**MERN Stack Theory Exercise**

**Module 2 – Mernstack-HTML**

**HTML Basics**

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

Ans> HTML (HyperText Markup Language) is the standard language used to create and structure content on web pages. Its purpose is to define elements like text, images, links, and layout, enabling browsers to display web content correctly. It forms the foundational structure of a webpage, which is then 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.

Ans> **Mandatory Tags and Purposes:**

1. **<!DOCTYPE html>**: Declares the document type and HTML version (HTML5).
2. **<html>**: Root element wrapping all content on the page.
3. **<head>**: Contains metadata (e.g., title, stylesheets, scripts) not displayed on the page.
4. **<title>**: Sets the page title (shown in the browser tab).
5. **<body>**: Holds all visible content (text, images, links, etc.).

These tags define the skeleton of an HTML page, ensuring browsers interpret and render it properly.

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

Ans> **Key Differences**

| **Feature** | **Block-Level Elements** | **Inline Elements** |
| --- | --- | --- |
| **Line Behavior** | Starts on a new line | Stays on the same line |
| **Width** | Takes full width of parent container | Takes only the width of its content |
| **Stacking** | Stacks vertically | Flows horizontally |
| **Typical Use** | Structural (e.g., sections, paragraphs) | Text-level (e.g., links, emphasis) |
| **Can Contain** | Both block and inline elements | Usually only inline elements |

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

Ans> **Role of Semantic HTML**

Semantic HTML involves using HTML tags that convey meaning about the content’s structure and purpose, rather than just its appearance. It enhances readability, maintainability, and compatibility with browsers and assistive technologies.

**Importance for Accessibility**

* Semantic elements (e.g., <nav>, <article>) provide context to screen readers, helping users with disabilities navigate and understand content more easily.
* Clear structure reduces reliance on visual cues alone.

**Importance for SEO**

* Search engines like Google use semantic tags to better understand page content, improving indexing and ranking.
* Meaningful markup (e.g., <header>, <main>) signals key sections, boosting relevance for search queries.

**Examples of Semantic Elements**

* <header>: Defines introductory content or a page header.
* <nav>: Marks navigation links.
* <article>: Represents standalone content (e.g., a blog post).
* <section>: Groups thematic content.
* <footer>: Contains footer information (e.g., copyright).

**HTML Forms**

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

Ans>HTML forms are used to collect user input on a webpage, enabling interactivity. They allow users to submit data (e.g., text, selections, or files) to a server for processing, such as signing up, searching, or submitting feedback. Forms are created using the <form> element and contain various interactive elements like inputs and buttons.

**Purpose of Key Form Elements**

1. **<input>**
   * **Purpose**: A versatile element for collecting single-line user input. Its behavior depends on the type attribute (e.g., text, email, password).
   * **Examples**:
     + <input type="text">: For general text (e.g., name).
     + <input type="email">: For email addresses with validation.
     + <input type="checkbox">: For multiple-choice options.
   * **Use**: Captures short, specific data.
2. **<textarea>**
   * **Purpose**: Allows multi-line text input, ideal for longer responses like comments or messages.
   * **Example**: <textarea rows="4" cols="50">Default text</textarea>
   * **Use**: Collects detailed or freeform text.
3. **<select>**
   * **Purpose**: Creates a dropdown menu for users to choose one (or multiple, with the multiple attribute) options from a predefined list.
   * **Example**:

html

CollapseWrapCopy

<select name="colors">

<option value="red">Red</option>

<option value="blue">Blue</option>

</select>

* + **Use**: Limits input to specific choices, improving consistency.

1. **<button>**
   * **Purpose**: Triggers an action, such as submitting the form or resetting it. Its behavior depends on the type attribute.
   * **Examples**:
     + <button type="submit">Submit</button>: Sends form data.
     + <button type="reset">Reset</button>: Clears form inputs.
     + <button type="button">Click Me</button>: For custom actions (e.g., with JavaScript).
   * **Use**: Controls form submission or other interactions.

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

Ans> **Difference Between GET and POST Methods**

* **GET**:
  + **How it Works**: Appends form data to the URL as query parameters (e.g., ?name=John&age=25).
  + **Visibility**: Data is visible in the URL and browser history.
  + **Security**: Less secure; suitable for non-sensitive data.
  + **Limit**: Limited data length (URL length restrictions).
* **POST**:
  + **How it Works**: Sends form data in the request body, not the URL.
  + **Visibility**: Data is hidden from the URL and history.
  + **Security**: More secure; ideal for sensitive data (e.g., passwords).
  + **Limit**: No practical size limit.

**When to Use Each**

* **GET**:
  + Use for retrieving data (e.g., search queries).
  + Example: <form method="get" action="/search"> for a search bar.
  + Why: Bookmarkable, idempotent (same request, same result).
* **POST**:
  + Use for submitting data that changes the server (e.g., user registration, file uploads).
  + Example: <form method="post" action="/login"> for a login form.
  + Why: Secure, handles larger or sensitive data.

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

Ans> The <label> element describes an input field’s purpose (e.g., "Name" for a text box). It links to the input via the for attribute, matching the input’s id.

**Accessibility Improvement**:

* Screen readers announce the label, helping visually impaired users understand the input’s context.
* Clicking the label focuses the input, aiding motor-impaired users.

**Example**:

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

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

**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>.

Ans> An HTML table organizes data in rows and columns.

- `<table>`: Defines the table container.

- `<tr>`: Creates a table row.

- `<th>`: Defines a header cell (bold, centered by default).

- `<td>`: Defines a data cell.

- `<thead>`: Groups header rows for styling or semantics.

\*\*Example\*\*:

<table>

<thead>

<tr><th>Name</th><th>Age</th></tr>

</thead>

<tr><td>John</td><td>25</td></tr>

</table>

Question 2: What is the Difference Between colspan and rowspan in Tables? Provide Examples

Ans> In HTML tables, colspan and rowspan are attributes used to control how table cells (<td> or <th>) span across multiple columns or rows, altering the table’s layout.

* **colspan (Column Span)**:
  + **Purpose**: Allows a single cell to stretch horizontally across multiple columns. The value of colspan specifies how many columns the cell should cover.
  + **How it Works**: It merges the space of adjacent columns into one wider cell, useful for headers or combined data.
  + **Example**:

<table border="1">

<tr>

<th colspan="2">Weekly Schedule</th> *<!-- Spans 2 columns -->*

</tr>

<tr>

<td>Monday</td>

<td>Tuesday</td>

</tr>

</table>

**Explanation**: The <th> with colspan="2" takes up the width of two columns, creating a single header over "Monday" and "Tuesday". Without colspan, it would only cover one column, leaving the second column empty or misaligned.

* **rowspan (Row Span)**:
  + **Purpose**: Allows a single cell to stretch vertically across multiple rows. The value of rowspan indicates how many rows the cell should occupy.
  + **How it Works**: It merges the space of adjacent rows into one taller cell, often used for grouping or labeling multi-row data.
  + **Example**:

<table border="1">

<tr>

<td rowspan="2">Morning</td> *<!-- Spans 2 rows -->*

<td>9 AM</td>

</tr>

<tr>

<td>10 AM</td> *<!-- No cell under "Morning" due to rowspan -->*

</tr>

</table>

**Explanation**: The <td> with rowspan="2" extends down to cover two rows, aligning "Morning" alongside both "9 AM" and "10 AM". The second <tr> has one fewer <td> because the first column is already occupied by the spanning cell.

**Key Difference**: colspan expands a cell sideways (across columns), while rowspan expands it downward (across rows). They’re often used together for complex table layouts.

Question 3: Why Should Tables Be Used Sparingly for Layout Purposes? What is a Better Alternative?

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

* **Original Purpose**: Tables (<table>) were designed for displaying tabular data (e.g., spreadsheets, schedules), not for structuring entire page layouts like columns or grids. Using them for layout misuses their semantic meaning.
* **Accessibility Issues**: Screen readers interpret tables as data structures. When used for layout, they confuse users relying on assistive technology, as unrelated content (e.g., a sidebar and main text) gets read as if it’s a single dataset.
* **Responsiveness Problems**: Tables have rigid, fixed structures that don’t adapt well to different screen sizes (e.g., mobile devices). Cells can’t easily rearrange or resize without complex workarounds.
* **Code Maintenance**: Layout tables lead to nested, bloated HTML, making it harder to update or style compared to modern alternatives. For example, adjusting a table-based layout often requires changing multiple rows and columns.
* **Historical Context**: In the early web (1990s), tables were a common layout hack because CSS wasn’t advanced. Today, better tools exist.

**Better Alternative: CSS (Flexbox or Grid)**:

* **Why CSS is Better**: CSS layout tools like Flexbox and Grid are designed for visual arrangement, offering flexibility, responsiveness, and cleaner code. They separate structure (HTML) from presentation (CSS), improving maintainability and accessibility.
* **Flexbox**: Great for one-dimensional layouts (rows or columns). It’s simple and adjusts dynamically.