Christophe Brun <christophe@cbrun.fr>

July 21, 2026

Introduction

- 1ère release : Juillet 2005
- Version : 6.0
- MVC : Model View Controller
- Grosse communauté

Création du projet

- Création du répertoire du projet

```
mkdir b2b
```

- Création du projet avec admin django

```
django-admin startproject setup  b2b
```

- Setup : Premier répertoire de l'application contient le settings, le routeur ...
- manage.py : cli de gestion
- Arborescence

```
b2b
  |- setup/
  |- manage.py
```

- Démarrage du projet

```
python manage.py runserver
```

- Go sur http://127.0.0.1:8000/

Création de la première app

- Utilisation de l'outil manage

```
python ./manage.py startapp contrat
```

- contrat étant le nom de l'app.
- Arborescence

contrat étant le nom de l'app. Arborescence

```
b2b
  |- setup/
  |- manage.py
  |- contrat
        |- __init__.py
        |- admin.py
        |- pps.py
        |- migrations/
        |- models.py
        |- tests.py
        |- views.py
```

Activation de l'application

Dans le fichier setup, il faut activer l'application/modules, ... Si on lance, on va avoir des warning, erreur, car il n'y a pas de base de donénes, pas de migrations, ...

Model

- Création d'un modèle: models.py

```
ENERGIE = {
    'GAZ': 'Gaz',
    'ELE': 'Electrique',
}

class Client(models.Model):
    nom = models.CharField(max_length=100)
    siret = models.CharField(max_length=20)
    email_contact = models.EmailField()
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return f'{self.nom}'

class Contrat(models.Model):
    libelle = models.CharField(max_length=100)
    client = models.ForeignKey(Client, on_delete=models.CASCADE)
    energie = models.CharField(max_length=3, choices=ENERGIE)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return f'{self.libelle} - {self.client.nom} - {self.energie}'
```

Migrations

- Migration et création de la base de données

```
(django) ->  b2b python ./manage.py makemigrations contrat
Migrations for 'contrat':
  contrat/migrations/0001_initial.py
    + Create model Client
    + Create model Contrat
```

```
(django) ->  b2b python ./manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, contrat, 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 contrat.0001_initial... OK
  Applying sessions.0001_initial... OK
```

Super User et BO

```
(django) ->  b2b python manage.py createsuperuser
Username (leave blank to use 'christophe.brun'): admin
Email address: admin@ilek.fr
Password:
Password (again):
The password is too similar to the username.
This password is too short. It must contain at least 8 characters.
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

Go sur http://localhost:8000/admin/login/?next=/admin/ Ajout de nos models dans admin.py

```
from django.contrib import admin
from .models import *
# Register your models here.

admin.site.register(Client)
admin.site.register(Contrat)
```

Un BO un peu plus stylé : Baton

Activation pour avoir un BO plus sympa

API Rest : DjangoRestFramework

Un framework dans un framework avec plein de fonctionnalités

- Versionning : plusieurs mode (URI, Host, Header, ...) voir custom
- Gestion de limitations d'appels : configuration globale, par endpoint, par user, par scope ou custom
- Cache : idem complètement configurable
- Auto Documation Swagger avec UI
- Serializers : pour l'ensemble des types de données, et pour les custom
- Gestion des droits, filtres de contenus, pagniation, negociation de contenu ...

Ressources

- Site Django : https://www.djangoproject.com/
- Docs très précises : https://docs.djangoproject.com/fr/6.0/
- DjangoRestFramework : https://www.django-rest-framework.org/
- Repo de packages : https://djangopackages.org/