

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

**Materia cursada:** Programación Web 2.

**Nombre del asesor de la materia**: Mtra. Míriam Patricia Maldonado Benítez.

**Número de la actividad:** 3A.

**Tema de la actividad:** Creación de un sitio web con base de datos

1.Crea un sitio web con acceso a base de datos, utiliza los lenguajes de tu preferencia, (se sugiere PHP, MySQL y HTML), para tal fin desarrolla los siguientes pasos:

a) Crea la estructura de un sitio web que pueda mostrar un catálogo de artículos, que contenga mínimo el nombre de un artículo, el precio y una imagen descriptiva. También debe contener elementos visuales y texto acordes a lo solicitado. Los artículos pueden ser de cualquier tipo.

b) Crea una base de datos que tenga mínimo dos tablas, una de artículos y una más de clientes, los campos que incluyan dichas tablas proponlos tú libremente.

c) Crea una conexión entre el sitio web y la base de datos.

d) Crea un formulario para recabar los datos de un artículo y uno más para recabar los datos del usuario.

e) Demuestra que la información recopilada por la página web se envía y que se almacena en la base de datos.

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

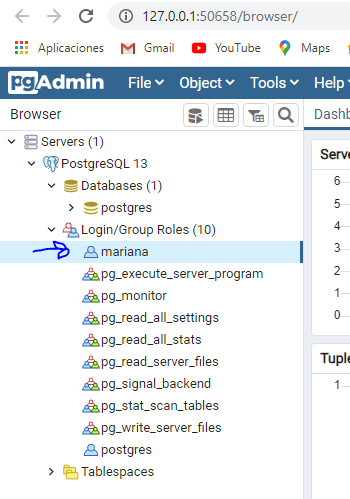
VALID UNTIL '2029-09-25T18:12:13-05:00'

PASSWORD 'xxxxxx';

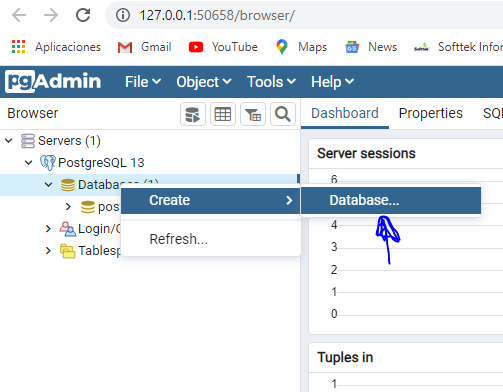
GRANT pg\_execute\_server\_program, pg\_monitor, pg\_read\_all\_settings, pg\_read\_all\_stats, pg\_read\_server\_files, pg\_signal\_backend TO mariana;

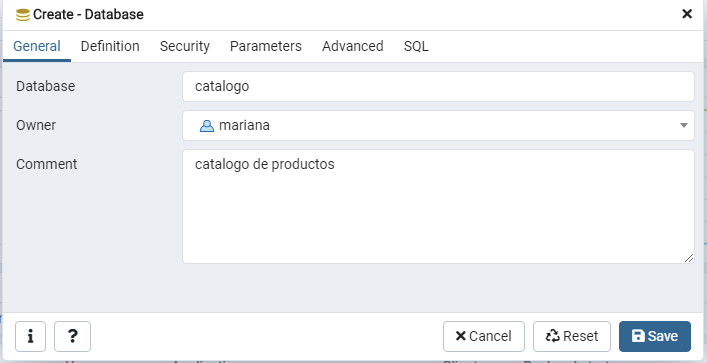
COMMENT ON ROLE **mariana** IS 'admin';

Con pgAdmin podemos visualizar nuestro login.



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





El SQL correspondiente es:

CREATE DATABASE catalogo

WITH

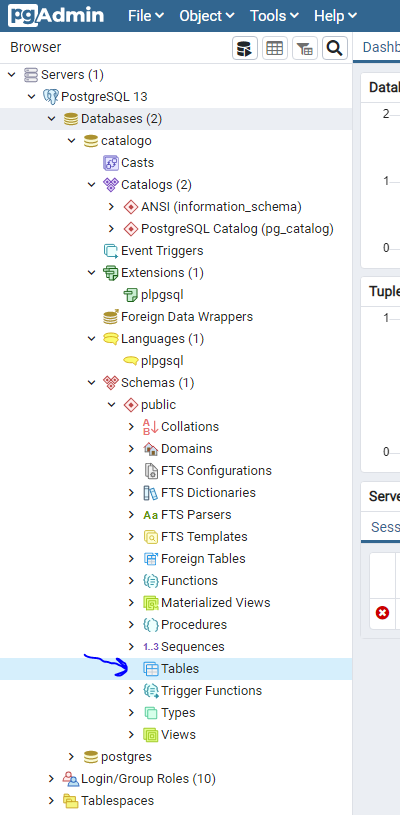
OWNER = mariana

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',

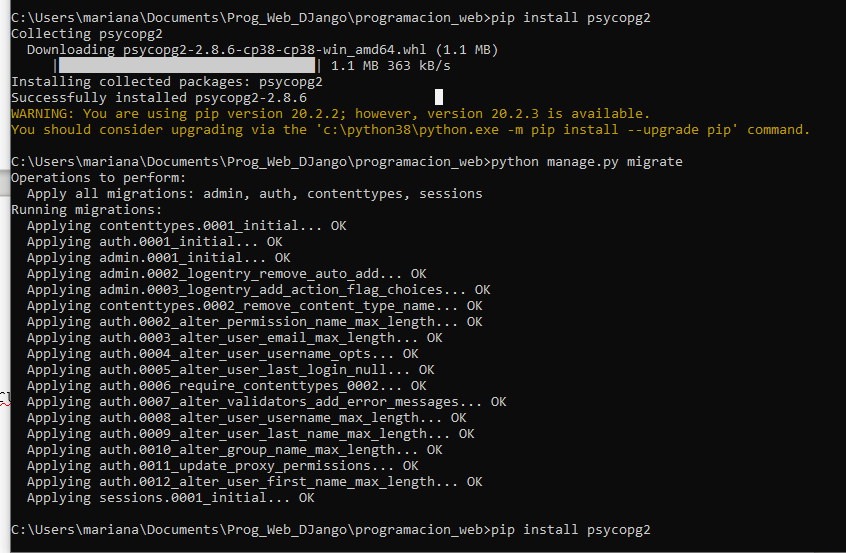
}

}

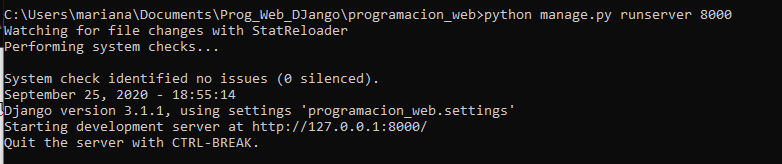
Desde un cmd ejecutamos estos comandos:

pip install psycopg2

python manage.py migrate

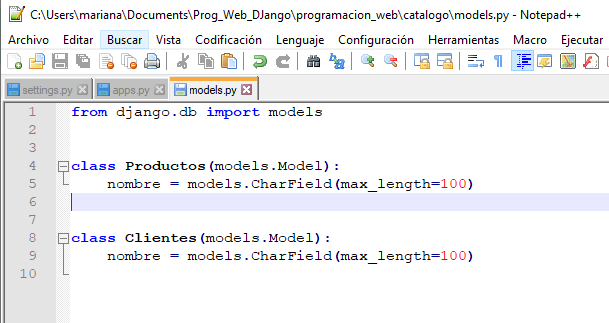


python manage.py runserver 8000



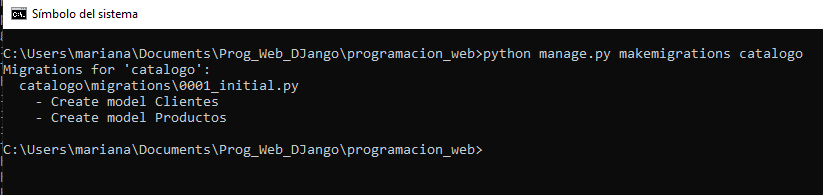
2. Redacta un reporte con pantallas del sitio creado, con evidencias (imágenes, fotos, etc.) que muestren que la estructura del sitio web es correcta y que contiene cada uno de los pasos y elementos solicitados.

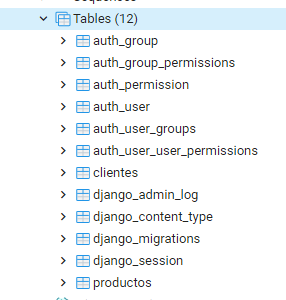
En Django los modelos representan tablas en la BD.



Ejecutamos:

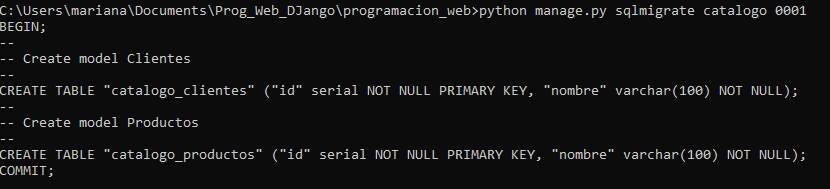
python manage.py makemigrations catalogo





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

python manage.py sqlmigrate catalogo 0001



Referencias:

<https://docs.djangoproject.com/en/3.1/intro/tutorial02/>

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

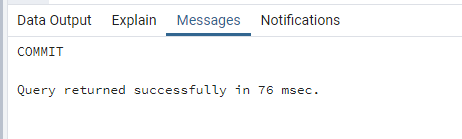
--

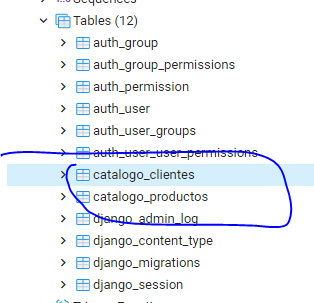
CREATE TABLE "catalogo\_productos" ("id" serial NOT NULL PRIMARY KEY, "nombre" varchar(100) NOT NULL);

COMMIT;

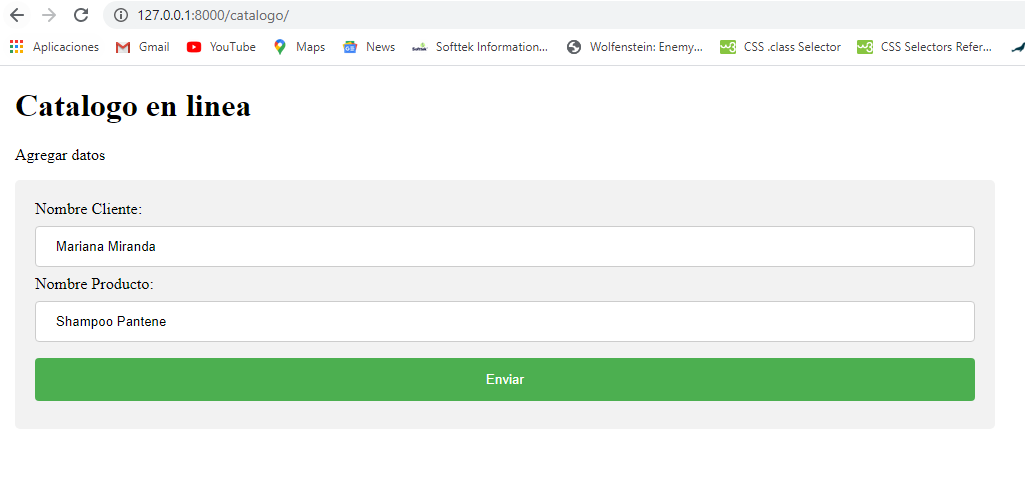
ALTER TABLE public.catalogo\_clientes OWNER to mariana;

ALTER TABLE public.catalogo\_productos OWNER to mariana;

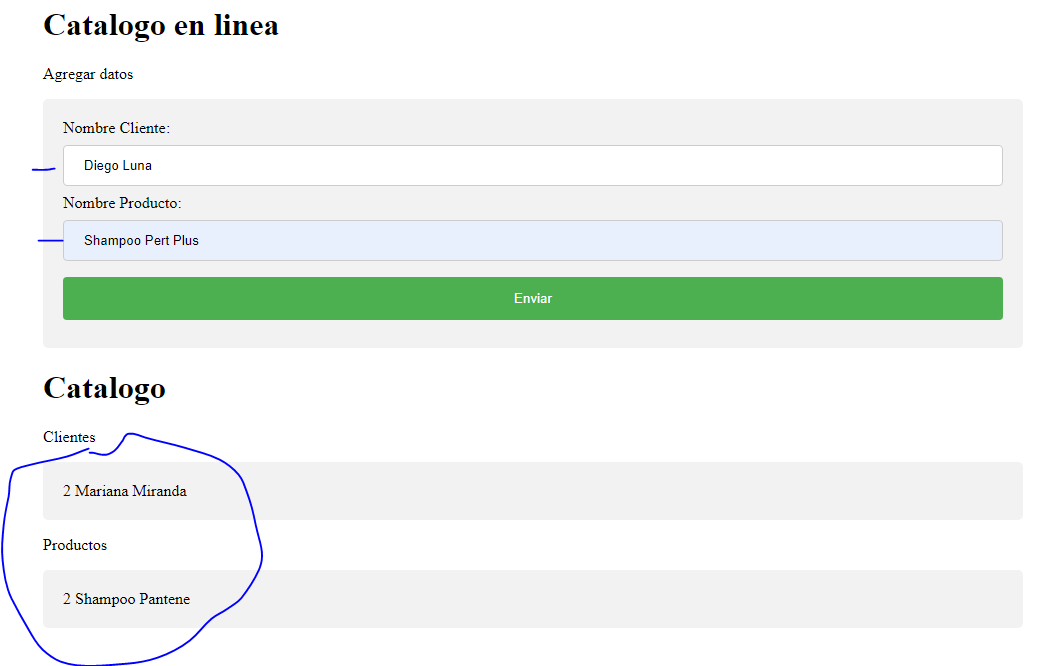




Después de codificar la aplicación presento estos screenshots como evidencia y al final adjunto screenshots del código.



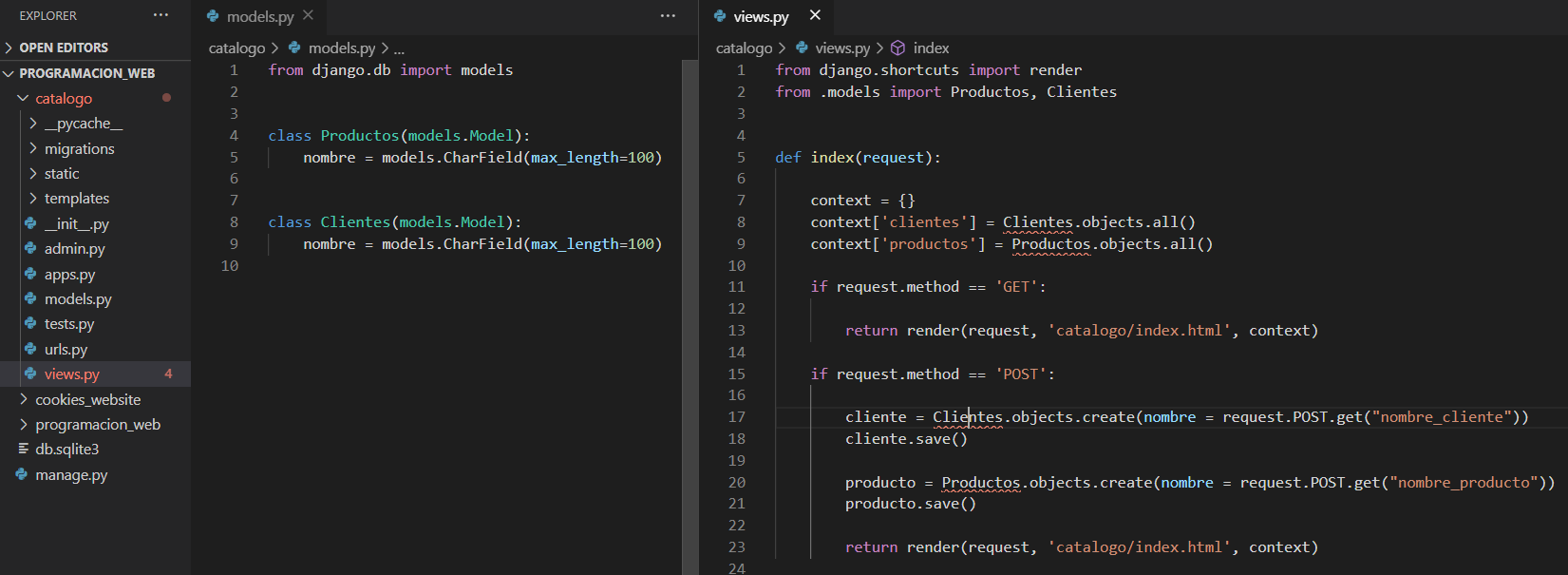
Damos click en enviar, lo que insertara esos registros en la BD y nos los muestra en la parte de abajo.

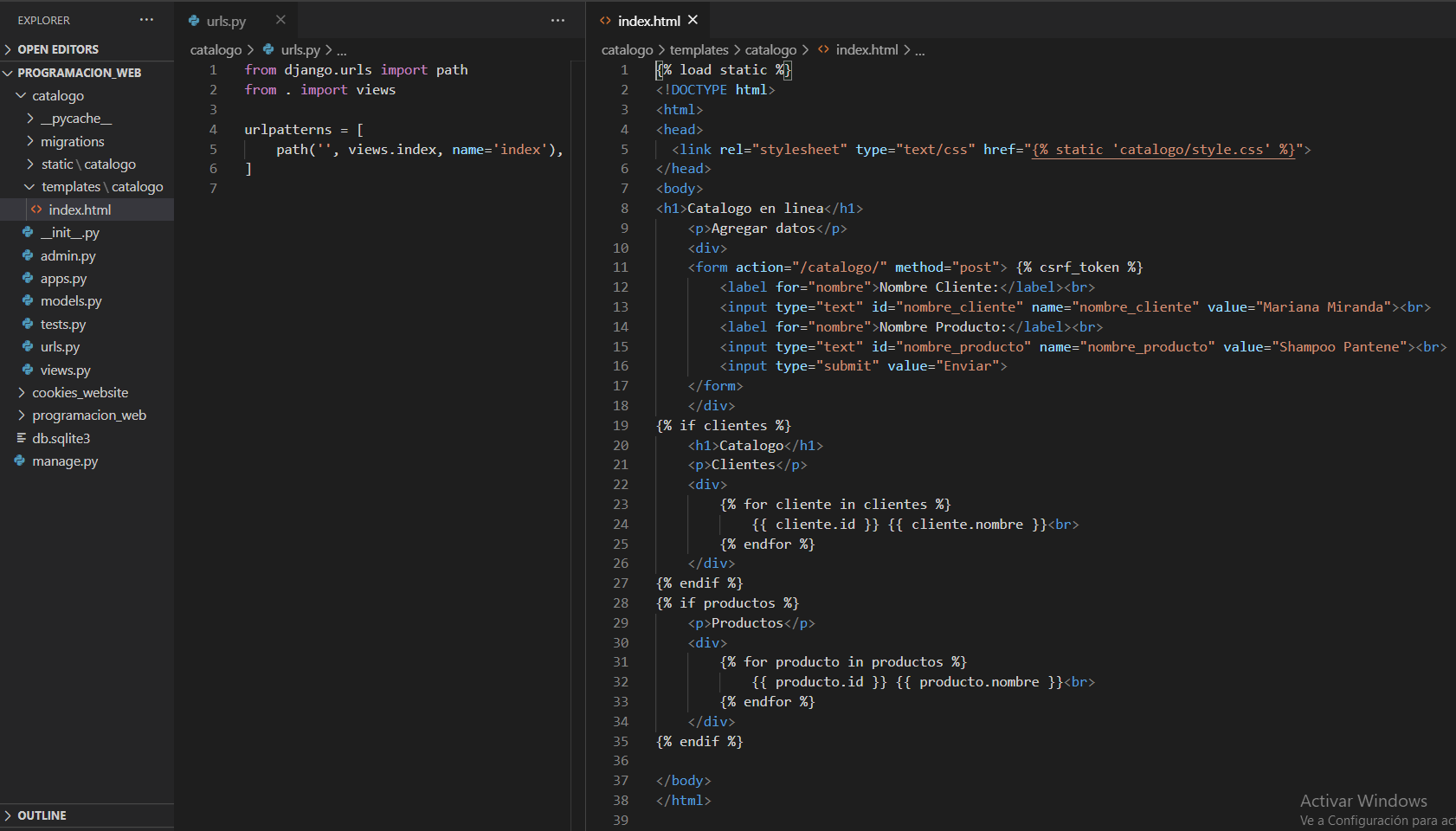


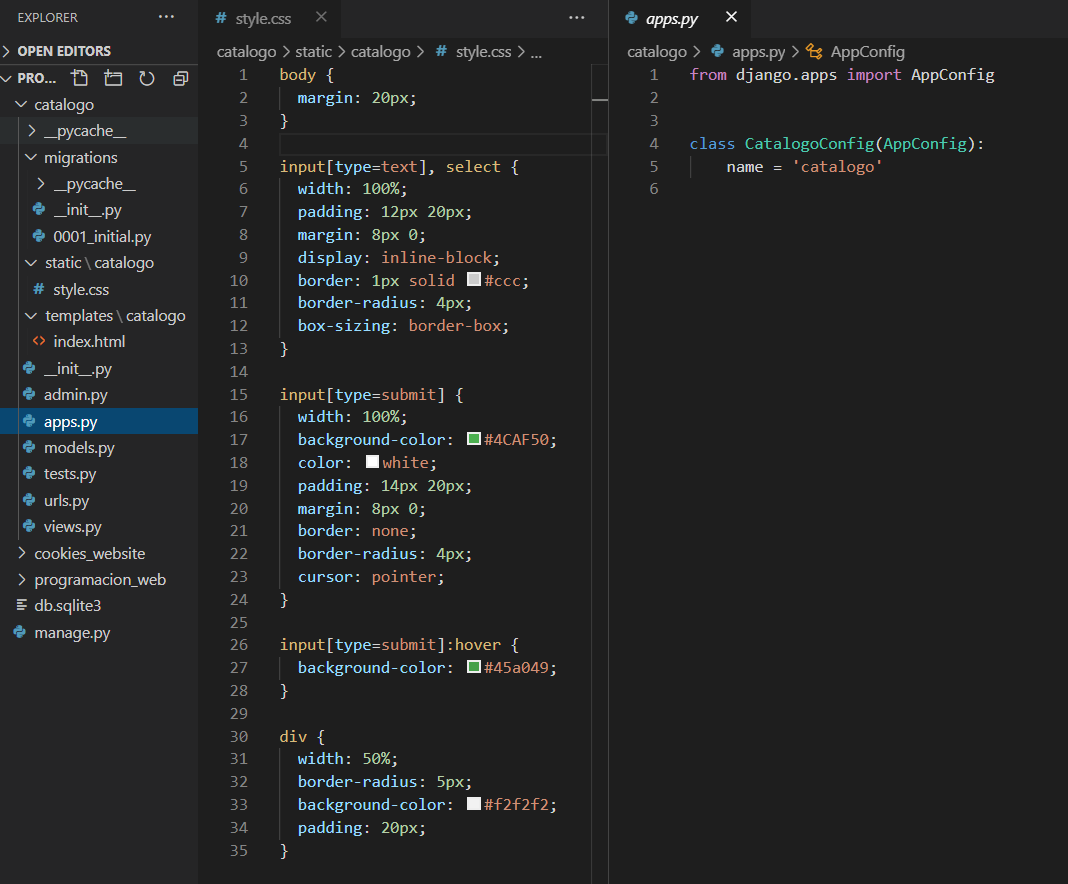
Agregamos mas datos y damos click en enviar.

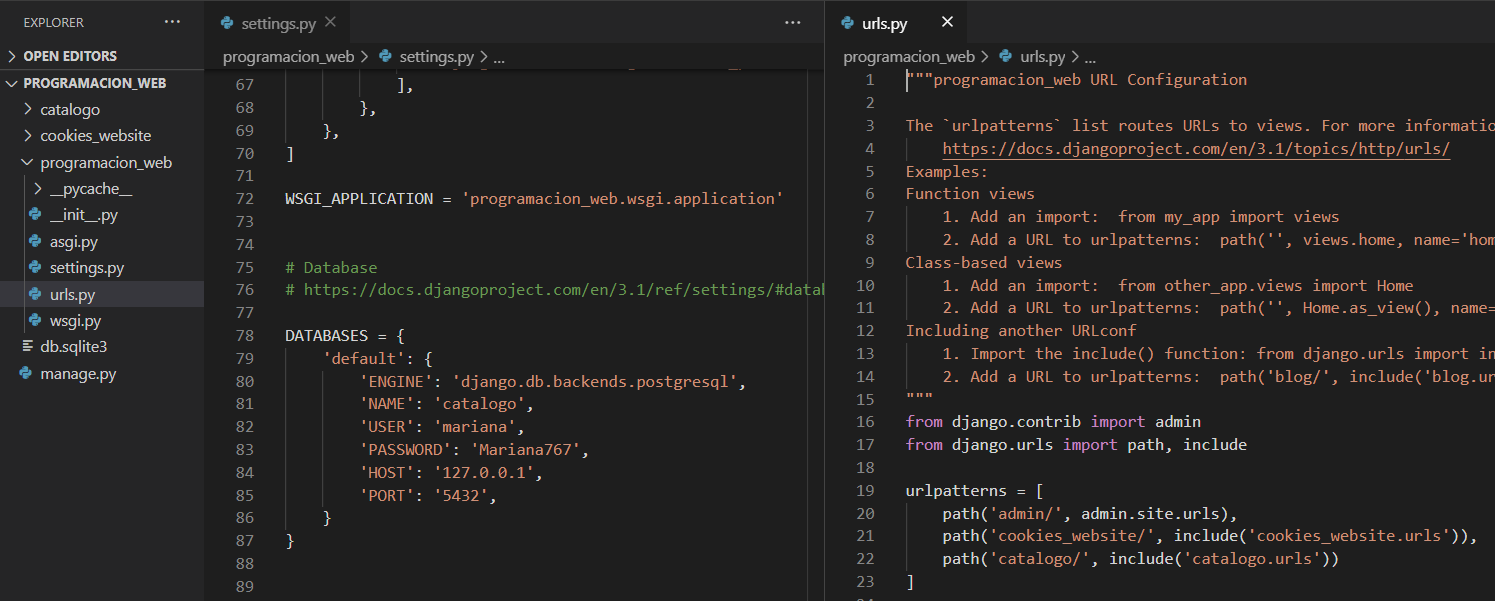


Código:









Referencias:

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