Module 2 – Frontend – HTML

Theory Assignment:

HTML BASICS

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

- **Ans. :**
- **Definition of HTML :**

  HTML stands for HyperText Markup Language and that is because it was specially created for documents with hypertext links likewise web pages use markup language so that's why it is called HTML. HTML defines many – many web pages which contains dozens of elements like Headings, Paragraphs, Links, and Tables etc. The elements are marked up with HTML tags in the source document which corresponds to what displays in the browser window.

- **Purpose :**

1. **Creating Web Pages/Applications**: HTML is standard markup language for creating web pages & applications. It is first step towards making a website. In other words it provides foundation of web pages.

2. **Provides Structure**: Just like human body has skeleton made of bones, HTML is also made up of <Head> and <Body> which contains different elements like headings, paragraphs, and lists etc.

3. **Display Content**: HTML defines how images, and multimedia appear on a webpage through **`<img>`**, **`<audio>`** and **`<video>`** tags to help format and present content effectively. It ensures a visually appealing and well-

organized webpage with **<p>**. This makes information easy to read and interact with.

4. **Enable Form Creation for User Interaction**: HTML provides form elements like **<input>, <textarea>**, and **<button>** to collect user data. Forms are essential for login pages, feedback collection, and online transactions. Combined with CSS and JavaScript, they improve interactivity. This allows businesses to gather and process user information efficiently.

5. **Enable Data Structuring with Tables:** HTML provides **<table>** element to organize information. It helps display data clearly and efficiently. Tables are useful for structured data like schedules and reports.

6. **Enable Responsive Web Design**: HTML, when linked with CSS, helps create responsive web pages. This ensures websites look good on many different screen sizes and devices. Responsive design improves user experience on desktops, tablets, and smartphones. It is essential for modern web development.

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

➔ **Ans.**

- **<u>Basic structure of HTML</u> :**

  By definition – "A Whole is made up of Parts". This is rational truth which doesn't need to be empirically proven, thus we know that HTML which a whole - is made up of tag & elements. Among these elements, it consists necessary/mandatory elements which are required for a systematically constructed webpage. These are fundamental elements which is consist of Document type declaration/html version, root element, head, title & body.

- Very basic elements of HTML which provides structure to it are following:

  1. **<!DOCTYPE html>**
  2. **<html>**
  3. **<head>**
  4. **<title>**
  5. **<body>**

1. **<!DOCTYPE html> :**

Identification: This tag is called Document Type Declaration, it always comes in the first line in an HTML document. It is written as **<!DOCTYPE html>**. It does not have a closing tag so it is self-closing.

Purpose: It declares the document type and version of HTML being used in the HTML document. This tag ensures the browser correctly interprets the page as HTML5.

2. **<html> :**

Identification: This tag is called "Root Element". It comes directly after the doctype tag. It is also called first opening tag. It must have a closing **</html>** tag at the end of the document. It often consists Lang attribute.

Purpose: It contains all other elements within the webpage. It indicates the starting point and ending of the HTML code.

3. **<head> :**

Identification: The head of an HTML document is a part whose content is not displayed in the browser on page loading. It just contains metadata about the HTML document which specifies data about the HTML document. It comes after root element and prior to body element. Typically contains **<Meta>, <title>, <link>,** and **<script>** elements.

Purpose: The HTML element is used as a container for metadata (data about data). It holds the document title, styles, and scripts. It also includes SEO-related elements like **<Meta>** tags.

4. **<title> :**

Identification: Title appears in the browser tab, bookmarks, and search engine results. It is placed inside the **<head>** section. It uses an opening **<title>** and closing **</title>** tag.

Purpose: It helps search engines understand the page content. It also provides a meaningful name for bookmarks.

5. **<body> :**

Identification: It is located inside the **<html>** element, after **<head>.** Starts with **<body>** and ends with **</body>.** Contains visible elements like headings, paragraphs, images, and links.

Purpose: It holds text, images, links, and other elements users interact with. It ensures a structured and readable layout for users.

❖ Here is an **example** of it :

```
<! DOCTYPE html>

<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">

    <title>Title of HTML Page</title>
</head>

<body>
    <h1>Welcome to My Web page</h1>
    <p>This is a simple HTML page structure. Which contains mandatory
    fields</p>

</body>

</html>
```

➔ These elements are **mandatory** for every HTML document to function correctly

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

➔ **Ans.**

➔ **Introduction :**

[1] Block Level : These are the elements, which structure main part of webpage, by dividing a page into coherent blocks. A block-level element always start with new line and takes the full width of web page, from left to right. These elements can contain block-level as well as inline elements.

[2] Inline : Inline elements are those elements, which differentiate the part of a give text and provide it a particular function. These elements does not start with new line and take width as per requirement. The Inline elements are mostly used with other elements.

➔ **Differences:**

Subject	Block Level	Inline
1. Definition	It occupies the full width available, starting on a new line.	It only takes up as much width as necessary, staying in the same line.
2. Nesting Behaviour	It can contain both block and inline elements.	It usually contains only text or other inline elements.
3. Common Usage	It is used to create sections, paragraphs, and containers	It is used to create sections, paragraphs, and containers
4. Width Control	You can have width and height set using CSS.	In this width and height cannot be controlled explicitly (only padding and margins apply).
5. Example	***<h1>…<h6>, <img>, , etc.***	***<td>, <label>, <i>, <b>, <mark>, etc.***

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

➔ **Ans.**

➔ **Meaning/role:** Semantic HTML refers to using HTML elements that convey meaning and structure to both browsers and developers. These elements clearly define their purpose, making web content easier to understand, maintain, and optimize for accessibility and SEO. Instead of using generic ** or ** elements, semantic tags like **<article>, <section>,** and **<nav>** provide structure and meaning to a webpage.

(A) Accessibility:

1. **Improving User Experience for Everyone:** Screen readers and assistive technologies can better interpret and navigate content. It helps visually impaired users understand page sections, improving user experience. It also provides clear landmarks **(<nav>, <header>, <footer>)** for easier browsing.
2. **Enhances Meaning and Context:** The **<article>** and **<section>** help define content roles, allowing screen readers to announce them properly**.** Tags like **<button>, <submit>, <reset>** tells assistive tools that the element is interactive.
3. **Supports Keyboard Navigation:** Semantic elements ensure better tab order and focus states for users who navigate with keyboards. The **<label>** tags improve accessibility for forms by linking text with input fields.

(B) SEO:

1. **Boosting Search Engine Ranking:** Search engines like Google prioritize well-structured, meaningful content. Semantic HTML improves indexing, ranking, and readability. The **<article>, <section>,** and **<aside>** help search engines categorize content properly.
2. **Enhances Rich Snippets and SERP Features:** Input like **<time>** and **<address>** provide additional metadata for events and locations. The **<meta>** tags in the **<head>** section improve search visibility.
3. **Reduces Unnecessary Usage:** Cleaner and well-structured HTML leads to better indexing and ranking which helps search engine crawlers find relevant content faster. A structured page with proper sections and headings keeps users engaged.

➔ **Example :** Here is an example semantic elements in HTML5 :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Semantic HTML Elements Example</title>
</head>
<body>
  <header>
    <h1>My Blog</h1>
    <nav>
      <ul>
        <li><a href="#">Home</a></li>
        <li><a href="#">About</a></li>
        <li><a href="#">Contact</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <article>
      <h2>Understanding Semantic HTML</h2>
      <p>Semantic HTML helps improve accessibility and SEO by providing meaningful elements.</p>
    </article>
  </main>

  <aside>
    <p>Related Articles: <a href="#">HTML Basics</a></p>
  </aside>

  <footer>
    <p>&copy; 2024 My Blog. All rights reserved. </p>
  </footer>
</body>
</html>
```

HTML Forms

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

➔ **Ans.**

➔ **What is a form and what are HTML forms used for?**

Form is a structure which is made up of label & input to collect data. We can know more about it by dividing it into two parts. The first part is the form that you see on the page itself. Forms are made up of buttons, text fields, and pull-down menus (collectively known as form controls) used to collect information from the user. Forms may also contain text and other elements. The other component of a web form is an application or script on the server that processes the information collected by the form and returns an appropriate response. It's what makes the form work.

➔ **Elements of form and their purpose:** Forms are added to web pages using the <form> element. The form element is a container for all the content of the form, including some number of form controls, such as text entry fields and buttons. It may also contain block elements, (h1, p, and lists, for example), however, it may not contain another form element.

1. **<Input>:** The majority of controls are added to a form using the input element. The functionality and appearance of the input element changes based on the type attribute. The input in web form is used to collect user-input. It can collect text, radio, email, checkbox, password etc.

2. **<textarea> :** At times, you'll want your users to be able enter more than just one line of text. For these instances, use the textarea element that is replaced by a multi-line, scrollable text entry box when displayed by the browser. Unlike the empty input element, the textarea element has content between its opening and closing tags. The content of the textarea element is the initial content of the text box when the form is displayed in the browser. In addition to the required name attribute, the textarea element uses the following attributes:

   Rows: Specifies the number of lines of text the area should display. Scrollbars will be provided if the user types more text than fits in the allotted space.

   Cols: Specifies the width of the text area measured in number of characters. The textarea is also called "Multiline text entry control".

3. **<select>:** The select element displays as a pull-down menu by default when no size is specified or if the size attribute is set to 1. In pull-down menus, only one item may be selected. You add both pull-down and scrolling menus to a form with the select element. Whether the menu pulls down or scrolls is the result of how you specify its size and whether you allow more than one option to be selected. Let's take a look at both menu types. The select element is just a container for a number of option elements. The content of the chosen option element is what gets passed to the web application when the form is submitted. If for some reason you want to send a different value than what appears in the menu, use the value attribute to provide an overriding value.

4. **<button>:** There are a number of different kinds of buttons that can be added to web forms. The most fundamental is the submit button. When clicked, the submit button immediately sends the collected form data to the server for processing. The reset button returns the form controls to the state they were in when the form loaded. Both submit and reset buttons are added using the input element. As mentioned earlier, because these buttons have specific functions that do not include the entry of data, they are the only form control elements that do not require the name attribute the form data to the server the form controls to their default settings Submit and reset buttons are straightforward to use. Just place them in the appropriate place in the form, in most cases, at the very end. By default, the submit button displays with the label "Submit" and the reset button is labelled "Reset." Change the text on the button using the value

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

➔ **Ans.**

The method attribute the method attribute specifies how the information should be sent to the server. There are only two methods for sending this encoded data to the server: POST or GET indicated using the method attribute in the form element.

We'll look at the difference between the two methods and when should each be used in the following:

Differences between GET and POST :

No.	Get Method	Post Method
(1)	It is by default.	You have to manually select this.
(2)	With the GET method, the encoded form data gets tacked right onto the URL sent to the server. A question mark character separates the URL from the following data.	When the form's method is set to POST, the browser sends a separate server request containing some special headers followed by the data.
(3)	Because the content of the form is in plain sight, GET is not appropriate for forms with private personal or financial information.	Only the server sees the content of this request, thus it is the best method for sending secure information such as credit card or other personal information.
(4)	The GET method is appropriate if you want users to be able to bookmark the results of a form submission (such as a list of search results).	The POST method doesn't provide such a thing because of being secure.
(5)	In addition, because there is a 256 character limit on what can be appended to a URL, GET may not be used for sending a lot of data or when the form is used to upload a file.	The POST method is also preferable for sending a lot of data, such as a lengthy text entry, because there is no character limit as there is for GET.

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

➔ **Ans.**

- **<Label>:** The label element is used to associate descriptive text with its respective form field. This provides important context for users with speech-based browsers.
- **Purpose :** It enhances **usability, clarity, and accessibility** by explicitly defining what information the user needs to enter. It clearly describes the purpose of an input field, reducing confusion. It helps organize form elements, creating a better user experience.

- **How `<label>` Improves Accessibility :**
  1. **Helps Screen Readers Identify Input Fields :** The screen readers announce the **`<label>`** text when users navigate the form. A properly associated **`<label>`** ensures the screen reader reads the field name**,** helping users understand what data is expected.
  2. **Expands Clickable Area for Users with Motor Disabilities:** The users with limited mobility or tremors may struggle to click on small checkboxes or radio buttons. When a **`<label>`** is linked to an input field, clicking on the label also activates the input. This reduces frustration and improves usability**,** especially on touchscreens and small screens so it supports keyboard navigation for users who cannot use a mouse.
  3. **Enhances Form Validation & Error Handling:** The Labels help users understand what to enter, reducing mistakes. The screen readers can provide better feedback when errors occur in labelled fields.

HTML TABLE

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

➔ **Ans.**

❖ **Table:** HTML tables were created for instances when you need to add tabular material [data arranged into rows and columns] to a web page. Tables may be used to organize calendars, schedules, statistics, or other types of information.

The elements that identify the table [<table>], rows [<tr> for "table row"], and cells [<th> for "table headers," and <td> for "table data"]. Cells are the heart of the table, because that's where the actual content goes. The other elements just hold things together. This is how structure of table works.

Structure:

```
<table>
        <tr>
                <th>1 header</th>
        </tr>

        <tr>
                <th>header 2</th>
                <td>data1</td>
                <td>data2</td>
        </tr>
</table>
```

❖ **Purpose of the elements:**

1. **<table>:** The **`<table>`** element in HTML is used to organize and display structured data in a tabular format consisting of rows **(`<tr>`)** and columns **(`<td>`).** Tables improve data readability, especially when combined with headers **(`<th>`)** for better organization. Additionally, they enhance accessibility when properly formatted with attributes like **`caption`.**
2. **<tr>:** The **`<tr>`** (Table Row) element in HTML is used to define a row within a **`<table>`.** It acts as a container for table data **(`<td>`)** and table header **(`<th>`)** cells, grouping them into a horizontal structure. Each **`<tr>`** represents a single row of data, and multiple **`<tr>`** elements are used to create multiple rows in a table.

3. **<td>:** The **<td>** (Table Data) element in HTML defines a single cell within a **<tr>** (table row). It is used to hold data inside a table, making up the body of the table along with other **<td>** elements. Each **<td>** corresponds to a column within a row and can contain text, images, links, or other HTML elements.
4. **<thead>:** The **<thead>** (Table Head) element in HTML is used to group the header content of a table. It typically contains one or more **<tr>** (table row) elements, which in turn hold **<th>** (table header) cells. The **<thead>** section helps organize and separate the table's header from the body **(<tbody>),** improving readability and accessibility.

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

➔ **Ans.**

Difference between colspan and rowspan

Colspan	Rowspan
Column spans or colspan is an attribute of <td> & <th>	Row span works in same elements like colspan
It is “1” by default	It is also “1” by default.
It stretches a cell to the right to span over the subsequent columns.	It causes the cell to span downward over several rows.
<table> <tr> <td colspan="3">Merged Columns</td> <tr> </table>	<table> <tr> <td rowspan="3">Merged Rows</td> <tr> </table>

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

➔ **Ans.**

The tables are created to display the data, not to create layout for webpage and they should be used sparingly for layout purposes because of following points:

1. **The issue of poor accessibility:** The tables are designed to structure tabular data**,** not for layouts. When it is used for design, screen readers misinterpret the content, making navigation difficult for visually impaired users. This creates so much confusing experience, as assistive technologies assume table elements represent structured data rather than layout components.

2. **Table being not responsive:** They do not adapt well to screen sizes which differs from devices to devices, especially on mobile devices. The table-based designs often require horizontal scrolling, making the content difficult to view on smaller screens. This leads to poor user experience and readability issues.

3. **SEO and Semantic Issues:** The search engines prioritize semantic HTML to understand web pages better. When the tables are being misused for layout, it reduces the SEO friendliness of the site, making it harder for search engines to properly index content. This can also negatively impact ranking and discoverability of a web page**.**

❖ **Better Alternative : HTML5**

HTML5 introduced new semantic elements and improved CSS layout techniques, making it a superior alternative to table-based layouts. Unlike the tables, which were originally designed for displaying tabular data, HTML5 provides a more flexible, responsive, and accessible way to structure web pages. Here are some points in detail:

1. Use of Semantic Elements: HTML5 introduces semantic tags like **`<header>`, `<nav>`, `<section>`, `<article>`, `<aside>`** and **`<footer>`.** These elements provide a clear structure, making web pages more accessible, SEO-friendly, and easier to understand for both developers and search engines.

(A) **Header**: - The header element represents a group of introductory or navigational aids. A header element is intended to usually contain the section's heading (an h1-h6 element or an hgroup element), but this is not required.

(B) **Nav**: - The nav element represents a section of a page that links to other pages or to parts within the page a section with navigation links. Not all links of a document must be in an element. The element is intended only for major block of navigation links. Browsers, such as screen readers for disabled users, can use this element to determine whether to omit the initial rendering of this content.

(C) **Section**: - The section element represents a generic section of a document application. A section in this context is a thematic grouping content typically with a heading.

(D) **Article**:- This article element represents a self- contained composition in a document, page, application, or site and that is intended to be independently distributable or reusable, e.g. In syndication

(E) **Aside**: - The aside element represents a section of a page that consists of content that is tangentially related to the content around the aside element, and which could be considered separate from that content.

(F) **Footer**:-The footer element represents a footer for its nearest ancestor sectioning content or sectioning root element. A footer typically contains information about its section such an s who wrote it. Links to related documents. Copyright data and the like.

2. Improved Responsiveness: HTML5 allow layouts to be fully responsive, adapting automatically to different screen sizes using media queries. This makes the web pages mobile-friendly without requiring any complex table structures that break on small screens. For ex. <meta name="viewport" content="width=device-width, initial-scale=1.0">

3. Better Accessibility & SEO: Semantic HTML improves accessibility for screen readers and makes content easier for search engines to index. Unlike tables, which confuse assistive technologies, HTML5 elements clearly define sections, improving navigation and usability**.**

The End