# **Vision**

* Produce globally employable Computer Engineers having core values and aptitude to innovate.

# **Mission**

* Revise curricula regularly to incorporate relevant technology advances while also maintaining strong emphasis on fundamentals.
* Deliver quality technical education by regularly reforming policies,  systems and processes at all levels.
* Promote innovative and best practices at all levels and create an environment in which research and partnerships with industries flourish.
* Imbibe core values.
* Foster faculty and staff members to meet challenges.

# **Program Educational Objectives**

1. Make technical contribution to the design, development, and production of computing systems.
2. Engage in lifelong learning with leadership qualities, professional ethics, and soft skills.
3. Adapt state of the art development in the field of computer engineering.

# **Program Outcomes**

* Immediately after completing the bachelor degree in Computer Engineering from BVM Engineering College, our student will be able to:

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. 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.
3. 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.
4. 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.
5. 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.
6. 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.
7. 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.
8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. 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.
11. 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.
12. 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)**

1. Develop computer engineering solutions for specific needs in different domains applying the knowledge in the areas of programming, algorithms, hardware-interface, system software, computer graphics, web design, networking, and advanced computing.
2. Analyze and test computer software designed for diverse needs.
3. Pursue higher education.

# **Course Outcomes(COs)**

* After successful completion of the course, the students will be able to:
* Understand the concepts of WWW including web protocols and web browser architecture.
* Develop the static web pages using the HTML and CSS with different layouts as per need of applications.
* Construct and validate semi-structured database.
* Develop client side scripting using JavaScript and AngularJS.
* Develop dynamic web pages using server side scripting language such as a PHP with database connectivity.
* Develop real world applications.

# **Experiment 1**

* Develop static web page of resume using html tags:
* Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Chirag Rathod Resume</title>

</head>

<body bgcolor="#f0f0f0" text="#333333">

<h1 align="center">Chirag Rathod</h1>

<p style="text-align: center;">

<img src="Chirag.jpg" alt="" width="150" align="center">

</p>

<h3 align="center">

<i>Software Engineer</i>

</h3>

<hr>

<h2>Profile Info</h2>

<p>

Hello, I'm CHIRAG RATHOD. I'm software developer. I believe i am a very ambitious person who loves to work on

developing the websites and software. Versatile software engineer skilled in designing and implementing robust

solutions for complex challenges.

</p>

<hr>

<h2>Education</h2>

<p>

<ol type="A">

<li>

<ul>

<u>

<li>2019-2020</li>

<li>SSC Board(10th Class)</li>

</u>

<li>BAPS SVM, Bakrol</li>

<li>GSEB Percentile : 98.52</li>

</pre>

</ul>

</li>

<br>

<li>

<ul>

<li><u>2021-2022</u></li>

<li><u>HSC Board (12th Class)</u></li>

<li>BAPS SVM,Bakrol</li>

<li>GSEB Percentile : 98.7</li>

</ul>

</li>

<br>

<li>

<ul>

<li><u>2022-2026</u></li>

<li>BTech in Computer Engineering</li>

<li>Birla Vishwakarma Mahavidyalaya, Anand</li>

</ul>

</li>

</ol>

</p>

<hr>

<h2>Certificate</h2>

<p>

<ul>

<li>

Fundamental Of Digital Marketing

<ul>

<li><del>Issued on : March 2023</del></li>

<li>Issued by

<a href="https://wwww.google.com">Google Learn Digital</a>

</li>

<li><ins>Cerdential ID : ZU2 J4G 589</ins></li>

</ul>

</li>

<li>

Cyber Security Internship Online

<ul>

<li>

6 Weeks cyber Security Internship in collaboration with AICTE and Edunet Foundation !

</li>

<li>

Enhanced my skills in ethical hacking, netwrok security, and incident responce.

</li>

</ul>

</li>

</ul>

</p>

<hr>

<h2>My Skills & Expertise</h2>

<p>

<ul>

<li>

Programming Language :

<pre><b>Java</b>,<em>JavaScript</em>, <strong>Python</strong></pre>

</li>

<li> Web Technologies : <small>HTML, CSS, React</small></li>

<li>Algorithm and Data Structure</li>

<li>Cyber Security</li>

<li>Code review</li>

</ul>

</p>

<hr>

<h2>Relevant Course</h2>

<p>

<ol>

<li>Programming with C</li>

<li>Object Oriented with C++</li>

<li>Object Oriented with Java</li>

<li>Programming with Python</li>

</ol>

</p>

<hr>

<h2>Contact</h2>

<p>

<ul>

<li>

<span>

Phone :

</span>

<a href="tel : +91 9558161280">

+91 9558161280

</a>

</li>

<li>

<span> Email : </span><a href="mailto:chiragrathod9987@gmail.com">

chiragrathod9987@gmail.com</a>

</li>

<li> <span>Linkedin : </span><a href="https://www.linkedin.com/in/chiragrathod25" target="\_blank">linkedin.com/chiragrathod25</a>

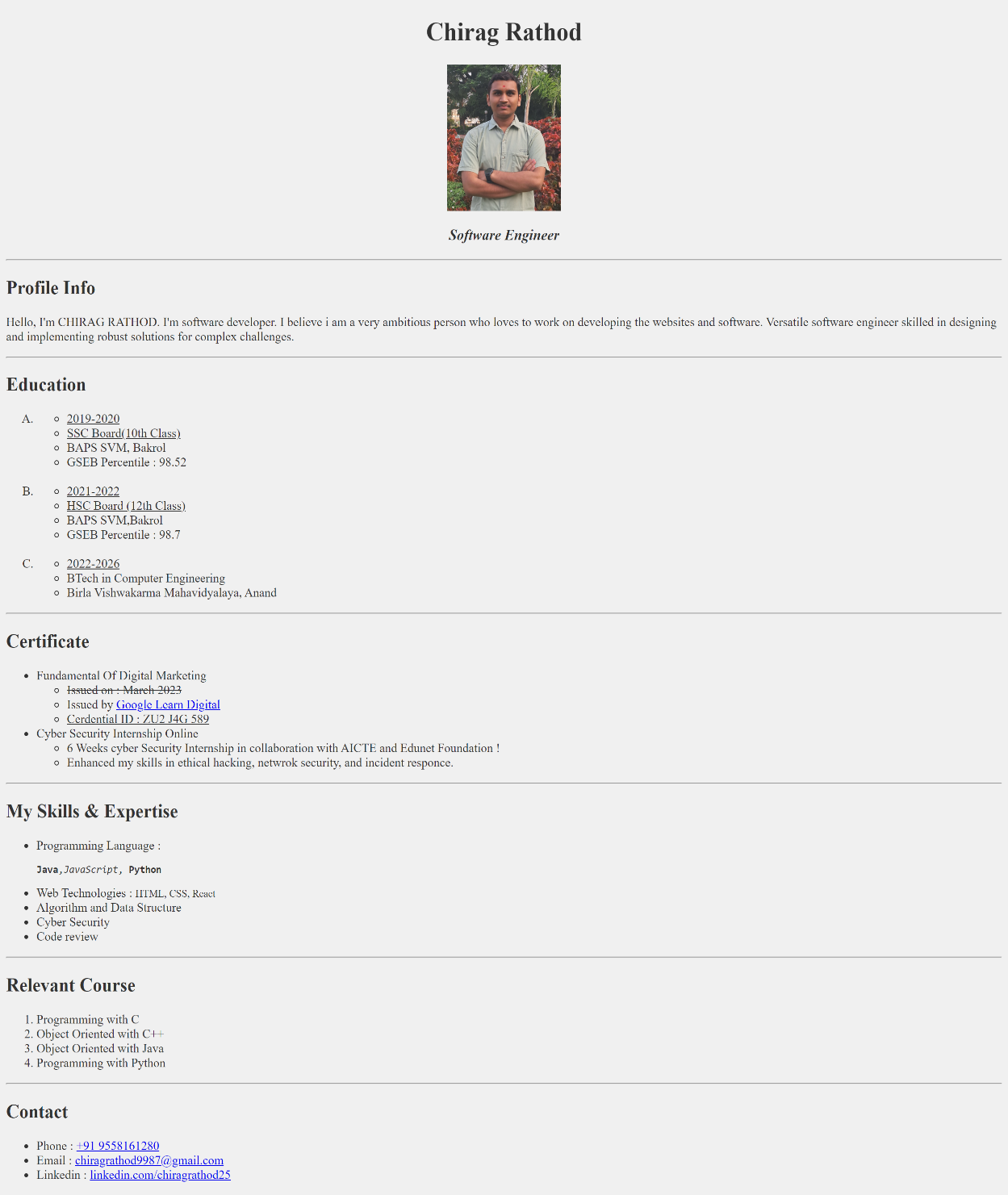
</li>

</ul>

</p>

</body>

</html>

* Output:  
  

*Figure 1 Resume*

# **Experiment 2**

* Develop static web page of irregular tables using the HTML Tags
* Code :

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Table designing</title>

</head>

<body>

<table border="1" cellspacing="3" cellpadding="5">

<tr>

<th rowspan="2" colspan="2">

Today's Opinion Poll Question

</th>

<th colspan="3">

Political Party

</th>

</tr>

<tr>

<th>Democrat</th>

<th>Republican</th>

<th>Independent</th>

</tr>

<tr>

<th rowspan="3" width="150" height="150">

Do you favor or oppose increasing the minimum wage?

</th>

<th>Favor</th>

<th>70%</th>

<th>35%</th>

<th>55%</th>

</tr>

<tr>

<th>Oppose</th>

<th>25%</th>

<th>60%</th>

<th>30%</th>

</tr>

<tr>

<th>Unsure</th>

<th>5%</th>

<th>5%</th>

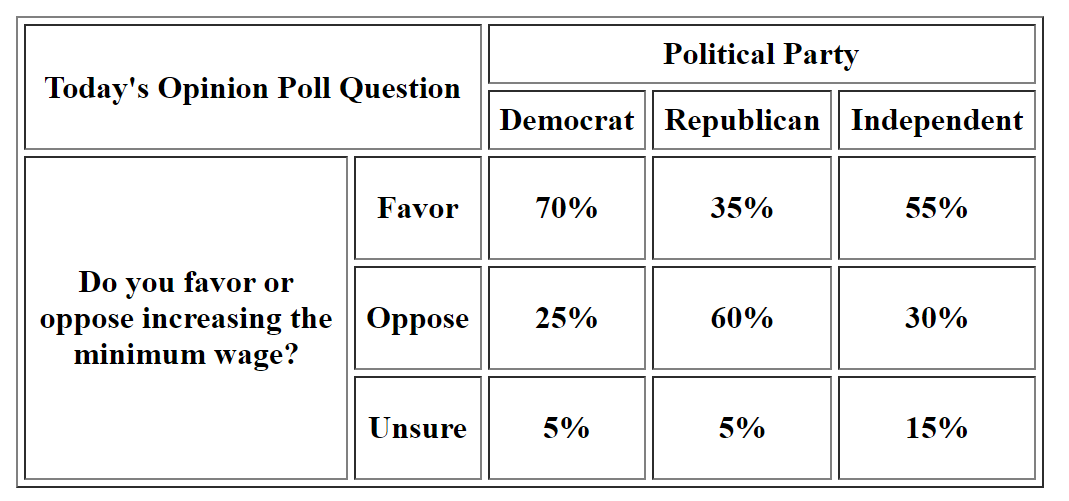
<th>15%</th>

</tr>

</table>

</body>

</html>

* Output: 

*Table 1 HTML Table*

* Develop static web page of Registration form using Following HTML tags:
* Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Registration</title>

</head>

<style>

th{

text-align: left;

padding: 3px;

}

#header{

text-align: center;

}

</style>

<body>

<form action="">

<table border="2"

width="50%" align="center">

<tr >

<th colspan="2" id="header"><h2>SEM 4 Registration Form</h2></th>

</tr>

<tr>

<th width="40%" ><label for="Academic">Select Academic Year : </label></th>

<th>

<select name="Academic" id="Academic">

<option value="2022-2023">2022-2023</option>

<option value="2023-2024">2023-2024</option>

</select>

</th>

</tr>

<tr>

<th width="40%" ><label for="semesterType">Semester Type </label></th>

<th>

<select name="semesterType" id="semesterType">

<option value="ODD">ODD</option>

<option value="EVEN">EVEN</option>

</select>

</th>

</tr>

<tr>

<th width="40%" ><label for="semester">Select Semester</label></th>

<th>

<select name="semester" id="semester">

<option value="1">1</option>

<option value="2">2</option>

<option value="3">3</option>

<option value="4" selected>4</option>

<option value="5">5</option>

<option value="6">6</option>

<option value="7">7</option>

<option value="8">8</option>

</select>

</th>

</tr>

<tr>

<th width="40%" ><label for="StartDate">Start Date </label></th>

<th>

<input type="date" name="StartDate" id="StartDate">

</th>

</tr>

<tr>

<th width="40%" ><label for="ID">ID</label></th>

<th>

<input type="text" name="ID" id="ID" placeholder="Enter your ID ">

</th>

</tr>

<tr>

<th width="40%" ><label for="Password">Password </label></th>

<th><input type="password" name="Password" id="Password" placeholder="Enter your Password"></th>

</tr>

<tr>

<th width="40%" ><label for="Name">Name </label></th>

<th><input type="text" name="Name" id="Name" placeholder="Enter your Name"></th>

</tr>

<tr>

<th width="40%" ><label for="Gender">Gender </label></th>

<th>

<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>

</th>

</tr>

<tr>

<th width="40%" ><label for="Address">Address </label></th>

<th><textarea name="Address" id="Address" cols="30" rows="3"></textarea></th>

</tr>

<tr>

<th width="40%" ><label for="Subject">Subject </label></th>

<th>

<input type="checkbox" name="DSA" id="DSA">

<label for="DSA">Data Structure</label>

<br>

<input type="checkbox" name="WEB" id="WEB">

<label for="WEB">Web Technologies</label>

<br>

<input type="checkbox" name="Maths" id="Maths">

<label for="Maths">Maths</label>

<br>

<input type="checkbox" name="Science" id="Science">

<label for="Science">Science</label>

<br>

</th>

</tr>

<tr>

<th width="40%" ><label for="fees">Fees </label></th>

<th><input type="text" name="fees" id="fees" placeholder="50/-"></th>

</tr>

<tr>

<th width="40%" ><label for="reset">Reset Form </label></th>

<th><input type="reset"></th>

</tr>

<tr>

<th width="40%" ><label for="submit">Submit </label></th>

<th><input type="submit"></th>

</tr>

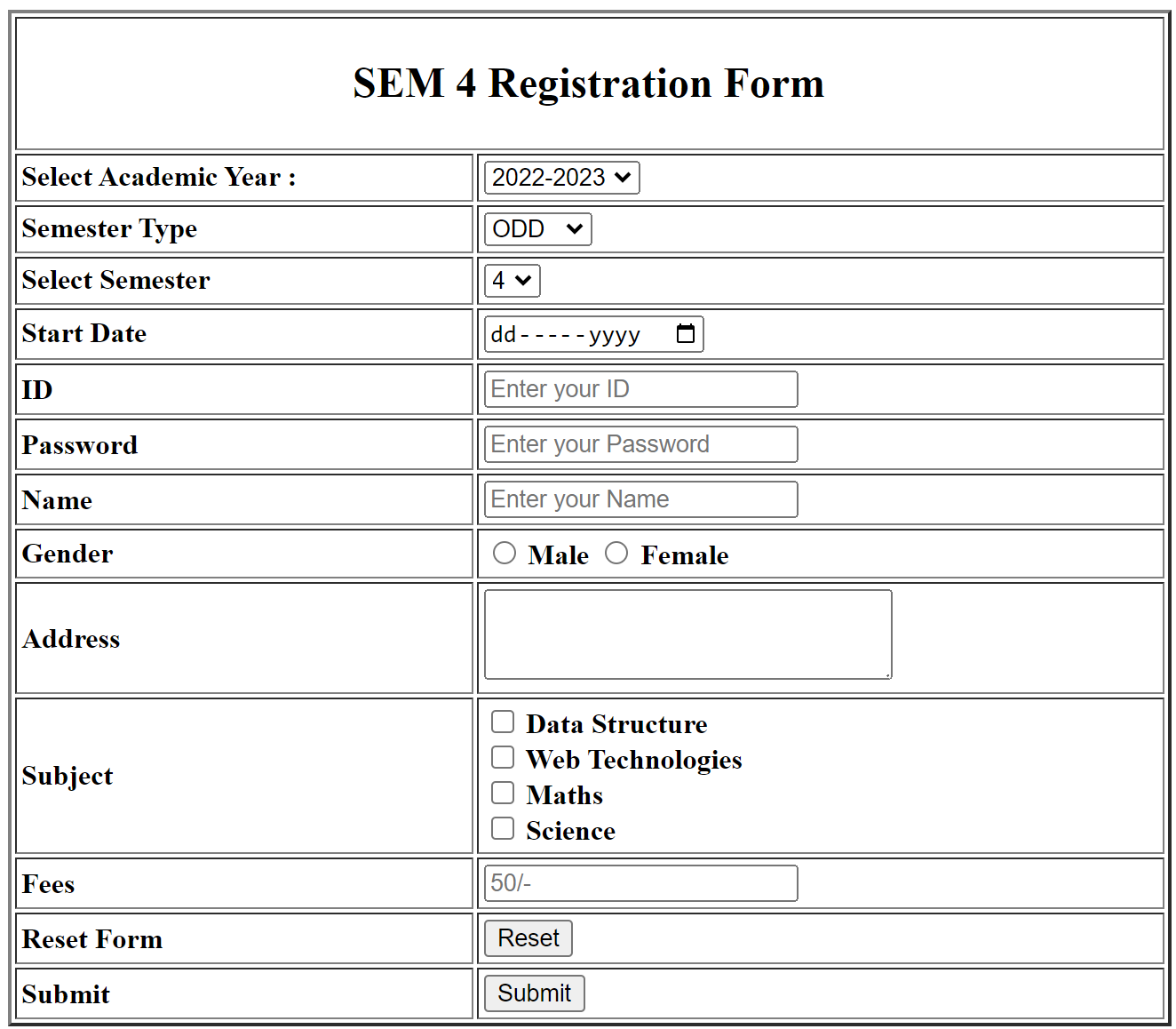
</table>

</form>

</body>

</html>

* Outcome:



*Figure 2 Registration Form*

# **Experiment 3**

* Develop a static web page of a Music player using the following HTML 5.0 tags
* Article, aside, footer, header, HTML Canvas, Media Tags
* Code :

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

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

<title>Media Player</title>

</head>

<style>

body{

display: flex;

align-items: center;

justify-content: center; }

p {

font-weight: bold;

}

section {

border: 2px solid black;

width: auto;

display: inline-block;

padding: 5px;

}

header {

font-size: 28px; }

footer {

font-size: 18px; }

</style>

<body>

<section>

<header align="center">Media Player</header>

<hr />

<article>

<ol type="1">

<li>

<aside>

<h4>song 1</h4>

<audio controls><source src="horse.mp3" type="audio/mpeg" /></audio>

</aside>

</li>

<li>

<aside>

<h4>song 2</h4>

<audio controls><source src="horse.mp3" type="audio/mpeg" /></audio>

</aside>

</li>

<li>

<aside>

<h4>song 3</h4>

<audio controls><source src="horse.mp3" type="audio/mpeg" /></audio>

</aside>

</li>

</ol>

</article>

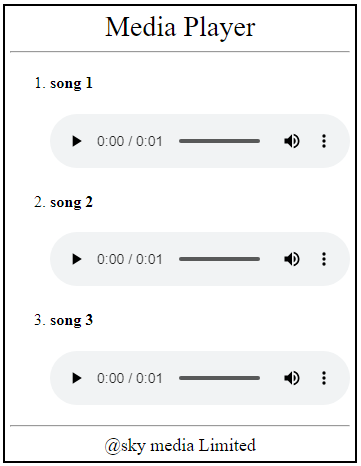
<hr /><footer align="center">@sky media Limited</footer>

</section>

</body>

</html>

* Output:



*Figure 3 Media Player*

# **Experiment 4**

* Cascading Style Sheets (CSS) : Create Mypage.html using an external style sheet.
* Adding CSS: External, Internal and Inline

1. Compare class and ID selectors in CSS with proper examples.
   * Class selectors and ID selectors in CSS both allow you to target specific HTML elements for styling, but they have key differences.
   * Class selectors begin with a dot (.) followed by the class name and can be applied to multiple elements in the HTML document.
     + Example: .button { color: blue; }
   * ID selectors begin with a hash (#) followed by the ID name and are used to uniquely identify a single element in the HTML document.
     + Example: #header { background-color: gray; }
2. What are the different types of cascading style sheets? Elaborate with an example.
   * External CSS: External CSS is stored in separate CSS files and linked to HTML documents using the <link> tag.
     + Example: <link rel="stylesheet" type="text/css" href="styles.css">
   * Internal CSS: Internal CSS is defined within the HTML document using the <style> tag within the <head> section.
     + Example: <style> body { font-family: Arial, sans-serif; } </style>
   * Inline CSS: Inline CSS is applied directly to individual HTML elements using the style attribute.
     + Example: <p style="color: red;">This is a paragraph.</p>
3. Develop web page contains a table, div, text, menus, pseudo-class selector .style the page using external CSS. Apply margin, border, bgcolor, list-items, font-styles, etc

* Code:
* HTML:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

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

<link rel="stylesheet" href="Experiment4.css" />

<title>Mypage.html</title>

</head>

<body>

<header>My Web Page</header>

<section>

<div class="para" id="para1">

<h2>Paragraph 1</h2>

<p>

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Porro,

ducimus sit inventore quaerat aspernatur fugit beatae maxime, nostrum

adipisci sunt temporibus debitis a obcaecati consectetur? Quia ea

modi, odit quisquam voluptates amet sunt error eligendi labore maxime

dignissimos doloremque quibusdam, possimus cum molestias iusto

nesciunt! Totam quae eum eaque modi!

</p>

</div>

<div class="para" id="para2">

<h2>Paragraph 2</h2>

<p>

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Adipisci

quas maxime illo blanditiis. Nisi obcaecati voluptate rem praesentium

fugiat. Distinctio?

</p>

</div>

</section>

<section id="studentList">

<h3>List of Student</h3>

<ol>

<li>22CP001</li>

<li>22CP002</li>

<li>22CP003</li>

</ol>

</section>

<section id="links">

<ul>

<li>

<aside>

<span>Click here to Visit Experiment 3 : </span>

<a href="Experiment 3.html" target="\_blank">Experiment 3</a>

</aside>

</li>

<li>

<aside>

<span>Click here to Visit Experiment Resume : </span>

<a href="resume.html" target="\_blan k">Resume</a>

</aside>

</li>

</ul>

</section>

</body>

</html>

* CSS :

\*{

margin: 0;

padding: 0;

box-sizing: border-box;

}

header{

margin-top: 1rem;

width: 100%;

font-size: 30px;

font-weight: bolder;

text-align: center;

}

section{

margin-top:15px ;

padding: 2rem;

}

.para{

margin: 1rem;

font-family: Arial, Helvetica, sans-serif;

background-color: rgb(171, 253, 226);

padding: 0.3rem;

border-radius: 10px;

}

.para \*{

margin:1rem ;

}

.para h2{

text-decoration: underline;

font-style: unset;

}

#studentList{

border: 1px solid black;

margin-top:15px ;

}

#studentList li{

margin:0.5rem;

}

#studentList li:first-child{

margin-top: 1rem;

}

#links ul li{

margin: 3px;

}

* Output:



*Figure 4 Experiment-4 External CSS*

1. Use of pseudo-class with the anchor tag. Give a proper example.
   * a {

color: blue;

text-decoration: none;

}

* + a:hover {

color: red;

}

* + a:visited {

color: purple;

}

# **Experiment 5**

* Cascading Style Sheets (CSS) @import CSS, Apply style to ordered List, Descendant Selectors, child selectors, first-line, first-letter, :before, :after pseudo element, positioning, BOX Model

1. Create below Table of Content using style to ordered List, Descendant Selectors, child selectors, :after, :before
   * Code :

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Counter</title>

<style>

body {

counter-reset: first;

}

body>ol>li {

counter-reset: second;

}

body>ol>li>ol>li {

counter-reset: third;

}

bodu>ol>li>ol>li>ol>li{

counter-reset: four;

}

body>ol>li::before {

content: "Chapter "counter(first)".";

counter-increment: first;

}

body>ol>li>ol>li::before {

content: counter(first)"." counter(second)".";

counter-increment: second;

}

body>ol>li>ol>li>ol>li::before {

content: counter(first)"." counter(second)"." counter(third);

counter-increment: third;

}

body>ol>li>ol>li>ol>li>ol>li::before{

content: counter(first)"."counter(second)"."counter(third)"."counter(four);

counter-increment: four;

}

li{

list-style-type:none;

}

</style>

</head>

<body>

<span>Title : Web Technologies</span>

<br>

<span>Table of content</span>

<ol type="1">

<li>html</li>

<li>css

<ol type="1">

<li>Introduction </li>

<li>Adding Style sheet

<ol type="1">

<li>extrenal

<ol type="1">

<li>extrenal style sheets are useful</li>

</ol>

</li>

</ol>

</li>

<li>browser incompatibility </li>

</ol>

</li>

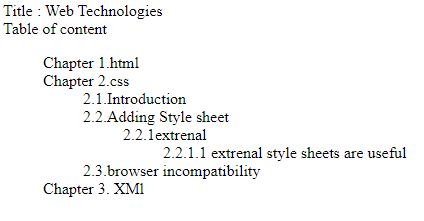
<li>XMl</li>

</ol>

</body>

</html>

* + Output :



*Figure 5 CSS Counter*

1. Create three divisions’ header, left side menu and main content. Apply style to each division using styles. Use positioning attribute for each division, i.e. static, relative, fixed, absolute, and sticky. And also use Z-index for overlapping division.
   * Code:
   * HTML:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Position css</title>

<link rel="stylesheet" href="02\_style.css">

</head>

<body>

<div class="header">Header</div>

<div class="left-menu">Left Menu</div>

<div class="main-content">Main Content</div>

</body>

</html>

* + CSS:

.header {

background-color: #333;

color: #fff;

height: 50px;

width: 100%;

position: fixed;

top: 0;

left: 0;

z-index: 3;

}

.left-menu {

background-color: #ddd;

width: 200px;

height: 100%;

position: fixed;

top: 50px;

left: 0;

z-index: 2;

}

.main-content {

background-color: #f0f0f0;

width: calc(100% - 200px);

height: 100%;

position: absolute;

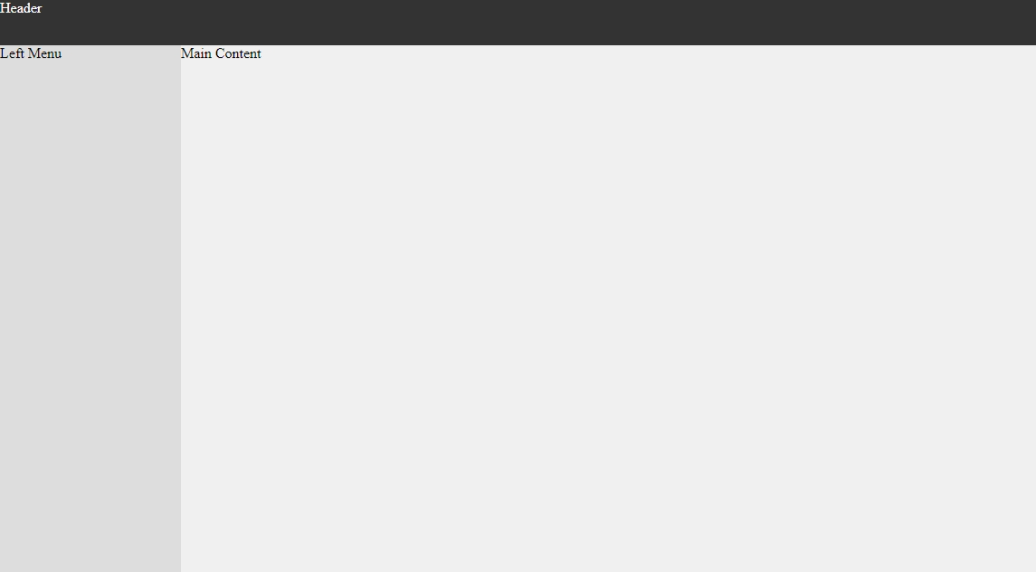
top: 50px;

left: 200px;

z-index: 1;

}

* + Output:



*Figure 6 Position CSS*

1. Develop BOX model using CSS Content, Padding, Border and Margin for your Project definition.
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Box Model</title>

<style>

.box{

border: 2px solid black;

background-color: antiquewhite;

width: 200px;

margin: 1rem;

}

.box-model-example{

border: 2px dotted black;

margin: 2rem;

padding: 1rem;

}

</style>

</head>

<body>

<div class="box">

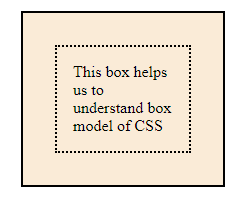
<div class="box-model-example">This box helps us to understand box model of CSS</div>

</div>

</body>

</html>

* + Output:



*Figure 7 CSS Box Model*

1. Use of CSS3 and Bootstrap framework.
   * CSS3 and Bootstrap are powerful tools for enhancing the design and functionality of web projects.
   * **CSS3**:Here are some key features of CSS3
   * Transitions and Animations
   * Transformation : rotate, scale, skew, translate to element
   * Gradients and Shadows : It provides support for creating gradients and shadow directly within style sheet, reducing the need for image-based effect.
   * Flexbox and Grid Layout : It allows for more responsive and complex layouts without relying heavily on floats or positioning.
   * Media Queries: It allows style based on the device’s screen size, resolution, and orientation.
   * **Bootstrap** Framework: Bootstrap is a popular front-end framework that simplifies the process of building responsive and mobile-first websites.
   * Grid System
   * Pre-styled Components: Bootstrap comes with a variety of pre-styled components such as buttons, forms, navigation bars, and alerts.
   * Responsive Design: Responsive design for different devices.
   * Extensive Documentation and Community Support: Bootstrap offers comprehensive documentation and a large community of users and contributors, making it easy to find resources, tutorials, and solutions to common problems.

# **Experiment 6**

* JavaScript: use of document.writeln(), comments, variables, local vs Global, type of operators, arithmetic, assignment, relational, logical & Bitwise Operators, Control structures, conditional statements, array, user define function, object, use if For/in.

1. Write a JavaScript program to demonstrate use of Operators
   * Code:

<!DOCTYPE html>

<html>

<head>

<title>Experiment 6</title>

</head>

<body>

<script>

let sum=2+3

document.writeln("2 + 3 ;= ",sum, typeof(sum),"</br>")

let clg="Hello"+" BVM"

document.writeln("\"Hello\" + \"bvm\" ;= ",clg,typeof(clg),"</br>")

let bits=2+"MB"

document.writeln("2 + \"MB\" ;= ",bits,typeof(bits),"</br>")

let a=3-2

document.writeln("3 - 2 ;= ",a,typeof(a),"</br>")

let b=3-"MB"

document.writeln("3 - \"MB\" ;= ",b,typeof(b),"</br>")

let multi=3\*2

document.writeln("3 \* 2 ;= ",multi,typeof(multi),"</br>")

let devide=7/3

document.writeln("7 / 3 ;= ",devide,typeof(devide),"</br>")

let infdevide=7/0

document.writeln("7 / 0 ;= ",infdevide,typeof(infdevide),"</br>")

let modu=10%3

document.writeln("10 % 3 ;= ",modu,typeof(modu),"</br>")

let zero\_modu=10%0

document.writeln("10 % 0 ;= ",zero\_modu,typeof(zero\_modu),"</br>")

let x=1;

document.writeln("x ;= ",x,typeof(x),"</br>")

x++

document.writeln("x++ ;= ",x,typeof(x),"</br>")

let y=++x

document.writeln("y=++x; y=",y," x=",x,typeof(x),"</br>")

y=x++

document.writeln("y=x++; y=",y," x=",x,typeof(x),"</br>")

x=5

document.writeln("x ;= ",x,typeof(x),"</br>")

x--

document.writeln("x-- ;= ",x,typeof(x),"</br>")

y=--x

document.writeln("y=--x; y=",y," x=",x,typeof(x),"</br>")

y=x--

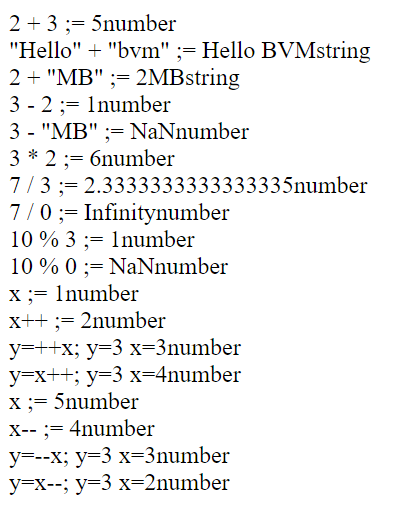
document.writeln("y=x--; y=",y," x=",x,typeof(x),"</br>")

</script>

</body>

</html>

* + Output:



*Figure 8 JavaScript Operators*

1. Write a JavaScript program to use bitwise Operators
   * Code:

<!DOCTYPE html>

<html>

<head>

<title>Experiment 6</title>

</head>

<body>

<script>

let a=5&2 //AND

document.writeln("5 & 2 ;= ",a,"</br>")

let b=5|2 //OR

document.writeln("5 | 2 ;= ",b,"</br>")

let c=~5 //NOT

document.writeln("~5 ;= ",c,"</br>")

let d=5^1 //XOR

document.writeln("5^1 ;= ",d,"</br>")

let e=5<<1 //Zero fill left shift

document.write("5<<1 ;= ",e,"</br>")

let f=5>>1 //Signed right shift

document.write("5>>1 ;= ",f,"</br>")

let g=5>>>1 //zero fill right shift

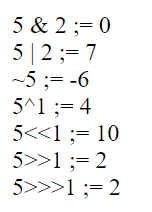
document.write("5>>>1 ;= ",g,"</br>")

</script>

</body>

</html>

* + Output:



*Figure 9 Bitwise Operators*

1. Write a JavaScript program to display first N Fibonacci number
   * Code:

<!DOCTYPE html>

<html>

<head>

<title>Experiment 6</title>

</head>

<body>

<script>

let n = 10

let a = 0

let b = 1

let c

var i;

for (i = 0; i < n; i++) {

c = a + b

document.writeln(a, " ")

a = b;

b = c;

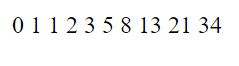
}

</script>

</body>

</html>

* + Output:



*Figure 10 Fibonacci Numbers*

1. Write JavaScript to finding all the prime number between user give ranges.
   * Code:

<!DOCTYPE html>

<html>

<head>

<title>Experiment 6</title>

</head>

<body>

<script>

let a = 3

let b = 23

let i = a;

for (i; i <= b; i++) {

flag = false

for (var j = 2; j < i; j++) {

if (i % j == 0) {

flag = true

}

}

if (!flag) {

document.writeln(i, " ")

}

}

</script>

</body>

</html>

* + Output:



*Figure 11 Prime Numbers*

1. Write JavaScript to use arrays methods e.g. reverse, concate, sort, join
   * Code:

<!DOCTYPE html>

<html>

<head>

<title>Experiment 6</title>

</head>

<body>

<script>

var friends=["Khushii",6,"Himanshu","Tulsi"]

var country=["India","Russia"]

document.writeln("Original Array : [",friends,"]","</br>")

friends.reverse()

document.writeln("Reversed Array : [",friends,"]","</br>")

friends.sort()

document.writeln("Sorted Array : [",friends,"]","</br>")

const concats\_array=friends.concat(country)

//concate method will concate two arrays and return a new array

document.writeln("Concateed Array :[" ,concats\_array,"]","</br>")

const new\_array=friends.join("-")

//join method will convert all array element into string and join all of them by '-'

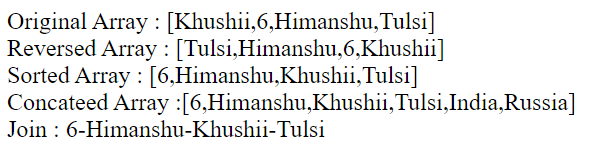
document.writeln("Join : ",new\_array)

</script>

</body>

</html>

* + Output:



*Figure 12 Array Methods*

1. Write a JavaScript program to sort (ascending, descending) the array of N elements
   * Code:

<!DOCTYPE html>

<html>

<head>

<title>Experiment 6</title>

</head>

<body>

<script>

function display(arr){

for(var i=0;i<arr.length;i++){

document.writeln(arr[i]," ")

}

}

function sort(arr){

for(var i=0;i<arr.length;i++){

for(var j=i+1;j<arr.length;j++){

if(arr[i]>arr[j]){

var temp=arr[i]

arr[i]=arr[j]

arr[j]=temp

}

}

}

}

var arr=[89,43,1,54,34,87,12,76,4,3,78]

document.writeln("Original Array : [")

display(arr)

document.writeln("]","</br>")

sort(arr)

document.writeln("Sorted Array : [")

display(arr)

document.writeln("]","</br>")

</script>

</body>

</html>

* + Output:



*Figure 13 Array Sorting*

# **Experiment 7**

1. Write a JavaScript to create a button. On click on the button a new window should open up.
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

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

<title>Button New Page</title>

<style>

.box {

width: 300px;

height: 100px;

display: grid;

justify-content: center;

align-content: center;

}

</style>

</head>

<body>

<script>

function newpage() {

location.href = "https://www.google.com";

}

</script>

<div class="box">

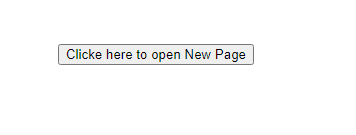
<button onclick="newpage()">Clicke here to open New Page</button>

</div>

</body>

</html>

* + Output:
  + When user click on button, it will open mentioned website in code



*Figure 14 New Window Open*

1. Write a JavaScript to get details of location object
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

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

<title>location Object details</title>

</head>

<body>

<p id="demo"></p>

<script>

for (i in location) {

document.getElementById("demo").innerHTML +=

i + ":" + location[i] + "</br>";

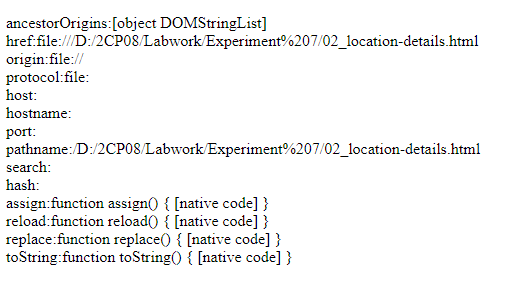
}

</script>

</body>

</html>

* + Output:



*Figure 15 Location Object*

1. Write a JavaScript to get details of navigator object
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

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

<title>Navigator</title>

</head>

<body>

<p id="demo"></p>

<script>

for (i in navigator) {

document.getElementById("demo").innerHTML +=

i + ":" + navigator[i] + "</br>";

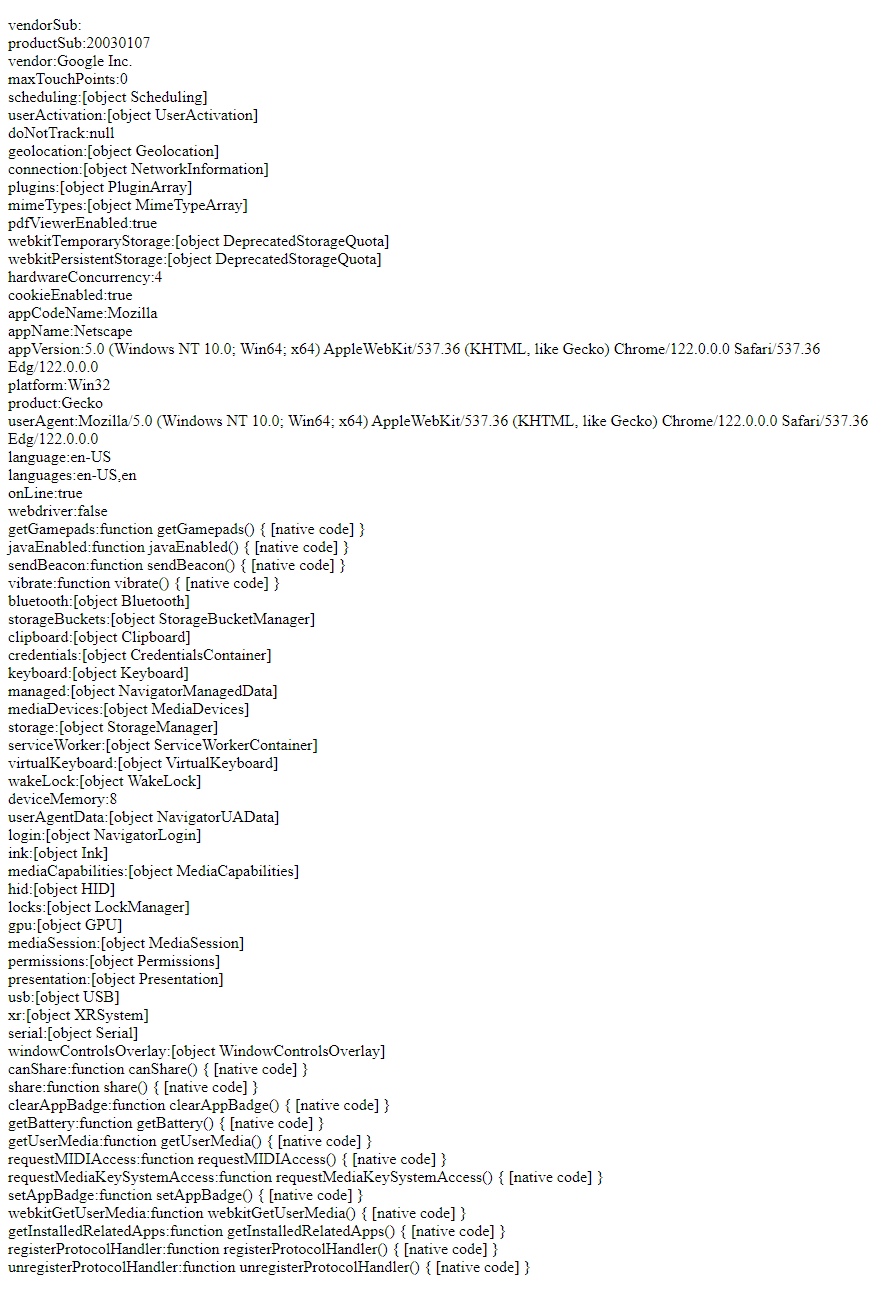
}

</script>

</body>

</html>

* + Output:



*Figure 16 Navigator Object*

1. Write a JavaScript to retrieve browser name, according browser name open specific version of index page. i.e. chrome\_version.html, edge\_version.html
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

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

<title>Browser Specific Page</title>

</head>

<script>

function getBrowserName() {

var userAgent = navigator.userAgent;

var browserName = "";

if (userAgent.includes("OPR")) {

browserName = "OPR";

} else if (userAgent.includes("Edg")) {

browserName = "Edge";

}

else if(userAgent.includes("Safari")){

browserName = "Safari";

}else {

browserName = "Chrome";

}

if (browserName == "OPR") {

location.href = "4\_OPR.html";

} else if (browserName == "Edge") {

location.href = "4\_EDGE.html";

} else if(browserName == "Safari"){

location.href = "4\_Safari.html";

}

}

</script>

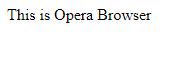
<body onload="getBrowserName()">

This is Chrome Page

</body>

</html>

* + Output:





*Figure 17 Different Page for different Browsers using JavaScript*

1. Write a JavaScript to get details of History object
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

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

<title>History Object details</title>

</head>

<body>

<p id="demo"></p>

<script>

for (i in history) {

document.getElementById("demo").innerHTML +=

i + ":" + history[i] + "</br>";

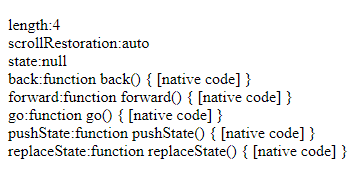
}

</script>

</body>

</html>

* + Output:



*Figure 18 History Object*

1. Write a JavaScript to Work with change event that occurs when a different item in the selection list is selected. Use innerHTML.
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>List Selection</title>

</head>

<body>

<select name="country" id="country" autocomplete="off">

<option value="country">Select your country</option>

<option value="India">India</option>

<option value="USA">USA</option>

<option value="Canada">Canada</option>

</select>

<select name="state" id="state" autocomplete="off">

<option>Select Country first</option>

</select>

<script>

const country = document.getElementById('country');

let state=document.getElementById('state');

country.addEventListener('change',function(){

state.innerHTML='';

if(country.value==='country'){

state.innerHTML=`<option>Select Country first</option>`;

}

else if(country.value==="India"){

let states=['Maharashtra','Gujarat','Rajasthan','Punjab','Kerala'];

for(i in states){

state.innerHTML+=`<option>${states[i]}</option>`

}

}

else if(country.value==='USA'){

let states=['California','Florida','Texas','New York','Washington'];

for(i in states){

state.innerHTML+="<option>"+states[i]+"</option>";

}

}

else if(country.value==='Canada'){

let states=['Ontario','Quebec','British Columbia','Alberta','Manitoba'];

for(i in states){

let option=document.createElement('option');

option.value=states[i];

option.innerHTML=states[i];

state.appendChild(option);

}

}

});

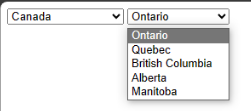
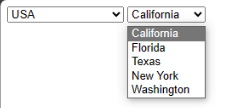
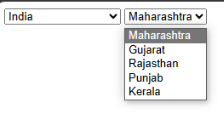
</script>

</body>

</html>

* + Output:





*Figure 19 Change in Select tag Options according to Selected Country*

1. Write a JavaScript to retrieve mouse location and display Unicode character and its key code enter from keyboard using event object.
   * Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

<title>retrive location and display Unicode character</title>

</head>

<body>

</body>

<script>

const mousePara = document.createElement("p");

document.querySelector("body").appendChild(mousePara);

const mouseHeading = document.createElement("h1");

mouseHeading.innerHTML = "Mouse Location";

mousePara.appendChild(mouseHeading);

const x\_position = document.createElement("span");

const y\_position = document.createElement("span");

mousePara.appendChild(x\_position);

mousePara.appendChild(y\_position);

document.querySelector("html").addEventListener("click", function (e) {

console.log(e);

x\_position.innerHTML = `X : ${e.clientX} `;

y\_position.innerHTML = `Y : ${e.clientY} `;

});

//keyboard key js

const keyBoardKey = document.createElement("p");

document.querySelector("body").appendChild(keyBoardKey);

const keyHeading = document.createElement("h1");

keyHeading.innerHTML = "Key Details";

keyBoardKey.appendChild(keyHeading);

const key = document.createElement("p");

const unicodeKey = document.createElement("p");

keyBoardKey.appendChild(key);

keyBoardKey.appendChild(unicodeKey);

document.querySelector("html").addEventListener("keydown", function (e) {

console.log(e)

key.innerHTML = `Key Pressed : ${e.key}`;

unicodeKey.innerHTML = `Unicode Charactar : ${e.key.charCodeAt(0)}`;

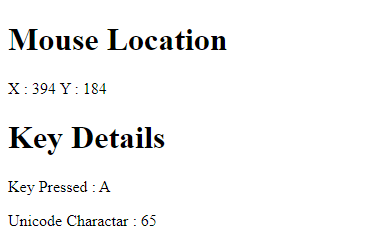
e.preventDefault(); //to prevent default behaviour of special keys

});

</script>

</html>

* + Output:



*Figure 20 Mouse Pointer and Key Unicode*

1. Write a JavaScript to validate registeryourself.html page.

* Give alert message if data is blank.
* Validate Email, name, password, mobile filed.
  + Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

<title>Form Validation</title>

<style>

label {

display: inline-block;

min-width: 130px;

margin-bottom: 10px;

}

</style>

</head>

<body>

<form action="" name="myForm" onsubmit="return validateForm(this)" method="post" >

<label for="fname">First Name : </label>

<input type="text" id="fname" name="fname" />

<br />

<label for="lname">Last Name : </label>

<input type="text" id="lname" name="lname" />

<br />

<label for="email">Email : </label>

<input type="text" id="email" name="email" />

<br />

<label for="mobileNumber">MobileNumber : </label>

<input type="text" id="mobileNumber" name="mobileNumber" />

<br />

<label for="userName">UserName : </label>

<input type="text" id="userName" name="userName" />

<br />

<label for="password">Password : </label>

<input type="password" id="password" name="password" />

<br />

<input type="submit" />

<input type="reset" />

</form>

</body>

<script>

function validateForm(form) {

if (form.fname.value === "") {

alert("please enter First Name");

form.fname.focus();

return false;

} else if (form.lname.value === "") {

alert("please enter Last Name");

form.lname.focus();

return false;

}

//email validation

const email= /^\w+@[a-zA-z]+\.[a-zA-z]{2,4}$/

if(!email.test(form.email.value)){

alert('please enter valid email');

form.email.focus();

return false;

}

//mobile Number

const mobileNumber=/^[1-9]\d{9}$/

if(!mobileNumber.test(form.mobileNumber.value)){

alert('please enter valid mobile number');

form.mobileNumber.focus();

return false;

}

if (form.userName.value === "") {

alert("please enter Username :");

form.userName.focus();

return false;

}

//password validation

const password=/^\w{5,8}$/

if (!password.test(form.password.value)) {

alert("please enter password (5-8 charactar)");

form.password.focus();

return false;

}

alert('Form Successfully Submitted');

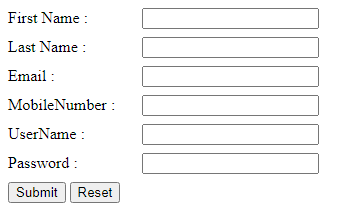
return true;

}

</script>

</html>

* + Output:
  + This form will give alert when input field is empty.
  + It generates alert when regular expression of email, password and mobile Number is not satisfied.



*Figure 21 Form Validation*

# **Experiment 8**

* Consider the following XML document fragment.
* The root element is "Employee\_Info" As the content for "Employee\_Info," "Employee" occurs 0 or more times As contents of "Employee," "Name," "Department," "Telephone," and "Email" elements occur once in respective order "Name," "Department," "Telephone," and "Email" contents are text strings "Employee" has an attribute called "Employee\_Number" "Employee\_Number" content must be int type
* Write an xml (EMP.XML) File for above description.
  + Code:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet href="emp.xsl" type="text/xsl"?>

<Employee\_Info>

<Employee Employee\_Number="123">

<Name>Chirag Rathod</Name>

<Department>Marketing</Department>

<Telephone>+91 98756 25462</Telephone>

<Email>rajeshKumar@microsoft.com</Email>

</Employee>

<Employee Employee\_Number="456">

<Name>Himanshu Parmar</Name>

<Department>Finance</Department>

<Telephone>+91 78982 32548</Telephone>

<Email>priyaSharma@gmail.com</Email>

</Employee>

</Employee\_Info>

* Write a document type definition (EMP.DTD) file to validate above XML file.
  + Code:

<!ELEMENT Employee\_Info (Employee\*)>

<!ELEMENT Employee (Name, Department, Telephone, Email)>

<!ATTLIST Employee Employee\_Number CDATA #REQUIRED>

<!ELEMENT Name (#PCDATA)>

<!ELEMENT Department (#PCDATA)>

<!ELEMENT Telephone (#PCDATA)>

<!ELEMENT Email (#PCDATA)>

* Write an xml schema EMP.xsd file to validate above XML file
  + Code:

<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="Employee\_Info">

<xs:complexType>

<xs:sequence>

<xs:element name="Employee" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="Name" type="xs:string"/>

<xs:element name="Department" type="xs:string"/>

<xs:element name="Telephone" type="xs:string"/>

<xs:element name="Email" type="xs:string"/>

</xs:sequence>

<xs:attribute name="Employee\_Number" type="xs:int" use="required"/>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>

* Write an xml style sheet file (EMP.xsl) file to print above XML data in table format.
  + Code:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

<body>

<h2>Employee Information</h2>

<table border="1">

<tr bgcolor="#9acd32">

<th>Employee Number</th>

<th>Name</th>

<th>Department</th>

<th>Telephone</th>

<th>Email</th>

</tr>

<xsl:for-each select="Employee\_Info/Employee">

<tr>

<td><xsl:value-of select="@Employee\_Number"/></td>

<td><xsl:value-of select="Name"/></td>

<td><xsl:value-of select="Department"/></td>

<td><xsl:value-of select="Telephone"/></td>

<td><xsl:value-of select="Email"/></td>

</tr>

</xsl:for-each>

</table>

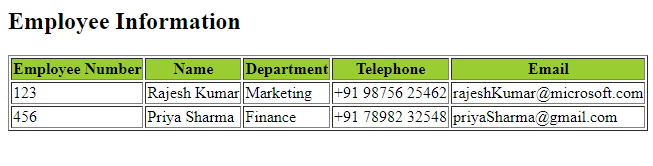
</body>

</html>

</xsl:template>

</xsl:stylesheet>

* + Output:



*Figure 22 Employee Information*

# **Experiment 9**

* Introduction and basic syntax

1. echo with html file
   * Code :

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>PHP Experiment 9</title>

</head>

<body>

<h3>echo</h3>

<?php

echo "<h2>Experiment 9</h2>";

echo "Hello India!<br>";

echo "I'm Completing Web Technologies Exp 9<br>";

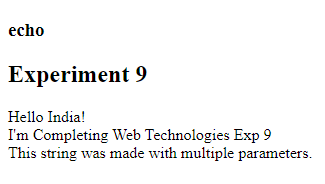
echo "This ", "string ", "was ", "made ", "with multiple parameters.";

?>

</body>

</html>

* + Output:



*Figure 23 echo with html file*

1. PHP Data Types
   * Code :

<?php

$x = 5.5; // float

$y = true; // boolean

$z = array(1, 2, 3); // array

class Person {

public $name;

public $age;

public function \_\_construct($name, $age) {

$this->name = $name;

$this->age = $age;

}

public function introduce() {

return "My name is " . $this->name . " and I am " . $this->age . " years old!";

}

}

echo "var\_dump() function returns the data type and the value<br>";

$person = new Person("John", 25);

var\_dump($person);

var\_dump($x);

echo "<br>";

var\_dump($y);

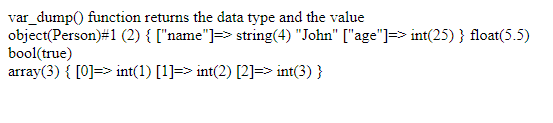
echo "<br>";

var\_dump($z);

echo "<br>";

?>

* + Output:



*Figure 24 Php dataTypes*

1. Decision making and Looping
   * Code :

<?php

$age = 20;

echo "<h4>Decision Making Example</h4>";

if ($age < 18) {

echo "You are a minor.";

} elseif ($age >= 18 && $age < 60) {

echo "You are an adult.";

} else {

echo "You are a senior citizen.";

}

echo "<br>";

echo "<h4>Looping Example</h4>";

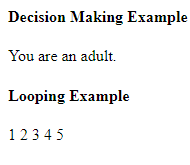
for ($i = 1; $i <= 5; $i++) {

echo $i . " ";

}

?>

* + Output :



*Figure 25 Decision making and Looping*

1. Php Arrays
   * Code :

<?php

echo "<h4>Array Example</h4>";

$colors = array("Red", "Green", "Blue");

echo "Access with index : "; // Output: Array

echo $colors[0];

echo $colors[1];

echo $colors[2];

echo "<br>Total Colors: ";

echo count($colors);

echo "<br>";

echo "Access with foreach Loop : "; // Output: Array

foreach ($colors as $color) {

echo $color . " ";

}

echo "<br>";

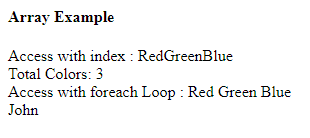
//array with key value pair

$person = array("name" => "John", "age" => 30);

echo $person["name"];

?>

* + Output :



*Figure 26 Php Array*

# **Experiment 10**

* PHP Advanced Features

1. PHP functions
   * Code:

<?php

function calculateSum($num1, $num2) {

$sum = $num1 + $num2;

return $sum;

}

echo "Sum of 5 and 10 is :" . calculateSum(5, 10) . "<br>";

function multiplicationTable($num) {

for ($i = 1; $i <= 10; $i++) {

$result = $num \* $i;

echo "$num x $i = $result <br>";

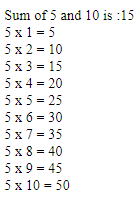
}

}

multiplicationTable(5);

?>

* + Output:



*Figure 27P Php functions*

1. Php Strings
   * Code:

<?php

$x = "Naman";

echo "Hello $x";

echo "<br>";

$x = "Namo";

echo 'Hello $x';

echo "<br>";

echo "String Length(Hello Chirag) : ";

echo strlen("Hello Chirag");

echo "<br>";

echo "Word Count(Hello Dhoni) : ";

echo str\_word\_count("Hello Dhoni");

echo "<br>";

echo "Position of word(Hello kohli) : ";

echo strpos("Hello kohli!", "kohli");

echo "<br>";

echo "Reverse word(BVM) : ";

$x = "BVM";

echo strrev($x);

echo "<br>";

echo "Concate String (Hello+Putin) : ";

$x = "Hello";

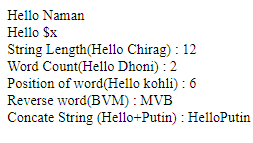
$y = "Putin";

$z = $x . $y;

echo $z;

?>

* + Output:



*Figure 28 Php Strings*

1. Form Processing
   * Code:
   * HTML Code :

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Contact Form</title>

</head>

<body>

<h2>Contact Us</h2>

<form action="form.php" method="post">

<label for="name">Your Name:</label><br>

<input type="text" id="name" name="name"><br>

<label for="email">Your Email:</label><br>

<input type="email" id="email" name="email"><br>

<label for="message">Message:</label><br>

<textarea id="message" name="message"></textarea><br>

<input type="submit" value="Submit">

</form>

</body>

</html>

* + Php Code file :

<?php

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

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

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

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

if (empty($name) || empty($email) || empty($message)) {

echo "Please fill out all required fields.";

} else {

echo "<h2>Thank you for your submission, $name!</h2>";

echo "<p><strong>Your Email:</strong> $email</p>";

echo "<p><strong>Message:</strong><br>$message</p>";

}

} else {

header("Location: form.html");

exit();

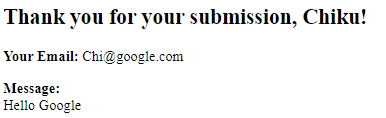
}

?>

* + Output:



*Figure 29 Contact Us form Input*



*Figure 30 Output of Contact form Using Php*

1. Object Oriented Programming With php
   * Code :

<?php

class Car {

public $name;

public $year;

public function \_\_construct( $name, $year) {

$this->name = $name;

$this->year = $year;

}

public function displayDetails() {

echo "Car: {$this->year} {$this->name}";

}

public function calculateAge() {

$currentYear = date('Y');

return $currentYear - $this->year;

}

}

$car1 = new Car("Toyota", 2018);

$car2 = new Car("Honda", 2015);

echo "Car 1 Details: ";

echo "<br>";

$car1->displayDetails();

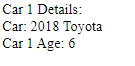
echo "<br>";

echo "Car 1 Age: " . $car1->calculateAge();

echo "<br>";

?>

* + Output :



*Figure 31 Car Object Details*

# **Experiment 11**

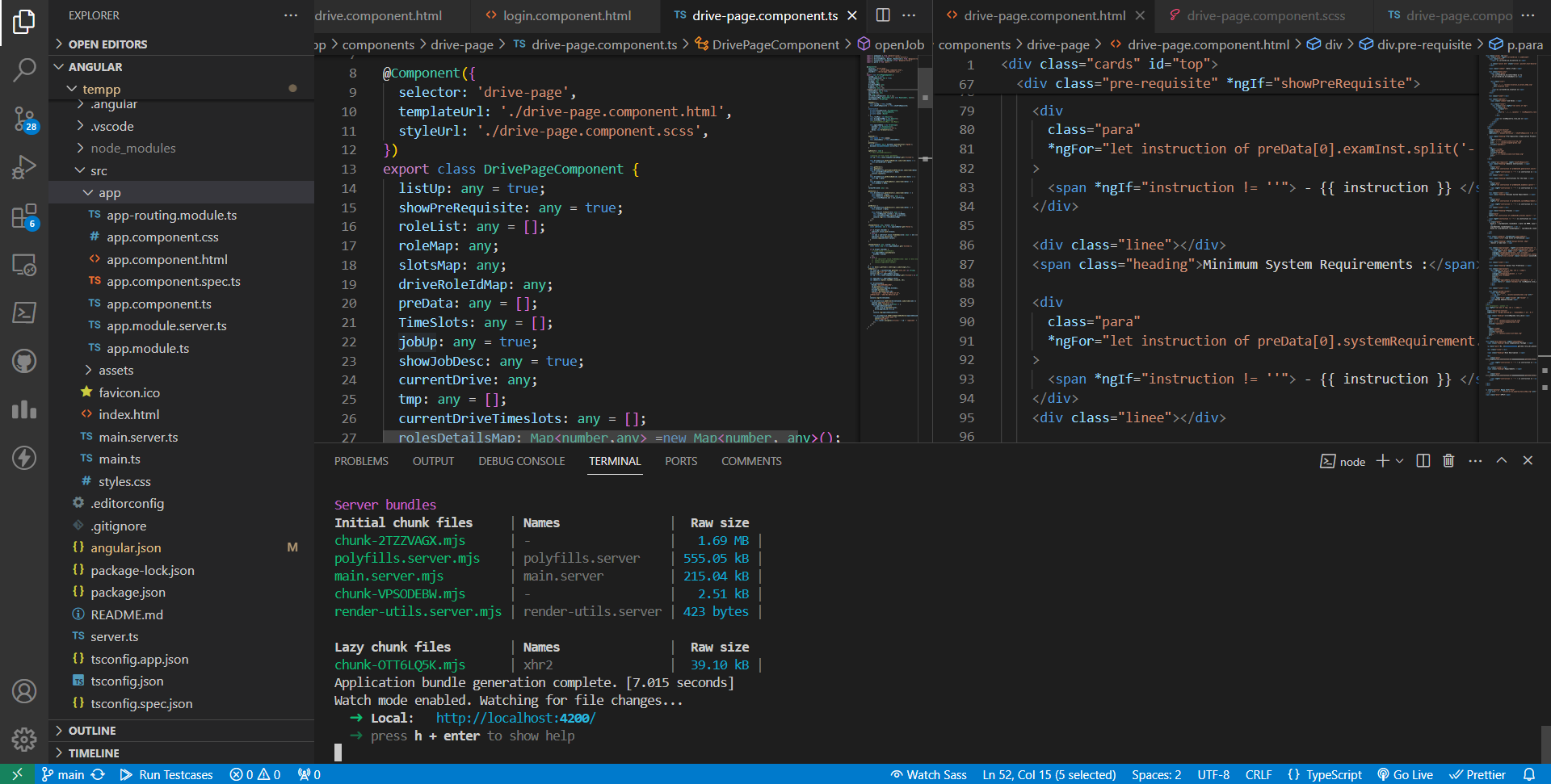
* Angular JS, developed by Google, is a powerful JavaScript framework designed for building dynamic web applications.
* It's an open-source framework that extends HTML with new attributes and binds data to HTML with expressions.
* AngularJS follows the Model-View-Controller (MVC) architecture, making it easy to organize and maintain code.
* **Loading AngularJS Framework**
* Download AngularJS - Website : <https://angularjs.org/>
* We include AngularJS in HTML file using a <script> tag.
* <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
* This will load the latest version of AngularJS
* **Installing Required Software**

1. Text Editor or IDE
2. Web Browser
3. Node.js and npm (Node Package Manager)
   * AngularJS development often involves using npm to manage packages and Node.js for running development servers and build tools.
4. AngularJS CLI (Command Line Interface)
   * npm install –g @angular/cli
   * ng v
5. HTTP Server (e.g., Express.js)
   * For serving AngularJS applications during development, you can use a simple HTTP server like Express.js (which can be installed via npm) or any other local server setup.
6. Creating a new project in Angular JS
   * ng new myFirstApp --standalone=false

* **Create a new Components**

1. ng g c firstComp
2. Main building blocks of angular js
3. app.comp.ts
4. app.comp.html
5. app.comp.css
6. app.module.ts
7. Index.html
8. main.ts

* **Project Structure**



*Figure 32 Project Structure Of AngularJS*

* **Simple Example to Add registered User**
* **Code :**

<!DOCTYPE html>

<html lang="en" ng-app="myApp">

<head>

<meta charset="UTF-8">

<title>User Registration Using Angular JS</title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

</head>

<body>

<div ng-controller="userListController">

<h1>User List</h1>

<ul>

<li ng-repeat="user in users">{{ user.name }}</li>

</ul>

<form ng-submit="adduser()">

<input type="text" ng-model="newuserName" placeholder="Enter new user">

<button type="submit">Add user</button>

</form>

</div>

<script>

angular.module('myApp', [])

.controller('userListController', function($scope) {

$scope.users = [

{ name: 'Chirag' },

{ name: 'Himanshu ' },

{ name: 'Himanshi' }

];

$scope.adduser = function() {

if ($scope.newuserName) {

$scope.users.push({ name: $scope.newuserName });

$scope.newuserName = '';

}

};

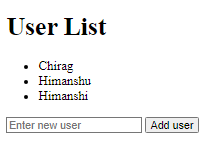
});

</script>

</body>

</html>

* **Output :**



*Figure 33 Add User to List*

# **Experiment 12**

* **Applications of AngularJS**

1. Single-Page Applications (SPAs):
   * AngularJS is ideal for building SPAs where most or all of the application's content is dynamically loaded on a single web page.
   * SPAs provide a seamless user experience by updating specific portions of the page without full page reloads.
2. Dynamic Dashboards:
   * AngularJS is well-suited for creating dynamic dashboards that display real-time data and allow users to interact with charts, graphs, and widgets.
3. Social Media Applications:
   * Applications requiring real-time updates, user interactions, and content sharing functionalities can benefit from AngularJS's two-way data binding and seamless DOM updates.
4. Content Management Systems (CMS):
   * AngularJS can be used to build intuitive and responsive content management systems where users can create, edit, and manage content efficiently.
5. E-commerce Platforms:
   * AngularJS enables the development of interactive and dynamic e-commerce websites that provide a smooth shopping experience with features like product catalogs, shopping carts, and checkout processes.
6. Data-Driven Applications:
   * Applications that heavily rely on data visualization, manipulation, and user-driven interactions can leverage AngularJS's data binding capabilities and powerful directives.
7. Real-Time Chat Applications:
   * AngularJS combined with WebSockets or other real-time communication technologies can be used to build interactive chat applications with instant messaging capabilities.
8. Collaborative Tools:
   * Applications that require real-time collaboration features such as document editing, shared whiteboards, or project management tools can benefit from AngularJS's ability to handle complex UI updates.
9. Customer Relationship Management (CRM) Systems:
   * AngularJS can be used to develop CRM systems that provide comprehensive customer data management, analytics, and reporting functionalities.
10. Learning Management Systems (LMS):
    * Educational platforms that deliver interactive courses, quizzes, and assessments can be efficiently developed using AngularJS to handle complex UI interactions and data management.