# NAME : G.SAI AMRUTHA

# ENROLLMENT NO : 2403A52017

# BATCH NO: 02

# SUBJECT: AI ASSISTANT CODING

### Assignment-14.3

Task Description #1 – AI-generated HTML Page  
Task: Ask AI to generate a simple HTML homepage for a "Student Info  
Portal" with a header, navigation menu, and footer.

### Prompt

Generate a simple HTML homepage for a **"Student Info Portal"**.  
The page should include a clear **header**, a **navigation menu** (e.g., Home, About, Contact), a **main content section**, and a **footer**.  
Make the structure clean, readable, and well-formatted with proper HTML5 tags and indentation.  
Also, explain how each section contributes to the overall webpage layout.

### Code and output

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

### Observations

* **Semantic HTML5:** The code effectively uses semantic HTML5 tags like <header>, <nav>, <main>, and <footer>. This makes the structure of the page clear and helps search engines and assistive technologies understand the content better.
* **Basic Structure:** The code establishes a clear and common webpage layout with distinct sections for the header, navigation, main content, and footer. This is a good foundation for most websites.
* **Internal CSS:** The CSS is embedded directly within the <style> tags in the <head>. While this is acceptable for small examples, for larger projects, it's generally better to link to an external CSS file for better organization and reusability.
* **Basic Styling:** The CSS provides basic styling for layout and appearance, including setting font family, handling margins and padding, and styling the header and navigation.
* **Responsive Design Meta Tag:** The <meta name="viewport"...> tag is included, which is essential for making the webpage responsive and ensuring it displays correctly on different devices.
* **Clear Navigation:** The navigation menu uses a simple list of links within the <nav> tag, which is a standard approach.

Task Description #2 – CSS Styling  
Task:  
Use AI to add CSS styling to Task #1 homepage for:  
• Responsive navigation bar.  
• Centered content section.  
• Footer with light gray background.

### Prompt

Add CSS styling to the "Student Info Portal" homepage created in Task #1. The page should include:

* A **responsive navigation bar** that adjusts for different screen sizes.
* A **centered main content section** for better visual balance.
* A **footer with a light gray background** and proper padding.  
  Make the **CSS clean, readable, and either embedded within the HTML or placed in a <style> block**.  
  Also, **explain how each CSS rule improves the webpage’s appearance and responsiveness**.

### Code and output

A screenshot of a computer

AI-generated content may be incorrect.

### Observation

* **Responsive Design Implementation:** The use of the viewport meta tag and the @media query for the navigation bar effectively implements a basic responsive design. The navigation shifts from horizontal to vertical on smaller screens, which is a common pattern for mobile-friendly websites.
* **Flexbox for Layout:** Using display: flex, flex-direction: column, and min-height: 100vh on the body and flex: 1 on the main content is a modern and effective way to create a sticky footer layout where the footer stays at the bottom of the viewport even if the main content is short.
* **Centering Content:** The max-width and margin: 20px auto; on the main content effectively centers the content block on wider screens and provides some padding around it.
* **Clear Visual Hierarchy:** The header and footer have distinct background colors and padding, which helps create a clear visual hierarchy and separates these sections from the main content.
* **Navigation Styling:** The styling for the navigation links provides good visual feedback on hover and ensures they are easily clickable blocks.
* **Shadow for Depth:** The subtle box-shadow on the main content adds a touch of visual depth and helps it stand out from the background.

The combination of semantic HTML and the applied CSS creates a visually appealing and functional basic homepage that adapts reasonably well to different screen sizes.

Task Description #3 – JavaScript Interactivity  
Task: Prompt AI to generate a JS script that validates a simple login  
form (non-empty username/password).

### Prompt

Generate a **JavaScript script** that adds interactivity to a simple **login form** by validating user input.  
The script should ensure that both **username** and **password** fields are not empty before submission.  
Display an **alert or message** if any field is left blank.  
Make the code clean, readable, and properly commented.  
Also, **explain how the JavaScript improves form interactivity and user experience**.

### Code and output

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

### Observation

* It defines a function calculate\_area that calculates the area of different shapes using a dictionary of lambda functions.
* It includes error handling for unsupported shapes and missing dimensions for rectangles.
* It uses math.pi for a more accurate circle area calculation.
* Example usage demonstrates how to call the function for different shapes, including cases with errors.

For the HTML code in cell 95fa046c:

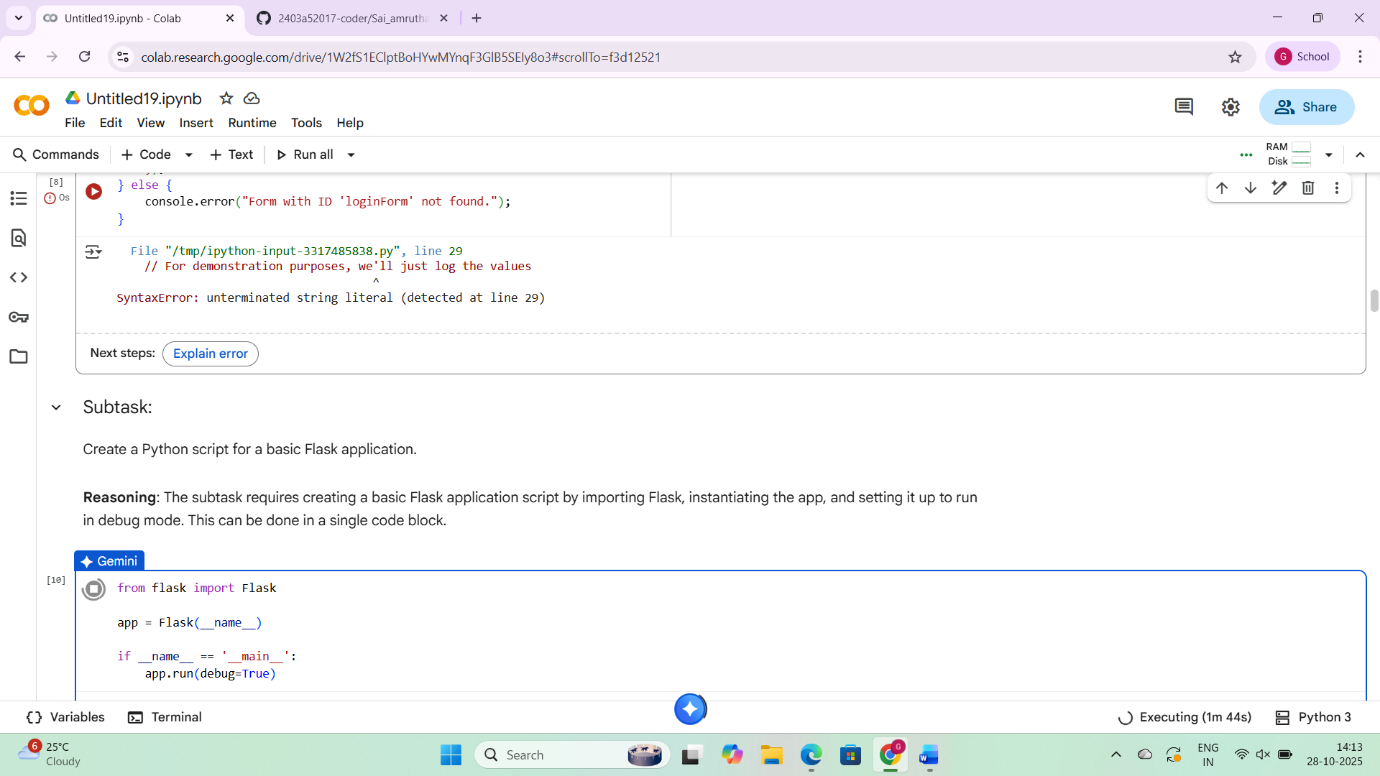
* It defines a basic HTML structure for a "Student Info Portal" homepage with a header, navigation, main content, and footer.
* It includes embedded CSS within the <style> tags for basic styling and layout.
* It uses semantic HTML5 tags like <header>, <nav>, <main>, and <footer>.
* It includes a viewport meta tag for responsiveness.

Task Description #4 – Python Backend Integration  
Task: Ask AI to generate a Flask app that serves the HTML form (Task  
#3) and prints the username on successful login.

### Prompt

Generate a flask app that serves html from task 3 and print the username on successful login

### Code and output



### Observation

* This code appears to be JavaScript intended for client-side form validation.
* It attempts to get form and input elements by their IDs.
* It adds an event listener for the form's submit event to prevent default submission.
* It checks if the username and password fields are empty after trimming whitespace.
* It uses alert and console.log for feedback, which are typical in browser-based JavaScript.
* The error you encountered earlier in this cell (SyntaxError: unterminated string literal) was because JavaScript was being run in a Python environment.

For the code in cell a930d1ce:

* This is a basic Python script for a Flask web application.
* It imports the Flask class.
* It creates an instance of the Flask application.
* The if \_\_name\_\_ == '\_\_main\_\_': block ensures that the development server runs only when the script is executed directly.
* app.run(debug=True) starts the development server in debug mode, which is useful for development as it provides detailed error pages and reloads automatically on code changes.