**Practical 1: HTML**

**a. Create a Basic Web Page**

**Aim:** Create a simple web page with a heading, paragraph, and an image

<!DOCTYPE html>

<html>

<head>

<title>My First Web Page</title>

</head>

<body>

<h1>Welcome to My Website</h1>

<p>This is my first paragraph.</p>

<img src="image.jpg" alt="Image Description">

</body>

</html>

.

**b. Formatting Text and Creating Lists**

**Aim:** Format text using headings, paragraphs, and lists.

<!DOCTYPE html>

<html>

<head>

<title>Text Formatting and Lists</title>

</head>

<body>

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<p>This is a paragraph with <strong>bold</strong> and <em>italic</em> text.</p>

<ul>

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>

<ol>

<li>Step 1</li>

<li>Step 2</li>

<li>Step 3</li>

</ol>

</body>

</html>

**c. Creating Hyperlinks and Tables**

**Aim:** Create hyperlinks and tables to organize content.

<!DOCTYPE html>

<html>

<head>

<title>Hyperlinks and Tables</title>

</head>

<body>

<a href="https://www.example.com">Visit Example Website</a>

<table>

<tr>

<th>Name</th>

<th>Age</th>

<th>City</th>

</tr>

<tr>

<td>Alice</td>

<td>25</td>

<td>New York</td>

</tr>

<tr>

<td>Bob</td>

<td>30</td>

<td>Los Angeles</td>

</tr>

</table>

</body>

</html>

**Practical 2**

**a. Creating Image Maps**

**Aim:** Create an image map with clickable areas.

<!DOCTYPE html>

<html>

<head>

<title>Image Map</title>

</head>

<body>

<img src="image.jpg" usemap="#myMap">

<map name="myMap">

<area shape="rect" coords="34,44,270,350" alt="Computer" href="https://in-media.apjonlinecdn.com/catalog/product/cache/b3b166914d87ce343d4dc5ec5117b502/c/0/c08192495\_33.png">

        <area shape="rect" coords="290,172,333,250" alt="Phone" href="https://ikall.in/wp-content/uploads/2023/06/51sJmHfw92L.\_SL1000\_.jpg">

        <area shape="circle" coords="337,300,44" alt="Cup of coffee" href="https://static.vecteezy.com/system/resources/thumbnails/025/282/026/small\_2x/stock-of-mix-a-cup-coffee-latte-more-motive-top-view-foodgraphy-generative-ai-photo.jpg">

</map>

</body>

</html>

**b. Creating Forms**

**Aim:** Create a form with various input fields.

<!DOCTYPE html>

<html>

<head>

<title>Form Example</title>

</head>

<body>

<h2>Contact Form</h2>

<form action="process\_form.php" method="post">

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

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

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

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

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

<textarea id="message" name="message" rows="5" cols="40"></textarea><br><br>

<label for="birthdate">Birthdate:</label>

<input type="date" id="birthdate" name="birthdate"><br><br>

<label for="phone">Phone Number:</label>

<input type="tel" id="phone" name="phone" pattern="[0-9]{10}"><br><br>

<label for="gender">Gender:</label>

<input type="radio" id="male" name="gender" value="male">

<label for="male">Male</label>

<input type="radio" id="female" name="gender" value="female">

<label for="female">Female</label>

<input type="radio" id="other" name="gender" value="other">

<label for="other">Other</label><br><br>

<label for="country">Country:</label>

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

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

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

<option value="uk">UK</option>

</select><br><br>

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

<input type="reset" value="Reset">

</form>

</body>

</html>

**c. Embedding Audio and Video**

**Aim:** Embed audio and video files on a web page.

<!DOCTYPE html>

<html>

<head>

<title>Audio and Video</title>

</head>

<body>

<audio controls>

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

Your browser does not support the audio element.

</audio>

<br><br>

<video controls>

<source src="video.mp4" type="video/mp4">

Your browser does not support the video element.

</video>

</body>

</html>

**Practical 3: CSS**

**a. Basic CSS Styling**

<!DOCTYPE html>

<html>

<head>

<title>Form Example</title>

<style>

form {

max-width: 500px;

margin: 0 auto;

padding: 20px;

border: 1px solid #ccc;

border-radius: 5px;

background-color: #f0f0f0;

}

label {

display: block;

margin-bottom: 5px;

}

input[type="text"],

input[type="email"],

textarea {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 3px;

}

input[type="submit"] {

background-color: #4CAF50;

color: white;

padding: 10px 20px;

border: none;

border-radius: 3px;

cursor: pointer;

}

input[type="submit"]:hover {

background-color: #3e8e41;

}

</style>

</head>

<body>

<h2>Contact Form</h2>

<form action="process\_form.php" method="post">

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

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

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

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

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

<textarea id="message" name="message" rows="5" cols="40"></textarea><br><br>

<label for="birthdate">Birthdate:</label>

<input type="date" id="birthdate" name="birthdate"><br><br>

<label for="phone">Phone Number:</label>

<input type="tel" id="phone" name="phone" pattern="[0-9]{10}"><br><br>

<label for="gender">Gender:</label>

<input type="radio" id="male" name="gender" value="male">

<label for="male">Male</label>

<input type="radio" id="female" name="gender" value="female">

<label for="female">Female</label>

<input type="radio" id="other" name="gender" value="other">

<label for="other">Other</label><br><br>

<label for="country">Country:</label>

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

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

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

<option value="uk">UK</option>

</select><br><br>

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

</form>

</body>

</html>

**b. CSS Box Model**

**Aim: Understand and apply the CSS box model.**

<!DOCTYPE html>

<html>

<head>

<title>CSS Layout</title>

<style>

body {

margin: 0;

}

header {

background-color: #f0f0f0;

padding: 20px;

}

nav {

background-color: #ccc;

padding: 10px;

}

section {

padding: 20px;

}

footer {

background-color: #f0f0f0;

text-align: center;

padding: 10px;

}

</style>

</head>

<body>

<header>

<h1>My Website</h1>

</header>

<nav>

<ul>

<li><a href="#">Home</a></li>

<li><a href="#">About</a></li>

<li><a href="#">Contact</a></li>

</ul>

</nav>

<section>

<p>This is the main content of the page.</p>

</section>

<footer>

&copy; 2023 My Website

</footer>

</body>

</html>

**c CSS Positioning**

**Aim: Position elements on a page using CSS.**

<!DOCTYPE html>

<html>

<head>

<title>CSS Positioning</title>

<style>

.box {

width: 100px;

height: 100px;

border: 1px solid black;

}

.absolute {

position: absolute;

top: 20px;

left: 20px;

}

.relative {

position: relative;

top: 20px;

left: 20px;

}

</style>

</head>

<body>

<div class="box absolute">Absolute Position</div>

<div class="box relative">Relative Position</div>

</body>

</html>

### 

**Practical 3: CSS**

### **a: CSS Flexbox Layout**

**Aim: Create a flexible layout using CSS Flexbox.**

<!-- CSS Flexbox Layout-->

<!DOCTYPE html>

<html>

<head>

<title>CSS Flexbox Layout</title>

<style>

.container {

display: flex;

justify-content: space-between;

}

.box {

width: 200px;

height: 100px;

border: 1px solid black;

margin: 10px;

}

</style>

</head>

<body>

<div class="container">

<div class="box">Box 1</div>

<div class="box">Box 2</div>

<div class="box">Box 3</div>

</div>

</body>

</html>

### **b: CSS Grid Layout**

**Aim: Create a grid-based layout using CSS Grid.**

<!-- CSS Grid Layout -->

<!DOCTYPE html>

<html>

<head>

<title>CSS Grid Layout</title>

<style>

.container {

display: grid;

grid-template-columns: 1fr 1fr 1fr;

grid-gap: 10px;

}

.box {

border: 1px solid black;

padding: 20px;

}

</style>

</head>

<body>

<div class="container">

<div class="box">Box 1</div>

<div class="box">Box 2</div>

<div class="box">Box 3</div>

<div class="box">Box 4</div>

<div class="box">Box 5</div>

<div class="box">Box 6</div>

</div>

</body>

</html>

### **c: CSS Transitions and Animations**

**Aim: Create dynamic effects using CSS transitions and animations.**

<!-- CSS Transitions and Animations -->

<!DOCTYPE html>

<html>

<head>

<title>CSS Transitions and Animations</title>

<style>

.box {

width: 100px;

height: 100px;

background-color: blue;

transition: all 0.5s ease-in-out;

}

.box:hover {

background-color: red;

transform: scale(1.2);

}

.animated-box {

animation: myAnimation 5s infinite;

}

@keyframes myAnimation {

0% {

transform: translateX(0);

}

50% {

transform: translateX(200px);

}

100% {

transform: translateX(0);

}

}

</style>

</head>

<body>

<div class="box"></div>

<div class="box animated-box"></div>

</body>

</html>

**Extra (b. CSS Selectors**

**Aim: Use CSS selectors to target specific elements.**

**c. CSS Layout**

**Aim: Create a simple layout using CSS.**

**Practical 5: CSS Typography**

**Aim: Style text using CSS.**

**Chapter 3 JavaScript:-**

### **Practical 1: Basic JavaScript in HTML**

**Aim: Create a simple HTML page with a button that displays an alert when clicked.**

<!DOCTYPE html>

<html>

<head>

<title>Basic JavaScript</title>

</head>

<body>

<button onclick="showAlert()">Click me</button>

<script>

function showAlert() {

alert("Hello, world!");

}

</script>

</body>

</html>

### **Practical 2: Variables and Operators**

**Aim: Calculate the area of a rectangle using JavaScript variables and operators.**

<!DOCTYPE html>

<html>

<head>

<title>Rectangle Area Calculator</title>

</head>

<body>

<script>

// Declare variables for length and width

let length = 10;

let width = 5;

// Calculate the area

let area = length \* width;

// Display the result using an alert box

alert("The area of the rectangle is: " + area);

// Or display it in the console for debugging:

console.log("The area of the rectangle is: " + area);

//Or display it on the page itself

document.write("The area of the rectangle is: " + area);

</script>

</body>

</html>

**ORRRRR**

<!DOCTYPE html>

<html>

<head>

<title>Rectangle Area Calculator</title>

</head>

<body>

<form id="areaForm">

Length: <input type="number" id="length"><br><br>

Width: <input type="number" id="width"><br><br>

<button type="button" onclick="calculateArea()">Calculate Area</button>

</form>

<div id="result"></div>

<script>

function calculateArea() {

let length = parseFloat(document.getElementById("length").value);

let width = parseFloat(document.getElementById("width").value);

let resultDiv = document.getElementById("result");

if (isNaN(length) || isNaN(width)) {

resultDiv.textContent = "Please enter valid numbers.";

} else {

let area = length \* width;

resultDiv.textContent = "The area of the rectangle is: " + area;

}

}

</script>

</body>

</html>

### **Practical 3: Control Flow Statements**

**Aim: Write a JavaScript program to check if a number is even or odd.**

<!DOCTYPE html>

<html>

<head>

<title>Control Flow Statements</title>

</head>

<body>

<script>

let number = 25;

if (number % 2 === 0) {

console.log(number + " is even.");

} else {

console.log(number + " is odd.");

}

</script>

</body>

</html>

**ORRRRRRRRRRR**

<!DOCTYPE html>

<html>

<head>

<title>Even or Odd Checker</title>

</head>

<body>

<script>

// 1. Get the number (you can change this value to test)

let number = 17;

// 2. The core logic: Use the modulo operator (%)

// The modulo operator (%) gives you the remainder of a division.

// If a number is perfectly divisible by 2, the remainder is 0 (even).

// Otherwise, the remainder is 1 (odd).

if (number % 2 === 0) {

// 3a. If the remainder is 0, it's even

console.log(number + " is an even number."); // Display in the console

document.write(number + " is an even number."); // Display on the page

} else {

// 3b. If the remainder is not 0, it's odd

console.log(number + " is an odd number."); // Display in the console

document.write(number + " is an odd number."); // Display on the page

}

</script>

</body>

</html>

### **Practical 4: Functions**

**Aim: Create a function to calculate the factorial of a number.**

<!DOCTYPE html>

<html>

<head>

<title>Functions</title>

</head>

<body>

<script>

function factorial(n) {

if (n === 0) {

return 1;

} else {

return n \* factorial(n - 1);

}

}

let result = factorial(5);

console.log("The factorial of 5 is:", result);

</script>

</body>

</html>

**ORRRRRRRRR**

### **Practical 5: JavaScript Objects**

**Aim: Create a JavaScript object to represent a person and access its properties.**

<!DOCTYPE html>

<html>

<head>

<title>JavaScript Objects</title>

</head>

<body>

<script>

let person = {

firstName: "John",

lastName: "Doe",

age: 30,

city: "New York"

};

console.log(person.firstName);

console.log(person["lastName"]);

</script>

</body>

</html>

### **Practical 6: DOM Manipulation**

**Aim: Change the background color of a button when it's clicked.**

<!DOCTYPE html>

<html>

<head>

<title>DOM Manipulation</title>

</head>

<body>

<button id="myButton">Click me</button>

<script>

let button = document.getElementById("myButton");

button.addEventListener("click", function() {

button.style.backgroundColor = "red";

});

</script>

</body>

</html>

### **Extra (Practical 7: Form Validation**

**Aim: Validate a form to ensure the user enters a valid email address.)**

Module 2

**Chapter 4 XML:-**

### **Practical 1: Basic XML Structure**

**Aim: Create a simple XML document to represent a book.**

### **Practical 2: XML with Entities**

**Aim: Use entity references to avoid special characters in XML.**

### **Practical 3: XML with Internal DTD**

**Aim: Define the structure of an XML document using an internal DTD.**

### **Practical 4: XML with External DTD**

**Aim: Define the structure of an XML document using an external DTD.**

### **(Practical 5: XSLT Transformation**

**Aim: Transform an XML document into HTML using XSLT.)**

**Chapter 5 AJAX:-**

### **Practical 1: Simple AJAX Request**

**Aim: Fetch data from a server and display it on the web page.**

### **Practical 2: Dynamic Content Loading**

**Aim: Load content dynamically into a specific HTML element.**

### **Practical 3: Form Submission without Page Refresh**

**Aim: Submit a form using AJAX and display the server response.**

### **Practical 5: Asynchronous Image Loading**

**Aim: Load images asynchronously to improve page load performance.**

### **(Practical 4: Real-time Updates**

**Aim: Fetch data from a server periodically and update the web page.)**

**Chapter 6 PHP:-**

### **Practical 1: Form Handling and Data Processing**

**Aim: Create an HTML form to collect user input and process it using PHP.**

### **Practical 2: File Handling**

**Aim: Create a PHP script to read and write to a file.**