SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: M. Tech/MCA/MSC		**Assignment Type: Lab**	**AcademicYear:**2025-2026
Course Coordinator Name		Venkataramana Veeramsetty	
Course Code		**Course Title**	AI Assisted Problem Solving Using Python
Year/Sem	II/I	**Regulation**	R25
Date and Day of Assignment	Week5 - Tuesday	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	**M**. Tech/MCA/MSC
AssignmentNumber:14.3(Presentassignmentnumber)/**24**(Totalnumberofassignments)			

Q.No.	Question	*ExpectedTime to complete*
1	Lab 14 – Web Frontend Development: AI-assisted HTML/CSS/JS with Python **Lab Objectives** • To understand how AI can generate HTML/CSS/JS templates. • To practice integrating frontend and backend (Python) for small apps. • To evaluate AI-generated code for readability, reusability, and responsiveness. **Learning Outcomes** After completing this lab, students will be able to: 1. Generate HTML/CSS layouts using AI tools. 2. Add JavaScript interactivity with AI suggestions. 3. Integrate basic Python (Flask/Streamlit) backend to serve frontend. 4. Evaluate AI-generated web code for responsiveness and usability. **5.** Debug and refine AI-generated frontend code. **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. **Expected Output:** • HTML code with `<header>`, `<nav>`, `<footer>`. • Clean indentation, proper tags, and comments.	Week5 - Tuesday

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.

Expected Output:

- HTML + CSS combined.
- AI explains how CSS classes apply.

Expected Output: AI refactors with with open() and try-except:

Task Description #3 – JavaScript Interactivity

Task: Prompt AI to generate a JS script that validates a simple login form (non-empty username/password).

Expected Output:

Working on submit JS validation.

Clear error messages if inputs are empty.

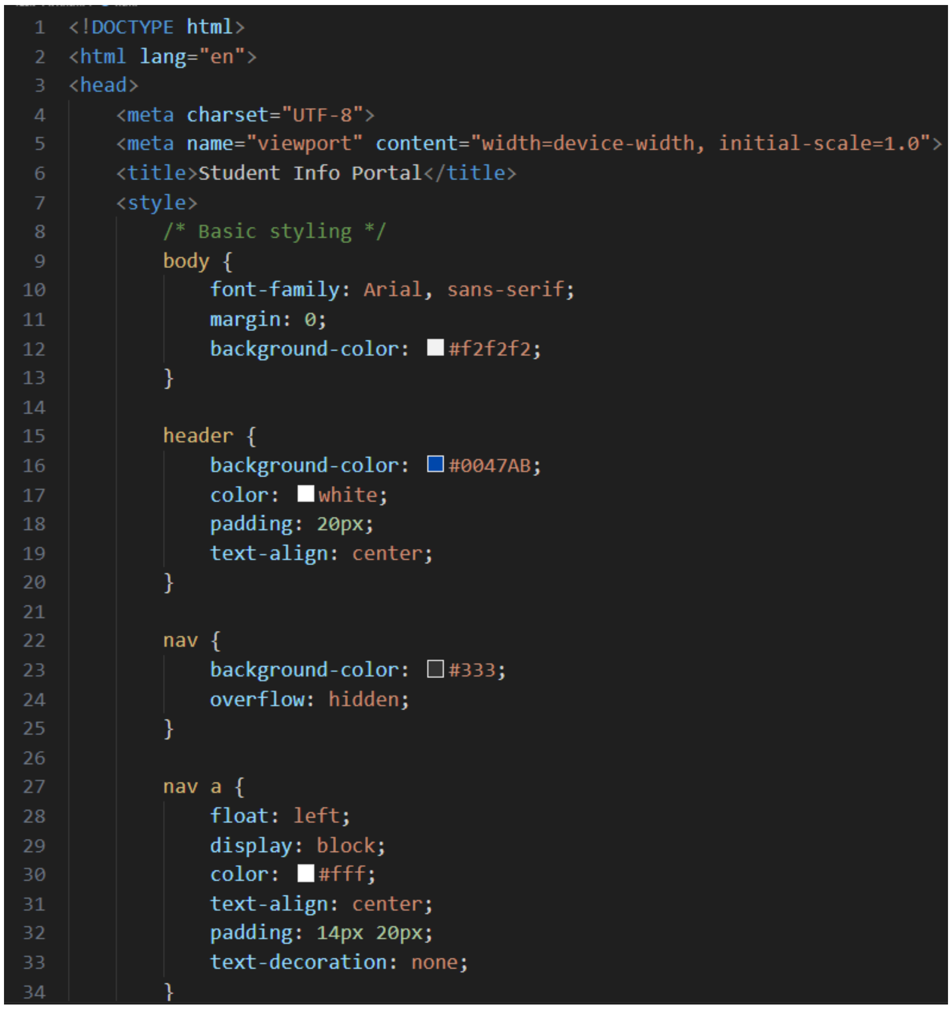
Task Description #4 – Python Backend Integration

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

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 HTML5 homepage for a 'Student Info Portal' with header, navigation menu, main section, and footer.

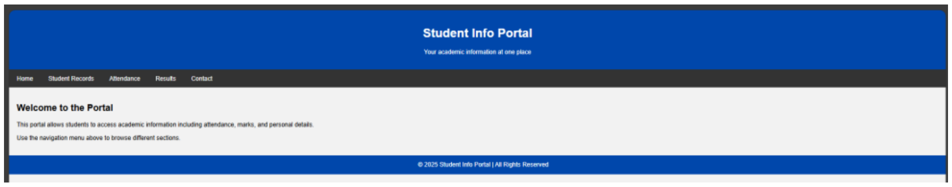


```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Info Portal</title>
    <style>
        /* Basic styling */
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            background-color: #f2f2f2;
        }

        header {
            background-color: #0047AB;
            color: white;
            padding: 20px;
            text-align: center;
        }

        nav {
            background-color: #333;
            overflow: hidden;
        }

        nav a {
            float: left;
            display: block;
            color: #fff;
            text-align: center;
            padding: 14px 20px;
            text-decoration: none;
        }
```

Expected Output:

- HTML code with `<header>`, `<nav>`, `<footer>`.
- Clean indentation, proper tags, and comments.

Practical output:



Student Info Portal

Your academic information at one place

Welcome to the Portal

This portal allows students to access academic information including attendance, marks, and personal details.

Use the navigation menu above to browse different sections.

© 2025 Student Info Portal | All Rights Reserved

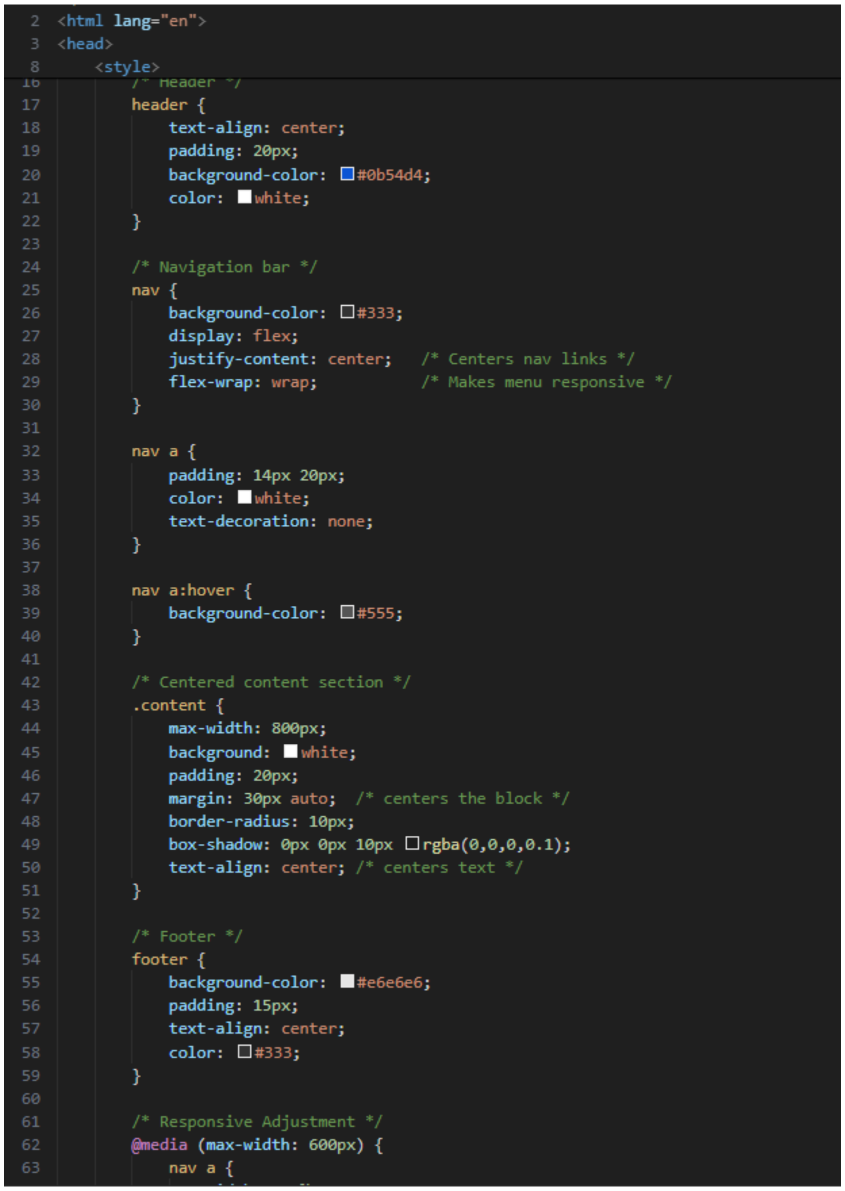
Task Description #2 – CSS Styling

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 HTML page: create a responsive navigation bar, center the main content section, and style the footer with a light-gray background. Use clean, readable CSS with proper comments.

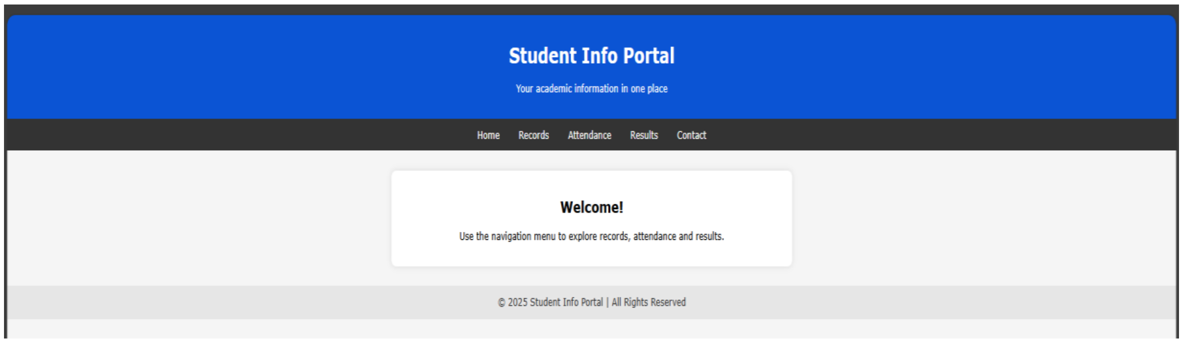


```
<html lang="en">
<head>
    <style>
        header {
            text-align: center;
            padding: 20px;
            background-color: #0b54d4;
            color: white;
        }

        /* Navigation bar */
        nav {
            background-color: #333;
            display: flex;
            justify-content: center;   /* Centers nav links */
            flex-wrap: wrap;           /* Makes menu responsive */
        }

        nav a {
            padding: 14px 20px;
            color: white;
            text-decoration: none;
        }

        nav a:hover {
            background-color: #555;
        }

        /* Centered content section */
        .content {
            max-width: 800px;
            background: white;
            padding: 20px;
            margin: 30px auto;  /* centers the block */
            border-radius: 10px;
            box-shadow: 0px 0px 10px rgba(0,0,0,0.1);
            text-align: center; /* centers text */
        }

        /* Footer */
        footer {
            background-color: #e6e6e6;
            padding: 15px;
            text-align: center;
            color: #333;
        }

        /* Responsive Adjustment */
        @media (max-width: 600px) {
            nav a {
```

Expected Output:

- HTML + CSS combined.
- AI explains how CSS classes apply.

Practical output :



Student Info Portal

Your academic information in one place

Home Records Attendance Results Contact

Welcome!

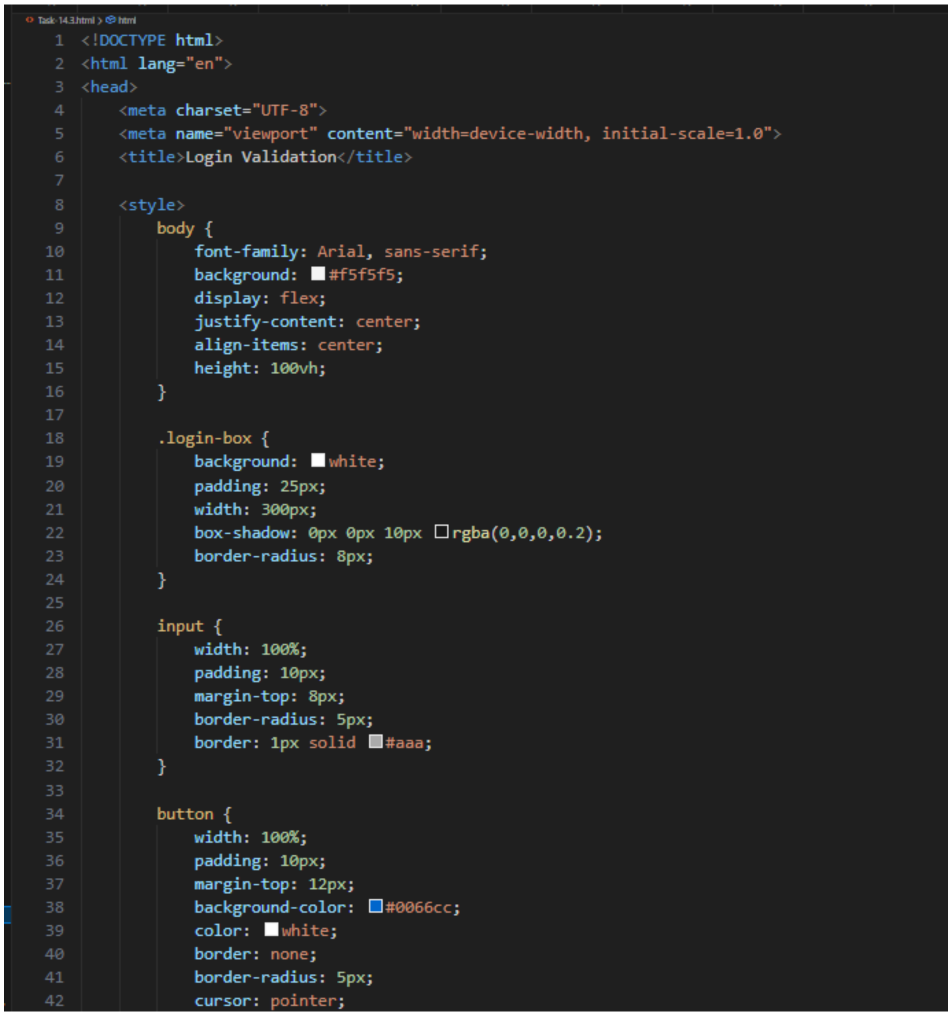
Use the navigation menu to explore records, attendance and results.

© 2025 Student Info Portal | All Rights Reserved

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 JavaScript code that validates a simple login form by checking that username and password are not empty. Show clear error messages on submit if any field is blank, using clean and well-commented JS.

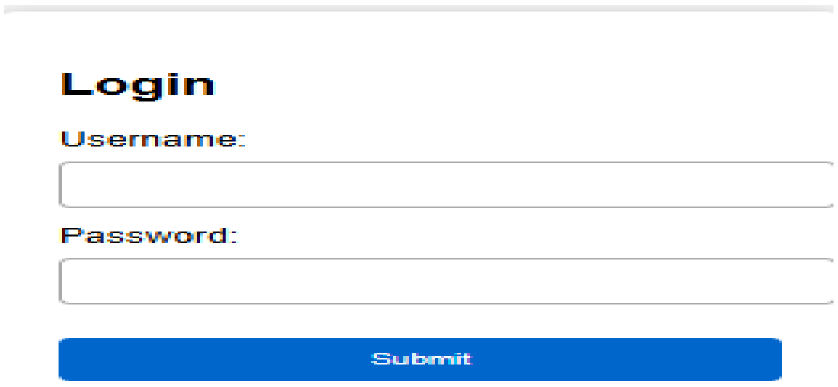


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login Validation</title>

    <style>
        body {
            font-family: Arial, sans-serif;
            background: #f5f5f5;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }

        .login-box {
            background: white;
            padding: 25px;
            width: 300px;
            box-shadow: 0px 0px 10px rgba(0,0,0,0.2);
            border-radius: 8px;
        }

        input {
            width: 100%;
            padding: 10px;
            margin-top: 8px;
            border-radius: 5px;
            border: 1px solid #aaa;
        }

        button {
            width: 100%;
            padding: 10px;
            margin-top: 12px;
            background-color: #0066cc;
            color: white;
            border: none;
            border-radius: 5px;
            cursor: pointer;
```

Expected Output:

Working on submit JS validation.
Clear error messages if inputs are empty.

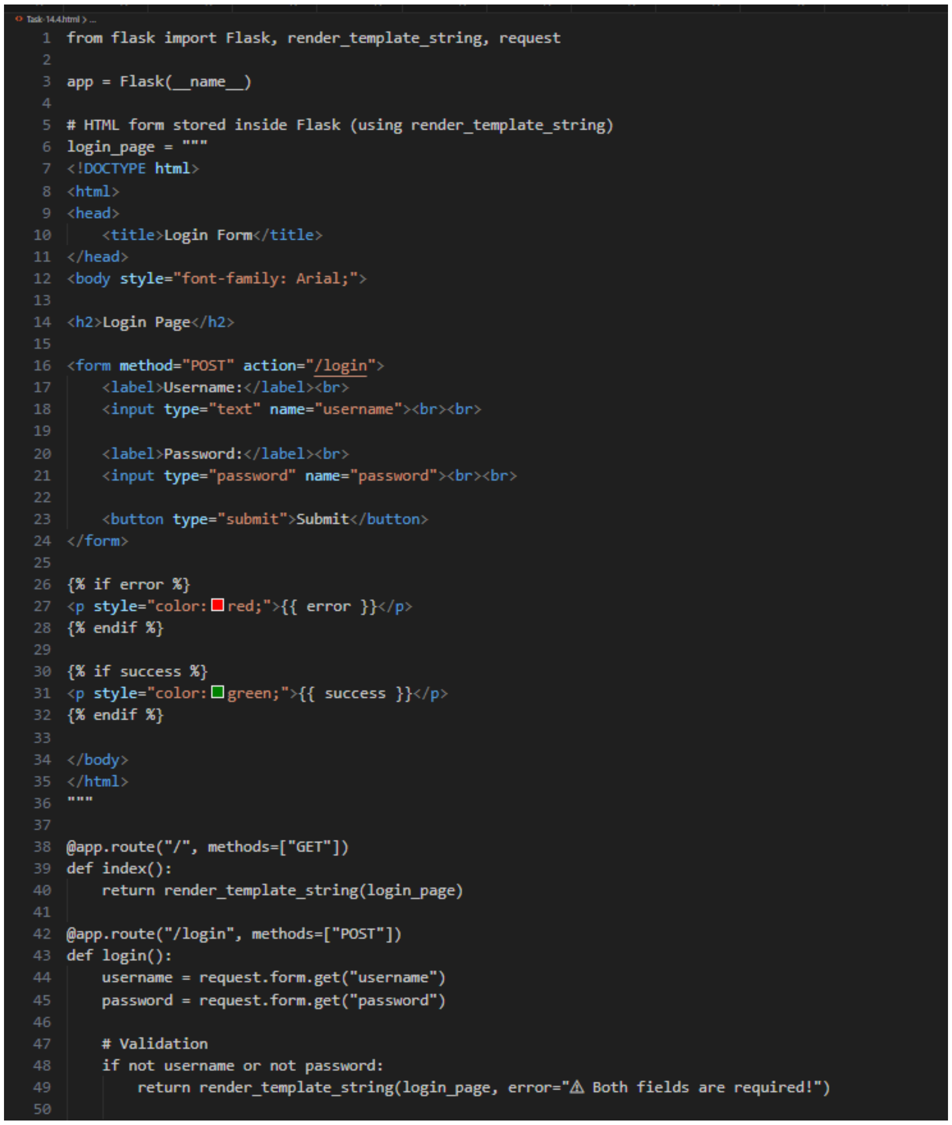
Practical output:



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 Python Flask app that serves the login form and processes the submitted data. On successful login (non-empty fields), display the entered username; otherwise show an error message.



```python
from flask import Flask, render_template_string, request

app = Flask(__name__)

# HTML form stored inside Flask (using render_template_string)
login_page = """
<!DOCTYPE html>
<html>
<head>
    <title>Login Form</title>
</head>
<body style="font-family: Arial;">

<h2>Login Page</h2>

<form method="POST" action="/login">
    <label>Username:</label><br>
    <input type="text" name="username"><br><br>

    <label>Password:</label><br>
    <input type="password" name="password"><br><br>

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

{% if error %}
<p style="color:red;">{{ error }}</p>
{% endif %}

{% if success %}
<p style="color:green;">{{ success }}</p>
{% endif %}

</body>
</html>
"""

@app.route("/", methods=["GET"])
def index():
    return render_template_string(login_page)

@app.route("/login", methods=["POST"])
def login():
    username = request.form.get("username")
    password = request.form.get("password")

    # Validation
    if not username or not password:
        return render_template_string(login_page, error="⚠ Both fields are required!")

```

Practical output:

Login Successful!

Welcome, Student1!

Back to Login