Weekly Assignment 27-Oct-24 : Python with Flask

Submission Date : 31-Oct-24 (Before 12:00 PM)

1. Hello Flask Website

- **Task**: Create a simple "Hello, World!" Flask application.
- **Requirements**: Make a single route (`/`) that displays “Hello, Flask!” on a web page.
- **Hint**: Start by setting up a basic Flask app with a single route. Use `app.route()` to set the URL path.

2. Personal Bio Page

- **Task**: Design a simple personal bio page.
- **Requirements**: Add a route (`/bio`) with basic information like name, age, and hobbies displayed in HTML.
- **Hint**: Use Flask's render_template function and create a basic HTML file to display personal details.

3. Calculator App

- **Task**: Build a simple calculator that can add two numbers.
- **Requirements**: Create a form where users enter two numbers. Display the result on submission.
- **Hint**: Use HTML forms and handle data in Flask using the `request.form` method to get the inputs.

4. Mini To-Do List

- **Task**: Create a basic to-do list web app where users can add tasks.
- **Requirements**: Implement a form to add tasks and display the tasks on the same page.
- **Hint**: Use a list to store tasks temporarily and a `POST` method to add new tasks to the list.

5. Random Quote Generator

- **Task**: Create an app that displays a random motivational quote from a predefined list each time the page is refreshed.
- **Requirements**: Display one random quote from a list of quotes every time the user visits the `/quote` page.
- **Hint**: Use Python’s `random.choice()` to select a quote from a list and display it using HTML.

6. Simple Login Page

- **Task**: Build a basic login form with username and password fields.
- **Requirements**: Display a welcome message if the username is “user” and the password is “password.”

- **Hint**: Use `POST` requests and `if` statements to check login credentials.

7. Image Gallery

- **Task**: Create a simple gallery page that displays three static images.
- **Requirements**: Display three images side by side on a page.
- **Hint**: Use HTML `<img>` tags in your HTML template and store the images in a `/static` folder in the project.

8. Feedback Form

- **Task**: Make a feedback form that saves users' names and feedback temporarily.
- **Requirements**: Save the submitted feedback to a list and display the list on the same page.
- **Hint**: Use Python lists or dictionaries to store each feedback entry, and display feedback history at the bottom of the page.

9. Basic Data Table with Jinja

- **Task**: Display a table of users and their details (name, age, city).
- **Requirements**: Use a predefined list of dictionaries and render it in an HTML table.
- **Hint**: Use Jinja templating with `for` loops to iterate over the list and display the data in a table format.

10. Temperature Converter

- **Task**: Create a simple temperature converter that converts Celsius to Fahrenheit.
- **Requirements**: Allow users to enter a Celsius value and display the converted Fahrenheit value.
- **Hint**: Use forms for input, `request.form` to get the Celsius value, and simple arithmetic in Python to calculate the Fahrenheit temperature.