Flask Python

Course Agenda

Flask Python

- Introduction to **web frameworks**
- Introduction to **Flask**
- **Installing** Flask
- **Understanding** Flask
- Flask **vs** Django
- **Applications** of Flask
- Advantages **and** disadvantages of Flask

Introduction to web frameworks

Introduction to web frameworks

Here is an overview:

Consider that you are swiping down the feed on Instagram:

- With every swipe and refresh, the application (Instagram) will have to generate content based on your interaction and activity in the app.
- It has to retrieve data from a couple of servers for querying, receiving and even data updation.

Introduction to web frameworks

Here is an overview:

- What if the app never had to do any real-time processing or data retrieval?
- It would be static and a lot simpler and faster to work with, correct?
- Web frameworks sometimes are associated with being slower that is because of the complexity involved to retrieve and serve the data.

Introduction to web frameworks

There are two sides to a web application ecosystem:

Introduction to web frameworks

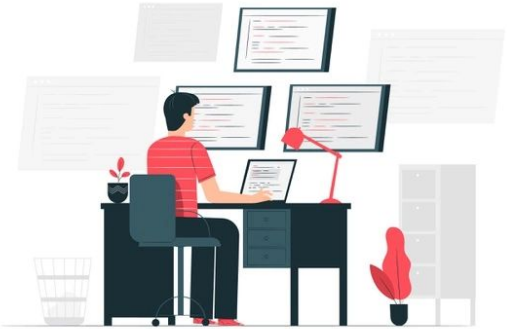
Quick look at the two sides:

- **Client-side:** The code gets executed on the user's browser and serves as a user interface to work with the application.
- **Server-side:** Consists of the entire backend needed to design, build, host and test applications over the net.

Introduction to web frameworks

So, what is a web framework?

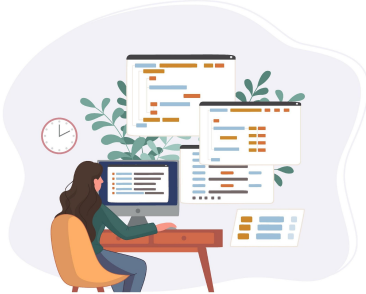
- A web framework is an architecture that includes various tools, libraries, and techniques for efficiently building and maintaining large online applications.
- Code reusability is a large factor in designing web applications.



Introduction to web frameworks

What about the coding?

- To create the backend of the web app, we require the usage of a server-side language.
- Python, being one of the world's most popular programming languages consists of a lot of frameworks like Django and Flask for this purpose.



Introduction to Flask

Introduction to Flask

What is Flask?

- Flask is a lightweight framework that that provides a wide variety of tools and techniques that can be used to build both small and large scale applications.
- Its core functionality is pretty lightweight as it depends on two external libraries called the Jinja2 template and Werkzeug WSGI toolkit to function.

Introduction to Flask

WSGI, Werkzeug and Jinja2?

- **WSGI** – Stands for Web Server Gateway Interface, it is used as a standard for the creation of an interface between the server and the web application.
- **Werkzeug** – It is a toolkit that is used to implement requests, response objects and other important utility functions. Serves as a foundation for a web framework itself.
- **Jinja2** – Jinja2 is a templating engine built for Python. It is popularly used to render dynamic web pages.

Introduction to Flask

For what tasks would you use Flask?

- Working on supporting secure cookies.
- Where there is a requirement for fast debugging.
- Jinja2 templating.
- Where a web app requires a modular approach to design.

Introduction to Flask

Things to keep in mind when constructing an API for a web application:

- New prospects or visitors should be able to create accounts for themselves.
- Users who register must be able to do tasks such as logging in, logging out and editing some information on their profile.
- Additional functionality for registered users can include access-based creation, updation and maintenance of tasks.

Introduction to Flask

Prerequisites to learning and using Flask:

Here are some of the important things you need to know:

- Python syntax and workflow
- Hands-on experience with HTML

Installing Flask

Installing Flask

Pretty simple steps:

- Install the latest version of Python for your operating system.
- Install an IDE such as Sublime, PyCharm or Visual Studio Code.
- Create a virtualenv in the project to work in.
- Run the following command:
- **$ pip install Flask**

Installing Flask

Verifying the installation:

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
   return "Hello World"

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

```
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

A sample piece of code running on Google Colab

Understanding Flask

Understanding Flask

Debugging in Flask:

- After you have executed a sample piece of code, the development server will not be running in the debug mode.
- Activating debug mode enables automatic reloading. Whenever changes are made to the code, the server automatically reloads.
- It also activates the Python debugger to check on variables during the execution.
- To go into debug mode, stop the running server and run the following piece of code:
- **FLASK_ENV=development flask run**

Understanding Flask

Using HTML templates

- A simple application only displays a message without using any HTML attributes.
- Almost all of the web apps use HTML to display information to the visitor.
- Flask provides a helper function called render_template() that allows the usage of the Jinja template engine.
- Doing this will ensure that you can write HTML code in .html files and even write functionality and logic of the application there too.

Understanding Flask

Setting up a database

- A database is required to store almost all the data that is used by the application.
- You can use many SQL databases and integrate it with sqlite3 module which can help you interact with the database easily and readily.
- Data conversion from one format to the other would be required as the data is used in the form of rows and columns in a database.

Flask vs Django

Flask vs Django

1. Handling admin tasks:

Flask	Django
Flask does not support any admin-related features.	Django provides support any admin-related features.

Flask vs Django

2. Template:

Flask	Django
Flask uses Jinja2 that is based on Django's template engine.	Django comes built-in with a template engine.

Flask vs Django

3. Support for visual debugging:

Flask	Django
Flask does not provide any native support for visual debugging.	Django provides native support for visual debugging.

Flask vs Django

4. Development styles:

Flask	Django
Flask offers a wide variety of tools and techniques to develop apps.	Django's workflow is considered to be very monolithic.

Flask vs Django

5. Development styles:

Flask	Django
Flask was built for rapid prototyping and development.	Django was built for easy and structured development.

Applications of Flask

Applications of Flask

There are numerous examples:

- The website Pinterest is one of the largest social media sites in the world, the creator shared that it was built using the Flask framework.
- Barack Obama's 2012 presidential campaign site used Flask a the main web framework.
- Lyft is a very popular ride sharing app that uses a lot of frameworks to work but it is mainly built around Flask.

Applications of Flask

Here are some of the companies that use Flask:

NETFLIX

SAMSUNG

patreon

Uber

airbnb

NGINX

teradata.

Advantages and disadvantages of Flask

Advantages and disadvantages of Flask

There are numerous advantages:

- Built for rapid prototyping.
- Overall code base is comparatively smaller and simpler.
- Highly scalable even for complex projects.
- Database integration is very simple and straightforward.

Advantages and disadvantages of Flask

There are numerous advantages:

- Flask has a small core that provides great extensibility.
- Minimalism is the approach used to help developers create better apps.
- Lots of documentation available on GitHub and other sources on the web.
- Backed by a very talented community of developers and experts.

Advantages and disadvantages of Flask

Here are some of the disadvantages:

- Setting up the environment for a large project requires a lot of previous experience.
- Maintenance seems to be complex for complex projects.
- Flask does not provide any login mechanism or authentication like login by default.
- Migration of apps can be very difficult.

Summary