**DJANGO DOCUMENTATION**

1. Open visual studio code
2. Go to terminal and create a dir and go to that dir : mkdir portfolio. cd portfolio.
3. Create a virtual environment with the command: virtualenv env(name of the venv). It will install all require package for the project.
4. Create a project : django-admin startproject portfolio.(project name)give a (.) which will create the project in the same dir.
5. Check it is working: python manage.py runserver. Go to 127.0.0.1/8000 and it will show congrats.
6. Few migrations are left. Django automatically create sqlite3 db.
7. If we work with sqlite3 the make migrations, if not then at first create new db then migrate
8. Python manage.py makemigrations
9. Python manage.py migrate.
10. Now create a new app/apps: python manage.py startapp sadat
11. A new app/apps will be created under portfolio dir.
12. We have created two apps named sadat and ashik
13. Incl the app/apps name in project’s settings.py file which is portfolio.
14. INSTALLED\_APPS = [
15. 'django.contrib.admin',
16. 'django.contrib.auth',
17. 'django.contrib.contenttypes',
18. 'django.contrib.sessions',
19. 'django.contrib.messages',
20. 'django.contrib.staticfiles',
21. 'sadat',
22. 'ashik',
23. #'sadat.apps.SadatConfig',
24. ]
25. Now configure the project’s url.py file to link the newly added app.
26. Path(‘admin/’, admin.site.urls), is default path for admin panel.
27. import include function like: from django.urls import path, include
28. Then in urlpatterns include the urls of new app/apps (sadat, ashik).
29. from django.contrib import admin
30. from django.urls import path, include
31. urlpatterns = [
32. path('admin/', admin.site.urls),
33. path('', include('sadat.urls')),# if blank, it will go to sadat url
34. path('', include('ashik.urls')),
35. # path('home/', include('sadat.urls')),# if /home/ , it will go to sadat url as it is created under sadat app
36. ]
37. This url will redirect the request to app url as per request:sadat.info/about/
38. There is no urls file in newly created app. So create a urls.py file under newly created app(sadat)
39. Remember the fol while writing url path
40. Function views
41. 1. Add an import:  from my\_app import views
42. 2. Add a URL to urlpatterns:  path('', views.home, name='home')
43. Class-based views
44. 1. Add an import:  from other\_app.views import Home
45. 2. Add a URL to urlpatterns:  path('', Home.as\_view(), name='home')
46. Including another URLconf
47. 1. Import the include() function: from django.urls import include, path
48. 2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
49. New app urls coding should be as fol:
50. from django.urls import path
51. from sadat import views
52. # from . import views
53. urlpatterns = [
54. path('', views.home, name='home'),
55. path('home/', views.home, name='home'),
56. path('about/', views.about),
57. path('example/', views.example),
58. # path('', views.home.as\_view(), name='home')
59. ]
60. Here . import means importing from same dir.
61. Inside the path link the function name with views which will be created inside view.py
62. Inside views.py create a function and the code should be fol
63. from django.shortcuts import render
64. #from django.http import HttpResponse
65. def home(request):
66. #return HttpResponse("Welcome to sadats home")
67. #return render,('index.html') #### this will also work========
68. return render(request, "index.html",{})
69. def about(request):
70. return render(request, "about.html",{})
71. def example(request):
72. return render(request, "example.html",{})
73. Each func will be created to perform some task: like calling index.html page

Templates file set up

1. Create a folder name templates in project dir where all templates will be stored in that folder or in app sub folder under this dir. It is better.
2. Put the templates ie; index.html inside templates folter
3. Note: But the index.html file can also be kept in the same app dir templates folder.
4. In settings.py create a variable for showing templates dir
5. TEMPLATE\_DIR = os.path.join(BASE\_DIR,”templates”) == this shows the templates folder of base dir/ project dir.
6. import os
7. # Build paths inside the project like this: os.path.join(BASE\_DIR, ...)
8. BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))
9. # TEMPLATES\_DIRS = os.path.join(BASE\_DIR, "templates/sadat\_templates")
10. # TEMPLATES\_DIR = os.path.join(BASE\_DIR, "templates")
11. Now put the variable in template list which will be put under a dict in setteings.py
12. TEMPLATES = [
13. {
14. 'BACKEND': 'django.template.backends.django.DjangoTemplates',
15. # 'DIRS': [],
16. # 'DIRS': [ TEMPLATES\_DIR ],
17. # 'DIRS': [ TEMPLATES\_DIRS ],
18. 'DIRS': ["templates"],
19. # 'DIRS': [os.path.join(BASE\_DIR, "templates")],
20. # 'DIRS': [os.path.join(BASE\_DIR, "templates/sadat\_templates")], #- check it works or not
21. 'APP\_DIRS': True,
22. Templates dir can be shown directly without variable .
23. 'DIRS': [],
24. Will bring templates from app templates
25. 'DIRS': ["templates"], everything ok
26. Or,
27. 'DIRS': [os.path.join(BASE\_DIR, "templates")], everything ok
28. Or
29. 'DIRS': [TEMPLATES\_DIR ], - here static files are not getting loaded
30. Will bring templates folder or sub folder from base dir

Static Files

1. Create a directory in project directory named static.
2. Create a variable in settings.py file to show the path of base dir
3. STATIC\_DIR = os.path.join(BASE\_DIR, "static")
4. At the bottom of settings.py create another list variable named STATICFILES\_DIRS and put the variable name as list ending with coma(,).
5. STATIC\_URL = '/static/'
6. STATICFILES\_DIRS = [STATIC\_DIR,]
7. Incl {% load static %} in the html file. It should be after <iDOCTYPE> , or may create some problem.
8. <!DOCTYPE html>
9. {% load static %}
10. In href= show the pathe of static file in base dir within {% static %}
11. <link rel="stylesheet" href="{% static 'css/main.css' %}">
12. <!-- <img src=" {% static 'images/django.jpg'%} " alt="something went wrong"> -->
13. <img src="{% static "images/django2.jpg" %} " alt="both are double quotes">
14. So far you are good to go.

**DJANGO MODEL**

1. Create class inside models.py
2. Each attributes of class represent a field which will act as column
3. In database like any SQL is like a giant table, in which with each column representing a field, and each row representing an entry.

|  |  |  |
| --- | --- | --- |
| Website ID | Website Name | Website URL |
| 1 | Google | www.google.com |
| 2 | Facebook | www.facebook.com |

1. Here website ID is int field, website name is char field and website URL is URL field.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Website ID | Website Name | Website URL |  | Website ID | Data accessed |
| 1 | Google | www.google.com | 1 |  |
| 2 | Facebook | www.facebook.com | 2 |  |

1. Here website ID is primary key in left table and foreign key in the right table
2. This primary key and foreign key is used to create relationship in database.