ASSIGNMENT-3

**1.What is flask,how does it differ from other web frameworks?**

Ans) Flask is a lightweight web framework for python that allows you to build web application quickly and with minimal ovehead it provides tools, libraries, and pattern to help developers create web applications easily. Flask differs from other web frameworks in its simplicity and flexibility. unlike some other frameworks li Django, Flask does not come with built in features such as an ORM or for validation. insteads, flask follows a minimal list philosophy, allowing developers to choose and integrate the specific libraries and tools they need for their project. this makes Flask very adaptable and suitable for a wide range of web development tasks, from smail project to large scale applications.

**2.Describe the basic structure of the Flask applications?**

Ans) The basic structure of a flask application typically consists of:

**1.Application** Object Creation: you create a flask application instance usually by instanting the flask class:

**Example :**

from flask import flask

app = Flask(\_\_name\_\_)

**2.Routes**: Define routes to different parts of your application.routes are urls that the applications responds to and they are associated with python functions that generates HTTP response.

Example:

Def index ():

Return ‘hello, world!’

**3.views** : views are python functions that are associated with routes,they handle requests and return response view can render templates process from data and perform other tasks:

Example:

Def profile(username):

Return f’user {username}’

**4.Templates**: Templates are HTML files that can be rendered by flask to generate dynamic content. Templates typically contain placeholders that are replaced with actual data when the page is rendered:

**Example:**

<html>

<head><title>{{title}}</title></head>

<body>

<h1>Hello,{{name }}!</h1>

<?body>

</html>

**5.configuration**: configuration options for the Flask application can be set using the config attributes or by using configuration files:

**Example:**

App. config{‘DEBUG’} =True

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

Ans) To install the flask and set up a project flask you can follow these the steps:

**1.install Flask:**

You can install flask using pip,the python package manger. Open your terminal or commands prompt and run the following command:

Pip install flask

**2.set up a flask project:**

Create a new directory for your Flask project .inside this directory,you can creatw your flask application files:

Mkdir my\_flask\_project

Cd my\_flask\_project

**3.create a python file:**

Create a python file for your flask application you can name it app.py

From flask import Flask

app= flask(--name--)

def hello();

return ‘Hello,world!’:

if \_\_name\_\_==’\_\_main\_\_’:

app.run(debug=True)

**4.Run the Flask application:**

To run the flask application execute the app.py file:

Python app.py

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

Ans)By using the route() decorator, we can define routes for different URLs and map them to the appropriate view functions that will generate the desired response. This allows us to create a structured and organized web application with distinct functionalities for different routes.

App routing is the technique used to map the specific URL with the associated function intended to perform some task. The Latest Web frameworks use the routing technique to help users remember application URLs. It is helpful to access the desired page directly without navigating from the home page. For this we use the url\_for() function which accepts name of the function as the first argument and the rest of the arguments as variable part of the URL rule. In the below example we pass the function names as arguments to the url\_for function and print out the result when those lines are executed.

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

Ans)Templates are files that contain static data as well as placeholders for dynamic data. A template is rendered with specific data to produce a final document. Flask uses the Jinja template library to render templates. In your application, you will use templates to render HTML which will display in the user's browser. **Flask Templates: Creating Dynamic Web Pages with Jinja2**

1. In web development, creating dynamic web pages that can adapt to different data and user inputs is crucial. ...
2. Before we start, make sure you have Flask installed. ...
3. Next, create a new Flask application and set up the basic structure.

The flask framework has been written in a way so that it looks for HTML template files in a folder that should be named templates. So, you should create such an empty folder and then put all the HTML templates in there. So, the Python script stays outside the templates folder.

Flask uses a templating engine called jinja. This allows you to write python code inside your html files. It also allows you to pass information from your back-end (the python script) to your HTML files. In your HTML file you can use the following syntax to evaluate python statements.

Because of that Flask configures the Jinja2 template engine for you automatically. Templates can be used to generate any type of text file. For web applications, you'll primarily be generating HTML pages, but you can also generate markdown, plain text for emails, and anything else.

**6.Describe how to pass variables from Flask routeHow are variables passed from app to template?**

Ans)The templates in Flask are handled by the Jinja template engine, which comes with Flask when you first install it. The render\_template() function both selects the template file to be used and passes to it any values or variables it needs.s to templates for rendering

You can pass variables as arguments to render\_template() to use those variables in a template file.

Flask uses the Jinja template library to render templates. In your application, you will use templates to render HTML which will display in the user's browser. In Flask, Jinja is configured to autoescape any data that is rendered in HTML templates.

**Steps for passing JavaScript variables to Python in Flask:**

1. Create a Flask route that will receive the JavaScript variable.
2. Create a JavaScript function that will send the variable to the Flask route using an AJAX request.
3. In the Flask route, retrieve the variable using the request object
4. in C++, when non-array variable are passed as arguments to a function, the function cannot change those variables. This is because non-array variables are ordinarily passed by value. A copy of the argument's value is passed to the function and stored in a corresponding parameter.
5. from flask import Flask, render\_template, request.
6. @app.route('/')
7. def student():
8. return render\_template('student.html')
9. @app.route('/result',methods = ['POST', 'GET'])
10. def result():
11. if request. method == 'POST':
12. return render\_ template ("result .html ", result = result)

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

**Working –**

1. importing flask and creating a home route which has both get and post methods.
2. defining a function with name gfg.
3. if requesting method is post, which is the method we specified in the form we get the input data from HTML form.

You can get form data from Flask's request object with the form attribute: from flask import Flask, request app = Flask(\_\_name\_\_) @app. route('/', methods=['GET', 'POST']) def index(): data = request. form['input\_name'] # pass the form field name as key .

We import our form from our forms package and instantiate it in the view. Then we run form. validate\_on\_submit() . This function returns True if the form has been both submitted (i.e. if the HTTP method is PUT or POST) and validated by the validators we defined in forms.py.

This attribute lets you specify the value of the Content-Type HTTP header included in the request generated when the form is submitted. This header is very important because it tells the server what kind of data is being sent. By default, its value is application/x-www-form - urlen code.

The Request, in Flask, is an object that contains all the data sent from the Client to Server. This data can be rec The get() method of the Form Data interface returns the first value associated with a given key from within a Form Data object. If you expect multiple values and want all of them, use the get All () method instead. overed using the GET/POST Methods.

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

If you template HTML, Jinja2 is more readable because its syntax (curly brackets) is easy to visually distinguish from HTML code. Mako, on the other hand has a fair share of angle brackets. One app I developed had moderately sized Mako templates, which were very hard to edit because its html-like syntax visually clashes with actual HTML.

Another difference is that Mako allows execution of arbitrary Python code within templates. If you want, you can have all actual application code inside the template, php-style :)

Jinja2, on the other hand, does not allow python code inside templates, and if you want something not included in standard filters (though they include lots of stuff!), you need to define it in your app (e.g. process data in the controller or register a custom filter)

One of the key benefits of using Jinja templates is the ability to reuse code across multiple pages. This can save you time and effort when creating complex web applications. To achieve code reusability in Jinja, you can use macros and include statements

Jinja is a text templating language. It allows you to process a block of text, insert values from a context dictionary, control how the text flows using conditionals and loops, modify inserted data with filters, and compose different templates together using inheritance and inclusion.

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

Ans) **Making a Request with fetch**

1. fetch() takes two arguments, a URL and an object with other options, and returns a Promise . ...
2. By default, the GET method is used. ...
3. To send data, use a data method such as POST, and pass the bo ou can get form data from Flask's request object with the form attribute: from flask import Flask, request app = Flask(\_\_name\_\_) @app. route('/', methods=['GET', 'POST']) def index(): data = request. form['input\_name'] # pass the form field name as key dy option. ...
4. To send form data, pass a populated FormData object.

youcan get form data from Flask's request object with the form attribute: from flask import Flask, request app = Flask(\_\_name\_\_) @app. route('/', methods=['GET', 'POST']) def index(): data = request. form['input\_name'] # pass the form field name as key.

Optionally, you can include an object as the second argument, where you can specify various settings such as the HTTP method, headers, and more. Here's a simple example of a basic fetch request: fetch('https://api.example.com/data') . then(response =>response

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

**ANS**: Prerequisites.

* Project Overview.
* Get Started. Create a Virtual Environment. Add Dependencies.
* Initiate Your Flask Project. Run the Flask Development Server. Transform Your Project Into a Package. ...
* Leverage Blueprints.
* Introduce Templates. Build a Base Template. ...
* Improve the User Experience. Include a Navigation Menu. ...
* Conclusion.

**Organize Flask Code Professionally**

1. create directory & virtual environment then run the virtual environment.
2. installing flask over pip (as well as other libraries)
3. export FLASK\_APP parameter i.e. “export FLASK\_APP=app.py”
4. put all your code in one file for example app.py.
5. execute flask run command in terminal.

Everything the app needs is in one folder, here named my-flask-app. That folder contains two folders, specifically named static and

**9 Best practices for structuring Python projects**

1. Organize your code. Properly organizing your code is crucial when working on a Python project. ...
2. Use consistent naming. ...
3. Use version control. ...
4. Use a package manager. ...
5. Create virtual environments. ...
6. Comment your code. ...
7. Test, test, test. ...
8. Lint and Style.

templates. The static folder contains assets used by the templates, including CSS files, JavaScript files, and images.

**The Pyramid Folder Structure: The Best Folder Structure for Project Management**

* Create a general overview folder of a project.
* Create folders for the different types of assets that will arise within that project.
* Build a nested structure so that everyone can quickly find what they need.