Module - 1

Theory Assignment

HTML Basic:

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

HTML (HyperText Markup Language) is the standard markup language used to create and structure content on the web. It defines the structure of web pages using **elements** like headings, paragraphs, links, images, tables, and more.

Purpose of HTML in Web Development:

1. **Structuring Content:**
   - ○ HTML provides the basic structure of a webpage, such as headings, paragraphs, lists, and sections.
2. **Embedding Media:**
   - ○ It allows embedding images, videos, audio, and other media types.
3. **Creating Links (Hypertext):**
   - ○ Enables navigation between different pages or sections using hyperlinks.
4. **Form Creation:**
   - ○ HTML is used to create forms for user input (e.g., contact forms, login forms).
5. **Semantic Meaning:**
   - ○ It uses semantic tags like `<header>`, `<footer>`, `<article>`, etc., to give meaning to content, which improves SEO and accessibility.
6. **Foundation for Styling and Scripting:**
   - ○ HTML works with **CSS** for styling and **JavaScript** for interactivity, forming the core trio of web development.

• Question 2: Explain the basic structure of an HTML document. Identify the mandatory tag their purpose

Basic Structure of an HTML Document

An HTML document is structured using a specific set of tags that define the layout and content of a webpage. The basic structure includes several mandatory tags that browsers use to interpret and render the content correctly.

Basic HTML Structure:

```
<!DOCTYPE html>

 <html>

 <head>

    <title>Page Title</title>

</head>

 <body>

        <h1>Welcome to My Website</h1>

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

</body>

</html>
```

Mandatory HTML Tags and Their Purposes:

Tag	Purpose
`<!DOCTYPE html>`	Declares the document type and HTML version (HTML5 here). Helps browsers render the page correctly.
`<html>`	Root element of the HTML document. All content is placed inside this tag.

`<head>`	Contains meta-information about the document (like title, charset, styles). Not visible on the page.
`<title>`	Sets the title of the page shown in the browser tab.
`<body>`	Contains all the visible content like headings, paragraphs, images, etc.

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

Block-Level Elements

Definition:
Block-level elements occupy the entire width of their container and always start on a new line.

Characteristics:

- Take up full width by default.
- Stack vertically.
- Can contain other block-level and inline elements.

Examples:

- `<div>` – a generic container
- `<p>` – paragraph
- `<h1>` to `<h6>` – headings
- `<ul>`, `<ol>` – lists
- `<li>` – list item
- `<section>`, `<article>` – semantic content blocks

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

```
<div>This is a div block.</div>
```

Inline Elements

Definition:
Inline elements only take up as much width as necessary and do not start on a new line.

Characteristics:

- Stay in line with other elements.
- Only contain text or other inline elements.
- Do not force a line break.

Examples:

- `<span>` – generic inline container
- `<a>` – hyperlink
- `<strong>`, `<em>` – formatting text
- `<img>` – inline image
- `<label>` – form labels

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

Role of Semantic HTML

Semantic HTML refers to using HTML tags that convey meaning about the content they contain. These tags not only define how the content looks but also describe its purpose to browsers, developers, and assistive technologies (like screen readers).

Why Is Semantic HTML Important

1. Accessibility

- Helps screen readers and other assistive technologies understand the page structure.
- Improves the experience for users with disabilities (e.g., blind or visually impaired users).

2. SEO (Search Engine Optimization)

- Search engines like Google use semantic tags to better index and rank web pages.
- Clear structure helps crawlers understand the importance of content, like headings, navigation, and main sections.

3. Readability & Maintainability

- Makes the code more readable and easier to maintain for developers.
- Clearly communicates the role of each part of a webpage.

Examples of Semantic Elements:

Tag	Purpose
`<header>`	Represents the introductory content (like logos, titles, nav)
`<nav>`	Represents navigation links
`<main>`	Main content area of the document
`<article>`	Independent, self-contained content (e.g., blog post)
`<section>`	Groups related content within a page
`<aside>`	Side content like ads, sidebars, or quotes
`<footer>`	Footer section of the document or section

HTML Form:

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

What are HTML Forms Used :

HTML forms are used to collect user input on websites. They are the main way users submit data to a web server, such as:

- Registering for an account
- Logging in
- Searching content
- Sending feedback
- Uploading files

Forms act as a container that wraps around different input elements to gather data.

Key Form Elements and Their Purposes

1. `<input>` Element

- Used to create various types of input fields like:
  - `text` (for single-line input)
  - `email`, `password`, `number`, `checkbox`, `radio`, etc.
- Purpose: Allows users to enter data (e.g., name, email, password).

Example:

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

2. `<textarea>` Element

- Used for multi-line text input (e.g., comments, descriptions).
- Purpose: Lets users enter longer messages or feedback.

Example:

```
<textarea name="message" rows="5" cols="30">Enter your
message...</textarea>
```

3. `<select>` Element

- Creates a drop-down list of options.
- Paired with `<option>` tags for each item.
- Purpose: Allows users to choose from a predefined list.

Example:

```
<select name="country">
  <option value="india">India</option>
  <option value="usa">USA</option>
  <option value="uk">UK</option>
</select>
```

4. `<button>` Element

- Represents a clickable button.
- Types:
  - `submit`: submits the form
  - `reset`: resets all form fields
  - `button`: general-purpose button (often used with JavaScript)
- Purpose: Allows users to perform actions like submitting a form or resetting inputs.

Example:

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

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

GET Method:

Definition:
Sends form data as part of the URL in a query string.

Characteristics:

- Appends form data to the URL
- Data is visible in the browser's address bar.
- Limited data length (because URLs have a size limit).
- Can be bookmarked or shared.
- Less secure (data is exposed).

When to Use:

- For search forms or data retrieval where no sensitive info is involved.
- When you want to bookmark or share the result via URL.

Example:

```
<form action="/search" method="get">
  <input type="text" name="query"
placeholder="Search...">
  <button type="submit">Search</button>
</form>
```

2. POST Method :

Definition:
Sends form data in the body of the HTTP request (not visible in the URL).

Characteristics:

- Data is not visible in the URL.

- No data length limit.
- More secure for sending sensitive information (like passwords).
- Cannot be bookmarked.
- Suitable for forms that modify server data.

When to Use:

- For login forms, registration, file uploads, and payments.
- When you need to send confidential or large amounts of data.

Example:

```
<form action="/login" method="post">
  <input type="text" name="username">
  <input type="password" name="password">
  <button type="submit">Login</button>
</form>
```

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

Primary Purposes of `<label>`:

1. Describes the Purpose of an Input:
   - It tells users what the input field is for.
   - Makes forms clearer and more user-friendly.
2. Connects Text with Form Controls:
   - When a label is clicked, the associated form control is focused automatically.

    - This helps improve usability, especially on mobile devices.

3. Improves Accessibility:
    - Screen readers use the `<label>` to read out the purpose of each input field to visually impaired users.
    - Helps users with disabilities navigate and understand forms effectively.

How to Use `<label>` Correctly:

There are two ways to associate a label with a form control:

1. Using the `for` Attribute (Explicit Association):

```
<label for="email">Email Address:</label>
<input type="email" id="email" name="email">
```

- The `for` value matches the `id` of the input.

2. Wrapping the Input Inside the Label (Implicit Association):

```
<label>
  Email Address:
  <input type="email" name="email">
</label>
```

Accessibility Benefits:

Feature	Benefit
Label + Input	Screen readers announce the label when the input is focused.
Clickable Label	Improves usability for all users, especially on touch devices.
Structured Forms	Helps users understand form layout and required data.

Example for Accessibility:

```
<form>
```

```
  <label for="username">Username:</label>
  <input type="text" id="username" name="username">
</form>
```

HTML Table :

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

Structure of an HTML Table and Purpose of Elements

HTML tables are used to display data in a grid format made up of rows and columns. Tables are helpful for organizing structured information like product listings, schedules, or reports.

Basic Table Structure:

An HTML table is made up of several key elements:

```
<table>
  <thead>
    <tr>
      <th>Header 1</th>
      <th>Header 2</th>
    </tr>
  </thead>
  <tr>
    <td>Data 1</td>
    <td>Data 2</td>
  </tr>
</table>
```

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

Difference Between `colspan` and `rowspan` in HTML Tables:

In HTML tables, `colspan` and `rowspan` are attributes used in `<td>` or `<th>` elements to merge cells across multiple columns or rows.

`colspan` – Column Span:

Definition:
`colspan` allows a cell to span across multiple columns (i.e., left to right).

Use Case:
Useful when you want a single cell to take the place of multiple columns in the same row.

Example:

```
<table border="1">
  <tr>
    <th colspan="2">Name & Age</th>
  </tr>
  <tr>
    <td>John</td>
    <td>25</td>
  </tr>
</table>
```

`rowspan` – Row Span :

Definition:
`rowspan` allows a cell to span across multiple rows (i.e., top to bottom).

Use Case:
Useful when a single cell should cover multiple rows in the same column.

Example:

```
<table border="1">
  <tr>
    <th rowspan="2">Name</th>
    <td>John</td>
  </tr>
  <tr>
    <td>25</td>
  </tr>
</table>
```

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

Why Tables Should Be Used Sparingly for Layout Purposes:

While HTML tables were once commonly used for web page layout, this practice is now discouraged for several important reasons:

Problems with Using Tables for Layout:

1. Not Semantic
   - Tables are meant for tabular data, not for positioning elements.
   - Misusing them confuses screen readers and reduces accessibility.
2. Hard to Maintain
   - Table layouts create complex, nested code that is difficult to edit or update.
   - A small change in structure may require editing multiple table rows or cells.
3. Poor Responsiveness
   - Tables do not adapt well to different screen sizes (e.g., mobile devices).
   - They make responsive web design harder.
4. Slow Loading & Rendering
   - Tables must be fully loaded before the content is displayed.
   - This can slow down page rendering, especially on slow networks.
5. Styling Limitations
   - It's harder to apply flexible styling and animations using tables compared to modern layout tools.

Better Alternative: Use CSS for Layout

CSS (Cascading Style Sheets) offers modern layout techniques that are:

- Flexible
- Responsive

- Accessible
- Easier to maintain