Lab 13 - HTML, CSS and Javascript

Learning Objectives

● Understand the purpose and structure of HTML, CSS, and JavaScript in web development.
● Explore HTML tags and elements used to build webpage content.
● Apply CSS to style HTML elements and improve webpage layout and presentation. ● Identify how JavaScript adds interactivity and dynamic behavior to web pages. ● Use basic JavaScript functions to display alerts and console messages.

Outcomes

● Demonstrate the ability to create a structured and styled webpage using HTML and CSS.
● Write and execute simple JavaScript functions for basic interactivity.
● Use window.alert() and console.log() statements to display messages. ● Understand the connection between programming concepts (like functions) and JavaScript behavior.
● Confidently test and debug simple scripts using the browser console.

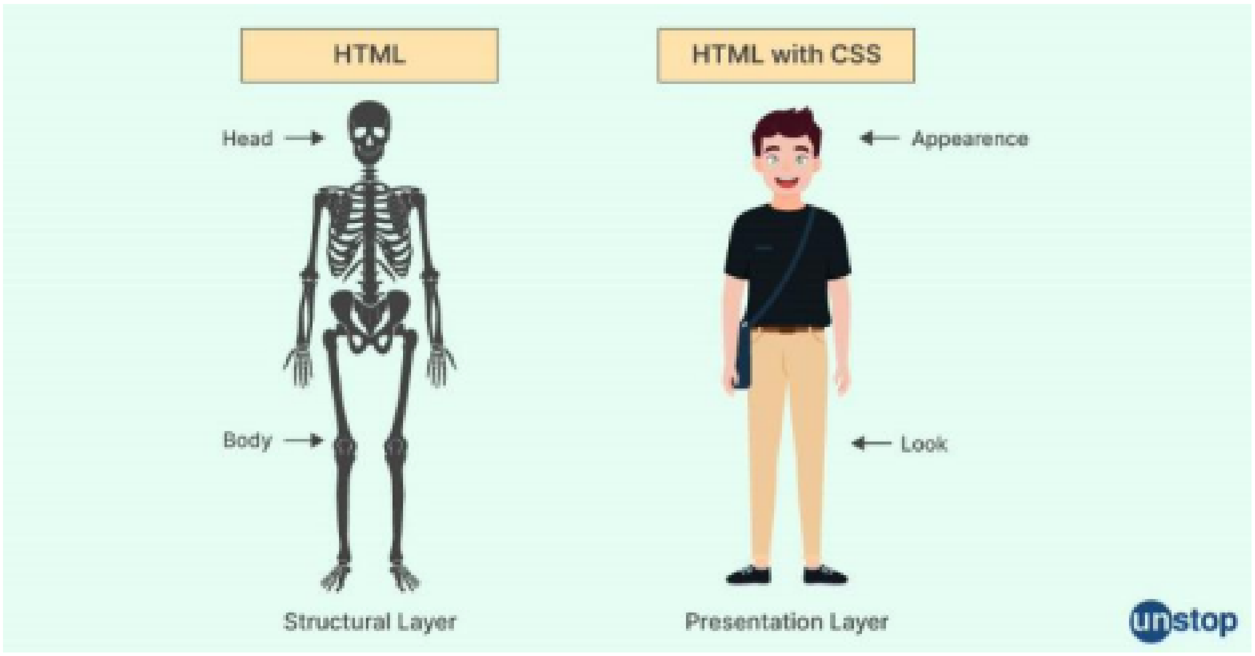
1.1 HTML

HTML, or HyperText Markup Language, is the standard language used to create the structure and content of web pages. It defines elements such as headings, paragraphs, links, images, tables, and forms using tags enclosed in angle brackets. Browsers read these tags to display the webpage's content in an organized way. In essence, HTML acts as the foundation or skeleton of a website, giving meaning and structure to all the text, media, and interactive elements on a page.

1.2 CSS

CSS, or Cascading Style Sheets, is a stylesheet language used to control the visual presentation of HTML content. It allows developers to define how elements should look, including colors, fonts, spacing, alignment, and layout. By separating design from structure, CSS makes it easy to maintain and update the appearance of a website without altering its

erlying HTML. It can be applied inline, within the same HTML file, or externally through a
ed stylesheet, ensuring consistent and visually appealing web pages across the entire site.



An analogy of html vs css

Webpage Without CSS
THIS IS MY BANNER TEXT
• HOME
• INFO
• CONTACT
• ABOUT US
This Is My Paragraph Header
Webpage With CSS
THIS IS MY BANNER TEXT
This Is My Paragraph Header

An image of webpage without css vs with css

1.3 Javscript

JavaScript is a scripting language used to make web pages interactive. While HTML provides the structure and CSS defines the style, JavaScript adds behavior. It enables web pages to respond when users click buttons, enter text, or move their mouse. You have already studied functions in Programming Fundamentals (which are reusable blocks of code that perform specific tasks). JavaScript functions work in the same way. They are written once and can be called anytime to perform an action that enhances the behaviour of different elements in your website.
DOM (Document Object Model) in **JavaScript** represents the HTML page in a treelike structure, providing access to elements and allowing dynamic changes.
We put scripts at the bottom so the page loads faster and JavaScript doesn't run before HTML is ready.
we write <script> at the bottom (before </body>) instead of inside <head>:
Because it makes the **webpage load faster and work correctly**.
✔ 1. Page loads first, JavaScript loads later
If you put <script> in the <head>, the browser **pauses the page loading** to download and run the JavaScript file first.
This makes the page feel **slow**.
When scripts are at the bottom, the browser loads **all HTML first**, shows the page to the user, **then runs the JavaScript**.
✔ 2. HTML elements are available when JavaScript runs

1.4 W3Schools

W3Schools is a popular online platform that provides free tutorials, examples, and interactive exercises for learning web technologies such as HTML, CSS, JavaScript, and more. It's especially useful for beginners because it offers clear explanations and a built-in "Try it Yourself" editor, allowing students to write and test code instantly in the browser. The site is structured in a step-by-step manner, making it easy to follow and practice concepts progressively. You can access W3Schools and start learning at their official website here: https://www.w3schools.com.

Task 1: Creating a Table and Adding Records in HTML (10)

Step 1: Create a New HTML File
Open any text editor such as Notepad, VS Code, or Sublime Text. Create a new file and save it as student_table.html in your desired location (e.g., Documents or Desktop).This file will be used to create and display a table containing student records.

Step 2: Write Basic HTML Structure
Start by writing the basic structure of an HTML page.
Include the main HTML tags such as `<!DOCTYPE html>`, `<html>`, `<head>`, `<title>`, and `<body>`.Set the title of the webpage as Student Records.

Step 3: Create a Table for Student Records
Inside the `<body>` section, create a table that displays information about students. Your table should include the following columns:

Column Name	**Description**
Student ID	Unique ID for each student
Name	Student's full name
Age	Student's age
Gender	Male/Female
Marks	Total marks obtained
City	Student's city

Add a heading above the table such as "Student Information Table."Then insert at least five (5) sample student records with realistic data for all columns.

Step 4: Save and Open in Browser
After completing the table, save your HTML file.Open it in a web browser (e.g., Chrome or Edge) to verify that the table appears correctly and all data is displayed neatly.

Task 2: Applying Inline CSS to the Table (10)

Step 1: Open Your Existing HTML File
Open the file named student_table.html that you created in the previous task.You will now enhance the appearance of your existing student table using inline CSS styling.

Step 2: Apply Inline CSS to the `<table>` Tag
Add inline CSS styles directly within the `<table>` tag to control the table's overall appearance.For example, you can define the table's width, border, and alignment so it appears centered and neatly spaced on the page.

Step 3: Style the Table Headers (`<th>`)
Add inline styles to your table headers to make them stand out.You can change the background color, text color, font size, and padding for the header cells.Choose colors that make your table easy to read and visually appealing.

Step 4: Style the Table Data Cells (`<td>`)
Apply inline CSS to your data cells to adjust text alignment, font style, and padding.For example, you may align text to the center and give each cell some space for readability.

Step 5: Save and Preview in Browser
After applying inline CSS to your table, save your HTML file.Open it in a web browser to view the changes.Check whether the styling looks consistent and that all data remains visible and properly aligned.

Tip: You can refer to the W3Schools Inline CSS Tutorial for examples of how to apply inline styles: https://www.w3schools.com/html/html_css.asp

Task 3: Creating Interactive Buttons Using JavaScript (10)

Step 1: Open your existing student_table.html file (or create a new file called interactive_buttons.html).

Step 2: Below your table or in a new section, add two buttons inside the <body> tag using the following code:

<button onclick="showStudentInfo()">Show Student Info</button>

<button onclick="showLabInfo()">Show Lab Info</button>

Step 3: Add a <script> section before the closing </body> tag and define the functions:

```
function showStudentInfo() {

console.log("Displaying student information...");

window.alert("This webpage displays student records in an HTML table.");  }

function showLabInfo() {

console.log("Displaying lab information...");

window.alert("Lab 11 covers HTML, CSS, and a basic introduction to JavaScript.");  }

</script>
```

For more: https://www.w3schools.com/js/js_htmldom_eventlistener.asp

Step 4: Save your file and open it in a browser. Click each button to see both the alert pop-up and the console message.

Step 5: Add a custom alert pop-up button and display a custom alert on your webpage that says "hi! [name]". Replace [name] with your name.

Step 6: Save your file and open it in a browser. Click each button to see the alert pop-ups and the console message.

Task 4: Push the Code to GitHub (10)

1. Create a Repository on GitHub

- Go to GitHub.
- Click the **"+"** → "New repository".
- Give it a name, add a description (optional), choose public/private, and click Create repository.
  (Copy the repository URL — HTTPS or SSH)

2. Initialize Git and Push

- git init
- git branch -M main
- git add .
- git commit -m "first commit"
- git remote add origin https://github.com/username/repoName
- git push -u origin main

3. Verify
Go to your GitHub repository page : your code should now appear there.

Submission Guidelines:

- Submit your HTML and CSS files with your **name, roll number and section** written in **comments** at the beginning of your code files.
- Name the submitted files as follows: SectionLetter_RollNumber_Task Number_LabNumber, for example: **E_i251234_Task1_Lab7**.
- The unethical usage of Generative AI is strictly prohibited and will result in 0 marks.
- In case a demo is conducted, there will be deductions based on your answers to the questions asked.