Aaron Sanmartin Marin

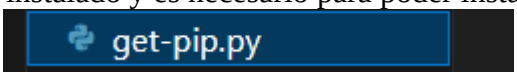
Proyecto Semáforo Focus Soft

El proyecto Semáforo Energético es una aplicación diseñada para ofrecer información, de forma clara y visual, sobre el precio de la energía eléctrica a lo largo del día. El semáforo presenta tres posibilidades dependiendo del precio de la luz, verde(el precio es bajo), amarillo(el precio es intermedio) o rojo(el precio es alto). La aplicación ha sido desarrollada utilizando el framework Django, donde se define la estructura de datos a través del archivo models.py. Un scraper automatizado se encarga de obtener en tiempo real los precios de la electricidad desde fuentes oficiales, y los almacena en una base de datos PostgreSQL gestionada mediante Docker Compose. La aplicación de la aplicación ha sido desarrollada en Java, ofreciendo una interfaz intuitiva que consulta los datos del backend a través de una API REST. Esta API conecta con el modelo de Django, permitiendo obtener los precios y el estado correspondiente del semáforo en función de la hora actual.

Docker:

Archivos que he utilizado para poder crear el docker y usar en el proyecto

Para empezar he descargado un archivo en google que se llama **get-pip.py** para poder instalar pip ya que en Windows no viene instalado y es necesario para poder instalar todos los **requeriments.**



requirements.txt:

```
asgiref==3.8.1
beautifulsoup4==4.13.4
certifi==2025.4.26
charset-normalizer==3.4.2
Django==4.2.22
djangorestframework==3.16.0
esios==0.25.0
idna==3.10
libsaas==0.4
psycopg2-binary==2.9.10
python-dateutil==2.9.0.post0
python-dotenv==1.1.0
requests==2.32.3
six==1.17.0
soupsieve==2.7
sqlparse==0.5.3
typing_extensions==4.14.0
urllib3==2.4.0
```

docker-compose.yml:

Archivo creado para poder crear la base de datos dentro de docker y que este todo unido con la web

```
version: "3.8"

services:
 db:
  image: postgres
  restart: always
  volumes:
   - postgres_data:/var/lib/postgresql/data/
  environment:
   POSTGRES_DB: db     # Aquí defines el nombre de la base de datos
   POSTGRES_USER: postgres
   POSTGRES_PASSWORD: postgres
  ports:
   - "5432:5432"

 web:
  build: .
  command: python manage.py runserver 0.0.0.0:8000
  volumes:
   - .:/code
  ports:
   - "8000:8000"
  depends_on:
   - db

volumes:
 postgres_data:
```

DockerFile:

He usado este archivo para que pueda instalar todo lo incluido en el archivo de requirements.txt para su funcionamiento

```
FROM python:3.9

ENV PYTHONDONTWRITEBYTECODE 1
ENV PYTHONUNBUFFERED 1

WORKDIR /code
COPY requirements.txt /code/
RUN pip install --upgrade pip && pip install -r requirements.txt
COPY . /code/
```

Una vez creado todos los archivos dando igual el orden y teniendo instalado pip tendremos que usar **docker-compose build** y para poder levantar el compose usaremos **docker-compose up**, esto hara que pueda ejecutar los diferentes archivos que hemos creado

Manage.py:

Para poder crear las tablas directamente dentro de la base de datos y en el docker con el comando **docker-compose run web pu manage.py migrate**

Y ya deberíamos de tener todo ya listo

DJANGO
API
APP