Step 1: What is Django? (Simple Explanation)

- Django is a **Python framework** used to build websites.
- It helps developers create web apps **quickly and efficiently**.
- It follows the **MVT (Model-View-Template) architecture**, making it easy to organize code.

Example: Think of Django as a "cooking recipe" for making websites. Instead of doing everything from scratch, we follow Django's recipe to save time and effort.

📌 Step 2: Installing Django

Before we start coding, we need to install Django.

☑ Install Python (If Not Installed)

Make sure Python is installed. Check by running:

```
bash
CopyEdit
python --version
```

If not installed, download it from python.org.

☑ Install Django

Now, install Django using pip:

```
bash
CopyEdit
pip install django
```

Check if Django is installed:

```
bash
CopyEdit
django-admin --version
```

📌 Step 3: Create a Django Project

A Django **project** is like a full website, and an **app** is like a small part of the website (e.g., login system).

☑ Create a New Project

Run the following command in your terminal:

```
bash
CopyEdit
django-admin startproject mywebsite
```

Navigate into the project folder:

```
bash
CopyEdit
cd mywebsite
```

Start the development server:

```
bash
CopyEdit
python manage.py runserver
```

Now, open your browser and go to:
👉 **http://127.0.0.1:8000/**
You will see Django's welcome page 🎉

📌 Step 4: Understanding Django Project Structure

Inside `mywebsite/`, you will see these files:

```
lua
CopyEdit
mywebsite/
│-- manage.py          # Main file to run Django commands
│-- mywebsite/         # Project settings folder
│    │-- __init__.py
│    │-- settings.py    # Configurations (database, security, etc.)
│    │-- urls.py        # URL mapping (like a directory)
│    │-- wsgi.py
```

We will mainly work with **`settings.py`**, **`urls.py`**, and **`views.py`**.

📌 Step 5: Creating a Django App

A Django **app** is like a module in a website. Let's create an app called **students**:

```
bash
CopyEdit
python manage.py startapp students
```

Now, the structure looks like this:

```
sql
CopyEdit
mywebsite/
|-- students/
|    |-- migrations/
|    |-- __init__.py
|    |-- admin.py
|    |-- apps.py
|    |-- models.py       # Defines database tables
|    |-- views.py        # Controls logic
|    |-- urls.py         # Routes URLs to views
```

☑ Add the App to `settings.py`

Open **`mywebsite/settings.py`**, find `INSTALLED_APPS`, and add:

```
python
CopyEdit
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'students',  # Add this line
]
```

📌 Step 6: Create a Simple Webpage

☑ Define a View (Logic)

Open **`students/views.py`**, and add:

```
python
CopyEdit
from django.http import HttpResponse

def home(request):
    return HttpResponse("<h1>Welcome to Django!</h1>")
```

☑ Set Up URLs

Now, create **`students/urls.py`** inside the `students` app:

```
python
CopyEdit
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home, name='home'),
]
```

Then, open **`mywebsite/urls.py`**, and update it:

```
python
CopyEdit
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('students.urls')),  # Link students app
]
```

📌 Step 7: Run the Server and Test

Now, start the Django server again:

```
bash
CopyEdit
python manage.py runserver
```

Visit **http://127.0.0.1:8000/** in your browser.
🎉 You should see **"Welcome to Django!"** displayed.

📌 Step 8: Creating a Model (Database)

Django uses **models** to store data in a database. Let's create a **Student** model.

Open **`students/models.py`**, and add:

```
python
CopyEdit
from django.db import models

class Student(models.Model):
    name = models.CharField(max_length=100)
```

```
    age = models.IntegerField()

    def __str__(self):
        return self.name
```

☑ Apply Migrations

Run these commands:

```
bash
CopyEdit
python manage.py makemigrations
python manage.py migrate
```

This creates the **students table** in the database.

📌 Step 9: Adding Data to the Database

Open **`students/views.py`**, and modify it:

```
python
CopyEdit
from django.shortcuts import render
from .models import Student

def show_students(request):
    students = Student.objects.all()
    return render(request, 'students.html', {'students': students})
```

☑ Create an HTML Template

Inside **`students/`**, create a new folder **`templates/`**, and inside that, create **`students.html`**:

```
html
CopyEdit
<!DOCTYPE html>
<html>
<head>
    <title>Students</title>
</head>
<body>
    <h1>Student List</h1>
    <ul>
        {% for student in students %}
            <li>{{ student.name }} - {{ student.age }} years old</li>
        {% endfor %}
    </ul>
</body>
</html>
```

☑ Update `students/urls.py`

```
python
CopyEdit
from django.urls import path
from . import views

urlpatterns = [
    path('', views.show_students, name='students'),
]
```

📌 Step 10: Adding Data via Django Admin Panel

☑ Create a Superuser

Run this command:

```
bash
CopyEdit
python manage.py createsuperuser
```

Enter a **username, email, and password**.

☑ Register Model in Admin Panel

Open **`students/admin.py`**, and add:

```
python
CopyEdit
from django.contrib import admin
from .models import Student

admin.site.register(Student)
```

Now, start the server:

```
bash
CopyEdit
python manage.py runserver
```

Visit **http://127.0.0.1:8000/admin/**
Login with the superuser credentials and add students! 🎉