**Module 2 – Mernstack – HTML**

**• HTML Basics**

* **Theory Assignment**

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

HTML stands for HyperText Markup Language. It is the standard language used to create and design the structure of web pages.

**🔹 Definition:**

HTML is a markup language used to structure content on the web. It tells the browser how to display text, images, links, and other elements on a webpage.

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

1. **Structure the Webpage**  
   HTML defines the layout and structure of a webpage using elements like headings, paragraphs, lists, tables, and more.
2. **Embed Multimedia**  
   HTML allows embedding images, audio, video, and other multimedia elements.
3. **Create Links (Hypertext)**  
   HTML enables linking to other pages or websites using anchor (<a>) tags.
4. **Form Handling**  
   HTML provides form elements like text boxes, checkboxes, buttons, etc., to collect user input.
5. **Semantic Meaning**  
   With HTML5, elements like <header>, <footer>, <article>, and <section> add meaning to content, improving SEO and accessibility.
6. **Foundation for CSS and JavaScript**  
   HTML forms the base structure, which is styled with CSS and made interactive with JavaScript.

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

**Basic Structure of an HTML Document**

An HTML document has a **standard structure** made up of specific tags that define different parts of the webpage. Here is the basic structure with explanations of **mandatory tags**:

**✅ Basic HTML Document Structure:**

<!DOCTYPE html>

<html>

<head>

<title>Page Title</title>

</head>

<body>

<h1>Welcome to My Website</h1>

<p>This is a basic HTML page.</p>

</body>

</html>

**🔹 Mandatory Tags and Their Purposes**

|  |  |
| --- | --- |
| **Tag** | **Purpose** |
| <!DOCTYPE html> | Declares the document type and version of HTML (HTML5 here). Helps browsers render the page correctly. |
| <html> | The root element that wraps all the content of the HTML page. |
| <head> | Contains meta-information about the document (not displayed on the page), such as title, character set, styles, and scripts. |
| <title> | Sets the title of the web page shown in the browser tab. |
| <body> | Contains the content that appears on the web page (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

* Start on a new line and take up the full width of their parent container (from left to right).
* Used to structure the layout of a web page.
* Can contain other block-level and inline elements.

#### Examples:

* <div>
* <p>
* <h1> to <h6>
* <ul>, <ol>, <li>
* <table>, <form>, <section>, <article>

#### Example:

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

<div>This is a block-level div.</div>

<h1>This is a heading</h1>

### 2. Inline Elements

* Do not start on a new line**.**
* Only take up as much width as necessary.
* Usually used for styling or formatting small parts of content.
* Can only contain text or other inline elements (not block elements).

#### Examples:

* <span>
* <a>
* <strong>, <em>
* <img>
* <label>, <input>

#### Example:

<p>This is a <strong>bold</strong> word in a sentence.</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.**

### What is Semantic HTML?

**Semantic HTML** uses meaningful tags that clearly describe the purpose of the content inside them. Unlike generic tags like <div> or <span>, semantic tags tell both browsers and developers what the content represents.

### Importance of Semantic HTML

#### 1. Accessibility

* Helps screen readers and assistive technologies understand the page content.
* Makes it easier for users with disabilities to navigate and interact with the website.
* Example: A screen reader can identify <nav> as a navigation section.

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

* Search engines like Google use semantic tags to understand page structure andcontent.
* Improves how search engines index and rank your site.
* Helps in featured snippets and better visibility.

#### 3. Code Clarity & Maintenance

* Easier for developers to read, understand, and maintain code.
* Reduces the need for excessive class names or comments.

### Examples of Semantic HTML Elements

| **Element** | **Purpose** |
| --- | --- |
| <header> | Defines the header of a page or section. |
| <nav> | Defines navigation links. |
| <main> | Represents the main content of the page. |
| <article> | Represents independent content (e.g., blog post). |
| <section> | Defines a section in a document. |
| <aside> | Contains side content (e.g., sidebar, ads). |
| <footer> | Defines the footer of a page or section. |
| <figure> | Groups media elements (e.g., image with caption). |
| <figcaption> | Caption for the <figure> content. |

* **HTML Forms**

**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.

🔹 Example use cases:

* User **registration**
* **Login** forms
* **Feedback** or **contact** forms
* **Search boxes**
* **Order** or **booking** forms

Forms are created using the <form> tag and contain various input elements.

### 🔹 Main Form Elements and Their Purposes

#### 1. <input>

Used to accept **short user input**, such as:

* Text
* Email
* Password
* Checkbox
* Radio buttons
* Numbers
* File uploads

🔸 **Example:**

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

#### 2. <textarea>

Used to accept long, multi-line text (like messages or comments).

🔸 **Example:**

<textarea name="message" rows="4" cols="50">Enter your message here...</textarea>

#### 3. <select> and <option>

Used to create a dropdown list where the user can select 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>*

Used to submit a form or perform other actions (like reset or custom JavaScript).

🔸 **Example:**

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

**Question 2: Explain the difference between the GETand POSTmethods in form submission. When should each be used?**

## GET vs POST – Key Differences

|  |  |  |
| --- | --- | --- |
| **Feature** | **GET** | **POST** |
| **Data Visibility** | Data is visible in the URL (e.g., ?name=abc) | Data is hidden in the body of the request |
| **Length Limit** | Has a limit on the amount of data sent | Can send large amounts of data |
| **Security** | Less secure (data shown in URL) | More secure (data not shown in URL) |
| **Use in URLs** | Can be bookmarked and cached | Cannot be bookmarked or cached |
| **Usage Purpose** | For retrieving or querying data | For submitting or updating data |

## When to Use GET:

* When the form is used to fetch or search data.
* When the data is not sensitive .
* When you want the URL to be shareable or bookmarkable.

**Example:**

<form method="GET" action="search.php">

## When to Use POST:

* When the form is used to **submit**, **insert**, or **update** data.
* When the data is **sensitive or large** (e.g., passwords, login, file uploads).
* When you don’t want data to appear in the URL.

**Example:**

<form method="POST" action="submit.php">

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

The **purpose of the <label> element** in a form is to provide a clear, descriptive text for form controls like input fields, checkboxes, and radio buttons.

**How <label> improves accessibility:**

1. **Associates text with form controls:**  
   It explicitly links a visible label with the corresponding input element, so users know what to enter or select.
2. **Helps screen readers:**  
   Screen readers read the label aloud when the user focuses on the input, making the form understandable to visually impaired users.
3. **Increases clickable area:**  
   Clicking on the label text focuses or activates the associated input, which is especially helpful for checkboxes and radio buttons.

**Example:**

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

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

* The label with for="email" connects to the input with id="email".

**• HTML Tables**

**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 Key Elements**

An **HTML table** is used to display data in a **tabular format** (rows and columns). Below is an explanation of the structure and purpose of key table-related elements:

**1. <table>**

* **Purpose**: The main container element that defines a table in HTML.
* **Usage**: All rows (<tr>) and other table sections like <thead>, <tbody>, and <tfoot> must be placed inside this element.

<table>

<!-- Table rows go here -->

</table>

**2. <tr> (Table Row)**

* **Purpose**: Defines a **row** in the table.
* **Usage**: Placed inside <table>, <thead>, <tbody>, or <tfoot>. Contains <th> or <td> cells.

<tr>

<!-- Table cells go here -->

</tr>

**🔹 3. <th> (Table Header)**

* **Purpose**: Represents a **header cell** in a table.
* **Usage**: Usually placed in the <thead> section. The text inside <th> is **bold and centered** by default.

<tr>

<th>Student Name</th>

<th>Grade</th>

</tr>

**🔹 4. <td> (Table Data)**

* **Purpose**: Defines a **data cell** in a table.
* **Usage**: Used inside <tr> for entering normal (non-header) data.

<tr>

<td>Ravi</td>

<td>A</td>

</tr>

**🔹 5. <thead> (Table Head)**

* **Purpose**: Groups the **header content** (usually column titles) of a table.
* **Usage**: Contains <tr> with <th> elements. Helps with styling and accessibility.

<thead>

<tr>

<th>Name</th>

<th>Age</th>

</tr>

</thead>

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

In HTML tables, both colspan and rowspan are **attributes** used with <td> or <th> elements to **merge cells** — but they work in different directions:

### colspan (Column Span):

* **Purpose**: Merges **multiple columns** into one.
* **Direction**: Horizontal (across columns).
* **Used When**: You want one cell to take the space of multiple columns.

#### Example of colspan:

<table border="1">

<tr>

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

</tr>

<tr>

<td>Raj</td>

<td>25</td>

</tr>

</table>

### 🔹 rowspan (Row Span):

* **Purpose**: Merges **multiple rows** into one.
* **Direction**: Vertical (down rows).
* **Used When**: You want one cell to take the space of multiple rows.

#### Example of rowspan:

<table border="1">

<tr>

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

<td>Raj</td>

</tr>

<tr>

<td>Kiran</td>

</tr>

</table>

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

### Why Not Use Tables for Layout?

HTML **tables were originally designed to display tabular data**, not to create web page layouts. Using tables for layout can cause several problems:

#### 1. **Poor Accessibility**

* Screen readers and assistive technologies may misinterpret layout tables as data tables.
* This creates confusion for visually impaired users.

#### 2. **Hard to Maintain**

* Table-based layouts are **complex** and **nested**, making them harder to edit or debug.
* Small changes in layout may require editing many <td> or <tr> elements.

#### 3. **Not Responsive**

* Tables do **not adapt well to different screen sizes** (e.g., mobile devices).
* Makes responsive design difficult or impossible without a lot of extra code.

#### 4. ****Mixes Content with Presentation****

* Violates the principle of **separation of concerns** (HTML for content, CSS for style/layout).
* Leads to cluttered and unmanageable code.

### ✅ Better Alternative: ****CSS (Cascading Style Sheets)****

Modern websites use **CSS for layout**, especially with these features:

#### 1. ****Flexbox****

* Easy to align items horizontally/vertically.
* Good for 1D layouts (like toolbars, menus).

<div style="display: flex;">

<div>Box 1</div>

<div>Box 2</div>

</div>

#### 2. ****CSS Grid****

* Perfect for 2D layouts (rows and columns).
* More control and flexibility than tables.

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

<div>Left</div>

<div>Right</div>

</div>