Test Guidelines

1. Explore and Use New HTML Elements:

- Explore and apply new HTML elements related to the given topics, even if they are not explicitly listed in the tasks.
- Challenge yourself to broaden your knowledge by integrating unfamiliar elements to enhance your code and understanding.

2. Learn Before You Code

- Do not rely on ChatGPT or any other AI tools to directly solve the tasks for you. Instead, use this opportunity to learn the concepts you're unfamiliar with by referring to documentation, tutorials, or other resources.
- Ensure you have a thorough understanding of the code you write. You should know exactly what each line of code does and why it is necessary for the task.

3. CSS Styling

- For each task, use a combination of internal, external, and inline CSS to apply styles. This will help you get comfortable with the different methods of CSS styling and understand their applications.

4. Separate Files for Each Task

- Create a separate file for each task:
- Each task should have its own HTML file (e.g., task1.html, task2.html).
- For tasks that require external styles, create a corresponding CSS file (e.g., styles.css) and link it appropriately to the HTML file.

5. Viva/Interview for Knowledge Check

- After completing the tasks, there will be a **viva/interview** to assess your understanding of any task you've completed.
- Be prepared to explain your thought process, the code you've written, and how the HTML and CSS concepts are applied in each task.

Task 1: Basic HTML Elements and Attributes

Objective: Create a simple webpage using basic HTML elements and attributes.

1. Create a webpage (index.html) with the following:
    - Add a heading (<h1>), a paragraph (<p>), and an image (<img>).
    - The image should have an alt attribute describing the image.
    - Add an unordered list (<ul>) and an ordered list (<ol>) with at least 3 items each.
    - Add a link to another website with the href attribute.

Topics Required:

- Basic HTML elements (headings, paragraphs, lists, images)
- Attributes (like alt, href)

Task 2: HTML Forms and Input Types

Objective: Create a registration form using different form elements and input types.

1. Design a registration form with the following fields:
- Name (text input)
- Email (email input)
- Date of Birth (date input)
- Password (password input)
- Gender (radio buttons)
- Hobbies (checkboxes)
- Add a Submit button

2. Ensure appropriate use of form attributes such as action, method, and required.

Topics Required:

- HTML forms and attributes
- Input types (text, email, date, password, radio, checkbox)
- Form attributes (action, method, required)

Task 3: Paragraph Styling, Headings, and Text Formatting

Objective: Style text and headings with different formats and styles.

1. Create a webpage with the following:
    - Add at least three paragraphs with different text.
    - Apply text formatting (bold, italic) using HTML tags.
    - Style each paragraph with a different font, font size, and color using CSS.
    - Add headings (<h1> to <h3>) with custom font sizes and colors.

Topics Required:

- Paragraph styling
- Headings
- Text formatting (bold, italic)
- CSS selectors and styling

Task 4: Page Title and Meta Tags

Objective: Add a title and meta tags for SEO and accessibility.

1. Create a webpage with a meaningful page title (<title>).
2. Add the following meta tags:
    - Description
    - Keywords
    - Author

Topics Required:

- Page title
- Meta tags

Task 5: HTML Tables

Objective: Create a structured HTML table for tabular data.

1. Create a table that displays the following information:
    - Student Name, Age, Grade
    - Use the <table>, <thead>, <tbody>, and <tfoot> elements.
    - Apply colspan and rowspan where appropriate.
    - Add borders and padding to the table cells using CSS.

Topics Required:

- HTML tables (<table>, <thead>, <tbody>, <tfoot>)
- Table attributes (colspan, rowspan)
- CSS borders, padding

Task 6: Block and Inline Elements, Divs, Lists

Objective: Create a web page layout using block and inline elements.

1. Create a section using to group content.
2. Add block elements such as <p>, <h2>, and inline elements such as .

3. Create an unordered list (<ul>) and style it to appear horizontally (as a navigation menu).

Topics Required:

- Block elements (div, p, h2)
- Inline elements (span)
- Lists

Task 7: Iframes

Objective: Embed an external webpage using an iframe.

1. Create a web page and embed a YouTube video using an <iframe>.
2. Set the width and height of the iframe to be responsive.

Topics Required:

- Iframes
- Attributes (width, height)

Task 8: Page Layout with Semantic Elements

Objective: Build a basic webpage layout using semantic HTML5 elements.

1. Structure a webpage with the following semantic elements:
    - <header>, <nav>, <main>, <article>, <footer>
2. Ensure proper content organization for readability**.**

Topics Required:

- Semantic elements (header, nav, main, footer, article)

Task 9: CSS Selectors and Styling (Internal, External, Inline)

Objective: Apply styling using different methods.

1. Create an HTML file with:
    - Internal CSS (inside <style> in the head).
    - External CSS (link to an external stylesheet).
    - Inline styles (using the style attribute directly on an element).
2. Apply styles such as text color, font size, and background color.

Topics Required:

- CSS selectors
- Internal, external, and inline CSS

Task 10: Colors, Borders, and Backgrounds

Objective: Apply colors, borders, and background styling to elements.

1. Create a webpage where:
    - The <header> has a background color.
    - The has a border of 2px solid black.
    - Apply a background image to a section and adjust its size and position using CSS.

Topics Required:

- Colors
- Borders
- Backgrounds

Task 11: Margins, Padding, Height, and Width

Objective: Adjust element spacing and dimensions using CSS.

1. Style a webpage where:
    - A has a margin of 20px and padding of 15px.
    - The height and width of a section are explicitly defined (e.g., 400px by 600px).

Topics Required:

- Margins
- Padding
- Height and width

Task 12: CSS Flexbox

Objective: Create a simple, responsive webpage layout using Flexbox that adjusts based on the screen size (mobile, tablet, and desktop).

1. Create a webpage with the following sections:

    · A header section with a logo (use a placeholder text) and navigation links.

- A main content section with three columns (e.g., "Tab 1", "Tab 2", and "Tab 3").
- A footer with some copyright text.

2. Use Flexbox to arrange the header items (logo and navigation) horizontally.
3. Use Flexbox to create a three-column layout for the main content section.
4. Make the layout responsive:

- On **desktop** (screen width >= 1024px): The three columns should be displayed side by side (in a row).
- On **tablet** (screen width between 768px and 1024px): The columns should stack two on top, and one below (2-column grid, followed by the last column below).
- On **mobile** (screen width < 768px): The columns should stack one on top of the other (single column).

Topics Required:

- CSS Flexbox
- Media Query

Task 13: CSS Grid Layout

Objective: Build a grid layout for a webpage section.

1. Create a grid with two columns and three rows.
2. Place different content inside each grid cell.

Topics Required:

- CSS Grid

Required Topics for All Tasks:

- Basic HTML elements (headings, paragraphs, lists, images)
- HTML attributes (alt, href, etc.)
- HTML forms and input types (text, email, date, password, etc.)
- Form attributes (action, method, etc.)
- Paragraph styling, headings, and text formatting (bold, italic)
- Page title and meta tags
- HTML tables (headers, rows, cells, colspan, rowspan)
- Block and inline elements, , and lists
- Iframes
- Layouts and semantic elements (header, nav, main, footer)

- CSS selectors (internal, external, inline)
- Colors, borders, backgrounds
- Margins, padding, height, and width
- Flexbox and Grid (Optional, but important for full-stack development)