

Django Introduction

What is Django

- Django is a MVC Framework in Python which is mainly used for Web Development and it is single page application
- It's a High Level Framework, which gives more maintainable and secure websites
- First version of Django released in sept-2008, now we are in 3rd version of Django
- First Project for Django was Newspaper Website after creation of multiple pages and application Django came as a open source framework

Why to Choose Django?

- Batteries Included → it gives the complete end to end configuration like DB configuration, Front end configuration, and it controls the flow of the application
- Versatile → you can able to create any kind of application by using Django
- Secure → "do the right things" which is used to avoid the common security issues including SQL injection, cross-site scripting, and cross-site request forgery
- Maintainable → it is easy to maintain the website, it is based on DRY (Don't Repeat Yourself) Principle which increases the Reusability
- Portable → It is platform oriented and it support with many website hosting

Applications by Django

- Instagram
- Mozilla
- BitBucket
- YouTube using Django
- Spotify
- Google Person Finder
- NASA
- Washington post

Django vs Flask

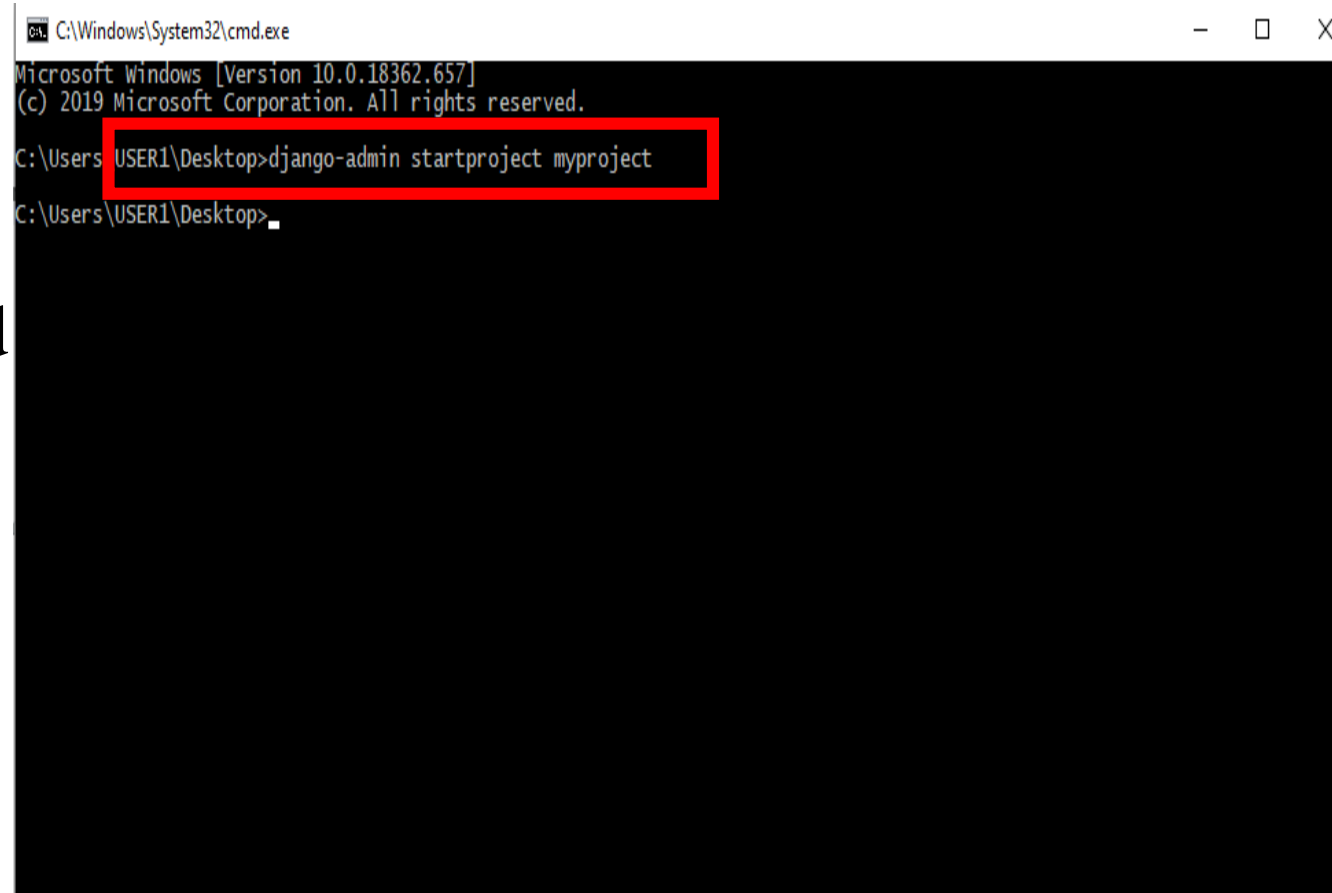
Django	Flask
Full Stack web Framework	Micro Framework
Batteries Include Principle	WSGI Web Server Gateway Interface
Suitable for Complex web application	Suitable for simpler application
It have limited flexibility	It gives full flexible
Little Difficult for beginners	It is easier
IT has built in ORM, so it supports all the Database	It used with SQLAlchemy
It work based on DRY	We need to specified everything

Installation

- The basic setup consists of installing
 - Python,
 - Django → `pip install django`

New Project Setup

- Lets we handle everything globally, create a new Project in Django in the desired root folder
- *django-admin startproject Projectname*
- We need to run the above command



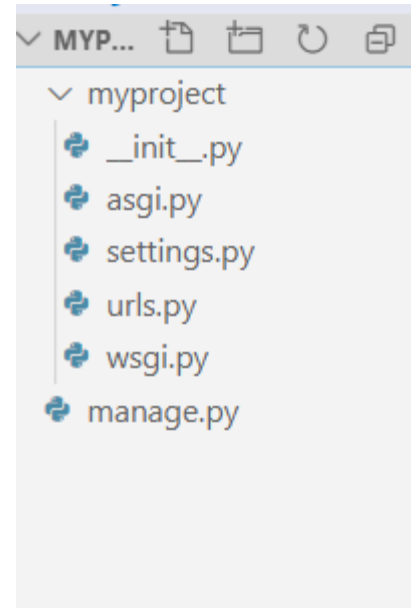
```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18362.657]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\USER1\Desktop>django-admin startproject myproject

C:\Users\USER1\Desktop>
```

Folder structure

- After we run the command above, it will generate the base folder structure for a Django project like the following image



Settings.py

- This is mainly used to handle the Data base, Time zone, Internalization, and Template Routing
- This is used to configure all the setting

URL's.py

- This is mainly used for Routing purpose, here we are defining the path and the function
- The path method where its used for URL Routing

```
from django.contrib import admin
from django.urls import path

urlpatterns = [
    path('admin/', admin.site.urls),
]
```

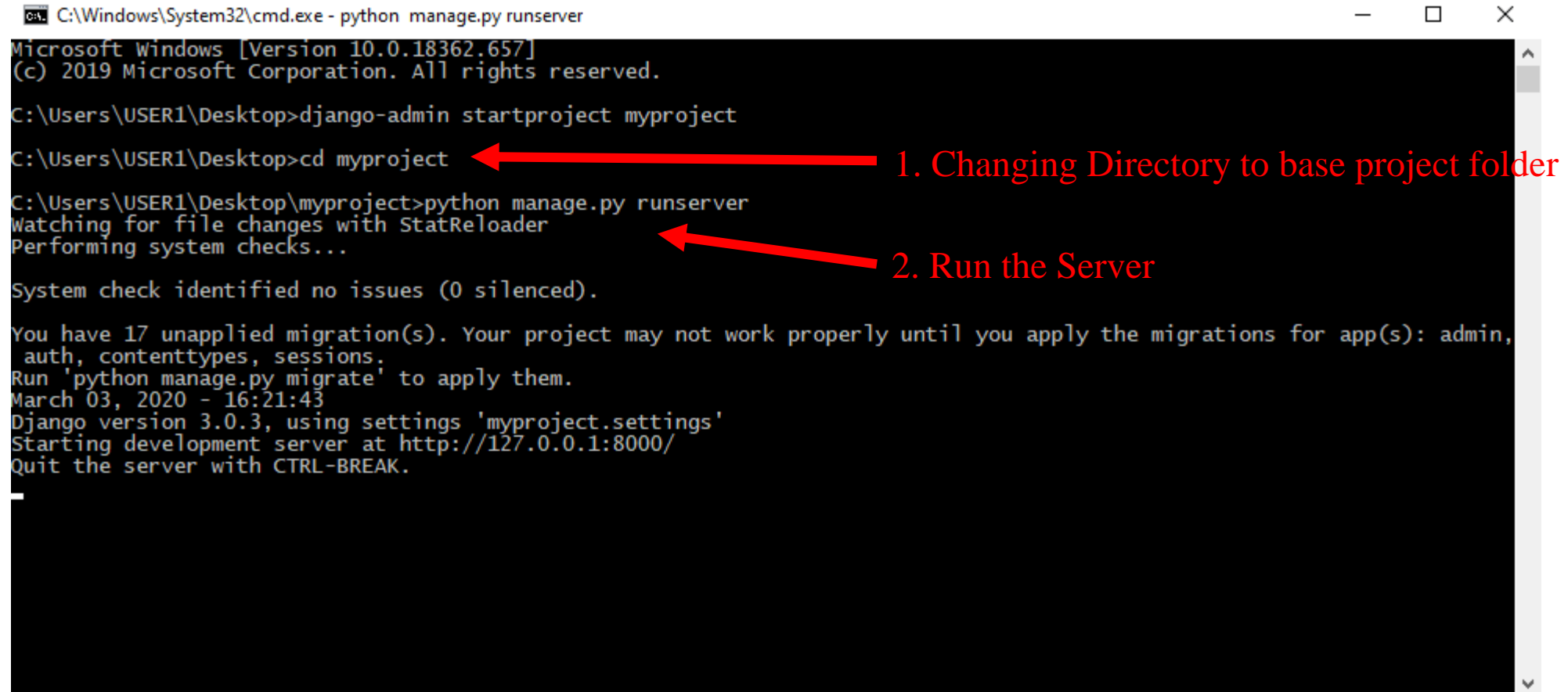
ASGI and WSGI

- Both files used to give the interface to the python application and the webserver
- WSGI for **Backward compatible** which means it is used to make interface with earlier version or other system
- ASGI which is used for asynchronous and synchronous apps interface
 - Synchronous Apps it handle by external commands (Example: Contact Form in Web Application)
 - Asynchronous Apps: It enable both client and server to react simultaneously (Example: Facebook, Gmail)

To Run a Server

- To run the Django server we need to follow the following Steps
 1. Changing the Root directory to the manage.py path in cmd
 2. Now run python manage.py runserver command in cmd
 3. If there is any error in application it gives the error else it start to run in the port number 8000
 4. When we open browser and paste the default host number the default page of Django will be running

Running a Server



```
C:\Windows\System32\cmd.exe - python manage.py runserver
Microsoft Windows [Version 10.0.18362.657]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\USER1\Desktop>django-admin startproject myproject

C:\Users\USER1\Desktop>cd myproject
C:\Users\USER1\Desktop\myproject>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

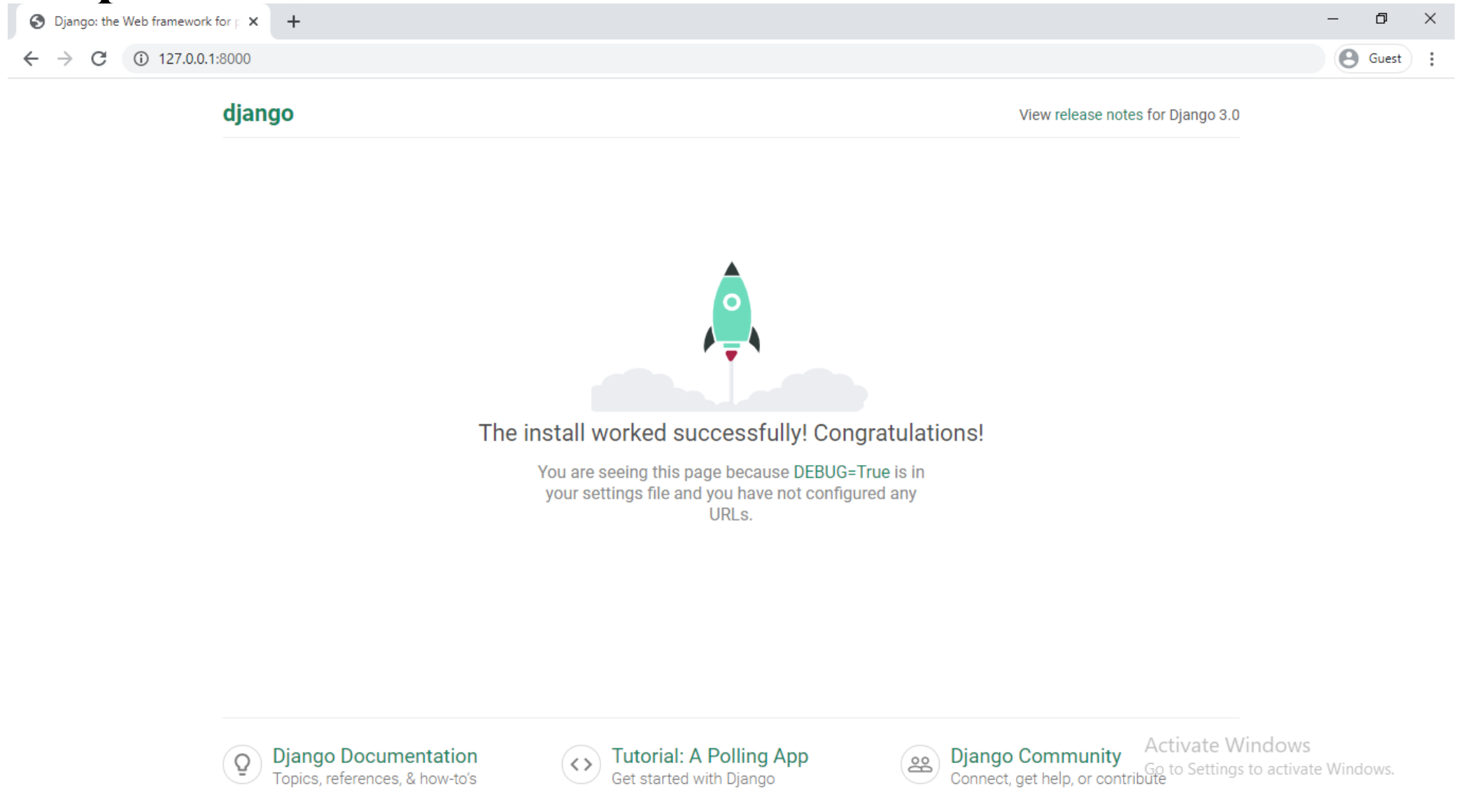
System check identified no issues (0 silenced).

You have 17 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.
March 03, 2020 - 16:21:43
Django version 3.0.3, using settings 'myproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

1. Changing Directory to base project folder

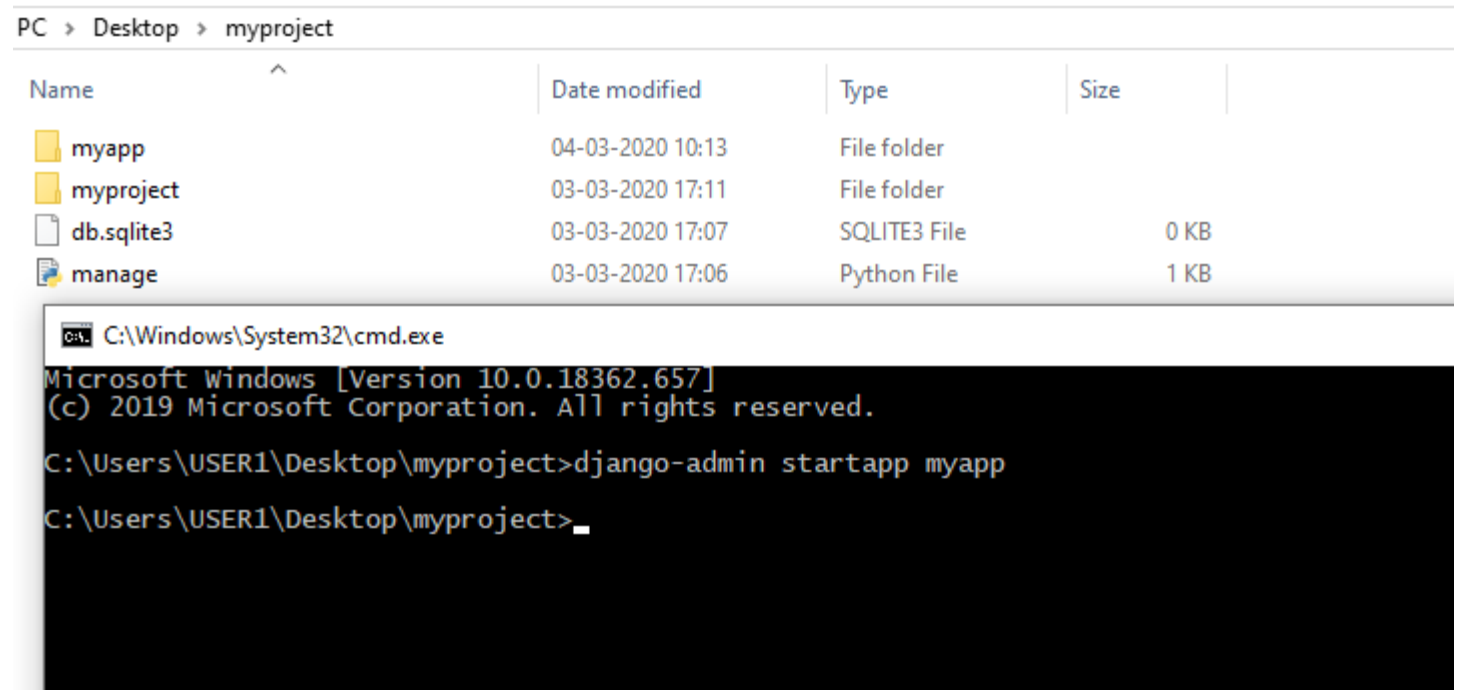
2. Run the Server

The Output



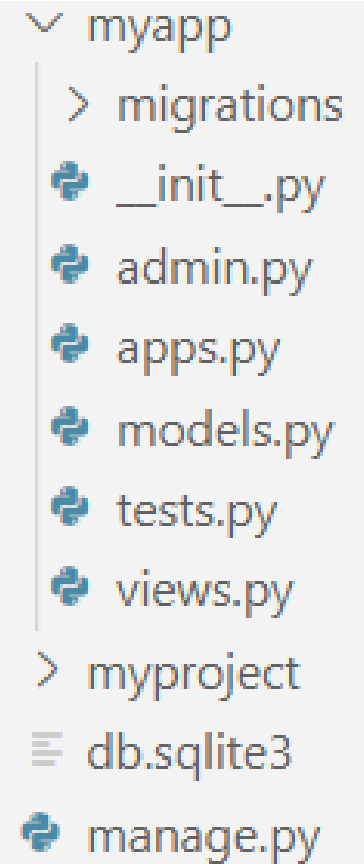
Start a App

- Go to the directory where manage.py file is there (this step is advisable)
- `django-admin startapp Applicationname`
- So Application created



Folder Structure

- migrations/: used to handle the changes which we are making in the Database (models.py is used to handle the DB)
- admin.py: Django has its in-build option called admin page this file is used to handle the admin
- apps.py: used to configure the own app
- models.py: used to write database queries in python
- tests.py: used to write test cases for your application
- views.py: where actually handling the request/response and the client view



```
▼ myapp
  > migrations
  🌀 __init__.py
  🌀 admin.py
  🌀 apps.py
  🌀 models.py
  🌀 tests.py
  🌀 views.py
  > myproject
  ≡ db.sqlite3
  🌀 manage.py
```

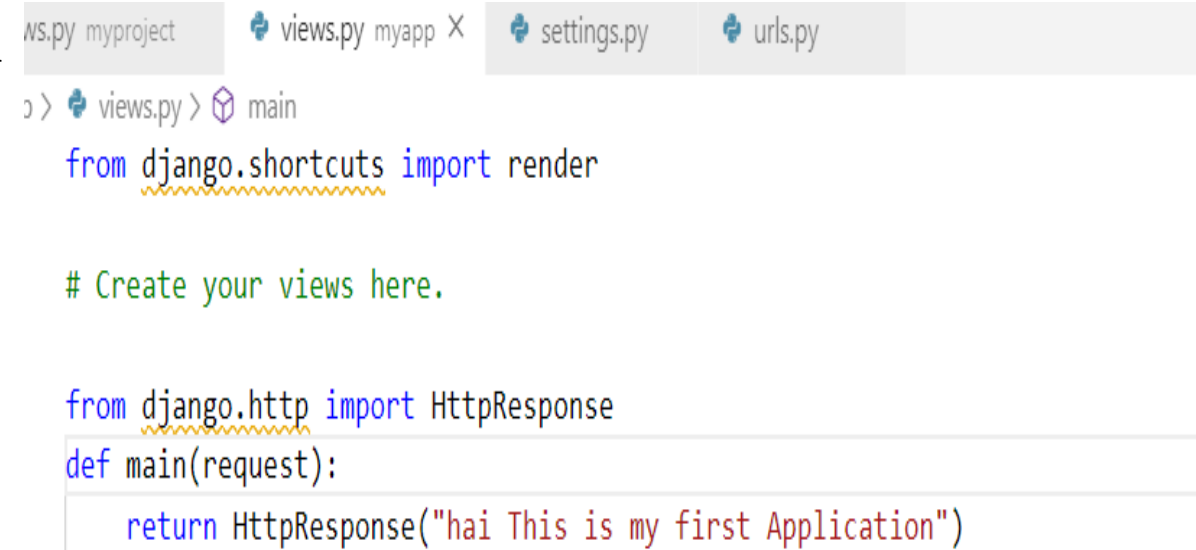
A screenshot of a file explorer window showing the Django project structure. The 'myapp' folder is expanded, showing subfolders 'migrations' and 'myproject', and files '__init__.py', 'admin.py', 'apps.py', 'models.py', 'tests.py', and 'views.py'. The 'myproject' folder is also expanded, showing 'db.sqlite3' and 'manage.py'.

Writing a First View in Application

Return Hello World

Views.py in Application

- Just writing the Hello world Http Response in Django Application views
- Open your views.py in Application and write the following views
- Now we wrote the views we need to configure this application with the Project



The screenshot shows a code editor with several tabs at the top: 'ws.py myproject', 'views.py myapp' (which is the active tab), 'settings.py', and 'urls.py'. The 'views.py' tab is selected, and the editor shows the following Python code:

```
views.py > main
from django.shortcuts import render

# Create your views here.

from django.http import HttpResponse
def main(request):
    return HttpResponse("hai This is my first Application")
```

Settings.py

```
# Application definition
```

```
INSTALLED_APPS: list
```

```
INSTALLED_APPS = [
```

```
    'django.contrib.admin',
```

```
    'django.contrib.auth',
```

```
    'django.contrib.contenttypes',
```

```
    'django.contrib.sessions',
```

```
    'django.contrib.messages',
```

```
    'django.contrib.staticfiles',
```

```
]
```

- In Settings installed apps mainly used for super user in Django
- Django.contrib.admin → It is used to handle the Admin Page
- Django.contrib.contenttypes → Used to track the information in the models installed in your project
- Django.contrib.session → Session used to track the activity, how much time the particular user logged in
- django.contrib.messages → used for Admin Interface

Configure Application with Project

- For that we need to *open settings.py* and installed apps list which is in the file
- We need to append our Application name in to the list of pre defined application in the installed_apps
- Now in this list we need to write our application name by using comma like the given way:

```
myproject > settings.py >
27
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'myapp'
41 ]
42
```

Routing

- After Configuration of Settings we need to give the path where to get the particular function in the application
- open the urls.py
- There is 3 different ways to configure with urls
 - Function based Method
 - Class based Method
 - Include Method
- In this example we are going to discuss about function based method

Urls.py script

- We need to import the views in the urls
- In path we need to give three arguments
 - `path('url', function_name , name_parament)`
 - *First argument* is actual path in the urls
 - *Second argument* is the function name
 - *Third argument* is name parameter which is used for future reference, this is actually a good practice to handle the templates this name parameter will help us for reference
- As we give the empty string in first parameter it represent the home page

```
from django.contrib import admin
from django.urls import path
from myapp import views
from myproject.views import main,time

urlpatterns = [
    path('admin/', admin.site.urls),
    path('some',main),
    path('time',time),
    path('',views.main,name="main")
]
```

Good Job