Assignment - 3

1. What is Flask, and how does it differ from other web frameworks?

- Flask is a micro web framework for Python based on Werkzeug, Jinja2, and other libraries. It's lightweight and modular, allowing developers to add only the components they need. Flask provides flexibility and simplicity, making it easy to get started with web development. Unlike some other frameworks like Django, Flask does not include built-in functionalities such as ORM (Object-Relational Mapping) or authentication.

2. Describe the basic structure of a Flask application.

- A basic Flask application typically consists of a Python script that defines the application instance, routes, and other configurations. It includes templates directory for HTML templates and a static directory for static files like CSS, JavaScript, and images.

3. How do you install Flask and set up a Flask project?

- Flask can be installed using pip:

```
pip install Flask
```

To set up a Flask project, you create a directory for your project, then create a Python script (usually named app.py), and set up directories for templates and static files.

4. Explain the concept of routing in Flask and how it maps URLs to Python functions.

- Routing in Flask is done using the @app.route() decorator. It maps URLs to Python functions, so when a request is made to a specific URL, Flask calls the associated Python function (view function) to handle the request.

5. What is a template in Flask, and how is it used to generate dynamic HTML content?

- A template in Flask is an HTML file with placeholders for dynamic content. Flask uses Jinja2 templating engine to render templates. Dynamic content can be passed from the view function to the template, allowing for the generation of HTML content based on data.

6. Describe how to pass variables from Flask routes to templates for rendering.

- Variables can be passed from Flask routes to templates by including them as arguments in the render_template() function or by using the {{ variable_name }} syntax in the template.

7. How do you retrieve form data submitted by users in a Flask application?

- Form data submitted by users can be retrieved in Flask using the request.form object. This object contains key-value pairs of form data submitted via POST request.

8. What are Jinja templates, and what advantages do they offer over traditional HTML?

- Jinja templates are HTML files with special syntax provided by the Jinja2 templating engine. They offer advantages over traditional HTML by allowing for dynamic content generation, including conditional statements, loops, template inheritance, and filters.

9. Explain the process of fetching values from templates in Flask and performing arithmetic calculations.

- Values from templates in Flask can be fetched using the {{ variable_name }} syntax. Arithmetic calculations can be performed directly in the template using Jinja2 syntax, such as {{ variable1 + variable2 }}.

10. Discuss some best practices for organizing and structuring a Flask project to maintain scalability and readability.

- Use blueprints to organize routes into separate modules.
- Separate concerns by using models for database interactions, views for handling requests, and templates for presentation logic.
- Utilize Flask extensions for common functionalities like authentication, database management, and form handling.
- Implement error handling to gracefully manage exceptions and errors.
- Use environment variables for configuration to keep sensitive information secure.
- Document code and follow PEP 8 guidelines for better readability and maintainability.
- Regularly refactor code to keep it organized and scalable as the project grows.

These concepts and practices should provide a solid foundation for developing Flask applications effectively.