

**Nombre del alumno:** Rosa Mariana Gutiérrez Miranda.

**Materia cursada:** Ingeniería de software.

**Nombre del asesor de la materia**: Mtra. Maribel Vergara Morgado.

**Número de la actividad:** 6A.

**Tema de la actividad:** Metodologías ágiles

Con base en los apuntes, en la consulta de las lecturas sugeridas para el bloque y en una investigación personal, elabora:

1.La descripción de un problema real que identifiques en tu entorno y que pueda resolverse con la implementación de un software.

Las empresas que se dedican a vender productos necesitan una base de datos para almacenar la información de sus productos y hacer consultas de las mismas, pero el lenguaje SQL no es amigable por lo que un software puede ayudar a interactuar con la información contenida en la base de datos de una manera amigable.

2.Realiza el proceso de desarrollo del proyecto del problema planteado, haciendo uso de la metodología XP, por los cual será necesario:

a) Llevar a cabo mínimo la primera iteración, considerando que podrás integrar a un equipo de cinco personas para el proyecto.

En este caso los miembros del equipo serán:

Mariana – Programador

Roberto – Encargado de pruebas

Alan – Encargado de Seguimiento

Luis – Entrenador

Arturo – Jefe de Proyecto

**Programador**

◦ Pieza básica en desarrollos XP

◦ Más responsabilidad que en otros modos de desarrollo

◦ Responsable sobre el código

◦ Responsable sobre el diseño (refactorización, simplicidad)

◦ Responsable sobre la integridad del sistema (pruebas)

◦ Capacidad de comunicación

◦ Acepta críticas (código colectivo)

**Encargado de Pruebas**

◦ Apoya al cliente en la preparación/realización de las pruebas funcionales

◦ Ejecuta las pruebas funcionales y publica los resultados

**Encargado de Seguimiento(Tracker)**

◦ Recoge, analiza y publica información sobre la marcha del proyecto sin afectar demasiado el proceso

◦ Supervisa el cumplimiento de la estimaciones en cada iteración

◦ Informa sobre la marcha de la iteración en curso

◦ Controla la marcha de las pruebas funcionales, de los errores reportados, de las responsabilidades aceptadas y de las prueba añadidas por los errores encontrados

**Entrenador (Coach)**

◦ Experto en XP

◦ Responsable del proceso en su conjunto

◦ Identifica las desviaciones y reclama atención sobre las mismas

◦ Guía al grupo de forma indirecta (sin dañar su seguridad ni confianza)

◦ Interviene directamente si es necesario

◦ Atajar rápidamente el problema

**Jefe del Proyecto**

◦ Favorece la relación entre usuarios y desarrolladores

◦ Confía en el equipo XP

◦ Cubre las necesidades del equipo XP

◦ Asegura que alcanza sus objetivos

b) Incluir mínimo un objeto o fragmento de programa en la parte de la codificación.

En esta ocación utlizaremos la BD PostgreSQL, obtenemos el instalador de:

<https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>

Lo ejecutamos y una vez instalado abrimos pgAdmin, inicialmente el usuario es “postgres” pero es mas seguro crear un usuario con menos privilegios para manejar nuestra base de datos:

CREATE ROLE **mariana** WITH

**LOGIN**

NOSUPERUSER

NOCREATEDB

NOCREATEROLE

INHERIT

NOREPLICATION

CONNECTION LIMIT -1

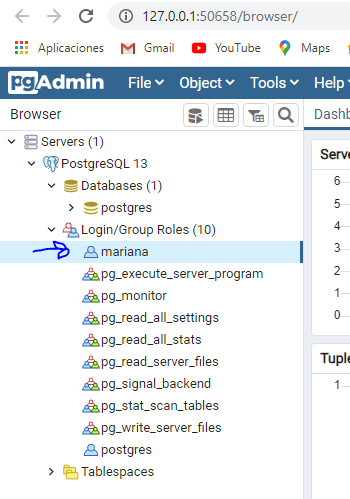
VALID UNTIL '2029-09-25T18:12:13-05:00'

PASSWORD 'xxxxxx';

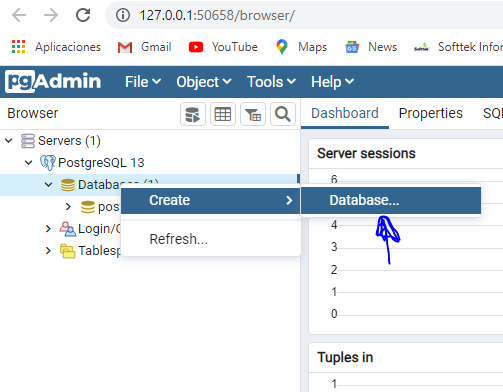
GRANT pg\_execute\_server\_program, pg\_monitor, pg\_read\_all\_settings, pg\_read\_all\_stats, pg\_read\_server\_files, pg\_signal\_backend TO mariana;

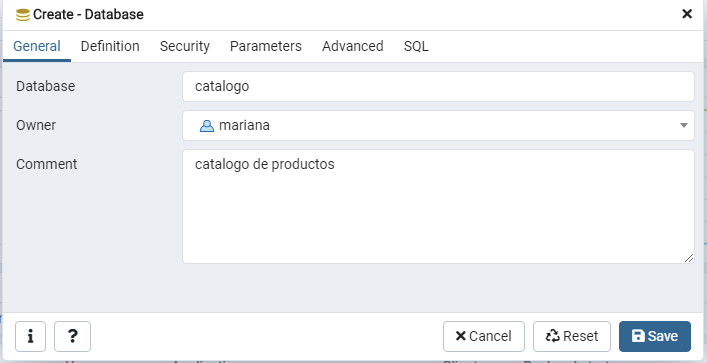
COMMENT ON ROLE **mariana** IS 'admin';

Con pgAdmin podemos visualizar nuestro login.



Luego , damos click derecho en Databases > Create > Database.





El SQL correspondiente es:

CREATE DATABASE catalogo

WITH

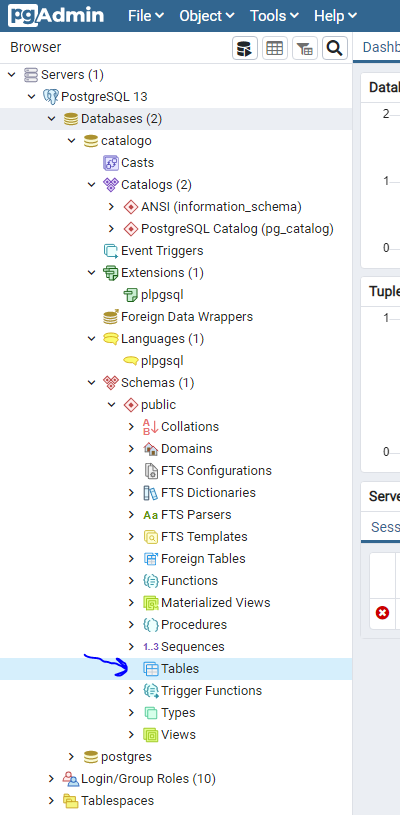
OWNER = mariana

ENCODING = 'UTF8'

CONNECTION LIMIT = -1;

COMMENT ON DATABASE catalogo

IS 'catalogo de productos';



A continuación creamos las tablas.

CREATE TABLE public.**productos**

(

id serial NOT NULL,

nombre character varying(100)[] NOT NULL,

PRIMARY KEY (id)

);

ALTER TABLE public.productos

OWNER to mariana;

CREATE TABLE public.**clientes**

(

id serial NOT NULL,

nombre character varying(100)[] NOT NULL,

PRIMARY KEY (id)

);

ALTER TABLE public.clientes

OWNER to mariana;

Desde un cmd creamos la app en un proyecto de Django:

cd programacion\_web

python manage.py startapp catalogo

Configuramos la conexión a postgres en el archivo settings.py:

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'cookies\_website.apps.CookiesWebsiteConfig',

'catalogo.apps.CatalogoConfig'

]

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.postgresql',

'NAME': 'catalogo',

'USER': 'mariana',

'PASSWORD': 'XXXXXXXXX',

'HOST': '127.0.0.1',

'PORT': '5432',

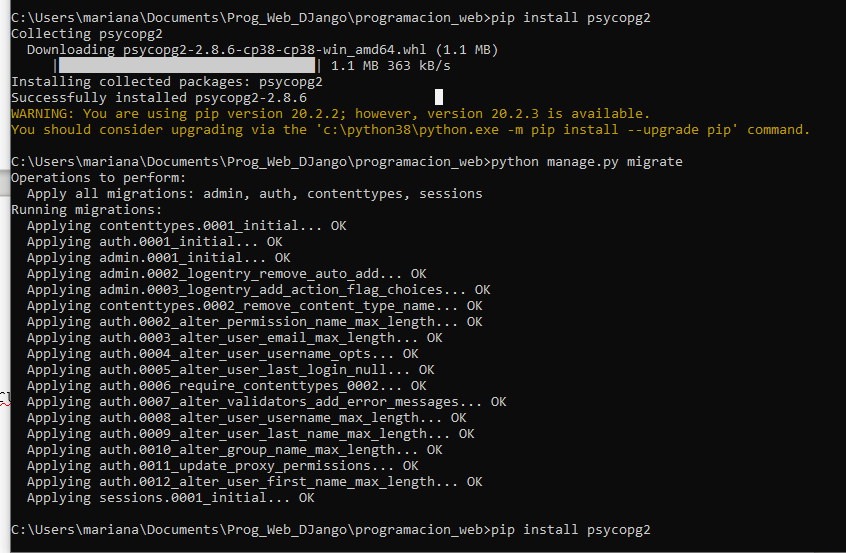
}

}

Desde un cmd ejecutamos estos comandos:

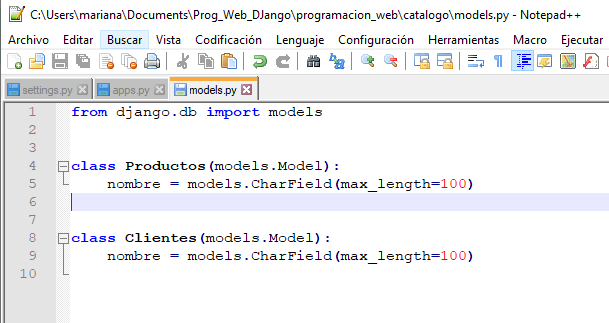
pip install psycopg2

python manage.py migrate



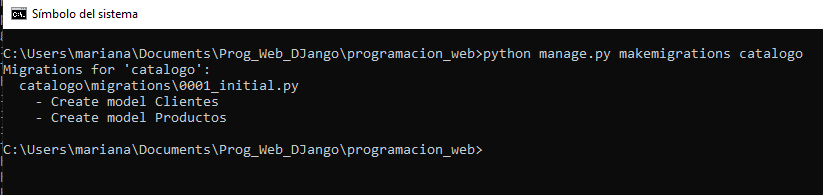
python manage.py runserver 8000

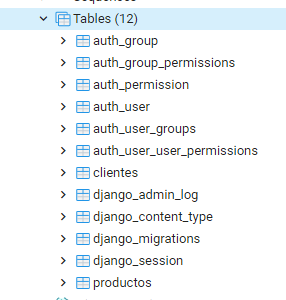
En Django los modelos representan tablas en la BD.



Ejecutamos:

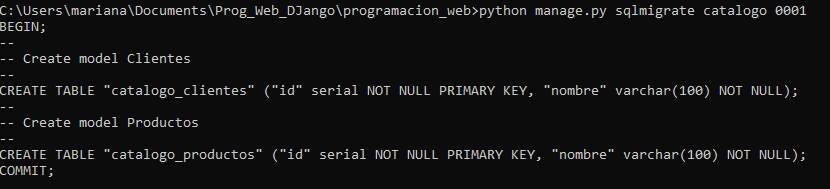
python manage.py makemigrations catalogo





Vemos como Django crea muchas tablas que necesita para funcionar aparte de la de clientes y productos.

python manage.py sqlmigrate catalogo 0001



Vemos que Django agrega el prefijo del nombre de la app a las tablas, entonces las que creamos anteriormente no servirán, pero podemos utilizar el SQL que nos arroja Django para crear estas tablas rápidamente:

BEGIN;

--

-- Create model Clientes

--

CREATE TABLE "catalogo\_clientes" ("id" serial NOT NULL PRIMARY KEY, "nombre" varchar(100) NOT NULL);

--

-- Create model Productos

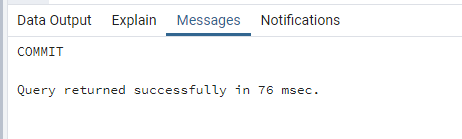
--

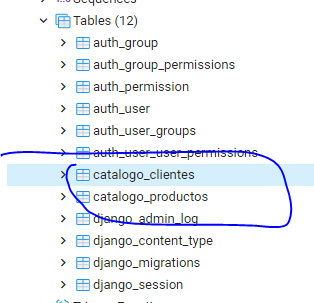
CREATE TABLE "catalogo\_productos" ("id" serial NOT NULL PRIMARY KEY, "nombre" varchar(100) NOT NULL);

COMMIT;

ALTER TABLE public.catalogo\_clientes OWNER to mariana;

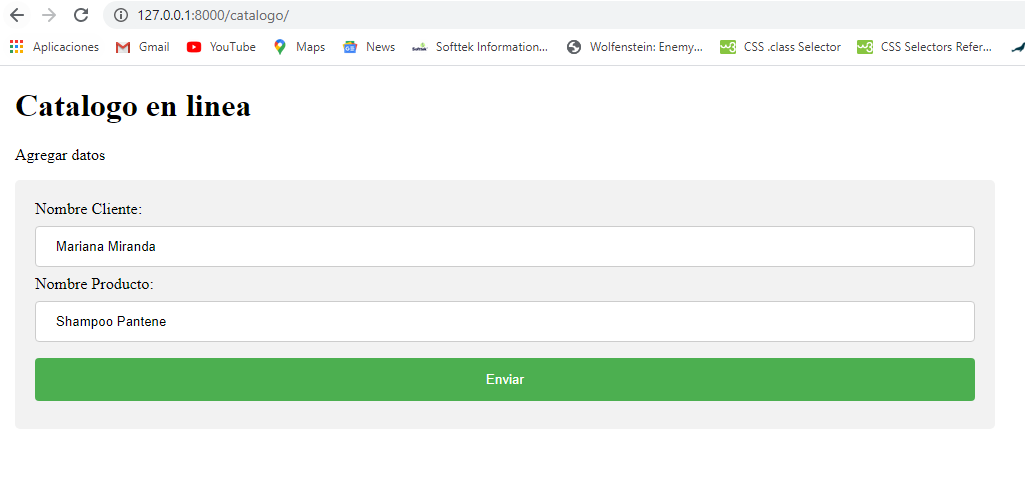
ALTER TABLE public.catalogo\_productos OWNER to mariana;



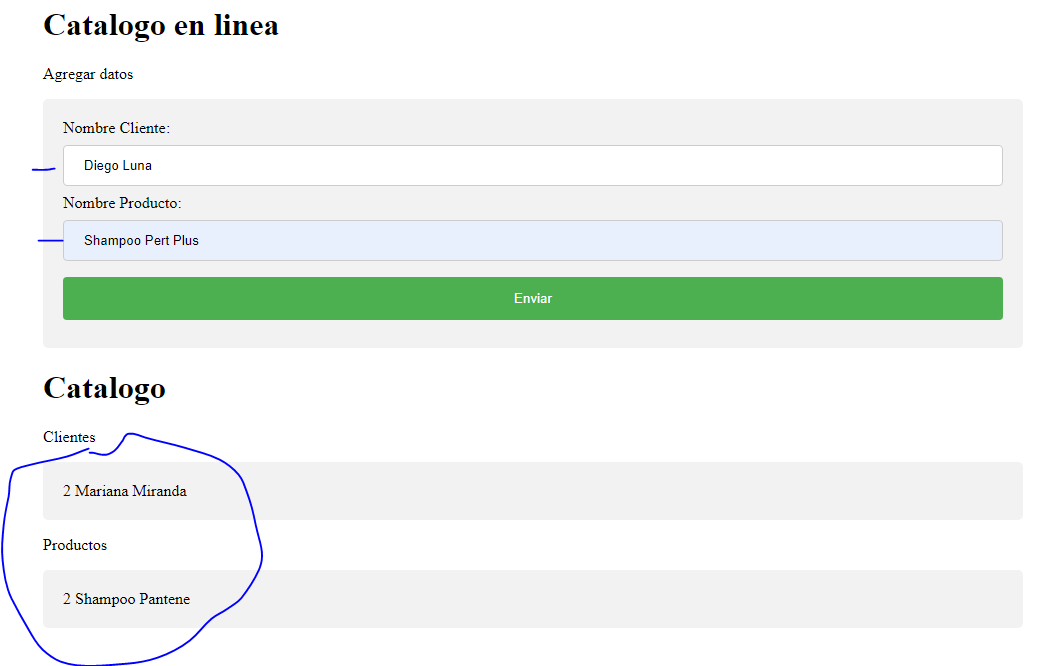


3.Redacta un documento con evidencias (imágenes, fotos, etc.) de la realización completa y correcta del proceso.

Después de codificar la aplicación presento estos screenshots como evidencia y al final adjunto screenshots del código.



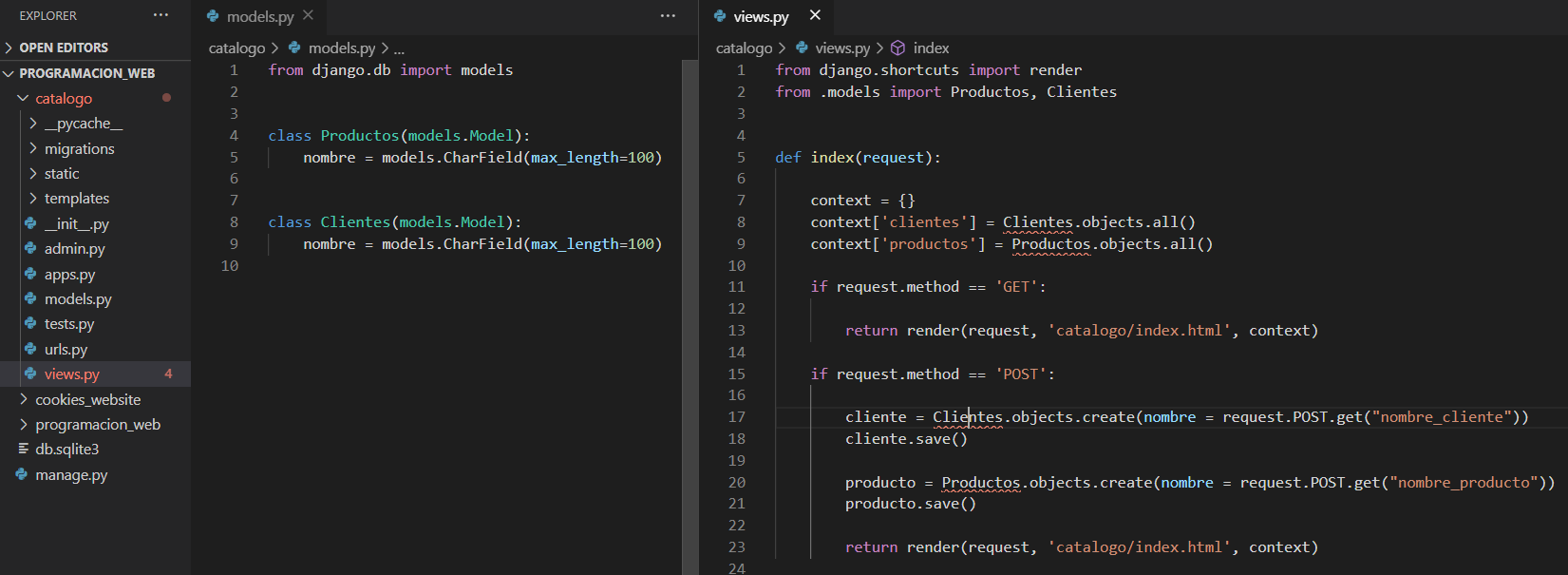
Damos click en enviar, lo que insertara esos registros en la BD y nos los muestra en la parte de abajo.

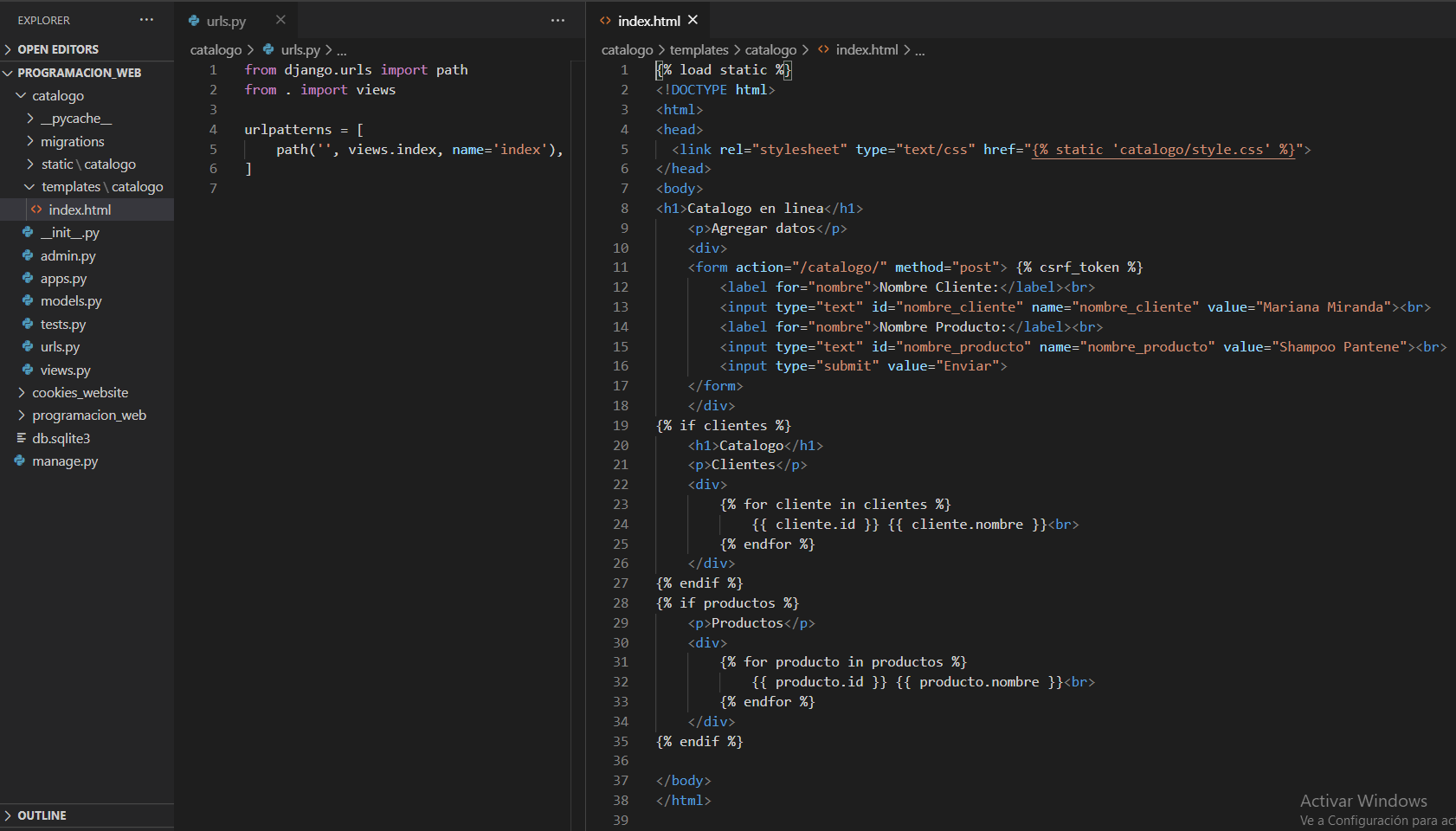


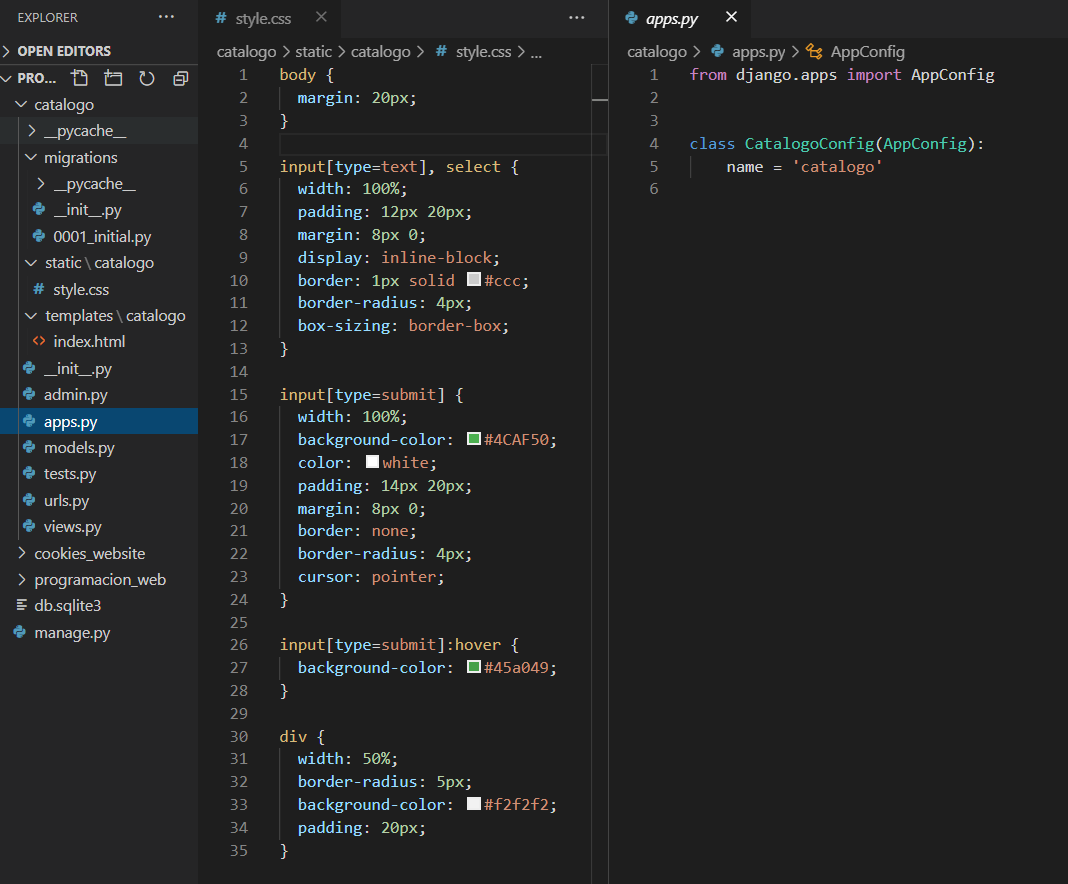
Agregamos mas datos y damos click en enviar.

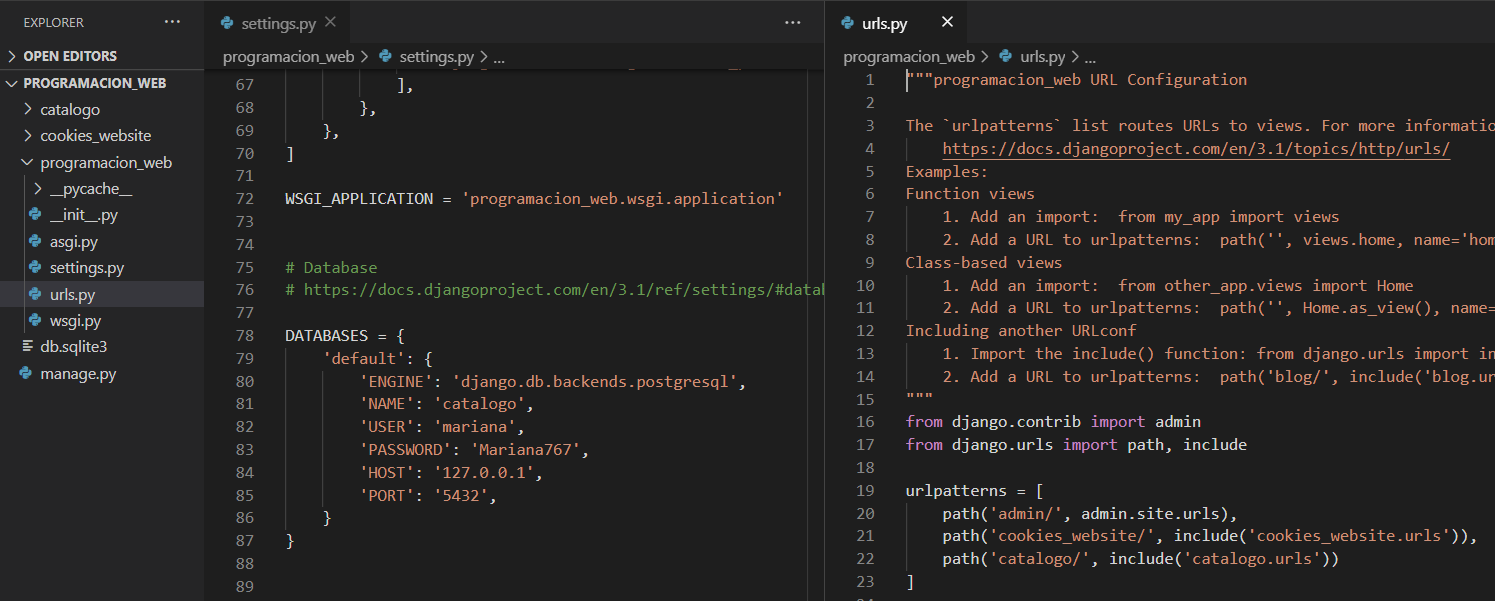


Código:









Referencias:

<https://docs.djangoproject.com/en/3.1/>