

# Session 2: *Basics of Django*

Xander Warszawski

**GitHub**



**IEEE Student Branch**  
KU Leuven Campus Brugge

0

## ***Contents of last session***

1. Introduction
2. Installing tools
3. Git and GitHub
4. Django setup
5. Docker setup

1

**GitHub**



**IEEE Student Branch**  
KU Leuven Campus Brugge

1

## Contents of this session

1. Django Architecture
2. Environment variables
3. Pages app
4. Static assets
5. Common app
6. Albums app

2

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

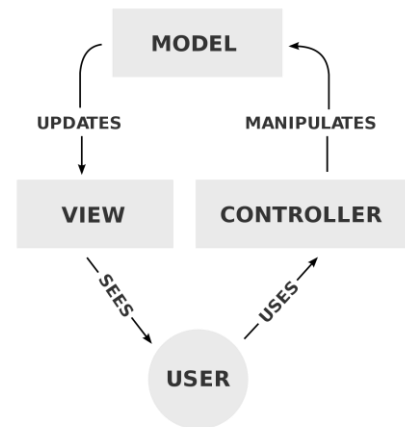
2

# 1. Django Architecture

3

## MVC: Model-View-Controller

- Software architectural pattern
- Used by all web frameworks
- **Model**: manages data & core business logic
- **View**: renders data from model
- **Controller**: accepts user input and performs application logic



4

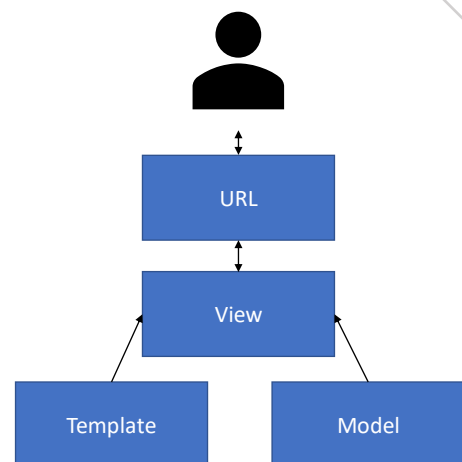
GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

4

## MVTU: Model-View-Template-Url

- Model: manages data & core business logic
- **View: describes which data is sent to user**
- **Template: presentation of data**
- **Url: Regex components configured to a view**
- Flow from request to response:
  1. User requests certain URL
  2. In urls.py a url pattern is found that matches it, this is linked to one view
  3. A view combines data from model (from models.py) and styling (\*.html file)
  4. A response is returned



5

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

5

## Generated files by startproject

- **manage.py**: command line utility for interaction with django
- **config/**
  - **\_\_init\_\_.py**: empty file that tells python this is a package
  - **settings.py**: settings for this project
  - **urls.py**: Main url declarations
  - **asgi.py**: Async Server Gateway Interface: useful for websockets (used for e.g. chatting, real-time apps)
  - **wsgi.py**: Web Server Gateway Interface: we will use this one

6




IEEE Student Branch  
KU Leuven Campus Brugge

6

## Contents of settings.py

- **BASE\_DIR**: base directory of project
- **SECRET\_KEY**: used to provide cryptographic signing
- **DEBUG**: when enabled will provide detailed error pages, unsecure in prod
- **ALLOWED\_HOSTS**: host/domains that can serve this site, to prevent Host header attacks
- **INSTALLED\_APPS**: list of all apps that are enabled in this project
- **MIDDLEWARE**: list of enabled middleware; these are low-level “plugins”
- **ROOT\_URLCONF**: represents full python import path to root URLconf
- **TEMPLATES**: contains all settings for templates
- **WSGI\_APPLICATION**: path to WSGI Application that Django servers will use
- **DATABASES**: database settings
- **AUTH\_PASSWORD\_VALIDATORS**: validators used to check password strength
- **STATIC\_URL**: url to use when referring to static files
- **DEFAULT\_AUTO\_FIELD**: default primary key field

7




IEEE Student Branch  
KU Leuven Campus Brugge

7

## Generated files by startapp

- **`__init__.py`**
- **`admin.py`**: config file for built-in Django admin app
- **`apps.py`**: config file for app itself
- **`migrations/`**: keeps track of changes to models, stays synced with db
- **`models.py`**: where we define our db models
- **`tests.py`**: for app-specific tests
- **`views.py`**: where we handle request/response logic

8

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

8

## Questions?

9

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

9

## 2. *Environment Variables*

10

### *Environment variables*

- Variables that will be different according to the environment
- Eg:
  - Database\_url
  - Debug
  - Secret\_Key
  - ...
- Part of the Twelve-Factor App Design: <https://12factor.net/>

11

GitHub

 IEEE Student Branch  
KU Leuven Campus Brugge

11

## Git checkpoint

1. Create new feature branch

12

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

12

## Environment variables

1. Pip install **environs[django]** & freeze
2. Update **docker-compose.yml**
  1. Take <value> for DJANGO\_SECRET\_KEY from **config/settings.py**
    1. If \$ is present in secret key: add another \$; \$ → \$\$

13

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

13

## ***docker-compose.yml***

```
depends_on:  
  - db  
environment:  
  - "DJANGO_SECRET_KEY=<value>"  
  - "DJANGO_DEBUG=True"
```

14

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

14

## ***Environment variables***

1. Pip install **environs[django]** & freeze
2. Update **docker-compose.yml**
3. Update **config/settings.py**

15

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

15



## config/settings.py

```
from environs import Env
from django.core.management.utils import get_random_secret_key

env = Env()
env.read_env()

SECRET_KEY = env("DJANGO_SECRET_KEY", get_random_secret_key())

DEBUG = env.bool("DJANGO_DEBUG", default=False)

ALLOWED_HOSTS = env("DJANGO_ALLOWED_HOSTS", "127.0.0.1,localhost").split(",")

DATABASES = {
    'default': env.dj_db_url("DATABASE_URL", default="postgres://postgres@db/postgres")
}
```

16




IEEE Student Branch  
KU Leuven Campus Brugge

16

## Git checkpoint

1. Commit changes
2. Create PR

17




IEEE Student Branch  
KU Leuven Campus Brugge

17

# Questions?

---

18

**GitHub****IEEE Student Branch**  
KU Leuven Campus Brugge

18

## 3. *Pages app*

19

## Flow for creating an application

1. Figure out what you will make
2. Design your models before coding them, think about relationships
  1. One-to-one: Table 1 has ID that points to entry in Table 2
  2. One-to-many: Table 2 has ID that points to entry in Table 1
  3. Many-to-many: Linking table that has IDs of Table 1 and Table 2
3. Code models
4. Code views
5. Code templates
6. Code urls
7. Write tests (with a bit of practice this can be step 3 → TDD)

20

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

20

## Music store: model design

- We sell albums
- One album has
  - One or more artists
  - Cover (image)
  - Price (decimal)
  - Record label (string)
- One artist has:
  - One or more albums
  - Artist name (string)
- CustomUser has:
  - One or more paid albums
- So:
  - Album-Artist: many-to-many relationship (1 album can have +1 artists, 1 artist can have +1 albums)
  - CustomUser-Album: many-to-many (1 user can have +1 albums, 1 album can be paid by +1 users)

21

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

21

## Some good practices

- When structuring your code, work by:
  - Fat models, utility modules, thin views, stupid templates
- **Fat models & thin views**: business logic in models instead of views
- **Utility modules**: create (a) separate module(s) for shared code
  - Eg app named **common** or core
- **Stupid templates**: templates should absolutely NOT contain logic, they should only 'print' data
- Also:
  - Try to keep **apps as small as possible**, they should be focused on one task

22

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

22

## Git checkpoint

1. Create new feature branch

23

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

23

## Pages app

1. Execute: **`docker-compose exec web python manage.py startapp pages`**
2. Configure templates:
  1. Update **`config/settings.py`**

24



 IEEE Student Branch  
 KU Leuven Campus Brugge

24

## `config/settings.py` (1)

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'pages.apps.PagesConfig'
]
```

25



 IEEE Student Branch  
 KU Leuven Campus Brugge

25

## config/settings.py (2)

```
TEMPLATES = [
    {
        ...
        'DIRS': [BASE_DIR / "templates"],
        ...
    },
]
```

26

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

26

## Pages app

1. Execute: ***docker-compose exec web python manage.py startapp pages***
2. Configure templates:
  1. Update ***config/settings.py***
  2. Add ***templates/\_base.html***

27

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

27

## templates/\_base.html (1)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>{% block title %}Music store{% endblock title %}</title>
  </head>
  <body>
```

28



 IEEE Student Branch  
 KU Leuven Campus Brugge

28

## templates/\_base.html (2)

```
  <div>
    <ul>
      <li><a href="">Home</a></li>
      <li><a href="">Music</a></li>
      <li><a href="">My music</a></li>
      <li><a href="">Log out</a></li>
      <li><a href="">Log in</a></li>
      <li><a href="">Sign up</a></li>
    </ul>
  </div>
  <div>
    {% block content %}
    {% endblock content %}
  </div>
</body>
</html>
```

29



 IEEE Student Branch  
 KU Leuven Campus Brugge

29

## Pages app

1. Execute: **`docker-compose exec web python manage.py startapp pages`**
2. Configure templates
3. Add **`templates/pages/home.html`**

30

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

30

## templates/pages/home.html

```
{% extends '_base.html' %}

{% block title %}Home{% endblock title %}

{% block content %}
    <h1>Welcome to the music store!</h1>
{% endblock content %}
```

31

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

31



## Pages app

1. Execute: **`docker-compose exec web python manage.py startapp pages`**
2. Configure templates
3. Add **`templates/pages/home.html`**
4. Update **`pages/views.py`**

32




IEEE Student Branch  
KU Leuven Campus Brugge

32

## pages/views.py

```
from django.views.generic import TemplateView

class HomePageView(TemplateView):
    template_name = "pages/home.html"
```

33




IEEE Student Branch  
KU Leuven Campus Brugge

33

## Pages app

1. Execute: **`docker-compose exec web python manage.py startapp pages`**
2. Configure templates
3. Add **`templates/pages/home.html`**
4. Update **`pages/views.py`**
5. Add **`pages/urls.py`**

34




IEEE Student Branch  
KU Leuven Campus Brugge

34

## `pages/urls.py`

```
from django.urls import path
from .views import HomePageView

urlpatterns = [
    path('', HomePageView.as_view(), name="home")
]
```

35




IEEE Student Branch  
KU Leuven Campus Brugge

35

## Pages app

1. Execute: **`docker-compose exec web python manage.py startapp pages`**
2. Configure templates
3. Add **`templates/pages/home.html`**
4. Update **`pages/views.py`**
5. Add **`pages/urls.py`**
6. Update **`config/urls.py`**

36



 IEEE Student Branch  
 KU Leuven Campus Brugge

36

## config/urls.py

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('pages.urls'))
]
```

37



 IEEE Student Branch  
 KU Leuven Campus Brugge

37

## Configure tests

1. Check if Testing icon is visible on left bar in VSCode
  1. Otherwise Right-click > Testing, to make it show up
  2. Then Configure Python Tests
  3. Pick: 'pytest'
  4. Pick: Root directory
  5. Add **.vscode** to **.gitignore**
  6. Pip install **pytest-django** & freeze
  7. Add **pytest.ini** next to manage.py

38



 IEEE Student Branch  
 KU Leuven Campus Brugge

38

## pytest.ini

```
[pytest]
DJANGO_SETTINGS_MODULE = config.settings.test
python_files = tests.py test_*.py *_tests.py
```

39



 IEEE Student Branch  
 KU Leuven Campus Brugge

39

## Configure tests

1. Check if Testing icon is visible on left bar in VSCode
  1. Otherwise Right-click > Testing, to make it show up
  2. Then Configure Python Tests
  3. Pick: 'pytest'
  4. Pick: Root directory
  5. Add **.vscode** to **.gitignore**
  6. Pip install **pytest-django** & freeze
  7. Add **pytest.ini** next to manage.py
  8. Add **settings** folder in **config**
    1. Copy **settings.py** as **base.py** into this folder
    2. Add **\_\_init\_\_.py**

40



 IEEE Student Branch  
 KU Leuven Campus Brugge

40

## config/settings/\_\_init\_\_.py

```
from .base import *
```

41



 IEEE Student Branch  
 KU Leuven Campus Brugge

41

## Configure tests

1. Check if Testing icon is visible on left bar in VSCode
  1. Otherwise Right-click > Testing, to make it show up
  2. Then Configure Python Tests
  3. Pick: 'pytest'
  4. Pick: Root directory
  5. Add **.vscode** to **.gitignore**
  6. Pip install **pytest-django** & freeze
  7. Add **pytest.ini** next to manage.py
  8. Add **settings** folder in **config**
    1. Copy **settings.py** as **base.py** into this folder
      1. Change **BASE\_DIR** to: `Path(__file__).resolve().parent.parent.parent`
    2. Add **\_\_init\_\_.py**
    3. Add **test.py**

42



 IEEE Student Branch  
 KU Leuven Campus Brugge

42

## config/settings/test.py

```

from .base import *

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': 'testdb'
    }
}

DEBUG = False

# Needed for later
SECURE_SSL_REDIRECT = False
SECURE_HSTS_SECONDS = 0
SECURE_HSTS_INCLUDE_SUBDOMAINS = False
SECURE_HSTS_PRELOAD = False
SESSION_COOKIE_SECURE = False
CSRF_COOKIE_SECURE = False

```

43



 IEEE Student Branch  
 KU Leuven Campus Brugge

43

## How to 'design' tests?

### 1. **Arrange**

- Create your data; eg. Fill your db

### 2. **Act**

- Perform your action; eg. When testing update of a model, execute that update method

### 3. **Assert**

- Check if your result matches your expected result

44



 IEEE Student Branch  
 KU Leuven Campus Brugge

44

## Types of TestCases

- **SimpleTestCase:**

- No db access

- **TestCase**

- Rolls back db changes using transactions

- **TransactionTestCase**

- Rolls back db changes by flushing all tables

- **LiveServerTestCase**

- Based on TransactionTestCase, also launches live server thread (for eg. Selenium)

- Want to **test business logic**? Use **SimpleTestCase**

- Want to **test functionality that needs a db**? Use **TestCase**

- TransactionTestCase and LiveServerTestCase are slower than the others

45



 IEEE Student Branch  
 KU Leuven Campus Brugge

45

## Types of setup methods

- **setUpClass**
  - Runs once at start of case class
- **setUp**
  - Runs at start of each test
- **setUpTestData**
  - Like setUpClass but rolls back transactions done on this data during yesys

46



 IEEE Student Branch  
 KU Leuven Campus Brugge

46

## Pages app

1. Execute: **`docker-compose exec web python manage.py startapp pages`**
2. Configure templates
3. Add **`templates/pages/home.html`**
4. Update **`pages/views.py`**
5. Add **`pages/urls.py`**
6. Update **`config/urls.py`**
7. Update **`pages/tests.py`**

47



 IEEE Student Branch  
 KU Leuven Campus Brugge

47



## pages/tests.py (1)

```
import pytest
from django.test import SimpleTestCase
from django.urls import reverse

@pytest.mark.django_db
class HomePageTests(SimpleTestCase):
    def setUp(self):
        # Arrange
        url = reverse("home")
        # Act
        self.response = self.client.get(url)
```

48




IEEE Student Branch  
KU Leuven Campus Brugge

48

## pages/tests.py (2)

```
def test_url_exists_at_correct_location(self):
    # self.assertEqual(self.response.status_code, 200)
    # option above is used frequently in Django, better to use option below when using pytest
    # Assert
    assert self.response.status_code == 200

def test_homepage_template(self):
    # self.assertTemplateUsed(self.response, "pages/home.html")
    # same remark as before
    assert "pages/home.html" in [x.name for x in self.response.templates]
```

49




IEEE Student Branch  
KU Leuven Campus Brugge

49

## Pages app

1. Execute: **`docker-compose exec web python manage.py startapp pages`**
2. Configure templates
3. Add **`templates/pages/home.html`**
4. Update **`pages/views.py`**
5. Add **`pages/urls.py`**
6. Update **`config/urls.py`**
7. Update **`pages/tests.py`**
8. Update **`templates/_base.html`**

50



 IEEE Student Branch  
 KU Leuven Campus Brugge

50

## templates/\_base.html

```
<li><a href="{% url 'home' %}">Home</a></li>
<li><a href="">Music</a></li>
<li><a href="">My music</a></li>
<li><a href="">Log out</a></li>
<li><a href="">Log in</a></li>
<li><a href="">Sign up</a></li>
```

51



 IEEE Student Branch  
 KU Leuven Campus Brugge

51

## Git checkpoint

1. Commit changes
2. Create PR
3. Tests will fail:
  1. Edit run on 'Run Tests' in action from **`python manage.py test`** to **`pytest`**
  2. Update branch in PR
4. Useful **recommend**:
  1. Pip install **`black`** and **`isort`** & freeze
  2. Run before each PR (can also be added to GitHub action):
    - **`black`**.
    - **`isort`**.

52

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

52

## Questions?

53

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

53

## 4. *Static assets*

54

### *Static assets*

---

- Static assets?
  - CSS
  - JavaScript
  - Images
- Local: auto served

55

**GitHub****IEEE Student Branch**  
KU Leuven Campus Brugge

55

## Bootstrap

- In templates you can freestyle a bit, I will provide the bare minimum
- Definitely try using bootstrap for these: <https://getbootstrap.com/>
- Also use crispy-forms: <https://django-crispy-forms.readthedocs.io/>

56

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

56

## Git checkpoint

1. Create new feature branch

57

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

57

## Image

1. Update **config/settings/base.py**
2. Add image to **static/images**
3. Add it to **home.html** using **{% static 'images/<NAME>' %}**
  1. Don't forget to add **{% load static %}**

63




IEEE Student Branch  
KU Leuven Campus Brugge

63

## config/settings/base.py

```
STATIC_URL = '/static/'
STATICFILES_DIRS = [BASE_DIR / "static"]
STATIC_ROOT = BASE_DIR / "staticfiles"
STATICFILES_STORAGE =
"django.contrib.staticfiles.storage.StaticFilesStorage"
```

65




IEEE Student Branch  
KU Leuven Campus Brugge

65

## Production development

1. Execute: **`docker-compose exec web python manage.py collectstatic`**

67

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

67

## Git checkpoint

1. Commit changes
2. Create PR

68

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

68

# Questions?

69

**GitHub****IEEE Student Branch**  
KU Leuven Campus Brugge

69

## 5. Common app

70



## Git checkpoint

1. Create new feature branch

71

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

71

## Common app

1. Execute: **`docker-compose exec web python manage.py startapp common`**
2. Update **`common/models.py`**

72

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

72

## common/models.py

```
from django.db import models

class TimeStampedModel(models.Model):
    created = models.DateTimeField(auto_now_add=True)
    modified = models.DateTimeField(auto_now=True)

    class Meta:
        abstract = True
```

73

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

73

## Git checkpoint

1. Commit changes
2. Create PR

74

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

74

## Questions?

---

75

**GitHub****IEEE Student Branch**  
KU Leuven Campus Brugge

75

## 6. Albums app

76

## Git checkpoint

1. Create new feature branch

77

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

77

## Albums app

1. Pip install ***pillow*** & freeze
2. Execute: ***docker-compose exec web python manage.py startapp albums***
3. Update ***albums/models.py***

78

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

78

## albums/models.py (1)

```
import uuid
from django.db import models
from common.models import TimeStampedModel
from django.urls import reverse

class Album(TimeStampedModel):
    id = models.UUIDField(
        primary_key=True,
        default=uuid.uuid4,
        editable=False
    )
    title = models.CharField(max_length=200)
    cover = models.ImageField(upload_to="covers/", blank=True)
    price = models.DecimalField(max_digits=6, decimal_places=2)
    record_label = models.CharField(max_length=200)
```

79




IEEE Student Branch  
KU Leuven Campus Brugge

79

## albums/models.py (2)

```
def artist(self):
    return self.artist_set.all()[0]

def artists(self):
    return self.artist_set.all()

def __str__(self):
    return self.title

def get_absolute_url(self):
    return reverse("album_detail", args=[str(self.id)])
```

80




IEEE Student Branch  
KU Leuven Campus Brugge

80

## *albums/models.py (3)*

```
class Artist(TimeStampedModel):
    name = models.CharField(max_length=200)
    albums = models.ManyToManyField(Album)

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

81




IEEE Student Branch  
KU Leuven Campus Brugge

81

## *Albums app*

1. Execute: ***docker-compose exec web python manage.py startapp albums***
2. Update ***albums/models.py***
3. Update ***albums/admin.py***

82




IEEE Student Branch  
KU Leuven Campus Brugge

82

## albums/admin.py

```
from django.contrib import admin
from .models import Album, Artist

class AlbumAdmin(admin.ModelAdmin):
    list_display = ("title", "cover", "price", "record_label", "artists")

class ArtistAdmin(admin.ModelAdmin):
    list_display = ("name",)

admin.site.register(Album, AlbumAdmin)
admin.site.register(Artist, ArtistAdmin)
```

83




IEEE Student Branch  
KU Leuven Campus Brugge

83

## Albums app

1. Execute: **`docker-compose exec web python manage.py startapp albums`**
2. Update **`albums/models.py`**
3. Update **`albums/admin.py`**
4. Update **`albums/views.py`**

84




IEEE Student Branch  
KU Leuven Campus Brugge

84

## albums/views.py (1)

```
from django.views.generic import ListView, DetailView
from django.db.models import Q

from .models import Album

class AlbumListView(ListView):
    paginate_by = 10
    model = Album
    context_object_name = "album_list"
    template_name = "albums/album_list.html"
```

85




IEEE Student Branch  
KU Leuven Campus Brugge

85

## albums/views.py (2)

```
def get_queryset(self):
    query = self.request.GET.get("q")

    if query is None:
        return self.model.objects.all()

    return self.model.objects.filter(
        Q(title__icontains=query) | Q(artist__name__icontains=query)
    )

class AlbumDetailView(DetailView):
    model = Album
    context_object_name = "album"
    template_name = "albums/album_detail.html"
```

86




IEEE Student Branch  
KU Leuven Campus Brugge

86



## Albums app

1. Execute: **`docker-compose exec web python manage.py startapp albums`**
2. Update **`albums/models.py`**
3. Update **`albums/admin.py`**
4. Update **`albums/views.py`**
5. Create **`templates/albums/album_list.html`**
6. Create **`templates/albums/album_detail.html`**

87

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

87

## templates/albums/album\_list.html (1)

```
{% extends "_base.html" %}

{% block title %}Albums{% endblock title %}

{% block content %}
    <form action="{% url 'album_list' %}" method="get">
        <input type="search" name="q" placeholder="Search">
        <button type="submit">Search</button>
    </form>

    {% for album in album_list %}
        <div>
            <h2><a href="{% album.get_absolute_url %}">
                {{ album.title }} - {{ album.artist.name }}
            </a></h2>
        </div>
    {% endfor %}
```

88

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

88

## templates/albums/album\_list.html (2)

```
<div class="pagination">
  <span class="step-links">
    {% if page_obj.has_previous %}
      <a href="?page=1">&laquo; first</a>
      <a href="?page={{ page_obj.previous_page_number }}">previous</a>
    {% endif %}

    <span class="current">
      Page {{ page_obj.number }} of {{ page_obj.paginator.num_pages }}.
    </span>

    {% if page_obj.has_next %}
      <a href="?page={{ page_obj.next_page_number }}">next</a>
      <a href="?page={{ page_obj.paginator.num_pages }}">last &raquo;</a>
    {% endif %}
  </span>
</div>
{% endblock content %}
```

89

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

89

## templates/albums/album\_detail.html (1)

```
{% extends "_base.html" %}

{% block title %}{{ album.title }}{% endblock title %}

{% block content %}
  <div>
    {% if album.cover %}
      
    {% endif %}
    <h2>{{ album.title }}</h2>
  </div>
{% endblock content %}
```

90

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

90

## templates/albums/album\_detail.html (2)

```
<p>Artist(s):</p>
<ul>
    {% for artist in album.artists %}
        <li>{{ artist.name }}</li>
    {% endfor %}
</ul>
<p>Record label: {{ album.record_label }}</p>
<p>Price: €{{ album.price }}</p>
</div>
{% endblock content %}
```

91

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

91

## Albums app

1. Execute: **`docker-compose exec web python manage.py startapp albums`**
2. Update **`albums/models.py`**
3. Update **`albums/admin.py`**
4. Update **`albums/views.py`**
5. Create **`templates/albums/album_list.html`**
6. Create **`templates/albums/album_detail.html`**
7. Add **`albums/urls.py`**

92

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

92

## albums/urls.py

```
from django.urls import path

from .views import AlbumListView, AlbumDetailView

urlpatterns = [
    path("", AlbumListView.as_view(), name="album_list"),
    path("<uuid:pk>/", AlbumDetailView.as_view(), name="album_detail")
]
```

93

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

93

## Albums app

1. Execute: ***docker-compose exec web python manage.py startapp albums***
2. Update ***albums/models.py***
3. Update ***albums/admin.py***
4. Update ***albums/views.py***
5. Create ***templates/albums/album\_list.html***
6. Create ***templates/albums/album\_detail.html***
7. Update ***albums/urls.py***
8. Update ***config/settings/base.py***

94

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

94

## config/settings/base.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'pages.apps.PagesConfig',
    'albums.apps.AlbumsConfig'
]
...
MEDIA_URL = "/media/"
MEDIA_ROOT = BASE_DIR / "media"
```

95




IEEE Student Branch  
KU Leuven Campus Brugge

95

## Albums app

1. Execute: ***docker-compose exec web python manage.py startapp albums***
2. Update ***albums/models.py***
3. Update ***albums/admin.py***
4. Update ***albums/views.py***
5. Create ***templates/albums/album\_list.html***
6. Create ***templates/albums/album\_detail.html***
7. Update ***albums/urls.py***
8. Update ***config/settings.py***
9. Update ***config/urls.py***

96




IEEE Student Branch  
KU Leuven Campus Brugge

96

## config/urls.py

```
from django.contrib import admin
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('pages.urls')),
    path('albums/', include('albums.urls'))
] + static(
    settings.MEDIA_URL, document_root=settings.MEDIA_ROOT
)
```

97




IEEE Student Branch  
KU Leuven Campus Brugge

97

## Albums app

1. Execute: **`docker-compose exec web python manage.py startapp albums`**
2. Update **`albums/models.py`**
3. Update **`albums/admin.py`**
4. Update **`albums/views.py`**
5. Create **`templates/albums/album_list.html`**
6. Create **`templates/albums/album_detail.html`**
7. Update **`albums/urls.py`**
8. Update **`config/settings.py`**
9. Update **`config/urls.py`**
10. Execute: **`docker-compose exec web python manage.py makemigrations`**

98




IEEE Student Branch  
KU Leuven Campus Brugge

98

## Examine migrations: albums/migrations/0001\_initial.py (1)

```
from django.db import migrations, models
import uuid
```

```
class Migration(migrations.Migration):

    initial = True

    dependencies = [
    ]
```

99




IEEE Student Branch  
KU Leuven Campus Brugge

99

## Examine migrations: albums/migrations/0001\_initial.py (2)

```
operations = [
    migrations.CreateModel(
        name='Album',
        fields=[
            ('created', models.DateTimeField(auto_now_add=True)),
            ('modified', models.DateTimeField(auto_now=True)),
            ('id', models.UUIDField(default=uuid.uuid4, editable=False, primary_key=True, serialize=False)),
            ('title', models.CharField(max_length=200)),
            ('cover', models.ImageField(blank=True, upload_to='covers/')),
            ('price', models.DecimalField(decimal_places=2, max_digits=6)),
            ('record_label', models.CharField(max_length=200)),
        ],
        options={
            'abstract': False,
        },
    ),
]
```

100




IEEE Student Branch  
KU Leuven Campus Brugge

100

## Examine migrations: albums/migrations/0001\_initial.py (3)

```
migrations.CreateModel(
    name='Artist',
    fields=[
        ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
        ('created', models.DateTimeField(auto_now_add=True)),
        ('modified', models.DateTimeField(auto_now=True)),
        ('name', models.CharField(max_length=200)),
        ('albums', models.ManyToManyField(to='albums.album')),
    ],
    options={
        'abstract': False,
    },
),
]
```

101




IEEE Student Branch  
KU Leuven Campus Brugge

101

## Migrations in production

1. Create your migrations in dev & run these on your PC
  1. Good practice to examine your migrations
2. No issues? Commit migrations
3. Backup production db
4. Execute your migrations
5. Check if everything is ok

102




IEEE Student Branch  
KU Leuven Campus Brugge

102



## Albums app

1. Execute: **`docker-compose exec web python manage.py startapp albums`**
2. Update **`albums/models.py`**
3. Update **`albums/admin.py`**
4. Update **`albums/views.py`**
5. Create **`templates/albums/album_list.html`**
6. Create **`templates/albums/album_detail.html`**
7. Update **`albums/urls.py`**
8. Update **`config/settings.py`**
9. Update **`config/urls.py`**
10. Execute: **`docker-compose exec web python manage.py makemigrations`**
11. Execute: **`docker-compose exec web python manage.py migrate`**
12. Update **`albums/tests.py`**

103

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

103

## `albums/tests.py (1)`

```
import pytest
from django.test import TestCase
from django.urls import reverse

from .models import Album, Artist

@pytest.mark.django_db
class AlbumTests(TestCase):
```

104

GitHub

IEEE Student Branch  
KU Leuven Campus Brugge

104

## albums/tests.py (2)

```
@classmethod
def setUpTestData(cls):
    cls.artist = Artist.objects.create(
        name = "Beartooth"
    )

    cls.album = Album.objects.create(
        title = "Below",
        price = "15.99",
        record_label = "Red Bull Records"
    )
    cls.artist.albums.add(cls.album)
```

105




IEEE Student Branch  
KU Leuven Campus Brugge

105

## albums/tests.py (3)

```
def test_album_listing(self):
    assert f"{self.album.title}" == "Below"
    assert f"{self.album.artists()[0].name}" == "Beartooth"
    assert f"{self.album.price}" == "15.99"
    assert f"{self.album.record_label}" == "Red Bull Records"

def test_album_list_view(self):
    response = self.client.get(reverse("album_list"))
    assert response.status_code == 200
    assert "Beartooth" in response.rendered_content
    assert "albums/album_list.html" in [x.name for x in response.templates]
```

106




IEEE Student Branch  
KU Leuven Campus Brugge

106

## albums/tests.py (4)

```
def test_album_detail_view(self):
    response = self.client.get(self.album.get_absolute_url())
    assert response.status_code == 200
    assert "Beartooth" in response.rendered_content
    assert "albums/album_detail.html" in [x.name for x in response.templates]
```

107




IEEE Student Branch  
KU Leuven Campus Brugge

107

## Albums app

1. Execute: ***docker-compose exec web python manage.py startapp albums***
2. Update ***albums/models.py***
3. Update ***albums/admin.py***
4. Update ***albums/views.py***
5. Create ***templates/albums/album\_list.html***
6. Create ***templates/albums/album\_detail.html***
7. Update ***albums/urls.py***
8. Update ***config/settings.py***
9. Update ***config/urls.py***
10. Execute: ***docker-compose exec web python manage.py makemigrations***
11. Execute: ***docker-compose exec web python manage.py migrate***
12. Update ***albums/tests.py***
13. Update ***templates/\_base.html***

108




IEEE Student Branch  
KU Leuven Campus Brugge

108

## templates/pages/\_base.html

```
<li><a href="{% url 'home' %}">Home</a></li>
<li><a href="{% url 'album_list' %}">Music</a></li>
<li><a href="">My music</a></li>
<li><a href="">Log out</a></li>
<li><a href="">Log in</a></li>
<li><a href="">Sign up</a></li>
```

109




IEEE Student Branch  
KU Leuven Campus Brugge

109

## Git checkpoint

1. Commit changes
2. Create PR

110




IEEE Student Branch  
KU Leuven Campus Brugge

110

# Questions?

---

111

**GitHub****IEEE Student Branch**  
KU Leuven Campus Brugge