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. la cual ha sido desarrollada y probada utilizando Postman. Esta API conecta con el modelo de Django, permitiendo obtener los precios y el estado correspondiente del semáforo en función de la hora actual.

DJANGO
API
DOCKER
APP

Docker-Compose y Base de datos en Docker:

Creación del contenedor en docker y ajuste de los archivos del proyecto django: requirements.txt, settings.py, docker-compose.yml y Dockerfile para que todo funcione correctamente y conecte con el contenedor.

para encender el contenedor en docker, he cambiado los parámetros de estos archivos:

SETTINGS.PY

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': ('db'),
        'USER': ('postgres'),
        'PASSWORD': ('postgres'),
        'HOST': 'db',
        'PORT': '5432',
        'OPTIONS': {'client_encoding': 'UTF8'},
    }
}
```

DOCKER-COMPOSE.YML

```
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:

```
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/
```

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

¡¡IMPORTANTE!!

he metido en la carpeta del proyecto el script get-pip.py para poder instalar pip y que todos los archivos del requeriments.txt se instalen y funcione todo.

para instalarlo, se ejecuta este comando: py get-pip.py (en entorno virtual)

después de esto, ya podemos montar la base de datos en docker:

en la carpeta raíz del proyecto, ejecutamos estos comandos para que se cree y se ejecute:

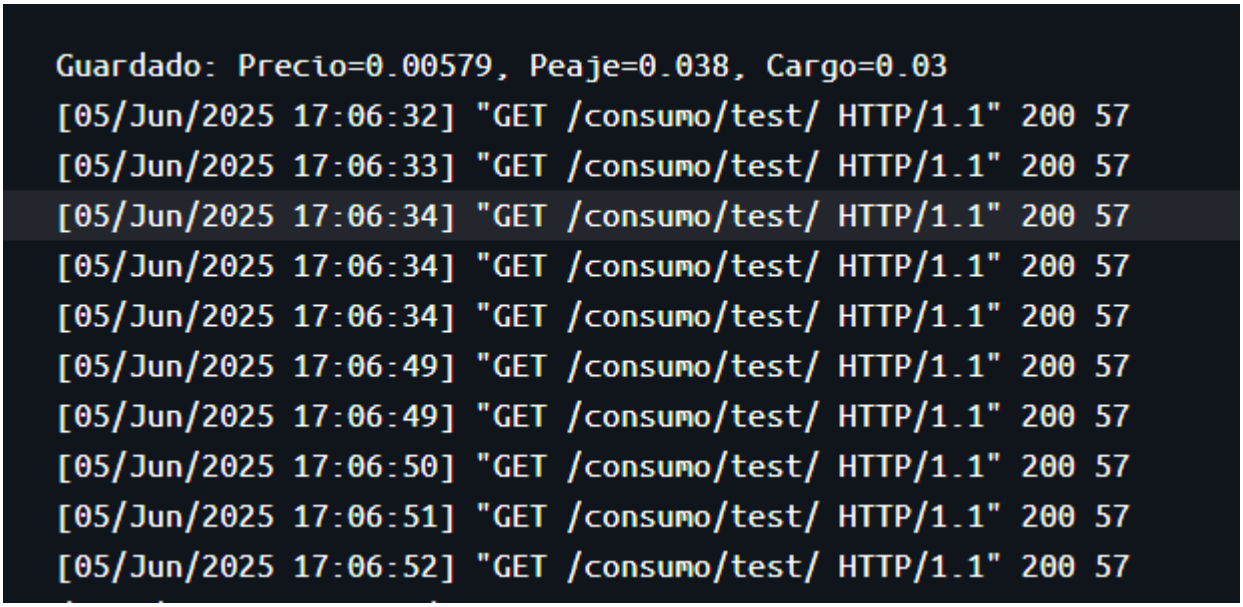
docker-compose build
docker-compose up

con este último comando, añadimos las tablas a la base de datos de postgres automáticamente:

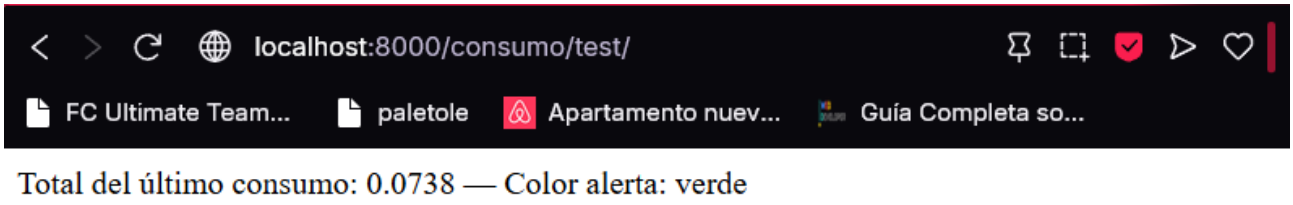
docker-compose run web py manage.py migrate
después de esto, he accedido por web para comprobar que todo funciona:

http://localhost:8000

cada vez que entramos o recargamos la página, se hace un GET para ver los últimos datos guardados en la base de datos.



Se mostraría tal que así:



Total del último consumo: 0.0738 — Color alerta: verde

No sé como mostrarlo con el semáforo gráfico ya que no he trabajado con esa parte, solo me he centrado en docker y poco más.