Module 2 – Frontend – HTML

HTML Basics

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 structure content on the web.

It is the main language used to create web pages.

It defines the structure of web pages using elements represented by **tags**, such as <html>, <head>, <body>, <p>, <a>, , and many others.

Purpose of HTML in Web Development:

1. **Structure:**
   HTML provides the basic layout and structure of a webpage, organizing content into sections, headings, paragraphs, etc.
2. **Content Display:**

   It shows text, images, videos, and more on the web page.

3. **Creates Links:**
   HTML lets you add links to other pages using the <a> tag.
4. **Semantic Meaning:**
   Semantic tags (like <header>, <footer>, <article>, <nav>) help browsers and search engines understand the meaning and purpose of content.
5. **Forms and User Input:**
   HTML supports creating forms for user interaction, such as login forms, surveys, and search bars.

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

- An HTML document follows a specific structure made up of **tags**. These tags tell the browser how to display the content.

Module 2 – Frontend – HTML

Basic HTML Document Structure:

```
<!DOCTYPE html>

<html>

        <head>

                <title>My First Web Page</title>

         </head>

 <body>

        <h1>Hello, World!</h1>

       <p>This is my first web page.</p>

 </body>

</html>
```

Mandatory Tags and Their Purposes:

1. **<!DOCTYPE html>**
    - Declares the document type and version of HTML.
2. **<html>**
    - All other tags go inside this tag.
    - The main container for the web page.
3. **<head>**
    - Holds information *about* the page (not shown to users).
    - Includes title, meta, styles, links, and scripts.
4. **<title>**
    - Sets the title of the web page (seen in the browser tab).
    - Placed inside the <head> tag.
5. **<body>**
    - Contains the actual content of the web page that is visible to users.

- Includes text, images, links, headings, etc.

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

- In HTML, block-level elements and inline elements define how content is displayed in the layout of a web page.

Block-level Elements:

- They start on a new line.
- They take up the whole row (full width).
- Think of them like big boxes.

Examples of Block-level Elements:

- This is a paragraph.
- This is a div.
- This is a div.
- <h1>-<h6>
- , <ul>, <li>, <ol>, <table>, <section>, <footer>, <nav>

Inline Elements:

- They stay in the same line.
- They take up only as much space as they need.
- Think of them like small words inside a sentence.

Examples of Inline Elements:

- This is a span element
- This is a link
- Bold text
- **,** <input>, <label>

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

Module 2 – Frontend – HTML

- **Semantic HTML** means using **HTML tags that describe the meaning of the content**.

  Instead of just using everywhere, we use special tags like:
  - <header> for the **top part**
  - <nav> for **menus**
  - <main> for **main content**
  - <footer> for the **bottom part**

Why is Semantic HTML Important?

1. **For Accessibility**
   - Screen readers (for blind users) can understand your page better.
   - Example: A screen reader knows that <nav> means **menu links.**
2. **For SEO (Search Engine Optimization)**
   - Search engines like Google read your HTML.
   - Helps your page **rank better** in search results.
3. **For Developers**
   - Makes your code **clean and easy** to read.
   - Other developers will **understand your page faster**.

Examples of Semantic Tags:

❌ *Without Semantic HTML:*

```
<div id="header">Welcome</div>
<div id="nav">Menu</div>
<div id="main">Page Content</div>
```

✅ *With Semantic HTML:*

```
<header>Welcome</header>
<nav>Menu</nav>
<main>Page Content</main>
```

Module 2 – Frontend – HTML

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 **get information from people** on a website. For example, when you sign up, log in, or send a message — you are using a form.

Key Elements in an HTML Form:

1. **< input>**
   - Purpose: Collects **single-line data** like text, numbers, passwords, dates, emails, etc.
   - Common type attributes:
     - text: for plain text input
     - password: hides input for security
     - email, number, date, etc.: for specific data types
     - checkbox, radio: for selecting options
     - submit, reset, button: for form actions

Example

```
<input type="text" placeholder="Your Name">

<input type="password" placeholder="Password">
```

2. **<textarea>**
   - Purpose: Collects multi-line text input, such as messages, comments, or descriptions.
   - Unlike <input>, it doesn't use a type attribute.

Example

```
<textarea name="message" rows="4" cols="50"></textarea>
```

Module 2 – Frontend – HTML

3. **<select>**
    - Purpose: Creates a dropdown menu to allow the user to choose from a list of predefined options.
    - Used with <option> tags inside it.

Example

```
<select name="country">
    <option value="usa">USA</option>
    <option value="canada">Canada</option>
</select>
```

4. **< button>**
    - This makes a clickable button.
    - It's used to submit the form or do some action.
    - Can specify type="submit", type="reset", or type="button".

Example

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

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

Here’s a simple explanation of the difference between GET and POST methods in form submission, and when to use each:

- **GET Method**

    - Sends data in the URL
      Example:
      www.example.com/form?name=John&age=20
    - **Data is visible** in the address bar.
    - **Data is visible** in the address bar.
    - **Data limit:** Small amount of data (URL length limit)

- Used for:
  - Simple or read-only requests
  - When security is **not important**

Example in form:

```
<form method="get" action="/search">
  <input type="text" name="query">
  <button type="submit">Search</button>
</form>
```

- **POST Method**
  - **Sends data in the body** of the request, **not shown in the URL**
  - **More secure** for sending sensitive information (like passwords)
  - **Data limit:** Can send a **large amount of data**
  - Used for:
    - Login forms
    - Registration
    - Submitting data to be stored in a database

Example in form:

```
<form method="post" action="/register">
  <input type="text" name="username">
  <input type="password" name="password">
  <button type="submit">Register</button>
</form>
```

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

- The <label> element is used to **name or describe an input field** in a form. It helps users understand what information they need to enter.

Module 2 – Frontend – HTML

🎯 Why use <label>?

- Clear Instructions:
- Connects to the Input:

♿ How Does <label> Improve Accessibility?

1. Helps Screen Readers:
   - Visually impaired users use screen readers to "listen" to a form.
   - A label **reads out loud the purpose** of the input.
2. Larger Click Area:
   - When a label is linked to a checkbox or radio button, clicking the label also selects the option.
   - This is helpful for people who have difficulty using a mouse precisely.

Example

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

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

Module 2 – Frontend – HTML

HTML Tables

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

- 📊 **HTML Table Structure**

An HTML table is used to **show data in rows and columns**, just like a table in your notebook.

Main Table Elements:

1. <table>
   - This is the **main tag** that creates the table.
   - It holds all the rows and columns.
   - Example:
     <table> ... </table>

2. <tr> = Table Row
   - It means **table row**.
   - It makes one row (line) in the table.
   - Example:
     <tr> ... </tr>

3. <th> = Table Header
   - It means **table heading** (top titles).
   - Usually used in the first row.
   - Text in <th> is **bold** and **centered**.
   - Example:
     <th>Name</th>
     <th>Age</th>

4. <td> = Table Data
   - It means **table data cell**.
   - Used inside rows to show the **real data**.

- Example:
  <td>Ravi</td>
  <td>22</td>

5. <thead> = Table Head Section

- Used to group the **header part** of the table.
- Good for styling or making tables accessible.
- Example:

```
<thead>
    <tr>
        <th>Name</th>
        <th>Age</th>
    </tr>
</thead>
```

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

Attribute	Meaning	What it does
colspan	Column Span	Merges (joins) multiple columns into one
rowspan	Row Span	Merges (joins) multiple rows into one

1. colspan Example (joins columns)

HTML Code:

```
<table border="1">
 <tr>
  <th colspan="2">Name & Age</th>
 </tr>
 <tr>
  <td>Ravi</td>
```

```
  <td>22</td>
 </tr>
</table>
```

Output:

Name & Age
Ravi

- Here, the heading "Name & Age" spans 2 columns using colspan="2".

2. rowspan Example (joins rows)

HTML Code:

```
<table border="1">
 <tr>
  <th rowspan="2">Name</th>
  <td>Ravi</td>
 </tr>
 <tr>
  <td>Asha</td>
 </tr>
</table>
```

Output:

| Name | Ravi **|**
| | Asha **|**

Here, "Name" is written once but covers 2 rows using rowspan="2".

Module 2 – Frontend – HTML

3. Tables should be used sparingly for layout purposes in modern web development for several important reasons:

➢ Tables should be used sparingly for layout purposes in modern web development for several important reasons:

❌ Why not use tables for layout?

- **Tables are for data**, not for designing how a page looks.
- **Hard to change** later — one small change can mess everything up.
- **Slower to load** sometimes.
- **Doesn't work well on phones** or small screens.

✅ Better Alternative: **CSS for Layout**

Use **CSS** to design your layout:

- **Flexbox** – Great for rows or columns.
- **Grid** – Great for making full page layouts.
- **Media Queries** – Help your site look good on all screen sizes.

🎯 Simple Rule:

- Use **tables** for data.
- Use **CSS (Flexbox or Grid)** for page layout.