task=1

multi threading and multitasking

1st=multi tasking

multi tasking is a general word

executing several tasks simultaneously is concept of multi tasking

ex; kitchen in mother

1st in prepare to cook on one side and another side cutting on vegetables and one side calling in mobile

and preparing to lunch box

this is an so many tasks on prepare at a time

this is called multi tasking

each task is a separate independent process



task=2

multi threading

multiple independent parts of simultaneously which

known as thread

a thread is a unit of execution on concurrent programming

where each task is independent part of the same program

ex; there is an 1 program

|  |
| --- |
| Thread1 |
| Thread2 |
| Thread3 |

is 1 program in 3 threads



how many programs are there in 1 program

how many independent threads are there 3

where each task is one independent same program

there was python program in checked in line by line

but threading is divided in 3 parts so program is executed in less time

***what is thread***;

a flow of execution is considered as a thread

it is a python object

most commonly used in concept of application level

multitasking concept at operating concept level

multi threading concept is used in ;

multimedia, lofing ,animation, video games, sever implementation

3 ways of multi threading

1st;Thread with out using any class

2nd :thread by extending thread class

3rd : thread with out extending thread class

Import threading #threading .current- thread

Import threading import #current thread

def display ():

for I in range (10):

print (‘child thread’)

t= thread () # by creating thread object

t= thread (target=display) #creation of thread object to execute display

t.start ()

for I in range (10) :

print (‘main thread’)

threading is a module name

thread is a class name

module is threading

main threading

import threading

print(“current executing threading : ,threading.current-thread () .get name ()”)

there is 2 functions in : current thread (). Get name ()

*from time import sleep  
 from threading import \*  
  
 class Hello(Thread):  
 def run(self):  
 for i in range(10):  
 print('Hello')  
 sleep(1)  
  
 class Hi(Thread):  
 def run(self):  
 for i in range(10):  
 print('Hi')  
 sleep(1)  
  
 t1=Hello()  
  
 t2=Hi()  
  
 t1.start()  
 sleep(0.5)  
 t2.start()  
  
 t1.join()  
 t2.join()  
  
 print('bye'...*

*#from time import sleep  
# from threading import \*  
#  
#  
# class Book(Thread):  
# def run(self):  
# for i in range(10):  
# print('Book')  
# sleep(1)  
#  
#  
# class Pen(Thread):  
# def run(self):  
# for i in range(10):  
# print('Pen')  
# sleep(0.5)  
#  
#  
# class Pencil(Thread) :  
# def run(self):  
# for i in range(10):  
# print('Pencil')  
# sleep(1)  
#  
#  
# t1 = Book()  
# t2 = Pen()  
# t3 = Pencil()  
#  
#  
# t1.start()  
#  
# time.sleep()  
#  
# t2.start()  
#  
# t3.start()  
  
  
# t1.join()  
# t2.join()  
# t3.join()  
# print('bye')...*

**we can not exactly output but question is why means**

***threading means is concatenation is at a time. so threads are same time coming to the main*** thread

1. *Multitasking users are allowed to perform many tasks by CPU*
2. *Multithreading executes many threads simultaneously*

*This is main concept of multithreading*

*2nd task Django*

Django is a web application development of frame work

Using Django is a development complete web application by doing all the server side programming

And we run ever application as a Django server

Django is a python frame work that makes it easier to create web sites using python

*Django is a especially help full for data base driven websites*

*How does Django work*

Django follows the MVT design pattern

M=MODEL

V=VIEW

T=TEMPLATE

The model provides data from to data base

The common way to extract data from a database is SQL

The models are usually models.py

VIEW

*a view is a function or method that takes http request as arguments .*

*imports the relavent model (s) and finds out what data to send to the template and returns the finally result the view are usually view .py*

*templates*

*templates are often html files with HTML CODE*

describing the layout of a web page

but we will concentrate on html files

the templates of an application is located in a folder named templates

URLS

Django also provides a way to navigate around the different pages in a websites

When a user request a url .

Django decides which view it will send it to this is done in a file called URLS.PY

*VIRTUAL ENVIRONMENT*

It is suggested to have a dedicated virtual environment for each Django project, and one way to manage a virtual environment is [venv](https://docs.python.org/3/tutorial/venv.html" \t "_blank), which is included in Python.

With venv, you can create a virtual environment by typing this in the command prompt, remember to navigate to where you want to create your project:

Install Django ;

Finally, we can install Django.

Remember to install Django while you are in the virtual environment!

Django is installed using pip, with this command:

Windows:

(myproject) C:\Users\*Your Name*>py -m pip install Django

Unix/MacOS:

(myproject) ... $ python -m pip install Django

DJANGO PROJECT …..

Introduction..

Getting started..

Create folder …

Create project…

Create app…

Next..

To views…

Urls…

Setting

Templates

Models

TEMPLATES..

Variables

Template tags

If else

For loop

Comment

SATATIC FILES..

Add css files

Add js file

Add image

QUERY SETS ….

MODELS:

**Models we can defined in class function**

**Different types of variants in we can defined**

**This function is an python function this function changeable in SQL format**

**JSON files used in this method this called method as makemigrations**

**Makemigrations changes to the format**

**Migrate create an table format**

1. Django receives the URL, checks the URLS.PY file, and calls the view that matches the URL.
2. The view, located in VIEW.PY, checks for relevant models.
3. The models are imported from the MODEL . PY , file.
4. The view then sends the data to a specified template in the TEMPLATE folder.
5. The template contains HTML and Django tags, and with the data it returns finished HTML content back to the browser.

Django can do a lot more than this, but this is basically what you will learn in this tutorial, and are the basic steps in a simple web application made with Django.

My First Project ;

Once you have come up with a suitable name for your Django project, like mine: MYWORLD, navigate to where in the file system you want to store the code (in the virtual environment), and run this command in the command prompt:

django-admin startproject myworld

Django creates a MY WORLD folder on my computer, with this content:

myworld  
    manage.py  
    myworld/  
        \_\_init\_\_.py  
        asgi.py  
        settings.py  
        urls.py  
        wsgi.py

These are all files and folders with a specific meaning, you will learn about some of them later in this tutorial, but for now, it is more important to know that this is the location of your project, and that you can start building applications in it.

Run the Django Project

Now that you have a Django project, you can run it, and see what it looks like in a browser.

Navigate to the MY WORLD folder and execute this command in the command prompt:

py manage.py runserver

Which will produce this result:

Watching for file changes with StatReloader  
Performing system checks...  
  
System check identified no issues (0 silenced).  
  
You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.  
Run 'python manage.py migrate' to apply them.  
December 02, 2021 - 13:14:51  
Django version 3.2.9, using settings 'myworld.settings'  
Starting development server at http://127.0.0.1:8000/  
Quit the server with CTRL-BREAK.

Open a new browser window and type [127.0.0.1:8000](http://127.0.0.1:8000/) in the address bar.

The result:

**The project is run in cmd or pycharm cmd :**

**Firstly chose to the c drive d drive**

**An create virtual environment**

**For creating directory**

**For creating a folder**

**Then open folder and creating an new project**

**It means project name**

**Then creating an project app**

**Means other project name**

**This is project cmd mode process**

and next move an Django project

project files create an Folder the Folder open to the show s in



This type apps in showing

And the open in settings:

App name is ;ex; shankar ;

Shankar open to setting in

:”shankar”

And then next views :

Import I Django

From Django http import HTTP Response ;

Means in http is capital letters 2nd http

And creating in function anything

Def display (request) :

Return HTTPResoponce <h1>shankar<h2> ;

Shankar name is creating in app name

And then moved on next URLS:

Urls in import in views path

From Django urls . import path

From app to import to views

Urls pattern

Path (‘hello’/.views. hello

And next move on templates ;

Open an new directry right click on open template

2nd also open an new template

And right click to html page

And then configure

HTML to (write a logic)

And next to setting and open

Import OS.

Base DIR .

TEMPLATE. Dir (OS.path.join//base dir ‘shankar’)

Installed apps DIR template DIR

Next move on VIEWS.

Create path.

Then complete of process .

And moved on commend

Python mange.py run server

Then open the code

First project is complete

MIDDLE WARE OF DJANGO :

**Middle ware is a frameworks of hooks**

**Django request / response processing**

**It s light ,low-level “plugin” system for globally altering Django input or output**

**Each middle ware component is responsble for doing some specific functions for example Django includes a middle ware component** *AUTHENTICATION MIDDLE WARE*

WHAT IS GET METHOD IN DJANGO:

***Get is an HTTP method in Django that encapsulates the data in a string and utilizes it to construct a URL . the URL includes the data keys and values as well as the address to which the data must be transmitted***

***WHAT IS GET METHOD USED FOR :***

*Get is used to request to data from a specified resources . some notes on get request:*

*Get request can be cached . get request remain in the browser history*

***What is request in Django:***

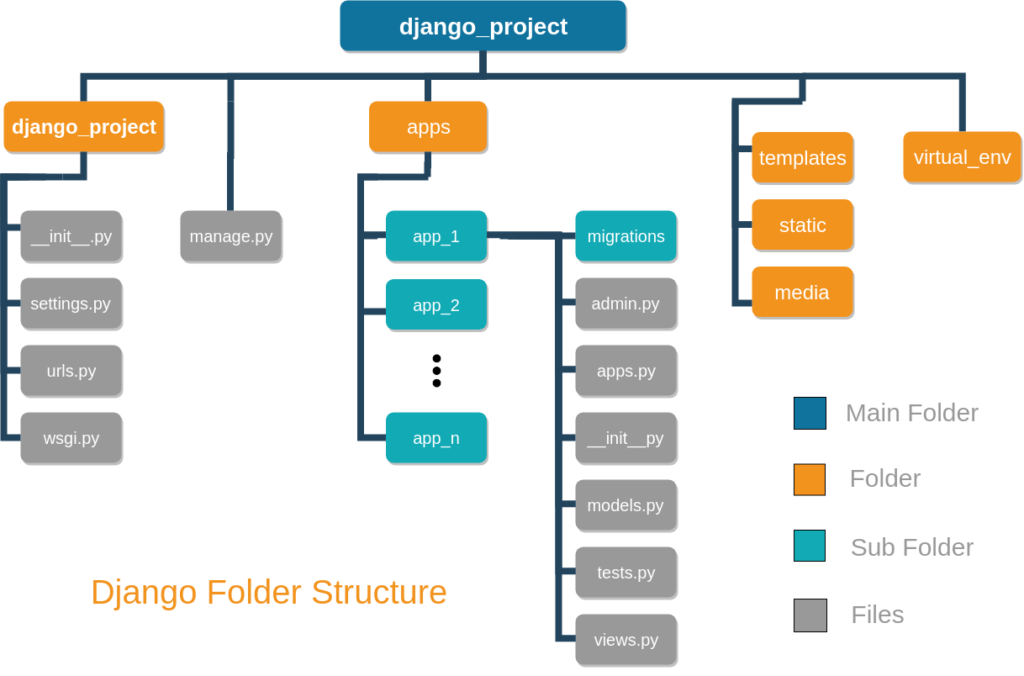
you will learn how to get data from get request in Django . when you send a request to the server , you can also send some parameters . generally ,we use a get request to get some data from the server . we can send parameters . with the request to get some specific data.

What is get and POST method in Django:

Get and post:

Django login from is a returned using the post method . in which the browser bundles up the form data , encodes it for transmission , sends it to the server , and then receives back its response , GET by contrast ,bundles the submitted data into a string . and uses this to compose a URL

DJANGO PROJECT STRUCTURE :



.

├── apps

│   └── app\_1

│       ├── admin.py

│       ├── apps.py

│       ├── \_\_init\_\_.py

│       ├── migrations

│       │ └── \_\_init\_\_.py

│       ├── models.py

│       ├── tests.py

│       └── views.py

├── django\_project

│   ├── \_\_init\_\_.py

│   ├── \_\_pycache\_\_

│   │ ├── \_\_init\_\_.cpython-35.pyc

│   │ └── settings.cpython-35.pyc

│   ├── settings.py

│   ├── urls.py

│   └── wsgi.py

├── manage.py

├── media

├── static

└── templates

MANAGE .PY

***This file is used basically as a command line utility and for deploying , debugging, or running our web application***

***This file is contains code for runserver ,or makemigrations or migrations ,etc that we use in the shell . anyway , we do not need to make any changes to the file.***

***\*RUNSERVER :***

**THISis command is used to run the server for our web application**

**MIGRATION:**

**This is used for applying the changes done to our models into the data bases.**

**That is if make any changes to our database then we use migrate command**

**This is used for first time we create database**

**MAKEMIGRATION:**

**This is code done to apply new migrations that have been carried out due to the changes in the databases**

WHAT IS DEBUG TRUE AND FALSE IN DJANGO:

**I first assumed DBUG=TRUE turned on django s logging capability and middleware for error reporting but now understanding how the django intrenal system work differently under the boolen settings**

Functionally, there are no differences. However, DEBUG defines whether the error message should be shown to the user at the browser level (DEBUG=True) v/s send an email to admins (DEBUG=False with some settings.)

FILTER AND GET DIFFERENCE IN DJANGO:

Get is an only 1 value returned

And get value is 0 and multiple values in there an result is ERROR

FILTER is an 0 and multiple values (2,3,4) is an returned

**SERIALIZATION :**

What is meant by serilization:

Serilization is the process of converting a data object of a combination of code ana data represented with in a region of data storage

The process of converting in object .from python supporter form to file support form is this called in seriliazation

**DE SERILIZATION :**



*Deserilization means the file supported form converted in to the python supported form this called de serilization*

*This is reverse process to serilization*

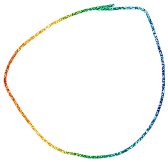
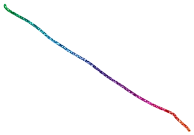
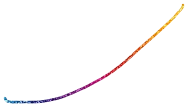
*Employe:number=100 === this is EMPLOYE OBJECT*

*Employe : name=ram*

*Employe: salry=1000*

*Employe : id =112*

**

write==== serilization & marshiling

pickling

THIS IS OBJECT FILE FORM OR NETWORK SUPPORT OBJECT

THIS IS PYTHON OBJECT READ == DESERILIZATION & UNMARSHILING ==

UNPICKLING

**PYTHON OBJECT IS NOT CONVERTING IN JAVA APP AND MOBILE APP**

*Python object is not which can not understand by java application and os applications*

*We required convert files in python files to java files is an used to json, yaml, files*

*Json means java script object notesion*

*JSON advantage to understand java , mobile, applications*

**What is JSON Django:**

***in this article . we will see how to add JSON fields to our Django models . JSON is a simple format to store data in key and value format.it is written in curly braces . many a time on developer website , we need to add developer data and JSON fields are useful in such cases . first create a Django project and an app***

***python object form to JSON object form converted into a called as SERILIZATION***

***JSON object form to python object form converted to its called as DESERILIZATION***

***PICKLE:***

***Pickle is a module***

***Pickle in python is primarity used in serializing and desrializing a python object structure in other words it s the process of converting a python object into a byte stream to store it in a file /database . maintain program state across session , or transport data over the network***

***Pickle function we can perform serilization***

***By using function in DUMP():***

***Which using function we can deserilization by using function in LOAD():***

***We known as unpickling***

***E=employe (100,’ram’,10000,’hyd’)***

***With open (‘emp.ser’ ‘wb’) as f:# writing binary***

***Pickle.dump(e,f) #serilization***

***With open (‘emp.ser’,’rb’) as f :***

***Emo-oby = pickle.load (f)***

***Binary data=not readble form***

***Import pickle***

***CLASS EMPLOYE :***

*Def \_\_init \_\_ (self, eno ,ename, esal,eaddr):*

*Self . eno = eno*

*Self . ename = ename #instance variables*

*Self . esal = esal*

*Self . eaddr = eaddr*

*Def display (self):*

*Print (‘E NO : { } , E NAME : { } ,E SAL : { }, E ADDR : { } ‘*

*. FORMAT ( self. E no , self . ename ,self . esal , self. eaddr))*

*E = employee ( 100 , ‘ ram ‘ , 10000 , ‘ hyd ‘ )*

*With open (‘emp . ser ,’ wb ‘) as f :*

*Pickle . dump ( e, f )*

*Print ( ‘ pickling of employee object completed ‘ )*