03. Django URLs and Views

Django URLs and Views

Django follows the **MTV (Model–Template–View)** architecture.
The part that connects the browser URL to the logic is the **URL dispatcher**.

1. URL Dispatcher (urls.py)

The URL dispatcher is responsible for:

- Mapping a URL pattern (like /home/)
- To a corresponding **view function** (in views.py)

Django uses two main functions for routing:

2. path() and re_path()

path()

- Used for simple and readable routes
- Supports dynamic segments (like <int:id>)

Example:

```
path('student/<int:roll>/', views.student_detail)
```

re_path()

- Used when you need **regular expressions** in URLs

Example:

```
re_path(r'^emp/(?P<empid>[0-9]{3})/$', views.employee_detail)
```

3. Views in Django (views.py)

A view is a **Python function** that:

1. Takes a request
2. Performs some logic
3. Returns a **response**

The simplest response is HttpResponse.
Example:

```
from django.http import HttpResponse

def home(request):
   return HttpResponse("Welcome to Django!")
```

4. Returning HttpResponse

```
from django.http import HttpResponse

def greet(request, name):
   return HttpResponse(f"Hello {name}, welcome!")
```

5. Building Dynamic Routes + Connecting to Views

Below is a clean and complete example.

Complete Working Example

Project Structure

```
mysite/
   manage.py
   mysite/
      urls.py
   app1/
      views.py
      urls.py
```

Step-by-Step Instructions

STEP 1: Create Project

```
django-admin startproject mysite
cd mysite
```

STEP 2: Create App

```
python manage.py startapp app1
```

STEP 3: Register the app

Open mysite/settings.py → add:

```
INSTALLED_APPS = [
   ...
   'app1',
]
```

STEP 4: Create Views (app1/views.py)

```
from django.http import HttpResponse

def home(request):
   return HttpResponse("This is the Home Page")

def greet(request, name):
   return HttpResponse(f"Hello, {name}! Welcome to Django.")

def marks(request, roll, mark):
   return HttpResponse(f"Roll Number: {roll} — Marks: {mark}")

def user_id(request, uid):
   return HttpResponse(f"User ID from regex route: {uid}")
```

STEP 5: Create URLs inside the app

Create file app1/urls.py

```
from django.urls import path, re_path
from . import views
```

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

    # dynamic route using path()
    path('greet/<str:name>/', views.greet),

    # multiple dynamic values
    path('report/<int:roll>/<int:mark>/', views.marks),

    # dynamic route using re_path()
    re_path(r'^userid/(?P<uid>[0-9]{4})/$', views.user_id),
]
```

STEP 6: Connect app URLs to project URL

Edit mysite/urls.py

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('app1.urls')),   # attaching app URLs
]
```

Now Run the Server

```
python manage.py runserver
```

Test the URLs in Browser

1. Home

http://127.0.0.1:8000/

2. Dynamic name

http://127.0.0.1:8000/greet/Ravi/

3. Dynamic roll + marks

http://127.0.0.1:8000/report/101/95/

4. Regex route (re_path)

http://127.0.0.1:8000/userid/1234/

Summary Table

Feature	Description	Example
URL Dispatcher	Connects URL to view	path('home/', views.home)
path()	Clean routes	<int:id> <str:name>
re_path()	Regex-based routes	re_path(r'^user/(?P<uid>[0-9]+)/$')
views.py	Contains logic	return HttpResponse()
Dynamic Routes	Accept parameters in URL	path('greet/<str:name>/')