**Module 2 – Frontend - HTML**

**HTML Basics**

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

HTML stands for **HyperText Markup Language**.

* **HyperText** → Text that contains links (hyperlinks) to other documents.
* **Markup** → Special tags (<tag>) used to structure and format content.
* **Language** → A set of rules that browsers can interpret.

1. **Creates Structure of Web Pages**
   * Acts like the **skeleton** of a website.
   * Defines elements such as headings, paragraphs, images, links, and lists.
2. **Connects with CSS and JavaScript**
   * HTML provides structure.
   * CSS (Cascading Style Sheets) adds design and styling.
   * JavaScript adds interactivity and dynamic behavior.
3. **Supports Hyperlinks and Navigation**
   * Allows linking between different pages or resources (<a> tag).

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

**Explanation of Mandatory Tags**

1. **<!DOCTYPE html>**
   * **Declares the type of document (HTML5).**
   * **Informs the browser how to interpret the page.**
2. **<html> … </html>**
   * The root element of the HTML page.
   * All code (head + body) must be inside this tag.
   * lang="en" specifies language (useful for accessibility and SEO).
3. **<head> … </head>**
   * **Contains metadata (information about the document, not shown directly on the page).**
   * Examples inside <head>:
     + <title> → Sets the page title (shown in browser tab).
     + <meta> → Provides extra info (like character set, description).
     + <link> → Connects CSS files.
     + <script> → Links JavaScript**.**
4. **<title> … </title>**
   * Defines the title of the webpage.
   * Important for SEO and user navigation.
5. **<body> … </body>**
   * Contains the visible content of the webpage.
   * All headings, paragraphs, images, links, tables, forms, etc. go here.

**3: What is the difference between block-level elements and inline elements in HTML?**

**Block-Level Elements**

* Start on a new line in the browser.
* By default, they take up the entire width of the container (100% width).
* Can contain other block-level elements and inline elements.
* Used for creating sections, paragraphs, layouts.

Examples of Block-Level Elements

* <div> → Generic block container
* <p> → Paragraph
* <h1> to <h6> → Headings
* <ul>, <ol>, <li> → Lists
* <section>, <article>, <header>, <footer>

**Inline Elements**

* Do not start on a new line.
* Only take up as much width as needed.
* Usually contain text or small pieces of content.
* Cannot contain block-level elements (only text or other inline elements).

Examples of Inline Elements

* <span> → Generic inline container
* <a> → Link
* <strong> → Bold text
* <em> → Italic text
* <img> → Image
* <label> → Form label

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

**What is Semantic HTML?**

* **Semantic HTML** means using **meaningful HTML tags** that describe the purpose of the content.
* Instead of using only <div> and <span> (which don’t explain meaning), developers use tags like <header>, <nav>, <article>, <footer>.
* This makes the code **self-explanatory** to both humans and machines.
  + .

1. **Boosts SEO (Search Engine Optimization)**
   * Search engines use semantic tags to understand the structure and importance of content.
   * Example: <h1> is the most important heading → improves page ranking.

**Examples of Semantic Elements in HTML**

* <header> → Defines the header of a page or section.
* <nav> → Navigation links (menus, site links).
* <main> → Main content of the webpage.
* <article> → Independent article or blog post.
* <section> → A thematic group of content.
* <aside> → Side content (ads, related links, sidebar).
* <footer> → Footer (copyright, contact info).
* <figure> & <figcaption> → Images with captions.

**HTML Forms**

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

**What are HTML Forms?**

* An **HTML form** is used to **collect user input** on a webpage.
* Forms send the collected data to a server for processing (e.g., login, registration, search, feedback).
* The form is created using the <form> tag.

**Purpose of Form Elements**

1. **<input>**
   * Used to create different types of input fields depending on the type attribute.
   * Common types:
     + type="text" → single-line text field
     + type="email" → email input
     + type="password" → password input
     + type="radio" → radio button
     + type="checkbox" → checkbox
     + type="file" → file upload

1. **<textarea>**
   * Creates a **multi-line text box** for longer input (like feedback or comments).
   * Unlike <input type="text">, it allows multiple lines.

1. **<select> (with <option>)**
   * Creates a **dropdown list** of choices.
   * The <option> tag defines the items inside the list.

1. **<button>**
   * Creates a **button** that can submit the form or trigger an action.
   * Types:
     + type="submit" → submits the form.
     + type="reset" → resets/clears form fields.
     + type="button" → general button (can be controlled with JavaScript).

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

**Difference Between GET and POST**

| **Feature** | **GET** | **POST** |
| --- | --- | --- |
| **Data location** | Data is appended to the URL as query parameters. | Data is sent in the request body (not shown in URL). |
| **Visibility** | Visible in the browser’s address bar. | Hidden from the URL (not directly visible). |
| **Data size** | Limited length (about 2000 characters, varies by browser). | No practical size limit (can send large data, including files). |
| **Security** | Less secure – data can be bookmarked, logged, or cached. | More secure – better for sensitive data (though encryption with HTTPS is recommended). |
| **Caching & Bookmarking** | Can be cached and bookmarked with query string. | Cannot be bookmarked with data. |
| **Use case** | Best for retrieving or searching information. | Best for submitting sensitive or large data. |

**When to Use Each**

* **Use GET when:**
  + You want to **retrieve information** only.
  + The data is **not sensitive** (like search queries, filters, pagination).
* **Use POST when:**
  + You are sending **sensitive information** (passwords, personal details).
  + You are uploading **large data** (like forms, images, or files).
  + Example: Login, signup, payment forms.

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

**Purpose of the <label> Element**

* The <label> element defines a **caption (text description)** for an input control in a form.
* It tells the user what the input field is for (e.g., "Name:", "Email:", "Password:").
* It is linked to an input field using the for attribute (which matches the input’s id).

**How <label> Improves Accessibility**

1. **For Screen Readers**
   * Visually impaired users rely on screen readers.
   * Labels give meaningful descriptions, so the screen reader can announce what the input is for.
   * Without labels, the input might be confusing (e.g., just “edit text” without context).
2. **Larger Clickable Area**
   * When <label> is connected to an input, clicking on the label automatically focuses/selects the input.
   * This improves usability, especially for checkboxes and radio buttons.

**HTML Tables**

**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

An HTML table organizes data in rows and columns.

It is made up of a combination of table-related tags.

**Purpose of Each Element**

1. **<table>**
   * The container element for all table content.
   * Defines that the content inside is a table.
   * Attributes like border, cellpadding, cellspacing (in old HTML) or CSS can be used to style it.
2. **<tr> (Table Row)**
   * Defines a row in the table.
   * Each row contains table header cells (<th>) or table data cells (<td>).
3. **<th> (Table Header Cell)**
   * Defines a heading cell in the table.
   * Text inside <th> is bold and centered by default.
   * Usually used in the first row (headings) or first column**.**
4. **<td> (Table Data Cell)**
   * Defines a data cell in the table.
   * Each <td> contains actual data (text, numbers, images, links, etc.).
5. **<thead> (Table Head Section)**
   * Groups the header rows of the table.
   * Helps in structuring the table, especially with large datasets.
   * Useful for styling (CSS) and for screen readers to improve accessibility.

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

**Definition**

In HTML tables, colspan and rowspan are attributes used with <td> or <th> to merge multiple cells**.**

* colspan → Merges columns (horizontal).
* rowspan → Merges rows (vertical).

**Difference Between colspan and rowspan**

| **Feature** |  |  |  | **colspan** |  |  | **rowspan** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Meaning |  |  |  | Extends a cell across multiple columns. |  |  | Extends a cell across multiple rows. |
| Direction |  |  |  | Horizontal merging. |  |  | Vertical merging. |
| Use case |  |  |  | Used when one heading or data should cover multiple columns. |  |  | Used when one heading or data should cover multiple rows. |

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

Why Tables Should Not Be Used for Layout

Originally, developers used HTML tables to create web page layouts (e.g., dividing a page into columns).  
But this practice is not recommended today because:

1. **Not Semantic**
   * Tables are meant for tabular data (rows & columns).
   * Using them for layout misrepresents content structure.
2. **Accessibility Issues**
   * Screen readers and assistive technologies get confused, reading layout tables as if they were real data tables.
   * This makes navigation harder for visually impaired users.
3. **Poor SEO**
   * Search engines cannot easily understand the meaning of content inside layout tables.
   * It reduces search ranking and discoverability.
4. **Less Flexible**
   * Tables are rigid and not responsive by default.
   * Difficult to adapt for mobile screens compared to modern layout techniques.

**Better Alternatives**

**Modern web design uses CSS instead of tables for layout:**

1. **CSS Flexbox**
   * One-dimensional layout (row or column).
   * Great for aligning and distributing space.

1. **CSS Grid**
   * Two-dimensional layout system (rows + columns).
   * Best for complex layouts.