# FLASK

1. What is a flask
2. Web framework
3. What is WSGI
4. What is jinja2
5. Templet engine
6. Futures of Flask
7. Advantages of Flask
8. Flask is a Frontend or Back end?
9. Installation process
10. Flask virtual environment
11. Sample program writing
12. Explaining the program

**What is FLASK:**

Flask is a web application framework written in python. Flask is very easy to learn and also its implementation is straightforward. In just a few lines of code, Flask is used in top tech companies also like: Netflix and Reddit

**Web Framework:**

Python Web framework is a collection of packages or modules that allow developers to write Web applications or services Creating a website and writing a code is called a web framework

**Modules:**

A module can define functions, classes, and variables and also include runnable code.

Flask is based on WSGI and JINJA2

**Why flask is used:**

Flask gives the developer varieties of choices when developing web applications, it provides you with tools, libraries, and mechanics that allow you to build a web application •## WHAT IS WSGI? WSGI defines the Web Server Gateway Interface. It is an interface between the web server and the web applications.

**WHAT IS JINJA2?**

Jjinja2 is a web template engine that combines a template with a certain data source to render the dynamic web page

**TEMPLATE ENGINE:**

A template engine enables you to use static template files in your application. At runtime, the template engine replaces variables in a template file with actual values.

**FEATURES OF FLASK:**

• Development server and debugger

• Integrated support for unit testing

• RESTful request dispatching

• Uses Jinja templating

• Support for secure cookies (client-side sessions)

• 100% WSGI 1.0 compliant

• Unicode-based

• Complete documentation

• Google App Engine compatibility

• Extensions available to extend the functionality

**ADVANTAGES OF FLASK:**

• Scalable. Size is everything, and Flask's status as a microframework means that you can use it to grow a tech project such as a web app incredibly quickly.

• Flexible

• Easy

• Lightweight.

• Documentation.

• Not a lot of tools.

• Difficult to get familiar with a more considerable Flask app.

• Maintenance costs.

**Flask a frontend or backend?**

Flask is a back-end framework, which means that it provides the technologies, tools, and modules that can be used to build the actual functionalities of the web app rather than its design or look of it.

**INSTALLATION PROCESS**

• First we have to check that python is there in our device.

If not we have to go to URL and python download

• Now we should download pip's latest version from the command prompt

COMMAND: pip install

• We have to install pip flask now which means from the URL copy the command and install it in the command prompt

''COMMAND: PIP install flask

Now move to the command prompt and check whether the installation was successfully done or not by using some commands

''COMMAND: python –version

COMMAND: pip –version ''

Now we have to create a file in the command prompt

'' COMMAND: mkdir. flask

(here flask is a file name ) Now we should check that the file was saved into drive

'' COMMAND: dir.

Now check the pip list command

Cd flask dir

(dir means checking whether that file is exited or not)

**Flask – Virtual Environment(venv)**

A virtual environment is a Python environment in which the Python interpreter, libraries, and scripts are installed.

Move to the command prompt and command

COMMAND: pip install virtualenv

Now we have to dir in command prompt Now we are moving that venv into cd

COMMAND: virtualenv venv

COMMAND: cd venv

COMMAND: dir

COMMAND: cd script

COMMAND: dir

COMMAND: Activate.bat

Now move to p y charm and write a program and run The output should be in the URL

(That means the code will come to that code can search in URL so the output can be visible to users )

**SAMPLE PROGRAMME WRITING**

from flask import Flask

app=Flask(name)

@app.route('/')

def hello world():

return 'this is python program'

if name=='main':

app. run()

If we want to debug the program use the command given below

COMMAND: app. run(debug = True)

**EXPLAINING ABOUT THE PROGRAMME:**

'/' URL is bound with hello() function. When the home page of a web server is opened in the browser, the output of this function will be rendered accordingly. The Flask application is started by calling the run() function. The method should be restarted manually for any change in the code. To overcome this, debug support is enabled to track any error

DJANGO

1. What is Django
2. Installation process
3. Django working
4. MVT
5. Explaining files
6. PROCESS OF EXECUTING A PROGRAMME IN PY CHARM
7. Difference between flask and Django
8. Conclusion of flask and Django

**What is Django:**

It is a back-end side web framework, and it is free, open source is written in Python. Django makes it easy to create web pages from Python. The latest version of Django is 4.0.3 (March 2022)

**INSTALLATION PROCESS**

NOW WE START WITH THE COMMAND PROMPT

We have to move to our local drive d

command: d:

Then we created a new folder into a drive-d

command: mkdir folder name(school)

Now create a virtual environment

command: pip install virtualenv

Next command: virtualenv venv Cd venv Scripts\activate Now again we should move back to local drive d

Here we have to install Django to that command the

command: pip install Django

Now we have to create a project for that

command: Django-admin startproject projectname(student)

Again come back to the cd student

Now we have to create an app to that

command: python manage.py startup app name(marks)

After finishing this process in the command prompt.

let us move to the p y charm project and open the project in p y charm files then open the given project that we created in the command prompt

In that project we can see the app that we created in the command prompt and in that app we can see migrations, --init--py, admin.py, apps.py, models.py, tests.py, views.py

Similarly, we can see that the project file that what we created in the command prompt, and in that we see inti.py,asgi.py,setting.py,url.py, wsgi.py, manage.py

**Django Working:**

Django works on the MVT design pattern (Model View Template).

• Now let us discuss the MVT in detail

**MVT**

**What is Model:**

It provides data from the database.

In Django, the data is delivered as an Object Relational Mapping (ORM), It is a technique designed to make it easy to work with databases.

The most common way to extract data from a database is SQL.

One problem with SQL is that you have to have a pretty good understanding of the database structure to be able to work with it.

Django, with ORM, makes it easier to communicate with the database, without having to write complex SQL statements.

The models are usually located in a file called models.py

**What are views:**

A view is a function or method that takes HTTP requests as arguments, imports the relevant model(s) finds out what data to send to the template, and returns to the final result.

The views are usually located in a file called views.py

**What are Templates:**

Templates are often .html files, with HTML code describing the layout of a web page, but they can also be in other file formats to present other results, but we will concentrate on .html files.

The templates of an application are located in a folder named templates.

**EXPLAINING ABOUT FILES**

**What is URLS:**

When a user requests a URL, Django decides which view it will send it to. This is done in a file called urls.py

**What is manage:**

A Manage is an interface through which database query operations are provided to Django models. At least one Manage exists for every model in a Django application. The Manage classes work is a document This document specifically touches on model options that depend on Manage This should be done in p y charm terminal

command: python \manage.py

**What is admin:**

The Django admin is an automatically-generated user interface for Django models. The register function is used to add models to the Django admin so that data for those models can be created, deleted, updated, and queried through the user interface.

**What are settings:**

In settings, we have to add the app name that we accessed in the command prompt

**PROCESS OF EXECUTING A PROGRAMME IN PY CHARM**

We have to open p y charm after creating a new project and new app then now click on files and open the file which you created and here we see the both app and project and in that we can see the files

Next, open the setting files and add the file name with respected strings and comma in the last

Now move to the models.py file first we have to check that the python version and check that the draft one should be clear

Then start any class program while writing a class function we use the model logic

which is the model. Model

Next, open the terminal in p y charm and now move to shell because we are connecting the database to python

Before going to the shell we move to d drive

Then now move to shell

command: py.\manage.py shell

Now we moved to shell next

command: from Django.DB import connection

Next

command: c= connection .cursor()

Now we have to exit from shell

command: exit()

Now make migration

command: py .\manage.py makemigrations

**Here python file is changing into SQL from app**

Now we should create a table so we should do migrate

Command: py .\manage.py migrate

Now to store the data we have to create the superuser

command: py .\manage.py createsuperuser

Now create any name

Email Address if u want to give then skip

Create password

Re-enter the password

Now you created a superuser successfully

After finishes, we have to run the server

command: py .\manage.py runserver

Finally, we get the URL code in output click on that URL and type admin, and enter the username, password then one page will open in which we can't see any app or projectname

Now to show the app and project move to p y charm and move to admin.py,

we have to do

from app. app name import class beside what we represent that we take here logic is:

from app. app name import name

**To see we have to register in that**

logic as

admin. site site site.register(class define name)

Run the server ctrl S

Now open the link to see the app name

Now add the objects and save

Here objects are created

But I want to see the details so again move to models.py in p y charm Now define function is called the dunder method

Logic is def –str—(self):

Then return self. Name

(what the Information was in class should be taken )

We can not get all the full details if we need full details

move to admin.py

and use the class function that is

classempolyerAdmin(admin.ModelAdmin):

List \_display =[what we want to display that we can give]

Now register here again

Admin. site. register(employee, employee admin)

Then run the server Then we get all the details in the rows and columns

**DIFFERENCES BETWEEN FLASK AND DJANGO FLASK DJANGO**

It is a lightweight framework that is commonly referred to as a micro framework.

Django is a web application framework that handles many standard features needed to create secure and maintained websites. Flask comes with a small collection of easy-to-learn comprehensive documentation.

It's a flexible framework that can be used to build any website (social network, news site, content management system, and so on) with content in HTML, XML, JSON, and other formats.

It can be used in tandem with any client-side framework.

Routing URLs is simple. End-to-end application testing is possible with Django.

The code base size is relatively smaller. Allow you to set patterns for your application's URLs.

It is easy to use for the simple case Framework for quick web development at a high level.

**CONCLUSION OF FASK AND DJANGO:**

Although Django and Flask share many fundamental concepts, Django is more complex and large, requiring a steep learning curve. Django requires more than twice as many lines of code compared to Flask Django is a production-ready framework. Each project can be a single application with numerous models and views, while the single application is in a flask.

PANDAS

1. What are pandas
2. What is panel Data
3. What is Data analysis
4. Why do we use pandas
5. What are data frames
6. What panda can do
7. Installation
8. Sample program writing
9. Explaining about program
10. Series in pandas
11. What is series
12. Sample program
13. Series with label
14. Sample program
15. Pandas in series of keys/values
16. Sample program
17. Features of pandas
18. Advantages of pandas
19. Dis advantages of pandas
20. Conclusion of pandas

**What are pandas:**

Pandas is a Python library. Pandas are used to analyze data. The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

**What is panel Data:**

Panel data, sometimes referred to as longitudinal data, contains observations about different cross sections across time. Examples of groups that may make up panel data series include countries, firms, individuals, or demographic groups.

**What is Data analysis:**

Data Analysis is the technique to collect, transform, and organize data to make future predictions, and make informed data-driven decisions

**Why we use pandas:**

Pandas are mainly used for data analysis and associated manipulation of tabular data in Data Frames. Pandas allow importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel.

**What are data frames:**

A Pandas Data Frame is a 2-dimensional data structure, like a 2-dimensional array, or a table with rows and columns.

**What pandas can do:**

Pandas give you answers about the data.

Like: • Is there a correlation between two or more columns?

• What is the average value?

• Max value?

• Min value?

Pandas are also able to delete rows that are not relevant or contain wrong values, like empty or NULL values.

This is called cleaning the data. # Installation Now first we have to create a folder

Command: mkdir folder name

Then now cd folder name

Next create a virtual environment

Command: pip install virtualenv

Next command: virtualenv venv

Next command: scripts\activate

let us install pandas To that

command: pip install pandas

Now move to p y charm and open files and open the folder that what we created in command prompt And execute a program as shown in below

**SAMPLE PROGRAMME WRITING**

import pandas as PD

data = { "calories": [420, 380, 390],

"duration": [50, 40, 45]

}

df = PD.DataFrame(data, index = ["day1", "day2", "day3"])

print(df)

HERE the output is below output:

calories duration

day1 420 50 day2 380 40 day3 390 45

**EXPLAINING ABOUT PROGRAM:**

Pandas are usually imported under the PD alias.

Create an alias with the as a keyword while importing:

Here alias means Refer to same thing Series in pandas

**What is series: A Pandas Series is like a column in a table. It is a one-dimensional array holding data of any type.**

**Sample program:**

import pandas as PD

a = [1, 7, 2]

myvar = PD.Series(a)

print(myvar)

output:

0 1

1 7

2 2

**Series with labels:**

The values are labeled with their index number The first value has an index of 0, the second value has an index of 1, etc. This label can be used to access a specified value. With the index argument, you can name your labels.

**Sample program:**

import pandas as PD

a = [1, 7, 2]

my-var = PD.Series(a, index = ["x", "y", "z"])

print(my-var)

output:

x 1

y 7

z 2

**Pandas in Series of key/values:**

Now key/value object, like a dictionary, when creating a Series.

**Sample program:**

import pandas as PD

calories = {"day1": 420, "day2": 380, "day3": 390}

my-var = PD.Series(calories)

print(my-var)

output:

day1 420

day2 380

day3 390

**FEATURES OF PANDAS:**

Handling of data.

Alignment and indexing.

Handling missing data.

Cleaning up data.

Input and output tools.

Multiple file formats supported.

Merging and joining of datasets.

A lot of time series.

**ADVANTAGES OF PANDAS:**

• It was Data representation. Pandas provide extremely streamlined forms of data representation.

• It is Less writing and more work done.

• An extensive set of features.

• It Efficiently handles large data.

• And Makes data flexible and customizable

**DISADVANTAGES OF PANDAS:**

• Steep learning curve. Pandas initially have a mild learning slope.

• It is Difficult syntax. While being a part of Python, Pandas can become tedious for syntax.

• Poor compatibility for 3D matrices.

• Bad documentation.

**CONCLUSION OF PANDAS:**

The Pandas library is an amazing tool to have in python.

This article just goes over the tip of the iceberg as to what you can accomplish with Pandas.

You can begin to see the true capabilities that Pandas have to offer when starting to work with data in python.

Pandas introduced two new types of objects for storing data that make.

It's analytical tasks easier and eliminates the need to switch tools: Series, which have a list-like structure, and Data Frames, which have a tabular structure.

Pandas serve as one of the pillar libraries of any data science workflow.

It allows you to perform processing, wrangling and munging of data.

Pandas provide a huge feature set to apply to that data you have so that you can customize, edit and pivot it according to your own will and desire.

It helps to bring the most out of your data.

We can use Pandas to perform various tasks like filtering your data according to certain states or segmenting and segregating the data according to preference, etc.

Using Pandas helps to shorten the procedure of handling data.

With the time saved, we can focus more on data analysis algorithms.