SALECOLD

PRUEBAS UNITARIAS: PYTEST

- Instalación de Pytest
- Configuración Pytest
- Pruebas

Instalación de Pytest

1. Activar el virtual enviroment de nuestro proyecto con el siguiente comando:

```
source env/bin/activate
```

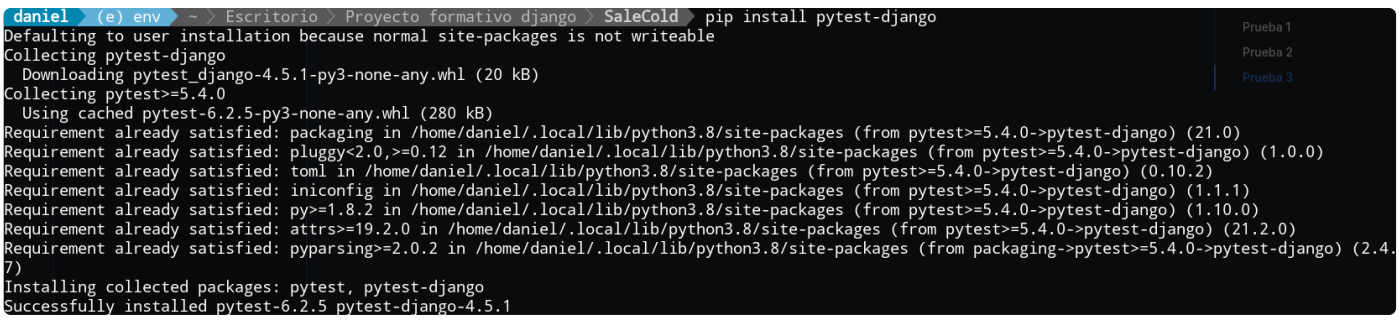
2.Instalar el gestor de paquetes de python con el siguiente comando:

> SI ya esta instalado omitir este paso.

```
sudo apt install python3-pip
```

3.Instalar la libreria pytest en nuestro proyecto con el siguiente comando:

```
pip install pytest-django
```



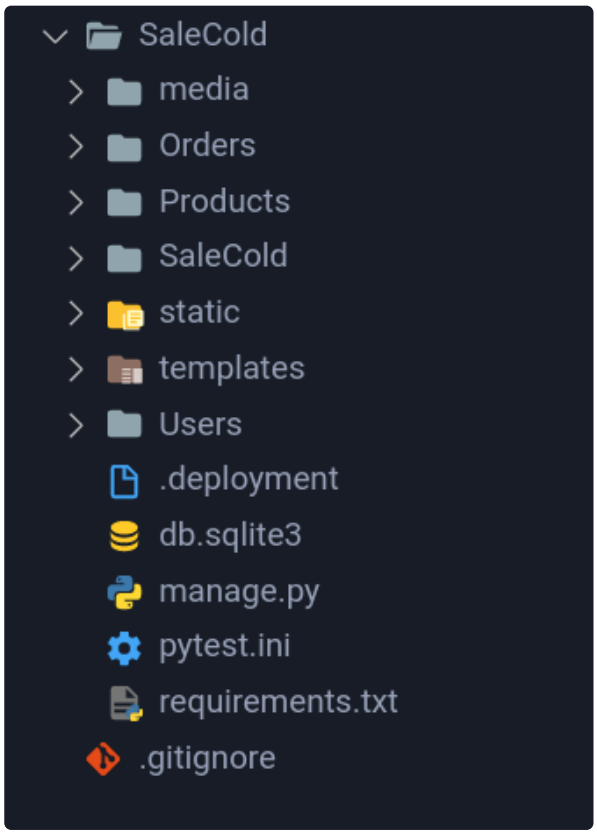
4.Ejecutar el siguiente comando para verificar que la instalación se realizo de manera correcta:

```
pip freeze
```

Configuración Pytest

1.Crear un archivo llamado pytest.ini en la carpeta principal del proyecto.

> ⓘ En este archivo se almacenara la configuración de Pytest.



2.En el archivo pytest.ini especificar la configuración con los siguientes aspectos:

- **DJANGO_*SETTINGS*_MODULE =** En esta variable constante se especificara el archivo que contiene la configuración del proyecto.
- **python_files =** En esta variable se especificara el nombre de los archivos que pytest buscara para ejecutar los test.

```
[pytest]

DJANGO_SETTINGS_MODULE = SaleCold.settings

```

```
python_files = test.py test_*.py *_test.py
```

3.Agregar el metodo create al modelo que le vamos a realizar las pruebas.

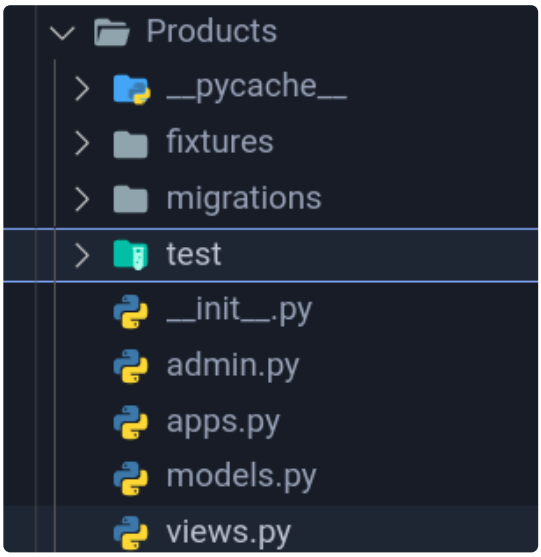
```
from django.db import models

class Category(models.Model):
    id_category = models.AutoField("Id categoria", primary_key = True)
    name = models.CharField("Nombre categoria", max_length = 55)
    description = models.CharField("Descripcion", null = True, blank = True, max_length = 
    def __str__(self):
        return self.name

    class Meta:
        verbose_name = "Categoria"
        verbose_name_plural = "Categorias"
        db_table = "category"
        ordering = ['id_category']

    @classmethod
    def create(cls, id_category, name, description):
        category = cls(id_category = id_category, name = name, description = description)
        return categorypy
```

4.Dentro de la aplicación o modulo crear una carpeta llamada test, en esta carpeta se almacenaran todos los test relacionados a la aplicación.



Pruebas

Se implementaron tres pruebas en la creación de instancias de el modelo Category, cada prueba se represento como una función.

Prueba 1

```python
import pytest
from Products.models import Category

@pytest.mark.django_db
def test_create_category():
    category = Category.create(
        id_category = 2,
        name = "categoria 2",
        description = ""
    )
    assert category.id_category == 2
```

Prueba 2

```python
import pytest
from Products.models import Category

@pytest.mark.django_db
def test_create_category2():
    category = Category.create(
        id_category = 3,
        name = "categoria 3",
        description = ""
    )
    assert category.id_category == 3
```

Prueba 3

```python
import pytest
from Products.models import Category

@pytest.mark.django_db
def test_create_category3():
    category = Category.create(
        id_category = 4,

        name = "categoria 4",
        description = ""
    )
    assert category.id_category == 4
```

Resultado

Para ejecutar las pruebas lo haremos con el siguiente comando:

> Para conocer mas información sobre que pruebas se ejecutaron, después de pytest colocamos el verbose -vv.

```
pytest -vv
```