Assignment-HTML

Module 1 – Foundation

#### THEORY EXERCISE:

1. What is a HTTP?

* HTTP (HyperText Transfer Protocol) is a communication protocol used to transfer data between a web browser (client) and a web server over the internet.
* HTTP is a protocol (set of rules) used to transfer data like web pages, images, videos, etc., between a web browser (client) and a web server.

### How it Works:

1. **Client (Browser)** sends a request for a web page (e.g., when you type a URL and press enter).
2. **Server** processes the request and sends back the response (like HTML, CSS, images, etc.).
3. The **browser displays** the received data.

### Example:

* When you open https://www.example.com, your browser sends an **HTTP request** to the server.
* The server responds with an **HTTP response** containing the web page data.

### HTTP Methods:

* GET: Request data (e.g., view a web page)
* POST: Submit data (e.g., fill out a form)
* PUT: Update data
* DELETE: Remove data

1. What is a Browsers? How they works?

* A **browser** (or **web browser**) is a **software application** used to access, retrieve, and display content from the **World Wide Web** (WWW).
* Common browsers include **Google Chrome, Mozilla Firefox, Microsoft Edge, Safari**, and **Opera**.

### How Does a Browser Work?

Here's a simple step-by-step explanation of how browsers work:

#### 1. ****User Enters a URL****

* You type something like https://www.google.com into the browser’s address bar.

#### 2. ****DNS Lookup****

* The browser sends the domain name (google.com) to a **DNS (Domain Name System)** server to get the **IP address** of the server.

#### 3. ****HTTP/HTTPS Request****

* The browser sends an **HTTP or HTTPS request** to the server at that IP address asking for the web page.

#### 4. ****Server Response****

* The web server responds with the necessary files (HTML, CSS, JavaScript, images, etc.).

#### 5. ****Rendering the Page****

* The browser receives the files and:
  + Parses the HTML
  + Applies the CSS for styling
  + Executes JavaScript for interactivity
  + Displays the final page on your screen

#### 6. ****Displaying Content****

* The browser renders (displays) the complete webpage as you see it.

1. What is Domain Name?

* A **domain name** is the **human-readable address** of a website on the internet.
* It is used to identify and access websites easily without needing to remember complex IP addresses.
* Using different Domain name for different countries is beneficial for several reasons.
* It helps with geographical targeting in search results, builds local trust and credibility and can improve search engine optimization(SEO) whiting specific countries.

### Why Are Domain Names Important?

Computers use **IP addresses** like 142.250.195.78 to communicate.  
But IPs are hard to remember, so we use **domain names** like google.com, which are easier for humans.

### Structure of a Domain Name:

pgsql

CopyEdit

www.example.com

| | |

| | └── Second-level domain (example)

| └────────── Subdomain (www)

└────────────── Top-level domain (TLD - .com)

### Common Top-Level Domains (TLDs):

* .com – Commercial
* .org – Organization
* .net – Network
* .edu – Educational
* .in, .uk, .us – Country-specific

1. What is hosting?

* **Hosting** (or **Web Hosting**) is a **service** that allows individuals or organizations to **store their website files** (HTML, images, videos, etc.) on a **server** that is connected to the **internet**, so people can access the website anytime, anywhere.
* **Hosting is like renting space on the internet** to keep your website online

Module 2 – Fundamentals of World Wide Web

#### THEORY EXERCISE:

1. Difference between Web Designer and Web Developer

### Difference Between Web Designer and Web Developer

|  |  |  |
| --- | --- | --- |
| Aspect | **Web Designer** | **Web Developer** |
| **Main Role** | Focuses on the **look and feel** of a website | Focuses on the **functionality** and structure |
| **Key Skills** | UI/UX design, Color theory, Typography, Adobe XD, Figma, Photoshop, HTML, CSS | HTML, CSS, JavaScript, PHP, Python, Databases, APIs |
| **Tools Used** | Figma, Adobe XD, Sketch, Canva | VS Code, Git, Terminal, Databases |
| **Languages** | HTML, CSS, some JavaScript | HTML, CSS, JavaScript, plus backend languages |
| **Focus** | User interface (UI), User experience (UX) | Site logic, server, database, interactivity |
| **Output** | Creates wireframes, mockups, page layouts | Builds the website based on design |

### Simple Example:

* A **Web Designer** designs a beautiful online store layout.
* A **Web Developer** writes code to make the store actually work (shopping cart, login, payment, etc.)

1. What is a W3C?

* **W3C** stands for the **World Wide Web Consortium**.
* It is the **main international organization** that develops **web standards** to ensure the **long-term growth and compatibility** of the web.

1. What is Domain?

A **domain** is the **unique name** used to identify a website on the internet.

1. What SEO?

* **SEO** stands for **Search Engine Optimization**.
* SEO is the process of **improving a website’s visibility** on **search engines** like Google, Bing, or Yahoo — to get **more free (organic) traffic**.

### Simple Definition:

* SEO helps your website **rank higher** in search results when people search for something related to your content.
* For example:  
  If you have a blog about healthy food, SEO helps your site appear on **Google’s first page** when someone searches for “healthy recipes”.

### Key Elements of SEO:

1. **Keywords** – Search terms people use (e.g., “best laptop 2025”)
2. **Title Tags & Meta Descriptions** – Shown in search results
3. **Backlinks** – Links from other sites (builds trust)
4. **Content Quality** – Useful, relevant, and original content
5. **Mobile Optimization** – Works well on phones and tablets
6. **Site Speed** – Fast loading = better user experience
7. **Alt Text** – Descriptions for images (important for SEO & accessibility)
8. What is SDLC life cycle?

* **SDLC** stands for **Software Development Life Cycle**.
* It is a **step-by-step process** used by software developers and project teams to **design, develop, test, and maintain** software applications.

### Purpose of SDLC:

* To ensure **high-quality software**
* Delivered **on time**
* Within **budget**
* With **minimum risk**

**Phases of SDLC:**

[ Planning] → [ Requirement Analysis] → [ Design] →

[ Development] → [ Testing] → [ Deployment] → [ Maintenance]

↑---------------------------------------------------↑

### Why SDLC is Important:

* Helps teams **stay organized**
* Reduces **errors and risks**
* Ensures **customer satisfaction**
* Allows for **proper documentation**

# Module 2 – Frontend- HTML

## HTML Basics

### Theory Assignment

**Question 1**: Define HTML. What is the purpose of HTML in web development?

**Definition of HTML:**

**HTML (HyperText Markup Language)** is the standard markup language used to create and structure content on the web. It tells the web browser how to display text, images, links, and other media on a webpage.

### ****Purpose of HTML in Web Development:****

1. **Structure the Content:**
   * HTML provides a structural foundation for web pages using elements like <header>, <footer>, <section>, and <article>.
2. **Display Text and Media:**
   * It enables the display of text, images, videos, audio, and other media using tags like <p>, <img>, <video>, etc.
3. **Link Webpages (Hypertext):**
   * HTML allows pages to be interconnected using hyperlinks (<a> tags), forming the basis of website navigation.
4. **Form User Input:**
   * HTML includes form elements like <form>, <input>, <textarea>, and <button> to collect data from users.
5. **Embed Scripts and Styles:**
   * HTML can integrate CSS for styling and JavaScript for behavior, helping build interactive and visually appealing websites.

**Question 2**: Explain the basic structure of an HTML document. Identify the mandatory tagsand their purposes.

### ****Basic Structure of an HTML Document****

An HTML document has a well-defined structure made up of essential tags that organize and define the content of a web page.

**Basic HTML Structure:**

<!DOCTYPE html>

<html>

<head>

<title>Page Title</title>

</head>

<body>

<h1>Main Heading</h1>

<p>This is a paragraph of text.</p>

</body>

</html>

### ****Mandatory HTML Tags and Their Purposes:****

|  |  |
| --- | --- |
| **Tag** | **Purpose** |
| <!DOCTYPE html> | Declares the document type and version of HTML (HTML5). Helps browsers render the page correctly. |
| <html> | Root element of the HTML document. Wraps all the content on the page. |
| <head> | Contains meta-information about the document (not visible to users), such as title, CSS, charset, etc. |
| <title> | Sets the title of the webpage (shown in the browser tab). |
| <body> | Contains all the visible content of the page like text, images, links, etc. |

**Question 3**: What is the difference between block-level elements and inline elements in HTML? Provide examples of each.

### ****1. Block-Level Elements****

* **Definition:** Block-level elements take up the full width of the container and start on a **new line**.
* **Purpose:** Used to create larger structures (like paragraphs, sections, dividers).
* **Examples:**
  + <div>
  + <p>
  + <h1> to <h6>
  + <ul>, <ol>, <li>
  + <table>, <section>, <article>

#### 📌 Example:

html

<p>This is a paragraph.</p>

<h1>This is a heading</h1>

<div>This is a division block</div>

### ****2. Inline Elements****

* **Definition:** Inline elements do **not start on a new line**. They only take up as much width as necessary.
* **Purpose:** Used for formatting small pieces of content within block elements.
* **Examples:**
  + <span>
  + <a>
  + <strong>, <em>
  + <img>
  + <label>, <input>

#### 📌 Example:

<p>This is a <span style="color: red;">red word</span> inside a paragraph.</p>

<a href="#">Click here</a>

**Question 4**: Discuss the role of semantic HTML. Why is it important for accessibility and SEO? Provide examples of semantic elements.

**Semantic HTML** uses HTML5 elements that clearly describe their meaning and purpose in the document structure — making the content **more understandable for browsers, developers, assistive technologies, and search engines**.

### ****Why Semantic HTML is Important****

#### 1. ****Accessibility (A11Y):****

* **Screen readers** and assistive tools rely on semantic tags to navigate and interpret content correctly.
* Example: <nav> helps a screen reader know it’s a navigation menu.

#### 2. ****SEO (Search Engine Optimization):****

* Search engines like Google understand the **context** and **importance** of different sections using semantic tags.
* Helps improve **ranking** and **visibility** in search results.

#### 3. ****Code Clarity & Maintainability:****

* Makes HTML **easier to read** and **maintain** for developers.
* Reduces confusion caused by using generic tags like <div> for everything.

### ****Examples of Semantic HTML Elements:****

|  |  |
| --- | --- |
| Tag | Purpose |
| <header> | Represents the top section of a page or section |
| <nav> | Defines navigation links |
| <main> | Specifies the main content of the document |
| <section> | Groups related content into a logical section |
| <article> | Represents a self-contained piece of content |
| <aside> | Holds side content like ads or related links |
| <footer> | Represents the bottom area of a page or section |
| <figure> / <figcaption> | Used for images with captions |
| <time> | Represents a specific time or date |

**Semantic HTML Example:**

<!DOCTYPE html>

<html>

<head>

<title>Semantic HTML Example</title>

</head>

<body>

<header>

<h1>My Blog</h1>

</header>

<nav>

<ul>

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

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

</ul>

</nav>

<main>

<article>

<h2>HTML for Beginners</h2>

<p>Learn the basics of HTML in this article.</p>

</article>

</main>

<footer>

<p>© 2025 My Blog</p>

</footer>

</body>

</html>

**HTML Forms**

### Theory Assignment

**Question 1:** What are HTML forms used for? Describe the purpose of the input, textarea, select, and button elements.

### ****What Are HTML Forms Used For?****

**HTML forms** are used to **collect user input** on a webpage and send it to a server for processing.

Forms are essential for tasks like:

* Logging in
* Registering users
* Searching
* Submitting feedback or surveys
* Uploading files

### ****Key Form Elements and Their Purposes:****

#### 1. 🔘 <input>

* Used to collect **single-line** data (text, email, password, checkbox, radio, file, etc.).
* Type is defined using the type attribute.

**Example:**

<input type="text" name="username" placeholder="Enter your name">

<input type="email" name="email" placeholder="Enter your email">

<input type="checkbox" name="subscribe"> Subscribe

#### 2. 📝 <textarea>

* Used for **multi-line** text input such as comments, messages, or feedback.

**Example:**

<textarea name="message" rows="4" cols="30" placeholder="Write your message..."></textarea>

#### 3. 🔽 <select> ****(with**** <option>****)****

* Creates a **dropdown list** for selecting one or more options.

**Example:**

<select name="country">

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

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

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

</select>

#### 4. 🔘 <button>

* Creates a clickable **button**, often used to submit the form or trigger JavaScript actions.

**Example:**

<button type="submit">Submit</button>

<button type="reset">Reset</button>

### ****Complete Form Example:****

<form action="/submit" method="post">

<label>Name:</label>

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

<label>Message:</label>

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

<label>Country:</label>

<select name="country">

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

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

</select><br>

<button type="submit">Send</button>

</form>

**Question 2:** Explain the difference between the GET and POST methods in form submission. When should each be used?

### ****1. GET Method****

* **Sends data as URL parameters** (visible in the address bar).
* Data is **appended** to the URL after a ?.
* **Example URL:**  
  example.com/search?query=html&sort=recent

#### ✅ Use GET when:

* You are **retrieving data** (searches, filters).
* The form data is **not sensitive**.
* You want to **bookmark or share the URL** with data.

#### ⚠️ Avoid GET when:

* Sending **passwords, personal, or sensitive information**.
* Submitting **large amounts of data** (limited by URL length).

### ****2. POST Method****

* **Sends data in the body** of the HTTP request (not visible in URL).
* **More secure** than GET for sending sensitive data.
* No length limitations.

#### ✅ Use POST when:

* You are **modifying data** on the server (registration, login, form submission).
* The data is **confidential or large** in size.
* You don't want data to appear in the URL.

**Question 3:** What is the purpose of the label element in a form, and how does it improve accessibility?

The <label> element is used to **define a text description for a form input**, such as a textbox, checkbox, or radio button.

### ****Main Purposes of**** <label>****:****

1. **Describes the input field:**
   * Helps users understand what to enter in the field.
2. **Connects text to input fields:**
   * Clicking on the label sets focus to the input automatically.
3. **Improves user experience and usability.**

**HTML Tables**

### Theory Assignment

**Question 1**: Explain the structure of an HTML table and the purpose of each of the following elements: <table>, <tr>, <th>, <td>, and <thead>.

An **HTML table** is used to display data in a structured format using rows and columns. Here's a breakdown of its structure and the purpose of each key element:

**Basic Structure of an HTML Table:**

<table>

<thead>

<tr>

<th>Heading 1</th>

<th>Heading 2</th>

</tr>

</thead>

<tr>

<td>Data 1</td>

<td>Data 2</td>

</tr>

</table>

**Explanation of Each Element:**

|  |  |
| --- | --- |
| Element | Purpose |
| <table> | **Container** element that defines the table and holds all table content. |
| <tr> | **Table Row** – groups a row of cells (can include <th> or <td>). |
| <th> | **Table Header Cell** – used inside a <tr> to define a column header. Displayed in **bold** and **centered** by default. |
| <td> | **Table Data Cell** – contains regular data in a row. Appears under the corresponding header. |
| <thead> | Groups the header content (typically the first row). Helps with **semantic structure** and styling, and is useful when printing or scrolling tables. |

**Question 2**: What is the difference between colspan and rowspan in tables? Provide examples.

The colspan and rowspan attributes in HTML tables are used to **merge cells** across **columns** and **rows**, respectively.

**Difference Between colspan and rowspan:**

|  |  |  |
| --- | --- | --- |
| Attribute | Purpose | Effect |
| colspan | Merges cells **across columns** | A cell spans multiple columns in a row |
| rowspan | Merges cells **across rows** | A cell spans multiple rows in a column |

**1.colspan** Example**:**

Merges **2 columns** in a row (horizontal merge):

<table border="1">

<tr>

<th colspan="2">Name & Score</th>

</tr>

<tr>

<td>Alice</td>

<td>95</td>

</tr>

</table>

**Output**: The first row has a single cell spanning two columns with the heading "Name & Score".

2. **rowspan** Example:

Merges **2 rows** in a column (vertical merge):

<table border="1">

<tr>

<th rowspan="2">Name</th>

<td>Math</td>

</tr>

<tr>

<td>Science</td>

</tr>

</table>

**Output**: The "Name" cell spans two rows vertically, next to "Math" and "Science".

**Question 3**: Why should tables be used sparingly for layout purposes? What is a better alternative?

Using **tables for layout purposes** (like positioning elements on a webpage) is **discouraged** in modern web development. Here's why—and what you should use instead:

**Why Tables Should Be Used Sparingly for Layout:**

|  |  |
| --- | --- |
| Reason | Explanation |
| Not Semantic | Tables are meant for displaying **tabular data**, not for layout. Misusing them confuses screen readers and assistive technologies. |
| Hard to Maintain | Table-based layouts are **complex, bulky, and harder to edit or style** than modern layout techniques. |
| Poor Accessibility | Screen readers and other accessibility tools rely on **semantic HTML**. Layout tables break this structure. |
| Less Responsive | Tables are **rigid** and don’t adapt well to different screen sizes (e.g. phones, tablets). |
| Slower Rendering | Browsers may take longer to render large or nested tables than simpler layouts using CSS. |

**Better Alternative: CSS with <div>, Flexbox, and Grid**

**1.Flexbox** – Good for 1-dimensional layouts (row or column):

<div style="display: flex;">

<div style="flex: 1;">Left</div>

<div style="flex: 2;">Right</div>

</div>

**2. CSS Grid – Best for 2-dimensional layouts (rows and columns):**

<div style="display: grid; grid-template-columns: 1fr 2fr;">

<div>Left</div>

<div>Right</div>

</div>

### Lab Assignment

#### Task:

Create a product catalog table that includes the following columns:

 Product Name

 Product Image (use placeholder image URLs)

 Price

 Description

 Availability (in stock, out of stock)

#### Additional Requirements:

 Use thead for the table header.

 Add a border and some basic styling using inline CSS.

 Use colspan or rowspan to merge cells where applicable.

# Module 3 – Frontend – CSS and CSS3

**CSS Selectors & Styling**

### Theory Assignment

* **Question 1**: What is a CSS selector? Provide examples of element, class, and ID selectors.
* **Question 2**: Explain the concept of CSS specificity. How do conflicts between multiple stylesget resolved?
* **Question 3**: What is the difference between internal, external, and inline CSS? Discuss the advantages and disadvantages of each approach.

### Lab Assignment

#### Task:

Style the contact form (created in the HTML Forms lab) using external CSS. Thefollowing should be implemented:

 Change the background color of the form.

 Add padding and margins to form fields.

 Style the submit button with a hover effect.

 Use class selectors for styling common elements and ID selectors for uniqueelements.

**CSS Box Model**

### Theory Assignment

* **Question 1**: Explain the CSS box model and its components (content, padding, border, margin). How does each affect the size of an element?
* **Question 2**: What is the difference between border-box and content-box box-sizing inCSS? Which is the default?

### Lab Assignment

#### Task

Create a profile card layout using the box model. The profile card shouldinclude:

 A profile picture.

 The user’s name and bio.

 A button to "Follow" the user.

#### Additional Requirements:

 Add padding and borders to the elements.

 Ensure the layout is clean and centered on the page using CSS margins.

 Use the box-sizing property to demonstrate both content-box and border- box

on different elements.

**CSS Flexbox**

### Theory Assignment

* **Question 1**: What is CSS Flexbox, and how is it useful for layout design? Explain the terms

flex-container and flex-item.

* **Question 2**: Describe the properties justify-content, align-items, and flex- direction used in Flexbox.

### Lab Assignment

#### Task

Create a simple webpage layout using Flexbox. The layout should include:

 A header.

 A sidebar on the left.

 A main content area in the center.

 A footer.

#### Additional Requirements:

 Use Flexbox to position and align the elements.

 Apply different justify-content and align-items properties to observe theireffects.

 Ensure the layout is responsive, adjusting for smaller screens.

**CSS Grid**

### Theory Assignment

* **Question 1**: Explain CSS Grid and how it differs from Flexbox. When would you use Grid over Flexbox?
* **Question 2**: Describe the grid-template-columns, grid-template-rows, and grid-gap

properties. Provide examples of how to use them.

### Lab Assignment

#### Task

Create a 3x3 grid of product cards using CSS Grid. Each card should contain:

 A product image.

 A product title.

 A price.

#### Additional Requirements:

 Use grid-template-columns to create the grid layout.

 Use grid-gap to add spacing between the grid items.

 Apply hover effects to each card for better interactivity.

**Responsive Web Design with Media Queries**

### Theory Assignment

* **Question 1**: What are media queries in CSS, and why are they important for responsive design?
* **Question 2**: Write a basic media query that adjusts the font size of a webpage for screens smaller than 600px

**Lab Assignment(Task)**

Build a responsive webpage that includes:

 A navigation bar.

 A content section with two columns.

 A footer.

#### Additional Requirements:

 Use media queries to make the webpage responsive for mobile devices.

 On smaller screens (below 768px), stack the columns vertically.

 Adjust the font sizes and padding to improve readability on mobile.

**Typography and Web Fonts**

### Theory Assignment

* **Question 1**: Explain the difference between web-safe fonts and custom web fonts. Whymight you use a web-safe font over a custom font?
* **Question 2**: What is the font-family property in CSS? How do you apply a custom GoogleFont to a webpage?

### Lab Assignment

#### Task

Create a blog post layout with the following:

 A title, subtitle, and body content.

 Use at least two different fonts (one for headings, one for body content).

 Style the text to be responsive and easy to read.

#### Additional Requirements:

 Use a custom font from Google Fonts.

 Adjust line-height, font-size, and spacing for improved readability.

# Module 5 – Frontend – HTML5

### Theory Assignment

* **Question 1**: Difference b/w HTML & HTML5?
* **Question 2**: What are the additional tags used in HTML5?

**Lab Assignment(Task)**

Create a audio video tag

 Also applied properties like muted loop autoplay

 Create some shape using canvas tag in html

 Create some shape using svg tag in html

# Module 8) JavaScript

**JavaScript Introduction**

### Theory Assignment

* **Question 1**: What is JavaScript? Explain the role of JavaScript in web development.
* **Question 2**: How is JavaScript different from other programming languages like Python orJava?
* **Question 3**: Discuss the use of <script> tag in HTML. How can you link an external JavaScript file to an HTML document?

**Lab Assignment**(Task)

 Create a simple HTML page and add a <script> tag within the page.

 Write JavaScript code to display an alert box with the message "Welcome toJavaScript!" when the page loads.

**Variables and Data Types**

### Theory Assignment

* **Question 1**: What are variables in JavaScript? How do you declare a variable using var, let,and const?
* **Question 2**: Explain the different data types in JavaScript. Provide examples for each.
* **Question 3**: What is the difference between undefined and null in JavaScript?

**Lab Assignment** (Task)

 Write a JavaScript program to declare variables for different data types (string,number, boolean, null, and undefined).

 Log the values of the variables and their types to the console using console.log().

**JavaScript Operators**

### Theory Assignment

* **Question 1**: What are the different types of operators in JavaScript? Explain with examples.
  + Arithmetic operators
  + Assignment operators
  + Comparison operators
  + Logical operators
* **Question 2**: What is the difference between == and === in JavaScript?

**Lab Assignment**(Task)

Create a JavaScript program to perform the following:

 Add, subtract, multiply, and divide two numbers using arithmetic operators.

 Use comparison operators to check if two numbers are equal and if onenumber is greater than the other.

 Use logical operators to check if both conditions (e.g., a > 10 and b < 5)are true.

**Control Flow (If-Else, Switch)**

### Theory Assignment

* **Question 1**: What is control flow in JavaScript? Explain how if-else statements work withan example.
* **Question 2**: Describe how switch statements work in JavaScript. When should you use a

switch statement instead of if-else?

### Lab Assignment

 Task 1:

 Task 2:

Write a JavaScript program to check if a number is positive, negative, or zero usingan

if-else statement.

Create a JavaScript program using a switch statement to display the day of theweek based on the user input (e.g., 1 for Monday, 2 for Tuesday, etc.).

**Loops (For, While, Do-While)**

### Theory Assignment

* **Question 1**: Explain the different types of loops in JavaScript (for, while, do-while). Provide abasic example of each.
* **Question 2**: What is the difference between a while loop and a do-while loop?

### Lab Assignment

 Task 1:

Write a JavaScript program using a for loop to print numbers from 1 to 10.

 Task 2:

Create a JavaScript program that uses a while loop to sum all even numbers

between 1 and 20.

 Task 3:

Write a do-while loop that continues to ask the user for input until they enter a number greater than 10.

**Functions**

### Theory Assignment

* **Question 1**: What are functions in JavaScript? Explain the syntax for declaring and calling a function.
* **Question 2**: What is the difference between a function declaration and a function expression?
* **Question 3**: Discuss the concept of parameters and return values in functions.

### Lab Assignment

 Task 1:

 Task 2:

Write a function greetUser that accepts a user’s name as a parameter and displaysa greeting message (e.g., "Hello, John!").

Create a JavaScript function calculateSum that takes two numbers as parameters,adds them, and returns the result.

**Arrays**

### Theory Assignment

* **Question 1**: What is an array in JavaScript? How do you declare and initialize an array?
* **Question 2**: Explain the methods push(), pop(), shift(), and unshift() used in arrays.

### Lab Assignment

 Task 1:

* Declare an array of fruits (["apple", "banana", "cherry"]). Use JavaScript to:
* Add a fruit to the end of the array.
* Remove the first fruit from the array.
* Log the modified array to the console.

 Task 2:

* Write a program to find the sum of all elements in an array of numbers.

**Objects**

### Theory Assignment

* **Question 1**: What is an object in JavaScript? How are objects different from arrays?
* **Question 2**: Explain how to access and update object properties using dot notation and bracket notation.

### Lab Assignment

Task:

 Create a JavaScript object car with properties brand, model, and year. UseJavaScript to:

* Access and print the car’s brand and model.
* Update the year property.
* Add a new property color to the car object.

**JavaScript Events**

### Theory Assignment

* **Question 1**: What are JavaScript events? Explain the role of event listeners.
* **Question 2**: How does the addEventListener() method work in JavaScript? Provide an example.

### Lab Assignment

Task

 Create a simple webpage with a button that, when clicked, displays an alert saying"Button clicked!" using JavaScript event listeners.

**DOM Manipulation**

### Theory Assignment

* **Question 1**: What is the DOM (Document Object Model) in JavaScript? How does JavaScript interact with the DOM?
* **Question 2**: Explain the methods getElementById(), getElementsByClassName(),and

querySelector() used to select elements from the DOM.

### Lab Assignment

Task:

 Create an HTML page with a paragraph (<p>) that displays "Hello, World!".

 Use JavaScript to:

* Change the text inside the paragraph to "JavaScript is fun!".
* Change the color of the paragraph to blue.

**JavaScript Timing Events (setTimeout, setInterval)**

### Theory Assignment

* **Question 1**: Explain the setTimeout() and setInterval() functions in JavaScript. Howare they used for timing events?
* **Question 2**: Provide an example of how to use setTimeout() to delay an action by 2 seconds.

### Lab Assignment

 Task 1:

* Write a program that changes the background color of a webpage after 5 secondsusing

setTimeout().

 Task 2:

* Create a digital clock that updates every second using setInterval().

**JavaScript Error Handling**

### Theory Assignment

* + **Question 1**: What is error handling in JavaScript? Explain the try, catch, and finally

blocks with an example.

* + **Question 2**: Why is error handling important in JavaScript applications?

### Lab Assignment

Task:

* Write a JavaScript program that attempts to divide a number by zero. Use try- catch to handle the error and display an appropriate error message.

# Module 9- Introduction to React.js

**THEORY EXERCISE**

* + **Question 1**: What is React.js? How is it different from other JavaScript frameworks and libraries?
  + **Question 2**: Explain the core principles of React such as the virtual DOM and component- based architecture.
  + **Question 3**: What are the advantages of using React.js in web development?

**LAB EXERCISE**

* + Task:
    - Set up a new React.js project using create-react-app.
    - Create a basic component that displays "Hello, React!" on the web page.

## JSX (JavaScript XML)

**THEORY EXERCISE**

* + **Question 1**: What is JSX in React.js? Why is it used?
  + **Question 2**: How is JSX different from regular JavaScript? Can you write JavaScript insideJSX?
  + **Question 3**: Discuss the importance of using curly braces {} in JSX expressions.

**LAB EXERCISE**

* + Task:
    - Create a React component that renders the following JSX elements:
      * A heading with the text "Welcome to JSX".
      * A paragraph explaining JSX with dynamic data (use curly braces to insert variables).

## Components (Functional & Class Components)

**THEORY EXERCISE**

* + **Question 1**: What are components in React? Explain the difference between **functional components** and **class components**.
  + **Question 2**: How do you pass data to a component using props?
  + **Question 3**: What is the role of render() in class components?

**LAB EXERCISE**

* + Task 1:
    - Create a functional component Greeting that accepts a name as a prop and displays "Hello, [name]!".
  + Task 2:
    - Create a class component WelcomeMessage that displays "Welcome to React!" anda

render() method.

## Props and State

**THEORY EXERCISE**

* + **Question 1**: What are props in React.js? How are props different from state?
  + **Question 2**: Explain the concept of state in React and how it is used to manage componentdata.
  + **Question 3**: Why is this.setState() used in class components, and how does it work?

**LAB EXERCISE**

* + Task 1:
    - Create a React component UserCard that accepts name, age, and location asprops and displays them in a card format.
  + Task 2:
    - Create a Counter component with a button that increments a count value usingReact state. Display the current count on the screen.

## Handling Events in React

**THEORY EXERCISE**

* + **Question 1**: How are events handled in React compared to vanilla JavaScript? Explain the concept of synthetic events.
  + **Question 2**: What are some common event handlers in React.js? Provide examples of

onClick, onChange, and onSubmit.

* + **Question 3**: Why do you need to bind event handlers in class components?

**LAB EXERCISE**

* + Task 1:
    - Create a button in a React component that, when clicked, changes the text from"Not Clicked" to "Clicked!" using event handling.
  + Task 2:
    - Create a form with an input field in React. Display the value of the input field dynamically as the user types in it.

## Conditional Rendering

**THEORY EXERCISE**

* + **Question 1**: What is conditional rendering in React? How can you conditionally render elements in a React component?
  + **Question 2**: Explain how if-else, ternary operators, and && (logical AND) are used in JSXfor conditional rendering.

**LAB EXERCISE**

* + Task 1:
    - Create a component that conditionally displays a login or logout button based onthe user’s login status.
  + Task 2:
    - Implement a component that displays a message like "You are eligible to vote" if theuser is over 18, otherwise display "You are not eligible to vote."

## Lists and Keys

**THEORY EXERCISE**

* + **Question 1**: How do you render a list of items in React? Why is it important to use keyswhen rendering lists?
  + **Question 2**: What are keys in React, and what happens if you do not provide a unique key?

**LAB EXERCISE**

* + Task 1:
    - Create a React component that renders a list of items (e.g., a list of fruit names). Usethe

map() function to render each item in the list.

* + Task 2:
* Create a list of users where each user has a unique id. Render the user list usingReact and assign a unique key to each user.

## Forms in React

**THEORY EXERCISE**

* + **Question 1**: How do you handle forms in React? Explain the concept of controlled components.
  + **Question 2**: What is the difference between controlled and uncontrolled components in React?

**LAB EXERCISE**

* + Task 1:
    - Create a form with inputs for name, email, and password. Use state to control the form and display the form data when the user submits it.
  + Task 2:
    - Add validation to the form created above. For example, ensure that the email input contains a valid email address.

## Lifecycle Methods (Class Components)

**THEORY EXERCISE**

* + **Question 1**: What are lifecycle methods in React class components? Describe the phases of a component’s lifecycle.
  + **Question 2**: Explain the purpose of componentDidMount(), componentDidUpdate(),and componentWillUnmount().

**LAB EXERCISE**

* + Task 1:
    - Create a class component that fetches data from an API when the component mounts using componentDidMount(). Display the data in the component.
  + Task 2:
    - Implement a component that logs a message to the console when it updates using componentDidUpdate(). Log another message when the component unmounts using componentWillUnmount().

## Hooks (useState, useEffect, useReducer, useMemo, useRef, useCallback)

**THEORY EXERCISE**

* + **Question 1**: What are React hooks? How do useState() and useEffect() hooks work in functional components?
  + **Question 2**: What problems did hooks solve in React development? Why are hooks considered an important addition to React?
  + **Question 3**: What is useReducer ? How we use in react app?
  + **Question 4:** What is the purpose of useCallback & useMemo Hooks?
  + **Question 5:** What’s the Difference between the useCallback & useMemo Hooks?
  + **Question 6 :** What is useRef ? How to work in react app?

**LAB EXERCISE**

* + Task 1:
    - Create a functional component with a counter using the useState() hook. Include buttons to increment and decrement the counter.
  + Task 2:
    - Use the useEffect() hook to fetch and display data from an API when the component mounts.
  + Task 3:
    - Create react app with use of useSelector & useDispatch.
  + Task 4:
    - Create react app to avoid re-renders in react application by useRef?

## Routing in React (React Router)

**THEORY EXERCISE**

* + **Question 1**: What is React Router? How does it handle routing in single-page applications?
  + **Question 2**: Explain the difference between BrowserRouter, Route, Link, and Switch

components in React Router.

**LAB EXERCISE**

* + Task 1:
    - Set up a basic React Router with two routes: one for a Home page and one for an About page. Display the appropriate content based on the URL.
  + Task 2:
    - Create a navigation bar using React Router’s Link component that allows users to switch between the Home, About, and Contact pages.

## React – JSON-server and Firebase Real Time Database

**THEORY EXERCISE**

* + **Question 1:** What do you mean by RESTful web services?
  + **Question 2:** What is Json-Server? How we use in React ?
  + **Question 3**: How do you fetch data from a Json-server API in React? Explain the role of

fetch() or axios() in making API requests.

* + **Question 4:** What is Firebase? What features does Firebase offer?
  + **Question 5**: Discuss the importance of handling errors and loading states when working with APIs in React

**LAB EXERCISE**

* + Task 1:
    - Create a React component that fetches data from a public API (e.g., a list of users)and displays it in a table format.
    - Create a React app with Json-server and use Get , Post , Put , Delete & patch method on Json-server API.
  + Task 2:
    - Create a React app crud and Authentication with firebase API.
    - Implement google Authentication with firebase API.
  + Task 3:
    - Implement error handling and loading states for the API call. Display a loading spinner while the data is being fetched.

## Context API

**THEORY EXERCISE**

* + **Question 1**: What is the Context API in React? How is it used to manage global state across multiple components?
  + **Question 2**: Explain how createContext() and useContext() are used in React for sharing state.

**LAB EXERCISE**

* + Task 1:
    - Create a simple theme toggle (light/dark mode) using the Context API. The themestate should be shared across multiple components.
  + Task 2:
    - Use the Context API to create a global user authentication system. If the user is logged in, display a welcome message; otherwise, prompt them to log in.

## State Management (Redux, Redux-Toolkit or Recoil)

**THEORY EXERCISE**

* + **Question 1**: What is Redux, and why is it used in React applications? Explain the core concepts of actions, reducers, and the store.
  + **Question 2**: How does Recoil simplify state management in React compared to Redux?

**LAB EXERCISE**

* + Task 1:
    - Create a simple counter application using Redux for state management. Implement actions to increment and decrement the counter.
  + Task 2:
    - Build a todo list application using Recoil for state management. Allow users to add, remove, and mark tasks as complete.
  + Task 3:
    - Build a crud application using Redux-Toolkit for state management. Allow users to add, remove, delete and update.