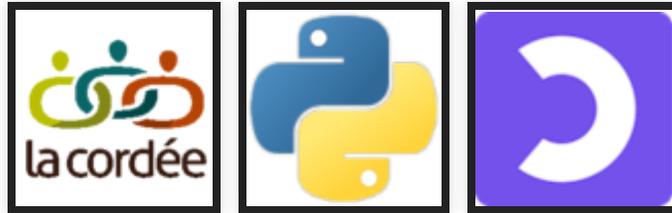


12 mois (moins un jour) à la découverte de **Python**
avec *OpenClassrooms*



Mon 1er Django en bricolant avec (entre autre) la base **openfoodfacts.org**



LA CORDÉE





- communauté de travail != #coworking
- du WIFI
- du café



... mais surtout :

- du partage
- une communauté IRL / virtuelle
- des jeux de mots / des goûters (de folie !)
- <https://www.la-cordee.net>

ME, MYSELF & I

ME, MYSELF & I

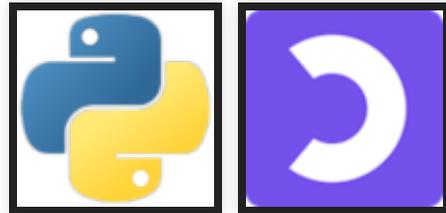


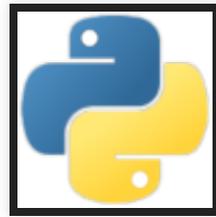




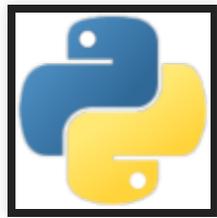


PYTHON & OPENCLASSROOMS





Pourquoi apprendre Python ?

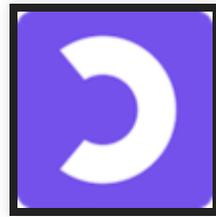


Sam & Max !



Sam & Max :

Python : meilleur nulle part, excellent partout.

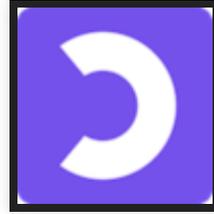


www.openclassrooms.com?



www.openclassrooms.com:

*OpenClassrooms est une école en ligne
offrant des parcours diplômants et
professionnalisants à plus de trois
millions d'étudiants chaque mois à
travers le monde.*



LE PARCOURS DAPY :

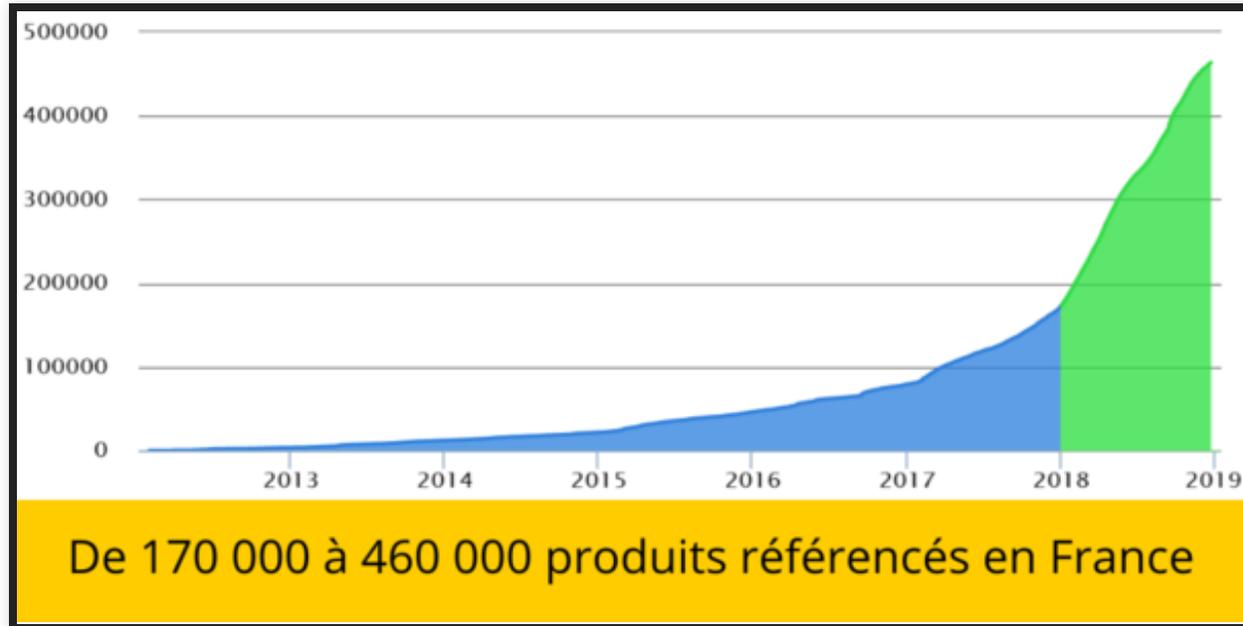
- Mentor individuel
- Diplôme RNCP 2 (bac +3/4)
- Durée flexible (12 mois)
- 13 projets

OPENFOODFACTS.ORG ?





- Depuis 2012
- Une base de produits alimentaires
- **Par & pour** tous
- Décoder les étiquettes
- Trouver des produits
- Comparer et changer
- Explorer et découvrir







DONNEZ DES SIOUX !

1447

Sante

2 005 €

1447 gouttes d'eau données à ce projet ?

*Donner vos gouttes d'eau
Je donne actuellement
mes gouttes au projet
Open Food Facts*

Un financement récurrent avec le moteur Lilo

A graphic with a white background and a black border. On the left, there is a green outline of a person's head and shoulders. A blue arrow points from the person towards the right. Inside the arrow, there is a yellow factory icon, the text "C'est emballé" (with "emballé" in a red box), and "près de chez vous". Below the arrow, the text "La carte des produits alimentaires" is written in green.

C'est **emballé**
près de chez vous

La carte des produits alimentaires

LE PROJET #8

(enfin !)

*Proposer un site web qui permet
trouver de "meilleurs" produits
alimentaires*

CONTRAINTE

- Django
- Heroku
- PostgreSQL
- Bootstrap
- Github

DÉCOUVERTE DE...

- *PostgreSQL*
- *Bootstrap*
- *ORM*
- *Django (!)*

DANS MA VALISE...



- DRY ... <3
- tests ... <3
- *Heroku* ... <3
- **API OpenFoodFacts**
- Github
- *comprehension* list, dict, foo, ... <3

LE MODÈLE DE DONNÉES

Physical data model

«django»
User

PK: **id** [INT NOT NULL UNSIGNED]

email [VARCHAR(254) NOT NULL]

username [VARCHAR(150) NOT NULL]

password [VARCHAR(128) NOT NULL]

Physical data model

Product
PK: id [INT NOT NULL UNSIGNED]
FK: category_id [INT NOT NULL UNSIGNED]
code [VARCHAR(13) UNIQUE NOT NULL] image_front_thumb_url [VARCHAR(255)] image_front_url [VARCHAR(255)] image_nutrition_url [VARCHAR(255)] name [VARCHAR(200)] nutrition_grades [VARCHAR(1)] url [VARCHAR(255)]

«django» User
PK: id [INT NOT NULL UNSIGNED]
email [VARCHAR(254) NOT NULL] username [VARCHAR(150) NOT NULL] password [VARCHAR(128) NOT NULL]

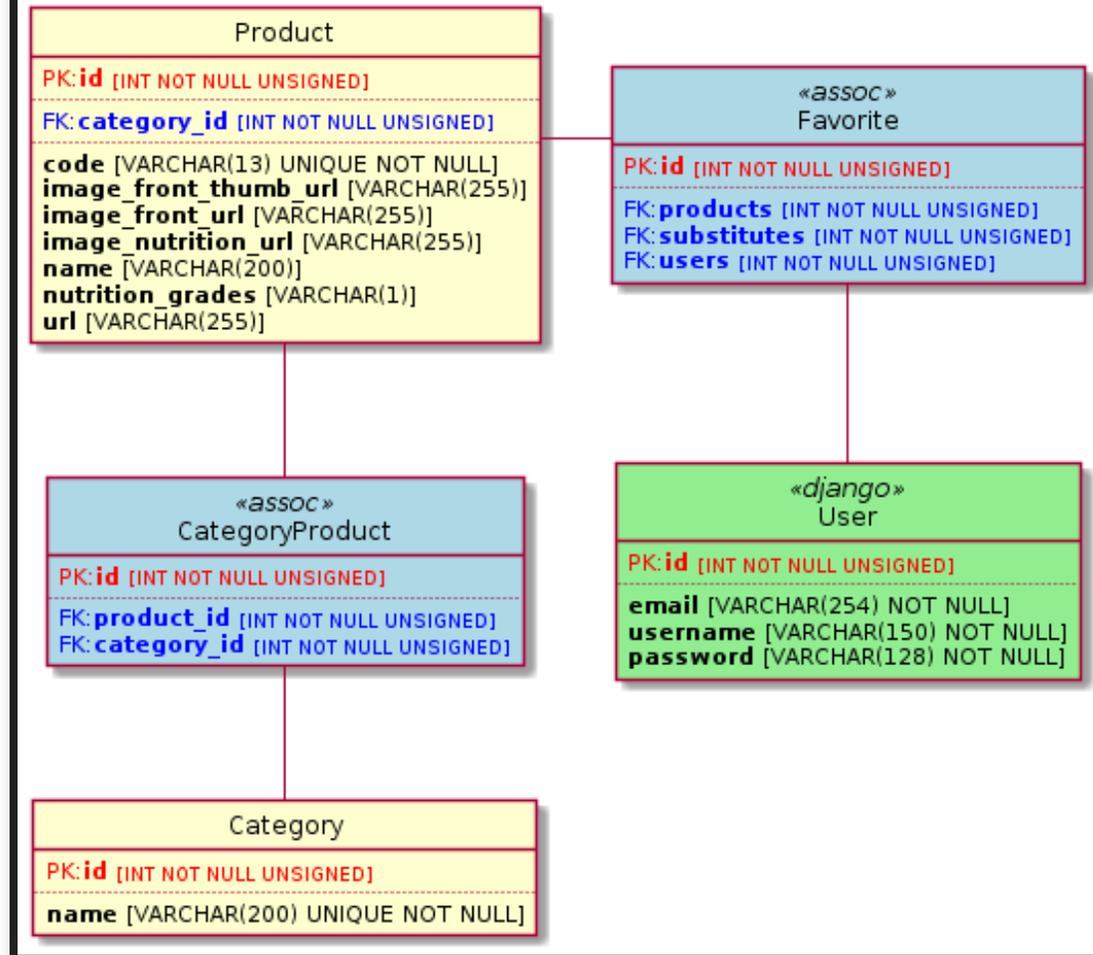
Physical data model

Product
PK: id [INT NOT NULL UNSIGNED]
FK: category_id [INT NOT NULL UNSIGNED]
code [VARCHAR(13) UNIQUE NOT NULL] image_front_thumb_url [VARCHAR(255)] image_front_url [VARCHAR(255)] image_nutrition_url [VARCHAR(255)] name [VARCHAR(200)] nutrition_grades [VARCHAR(1)] url [VARCHAR(255)]

«django» User
PK: id [INT NOT NULL UNSIGNED]
email [VARCHAR(254) NOT NULL] username [VARCHAR(150) NOT NULL] password [VARCHAR(128) NOT NULL]

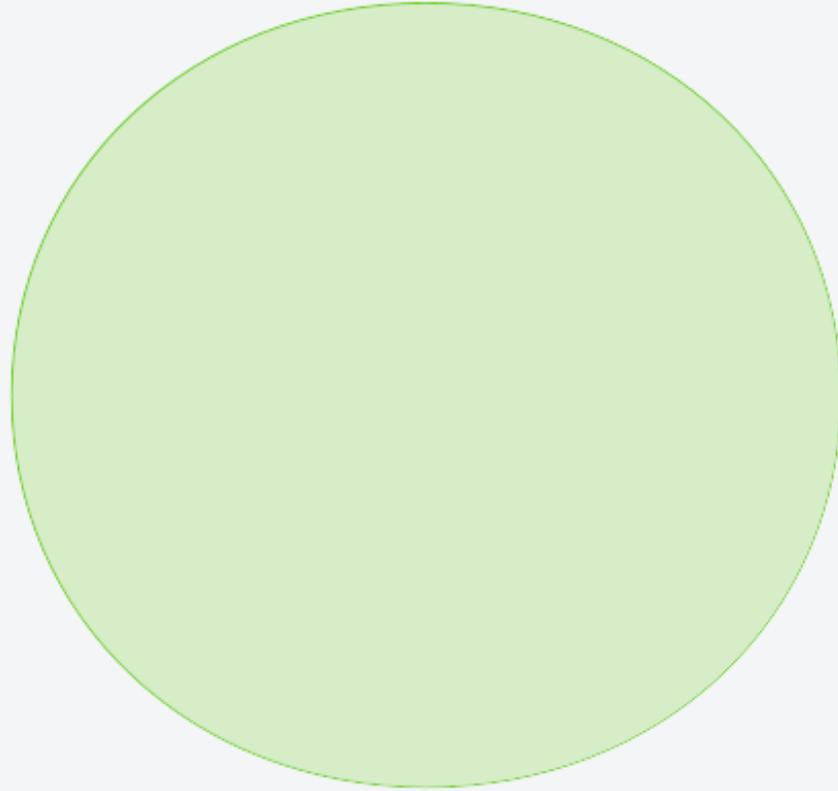
Category
PK: id [INT NOT NULL UNSIGNED]
name [VARCHAR(200) UNIQUE NOT NULL]

Physical data model

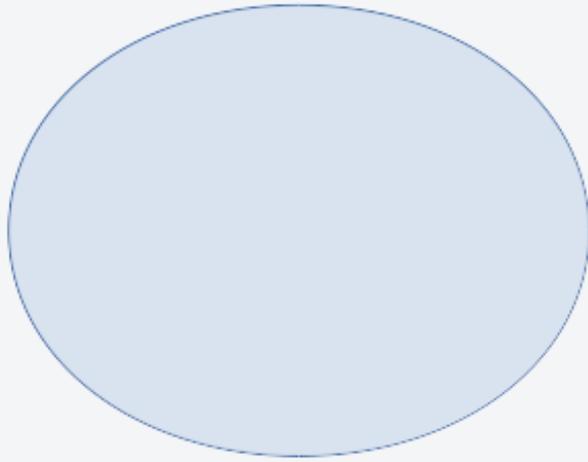


FACILE, MAIS ...

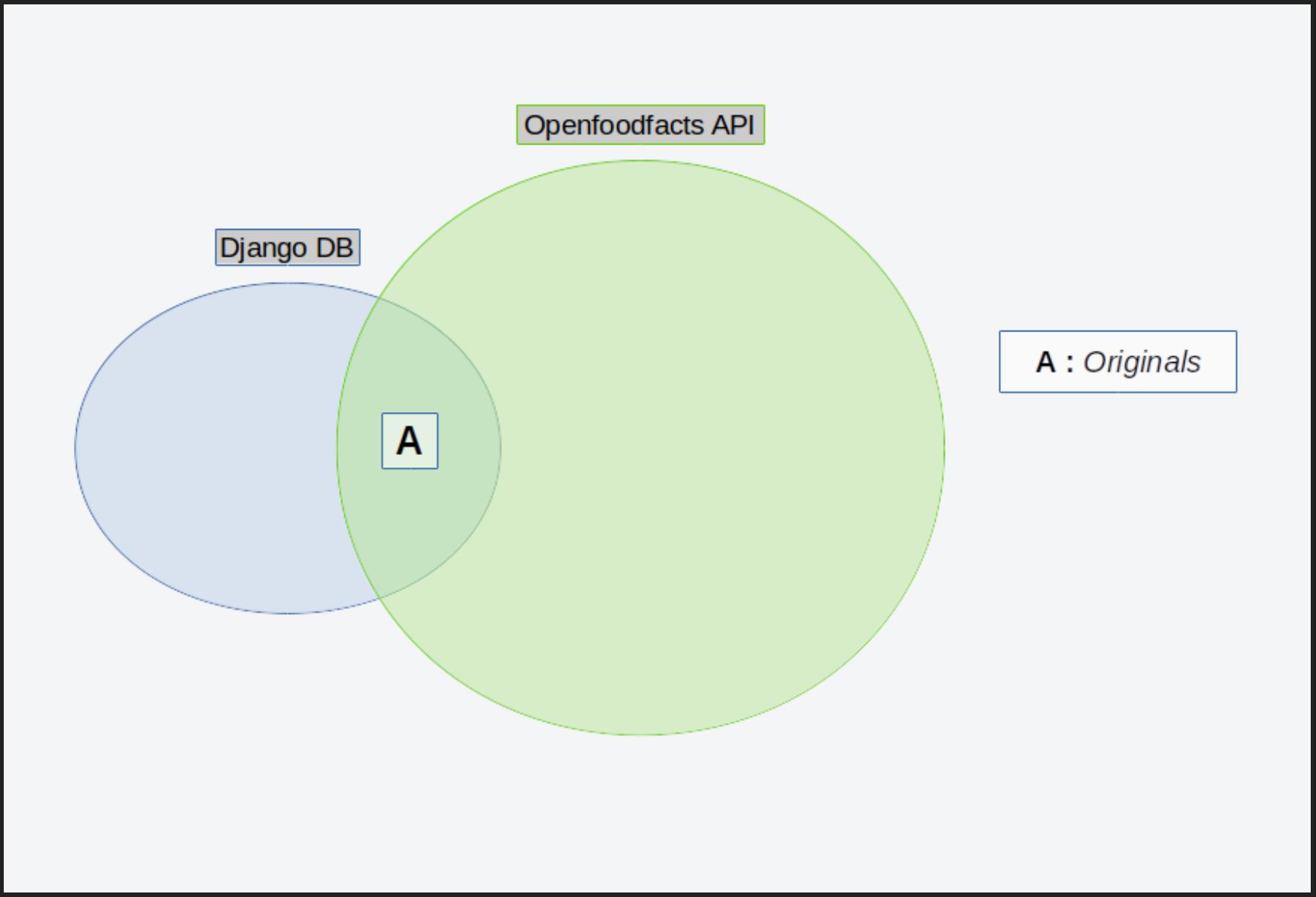
Openfoodfacts API

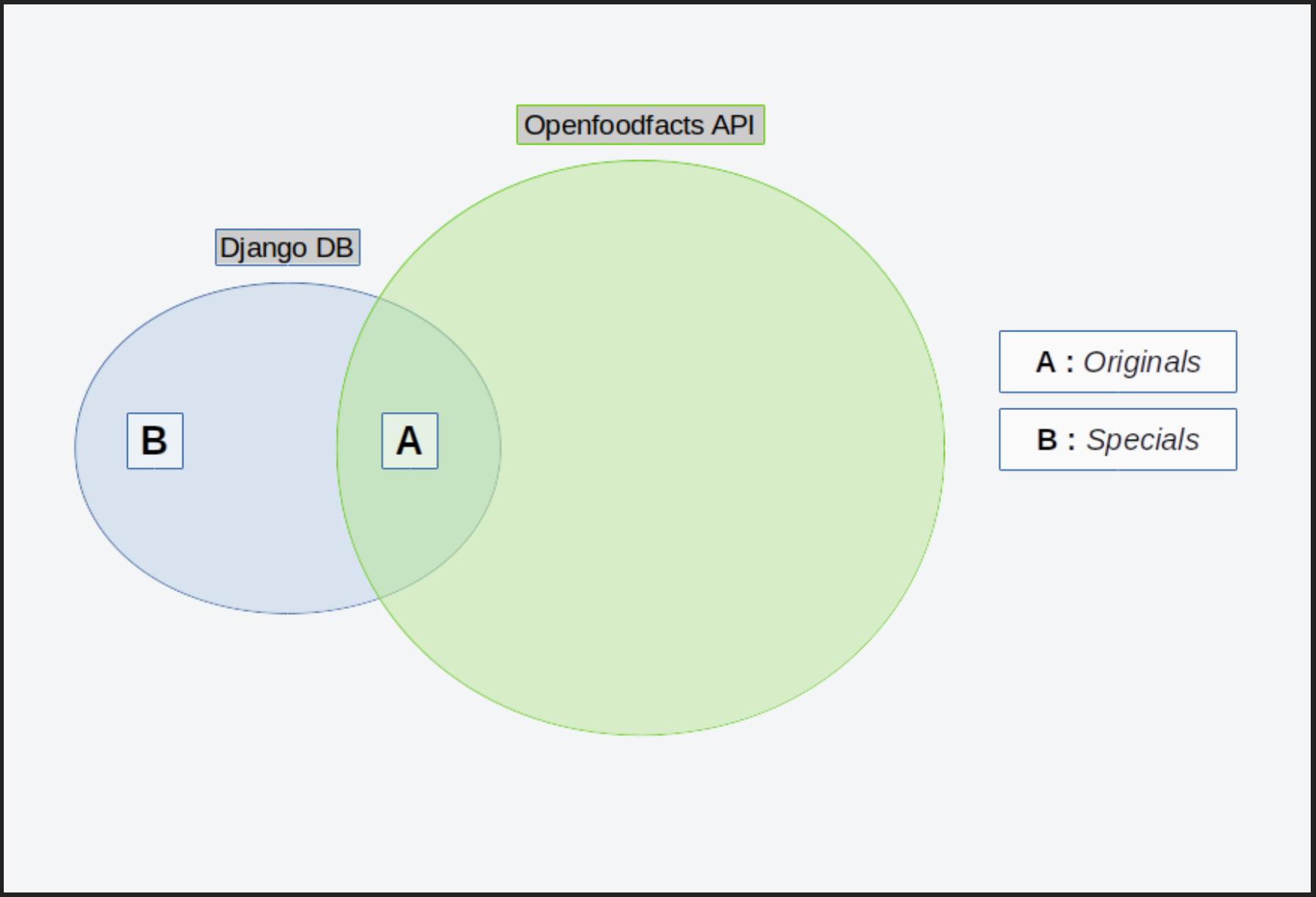


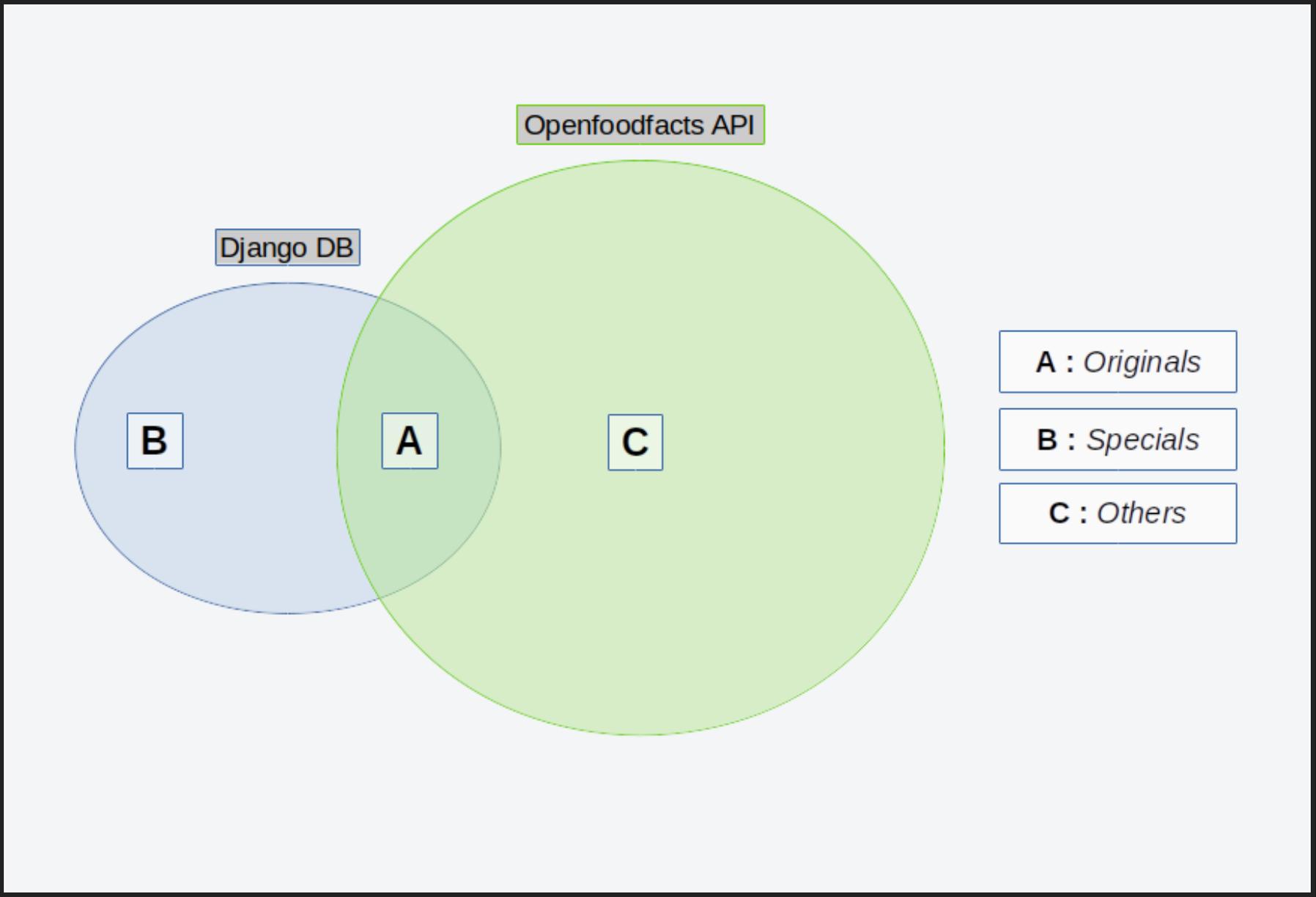
Django DB











Product models

```
from django.db import models
from django.contrib.auth.models import User

class Category(models.Model):
    name = models.CharField(max_length=200, unique=True)

    def __str__(self):
        return self.name

class Product(models.Model):
    """
    Most attributes are original OpenFF API `product field`
    Other are specific fields processed , see @staticmethod in
    ersatz.view.toolbox.SearchProduct to process this fields
    """
```

Configuration DRY ?

```
"""
This file is part of [ocp8](https://github.com/freezed/ocp8) p
"""
from ersatz.models import Product

API = {
    'URL_SEARCH': 'https://fr.openfoodfacts.org/cgi/search.pl',
    'PARAM_SEARCH': {
        'search_simple': 1,
        'action': 'process',
        'json': 1,
    },
    'NO_PROD': 'OpenFoodFacts ne donne pas de produits pour la
    'EMPTY': 'Bizarrement OpenFoodFacts ne donne pas de produi
}
```

Traitement des champs d'API

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
This file is part of [ocp8](https://github.com/freezeed/ocp8) p
A place for Objects & Functions needed in the View of the ersa
All are serving `ersatz.views.views.py`
"""
from pprint import pformat as pf
import urllib.parse as up

import requests

from django.contrib.auth.models import User
```

DU VDDD AU TDD !

Tests

```
import os
import json
import urllib.parse as up
from ersatz.views import toolbox as toolbox
from ersatz.views import views as views
from ersatz.config import API, PRODUCT_FIELD

#####
#   ersatz.views.toolbox.get_search_context()
#####

# For a invalid user request, app returns a JSON standardised
class FakeUserRequestGet:
    def __init__(self, QUERY_STRING):
        parts = up.parse_qs(QUERY_STRING)
```

from __future__ import ideas

- Coverage 100% ! (TDD, DB & Selenium) <3
- Bascule vers l'API si pas de substitut en DB
- MàJ DB vs. OFF (admin <3)
- Ajout de critères de substitution (modèle OO)
- ... et plus ! (cf. issues)

DIFFICULTÉS RENCONTRÉES

- Tests dans le contexte *Django*
- Découverte de *Bootstrap*
- Héritage de templates et position de la logique
- Comprendre l'arborescence *Django*
- **rester dans dans les spec client (même si...) !**

**QUESTIONS, REMARQUES,
PLAINTES ?**

...

Merci !!



<http://pro.zind.fr>