1) **Start project**

Command: django-admin startproject project1

Cd project1

2) start application

Command: py manage.py startapp testapp

3) create “templates” folder at project level

4) in “templates” folder, create an application “testapp”

5) in “testapp” folder, create a html file “results.html”

6) Similar to templates, create a folder “static” for static content at project level.

7) in “static” folder,

create a folder “css”, in which create some .css files

create a folder “images”, in which create some .img files

8) immediately go for “Settings.py”, at project level.

In settings.py,

Add Application in INSTALLED\_APPS list

* INSTALLED\_APPS = [testapp,]

Add template folder path (for html code)

* + - TEMPLATE\_DIR = os.path.join (BASE\_DIR, ‘templates’)
    - DIRS = [ TEMPLATE\_DIR]

Note: BASE\_DIR points till project level(main/outer)

Add static folder path (for static contents)

* + - STATIC\_DIR = os.path.join (BASE\_DIR, ‘static’)
    - After line STATIC\_URL = ‘/static/’ (at bottom of file)
      * Add STATICFILES\_DIRS=[STATIC\_DIR,]

9) Go to views.py, define your views and define its html code in results.html under templates file.

9) for any links to static content (like images, css stylesheets) from html file, define such info in corresponding img or css section under static file.

10) go to urls.py (preferably at app level), add url patterns

11) Bootstrap CDN link (v 4.4): include this link in html code under <head> tag, in order to use pre-defined CSS styles.

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js" integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6" crossorigin="anonymous"></script>

12) run server to execute: py manage.py run server

13) Tree structure of project look like:

Project1

templates

testapp

results.html

static

css

abc.css

images

xyz.jpg

Project1

Default\_files(including settings.py and urls.py)

testapp

urls.py

views.py

14)

Html: deals with content (text, image, video) of a web page

Css: deals with visual appeal of a web page

15) To include static file in .html file,

Add this line after <!DOCTYPE html>,

{%load staticfiles%}

And if link to custom image has to included:

<img src = ”{%static “images/sunny.jpg” %}”>

If link to custom css has to be included: under <head> tag

<link rel=”stylesheet” href=”{%static “css/dnews.css” %}”>

If Bootstrap link has to be added, add it under <head> tag as

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js" integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6" crossorigin="anonymous"></script>

16) Models: for database and its operations

* Default is Sqllite(enough for small to mid level complexity applications)
* We can configure other databases in settings.py, if required like oracle db, Mysql, etc
* To check whether a sqllite or any other db is properly configured/installed or not, try py manage.py shell, should give some output like python shell.then check for
  + from Django.db import connection
  + c = connection.cursor()

If they work properly, it will be properly installed.

* We are not required to write single line of SQL query in Django,
* Once we define our class in models.py, that python code in class will be converted to SQL code using
* py manage.py makemigrations – creates a folder 0001\_initials.py file in migrations folder
* Note: We can directly write sql code anr create tables for it, but we are using migrate options without specifying any specific application, because we want other tables of other in-built applications(see in Installed\_APPS in stetting.py) to be created tables or have refresh for them.
* py manage.py sqlmigration testapp 0001 – creates sql code for the python code in above class.
* Py manage.py migrate – applies migrate for all applications.
* Now our database table will be ready.
* Admin – is one of beautiful in-built application provided by Django.
* How to launch admin interface?
  + In the admin interface we can check our created table we can perform CRUD(Create, Read, Update & Delete) operations on that table.
  + To send request to admin interface, 127.0.0.1:8000/admin
  + Admin will ask username and password, for which we have to first create a superuser.
  + Py manage.py createsuperuser
  + (Username- email can be given)
* Now to see the table in admin, we have to first register table in admin interface.
* Go to admin.py in testapp,
  + From testapp.models import Employee
  + #register your models here
  + Admin.site.register(Employee)
* Now your EMployeee table will appear in admin page.
* We have to create Model by inheriting model.Model class inside models.py, without which no table will be their
* Register your model in admin.py by Admin.site.register(Employee)