroundColor()">
<button onclick="changeBackgroundColor()">Change Background Color</button>
</body>
</html>
```

changeColor.js

```
function getRandomColor() {
var letters = '0123456789ABCDEF';
var color = '#';
for (var i = 0; i < 6; i++) {
color += letters[Math.floor(Math.random() * 16)];
}
return color;
}
function changeBackgroundColor() {
document.body.style.backgroundColor = getRandomColor();
```

```
}
```

Result:

C. Display calendar for the month and year selected from combo box

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Display Calendar</title>
<script src="displayCalendar.js"></script>
</head>
<body>
<label for="month">Month:</label>
<select id="month" onchange="displayCalendar()">
<option value="0">January</option>
<option value="1">February</option>
<option value="2">March</option>
<option value="3">April</option>
<option value="4">May</option>
<option value="5">June</option>
<option value="6">July</option>
<option value="7">August</option>
<option value="8">September</option>
<option value="9">October</option>
<option value="10">November</option>
<option value="11">December</option>
</select>
<label for="year">Year:</label>
<input type="number" id="year" value="2023" onchange="displayCalendar()">
```

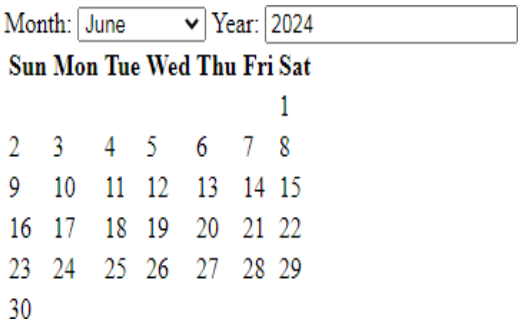
```
<div id="calendar"></div>
</body>
</html>
```

displayCalendar.js

```
function displayCalendar() {
    var month = document.getElementById("month").value;
    var year = document.getElementById("year").value;
    var firstDay = new Date(year, month, 1).getDay();
    var daysInMonth = new Date(year, parseInt(month) + 1, 0).getDate();

 var calendar = "<table
border='1'><tr><th>Sun</th><th>Mon</th><th>Tue</th><th>Wed</th><th>Thu</th><th
>Fri</th><th>Sat</th></tr><tr>";

    // Add empty cells for the first week
    for (var i = 0; i < firstDay; i++) {
        calendar += "<td></td>";
    }

    for (var day = 1; day <= daysInMonth; day++) {
        calendar += "<td>" + day + "</td>";

        // Ensure proper row breaking
        if ((firstDay + day) % 7 == 0 && day !== daysInMonth) {
            calendar += "</tr><tr>";
```

```
        }
    }
    // Close the last row properly
    calendar += "</tr></table>";
    document.getElementById("calendar").innerHTML = calendar;
}
```

Result:



E. On mouse over event

index.html

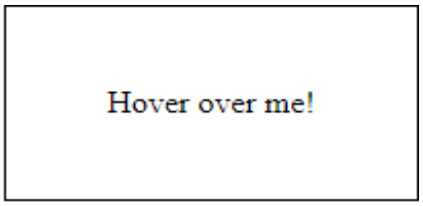
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Mouse Over Event</title>
<script src="mouseOverEvent.js"></script>
<style>
```

```
.hover-box      {
width:    200px;
height: 100px;
border: 1px solid black;
text-align: center;
line-height:100px;
margin-top: 20px;
}
</style>
</head>
<body>
<div class="hover-box" onmouseover="mouseOver()" onmouseout="mouseOut()">Hover over me!</div>
<div id="message"></div>
</body>
</html>
```

mouseOverEvent.js

```
function mouseOver() {
 document.getElementById("message").innerText = "Mouse is over the box!";
 }
 function mouseOut()
{
document.getElementById("message").innerText = "";
 }
```

Result:



6.7 Pre-Requisite Questions:

1. What events will you handle in your JavaScript programs?
2. How will you set up event listeners for user interactions?
3. Are you familiar with the concept of event delegation?

6.8 Post-Requisite Questions:

1. Did the event handling work as expected for all user interactions?
2. What issues arose with event propagation or default actions?
3. How did the implementation of events improve user experience?

Experiment 07: Demonstrate a simple web application using PHP,MYSQL

7.1 Objective:

To create a simple web application that allows users to submit and display their feedback using PHP and MySQL. This application will include a form for submitting feedback, storing it in a MySQL database, and retrieving the feedback for display.

7.2 Apparatus Required:

- A computer or laptop
- A local server environment (e.g., XAMPP, WAMP, or MAMP) that supports PHP and MySQL
- A web browser (e.g., Chrome, Firefox)
- A text editor (e.g., Notepad, Visual Studio Code, Sublime Text)

7.3 Pre-Requisite:

- HTML for creating forms and displaying data
- PHP for server-side scripting
- MySQL for database management
- Familiarity with SQL queries (INSERT, SELECT)

7.3 Introduction:

PHP is a widely-used open-source server-side scripting language that is especially suited for web development. When paired with MySQL, a popular relational database management system, it allows for dynamic web applications that can store and retrieve data efficiently.

This project will guide you through creating a simple feedback application where users can submit their comments, which are then stored in a MySQL database. Users will also be able to view previously submitted feedback on the same page. The application consists of three main components:

1. **HTML Form**: For users to submit their feedback.
2. **PHP Script**: To handle form submission and database operations.
3. **MySQL Database**: To store and retrieve feedback data

7.3 Procedure:

```
CREATE TABLE users (
id INT(11) AUTO_INCREMENT PRIMARY KEY, username
VARCHAR(50) NOT NULL,
email VARCHAR(100) NOT NULL,
passwordVARCHAR(255) NOT NULL
);
```

register.php

```
<!DOCTYPE html>
<html>
<head>
<title>User Registration</title>
</head>
<body>
<h2>Register</h2>
<form method="post" action="register.php">
Username: <input type="text" name="username" required><br><br> Email:
<input type="email" name="email" required><br><br>
Password: <input type="password" name="password" required><br><br>
<input type="submit" name="register" value="Register">
</form>
</body>
</html>
```

```
<?php
if (isset($_POST['register'])) {
```

```
$username = $_POST['username'];
$email = $_POST['email'];
$password = password_hash($_POST['password'], PASSWORD_DEFAULT);


$conn = new mysqli('localhost', 'root', '', 'simple_web_app');
if($conn->connect_error) {
die('Connection Failed: ' . $conn->connect_error);
}
$stmt = $conn->prepare("INSERT INTO users (username, email, password) VALUES (?,
?, ?)");
$stmt->bind_param("sss", $username, $email, $password);
if ($stmt->execute()) {
echo "Registration successful!";
} else {
echo "Error: " . $stmt->error;
}
$stmt->close();
$conn->close();

}
?>
```

login.php

```
<!DOCTYPE html>
<html>
<head>
<title>User Login</title>
</head>
```

```
<body>
<h2>Login</h2>
<form method="post" action="login.php">
Username: <input type="text" name="username" required><br><br> Password:
<input type="password" name="password" required><br><br>
<input type="submit" name="login" value="Login">
</form>
</body>
</html>
<?php
if (isset($_POST['login'])) {
$username = $_POST['username'];
$password = $_POST['password'];
$conn = new mysqli('localhost', 'root', '', 'simple_web_app'); if

($conn->connect_error) {
die('Connection Failed: ' . $conn->connect_error);
}
$stmt = $conn->prepare("SELECT * FROM users WHERE username=?");
$stmt->bind_param("s", $username);
$stmt->execute();
$result = $stmt->get_result();
$user = $result->fetch_assoc();

if ($user && password_verify($password, $user['password'])) {
echo"Login successful! Welcome, " . $user['username'];
} else {
echo "Invalid username or password.";
}
```

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

7.7 Result:

Register

Username:

Email:

Password:

Register

Registration successful!

Login

Username:

Password:

Login

Login successful! Welcome, divyaLogin successful! Welcome, divya

7.7 Pre-Requisite Questions:

1. What functionalities will your web application provide?
2. How will you structure the MySQL database to support the application?

7.8 Post-Requisite Questions:

1. Did the application connect to the database successfully and perform CRUD operations?
2. What errors or challenges did you encounter during development?
3. How did you ensure data security in the application?