HTML NOTES

🌐 How Websites Work

🧍 3 Main Roles:

1. **Client** – You (the user)
2. **Browser** – Your tool (e.g., Chrome, Firefox)
3. **Server** – The system that stores and serves the website (e.g., YouTube's server)

🔁 Process Flow:

Step	Description
1️⃣	Client types a URL (e.g., youtube.com) in the browser.
2️⃣	Browser sends a request to the **server**.
3️⃣	Server processes the request and sends back the **response** (HTML, images, etc.).
4️⃣	Browser **parses** the response and shows the webpage to the user.
5️⃣	Any further interaction (e.g., clicking/searching) sends **new requests** to the server via the browser.

🍽️ Restaurant Analogy:

Web	Restaurant
Client	Customer
Browser	Waiter
Server	Kitchen
Request	Order
Response	Prepared food (HTML, images, CSS, etc.)

- Server can sometimes respond with:

- ✅ **200 OK** → Success
- ❌ **404** → Page Not Found
- ⚠️ **500** → Server Error

🔧 **What is Backend?**

- The **backend** is how the **server processes the request** and returns the correct data or web page.
- It's responsible for **generating dynamic content** based on inputs.
- E.g., When you search something on Google, the server dynamically builds and returns the relevant page.

🔁 **Backend Technologies:**

- Common backend frameworks/languages:
  - **Node.js**
  - **Django**
  - **Flask**
  - **PHP**
  - **Ruby on Rails**
- Backend often communicates with a **database** to fetch or store data.
- Backend code is **invisible to the client**. Only the final HTML/CSS/JS is sent to the browser.

🎨 **What is Frontend?**

- **Frontend** is what the user sees and interacts with — the **visual part** of the website.

💻 **Frontend Languages:**

Language	Purpose
HTML	Structure – acts as the skeleton of a web page
CSS	Styling – controls colors, layout, fonts, etc.

Language Purpose

JavaScript Functionality – adds interactivity like forms, clicks, animations

---

🧱 **HTML is the Foundation:**

- Every website **must** be built using **HTML**.
- HTML defines **what** elements are on the page (e.g., headings, images, buttons).
- Browsers are built to understand and **parse HTML**.

🎨 **CSS Adds Design:**

- CSS styles the page: button colors, fonts, layouts, spacing, etc.
- CSS makes the page **visually appealing**.
- Example: A red curved button is created using CSS.

📄 **Basic Structure of an HTML File**

🔷 **1. Boilerplate Code**

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="UTF-8" />
  <title>Page Title</title>
 </head>
 <body>
  <!-- Content here -->
 </body>
</html>
```

- <!DOCTYPE html> → Declares the document type.
- <html> → Root tag of an HTML document.
- <head> → Contains metadata, scripts, styles, etc.
- <body> → Visible content of the web page.

🔹 **2. HTML Tags Types**

- **Pair Tags**: Have opening and closing tags.
  Example: <title>Title</title>, <body></body>
- **Self-Closing Tags**: Do not require closing.
  Example: <meta />, <img />

🎨 3. CSS and JavaScript Integration

- **CSS** changes the appearance of HTML.
- **JavaScript** adds interactivity (e.g., alert boxes).
- You can enable/disable them to observe their effects.

👄 4. Title Tag

- Defines the title of the HTML page.
- Appears in the browser tab.
- Important for **SEO** and **Search Engine Rankings**.
- Helps users and search engines understand page content.

🧠 5. Head Section (<head>)

Includes:

- <title> – Page title
- <meta> – Metadata (description, keywords)
- <link> – External CSS files
- <script> – JS files

💡 SEO Tags:

```
<meta name="description" content="This is a web dev tutorial" />
```

```
<meta name="keywords" content="HTML, CSS, JavaScript" />
```

📦 6. Body Section (<body>)

- Main visible content.

- Everything inside <body> is rendered on the screen.
- Any <script> inside the body will also execute.

🖍️ 7. HTML Attributes

- Format: key="value" inside opening tags.
- Example: <html lang="en">
- Example: <meta name="description" content="..." />

📱 8. Mobile Preview Tip (Live Server with Local IP)

Steps:

1. Run ipconfig in command prompt to get your IPv4 address.
2. Use this IP to preview your local site on mobile:

   `http://192.168.X.X:5000`
3. Ensure Wi-Fi is set to **private network**.
4. Update settings in Live Server to use the IP.

Helps in testing websites on mobile devices before deployment.

🔠 HTML Headings

- HTML provides six heading levels: <h1> to <h6>.
- **<h1>** is the largest, **<h6>** is the smallest.
- Practical usage beyond <h6> is unnecessary.
- Use headings to create **content hierarchy**:

  `<h1>HTML Introduction</h1>`

  `<h2>What is HTML?</h2>`

  `<h3>Features of HTML</h3>`

📄 HTML Paragraphs

- Use the <p> tag for paragraph content.

```
<p>This is a paragraph.</p>
```

📌 **Inline CSS**

- Used via style attribute for quick styling.

  ```
  <p style="background-color: thistle;">This is styled</p> <!-- style is an attribute -->
  ```

- Not recommended for **production-level code** due to:
  - Poor maintainability
  - Lack of reusability

📁 **Bookmark Manager Project**

🧱 **Structure**

```
<h1>My Bookmarks</h1>


<h2>Primary Bookmarks</h2>
<p><a href="https://www.google.com" target="_blank">Open Google</a></p>
<p><a href="https://www.facebook.com" target="_blank">Open Facebook</a></p>


<h2>Secondary Bookmarks</h2>
<p><a href="https://www.wikipedia.org" target="_blank">Wikipedia</a></p>
<p><a href="https://stackoverflow.com" target="_blank">StackOverflow</a></p>
```

🔗 **Anchor Tag (<a>)**

- Syntax:

  ```
  <a href="URL" target="_blank">Link Text</a>
  ```

- target="_blank" → Opens link in **new tab**.
- Link colors change:
  - Blue (unvisited)
  - Purple (visited)

- **Comment Out Styles**: Use <!-- --> or Ctrl + / to disable parts temporarily.

HTML Images (<img> tag):

- Use: Display images in a webpage.
- Syntax:

  ```
  <img src="image.png" alt="Train Image" width="230" height="230">
  ```

- Key Attributes:
  - src: Path to the image.
  - alt: Alternate text if image fails to load (helps with SEO + accessibility).
  - width & height: In pixels (auto-adjusts if only one is given).
  - Can also use style attribute for inline CSS.

HTML Tables (<table> tag):

- Used to display data in tabular form.
- Key Tags:
  - <tr>: Table Row
  - <th>: Table Header
  - <td>: Table Data
  - <caption>: Title of the table
  - <thead>, <tbody>, <tfoot>: Logical grouping of table parts
- **Rowspan & Colspan:**
  - rowspan="2": Merges 2 rows vertically
  - colspan="2": Merges 2 columns horizontally

Example:

```
<table>
    <caption>Employee Details</caption>
    <thead>
        <tr>
```

```
            <th>Name</th>
            <th>Role</th>
            <th>Language</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>Harry</td>
            <td>Programmer</td>
            <td>JavaScript</td>
        </tr>
        <tr>
            <td colspan="2">Sam</td>
            <td rowspan="2">Java</td>
        </tr>
        <tr>
            <td>Mike</td>
            <td>Designer</td>
        </tr>
    </tbody>
    <tfoot>
        <tr>
            <td colspan="3">Footer</td>
        </tr>
    </tfoot>
</table>
```

✅ **HTML Lists in Detail**

There are **3 types of lists** in HTML:

1. 🔹 **Unordered List (<ul>)**
   - Used when order **doesn't matter**.
   - List items marked with bullets (● by default).
   - Structure:

     ```
     <ul>
      <li>Harry</li>
      <li>Rohan</li>
      <li>Shubham</li>
     </ul>
     ```

   - Customize bullet **type**:
     - type="disc" (default)
     - type="circle"
     - type="square"

---

2. 🔸 **Ordered List (<ol>)**
   - Used when order **matters**.
   - Items are numbered by default (1, 2, 3, ...).
   - Structure:

     ```
     <ol type="A">
      <li>Harry</li>
      <li>Rohan</li>
      <li>Shubham</li>
     </ol>
     ```

   - Customize number format using type:
     - type="1" → 1, 2, 3,... (default)
     - type="A" → A, B, C,...
     - type="a" → a, b, c,...

- type="I" → I, II, III,...
- type="i" → i, ii, iii,...

---

3. 📘 Definition List (<dl>)

- Used to define terms (less common).
- Structure:

```
<dl>
 <dt>HTML</dt>
 <dd>Hyper Text Markup Language</dd>
 <dt>CSS</dt>
 <dd>Cascading Style Sheets</dd>
</dl>
```

- Tags used:
  - <dl>: Definition list
  - <dt>: Term
  - <dd>: Description

🔍 What is SEO?

- **SEO (Search Engine Optimization)** helps improve the visibility of a website on search engines like Google.
- Example: **Amazon.in** optimizes its site to load fast and appear higher in search results.
- Google shows:
  - Fast-loading websites
  - Websites with good content
  - Sites with good user experience (UX)

🎯 Goal of SEO:

- Rank higher on search engines by making your site:

- Fast
- Informative
- User-friendly

⚙️ **Core Web Vitals (Important for SEO Ranking)**

1. CLS (Cumulative Layout Shift)

- Measures **unexpected shifts** in layout during page load.
- Bad CLS = Elements move around while user is interacting.
- **To avoid CLS:**
  - Always set width and height for images.
  - Prevent unexpected layout shifts.

2. LCP (Largest Contentful Paint)

- Measures time taken to load the **largest visible element** (e.g., big image or heading).
- Should be **≤ 2.5 seconds** for good UX.

3. FID (First Input Delay)

- Measures the time between **user interaction (like a button click)** and browser response.
- Should be **< 100ms**.

📋 **Lighthouse Report – How to Check SEO Performance**

📌 **Steps to generate Lighthouse Report:**

1. Right-click → Inspect → Go to **Lighthouse** tab.
2. Choose **Mobile/Desktop** view.
3. Click **Analyze Page Load**.
4. Report shows:
   - **Performance**
   - **Accessibility**
   - **Best Practices**

    - **SEO**
    - **PWA (Progressive Web Apps)**

🛠 **What Lighthouse Tells You:**

- Suggestions like:
    - “Size your images properly”
    - “Add width/height attributes”
- Helps in improving:
    - Load speed
    - User experience
    - SEO

🏷 **Meta Tags in SEO**

✅ **Title Tag:**

- Appears on browser tab & search engine results.
- Should **reflect the page’s content**.
- Example:

  `<title>About Harry - CodeWithHarry</title>`

✅ **Meta Description:**

- Gives a summary of the page.
- Example:

  `<meta name="description" content="This page contains description about Harry.">`
- Search engines **may** use it in results.

⚠️ **Meta Keywords:**

- **Deprecated** and **ignored** by modern search engines.
- Example:

  `<meta name="keywords" content="web, development, HTML">`
- No longer helps in SEO due to keyword stuffing abuse.

📱 Responsive View Testing in Browser

- Use **Inspect → Toggle Device Toolbar**.
- Choose device (iPhone, iPad, etc.) or responsive layout.
- Helps check how the site looks on different screen sizes.

🧠 Advanced Web Vitals (from Lighthouse)

Other metrics:

- **FCP (First Contentful Paint)**: When first text/image is painted.
- **Speed Index**: How quickly content is visibly populated.
- **Total Blocking Time**: Time a page is blocked from responding.

📚 Learn more at https://web.dev

🔶 What is a Form in HTML?

- Forms are used to collect user input and send it to a server.
- Examples: Exam forms (IIT-JEE, NEET), reservation forms, registration forms, etc.

🏷️ HTML <form> Tag

- Basic structure:

```
<form action="submit.php" method="post">
  <!-- form elements go here -->
</form>
```

- **action**: URL where the form data is sent.
- **method**: Either GET or POST
  - **GET**: Sends data via URL; for small, non-sensitive data.
  - **POST**: Sends large or sensitive data in the request body (e.g. passwords).

🔹 Input Elements (<input>)

- Most common types:
    - type="text"
    - type="number"
    - type="radio"
    - type="checkbox"
    - type="date"
    - type="email"

🔸 **Example:**

```
<label for="username">Enter your username:</label>
<input type="text" id="username" name="username" placeholder="Enter your username" required autofocus>
```

- placeholder: Placeholder text inside the input field.
- required: Field must be filled before form submission.
- autofocus: Auto-focuses this field when page loads.
- name: Used as key in key-value pair sent to server.

🧩 **Label Tag (<label>)**

- Associates text with a form element.
- Uses for attribute, which matches the id of the input element.

⭕ **Radio Buttons**

- Allows one selection from a group.
- Must share the same name to be grouped.

```
<input type="radio" id="male" name="gender" value="male">
<label for="male">Male</label>
<input type="radio" id="female" name="gender" value="female">
<label for="female">Female</label>
```

✅ Checkbox

- Allows multiple selections independently.

```
<input type="checkbox" id="subscribe" name="subscribe" value="yes">
<label for="subscribe">Subscribe to newsletter</label>
```

📝 Text-area Tag

- Multi-line input field.

```
<label for="comment">Enter your comment:</label>
<textarea id="comment" name="comment" rows="4" cols="50" placeholder="Enter your comment"></textarea>
```

🔽 Select Tag

- Dropdown with multiple options:

```
<label for="fruits">Choose a fruit:</label>
<select id="fruits" name="fruits">
  <option value="apple">Apple</option>
  <option value="banana">Banana</option>
  <option value="cherry">Cherry</option>
</select>
```

❌ Avoid Using
 for Layout

- Prefer CSS or for layout structure.
-
 is for line breaks, not spacing layout.

✅ Block Elements

- Take up the **entire width** of their container.
- Always start on a **new line**.

- Examples: , <p>, <h1> to <h6, <ul>, <ol>, <li>, <form>, <section>, <article>
- Visually, block elements **push content below** them.

✅ **Inline Elements**

- Only take **as much width as required** by their content.
- Do **not start on a new line**.
- Allow other inline elements to sit **beside them**.
- Examples: , <a>, , <img>, <label>, <input>

HTML Spacing Notes:

- **Extra spaces** or **new lines** in HTML are **ignored**.
- To add a space: use   (non-breaking space)

  ```
  Hello   World
  ```

✅ **Best Practices:**

- Use for layout blocks.
- Use for small inline pieces of content.
- Avoid unnecessary
 tags for layout (use CSS for spacing).

🆔 **ID and Classes in HTML**

🔹 **What is an ID?**

- A unique identifier for an HTML element.
- It cannot be same for more than one element.
- We can access element through id in CSS using **#id_name**
- Like your **Aadhar number** – **only one per person**, can't be shared.

  html

  ```
  <div id="first">I am uniquely identified</div>
  ```

css

```
#first {
  background-color: red;
}
```

- **What is a Class?**
  - It can be given to more than one element.
  - You can give **multiple classes** to one element by separating them with spaces.
  - We can access element through class in CSS using **.classname**
  - Used to apply **common styles** to multiple elements.

html

```
<div class="red bg-yellow">I share styling with others</div>
```

css

```
.red {
  color: red;
}
.bg-yellow {
  background-color: yellow;
}
```

Special Use of ID: Linking to Page Sections

- IDs can be used for **in-page navigation**.
- Example: Clicking a link-jumps to a section with a specific ID.

html

```
<a href="#about">Go to About</a>
<section id="about">
  <h2>About Us</h2>
</section>
```

- Works even in external links:

https://example.com/about.html#team

---

1. HTML <video> Tag: Used to embed a video on a web page.

Basic Syntax:

```
<video src="video.mp4" width="555" height="255" controls></video>
```

Important Attributes:

Attribute	Description
src	Path to the video file (can be local or external URL)
controls	Adds play, pause, volume, fullscreen, etc.
autoplay	Plays video automatically when the page loads
loop	Repeats the video indefinitely
muted	Mutes the video by default
poster	Specifies an image to be shown before the video plays
width/height	Defines the dimensions of the video

- **2. HTML <audio> Tag:** Used to embed audio files.

Basic Syntax:

```
<audio src="sachin.mp3" controls></audio>
```

Important Attributes:

Attribute	Description
src	Path to the audio file
controls	Adds play/pause and volume control
autoplay	Audio plays automatically on load
loop	Repeats audio indefinitely
muted	Mute audio by default
preload	Specifies how the audio should be loaded when the page loads

Preload Options:

Value	Description
none	Browser does not preload the audio
metadata	Only metadata (length, duration and dimensions) is preloaded
auto	Entire audio file is preloaded

🔹 3. SVG (Scalable Vector Graphics)

- It's an **XML-based** image format used to define **vector-based graphics** that can scale (resize) **without losing quality**.
- Can be embedded like an image or written directly in HTML using <svg>.
- If SVG is not displaying, ensure to include the XML namespace declaration:

```
<svg xmlns="http://www.w3.org/2000/svg">
<!-- SVG Content -->
</svg>
```

🔹 4. HTML <iframe> Tag

Used to embed external content (like another website or a YouTube video).

Basic Syntax:

```
<iframe src="https://example.com" width="300" height="200"></iframe>
```

Embedding YouTube Video:

- Use **YouTube's Share → Embed** option.
- It provides an <iframe> code like:

```
<iframe width="560" height="315"
src="https://www.youtube.com/embed/VIDEO_ID?start=SECONDS"
frameborder="0" allowfullscreen></iframe>
```

✅ What are Semantic Tags in HTML?

Semantic tags are HTML5 elements that clearly describe their **meaning and purpose** to the browser, developers, and search engines.

They don't just define **how** content looks—they define **what** the content is.

✅ Why Use Semantic Tags?

- Help **browsers, screen readers**, and **search engines (like Google)** understand the **structure** of your webpage.
- It is very important for ranking any website.
- Improve **SEO** (Search Engine Optimization) and accessibility.
- Make your HTML **cleaner, more organized, and readable**.
- Help **other developers** or tools to understand your content better.

📌 **Even if you don’t use semantic tags, the site will work—but it won’t be as understandable or SEO-friendly.**

✅ Examples of Semantic Tags:

Tag	Meaning / Use
<header>	Defines the top part of the page (like logo, nav bar)
<nav>	Contains navigation links
<main>	Main content of the page (only one per page)
<section>	Groups related content (e.g., different page sections)
<article>	Independent content (e.g., blog post, article)
<aside>	Sidebars or content related to the main content
<footer>	Bottom part (copyright, contact, links)
<figure>	Self-contained media like images or charts
<figcaption>	Caption for the <figure>
<time>	Represents date/time for events

✅ **Non-Semantic vs Semantic (Example):**

```
<!-- Non-semantic (not descriptive) -->
<div id="header">
 <div id="nav">Home | About | Contact</div>
</div>

<!-- Semantic (descriptive) -->
<header>
 <nav>
  <ul>
   <li>Home</li>
   <li>About</li>
   <li>Contact</li>
  </ul>
 </nav>
</header>
```

✅ **Important Clarification:**

- Semantic tags **do not style your content**.
- They're only for **structure and meaning**.
- You still use **CSS** to control how things look.

✅ **Real-Life Analogy (from the video):**

Using semantic tags is like **organizing files in folders**. If everything is in one folder, it's a mess. But if you use separate folders for **images**, **videos**, **documents**, it's easier to understand and find things—just like semantic HTML helps browsers understand your page better.

🧩 1. HTML Entities

Entities are used in HTML to display reserved characters or symbols that would otherwise be interpreted as code.

Common HTML Entities:

Symbol	Entity Code	Meaning
<	<	Less than
>	>	Greater than
&	&	Ampersand
"	"	Double quotation
'	'	Single quotation
©	©	Copyright
		Non-breaking space

🔤 2. Why Use Entities?

- If you write something like:

`<p>3 < 4</p>`

The browser will misinterpret < as a tag. To fix this, use:

3 < 4

📜 3. <pre> Tag

- Preserves **formatting**, **spacing**, and **new lines**.

🗣️ 4. Quotation Tags

- **<blockquote>** – For long quotations (block level).

  `<blockquote cite="https://en.wikipedia.org">Quote</blockquote>`

  Output:

  Quote

- **<q>** – For short inline quotations.

  <q> some text </q>

  Output:

  “some text”

- **<cite>** – Text in the <cite> usually rendered in Italic.

```
<p><cite>The Scream</cite> by Edvard Munch. Painted in 1893.</p>
```

  Output:

  The Scream by Edvard Munch. Painted in 1893.

---

5. <abbr>: tag defines an abbreviation or an acronym

The <abbr title="World Health Organization">WHO</abbr> was founded in 1948.

Output:

The WHO was founded in 1948.

---

6. <address>: tag defines the used for address.

<address>

Written by John Doe.

Visit us at:

Example.com

Box 564, Disneyland

USA

</address>

Output:

Written by John Doe.
Visit us at:
Example.com
Box 564, Disneyland
USA

---

7. <bdo>: used to override the current text direction

```
<bdo dir="rtl">This line </bdo>
```

Output:

enil sihT.

🚫 7. Obsolete Tags (Avoid Using)

- <font> – Use CSS for font styling instead.
- – Use CSS Flexbox/Grid for centering.

---

8. <code> and <pre> for Code Blocks

- Wrap code snippets using:

  ```
  <pre><code>let a = 10;</code></pre>
  ```

- Keeps code formatting and indentation.
- It is used for writting any code.
- We can also write code without it by using only pre tag. But It is recommended to use it.