LAB ASSIGNMENT – 14.3

NAME : VADIYALA CHANDANA

ROLL NO : 2403A54103

BATCH : 01

COURSE : AI ASSISTED CODING

TASK 1 :

PROMPT:
Create a simple HTML homepage for a 'Student Info Portal'. The page should include a header with the title, a navigation menu with links (like Home, About, Courses, Contact), and a footer. Make sure the layout is clean and well-structured.

CODE :

```
from IPython.display import HTML

html_content = """
<!DOCTYPE html>
<html>
<head>
<title>Student Info Portal</title>
<style>
  body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
  }
  header {
    background-color: #f2f2f2;
    padding: 10px;
    text-align: center;
  }
  nav {
    background-color: #e0e0e0;
    padding: 10px;
    text-align: center;
  }
  nav a {
    margin: 0 15px;
    text-decoration: none;
    color: #333;
  }
  footer {
    background-color: #f2f2f2;
```

```
    background-color: #f2f2f2;
    padding: 10px;
    text-align: center;
    position: fixed;
    bottom: 0;
    width: 100%;
  }
</style>
</head>
<body>

<header>
  <h1>Student Info Portal</h1>
</header>

<nav>
  <a href="#">Home</a>
  <a href="#">About</a>
  <a href="#">Courses</a>
  <a href="#">Contact</a>
</nav>

<main>
  <!-- Content goes here -->
</main>

<footer>
  <p>&copy; 2023 Student Info Portal</p>
</footer>

</body>
```

TASK 2 :

PROMPT :

Add CSS styling to the 'Student Info Portal' homepage created earlier. Include the following features: • A responsive navigation bar that looks good on both desktop and mobile. • A centered content section for the main text or information. • A footer with a light gray background and centered text. Make sure the design is clean and visually appealing."

CODE :

```
from IPython.display import HTML

html_content = """
<!DOCTYPE html>
<html>
<head>
<title>Student Info Portal</title>
<style>
  body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
  }
  header {
    background-color: #f2f2f2;
    padding: 10px;
    text-align: center;
  }
  nav {
    background-color: #e0e0e0;
    padding: 10px;
    text-align: center;
  }
  nav a {
    margin: 0 15px;
    text-decoration: none;
    color: #333;
    display: inline-block; /* Make links behave like blocks for padding */
    padding: 5px 10px; /* Add padding for better touch targets */
```

```
  /* Responsive navigation */
  @media (max-width: 600px) {
    nav {
      display: flex; /* Use flexbox for alignment */
      flex-direction: column; /* Stack links vertically on small screens */
      align-items: center; /* Center align items */
    }
    nav a {
      margin: 5px 0; /* Adjust margin for vertical stacking */
    }
  }

  main {
      max-width: 800px; /* Set a maximum width for content */
      margin: 20px auto; /* Center the main content with auto margins */
      padding: 0 20px; /* Add some padding on the sides */
  }


  footer {
    background-color: #f2f2f2;
    padding: 10px;
    text-align: center;
    position: fixed;
    bottom: 0;
    width: 100%;
  }
</style>
</head>
<body>
```

```
<header>
  <h1>Student Info Portal</h1>
</header>

<nav>
  <a href="#">Home</a>
  <a href="#">About</a>
  <a href="#">Courses</a>
  <a href="#">Contact</a>
</nav>

<main>
  <!-- Content goes here -->
    <h2>Welcome to the Student Info Portal</h2>
    <p>This portal provides information about courses, schedules, and other important resources for students
</main>

<footer>
  <p>&copy; 2023 Student Info Portal</p>
</footer>

</body>
</html>
"""

display(HTML(html_content))
```

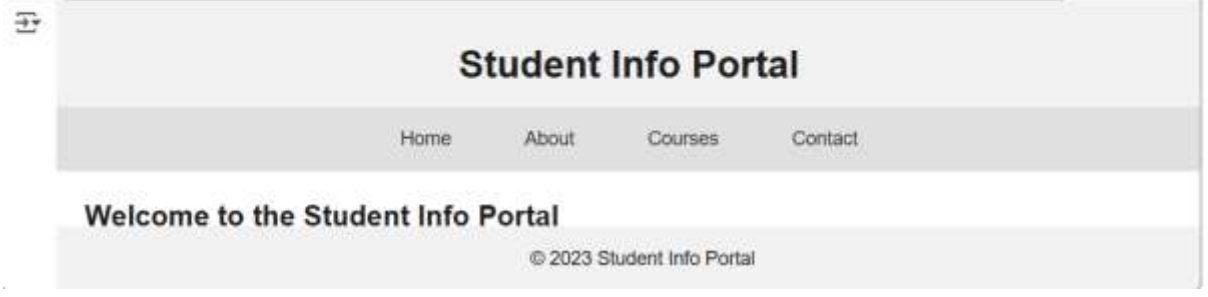

```
  <a href="#">Courses</a>
  <a href="#">Contact</a>
</nav>

<main>
  <!-- Content goes here -->
    <h2>Welcome to the Student Info Portal</h2>
    <p>This portal provides information about courses, schedules, and other important resources for students
</main>

<footer>
  <p>&copy; 2023 Student Info Portal</p>
</footer>

</body>
</html>
"""

display(HTML(html_content))
```



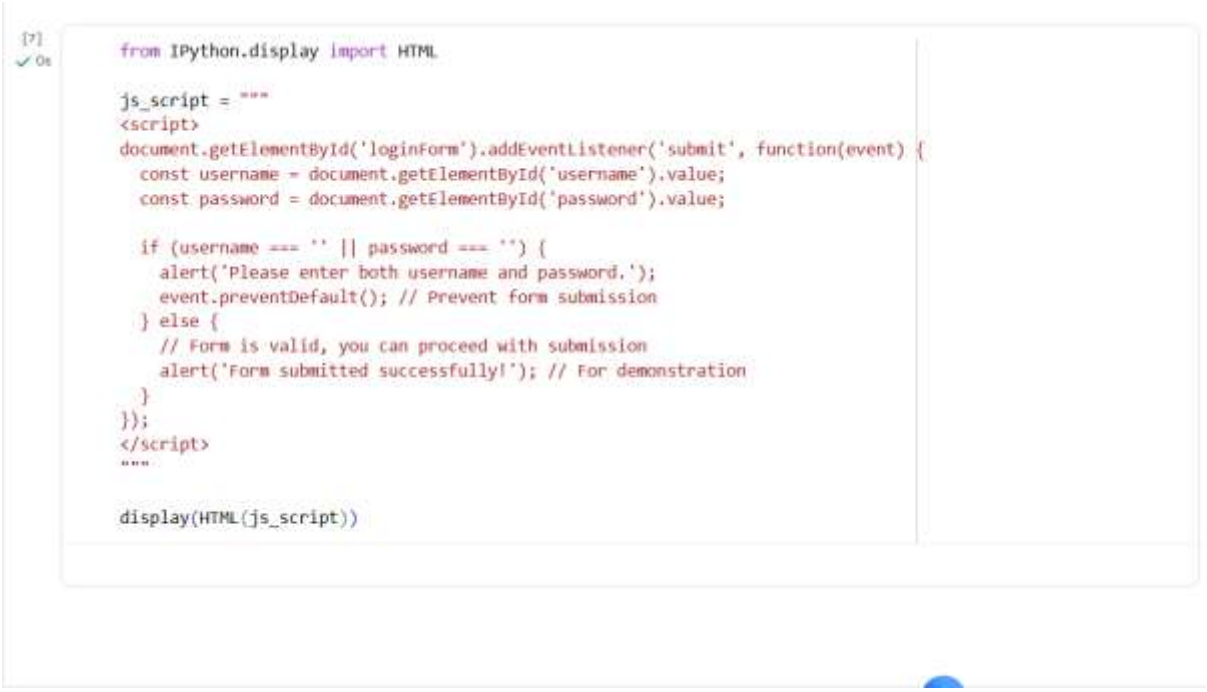
TASK 3:

PROMPT:

Generate a JavaScript script that validates a simple login form. The script should check that both the username and password fields are not empty. If any field is empty, display an alert message to the user. Otherwise, allow the form to be submitted

CODE :

Now, here is the JavaScript code to validate the form. This script will check if the username and password fields are empty when the form is submitted.



```
[7]
✓ 0s
from IPython.display import HTML

js_script = """
<script>
document.getElementById('loginForm').addEventListener('submit', function(event) {
  const username = document.getElementById('username').value;
  const password = document.getElementById('password').value;

  if (username === '' || password === '') {
    alert('Please enter both username and password.');
    event.preventDefault(); // Prevent form submission
  } else {
    // Form is valid, you can proceed with submission
    alert('Form submitted successfully!'); // For demonstration
  }
});
</script>
"""

display(HTML(js_script))
```

TASK 4 :

PROMPT :

Generate a simple Flask web application that serves the login HTML form (from Task #3). When the user submits the form with a valid username and password, display a new page that prints 'Welcome, [username]'. Make sure to include both the HTML form and Flask backend code

CODE :

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

app = Flask(__name__)

html_form = """
<form id="loginForm" action="/login" method="post">
  <div>
    <label for="username">Username:</label>
    <input type="text" id="username" name="username">
  </div>
  <div>
    <label for="password">Password:</label>
    <input type="password" id="password" name="password">
  </div>
  <button type="submit">Login</button>
</form>
"""

@app.route('/')
def login():
  return render_template_string(html_form)

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

    # Simple validation
```

```python
@app.route('/')
def login():
  return render_template_string(html_form)

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

    # Simple validation
    if username == 'admin' and password == 'password123':
        return f'<h1>Welcome, {username}!</h1>'
    else:
        return '<h1>Invalid username or password</h1>'

if __name__ == '__main__':
    app.run(debug=True)
```

```
 * Serving Flask app '__main__'
 * Debug mode: on
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a producti
 * Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with watchdog (inotify)
```