Programme: B E – Computer Science and Engineering (AI&ML)
&
Computer Science and Engineering (Cyber Security)

ADVANCE WEB TECHNOLOGIES LABORATORY MANUAL

TERM: 15/04/2024 to 27/6/2024
COURSE CODE: CIL48/CYL48
SEM: IV

HTML

1. Imagine you are tasked with developing a static website for an online book management system tailored specifically for engineering students to view books related to their respective branches. The website should include the following pages and features:

Home Page:

The home page is divided into three frames:

Top Frame: This frame displays the logo and the college name, along with navigation links to the Home page, Login page, and Registration page.

Left Frame: This frame includes at least four navigation links of the branches, when clicked; display the catalogue of books relevant to each link of the department.

Right Frame: This frame initially shows a description of the website and will load the pages corresponding to the links in the left frame.

Login Page:

The login page should provide fields for the user to enter their login credentials, along with submit and reset buttons.

Registration Page:

The registration page to enter the student’s details.

- Name
- Password
- E-mail id
- Phone number
- Gender
- Dob
- Language known
- Submit and cancel button

index.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title>Online Book Store</title>
</head>
<frameset rows="15%,10%,*">
<frameset cols="20%,*">
<frame name="f11" src="logo.html" scrolling="no"></frame>
<frame name="f12" src="title.html" scrolling="no"></frame>
</frameset>
<frameset cols="20%,20%,20%,20%,*">
<frame name="f21" src="home.html"></frame>
```

```
<frame name="f22" src="login.html"></frame>
<frame name="f23" src="reg.html"></frame>
<frame name="f24" src="cat.html"></frame>
<frame name="f25" src="cart.html"></frame>
</frameset>
<frameset cols="20%,*">
<frame name="f31" src="branches.html"></frame>
<frame name="f32" src="homepage.html"></frame>
</frameset>
</frameset>
</html>
```

logo.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title></title>
</head>
<body>
<center><img src="images/logo.jpg" width="100" height="100"
/></center>
</body>
</html>
```

title.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
```

```
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title></title>
</head>
<body>
<h1 align="center">Online Book Store</h1>
</body>
</html>
```
home.html
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title></title>
</head>
<body>
<a href="homepage.html" target="f32">Home</a>
</body>
</html>
```
login.html
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title></title>
</head>
```

```
<body>
<a href="loginpage.html" target="f32">Login</a>
</body>
</html>
```
reg.html
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title></title>
</head>
<body>
<a href="regpage.html" target="f32">Registration</a>
</body>
</html>
```
branches.html
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title></title>
</head>
<body>
<ul>
<li>Civil</li>
<li>EEE</li>
```

```
<li>Mech</li>
<li>ECE</li>
<li>CSE</li>
<li>MBA</li>
<li>MCA</li>
</ul>
</body>
</html>
```
homepage.html
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title></title>
</head>
<body>
<p align="justify">Welcome to Online Book Store.<br>Here you find
all types of books, magazines and national and international
journals.</p>
</body>
</html>
```
loginpage.html
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
```

```
<title></title>
</head>
<body>
<form>
<table align="center" cellspacing="15px">
<tr>
<td><b>Login</b></td>
<td><input type="text" name="uname" /></td>
</tr>
<tr>
<td><b>Password</b></td>
<td><input type="password" name="upwd" /></td>
</tr>
<tr>
<td colspan="2" align="center">
<input type="submit" />&nbsp; &nbsp;
<input type="reset" />
</td>
</tr>
</table>
</form>
</body>
</html>
```

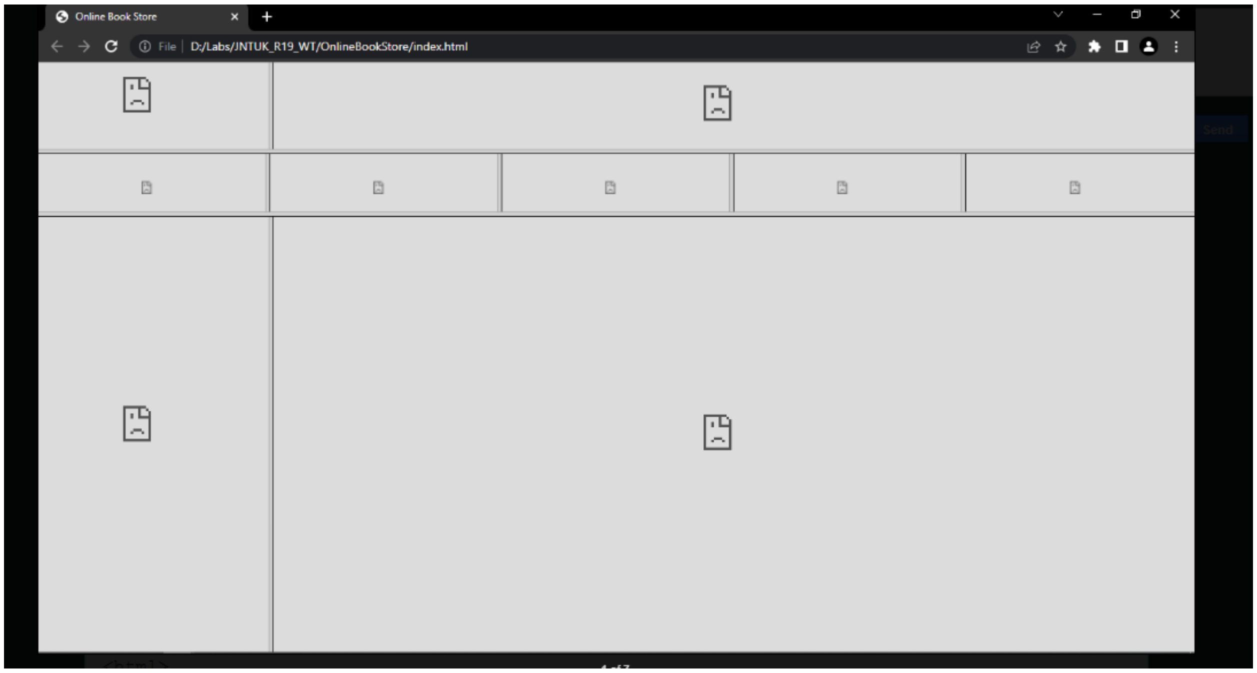
regpage.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-
scale=1">
```

```
</head>
<body>
<form>
<h1 align="center"> REGISTRATION </h1>
<table align="center" cellspacing="10">
<tr>
<td>Name::</td>
<td> <input name="uname" type="text"> </td>
</tr>
<tr>
<td>Password::</td>
<td> <input name="pwd" type="password"> </td>
</tr>
<tr>
<td> E-mail ID:: </td>
<td> <input name="email" type="text"> </td>
</tr>
<tr>
<td> Phone Number:: </td>
<td> <input name="phno" type="text" maxlength="10"> </td>
</tr>
<tr>
<td> Gender:: </td>
<td><input name="gender" type="radio"
value="Male">&nbsp;&nbsp;Male &nbsp;&nbsp;&nbsp;&nbsp;
<input name="gender" type="radio"
value="Female">&nbsp;&nbsp;Female</td>
</tr>
<tr>
<td> Date of Birth </td>
```

```
<td>
<input type="date" name="dob" />
</td>
</tr>
<tr>
<td valign="top">Languages Known::</td>
<td>
<input name="lk" type="checkbox"
value="English">&nbsp;English<br>
<input name="lk" type="checkbox"
value="Telugu">&nbsp;Telugu<br>

<input name="lk" type="checkbox" value="Hindi">&nbsp;Hindi<br>

<input name="lk" type="checkbox" value="Tamil">&nbsp;Tamil<br>
</td>
</tr>
<tr>
<td>Address</td>
<td><textarea></textarea></td>
</tr>
<tr>
<td colspan="2" align="center"> <input type="submit"
value="Register">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<input type="reset" value="Cancel"> </td>
</tr>
</table>
</form>
</body>
</html>
```

Output :



HTML5

2. Create a webpage that provides detailed information about a memorable travel experience. This webpage should utilize various HTML5 features and semantic tags to ensure a well-structured and interactive user experience. Follow the detailed requirements below:

HTML Structure:

- Create an HTML document that includes all the specified tasks and semantic tags.

SVG Square:

- Add an SVG element to draw a green square with a 6px brown stroke.

MathML Expression:

- Include a MathML element to display the mathematical expression d=x2−y2d = x^2 - y^2d=x2−y2.

Meta Tag for Redirection:

- Use a meta tag in the <head> section to redirect the page after 5 seconds.

Semantic Tags for Travel Experience:

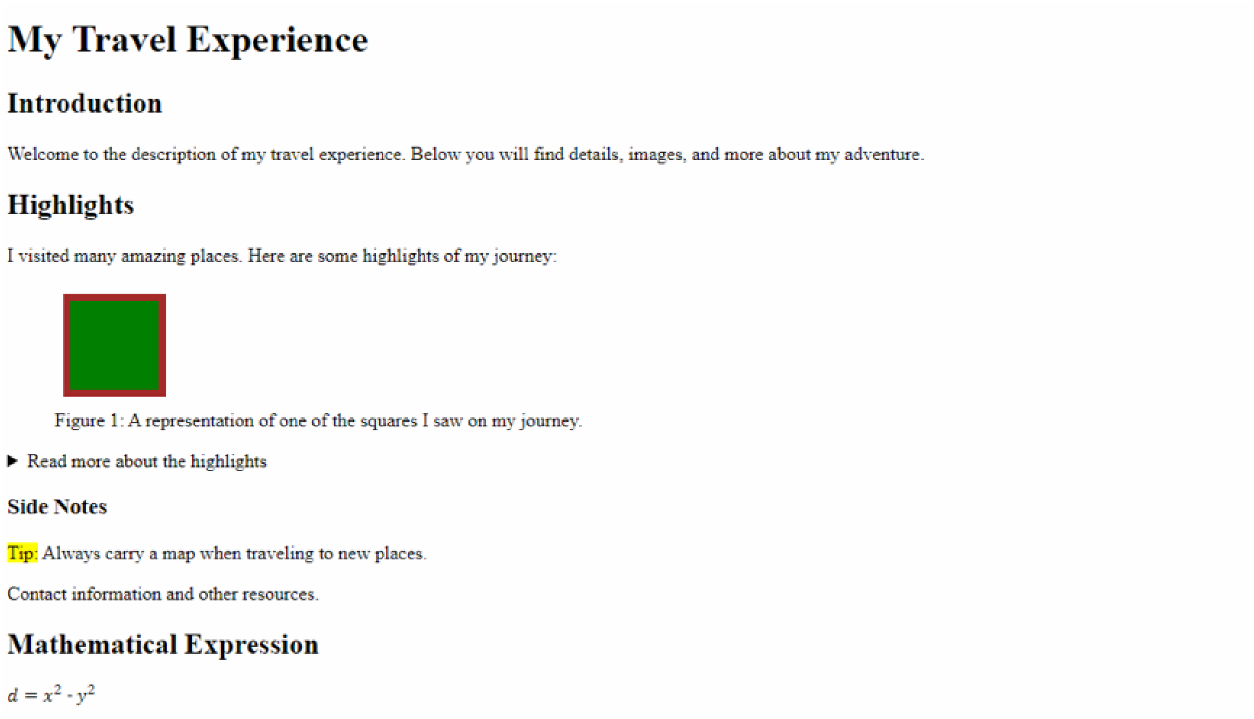
- Use the semantic tags to create sections for your travel experience, including articles, side notes, detailed information, images with captions, and other relevant content.

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Travel Experience</title>

  <meta http-equiv="refresh" content="10; url=https://example.com">

  <style>

    .square {

      fill: green;

      stroke: brown;

      stroke-width: 6px;
```

```
        }
    </style>
</head>
<body>
    <header>
        <h1>My Travel Experience</h1>
    </header>

    <main>
        <article>
            <section>
                <h2>Introduction</h2>
                        <p>Welcome to the description of my travel experience.
Below you will find details, images, and more about my adventure.</p>
            </section>


            <section>
                <h2>Highlights</h2>
                  <p>I visited many amazing places. Here are some highlights
of my journey:</p>
                <figure>
```

```
                    <svg width="100" height="100">
                        <rect x="10" y="10" width="80" height="80"
class="square"></rect>
                    </svg>
                            <figcaption>Figure 1: A representation of one of the
squares I saw on my journey.</figcaption>
                </figure>
                <details>
                    <summary>Read more about the highlights</summary>
                        <p>The journey included visits to historical sites, nature
trails, and vibrant cities.</p>
                </details>
            </section>
        </article>


        <aside>
            <h3>Side Notes</h3>
              <p><mark>Tip:</mark> Always carry a map when traveling to
new places.</p>
        </aside>
    </main>
    <footer>
        <p>Contact information and other resources.</p>
    </footer>

    <section>
        <h2>Mathematical Expression</h2>
        <math xmlns="http://www.w3.org/1998/Math/MathML">
            <mi>d</mi>
            <mo>=</mo>
            <msup><mi>x</mi><mn>2</mn></msup>
            <mo>-</mo>
            <msup><mi>y</mi><mn>2</mn></msup>
        </math>
    </section>
</body>
</html>
```

Output:



My Travel Experience

Introduction

Welcome to the description of my travel experience. Below you will find details, images, and more about my adventure.

Highlights

I visited many amazing places. Here are some highlights of my journey:

Figure 1: A representation of one of the squares I saw on my journey.

▶ Read more about the highlights

Side Notes

Tip: Always carry a map when traveling to new places.

Contact information and other resources.

Mathematical Expression

$d = x^2 - y^2$

CSS (cascading style sheet)

3.Create an interactive and visually appealing online resume for a student using various CSS techniques. The resume should incorporate inline, internal and external style sheets, and demonstrate a wide range of CSS properties to enhance its appearance and functionality. Follow the detailed requirements below:

HTML Structure:

- Create an HTML document with sections for personal information, education, skills, experience, and contact information.

Inline, Internal, and External Style Sheets:

- Demonstrate the use of inline, internal, and external CSS in your HTML document.

Font and Text Styles:

- Apply various font and text styles using CSS.

Background Image:

- Set a background image for the entire page and another for a specific section.

Layers:

- Use CSS positioning to create layered effects.

Customized Cursor:

- Implement a customized cursor for the resume.

Link Styles:

- Define styles for links, including hover effects.

Box Model:

- Use margin, padding, border, and other box model properties.

Colors:

- Apply different colors to various elements.

Resume.html

```
<!DOCTYPE html>

<html lang="en">

<head>

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

```
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Student Resume</title>
    <link rel="stylesheet" href="External.css">
    <style>
        .internal-style {
            background-color: #f0f8ff;
            padding: 20px;
            margin: 10px;
        }
        .custom-cursor {
            cursor: url('cursor.png'), auto;
        }
    </style>
</head>
<body>
    <div class="custom-cursor">
        <header>
            <h1 style="color: hotpink;">Student Name</h1>
            <p class="internal-style">Email: student@example.com | Phone:
123-456-7890</p>
```

```
        </header>

        <section id="education">

           <h2 style="font-style: italic;"> Education</h2>

           <p>Bachelor of Science in Computer Engineering</p>

        </section>

        <section id="skills">

           <h2 style ="font-style: italic;">Skills</h2>

           <ul style="list-style-type: none;">

                 <li style="display: inline; margin: 5px; padding: 10px 20px;
border: medium double black;">HTML</li>

                 <li style="display: inline; margin: 5px; padding: 10px 20px;
border: medium double black;">CSS</li>

                 <li style="display: inline; margin: 5px; padding: 10px 20px;
border: medium double black;">JavaScript</li>

           </ul>

        </section>

        <section id="experience">

           <h2 style="font-style:italic;">Experience</h2>

           <p>Intern at XYZ Company</p>

        </section>

        <section id="contact">

           <h2 style="font-style:italic;">Contact</h2>
```

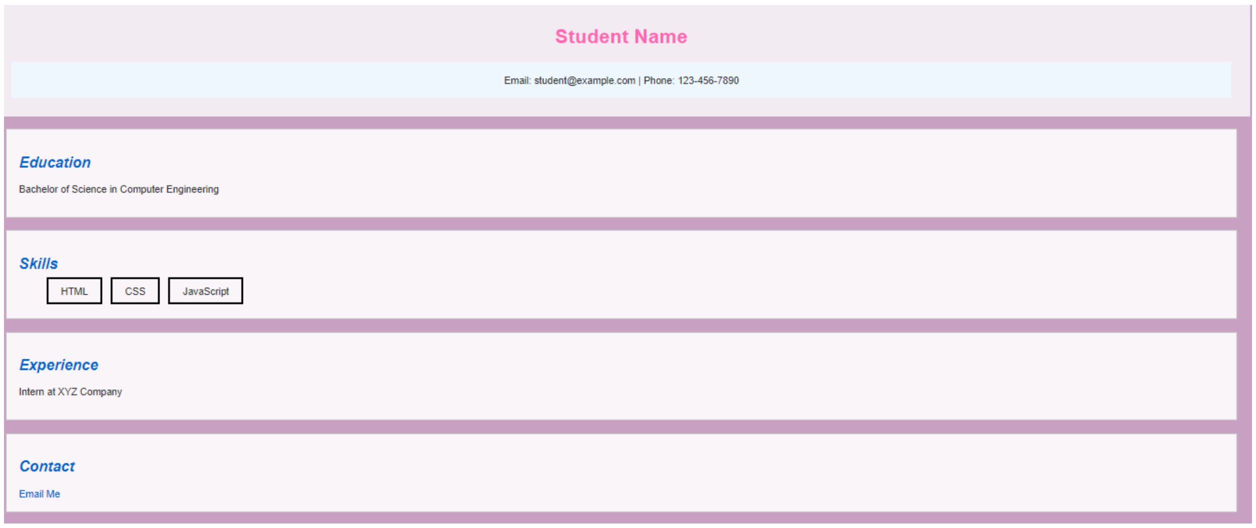
```
            <a href="mailto:student@example.com">Email Me</a>
        </section>
    </div>
</body>
</html>
```

Resume.css

```
body {
    font-family: Arial, sans-serif;
    background-image: url('background.jpg');
    background-size: cover;
    color: #333;
    background-color: #c9a1c3;
}

header {
    background: rgba(255, 255, 255, 0.8);
    padding: 20px;
    text-align: center;
}

h2 {
    font-size: 1.5em;
    color: #0066cc;
}

a {
    color: #0066cc;
    text-decoration: none;
}

a:hover {
    text-decoration: underline;
    color: #004080;
}

/* Box model example */
#education, #skills, #experience, #contact {
    border: 1px solid #ccc;
    padding: 20px;
    margin: 20px;
    background: rgba(255, 255, 255, 0.9);
}
```

```
/* Custom cursor */
.custom-cursor {
    cursor: url('cursor.png'), auto;
}
```

Output:



Student Name

Email: student@example.com | Phone: 123-456-7890

Education

Bachelor of Science in Computer Engineering

Skills

HTML CSS JavaScript

Experience

Intern at XYZ Company

Contact

Email Me

Java script

4. To create an html page with 2 combo box populated with month & year, to display the calendar for the selected month & year from combo box using JavaScript.

Create a html file named as “Claendar_month.html”

a. Add two combo box one to display month & another for year and one buttons.

b. When the button is clicked display the calendar for the selected values.

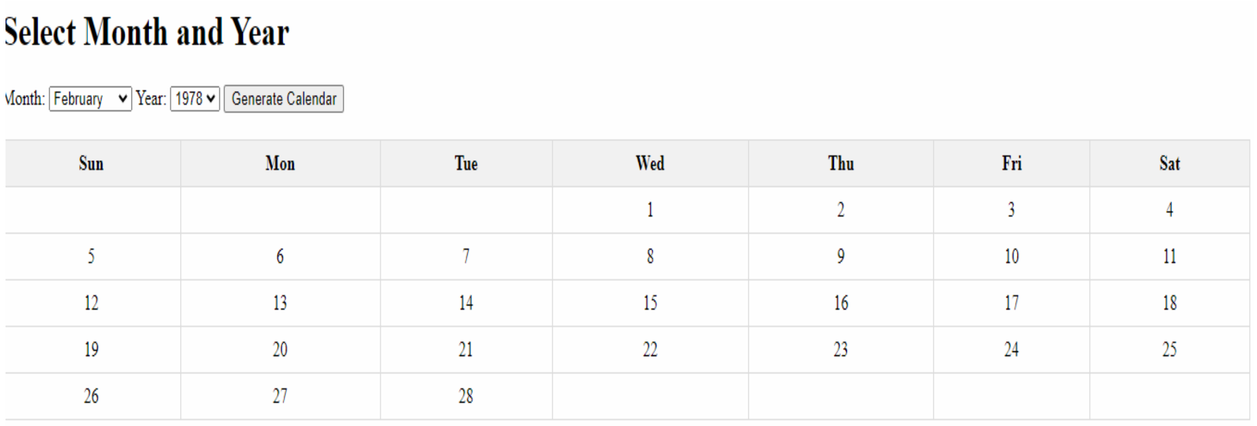
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Calendar Month</title>
    <style>

        table {
            border-collapse: collapse;
            width: 100%;
            margin-top: 20px;
        }
        th, td {
            border: 1px solid #ddd;
            padding: 8px;
            text-align: center;
        }
        th {
            background-color: #f2f2f2;
        }
    </style>
</head>

<body>
    <h1>Select Month and Year</h1>
    <form>
```

```
        <label for="month">Month:</label>
        <select id="month" name="month">
            <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>
       <select id="year" name="year">

     <script>
                const currentYear = new Date().getFullYear();
                              for (let year = currentYear - 50; year <=
         currentYear + 50; year++)       {
                                            document.write(`<option
value="${year}">${year}</option>`);
                }
            </script>

        </select>
```

```
                                        <button       type="button"
onclick="generateCalendar()">Generate Calendar</button>
  </form>

 <div id="calendar"></div>
  <script>
    function generateCalendar() {
      const month = document.getElementById('month').value;
      const year = document.getElementById('year').value;
       const daysInMonth = new Date(year, parseInt(month) + 1,
0).getDate();
      const firstDay = new Date(year, month, 1).getDay();

      let table = '<table>';
                                                  table      +=
'<tr><th>Sun</th><th>Mon</th><th>Tue</th><th>Wed</th><th
>Thu</th><th>Fri</th><th>Sat</th></tr>';

      let day = 1;
      for (let i = 0; i < 6; i++) { // Assuming maximum 6 rows for
the calendar
        table += '<tr>';
        for (let j = 0; j < 7; j++) {
          if ((i === 0 && j < firstDay) || day > daysInMonth) {
            table += '<td></td>';
          } else {
            table += `<td>${day}</td>`;
            day++;
          }
        }
```

```
            table += '</tr>';
             if (day > daysInMonth) break; // Exit loop if all days are
added
         }

         table += '</table>';
         document.getElementById('calendar').innerHTML = table;
      }
   </script>
</body>
</html>
```

Output:



Select Month and Year

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				

5. Develop an online exam application for a CSE (ai/ml) course. The exam consists of multiple-choice questions that students can answer by selecting radio buttons. The task is to create a web page that displays the exam, collects the student's answers, calculates the result, and allows the student to reset the page for a new attempt.

Create an HTML file named exam.html.

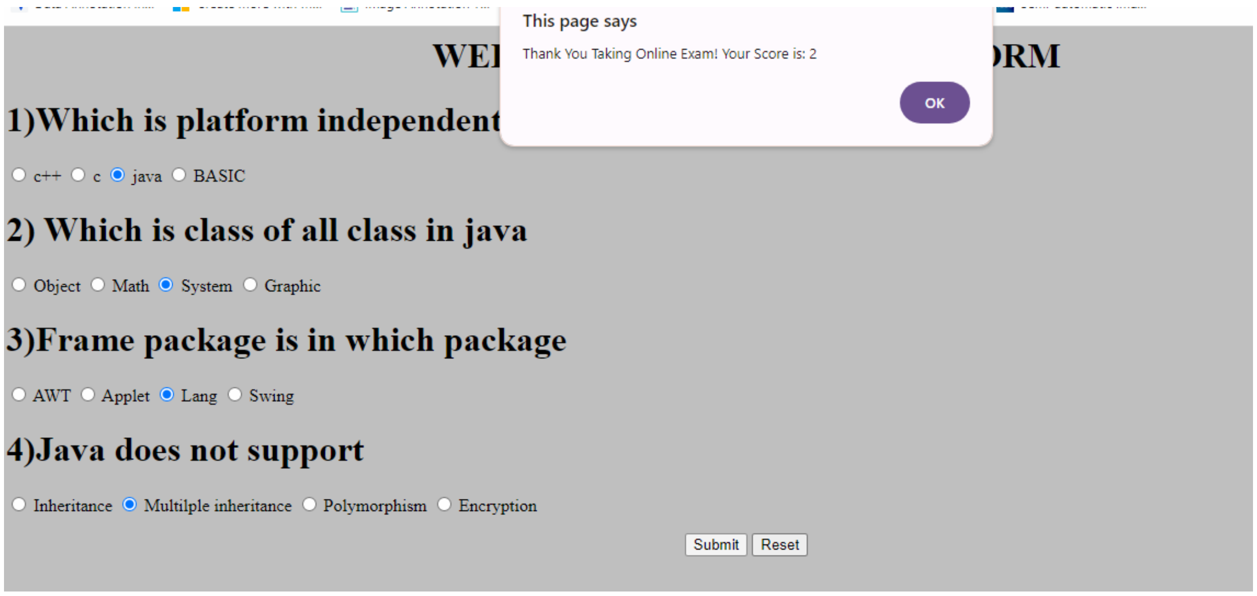
- **Step 1: Display four questions related to computer science. Each question should have four optional answers, represented by radio buttons.**
- **Step 2: When the student submits their answers, display the result in an alert box showing the number of correct answers.**
- **Step 3: Provide a reset button to clear the current answers and reset the page to its initial state for the next exam attempt.**

```
<html>

<head>

<title>Exam</title>

<script language="javascript">

 function exam(form)

 {

 var i=0;

 if(form.one[2].checked)

 i=i+1;

 if(form.three[0].checked)

 i=i+1;

 if(form.four[0].checked)

 i=i+1;

 if(form.five[1].checked)

 i=i+1;
 window.alert("Thank You Taking Online Exam! Your Score is: "+i);
 }
</script>
</head>
<body bgcolor=silver>
<form onSubmit="exam(this)">
<center><h1><blink>WELCOME TO ONLINE EXAM
FORM</blink></h1></center>
<p>
 <h1>1)Which is platform independent language</h1>
<input type="radio" name="one" value="c++">
<label>c++</label>
<input type="radio" name="one" value="c">
<label>c</label>
<input type="radio" name="one" value="java">
<label>java</label>
<input type="radio" name="one"value="basic">
<label>BASIC</label>
 </p>
 <p>
```

```
<h1>2) Which is class of all class in java</h1>
<input type="radio" name="three" value="object">
<label>Object</label>
<input type="radio" name="three" value="math">
<label>Math</label>
<input type="radio" name="three" value="system">
<label>System</label>
<input type="radio" name="three" value="graphic">
<label>Graphic</label> </p>
 <p>
<h1>3)Frame package is in which package</h1>
<input type="radio" name="four" value="awt">
<label>AWT</label>
<input type="radio" name="four" value="applet">
<label>Applet</label>
<input type="radio" name="four" value="lang">
<label>Lang</label>
<input type="radio" name="four" value="swing">
<label>Swing</label>
</p>
```

```
 <p>
 <h1>4)Java does not support</h1>
<input type="radio" name="five" value="inheritance">
<label>Inheritance</label>
<input type="radio" name="five" value="multiple inheritance">
<label>Multilple inheritance</label>
<input type="radio" name="five" value="polymorphism">
<label>Polymorphism</label>
<input type="radio" name="five" value="encryption">
<label>Encryption</label>
</p>
<p><center>
<input type="submit" value="Submit">
<input type ="reset" value="Reset">
</center></p>
</body>
</html>
```

Output:


This page says

Thank You Taking Online Exam! Your Score is: 2

WEB ... RM

1)Which is platform independent

( ) c++ ( ) c [✓ java] ( ) BASIC

2) Which is class of all class in java

( ) Object ( ) Math [✓ System] ( ) Graphic

3)Frame package is in which package

( ) AWT ( ) Applet [✓ Lang] ( ) Swing

4)Java does not support

( ) Inheritance [✓ Multilple inheritance] ( ) Polymorphism ( ) Encryption

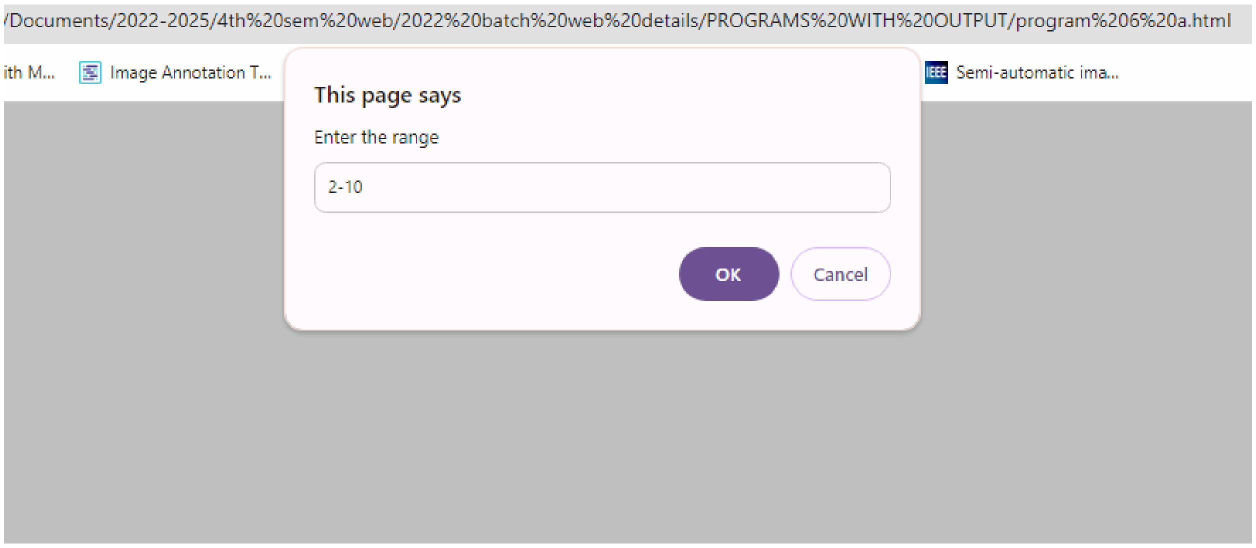
6. a. Write a JavaScript that calculates the squares and cubes of the numbers from 0 to 10 and outputs HTML text that displays the resulting values in an HTML table format.

```
<html>

<center>

<body onload=sc()>



<script>

        function sc()

{

rng=prompt('Enter the range');

res=rng.split("-");

if(res.length!=2)

{
```

```
alert("invalid range ");

return;

}

first=parseInt(res[0]);

second=parseInt(res[1]);

if(first>second)

{

alert("invalid range ");

return;

}

str="<table
border=2><tr><th>Number</th><th>Square</th><th>Cube</th></tr>";

for(i=first;i<=second;i++)

{

str=str+"<tr><td>"+i+"<td>"+(i*i)+"<td>"+(i*i*i);

}

document.write(str);

}

</script>

</body>
```

```
</center>

</html>
```

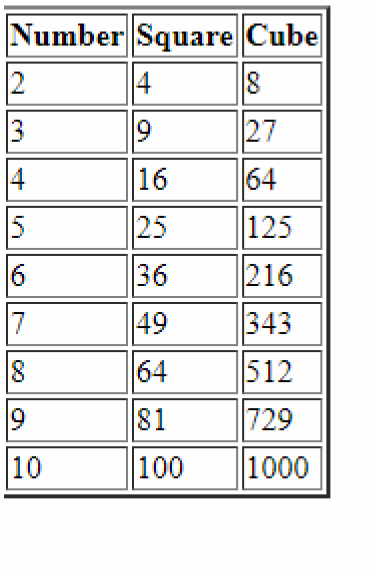
Output a:


/Documents/2022-2025/4th%20sem%20web/2022%20batch%20web%20details/PROGRAMS%20WITH%20OUTPUT/program%206%20a.html

This page says

Enter the range

2-10



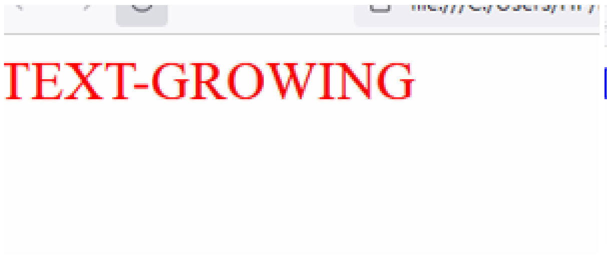
Number	Square	Cube
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

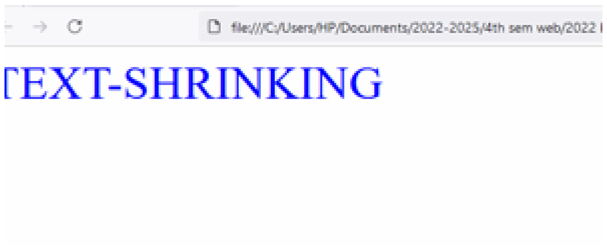
6. b Write a JavaScript code that displays text "TEXT-GROWING" with increasing font size in the interval of 100ms in RED COLOR, when the font size reaches 50pt it displays "TEXT-SHRINKING" in BLUE color. Then the font size decreases to 5pt.

```
<html>
<script>
        var grow=true;
function fnts()
{
fntsize=document.getElementById("p1").style.fontSize;
document.getElementById("p1").style.color="red";
ifntsize=parseInt(fntsize);
window.setTimeout(fntGS,100,ifntsize);
}
function fntGS(ifs)
{
if(grow)
{
ifs=ifs+1;
if(ifs<=50)
{
document.getElementById("p1").style.fontSize=ifs+"pt";
}
else
{
grow=false;
document.getElementById("p1").style.color="blue";
document.getElementById("p1").innerHTML="TEXT-SHRINKING";
}
}
else
{
ifs=ifs-1;
if(ifs<5)
return;
document.getElementById("p1").style.fontSize=ifs+"pt";
}
window.setTimeout(fntGS,100,ifs);
}
</script>
<body onload=fnts()>
<p id="p1" style="font-size:12pt;color=black"> TEXT-GROWING
</p>
</body>
```

</html>

Output b:


TEXT-GROWING

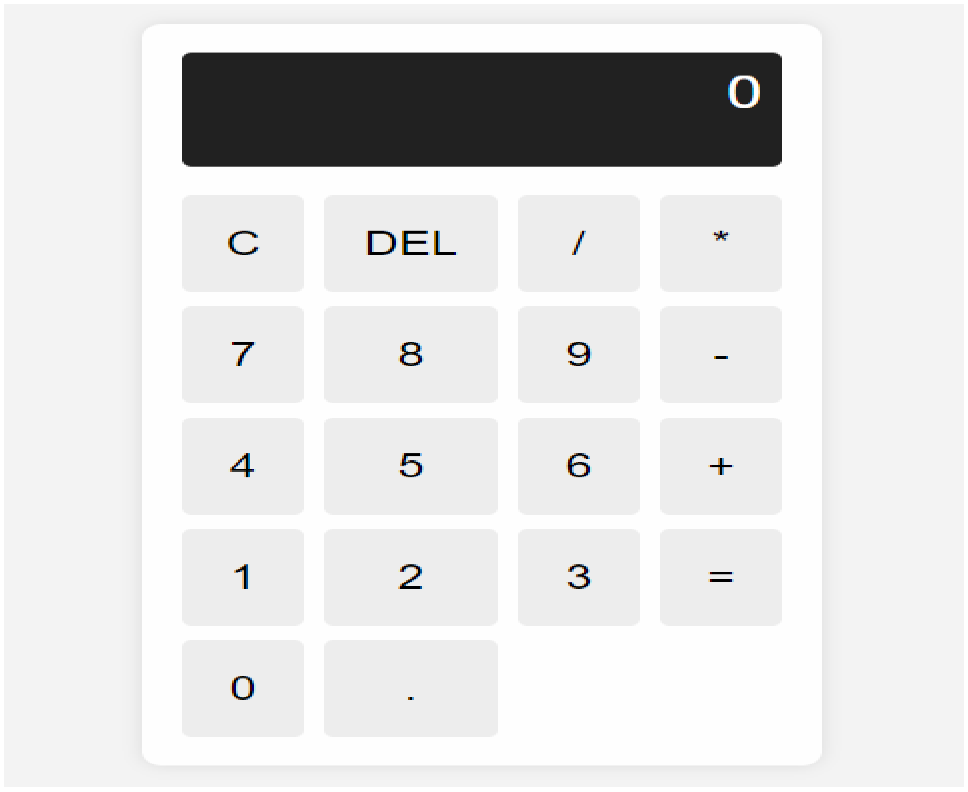

TEXT-SHRINKING

7. Develop a simple program to create a calculator using HTML, CSS, and JavaScript that performs basic mathematical operations like addition, subtraction, multiplication, and division.

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Calculator</title>

    <link rel="stylesheet" href="styles.css">

</head>

<style>

    /* styles.css */

body {

    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    background-color: #f4f4f4;
    margin: 0;
    font-family: 'Arial', sans-serif;
}

.calculator {
    background-color: #fff;
    border-radius: 10px;
    box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
    padding: 20px;
    width: 300px;
}

.calculator-display {
    background-color: #222;
    color: #fff;
    font-size: 2em;
    padding: 10px;
    text-align: right;
    border-radius: 5px;
    margin-bottom: 20px;
    height: 60px;
}

.calculator-buttons {
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    grid-gap: 10px;
}

.btn {
    background-color: #eee;
    border: none;
    border-radius: 5px;
    padding: 20px;
    font-size: 1.5em;
    cursor: pointer;
    transition: background-color 0.2s;
}
```

```
.btn:hover {
    background-color: #ddd;
}

.btn:active {
    background-color: #ccc;
}

</style>
<body>
    <div class="calculator">
        <div class="calculator-display" id="display">0</div>
        <div class="calculator-buttons">
            <button class="btn" onclick="clearDisplay()">C</button>
            <button class="btn" onclick="deleteLast()">DEL</button>
            <button class="btn" onclick="appendToDisplay('/')">/</button>
            <button class="btn" onclick="appendToDisplay('*')">*</button>
            <button class="btn" onclick="appendToDisplay('7')">7</button>
            <button class="btn" onclick="appendToDisplay('8')">8</button>
            <button class="btn" onclick="appendToDisplay('9')">9</button>
            <button class="btn" onclick="appendToDisplay('-')">-</button>
            <button class="btn" onclick="appendToDisplay('4')">4</button>
            <button class="btn" onclick="appendToDisplay('5')">5</button>
            <button class="btn" onclick="appendToDisplay('6')">6</button>
            <button class="btn"
onclick="appendToDisplay('+')">+</button>
            <button class="btn" onclick="appendToDisplay('1')">1</button>
            <button class="btn" onclick="appendToDisplay('2')">2</button>
            <button class="btn" onclick="appendToDisplay('3')">3</button>
            <button class="btn" onclick="calculateResult()">=</button>
            <button class="btn" onclick="appendToDisplay('0')">0</button>
            <button class="btn" onclick="appendToDisplay('.')">.</button>
        </div>
    </div>
    <script>
        // script.js
function appendToDisplay(value) {
    const display = document.getElementById('display');
    if (display.innerText === '0') {
        display.innerText = value;
    } else {
        display.innerText += value;
    }
}

function clearDisplay() {
    document.getElementById('display').innerText = '0';
}

function deleteLast() {
    const display = document.getElementById('display');
    display.innerText = display.innerText.slice(0, -1);
    if (display.innerText === '') {
        display.innerText = '0';
    }
}

function calculateResult() {
    const display = document.getElementById('display');
    try {
        display.innerText = eval(display.innerText);
    } catch {
        display.innerText = 'Error';
    }
}
</script>
</body>
</html>
```

Output:



8. Create a JavaScript-based student course registration form that captures details like student name, USN, phone number, email ID, gender, and the course to be selected, and then displays the form details on the same page

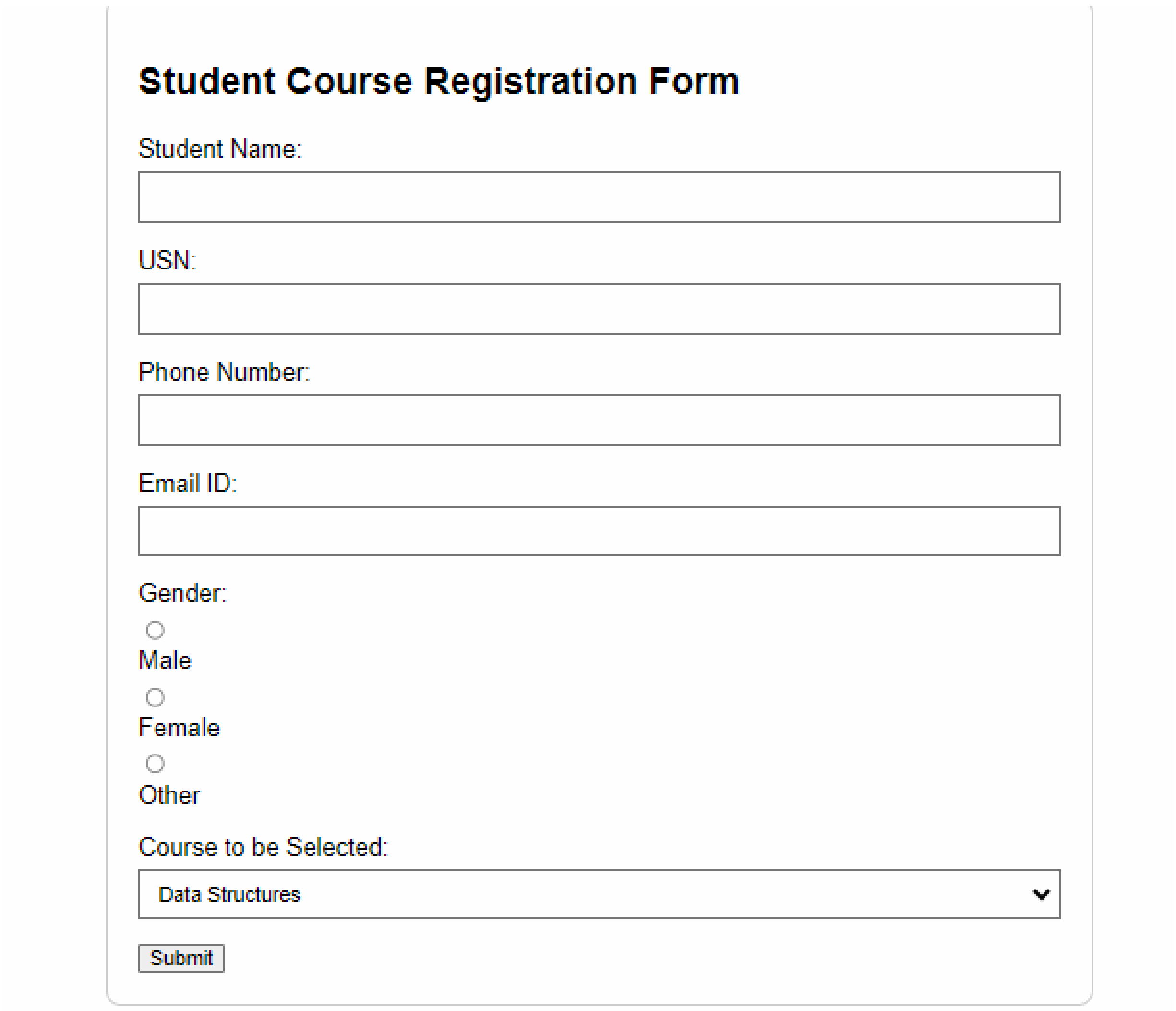
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Course Registration Form</title>
    <style>
        body {
            font-family: Arial, sans-serif;
        }
        .container {
            max-width: 600px;
            margin: auto;
            padding: 20px;
            border: 1px solid #ccc;
            border-radius: 10px;
        }
        .form-group {
            margin-bottom: 15px;
        }
        .form-group label {
            display: block;
            margin-bottom: 5px;
        }
        .form-group input, .form-group select {
            width: 100%;
            padding: 8px;
            box-sizing: border-box;
        }
        .form-group input[type="radio"] {
            width: auto;
        }
        .result {
            margin-top: 20px;
            padding: 20px;
            border: 1px solid #ccc;
            border-radius: 10px;
        }
    </style>
</head>
<body>
```

```
<div class="container">
    <h2>Student Course Registration Form</h2>
    <form id="registrationForm">
        <div class="form-group">
            <label for="name">Student Name:</label>
            <input type="text" id="name" name="name" required>
        </div>
        <div class="form-group">
            <label for="usn">USN:</label>
            <input type="text" id="usn" name="usn" required>
        </div>
        <div class="form-group">
            <label for="phone">Phone Number:</label>
            <input type="text" id="phone" name="phone" required>
        </div>
        <div class="form-group">
            <label for="email">Email ID:</label>
            <input type="email" id="email" name="email" required>
        </div>
    <div class="form-group">
      <label>Gender:</label>
      <input type="radio" id="male" name="gender" value="Male"
required>
      <label for="male">Male</label>
      <input type="radio" id="female" name="gender"
value="Female" required>
      <label for="female">Female</label>
      <input type="radio" id="other" name="gender" value="Other"
required>
      <label for="other">Other</label>
    </div>

    <div class="form-group">
      <label for="course">Course to be Selected:</label>
      <select id="course" name="course" required>
        <option value="Data Structures">Data Structures</option>
        <option value="Algorithms">Algorithms</option>
        <option value="Operating Systems">Operating
Systems</option>
        <option value="Database Systems">Database
Systems</option>
```

```
            <option value="Web Development">Web
Development</option>

          </select>

        </div>

        <button type="button"
onclick="displayDetails()">Submit</button>

    </form>


    <div id="result" class="result" style="display:none;">

        <h3>Registration Details</h3>

        <p><strong>Student Name:</strong>

          <span id="displayName"></span></p>


        <p><strong>USN:</strong> <span id="displayUsn"></span></p>

        <p><strong>Phone Number:</strong> <span
id="displayPhone"></span></p>

        <p><strong>Email ID:</strong> <span
id="displayEmail"></span></p>

        <p><strong>Gender:</strong> <span
id="displayGender"></span></p>

        <p><strong>Course Selected:</strong> <span
id="displayCourse"></span></p>

    </div>
```

```
</div>

<script>
    function displayDetails() {
        // Get form values
        const name = document.getElementById('name').value;
        const usn = document.getElementById('usn').value;
        const phone = document.getElementById('phone').value;
        const email = document.getElementById('email').value;
        const gender =
document.querySelector('input[name="gender"]:checked').value;
        const course = document.getElementById('course').value;

        // Display the values
        document.getElementById('displayName').textContent = name;
        document.getElementById('displayUsn').textContent = usn;
        document.getElementById('displayPhone').textContent = phone;
        document.getElementById('displayEmail').textContent = email;
        document.getElementById('displayGender').textContent = gender;
        document.getElementById('displayCourse').textContent = course;
        // Show the result div
        document.getElementById('result').style.display = 'block';
    }
```

Output:



Student Course Registration Form

Student Name:

USN:

Phone Number:

Email ID:

Gender:

○ Male

○ Female

○ Other

Course to be Selected:

Data Structures

Simple Web Server

9. Write a Node.js program to create a simple web server that responds with "Hello, Node.js!". Modify the server to respond with different messages based on the URL path (e.g., /about responds with "About Us", /contact responds with "Contact Us").

```
const http = require('http');
// Create an HTTP server
const server = http.createServer((req, res) => {
  // Set the response HTTP header with HTTP status and Content type
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  // Get the URL path
  const path = req.url;
  // Respond with different messages based on the URL path
  if (path === '/') {
    res.end('Hello, Node.js!');
  } else if (path === '/about') {
    res.end('About Us');
  } else if (path === '/contact') {
    res.end('Contact Us');
  } else {
    res.writeHead(404, { 'Content-Type': 'text/plain' });
    res.end('404 Not Found');
  }
});
// The server listens on port 3000
server.listen(3000, () => {
    console.log('Server running at http://localhost:3000/');
});
```

File System Operations

10. Write Node.js program that reads a text file and prints its contents to the console. Then, extend the program to create a new file and write some data into it.

Step 1: Reading a Text File

```
const fs = require('fs');
const path = require('path');
// Specify the path to the text file
const filePath = path.join(__dirname, 'input.txt');
// Read the file
fs.readFile(filePath, 'utf8', (err, data) => {
    if (err) {
        console.error('Error reading the file:', err);
        return;
    }
    console.log('File contents:');
    console.log(data);
});
```

Step 2: Creating and Writing to a New File

```
const fs = require('fs');
// Create and write to a file
fs.writeFile('example.txt', 'Hello, World!', (err) => {
  if (err) throw err;
  console.log('File created and written to!');

  // Read the file
  fs.readFile('example.txt', 'utf8', (err, data) => {
    if (err) throw err;

    console.log('File content:', data);

    // Append to the file
    fs.appendFile('example.txt', ' How are you?', (err) => {
      if (err) throw err;
      console.log('File updated!');

      // Read the updated file
      fs.readFile('example.txt', 'utf8', (err, updatedData) => {
        if (err) throw err;
        console.log('Updated file content:', updatedData);
        // Delete the file
        fs.unlink('example.txt', (err) => {
          if (err) throw err;
          console.log('File deleted!');
        });
      });
    });
  });
});
```

Database connection

11.Write a Node.js program using the mysql package to perform create database, create table, insert value, select from, update values, delete values on a mysql collection using student data

```
const mysql = require('mysql');
// Create a connection to the MySQL server
const connection = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'root'
```

```
});
// Connect to the MySQL server
connection.connect((err) => {
   if (err) {
      return console.error('Error connecting to the MySQL server:', err);
   }
   console.log('Connected to the MySQL server.');
   // Create a new database
   connection.query('CREATE DATABASE IF NOT EXISTS school',
(err, result) => {
      if (err) throw err;
      console.log('Database created or already exists.');
      // Use the new database
      connection.query('USE school', (err, result) => {
         if (err) throw err;
         // Create a new table
         const createTableQuery = `
            CREATE TABLE IF NOT EXISTS students (
               id INT AUTO_INCREMENT PRIMARY KEY,
               name VARCHAR(255) NOT NULL,
               age INT NOT NULL,
            grade VARCHAR(10) NOT NULL
        )
    `;
    connection.query(createTableQuery, (err, result) => {
        if (err) throw err;
        console.log('Table created or already exists.');
        // Insert values into the table
        const insertQuery = `
            INSERT INTO students (name, age, grade)
            VALUES ('John Doe', 18, 'A'),
                   ('Jane Smith', 20, 'B'),
                   ('Alice Johnson', 19, 'A')
        `;
        connection.query(insertQuery, (err, result) => {
            if (err) throw err;
            console.log('Values inserted.');
            // Select values from the table
            connection.query('SELECT * FROM students', (err, results)
=> {
                if (err) throw err;
                console.log('Selected values:');
console.log(results);

// Update values in the table
const updateQuery = `
   UPDATE students
   SET grade = 'A+'
   WHERE name = 'Jane Smith'
`;
connection.query(updateQuery, (err, result) => {
   if (err) throw err;
   console.log('Values updated.');

    //Delete values from the table
   const deleteQuery = `
      DELETE FROM students
      WHERE name = 'Alice Johnson'
   `;
   connection.query(deleteQuery, (err, result) => {
      if (err) throw err;
      console.log('Values deleted.');
```

```
                                // Close the connection
                                connection.end((err) => {
                                    if (err) throw err;
                                    console.log('Connection closed.');
                                });
                            });
                        });
                    });
                });
            });
        });
    });
});
```