***Wd-html (HTML Basics)***

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

HTML (Hyper Text Markup Language) is the standard language used to create and structure web pages. It provides the fundamental framework for displaying text, images, links, and other multimedia content on the internet. Essentially, HTML acts as the blueprint for how a webpage is organized and presented to the user.

Purpose of HTML in web development is to provide the backbone and structure of a web page. It defines the layout and organization of content on a web page by using a ranked structure of elements. HTML tags allow developers to specify the type of content being displayed and how it should be presented.

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

The basic structure of an HTML document consists of a specific arrangement of mandatory tags that define the content and metadata of a web page. Each component serves a distinct function essential for the correct display and interpretation of the document by browsers.

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<title> Page Title </title>**

**</head>**

**<body>**

**<!-- Visible page content goes here -->**

**</body>**

**</html>**

**<!DOCTYPE html>**

Declares the HTML version to the browser

**<html>…</html>**

Root element encompassing all HTML content

**<head>…</head>**

Contains metadata (not displayed directly)

**<title>…</title>**

Sets the page title (required inside <head>)

**<body>…</body>**

Contains all visible content displayed to the user

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

**Block-level element**

In HTML block level element starts in a new line, takes on a full width and can contain other block elements as well as inline elements.

**Examples;**

<div>

<p>

 <h1>

 <ul>

 <table>

<br>

**Inline element**

While inline element doesn’t start on new line, takes as much with as content requires and cannot contain other block level elements, only other inline element in plain text.

**Examples;**

<span>

 <a>

<img>

<strong>

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

Semantic HTML refers to the use of HTML elements that clearly convey the meaning and structure of web content such as <header>, <nav>, <main>, <article>, and <footer> unlike non-semantic tag semantic elements describe their role within the page, making it easier for both browsers and developers to understand the content’s purpose.

**Importance for Accessibility:**

* **Improved Screen Reader Navigation:** Semantic elements provide meaningful context for assistive technologies, such as screen readers, allowing users with disabilities to navigate the content efficiently. For example, a screen reader can recognize a <nav> section for menus or a <main> tag for primary content, enabling users to jump directly to those sections.
* **Landmarks and Logical Structure:** Using tags like <header>, <nav>, <main>, <article>, and <footer> creates clear “landmarks” that help users with assistive technology orient themselves and move through the page logically.
* **Keyboard Accessibility:** Interactive semantic elements, like <button> and <a>, natively support keyboard navigation, ensuring all users can access site functionality.

**Importance for SEO:**

* **Enhanced Crawling and Indexing:** Search engine crawlers interpret semantic HTML tags to understand the structure and context of content. This helps improve the ranking and discoverability of pages by associating relevant keywords with the correct parts of the site.
* **Rich Snippets and Discoverability:** Using semantic tags appropriately (e.g., <article> for news) allows search engines to extract structured data—potentially resulting in rich snippets, featured results, or improved presentation in SERPs.
* **Better User Experience Signals:** Well-structured semantic HTML typically leads to more accessible, faster-loading, and easy-to-navigate sites, which can indirectly affect SEO by improving user engagement metrics like bounce rate and time-on-page.

**Examples of Semantic Elements:**

* **<header>**: Defines the introductory content or navigation for a section or page.
* **<nav>**: Used for navigation links.
* **<main>**: Specifies the primary content of a page.
* **<article>**: Represents a self-contained composition (e.g., blog post, news item).
* **<section>**: Defines thematic grouping of content, often with a heading.
* **<aside>**: Contains content tangentially related to the main content (e.g., sidebars).
* **<footer>**: Identifies the footer for a section or page.
* **<figure>** and **<figcaption>**: For images with captions.
* **<mark>**, **<time>**, **<address>**: Other examples adding semantic meaning to text

***Wd-html (HTML Forms)***

***Question 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 on web pages, enabling users to interact with websites by submitting data that is typically sent to a server for processing (such as searching, logging in, registering, or submitting feedback).

**Input:**

input crates wide variety of interactive controls, depending on its type attributes like text, password, Email, checkbox etc. it is commonly used for gathering basic user input including single line text, numeric entries, file upload, dates etc.

**textarea:**

Provides a multi-line text input area. It is ideal for collecting longer, free-form responses from users, such as comments, messages, or addresses.

**Select:**

Displays drop-down list of options from which user can choose. Each option is defined by option element nested inside select. Option is used whenever user needs to choose for one (or sometime multiple, with multiple attribute)value from predefined set.

**Button element:**

Represents a clickable button that can perform actions; in forms, it is most often used to submit form data (with type="submit") or reset the form (type="reset"), but it can also trigger custom scripts or behaviors (type="button").

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

**GET:**

GET appends the form data as part of the URL, visible as query parameters This means data is stored in browser history, can be bookmarked. visible to anyone who has access to the URL. GET should never be used to transmit sensitive information like passwords or credit card numbers.

**Uses of GET:**

Retrieving or querying data (idempotent operations that don’t modify resources).

Data is not sensitive and can be safely exposed in the URL.

The user may want to bookmark or share the resulting page (such as search results)

**Post:**

Submitting sensitive data or data intended to update, insert, or delete server resources (such as login forms, file uploads, or transactions).

The form data is large or includes binary content.

The operation should not be cached or bookmarked for privacy or data integrity reasons.

***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 and accessible text description for a corresponding form control, such as an input, textarea, checkbox, or radio button. This connection improves both usability and accessibility in several key ways:

**Describes form controls for all users**: The label text tells users what data is expected in each field, reducing confusion and errors—especially important in longer or complex forms.

**Crucial for screen reader users**: Screen readers announce the label text when a user navigates to a form field, making it possible for users who are blind or have low vision to accurately complete forms. If labels are missing, screen readers do not provide any information about the field’s purpose, rendering the form unusable for many.

**Larger clickable area**: Associating a label with a form control (either by wrapping the control inside the label or using the label’s for attribute to reference the control’s id) allows users to click on the label to focus or toggle the control, which is especially beneficial for users with motor impairments or limited dexterity.

**Meets accessibility standards**: Proper use of label elements is required by web accessibility guidelines such as WCAG to make forms inclusive for users with disabilities.

**Improving accessibility:**

1. Ensures all users know what information to enter in each field.

2. Expands the clickable area for controls, aiding those with mobility challenges.

3. Satisfies legal and technical accessibility requirements.

***Wd-html (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>.**

An HTML table organises data in rows and columns using specific structure elements.

**<table>…</table>:**

This is the container element that defines start and the end of the table. All the other elements reside inside it and together they create complete table element.

**<tr>…</tr>:**

This element used to define single row in within table. All data or header cells contained within this element. Multiple <tr> element can be used to create rows.

**<th>…</th>:**

The <th> element defines a header cell for a column or row. typically found at the top, text written in this tag usually be bold and centered by default. Provides context for data cells beneath it making table easy to understand and more accessible.

**<td>…</td>:**

The <td> element used to defines standard data cell containing actual data items. these goes inside <tr> elements and comprise the main body of table.

**<thead>…/<thead>:**

The <thead> element groups the header content of a table, typically enclosing the row(s) of <th> elements. This grouping improves semantic clarity and accessibility by logically separating header rows from other table content, and helps browsers and assistive technologies parse the table effectively. Only header rows should be placed inside <thead>.

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

colspan and rowspan are HTML table cell attributes used to merge cells, colspan merges cells horizontally across columns, while rowspan merges cells vertically across rows.

**Colspan:**

<table border="1">

<tr>

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

<th>Age</th>

</tr>

<tr>

<td>Jill</td>

<td>Smith</td>

<td>43</td>

</tr>

<tr>

<td>Eve</td>

<td>Jackson</td>

<td>57</td>

</tr>

</table>

**rowspan:**

<table border="1">

<tr>

<th>Name</th>

<td>Jill</td>

</tr>

<tr>

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

<td>555-1234</td>

</tr>

<tr>

<td>555-8745</td>

</tr>

</table>

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

Tables should be used sparingly for layout purposes because they make your web pages less flexible, less accessible, harder to maintain, and less search engine-friendly. The preferred alternative is using CSS-based layouts (with elements like <div>, Flexbox, or CSS Grid).

1.Tables confuse screen readers and assistive technologies, making content harder to understand for visually impaired users.

2.Table layouts are rigid and do not adapt well to different screen sizes or devices, leading to poor experiences on phones and tablets.

3.Nested and complex tables make your HTML code messy and much harder to update or debug.

 4.Large tables require the browser to process more HTML, increasing page size and slowing down rendering.

5.Search engines have more difficulty parsing table layouts compared to semantic HTML, potentially lowering your site's search rankings.

6.Tables are intended for tabular data, not visual layout. Using them for layout misuses HTML semantics and can reduce code clarity.

**Alternative:**

Use CSS (Cascading Style Sheets) with structural elements like <div>, employing modern layout techniques:

* **Flexbox:** Great for one-dimensional layouts (rows or columns).
* **CSS Grid:** Powerful for two-dimensional layouts (rows and columns).
* **Media Queries:** Enable responsive designs that adapt gracefully to different screen sizes.