In this Lesson we are going to learn how to create projects using Django

Django is a high level web framework for python that allows developers to build web applications quickly and efficiently

Requirements

You have to install Django where you will start a project and an application

Here are the steps to follow,the steps can only be used by someone using windows

Step1

You first have to configure a virtual environment and name it with the command

**Virtualenv --name of the virtual environment e.g venv**

Step2

You have to activate the virtual environment using the command

**Venv\scripts\activate**

Then to deactivate you write the command

**Venv\scripts\deactivate.bat**

Step3

Then you install django using the command

Pip install django==version e.g 3.0.8 which is the latest version used in this time and older versions of django are recommended because they are more stable than the new ones

Step4

Then you create a project with the command

**Django-admin startproject (project name)**

Step5

Then you migrate with the command

**Python manage.py migrate**

Step6

Then you create a superuser with the command

**Python manage.py createsuperuser**

Step7

You start an application with the command

**Python manage.py startapp (Application name)**

Now let us talk about the files that come when you install django

Django provides a set of tools and libraries to handle common web development tasks, such as handling URLS, managing database, handling forms, and handling user authentication.

When you create a Django project, it generates several files and folders that are essential for the proper functioning of the project. Here is a brief overview of some important files and folders that come when you create a django project

Manage.py: This is a command-line utility that allows you to interact with the project. You it for various like running the development server, creating database tables and many more

NB: The manage.py file MUST not be edited

*Settings.py*: This file contains all the project settings and configurations. You can define database settings, installed apps, middleware, timezone, static files configurations.

Urls.py: This file is responsible for mapping URLs to corresponding views . It contains a list of Urls pattern and their corresponding views

Views.py: Views are functions that handle HTTP requests and return HTTP responses. This file contains the views function that handles various parts of your web application’s functionality

Models.py: Models are python classes that represent database tables and encapsulate the data logic of you application. This file is where you define database model

Static/: This is a folder that store static files like CSS, images and JavaScript

Templates: This stores HTML files

Admin.py: This file is used to register your models with Django admin interface, allowing you to manage data web base administrative way

**STEPS TO FOLLOW WHEN ON A DJANGO PROJECT**

When you have started an application, you have to open another URLs file where you will create a path e.g if you are creating a path for an index file

Path (‘’,views.index, name=’index’),

This is done for every file that you have opened in the template folder

You have to add a directory for the templates and static folders in the settings.py file and the name of the application is registered in the settings file under the installed application

In the application you then add the forms and filter files and then you import them from the models

Like this from django.forms import ModelForms

The filters enable searching of the pages in the application

You import filters in django and creat a field for searching

The forms handles what is displayed to the users

Let us talk about the models

In the model.py file,you define your data models as python classes that inherit from django’s models.model class.Each class attribute represents a field in the database tables.You have to first import

From django.db import

* After defining the models, you need to create database tables based on these models. Django provides a migration system that tracks changes in models.py and generates migration files. To apply these changes to the database, you run the python manage.py migrate command. This process creates or modifies the database tables according to the current state of your models.
* By registering your models in the admin.py file, you enable Django's built-in admin interface to manage your data. The admin interface allows authorized users to perform operations on the data through a user-friendly web-based interface, without writing custom views or forms.
* As your project evolves, you might need to modify your data models to accommodate new requirements. In such cases, you create new migrations using the python manage.py makemigrations command and apply them with python manage.py migrate. The migration system handles the changes you make in the models file at all times