Features Of Django

- Superfast
- Secure
- Scalable
- Versatile

Companies use Django

- Instagram
- Spotify
- Dropbox

Commands run

```
django-admin startproject project-name
```

```
python3 manage.py runserver --help
```

```
python3 manage.py runserver
```

Django => MVT Architecture Model, Views, Templates

Client(Browser) -> URLs -> View -> Templates & Model

Urls - URL Mapper to redirect HTTP requests to approporiate View based on the request URL - They can also redirect on basis of pattern

View - recieves HTTP request returns HTTPS response - access the data from models - formatting from templates

Models - Python objects that define the structure of Web app's data - So they provide mechanism to add/delete/updated in the database, its handled models

Templates - Text files defining the structure of the HTML page - Placeholders to add data from models

These files are:

The outer mysite/ root directory is a container for your project. Its name doesn’t matter to Django; you can rename it to anything you like. manage.py: A command-line utility that lets you interact with this Django project in various ways. You can read all the details about manage.py in django-admin and manage.py. The inner mysite/ directory is the actual Python package for your project. Its name is the Python package name you’ll need to use to import anything inside it (e.g. mysite.urls). mysite/**init**.py: An empty file that tells Python that this directory should be considered a Python package. If you’re a Python beginner, read more about packages in the official Python docs. mysite/settings.py: Settings/configuration for this Django project. Django settings will tell you all about how settings work. mysite/urls.py: The URL declarations for this Django project; a “table of contents” of your Django-powered site. You can read more about URLs in URL dispatcher.

migrations -> Holds the changes made to the models [DB Specific info] admin.py -> Helps us to make changes to the model

How to create a new Web application in Django

> django-admin startapp name

```
- Then add app name to INSTALLED_APPS in settings.py
- Add URL mapping to the views of Application
    - Import Views from the app
    - path('', name_of_function)
- Add a new view in views.py
    - Imported django.http -> HttpResponse
    - return HttpResponse(str)
```

Static and Dynamic Routing

```
- We saw, that when we gave ''/'try' in url paths, empty string -> Static
- If a user accesses the API endpoint with some number, We want to print
that number as his/her age
- <int:num> - num is the variable passed as the query param

- Params we can pass:
    - str - Matches any non-empty string, excluding the path separator,
'/'. This is the default if a converter isn’t included in the expression.
    - int - Matches zero or any positive integer. Returns an int.
    - slug - Matches any slug string consisting of ASCII letters or
numbers, plus the hyphen and underscore characters. For example, building-
your-1st-django-site.
    - uuid - Matches a formatted UUID. To prevent multiple URLs from
mapping to the same page, dashes must be included and letters must be
lowercase. For example, 075194d3-6885-417e-a8a8-6c931e272f00. Returns a
UUID instance.
    - path - Matches any non-empty string, including the path separator,
'/'. This allows you to match against a complete URL path rather than a
segment of a URL path as with str.
```

URL Mappings

```
- If we have multiple applications

2 Applications -> Student and Teacher
2 views.py

/student/ -> Student Application
    /student/name/<id>
    /student/address/<id>

- urls.py
    - name
```

```
    - address

/teacher/name/<id>
/teacher/address/<id>

/teacher/ -> Teacher Application
    /teacher/name/<id>
    /teacher/address/<id>

- urls.py
    - name
    - address
```

Steps: 1. We have to import include from django.urls 2. We added urls.py in the application and add all the urls of that application to that urls.py 3. Change the url mapping the urls.py file of the project