Module 2 – Frontend – HTML

HTML Basics

Theory Assignment

- **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 design the structure of web pages.

HTML is a markup language that uses a system of tags and elements to define the content and layout of a webpage, such as text, images, links, headings, lists, tables, and more.

Purpose of HTML in Web Development:

1. **Defines the Structure of Web Pages**
   HTML provides the *skeleton* or *framework* for web pages by organizing content into elements like headers, paragraphs, sections, and footers.
2. **Displays Content in Browsers**
   Web browsers read HTML code and render it visually so that users can see formatted text, images, videos, and links.
3. **Creates Hyperlinks and Navigation**
   HTML allows linking between pages using <a> (anchor) tags, enabling users to navigate websites.
4. **Works with CSS and JavaScript**

- **CSS (Cascading Style Sheets)** adds styles (colours, layouts, fonts) to HTML elements.
- **JavaScript** adds interactivity and dynamic features.
  Together, they create complete and interactive websites.

5.**Ensures Accessibility and SEO**
HTML includes semantic tags (like <article>, <nav>, <header>) that help search engines and assistive technologies understand page content better.

2. Explain the basic structure of an HTML document. Identify the mandatory tagsand their purposes.

An HTML document follows a specific structure to ensure that web browsers can correctly interpret and display the content. Below is the typical format of a basic HTML page.

Structure Overview:

```
<!DOCTYPE html>

<html>

  <head>
```

```
  <title>Page Title</title>
 </head>
 <body>
  <h1>This is a Heading</h1>
  <p>This is a paragraph. </p>
 </body>
</html>
```

Tag	Purpose
<!DOCTYPE html>	Declares the document type and version of HTML being used (HTML5). Helps the browser render the page correctly.
<html>	Root element of the HTML document. All other elements are nested inside this tag.
<head>	Contains meta-information about the page (like title, character encoding, linked CSS, etc.). Not displayed in the main content.
<title>	Sets the title of the webpage that appears in the browser tab. It is a child of <head>.
<body>	Contains all the visible content of the web page such as text, images, links, etc. Displayed in the browser window.

1. The <html>, <head>, and <body> tags must appear in every HTML document.
2. Tags are typically nested properly to follow HTML syntax.
3. HTML is not case-sensitive, but the convention is to use lowercase tags.

3. <u>What is the difference between block-level elements and inline elements in HTML? Provide examples of each.</u>

- ⇨ HTML elements are categorized based on how they behave in the document layout: Block-level elements
- ⇨ Inline elements

1. Block-Level Elements

- Take up the entire width of their container, starting on a new line.
- Stack vertically on the page.
- Can contain other block-level or inline elements.

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

This is a paragraph

<h1>This is a heading</h1>

List item

2. Inline Elements

- Take up only as much width as necessary (do not start on a new line).
- Flow within lines of text.
- Can contain only text or other inline elements.

This is a span

This is a link

Bold text

Italic text

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

Semantic HTML uses meaningful tags that describe the content they enclose. These tags clearly define the role or purpose of the content to browsers, screen readers, search engines, and developers.

1. Accessibility (A11y)

- Helps screen readers and assistive technologies understand the structure of the page.
- Allows visually impaired users to navigate using landmarks (like headers, navigation, or main content).

2. SEO (Search Engine Optimization)

- Search engines better understand the content and context of a page.
- Improves ranking and visibility by highlighting important areas like articles, headings, and navigation.

3. Maintainability & Readability

- Makes HTML easier to read and maintain for developers.
- Improves collaboration by using standard, meaningful tags instead of generic or elements.

Element	Meaning / Usage
<header>	Page or section **header** content
<nav>	Contains **navigation** links
<main>	The **main content** of the document
<article>	A **self-contained** piece of content
<section>	A **thematic grouping** of content
<aside>	**Side content** (e.g., sidebar)
<footer>	Page or section **footer** information
<figure> + <figcaption>	Image or media with a **caption**

Non-Semantic Tags (for comparison)

Tag Purpose

 Generic container

 generate inline container

HTML Forms

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

HTML forms are used to **collect user input** and send it to a server for processing. Common uses include:

- User registration
- Login pages
- Feedback or contact forms
- Online surveys
- Search bars

Forms are created using the <form> tag, which can include various **input elements** for different types of data.

Element	**Purpose**
<input>	Single-line user inputs
<textarea>	Multi-line text inputs
<select>	Dropdown menu for selections
<button>	Triggers form actions (e.g., submit)

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

Feature	**GET**	**POST**
Data Location	Appended to URL (visible)	In the request body (hidden)
Data Size	Limited	Large
Security	Less secure	More secure
Cacheable	Yes	No
Bookmarkable	Yes	No
Use Case	Fetching data/search/filter	Sending confidential/form data

➢ Use GET for actions that do not change data (like search).

➢ Use POST for actions that change or submit data (like registration or login)

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

The <label> element is used in HTML forms to define a label for an input element (like a text box, checkbox, radio button, etc.). It provides a clear, human-readable description of what the input is for.

Why <label> Is Important:

1. Improves Accessibility

- Screen readers can read the label text along with the input field, helping visually impaired users understand what the field is for.
- Users with disabilities who use voice control or keyboard navigation can interact with form fields more effectively.

2. Improves Usability

- Clicking on the <label> also activates the associated input field (e.g., checks a checkbox), making forms easier to use.
- always use a <label> for every interactive input—especially for checkboxes, radio buttons, and text fields—to create accessible and user-friendly forms.

HTML Tables

HTML tables are used to organize data in rows and columns. They are commonly used for displaying tabular data such as schedules, price lists, and reports.

Tag	Used For	Required?
<table>	Main container	✅ Yes
<tr>	Table rows	✅ Yes
<th>	Header cells	❌ No, but recommended for headings
<td>	Data cells	✅ Yes
<thead>	Grouping header rows	❌ No, but improves structure

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

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

Attribute	Spans Across	Merges Cells In	Example Use
colspan	Multiple columns	Same row	One heading for 2+ columns
rowspan	Multiple rows	Same column	One label for 2+ rows

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

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

1. Not Semantic

- Tables are meant for tabular data, not layout.
- Misusing them breaks the semantic structure of HTML.
- Hurts SEO and confuses assistive technologies**.**
- **2. Poor Accessibility**
- Screen readers interpret tables as data grids, which leads to confusing navigation for visually impaired users.
- Hard to identify which part is layout vs. data.

3. Hard to Maintain

- Layout tables require complex nested rows and columns.
- Code becomes long, repetitive, and difficult to update.

4. Not Responsive

- Tables are rigid and don't adapt well to different screen sizes (like mobile).
- Making them responsive with CSS is very difficult.

5. Slower Page Rendering

- Large tables take longer for browsers to render and reflow.
- Can negatively impact page performance and user experience.

6. Limited Flexibility

- Tables offer less design flexibility compared to CSS Grid or Flexbox.
- Difficult to control spacing, alignment, and layering.

Better Alternative: Use CSS for Layout

- Use Flexbox for one-dimensional layouts (rows or columns).
- Use CSS Grid for complex two-dimensional layouts.
- Use media queries for responsive designs.