# API django_assesment

En este proyecto se implementa una API REST con arquitectura hexagonal y desarrollo basado en unittest de compoortamieto.

## Clonar repositorio

```
git clone https://github.com/nxiodev/django_arq_hex_example
```

## Bases de datos:

- **PostgreSQL** : Se utiliza para almacenar la información interna de la aplicación

NOTA: De acuerdo a lo propuesto en la HU_SPRINT.md se genera un diagrama de entidad relación en un png

## Desplegar proyecto

1. Crea tu ambiente virtual ejecuta el siguiente comando

```
py -m venv venv;
venv\Scripts\activate
```

2. Instala las dependencias del proyecto

```
 pip install -r requirements.txt
```

3. Crear una base de datos dyp-db en postgres y situate en la carpeta django_test para crear tu archivo .env

```
 DEBUG=True
 SECRET_KEY='jXn2r5u7x!A%D*G-KaPdSgVkYp3s6v9y/B?
E(H+MbQeThWmZq4t7w!z%C&F)J@NcRfUjXn2r5u8x/A?D(G-
KaPdSgVkYp3s6v9y$B&E)H@MbQeThWmZq4t7w!z%C*F-J'
 ALLOWED_HOSTS=*

CORS_ORIGIN_WHITELIST=http://localhost:3000,http://localhost:8000,http://loc
alhost:8080

 # Database
 DB_USER=postgres
 DB_PASSWORD=1234
 DB_HOST=127.0.0.1
```

```
#DB_HOST=db
DB_PORT=5432
DB_NAME=dyp-db
#DB_NAME=dyp-docker-db
DB_ENGINE=django.contrib.gis.db.backends.postgis


OSGEO4W_ROOT=C:\OSGeo4W
```

NOTA: Usa tu usuario y contraseña de postgres y el host y dbname de acuerdo a si usas docker o tu local

4. En la misma carpeta ejecuta las migraciones

```
python manage.py makemigrations;
python manage.py migrate
```

5. Ahora instala los fixtures para el dummy data

```
python .\manage.py loaddata .\fixtures\data.json
```

6. Inicia el servidor

```
python .\manage.py runserver
```

## Desplegar proyecto con docker

1. construye la imagen

```
docker compose build
```

2. Crea el contenedor de la base

```
docker compose up -d db
```

3. Crea el contenedor de la api

```
docker compose up
```

4. Lista los contenedores para obtener los ids

```
docker ps
```

5. Ingresa al contenedor de la base para instalar postgis y poder hacer migraciones

```
docker exec -it <id_container> bash
```

6. Instala postgis

```
apt-get update
apt-get install -y postgis
```

7. Crea la extension de postgres

```
psql -U postgres -c "CREATE EXTENSION postgis;"
```

8. Crea las migraciones en el contenedor de la api

```
docker exec -it <id_container> bash
python manage.py makemigrations;
python manage.py migrate
```

## Info del proyecto

Este proyecto fue desarrollado basado en una practica la cuál es generar contratos para el backend y front end los cuales pérmitan de manera sencilla la integración de los servicios y la comunicación entre los mismos.

Se generó un documento contratos_sprint.md el cual contiene los contratos de los servicios que se implementaron en el proyecto. También se generó un diagrama entidad relación en el archivo diagrama ER.png El proyecto cuenta con dos usuarios cargados en los fixtures. Los usuarios tienen dos perfiles, super_administrator y administrator.

- **super_administrator** : Puede crear, editar y eliminar clientes y pagos de clientes
- **administrator** : Solo puede ver los clientes y pagos de clientes

Las credenciales de cada uno son las siguientes:

- **super_administrator** :
  - email: dj-superadmin@forte.io
  - password: admin123
- **administrator** :

- email: dj-admin@forte.io
- password: admin123

La api está documentada en el url:

http://localhost:8000/api/swagger/

usa el servicio de token/ para obtener el token de autenticación y usa el token en el Authorize con la palabra Bearer antes del token.

EJEMPLO: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNjY5ODQ4MDQwLCJpYXQiOjE2Njk4NDc3NDAsImp0aSI6IjQwYTQ0MGRkYTk1MjRjYTliYmRlNTUwODMzZjYwMGQwIiwidXNlcl9pZCI6MX0.11mvl3StKfxX0hwKVHXzgSdlI0TUN7kJEEVPeaeLdYg

### Ejecutar pruebas

Para ejecutar las pruebas unitarias ejecuta el siguiente comando

```
python .\manage.py test
```

## Pre-commit y commitzen configuration

Para configurar el pre-commit y commitzen ejecuta los siguientes comandos

1. Instalacion de choco para instalar makefile en windows - ejecutar como administrador

```
Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-
Object
System.Net.WebClient).DownloadString('https://community.chocolatey.org/insta
ll.ps1'))
```

2. Instalacion de makefile

```
choco install make
```

3. Comprobamos que exista make

```
make -v
```

4. Instalacion de pre-commit

```
pre-commit install
```

## Flujo de integracion y versionado

1. Se formaeta el codigo con la funcion de nuestro makefile "format"

```
make format
```

2. Se agregan los archivos a staged y se genera el commitzen para generar el mensaje de commit

```
git add .; git cz commit
```

3. Se hace push

```
git push
```