

## INSTITUTO TECNOLÓGICO SUPERIOR GUAYAQUIL

### DESARROLLO DEL SOFTWARE

NOMBRE: REY ESTEBAN GARCIA RIVAS

CURSO: 3/ E

DOCENTE: CARLOS LUIS PAZMIÑO PALMA



# INDICE

1) ¿Qué es Django? .....	pag 5
2) ¿Qué es la máquina virtual en Django? .....	pag 6
3) ¿Qué es MVT en Django. ? .....	pag 7
4) Crear un proyecto con la máquina virtual .....	pag 8
5) Descargar los instaladores de Django al proyecto.....	pag 9
6) Crear un proyecto para programar en Django.....	pag 10
7) Ejecutar el proyecto y el mensaje de felicitaciones.....	pag 11
8) Crear una Apps core.....	pag 12
9) ¿Qué es la Carpeta Templates?.....	pag 13
10) ¿Qué es la Carpeta stactic?.....	pag 14

# INDICE

- 11) Crear un archivo base html en la APPS core.....pag 15
- 12) Como se llaman a los CSS desde el archivo base html.....pag 16
- 13) Como consume un archivo hijo html al utilizar la herencia del  
archivo base html.....pag 17
- 14) Crear un view que llame al html hijo.....pag 18
- 15) Crear la urls que llame al views.....pag 19
- 16) Integrar la aplicación APPS core al proyecto principal.....pag 20
- 17) Crear las tablas del sistema de usuarios para utilizar el panel de  
administración.....pag 21

# INDICE

- 18) Crear un usuario para poder ingresar al Panel de Administración.....pag 22
- 19) Que es un modelo en Django.....pag 23
- 20) Crear un modelo en Django.....pag 24
- 21) Migrar el Modelo a la base del Panel de Administración.....pag 25
- 22) Integrar el Modelo al Panel de Administración.....pag 26
- 23) Ingresar información al modelo por el Panel de Administración.....pag 27
- 24) Realizar la consulta de todo lo ingresado en el modelo desde el  
views.....pag 28
- 25) Mostrar los datos guardados en el modelo al html hijo.....pag 29

# QUE ES DJANGO

Django es un framework web diseñado para realizar aplicaciones de cualquier complejidad en unos tiempos muy razonables. Está escrito en Python y tiene una comunidad muy amplia, que está en continuo crecimiento.



## ¿Qué es la máquina virtual en Django?



Instituto Superior Tecnológico  
**GUAYAQUIL**

## DESARROLLO DE SOFTWARE

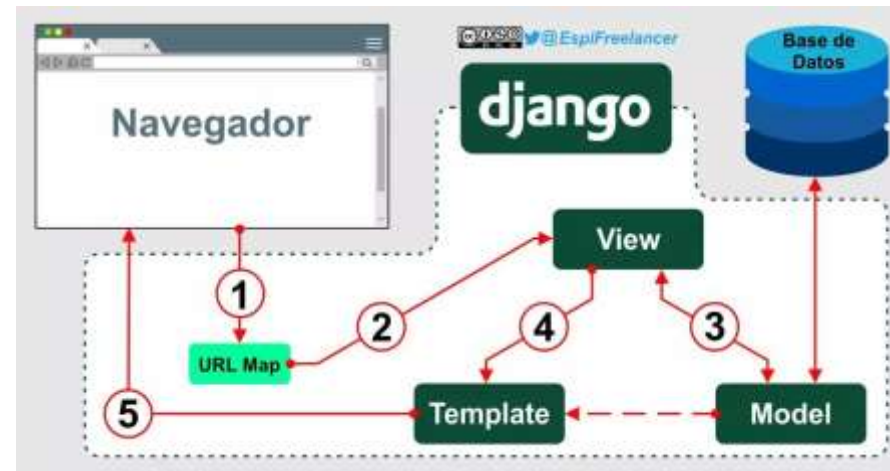
El entorno de desarrollo es una instalación de Django en tu computadora local que puedes usar para desarrollar y probar apps Django antes de desplegarlas al entorno de producción. El mismo Django proporciona un conjunto de scripts de Python para crear y trabajar con proyectos Django, junto con un simple *servidor web de desarrollo* que puedes usar para probar de forma local las aplicaciones web Django con el explorador web de tu computadora.



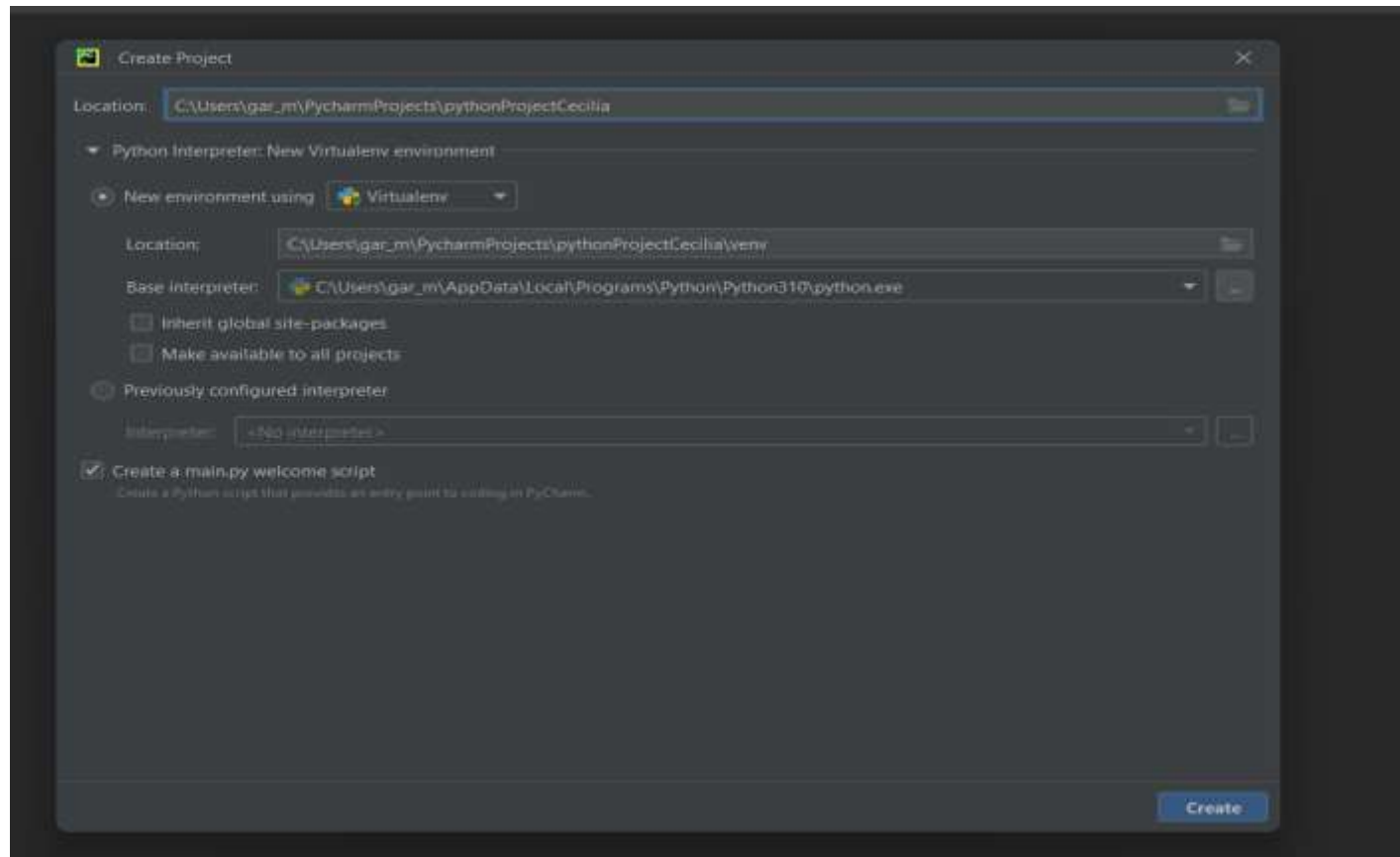
## ¿Qué es MVT en Django?



En el mundo de la programación seguro **que** habéis oído hablar del famoso patrón MVC: Modelo-Vista-Controlador. **Django** redefine este modelo como **MVT**: Modelo-Vista-Template. Hasta ahora lo **que** hemos hecho no requería de interactuar con la base de datos.



CREAR UN PROYECTO CON LA MAQUINA VIRTUAL

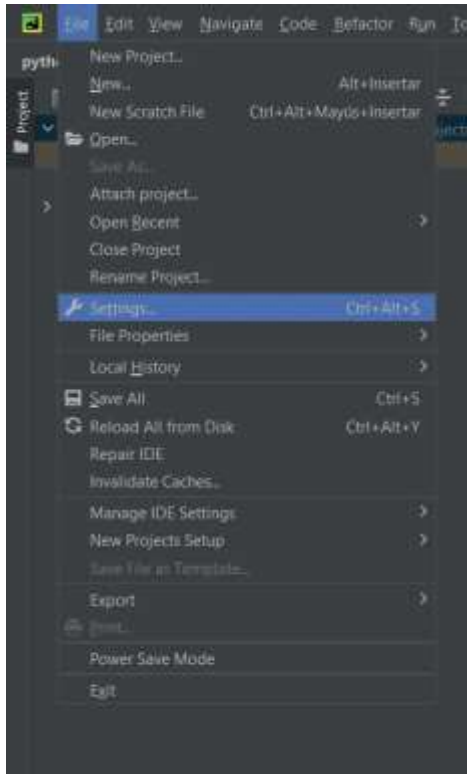


Descargar los instaladores de Django al proyecto

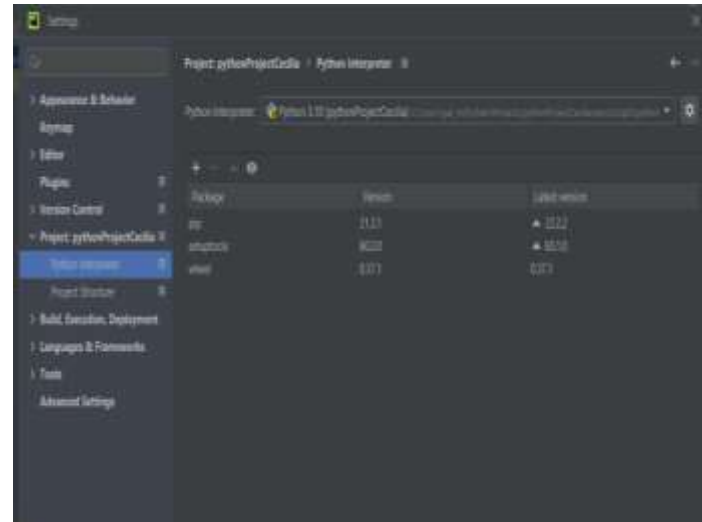




## PASO 1

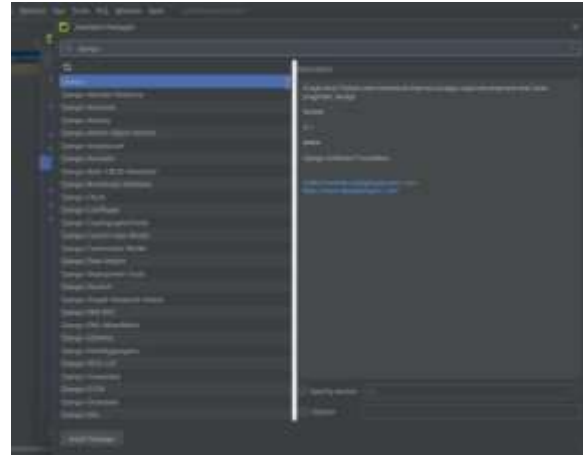


## PASO 2

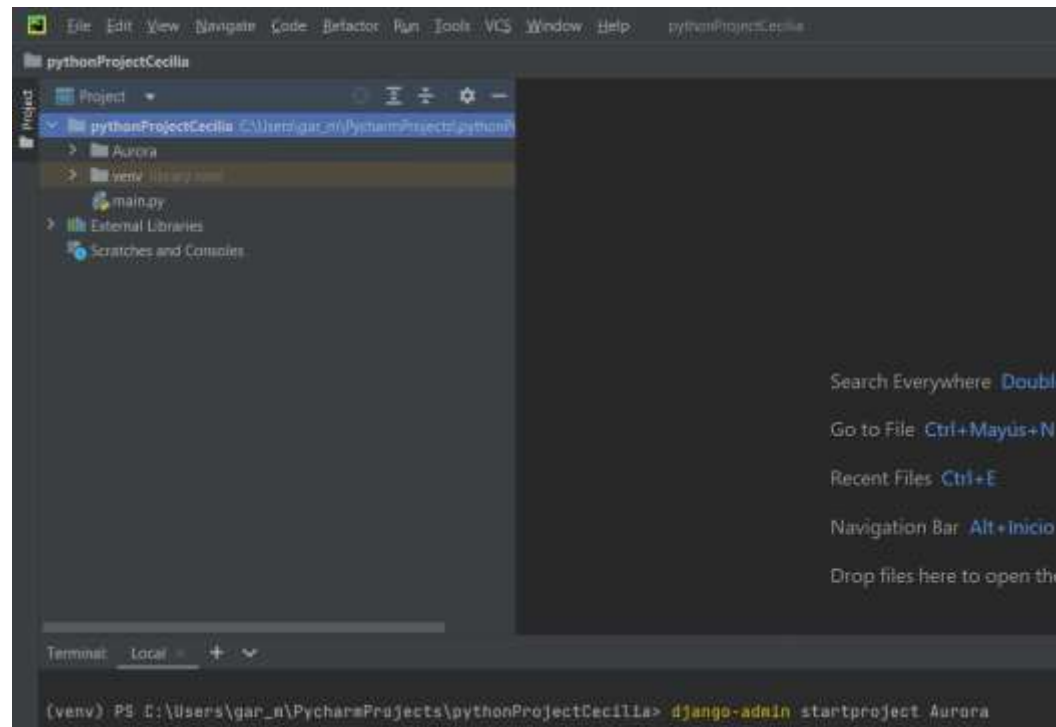




### PASO 3



Crear un proyecto para programar en Django

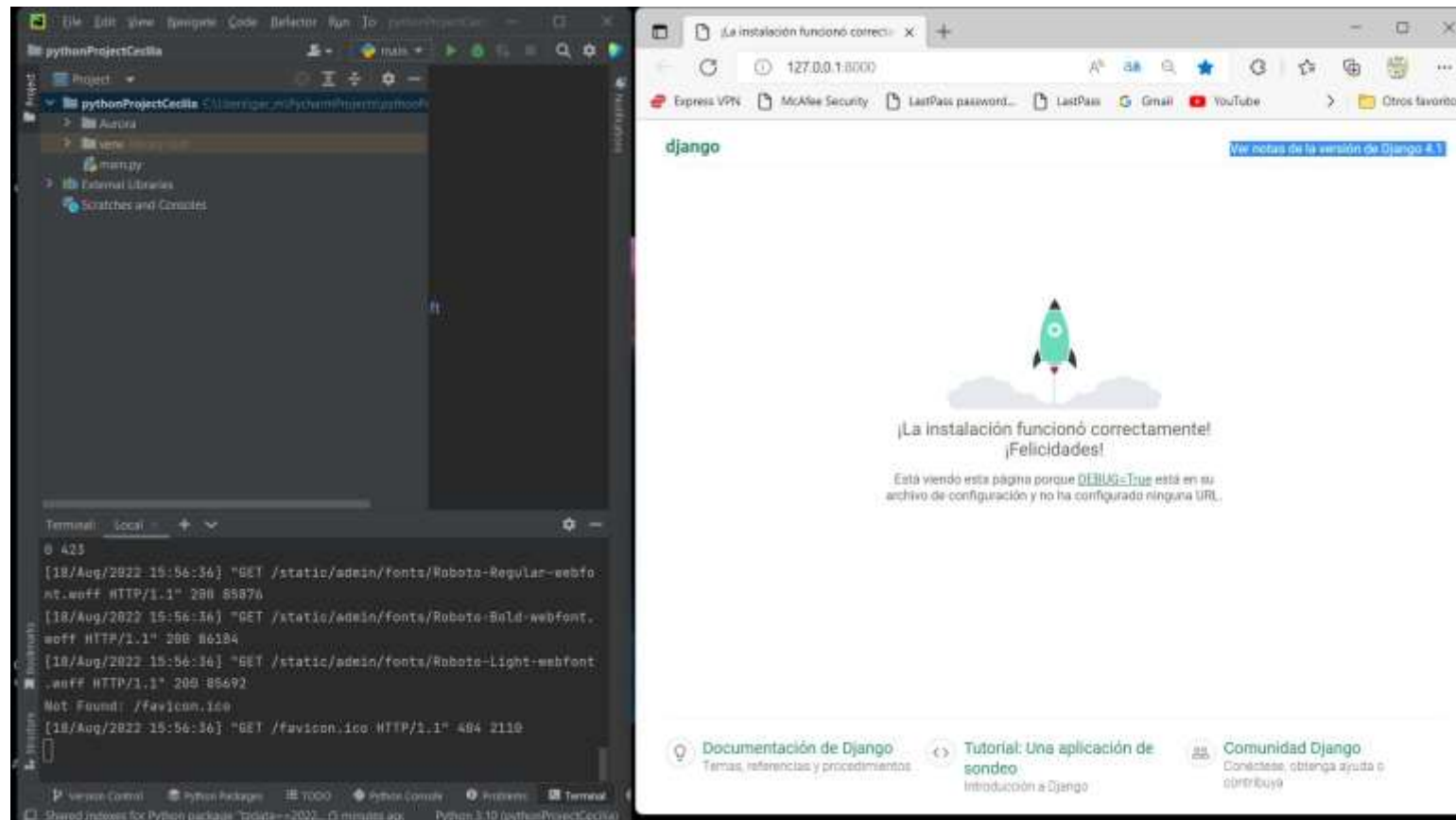


Ejecutar el proyecto y el mensaje de felicitaciones.

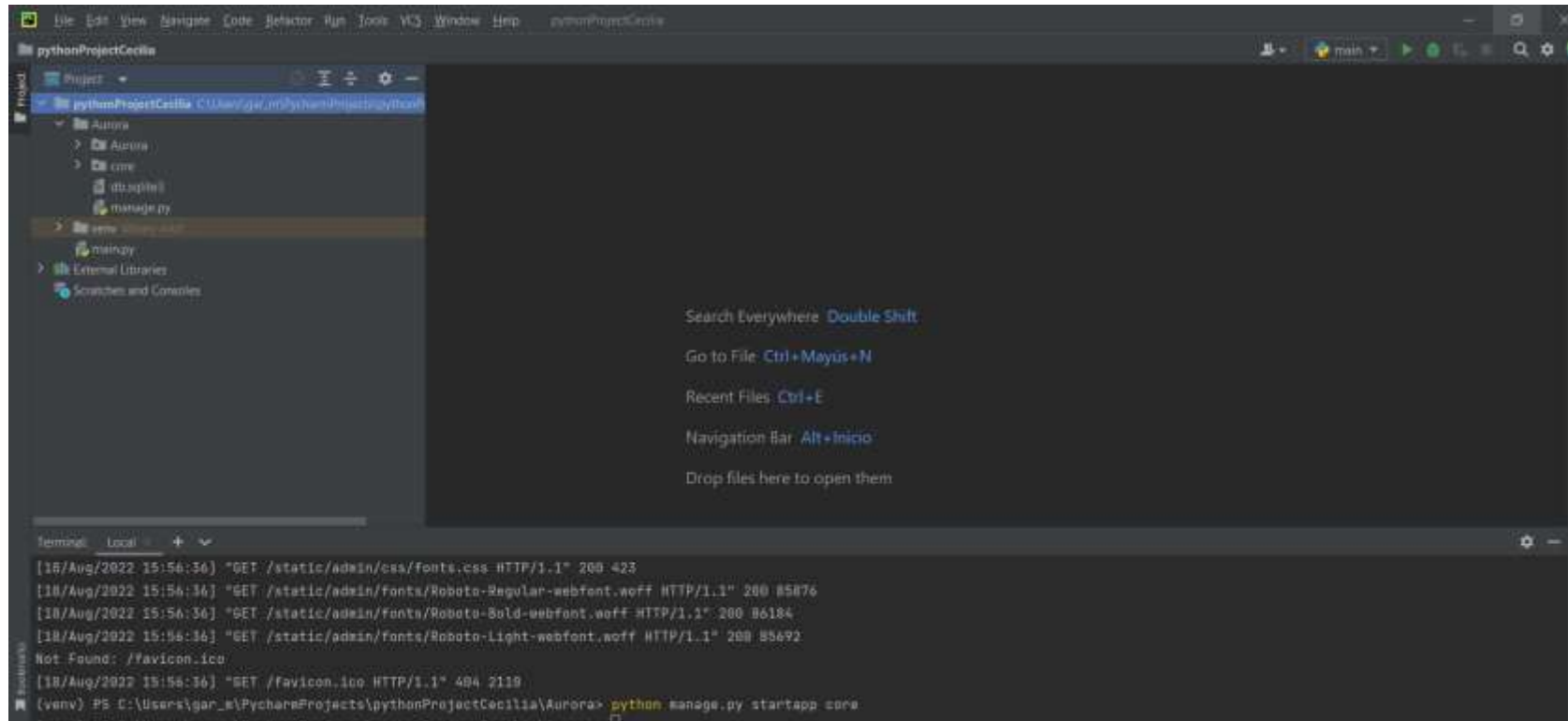


Instituto Superior Tecnológico  
**GUAYAQUIL**

## DESARROLLO DE SOFTWARE



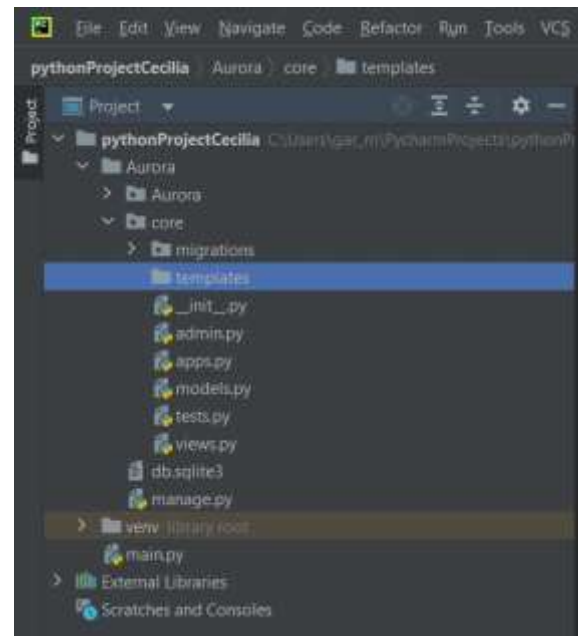
Crear una Apps core.



## ¿Qué es la Carpeta Templates?

En la carpeta Templates, **las subcarpetas contienen las plantillas que se muestran en el cuadro de diálogo Crear nuevo archivo**. Puede crear y guardar plantillas personalizadas en la carpeta

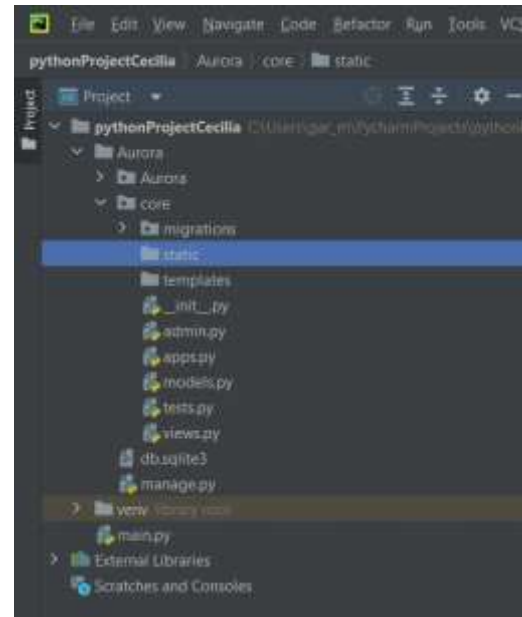
Templates. Contiene plantillas para la creación de archivos. Se puede guardar el archivo activo como una plantilla



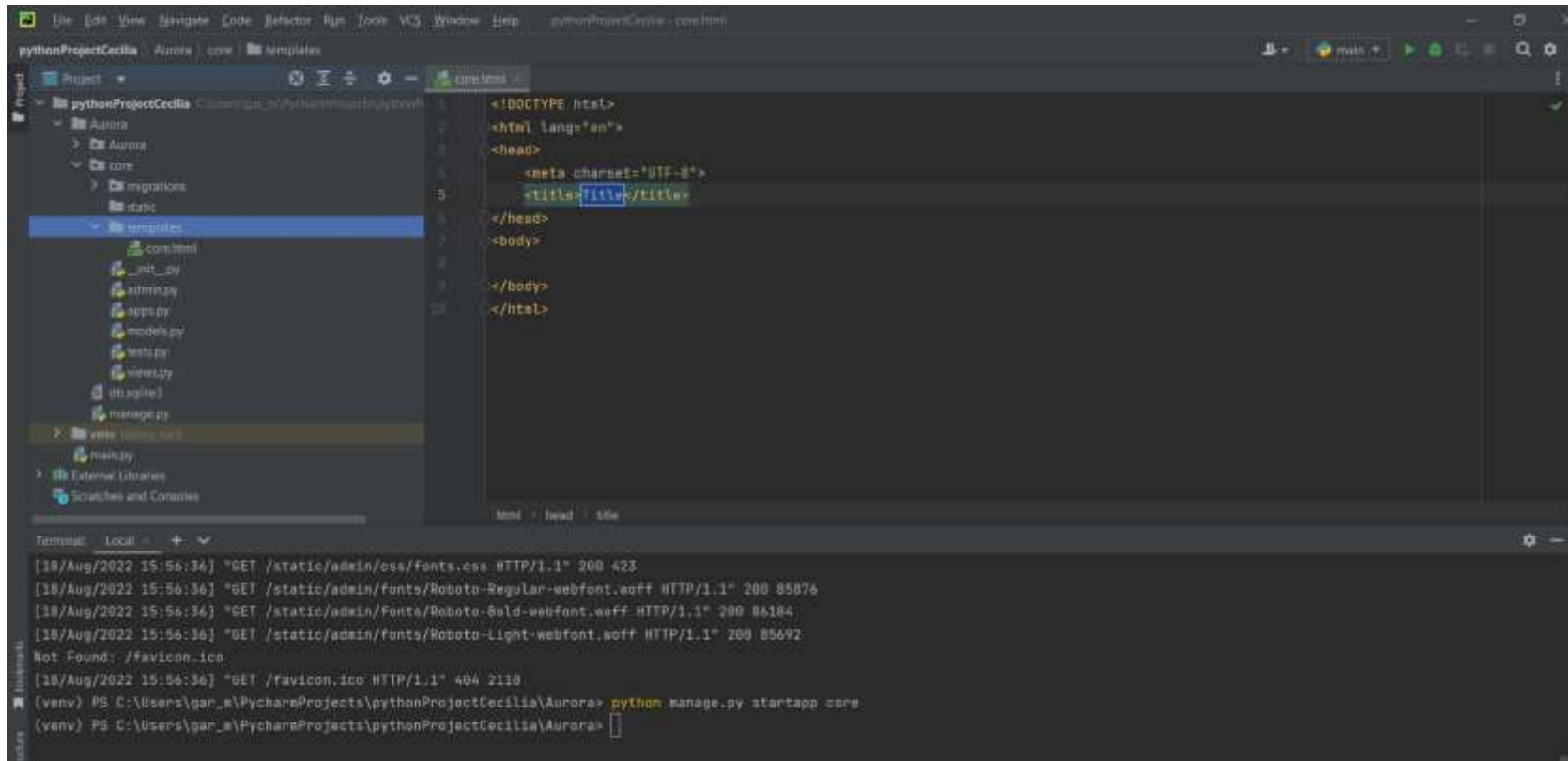
## ¿Qué es la Carpeta Static?

Esta carpeta **contiene todo los archivos de estilos: css y scss, archivos de comportamientos JavaScript, imágenes incluidas dentro del theme y el archivo checkout.**





Crear un archivo base html en la APPS core



The screenshot shows the PyCharm IDE interface. On the left, the Project Explorer displays the file structure of a Django project named 'pythonProjectCecilia'. The 'templates' directory is expanded, showing 'base.html' as the selected file. The main editor window displays the content of 'base.html', which is a basic HTML template with the following structure:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
</body>
</html>
```

At the bottom, the Terminal window shows the output of a Django development server. It displays several GET requests for static files (CSS and fonts) and a 404 error for a missing favicon. The terminal also shows the command used to start the server:

```
(venv) PS C:\Users\gar_m\PycharmProjects\pythonProjectCecilia\Aurora> python manage.py startapp core
(venv) PS C:\Users\gar_m\PycharmProjects\pythonProjectCecilia\Aurora>
```

Como se llaman a los CSS desde el archivo base html



```
{% load static %}  
<link rel="STYLESHEET" href="{%static 'core/estilos.css'%}">  
💡 <link rel="STYLESHEET" href="{%static 'core/menuUniverso.css'%}">
```

Como consume un archivo hijo html al utilizar la herencia del archivo base html.

```
contacto.html ×  
{% extends 'herencias/padre.html' %}  
{% block titulo %}  
clientes  
{% endblock %}  
{% block contenido %}
```

Crear un view que llame al html hijo



The screenshot shows a web development IDE with a project structure on the left and a Python view file in the center. The project structure includes a 'herencias' folder with several HTML files and a 'views.py' file. The 'views.py' file contains the following code:

```
##### HERENCIA#####  
def inicio(request):  
    return render(request, 'herencias/inicio.html')  
  
def quienes_somos(request):  
    return render(request, 'herencias/quienes_somos.html')  
  
def nuestro_producto(request):  
    return render(request, 'herencias/nuestro_producto.html')  
  
def contacto(request):  
    return render(request, 'herencias/contacto.html')  
##### INFORMACION#####
```

Crear la urls que llame al views.



```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.holaMundoCore, name='core'),
    path('noticias', views.noticias, name='noticias'),
    path('deportes', views.deportes, name='deportes'),
    path('inicio', views.inicio, name='inicio'),
    path('quienes_somos', views.quienes_somos, name='quienes_somos'),
    path('nuestro_producto', views.nuestro_producto, name='nuestro_producto'),
    path('contacto', views.contacto, name='contacto'),
]
```

Integrar la aplicación APPS core al proyecto principal.



```
Run Tools VCS Window Help REY-GARCIA-R [C:\Users\gar_m\Desktop\REY-GARCIA-R] - settings.py
py
settings.py
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'core',
41 ]
```

Crear las tablas del sistema de usuarios para utilizar el panel de administración.

blog_contacto			CREATE TABLE "blog_contacto" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "nombre" text NOT NULL, "telefono"
id	integer		"id" integer NOT NULL
nombre	text		"nombre" text NOT NULL
telefono	text		"telefono" text NOT NULL
correo	text		"correo" text NOT NULL
asunto	text		"asunto" text NOT NULL
mensaje	text		"mensaje" text NOT NULL
blog_portfolio			CREATE TABLE "blog_portfolio" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "titulo" text NOT NULL, "contenido"



```
0002_contacts.py
Generated by Django 4.0.5 on 2022-08-04 00:54

from django.db import migrations, models

class Migration(migrations.Migration):

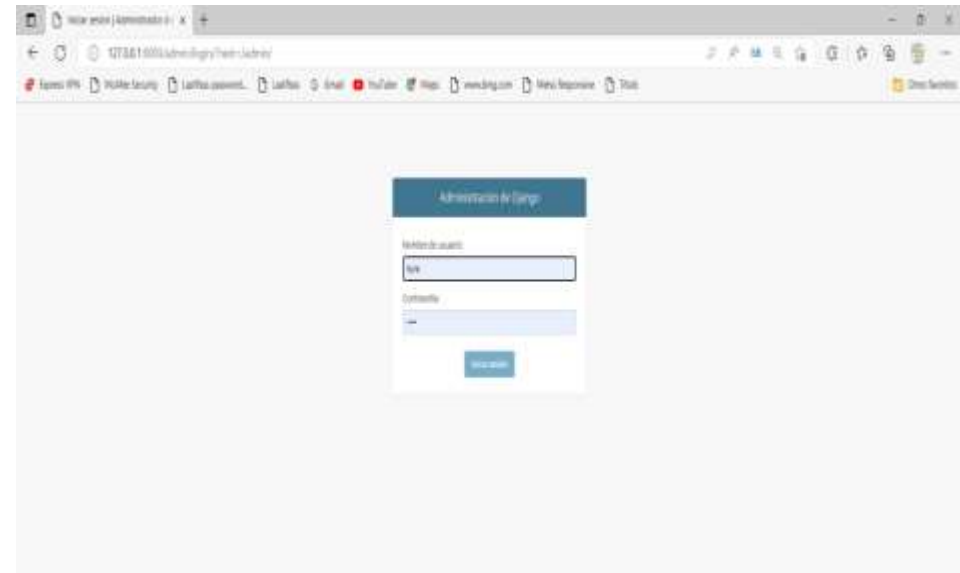
    dependencies = [
        ('log', '0001_initial'),
    ]

    operations = [
        migrations.CreateModel(
            name='Contacts',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('nombre', models.TextField()),
                ('telefono', models.TextField()),
                ('correo', models.TextField()),
                ('edad', models.TextField()),
                ('mensaje', models.TextField()),
            ],
        ),
    ]
```

Crear un usuario para poder ingresar al Panel de Administración



```
Terminal - Local +
(env) PS C:\Users\gar_m\Desktop\AURORA\KEY-GARCIA-M\GARCIA-M> python manage.py createsuperuser
Username (leave blank to use 'gar_m'): kyra
Email address: rosgarcia@est.steg.edu.ec
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

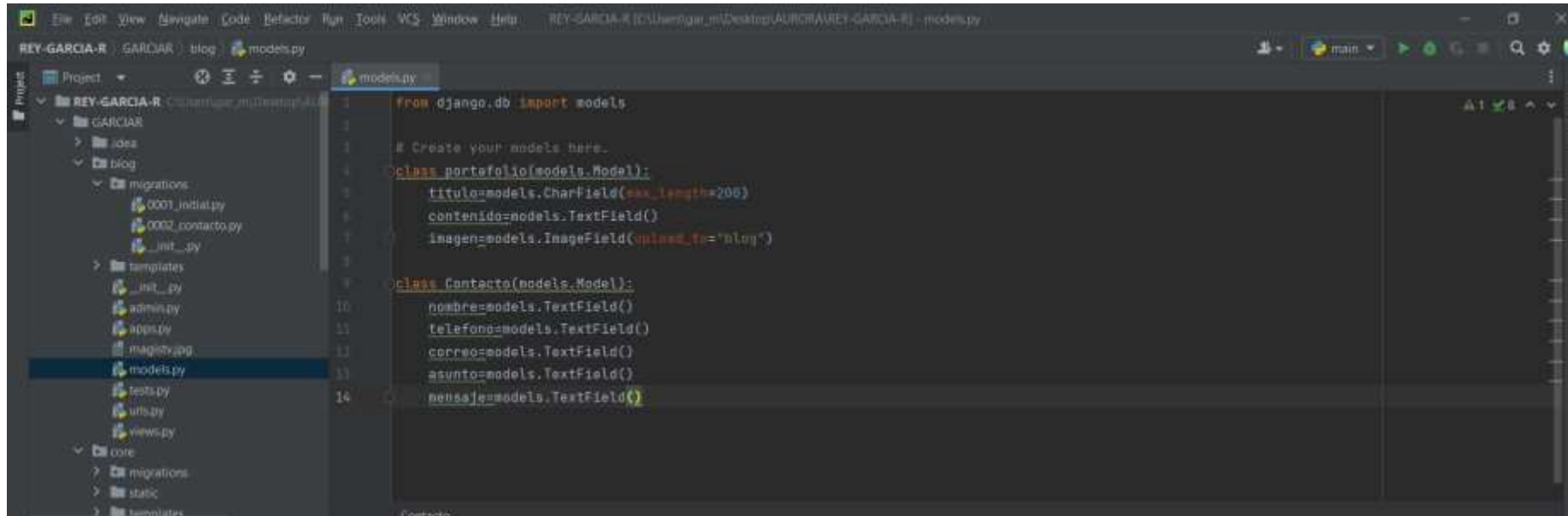


## Que es un modelo en Django

Un modelo en Django es un tipo especial de objeto que se guarda en la base de datos . Una base de datos es una colección de datos. Es un lugar en el cual

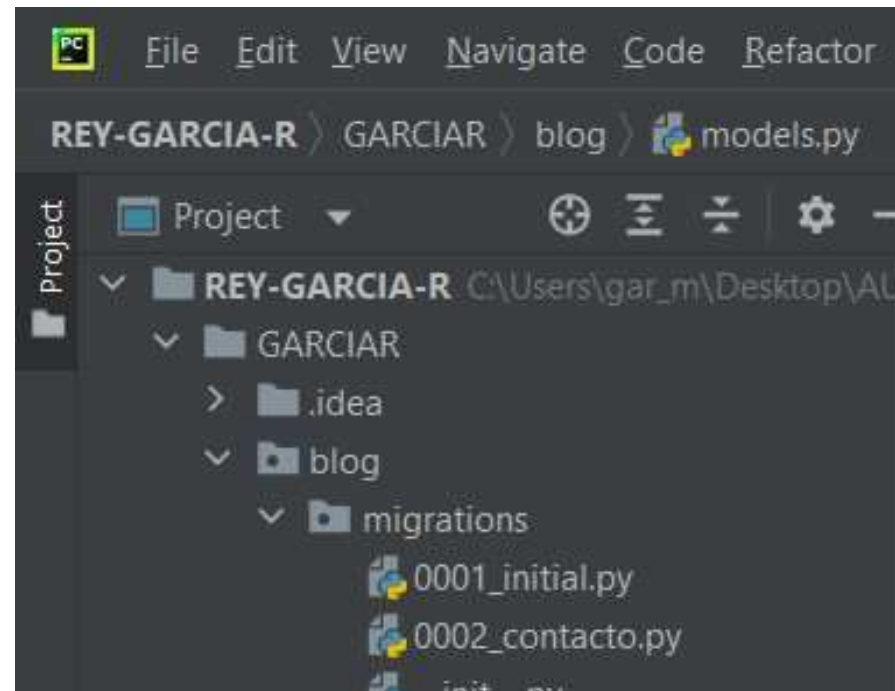
almacenarás la información sobre usuarios, tus entradas de blog, etc. Utilizaremos una base de datos SQLite para almacenar nuestros datos.

Crear un modelo en Django.

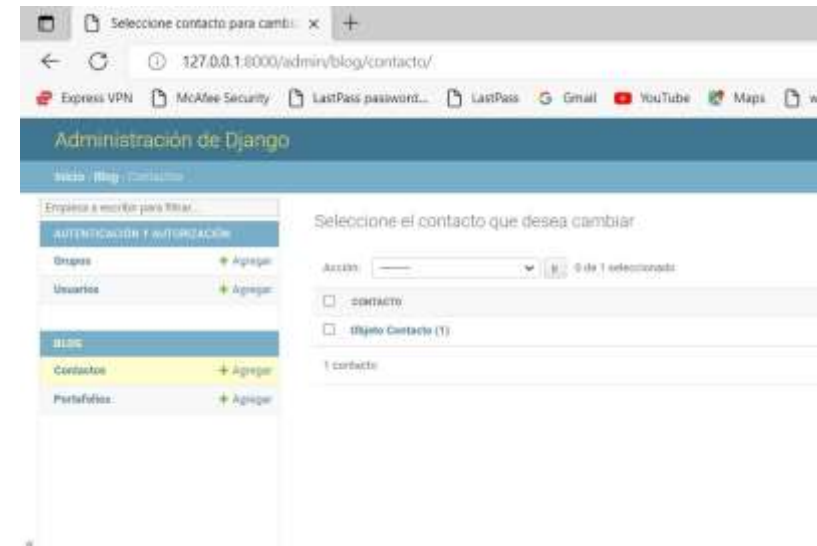
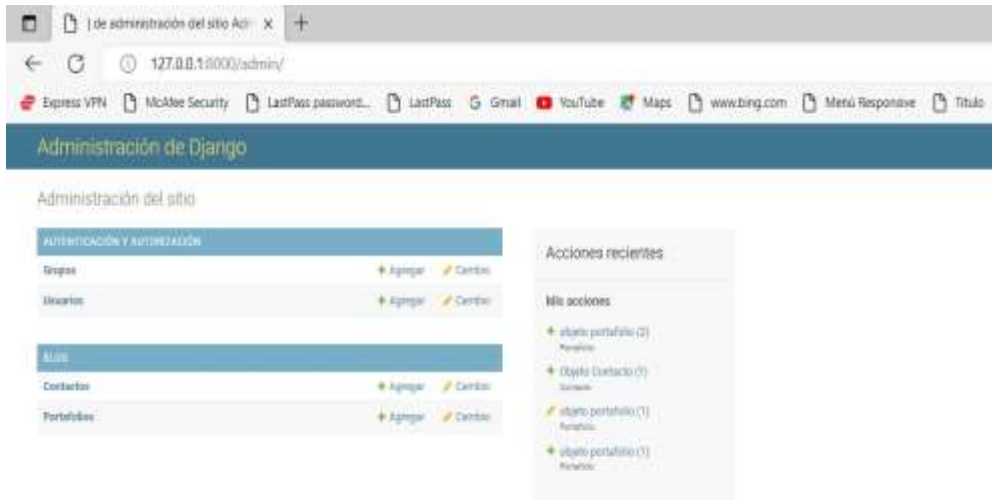
A screenshot of a code editor window showing a Django project. The left sidebar displays a file explorer with a project structure including 'migrations', 'templates', and 'core'. The main editor area shows the 'models.py' file with two model classes: 'portfolio' and 'Contacto'. The 'portfolio' model has fields for 'titulo', 'contenido', and 'imagen'. The 'Contacto' model has fields for 'nombre', 'telefono', 'correo', 'asunto', and 'mensaje'. The code is written in Python and uses Django's ORM syntax.

```
1 from django.db import models
2
3 # Create your models here.
4 class portfolio(models.Model):
5     titulo=models.CharField(max_length=200)
6     contenido=models.TextField()
7     imagen=models.ImageField(upload_to="blog")
8
9 class Contacto(models.Model):
10     nombre=models.TextField()
11     telefono=models.TextField()
12     correo=models.TextField()
13     asunto=models.TextField()
14     mensaje=models.TextField()
```

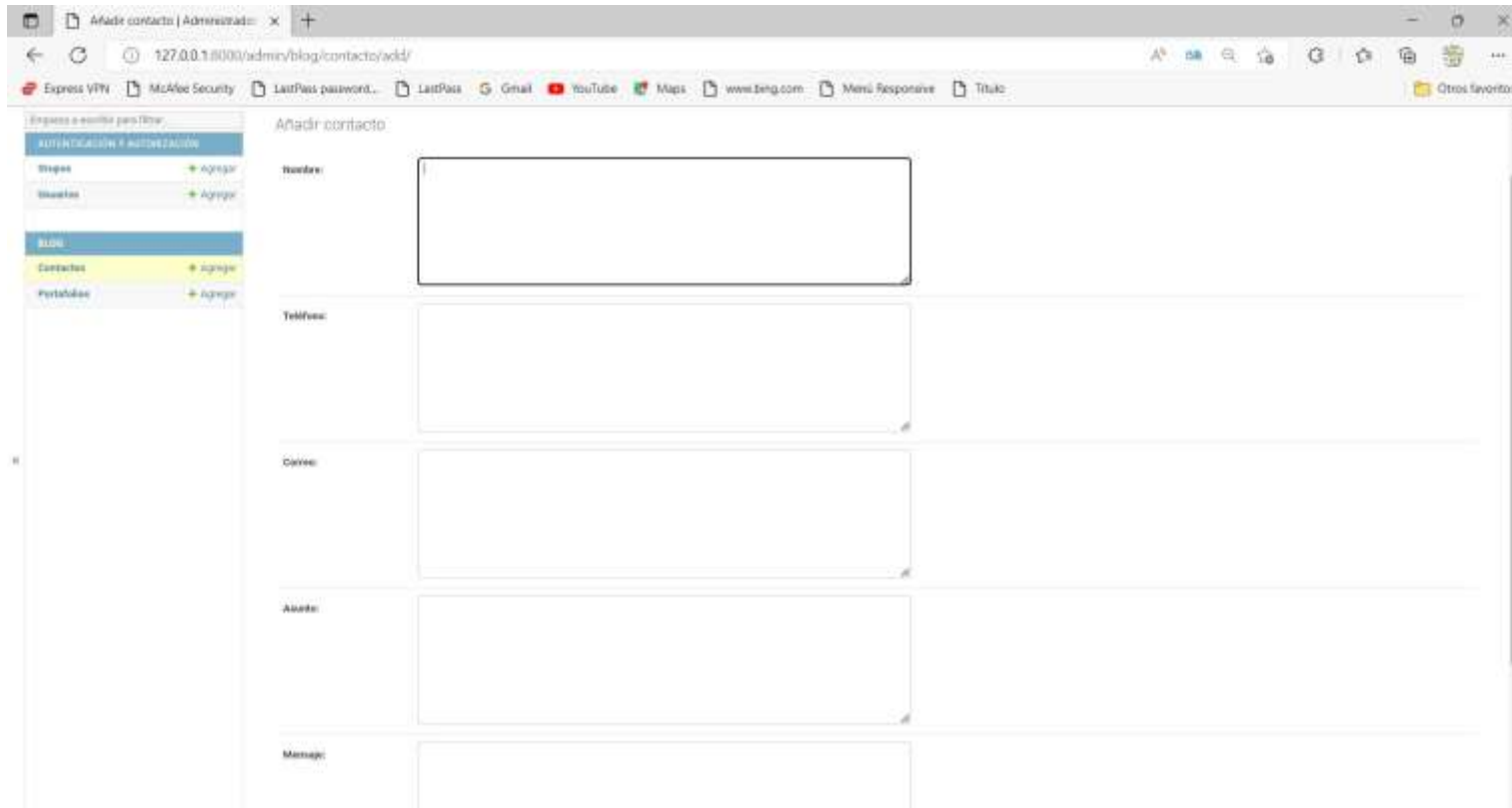
Migrar el Modelo a la base del Panel de Administración..



Integrar el Modelo al Panel de Administración.



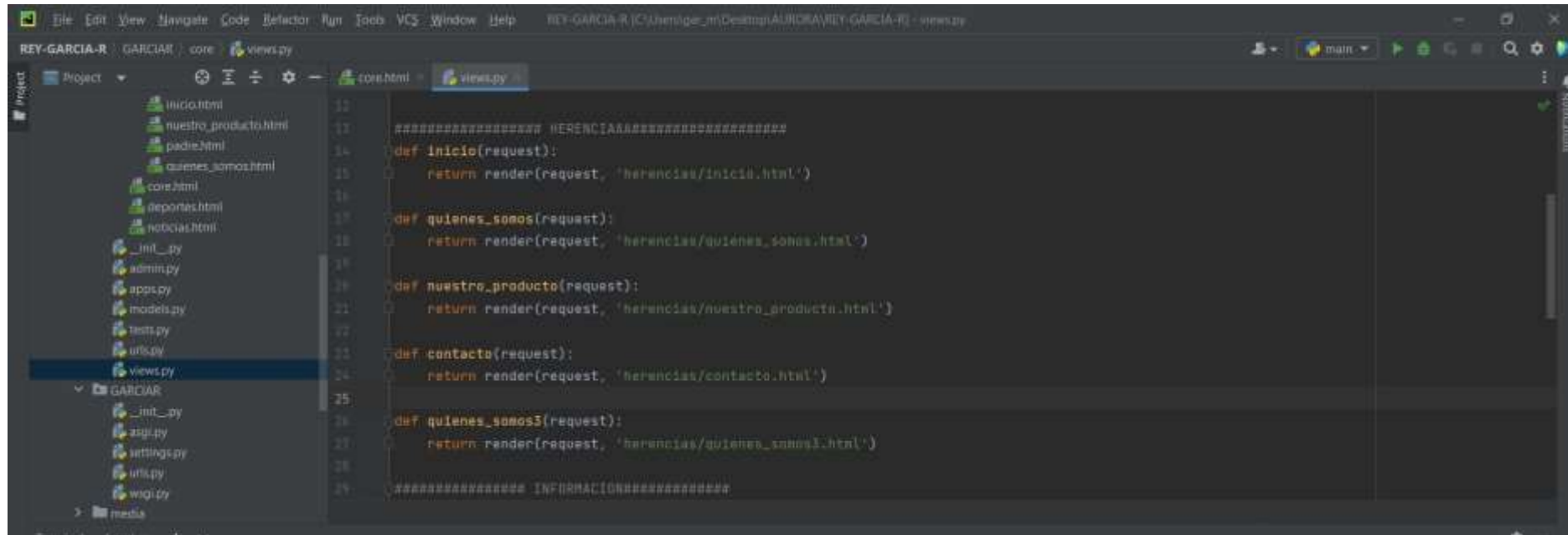
Ingresar información al modelo por el Panel de Administración.



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8000/admin/blog/contacto/add/`. The browser's address bar and tabs are visible at the top. On the left side, there is a sidebar menu with the following items: 'Empresas a evaluar por filtro', 'AUTENTICACIÓN Y AUTORIZACIÓN', 'Empresas' (with a '+ Agregar' button), 'Inventarios' (with a '+ Agregar' button), 'Bases', 'Contactos' (highlighted in yellow, with a '+ Agregar' button), and 'Portafolios' (with a '+ Agregar' button). The main content area is titled 'Añadir contacto' and contains a form with the following fields: 'Nombre' (a large text input field), 'Teléfono' (a text input field), 'Correo' (a text input field), 'Asunto' (a text input field), and 'Mensaje' (a text input field). The form is designed with a clean, modern aesthetic, featuring light gray borders and a white background.

Realizar la consulta de todo lo ingresado en el modelo desde el views





The screenshot shows a code editor with a project structure on the left and a Python file (views.py) open in the main editor. The project structure includes files like inicio.html, nuestro\_producto.html, padre.html, quienes\_somos.html, core.html, deportes.html, noticias.html, \_init\_.py, admin.py, app.py, models.py, tests.py, urls.py, views.py, and a subdirectory GARCIA with \_init\_.py, app.py, settings.py, urls.py, and wsgi.py. The views.py file contains several function definitions for rendering HTML templates, each taking a request object and a template path as arguments. The functions are: inicio(request), quienes\_somos(request), nuestro\_producto(request), contacto(request), and quienes\_somos3(request). Each function returns a render object with the request and a template path. The template paths are: herencias/inicio.html, herencias/quienes\_somos.html, herencias/nuestro\_producto.html, herencias/contacto.html, and herencias/quienes\_somos3.html. The file is named views.py and is located in the core directory.

```
11 ##### HERENCIA #####
12
13 def inicio(request):
14     return render(request, 'herencias/inicio.html')
15
16 def quienes_somos(request):
17     return render(request, 'herencias/quienes_somos.html')
18
19 def nuestro_producto(request):
20     return render(request, 'herencias/nuestro_producto.html')
21
22 def contacto(request):
23     return render(request, 'herencias/contacto.html')
24
25 def quienes_somos3(request):
26     return render(request, 'herencias/quienes_somos3.html')
27
28 ##### INFORMACION #####
```

Mostrar los datos guardados en el modelo al html hijo.

