
Django Learning Report

Name : Prajwal Bashyal
Institution: Bhaktapur Multiple Campus, BIT
Date : 2025-12-14
Topic : Django – App core concept

Django App

What is a Django App?

A **Django app** is a **small, specific module** that performs **one particular function** inside a Django project.

Think like this:

- **Project** → whole website
- **App** → one feature of the website

Real-life example

If your website is a **college management system**:

App Name	Purpose
`accounts`	Login, signup, users
`students`	Student records
`teachers`	Teacher details
`results`	Marks and grades

Each app does **one job only**.

Django Project vs Django App

Django Project	Django App
Entire website	Part of website
Contains settings	Contains logic
Controls whole system	Controls one feature
Created once	Can create many

👉 A project **contains multiple apps**.

Why Django Uses Apps?

Django apps exist to make development:

- **Organized**
- **Reusable**
- **Maintainable**
- **Scalable**

You can **reuse the same app** in another project without rewriting code.

Creating a Django App

Command:

```
python manage.py startapp blog
```

This creates a folder:

```
blog/
```

Files Inside a Django App (Very Important)

1. `__init__.py`

- Marks the folder as a **Python package**
- Usually **empty**
- Allows Django to treat this folder as an app

You rarely touch this file.

---

2. `apps.py`

Defines app configuration of a django app .

It tells Django:

"This app exists, and this is its name."

```
from django.apps import AppConfig

class BlogConfig(AppConfig):
    name = 'myfirstapp'
```

- Registers the app
- Used internally by Django

- Helps Django to recognize the app
- Usually **auto-created** by Django

---

3. `admin.py`

Used to register models in **Django Admin Panel**.

It is used to show and manage database data in the browser through Django Admin Panel.

Example:

```
from .models import Post

admin.site.register(Post)
```

Purpose:

- Manage data visually
- Add, edit, delete records from browser(admin pannel)

---

4. `models.py`

Defines **database structure**.

Example:

```
class Post(models.Model):
    title = models.CharField(max_length=100)
    content = models.TextField()
```

Each model:

- Represents a **database table**
- Each field = table column

---

5. `views.py`

Contains **logic** of the app.

Example:

```
def home(request):
    return render(request, 'home.html')
```

Purpose:

- Handles requests

- Returns responses (HTML, JSON, etc.)

---

6. `urls.py` (created manually)

Maps URLs to views.

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

Connects **URL → View**

---

7. `tests.py`

Used for **testing the app**.

- Write unit tests
- Ensure code works correctly
- Important for production apps

---

8. `migrations/`

Handles database changes.

- Auto-generated by Django
- Tracks model changes
- Helps sync models with database

Registering App in Project

After creating app, add it to:

```
INSTALLED_APPS = [
    'myfirstapp',
]
```

Without this, Django **ignores the app**.

How Django App Works (Flow)

User → URL → urls.py → views.py → models.py → Template → Response