infoShare ACADEMY

Lords of The Django

Fellowship of The Code

HELLO

Damian Filipkowski

infoShare ACADEMY

Shire

Instalacja i Konfiguracja

infoShare ACADEMY

infoShare ACADEMY

> Django makes it easier to build better web apps more quickly and with less code.

> The web framework for perfectionists with deadlines.

www.djangoproject.com

Model Template View

Model

Podobnie jak we wzorcu MVC, zawiera wszystko, co jest związane z dostępem do danych i ich walidacją.

Template

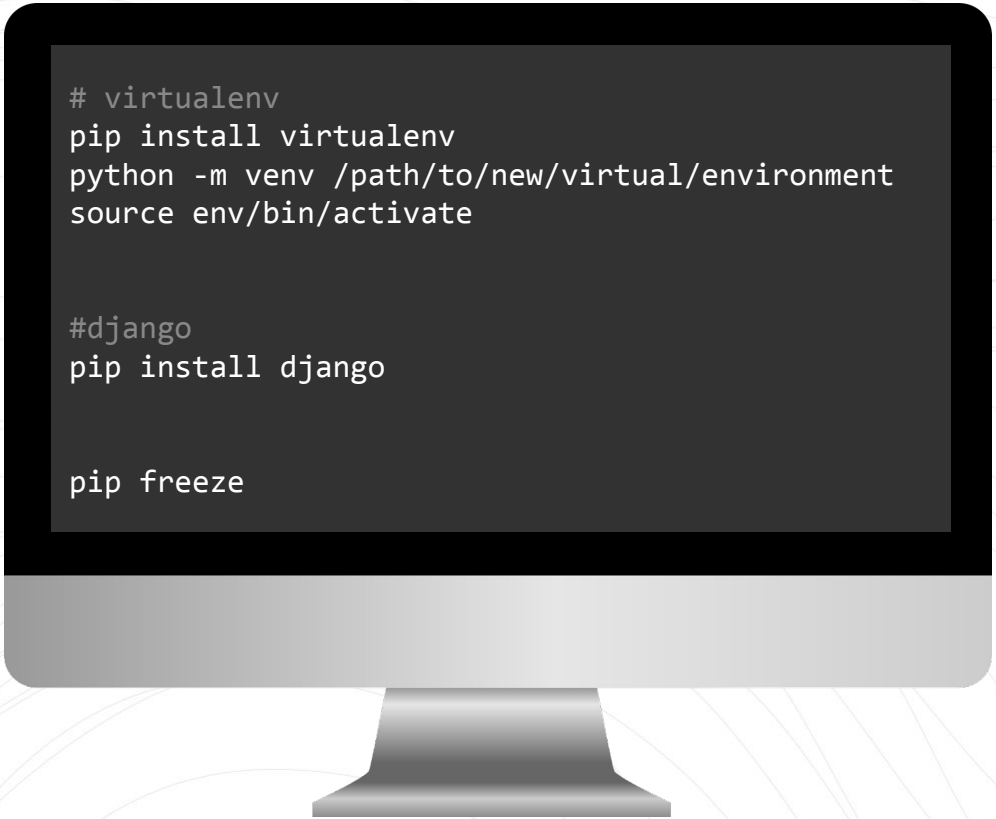
Odnosi się to do widoku we wzorcu MVC, obsługuje logikę prezentacji i kontroluje, co i jak powinno być wyświetlane użytkownikowi.

View

Ta część odnosi się do kontrolera we wzorcu MVC i obsługuje całą logikę biznesową. Służy jako pomost między modelem a szablonem

Django Reinhardt

Instalacja

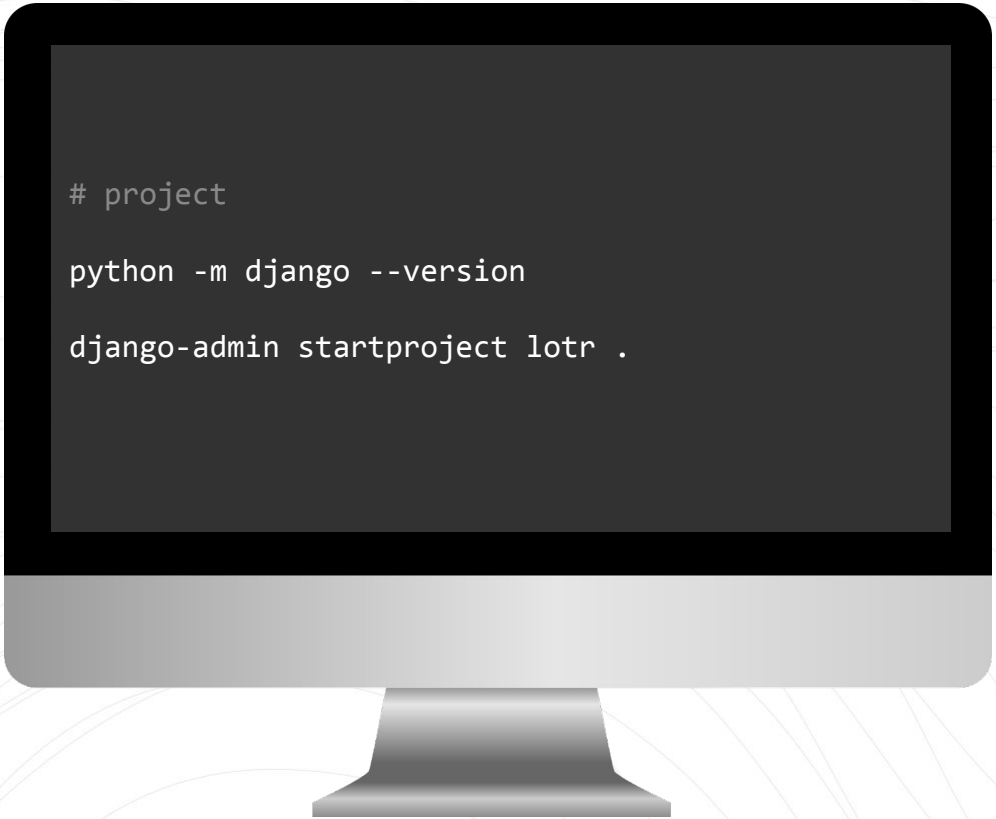


https://docs.djangoproject.com/en/5.0/

Bree

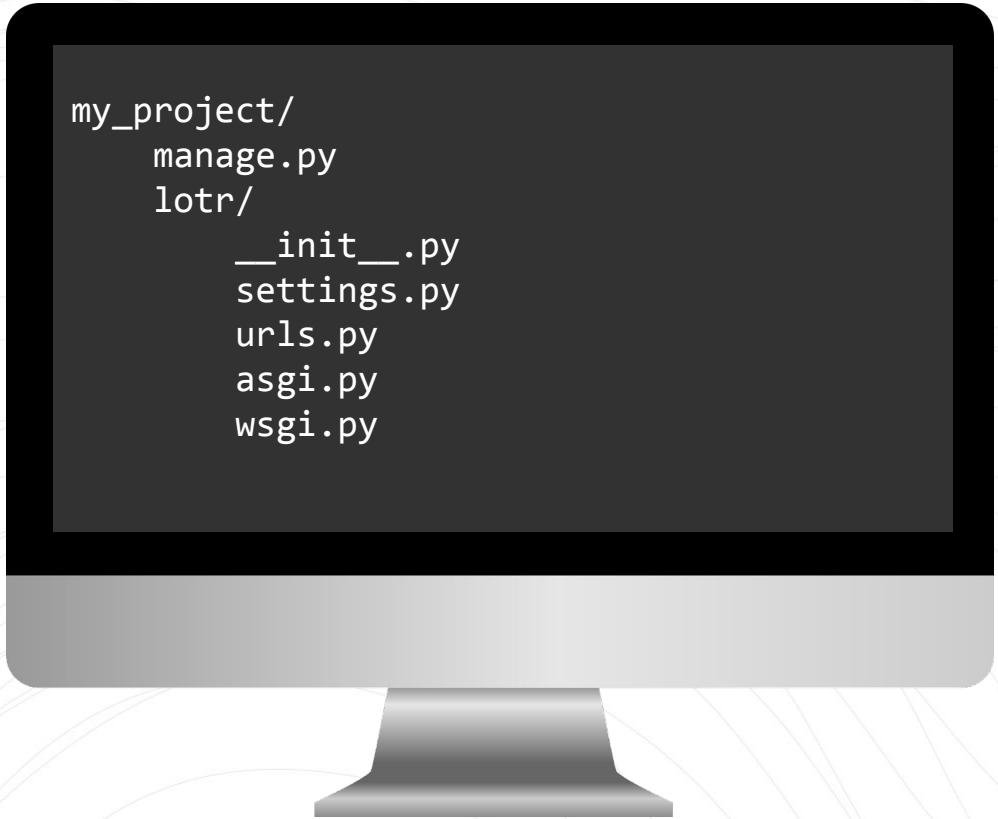
Projekt i Aplikacja

Projekt



https://docs.djangoproject.com/en/5.0/intro/tutorial01/

Struktura Plików



https://docs.djangoproject.com/en/5.0/intro/tutorial01/

manage .py

- **manage.py** – narzędzie linii komend, które pozwala ci oddziaływać z tym projektem Django na wiele sposobów, jest sklonowanym django-admin pod inną nazwą.

Jakie znaczenie ma utworzenie kopii narzędzia django-admin w naszym projekcie?

- narzędzie jest przypisane tylko i wyłącznie do naszego projektu,
- różne biblioteki i dodatki do Django modyfikują działanie narzędzia, dodając nowe funkcje,
- dzięki temu mamy spójne i wygodne narzędzie do zarządzania projektem, dostosowane do naszych potrzeb.

https://docs.djangoproject.com/pl/5.0/ref/django-admin/

settings .py

- **DEBUG** – ustawienie na True, przestawić naszą aplikację w tryb deweloperski,
- **BASE_DIR** – dokładna ścieżka dostępu do projektu,
- **INSTALLED_APPS** – aplikacje Django zawarte w naszym projekcie,
- **DATABASES** – konfiguracja baz danych używanych przez projekt,
- **STATIC_URL** – URL, pod jakim będą serwowane pliki.

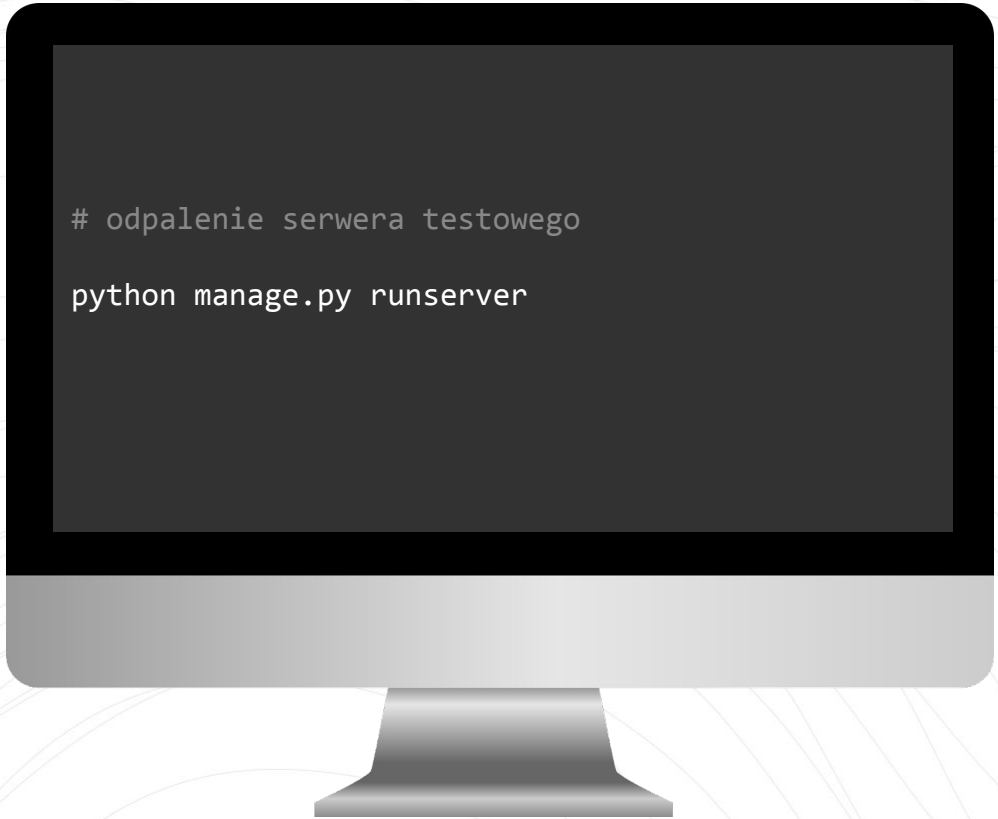
https://docs.djangoproject.com/en/5.0/ref/settings/

Pliki

- **__init__.py**: Pusty plik, który mówi Pythonowi, że ten katalog powinien być uważany za pakiet Pythona.
- **urls.py**: Deklaracje URL-i dla tego projektu Django; „spis treści” twojej strony opartej na Django. Możesz przeczytać więcej o URL-ach w URL dispatcher.
- **asgi.py**: Punkt wejściowy dla serwerów WWW kompatybilnych z ASGI do serwowania twojego projektu.
- **wsgi.py**: Punkt wejściowy dla serwerów WWW kompatybilnych z WSGI do serwowania twojego projektu.

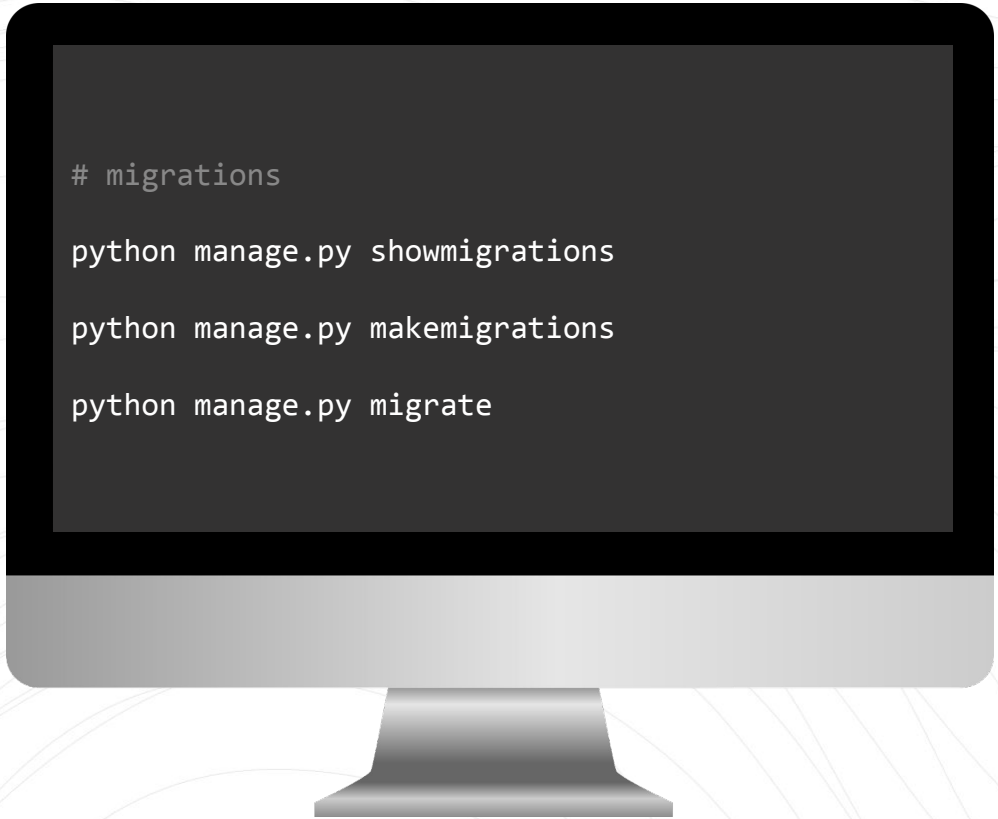
https://docs.djangoproject.com/pl/5.0/intro/tutorial01/

Serwer Testowy



https://docs.djangoproject.com/en/5.0/intro/tutorial01/

Migracje



https://docs.djangoproject.com/en/5.0/intro/tutorial01/

Migracje

Komenda migrate spogląda w ustawienie INSTALLED_APPS i tworzy potrzebne tabele bazy danych w nawiązaniu do ustawień bazy danych w twoim pliku infoshare/settings.py i do migracji bazy danych zawartych w aplikacji. Zobaczysz wiadomość o każdej migracji, która zostanie zastosowana.

```
(venv) → pythonProject python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
```

https://docs.djangoproject.com/pl/5.0/intro/tutorial02/

Migracje

Framework bierze na siebie ich obsługę, uwalniając nas od konieczności programowania w języku SQL. Zmiany w bazach danych (tworzenie i modyfikowanie tabel, operacje na danych, itp.) tworzone są przy pomocy Pythona.

Programista tworzy strukturę bazy danych, pisząc klasy w Pythonie, następnie musi uruchomić skrypt, który skonwertuje jego kod na strukturę bazy danych i założy (bądź zmodyfikuje) odpowiednie tabele.

https://docs.djangoproject.com/pl/5.0/intro/tutorial01/

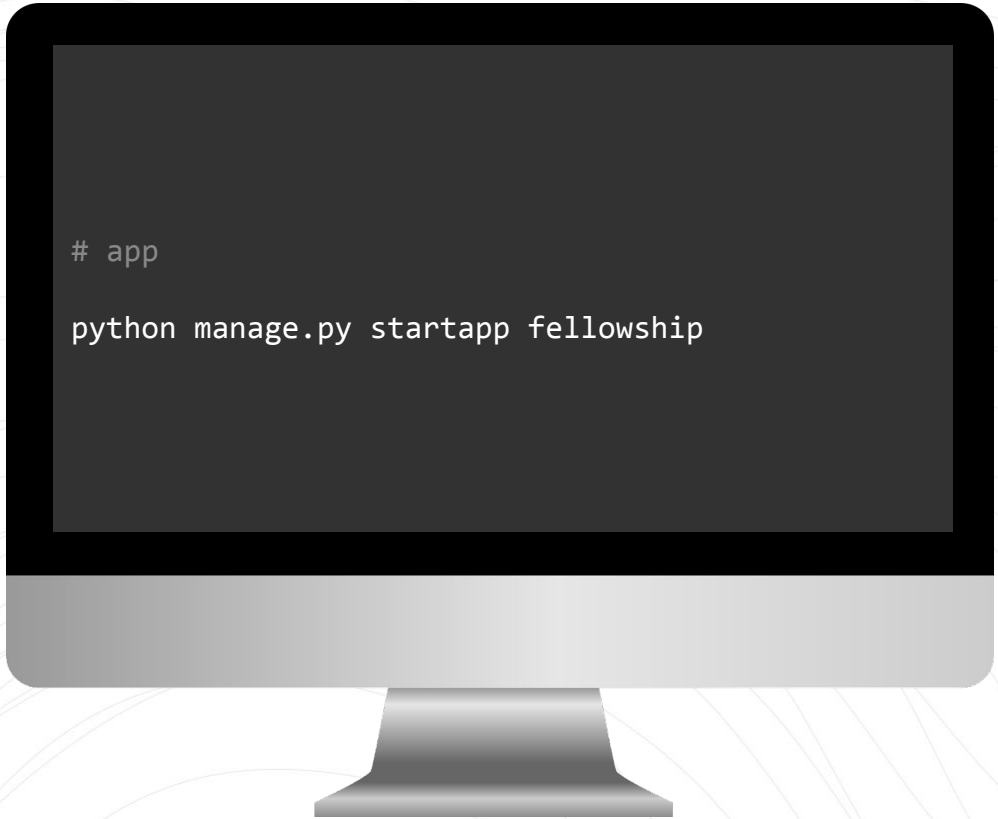
Aplikacje

Istotą projektu Django są aplikacje. Są to osobne foldery, zorganizowane według funkcjonalności. Każdy projekt Django musi zawierać przynajmniej jedną aplikację. Jeden projekt może posiadać wiele aplikacji. Żadna aplikacja nie może nazywać się tak jak projekt.

Każda aplikacja, którą piszesz w Django, składa się z pakietu Pythona, który realizuje pewną konwencję. Django zawiera narzędzie, które automatycznie generuje podstawową strukturę katalogów aplikacji, abyś mógł skupić się na pisaniu kodu zamiast na tworzeniu katalogów.

https://docs.djangoproject.com/pl/5.0/intro/tutorial01/

Aplikacja



https://docs.djangoproject.com/en/5.0/intro/tutorial01/

Aplikacja

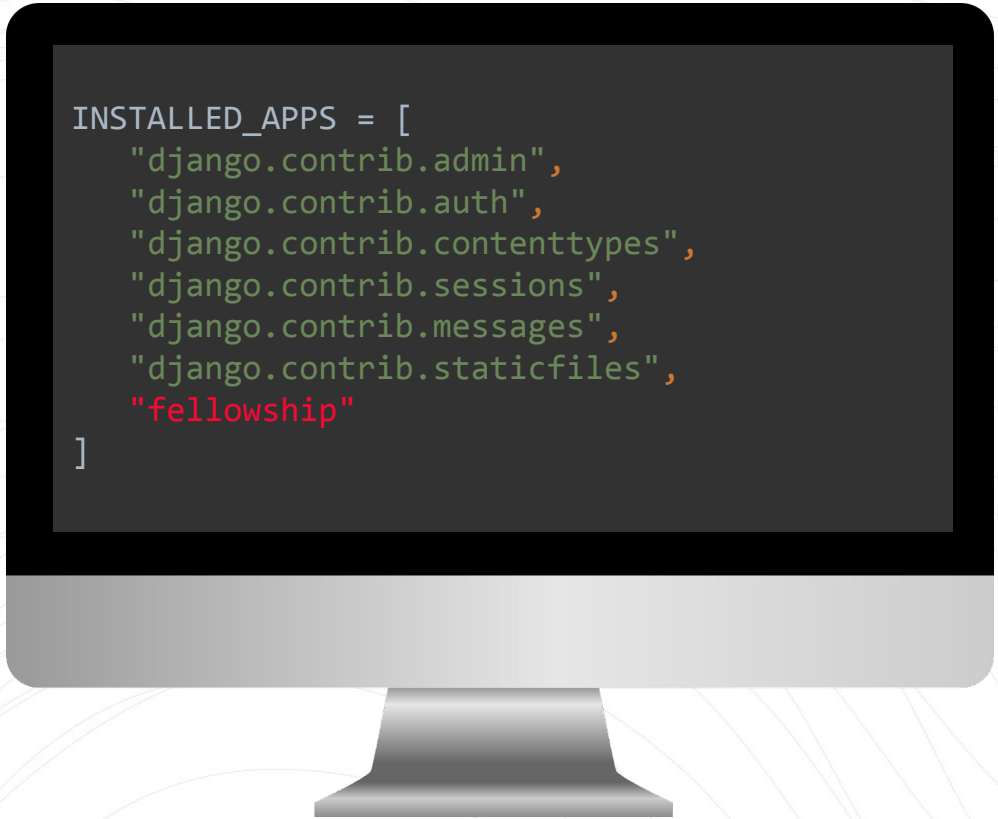
https://docs.djangoproject.com/en/5.0/intro/tutorial01/

Aplikacje

- **__init__.py** – standardowy plik inicjujący moduł,
- **admin.py** – plik z funkcjonalnością admina,
- **apps.py** – plik z tzw. „rejestrem aplikacji”,
- **models.py** – plik z modelami bazy danych,
- **tests.py** – plik z testami automatycznymi,
- **views.py** – plik z definicjami widoków.

https://docs.djangoproject.com/pl/5.0/intro/tutorial01/

Aplikacja

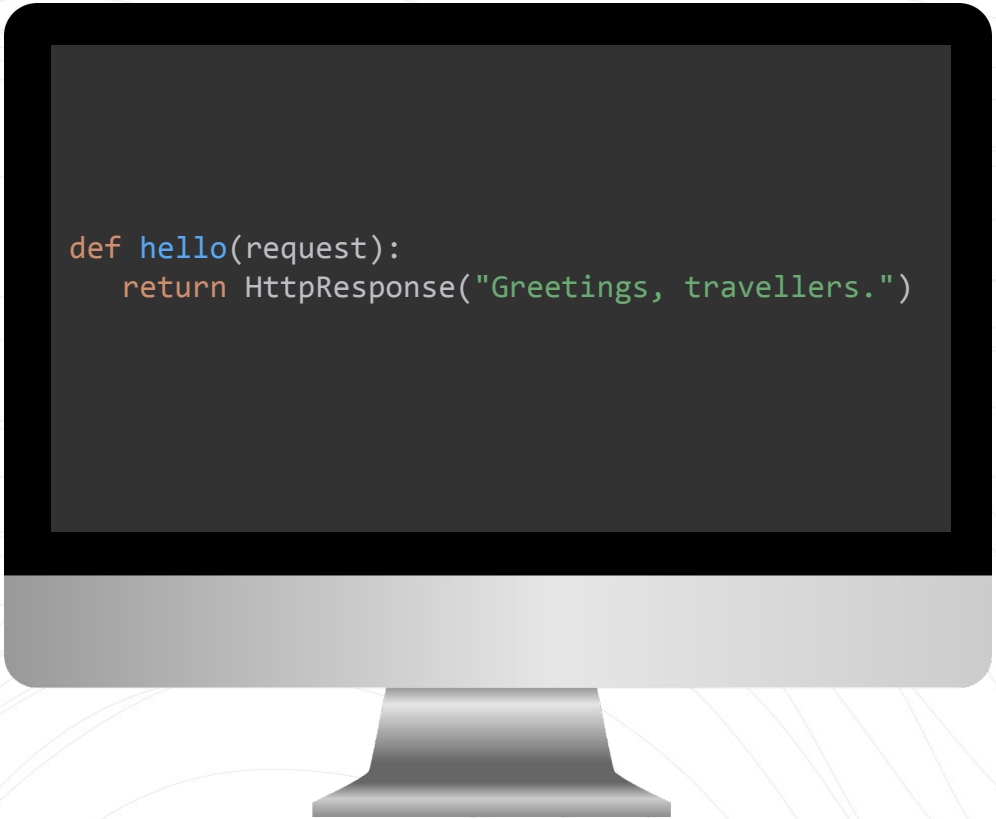


https://docs.djangoproject.com/en/5.0/intro/tutorial01/

03. Rivendell

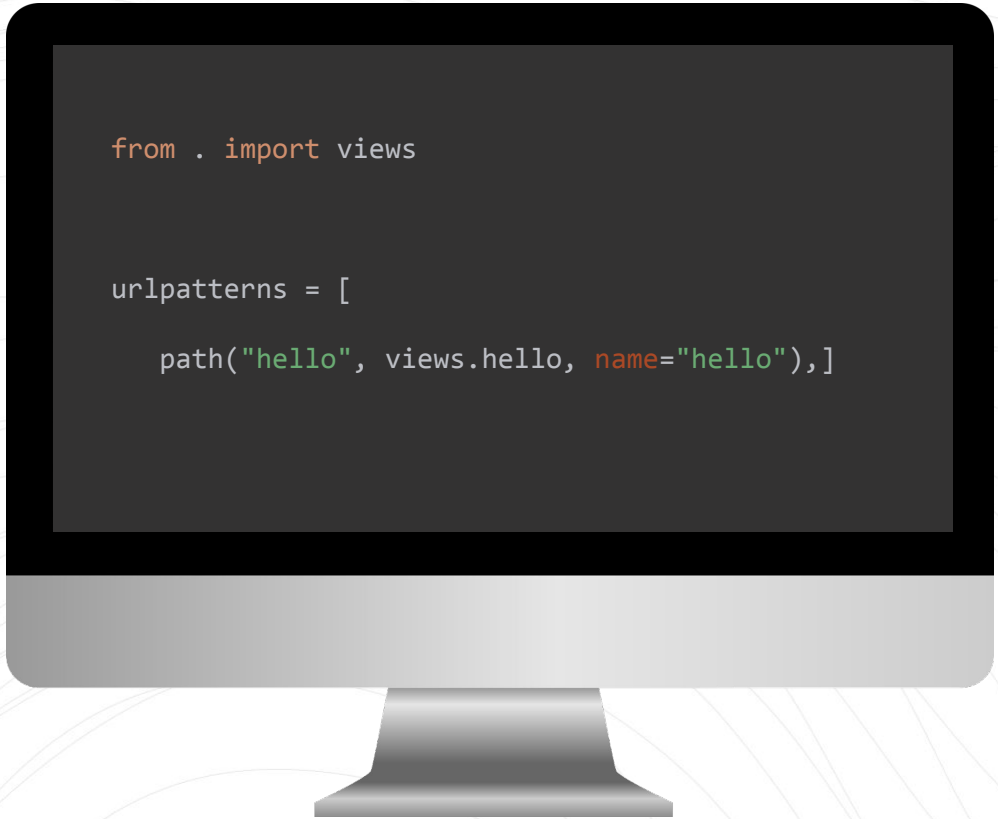
Widoki

views.py



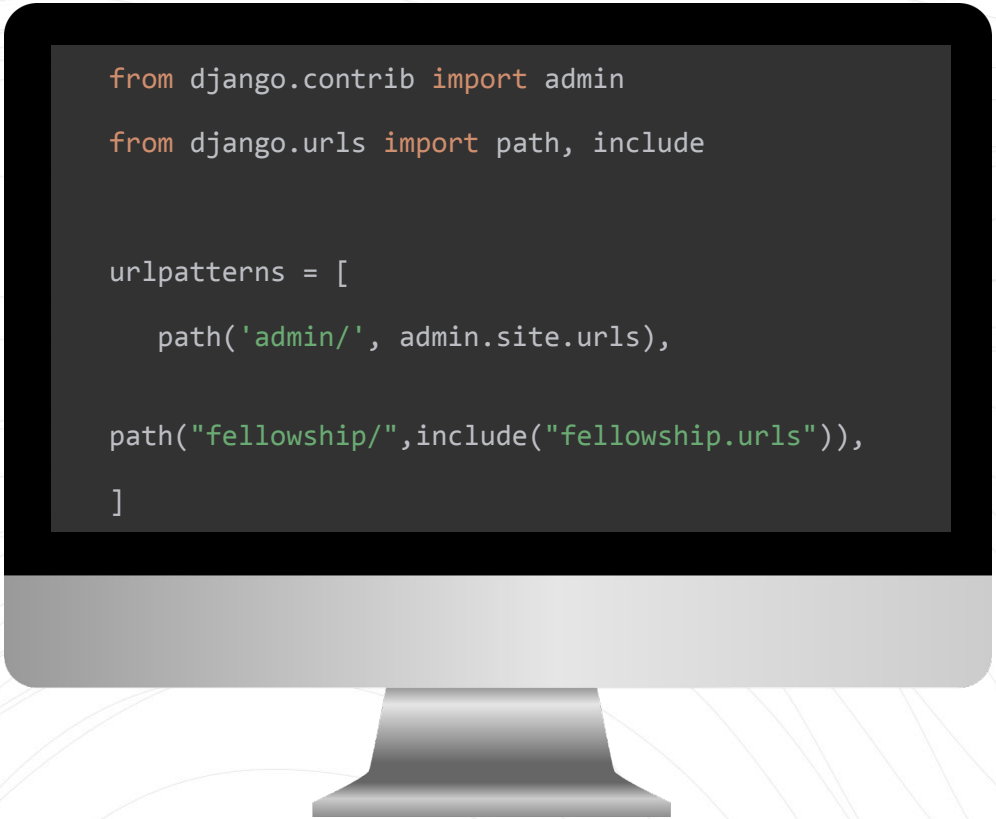
https://docs.djangoproject.com/en/5.0/intro/tutorial01/

hello/urls.py



https://docs.djangoproject.com/en/5.0/intro/tutorial01/

urls.py



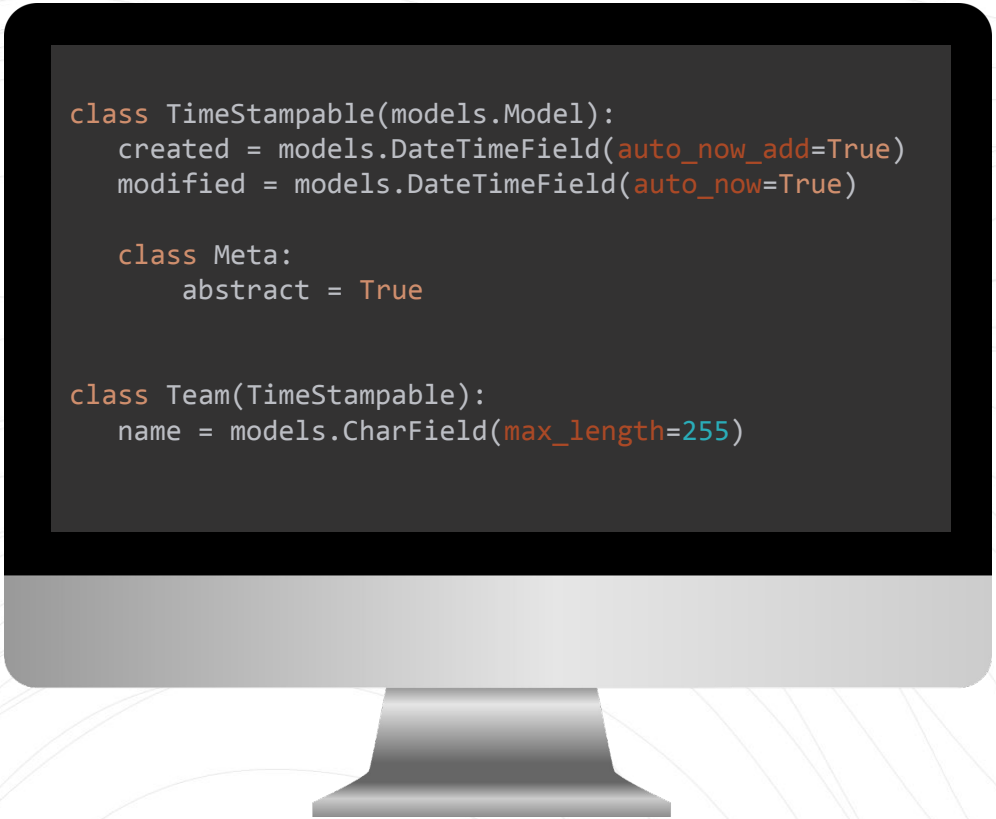
https://docs.djangoproject.com/en/5.0/intro/tutorial01/

04. Rivendell

Modele

infoShareAcademy.com

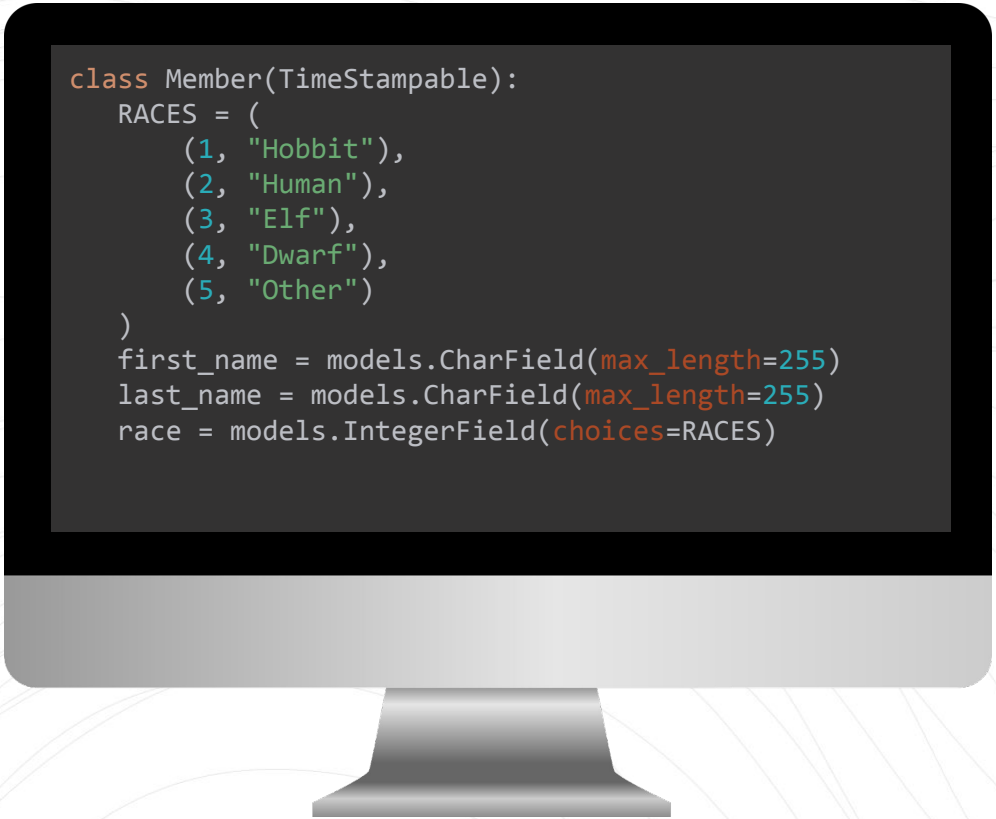
Modele



```python
class TimeStampable(models.Model):
    created = models.DateTimeField(auto_now_add=True)
    modified = models.DateTimeField(auto_now=True)

    class Meta:
        abstract = True


class Team(TimeStampable):
    name = models.CharField(max_length=255)
```

https://docs.djangoproject.com/en/5.0/intro/tutorial01/

Modele

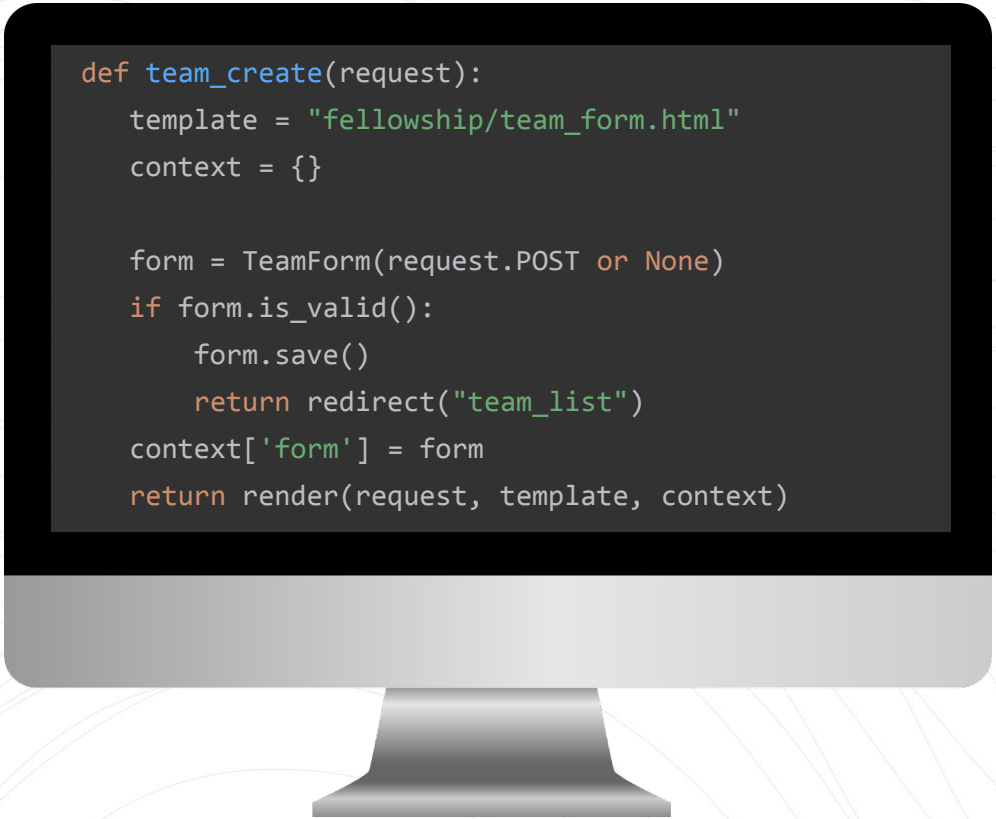


https://docs.djangoproject.com/en/5.0/intro/tutorial01/

05. Moria i Amon Hen

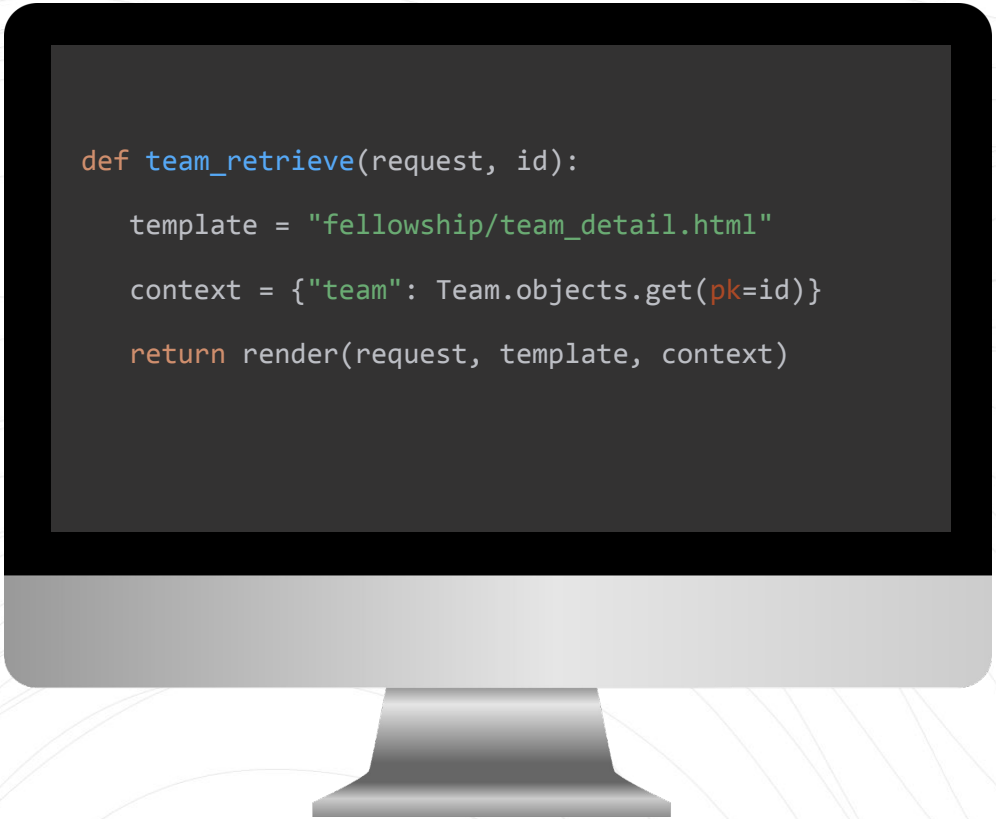
Widoki -CRUD

Widoki



https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

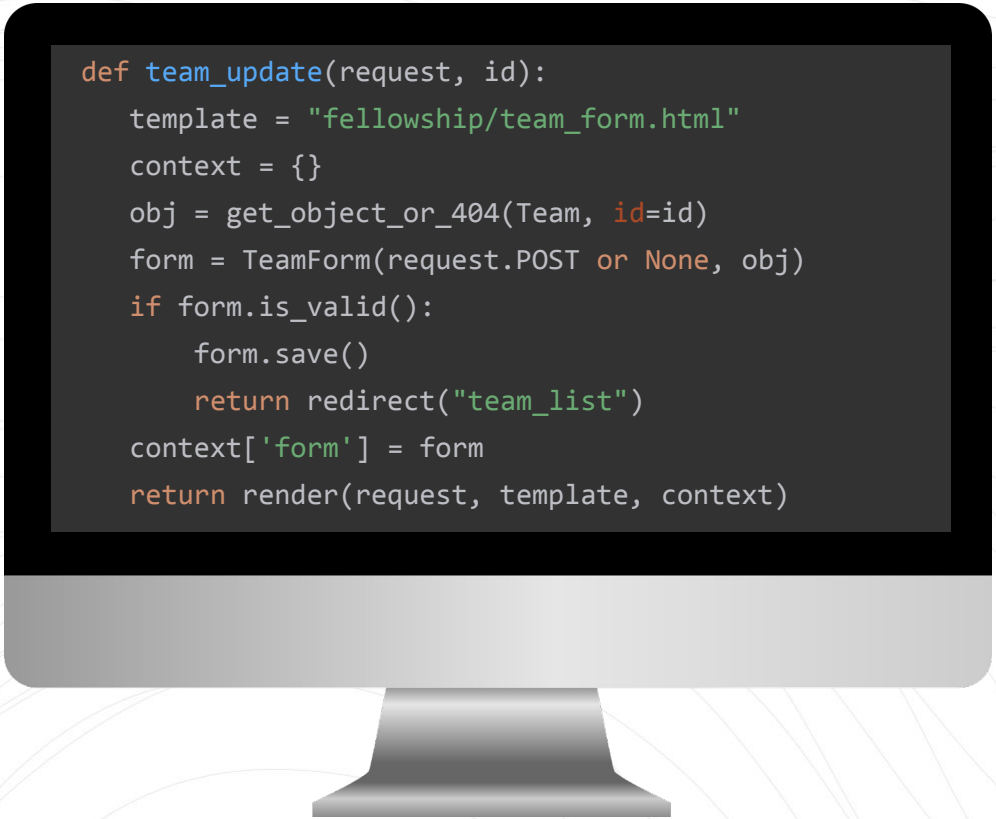
Widoki



```
def team_retrieve(request, id):
    template = "fellowship/team_detail.html"
    context = {"team": Team.objects.get(pk=id)}
    return render(request, template, context)
```

https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

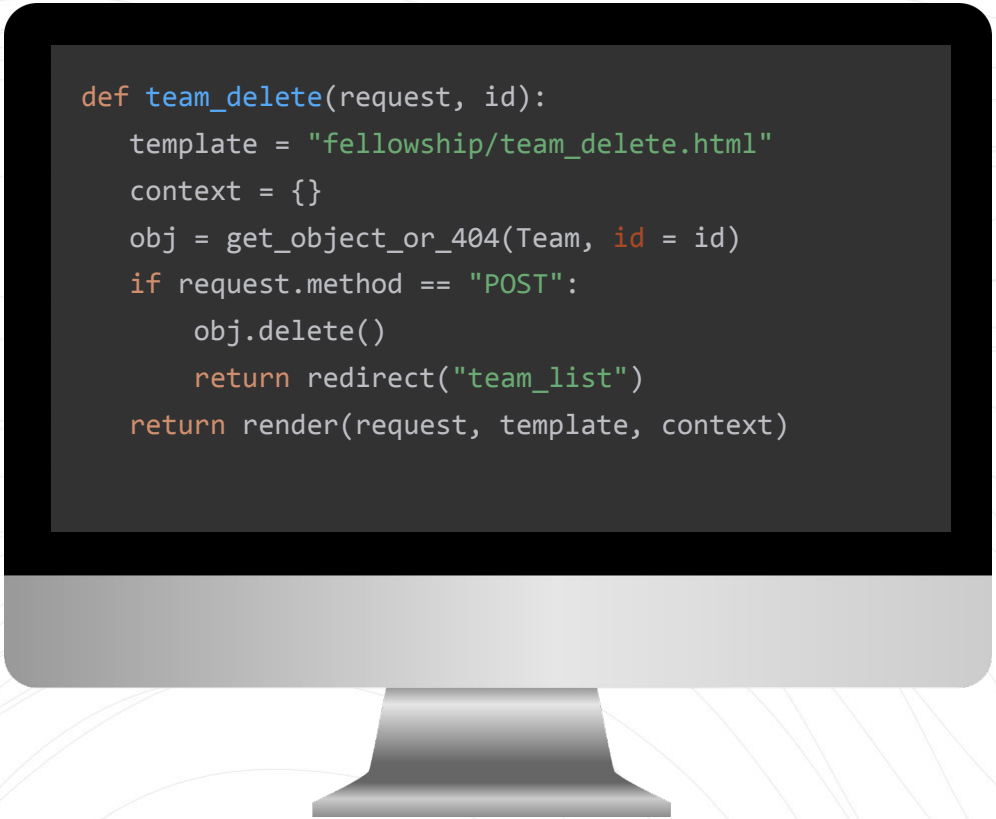
Widoki



```
def team_update(request, id):
    template = "fellowship/team_form.html"
    context = {}
    obj = get_object_or_404(Team, id=id)
    form = TeamForm(request.POST or None, obj)
    if form.is_valid():
        form.save()
        return redirect("team_list")
    context['form'] = form
    return render(request, template, context)
```

https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

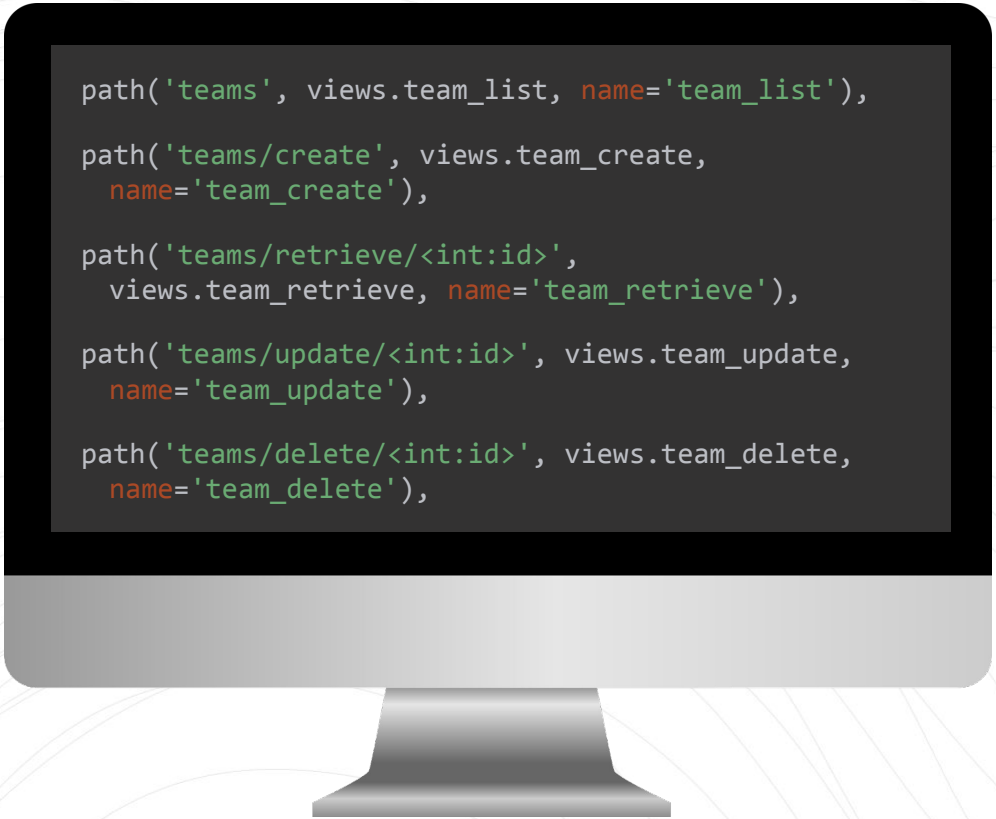
Widoki



```
def team_delete(request, id):
    template = "fellowship/team_delete.html"
    context = {}
    obj = get_object_or_404(Team, id = id)
    if request.method == "POST":
        obj.delete()
        return redirect("team_list")
    return render(request, template, context)
```

https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

urls.py



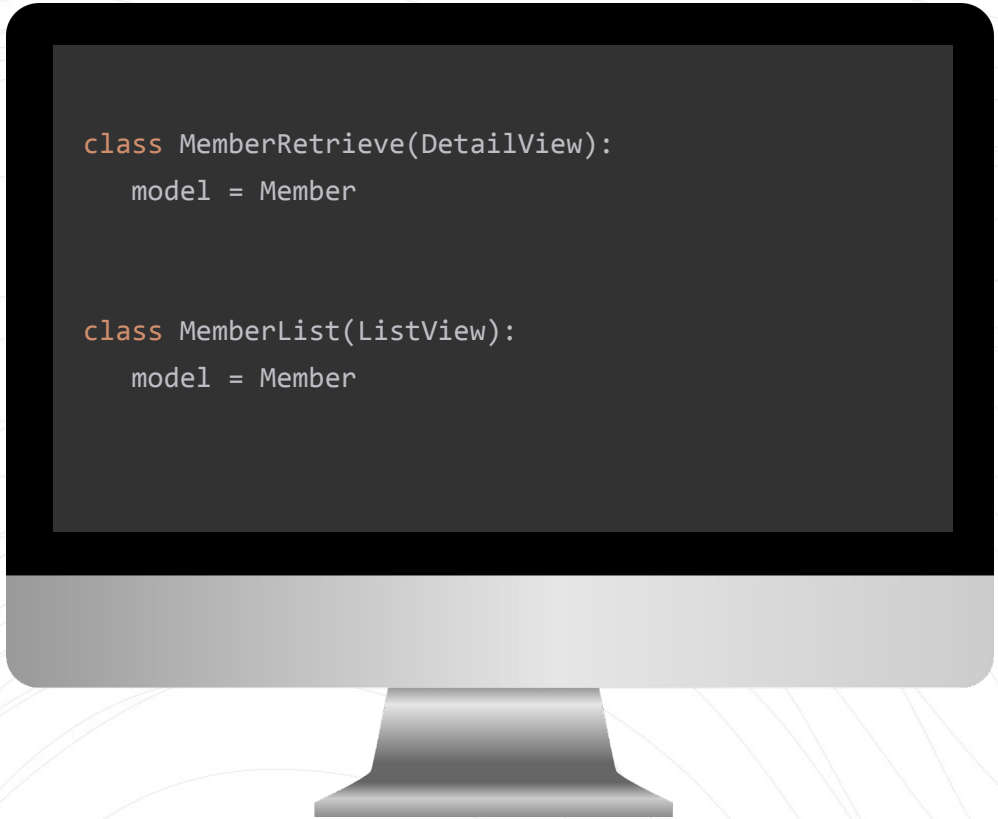
```
path('teams', views.team_list, name='team_list'),

path('teams/create', views.team_create,
  name='team_create'),

path('teams/retrieve/<int:id>',
  views.team_retrieve, name='team_retrieve'),

path('teams/update/<int:id>', views.team_update,
  name='team_update'),

path('teams/delete/<int:id>', views.team_delete,
  name='team_delete'),
```

https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

Widoki

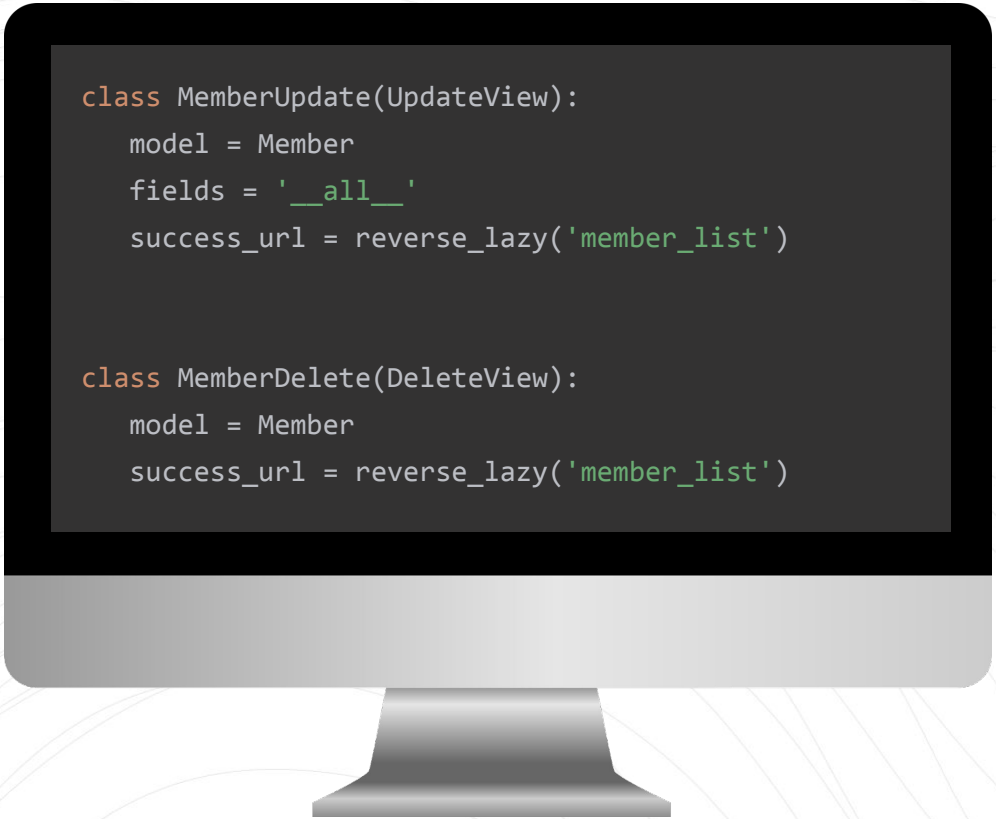
https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

Widoki



https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

Widoki



https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

urls.py

```
path('members',
  views.MemberList.as_view(),name='member_list'),

path('members/create',
  views.MemberCreate.as_view(),name='member_create'),

path('members/retrieve/<int:pk>',
  views.MemberRetrieve.as_view(),name='member_retrie'),

path('members/update/<int:pk>',
  views.MemberUpdate.as_view(), name='member_update'),

path('members/delete/<int:pk>',
  views.MemberDelete.as_view(), name='member_delete'),
```

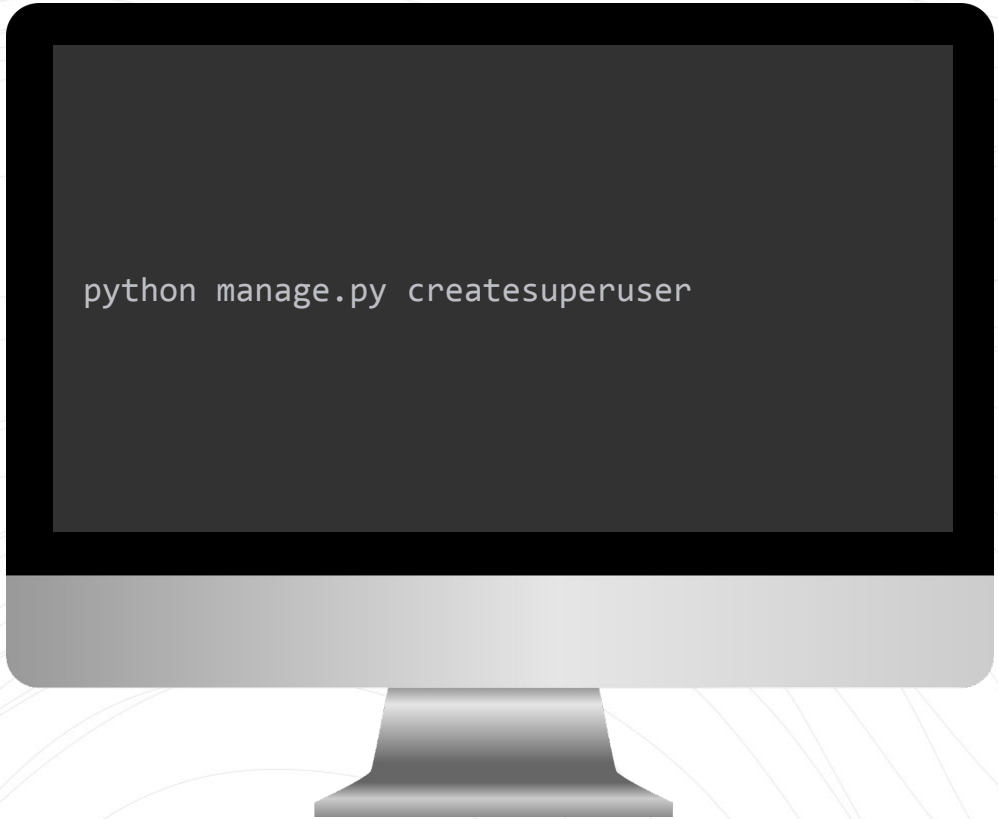
https://docs.djangoproject.com/pl/5.0/intro/tutorial03/

One Ring to Rule Them All

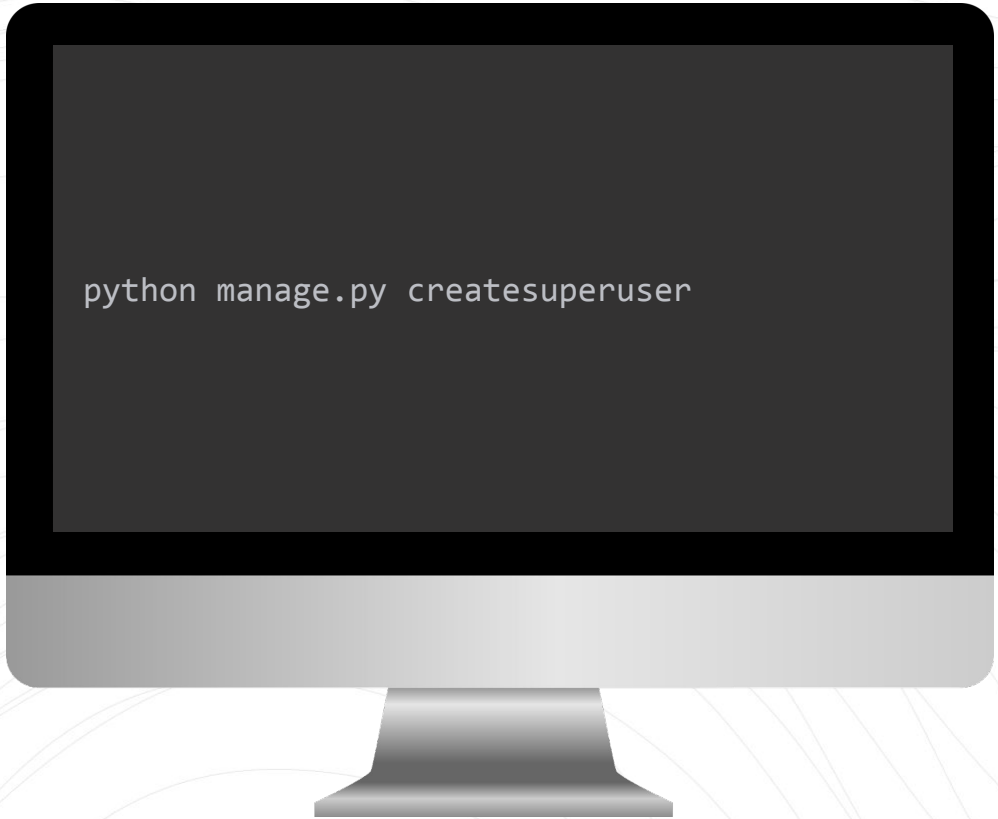
Django Admin

infoShareAcademy.com

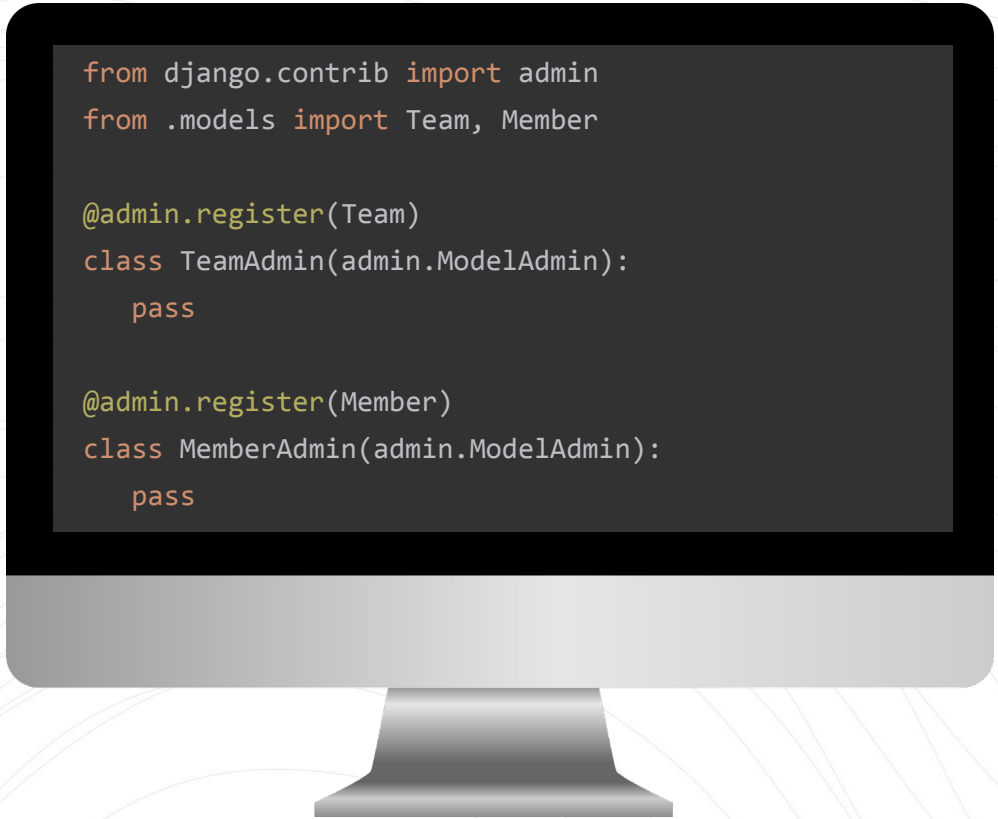
createsuperuser



createsuperuser



admin.py



```python
from django.contrib import admin
from .models import Team, Member

@admin.register(Team)
class TeamAdmin(admin.ModelAdmin):
    pass

@admin.register(Member)
class MemberAdmin(admin.ModelAdmin):
    pass
```

Links

https://tutorial.djangogirls.org/en/

https://jeffkit.gitbooks.io/django-girls-tutorial/content/pl/django_start_project/index.html

https://docs.djangoproject.com/en/5.0/

THANK YOU FOR YOUR ATTENTION

infoShareAcademy.com