


# **FILES AND FOLDERS IN DJANGO**



Explain the files of the project and app folders?

## Files in the Project Folder

### Introduction

When you create a Django project for the first time using the command, `django-admin startproject<project_name>` there are several files and directories generated by the Django framework to help you get started with your project. These files and directories provide the basic structure and configuration for your Django project. Here's a brief explanation of some of the key files and directories you'll encounter in a newly created Django project:

**1. `manage.py`:** This is a command-line utility that lets you interact with your Django project. Use to run development servers, create database migrations, and perform various other tasks related to managing your Django project.

**2. `__init__.py`:** This file is present in every Python package (directory with Python files) to indicate that the directory should be treated as a Python package.

**3. `asgi.py`:** ASGI stands for Asynchronous Server Gateway Interface. This file provides an ASGI-compatible interface for asynchronous servers, such as Daphne, which is used with Django Channels for handling WebSockets and other asynchronous protocols.

**4. `settings.py`:** The `settings.py` file is typically used to store configuration information in Django. It may contain the site URL, paths to various directories, and other values that the executable code may use. By changing and supplementing this file, the developers configure Django projects.

**5. `urls.py`:** This file contains URL patterns for your Django project. It maps URL paths to views, which are Python functions responsible for processing the requests and returning responses.

**6. wsgi.py:** Web Server Gateway Interface. Like asgi.py, files are used for deploying your Django project to production servers. It provides a WSGI-compatible interface for web servers to interact with your Django application.

These are some of the essential files and directories you'll find in a new Django project. As you develop your project further, you'll likely create additional files and directories to organize your code and resources effectively.

## Files in the App Folder

### Introduction

In a Django project, each app typically consists of several files and directories. When you create a new Django app using the `django-admin startapp <app_name>` command, it generates a set of files and directories for that app. Here's a brief explanation of some of the main files and directories you'll find in a Django app at the time of creation:

**1. migrations:** This directory contains database migration files generated by Django's migration system. Migrations are used to manage changes to your app's database schema over time, allowing you to easily update the database as your models change.

**2. \_\_init\_\_.py:** This file is required for Python to recognize the directory as a package. It can be left empty.

**3. admin.py:** This file is used to register models to the Django admin interface, allowing you to manage your app's data through a web interface.

**4. apps.py:** This file defines the configuration for the app, such as its name and any configuration options. It is where you can customize certain app-level settings.

**5. models.py:** This file is where you define the data models for your app using Django's ORM (Object-Relational Mapping) system. Models define the structure of your database tables and the relationships between them.

**6. tests.py:** This file is where you can write unit tests for your app to ensure its functionality behaves as expected. Django provides a testing framework that makes it easy to write and run tests for your app.

**7. views.py:** This file contains the views, which are Python functions or classes that handle HTTP requests and return HTTP responses. Views typically interact with models to retrieve or manipulate data before rendering it to a template or returning it as JSON.

These files and directories provide the basic structure and functionality for a Django app, allowing you to define models, views, URLs, and templates to build a web application. Over time, you may add additional files or directories as your app grows and evolves.