



# **Desarrollo de software**

## **Estudiantes:**

**Joselyn Benites Villalta.**

## **Curso:**

**3ero G.**

## **Docente:**

**Ing. Carlos Pazmiño.**





## Contenido

1) ¿Qué es Django? .....	3
2) ¿Qué es la máquina virtual en Django? .....	3
3) ¿Qué es MVT en Django? .....	3
4) Crear un proyecto con la máquina virtual. ....	4
5) Descargar los instaladores de Django al proyecto. ....	5
6) Crear un proyecto para programar en Django.....	6
7) Ejecutar el proyecto y el mensaje de felicitaciones. ....	7
8) Crear una Apps core.....	8
9) ¿Qué es la Carpeta Templates? .....	15
10) ¿Qué es la Carpeta static? .....	15
11) Crear un archivo base html en la APPS core.....	16
12) Como se llaman a los CSS desde el archivo base html. ....	16
13) Como consume un archivo hijo html al utilizar la herencia del archivo base html. ....	17
14) Crear un view que llame al html hijo.....	17
15) Crear la urls que llame al views. ....	17
16) Integrar la aplicación APPS core al proyecto principal.....	18
17) Crear las tablas del sistema de usuarios para utilizar el panel de administración. ....	19
18) Crear un usuario para poder ingresar al Panel de Administración...	20
19) Que es un modelo en Django. ....	21
20) Crear un modelo en Django. ....	21
21) Migrar el Modelo a la base del Panel de Administración.....	21
22) Integrar el Modelo al Panel de Administración. ....	22
23) Ingresar información al modelo por el Panel de Administración. ....	22
24) Realizar la consulta de todo lo ingresado en el modelo desde el views. ....	23
25) Mostrar los datos guardados en el modelo al html hijo. ....	24



## 1) ¿Qué es Django?

Django es un framework de aplicaciones web gratuito y de código abierto escrito en Python. Un framework web es un conjunto de componentes que te ayudan a desarrollar sitios web más fácil y rápidamente. Los frameworks sirven para que no tengamos que reinventar la rueda cada vez y que podamos avanzar más rápido al construir un nuevo sitio.

## 2) ¿Qué es la máquina virtual en Django?

Es un software que crea una capa independiente donde se emula el funcionamiento de un ordenador real con todos los componentes de hardware que necesita para funcionar.

## 3) ¿Qué es MVT en Django?

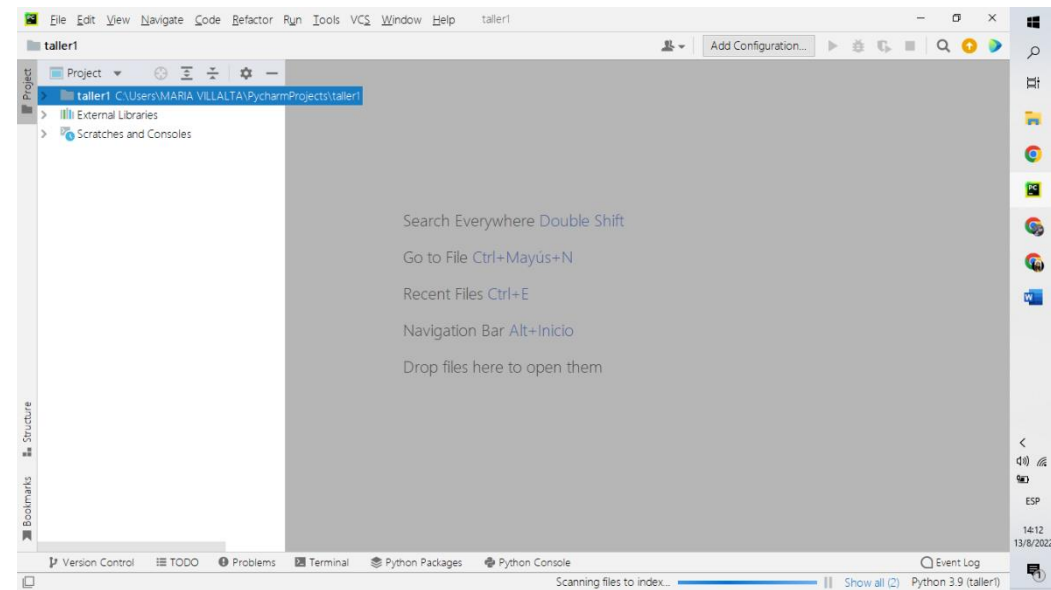
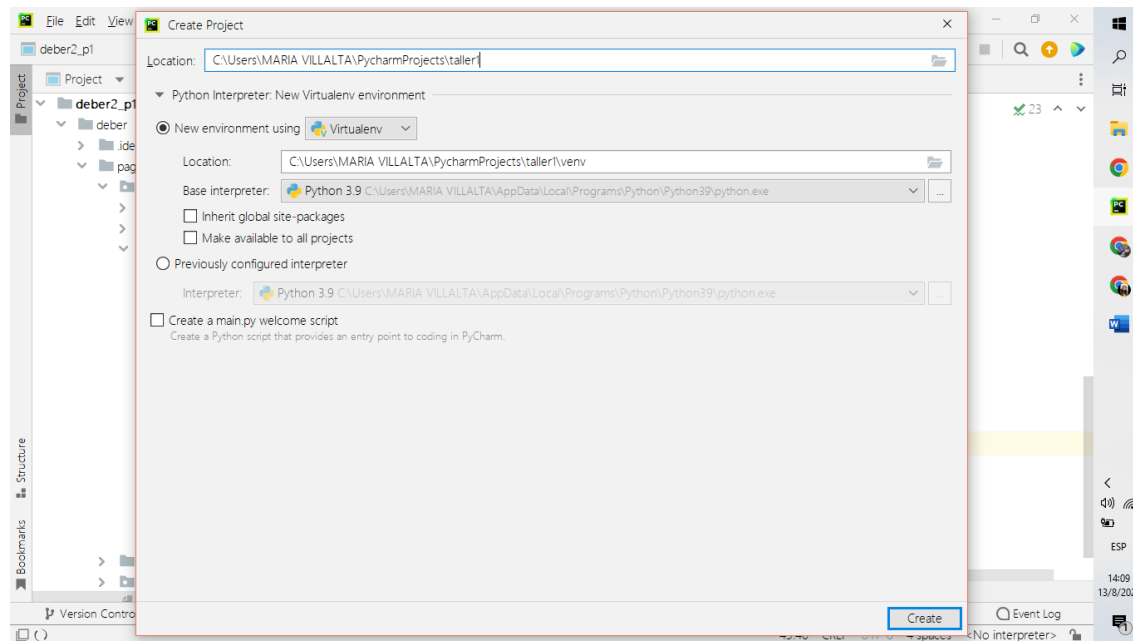
Las siglas en inglés *MTV* corresponden a *Model (Modelo)*, *Template (Plantilla)* y *View (Vista)*. En este post me referiré a las plantillas como templates, ya que su uso en el español es bastante común.

En la práctica el patrón MTV es muy similar al MVC a tal punto que se puede decir que Django es un framework MVC. Realmente este no se desvía demasiado del patrón Modelo Vista Controlador, simplemente lo implementa de una manera distinta y para evitar confusiones es llamado MTV.

En Django, el controlador sigue estando presente, nada más que de una manera intrínseca, ya que todo el framework Django es el controlador.

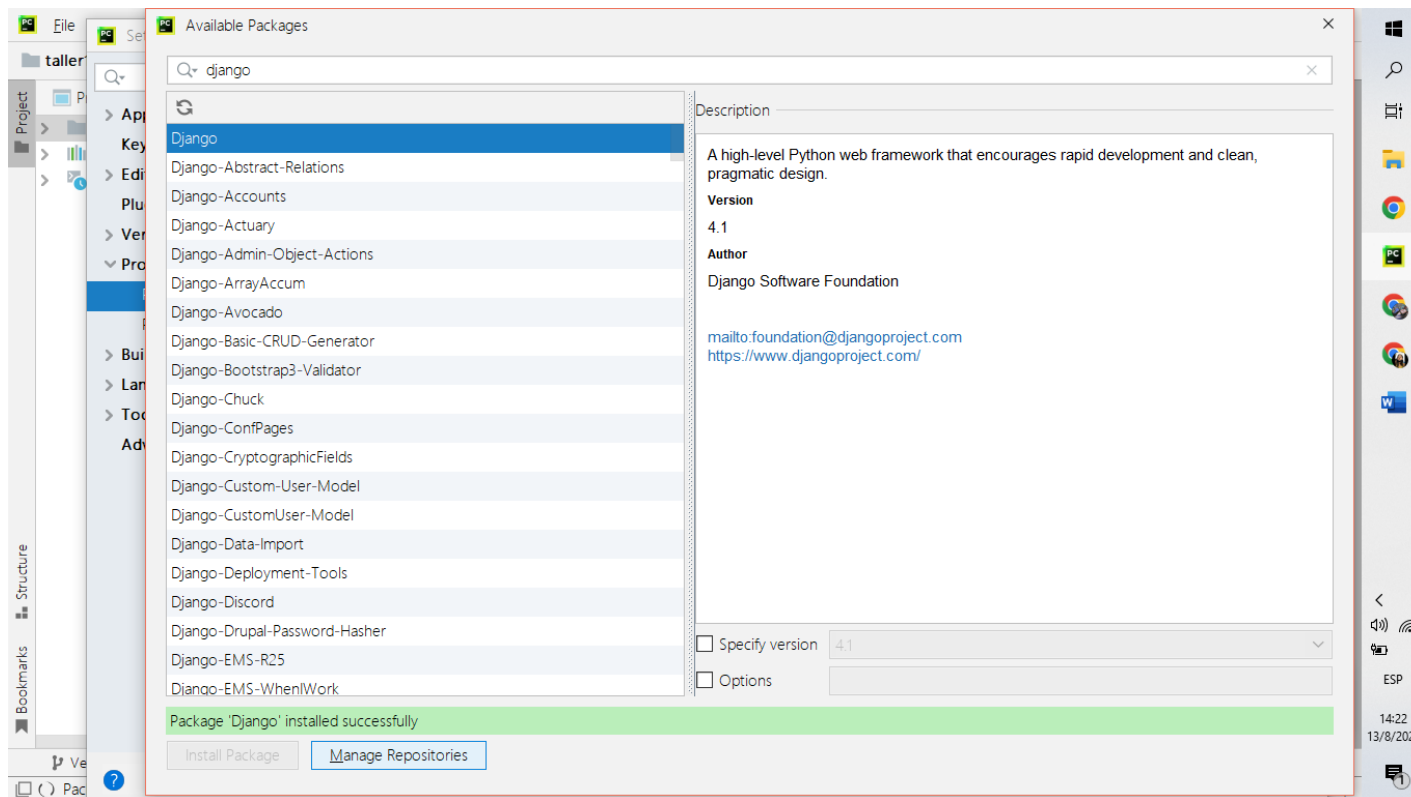
- **Modelo:** Maneja todo lo relacionado con la información, esto incluye como acceder a esta, la validación, relación entre los datos y su comportamiento.
- **Vista:** Es un enlace entre el *modelo* y el *template*. Decide qué información será mostrada y por cual *template*.
- **Template:** Decide como será mostrada la información.

#### 4) Crear un proyecto con la máquina virtual.



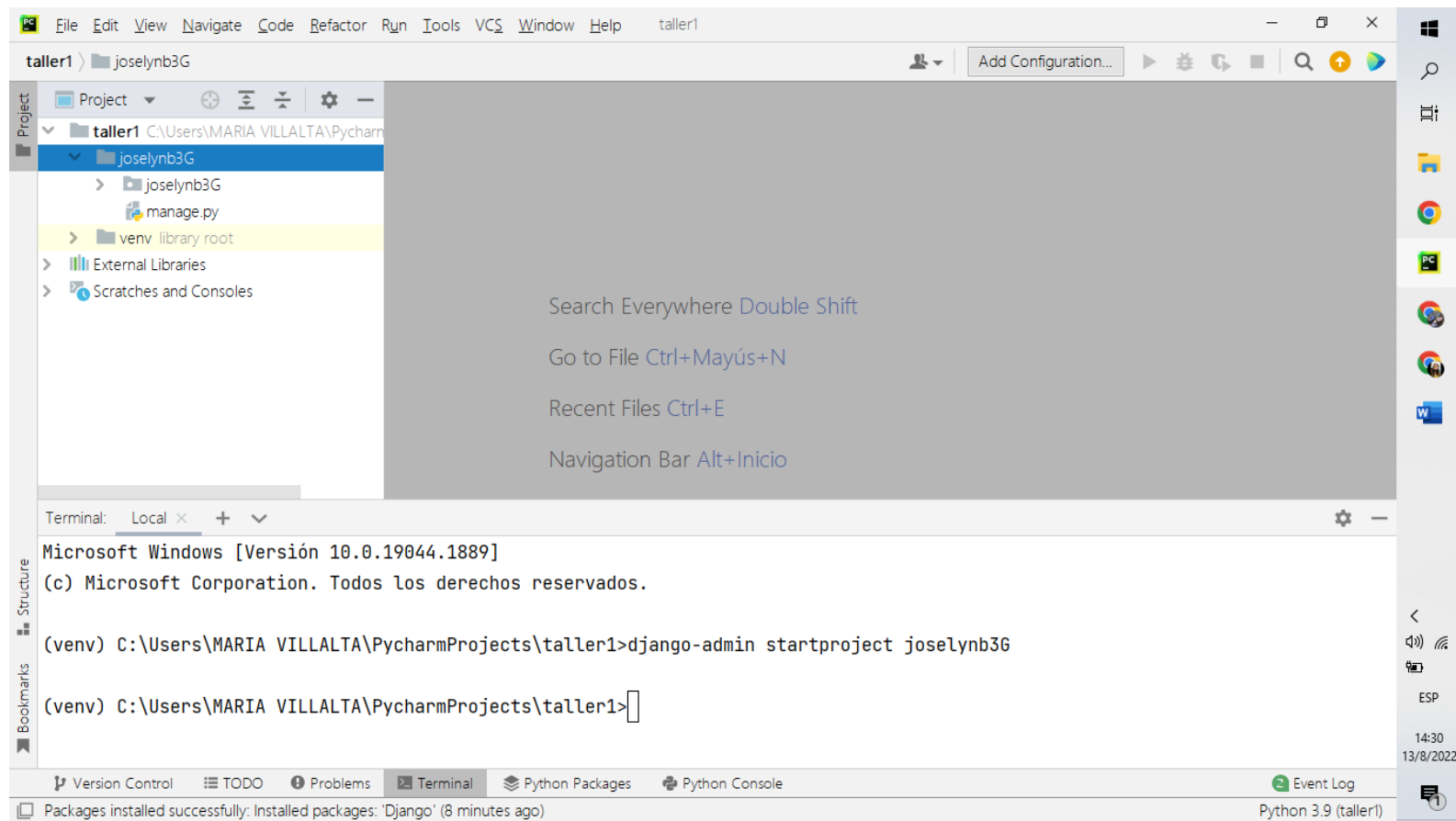
## 5) Descargar los instaladores de Django al proyecto.

- Dentro de nuestro pycharm nos dirigimos a la parte superior de nuestra pantalla y hacemos clic en **file**.
- Luego a settings.
- En settings nos vamos a **Project: taller1** y luego **Python interpreter**
- Le damos clic en **+** y buscamos **Django**, procedemos a instalar el paquete.



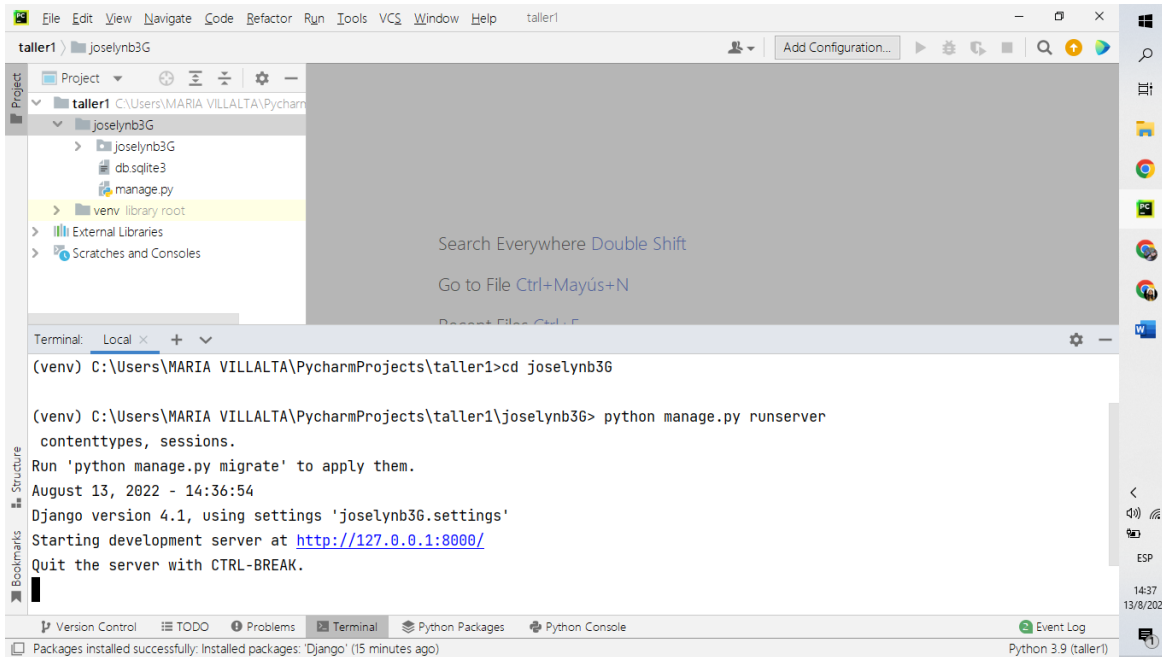
## 6) Crear un proyecto para programar en Django

Para crear un proyecto en django ejecutamos con el comando **django-admin startproject joselynb3G** en el terminal de nuestro pycharm.

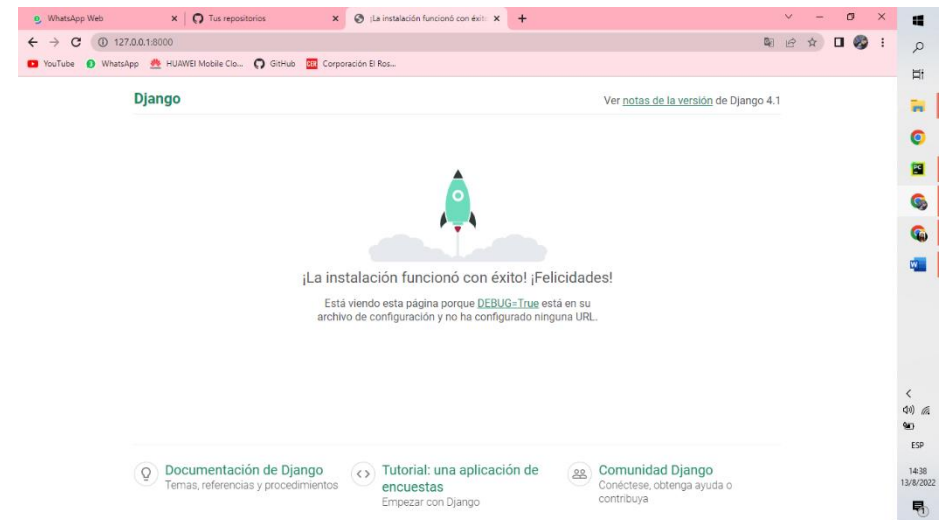


## 7) Ejecutar el proyecto y el mensaje de felicitaciones.

Para ejecutar el proyecto utilizamos el comando **python manage.py runserver**.



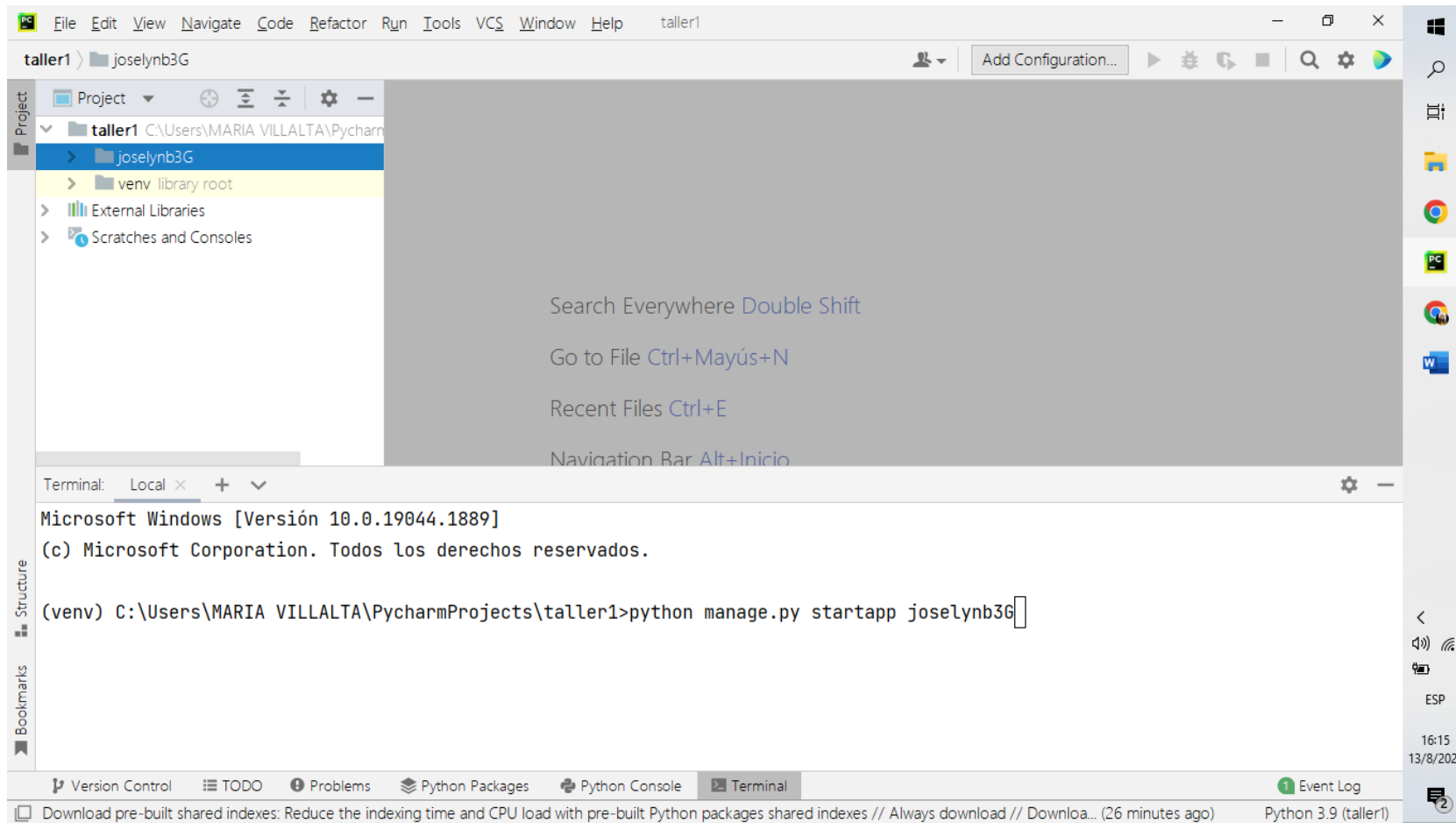
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help taller1
taller1 | joselynb3G
Project
  Project
  C:\Users\MARIA VILLALTA\PycharmProjects\taller1
  joselynb3G
    joselynb3G
    db.sqlite3
    manage.py
    venv library root
  External Libraries
  Scratches and Consoles
Search Everywhere Double Shift
Go to File Ctrl+Mayús+N
Recent Files Ctrl+E
Terminal: Local
(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1>cd joselynb3G
(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1\joselynb3G> python manage.py runserver
contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
August 13, 2022 - 14:36:54
Django version 4.1, using settings 'joselynb3G.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
Version Control TODO Problems Terminal Python Packages Python Console Event Log
Packages installed successfully: Installed packages: 'Django' (15 minutes ago) Python 3.9 (taller1)
```





## 8) Crear una Apps core.

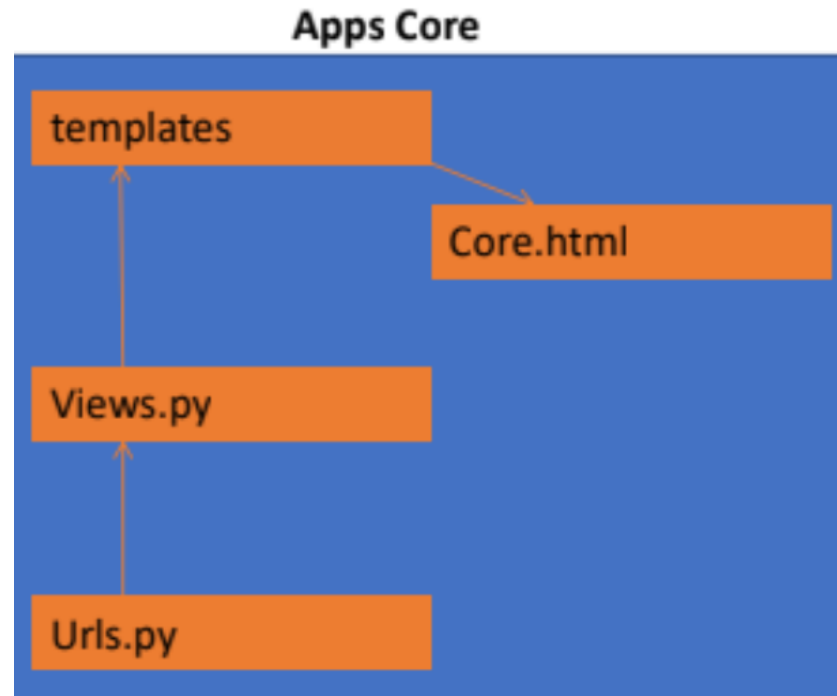
\*En el terminal se escribe el código **python manage.py startapp** y el nombre del proyecto.







\* Debe tener creado los siguiente dentro de la core:





\* Dentro de nuestro Proyecto Core, creamos la carpeta **templates**, dentro de ella creamos un archivo html.

The screenshot shows the PyCharm IDE interface. The top toolbar includes menus like File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The breadcrumb navigation at the top reads: taller1 > joselynb3G > core > templates. The left sidebar shows the Project view with the following structure:

- taller1
  - joselynb3G
    - core
      - migrations
      - templates (selected)
        - core.html (new file)
      - \_\_init\_\_.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - views.py
    - joselynb3G
      - db.sqlite3
      - manage.py
    - venv library root
    - External Libraries
    - Scratches and Consoles

The main editor window displays the content of the new file `core.html` with the following code:

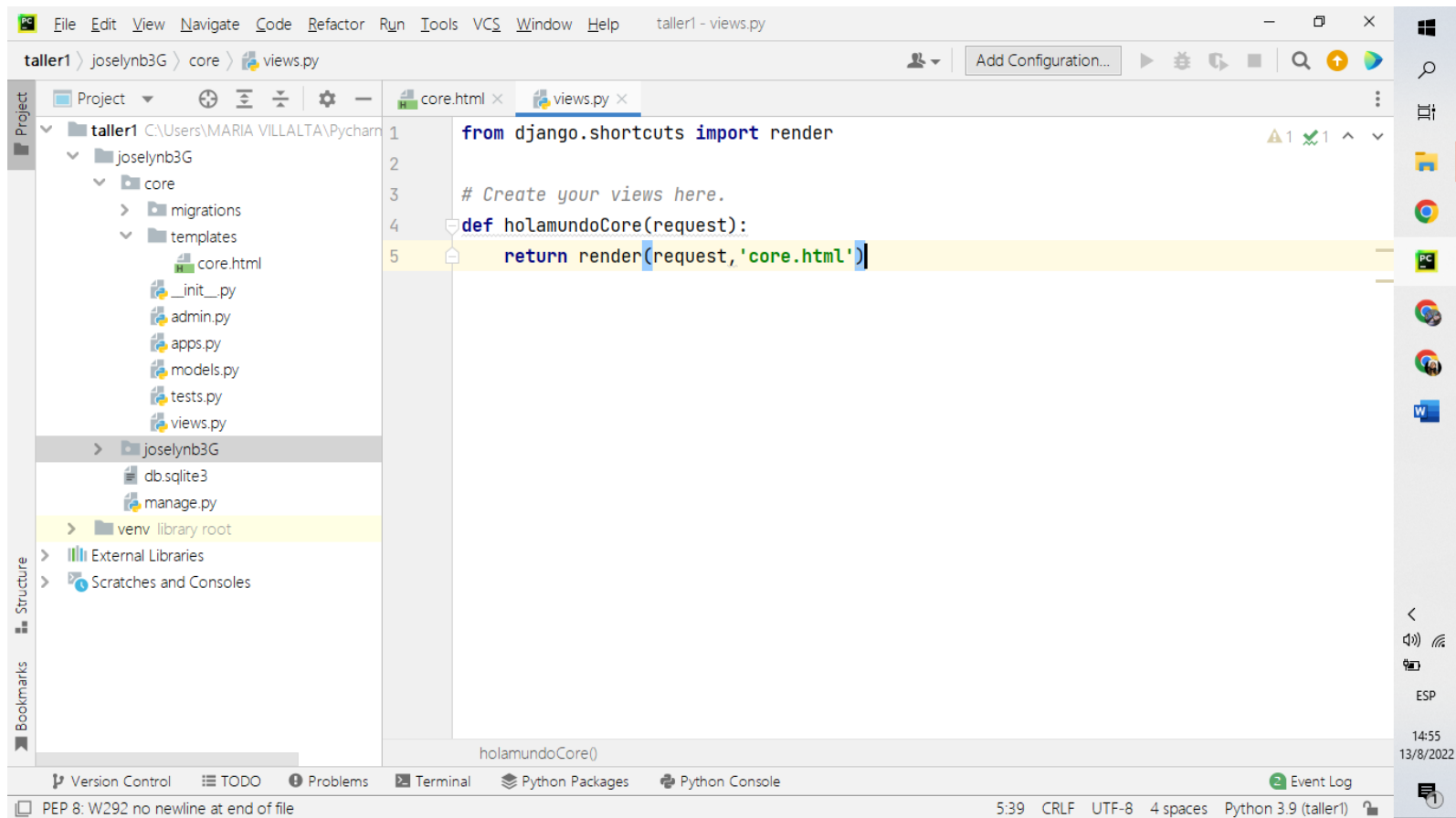
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Title</title>
6 </head>
7 <body>
8
9 </body>
10 </html>
```

The status bar at the bottom indicates: Packages installed successfully: Installed packages: 'Django' (23 minutes ago). The bottom right corner shows the file encoding and settings: 5:17 (5 chars) CRLF UTF-8 2 spaces\* Python 3.9 (taller1).



\* Dentro de nuestro **proyecto Core**, modificamos el **Views**, llamando a nuestro Html.

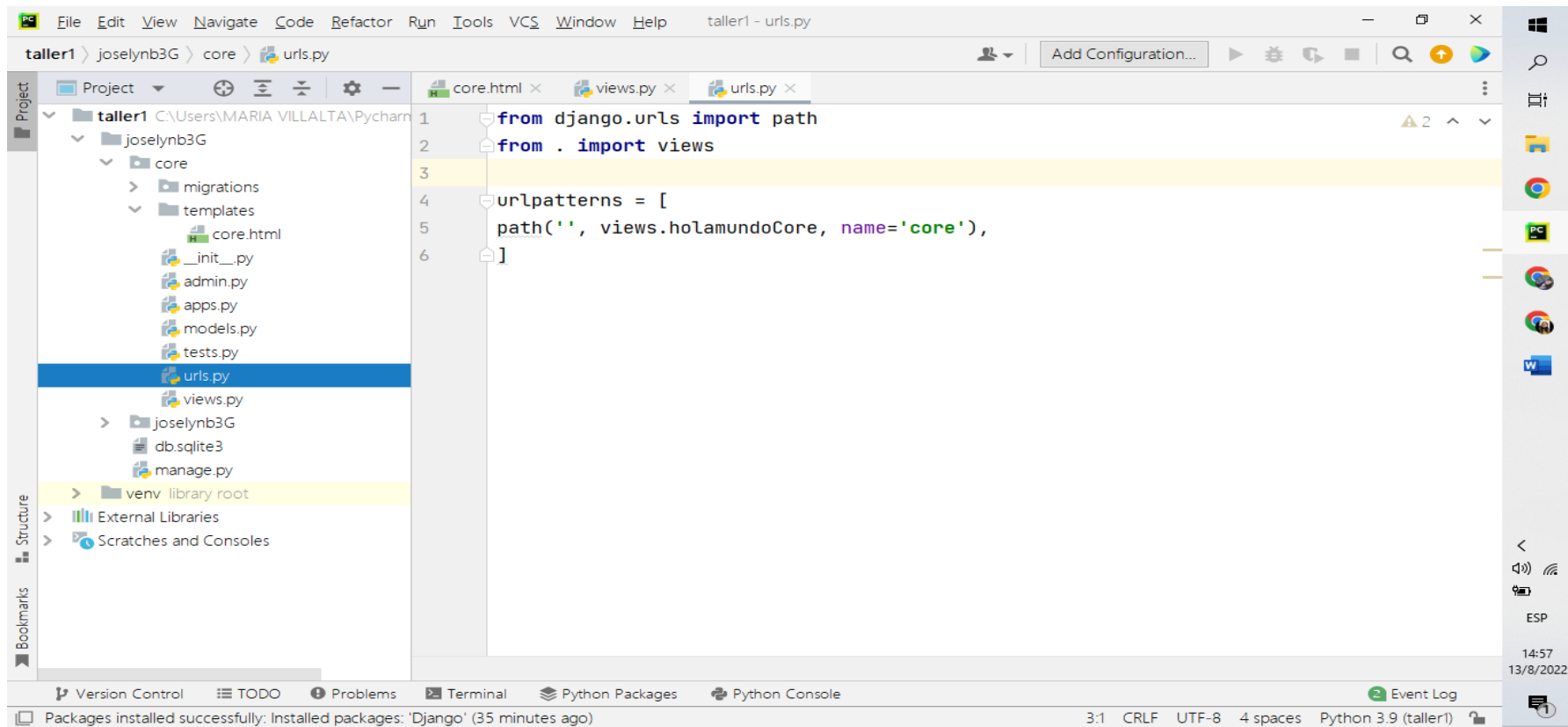
```
def holamundoCore(request):  
    return render(request, 'base.html')
```





\* Dentro de nuestro **proyecto Core**, creamos el `Urls.py`

```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.holamundoCore, name='core'),
]
```

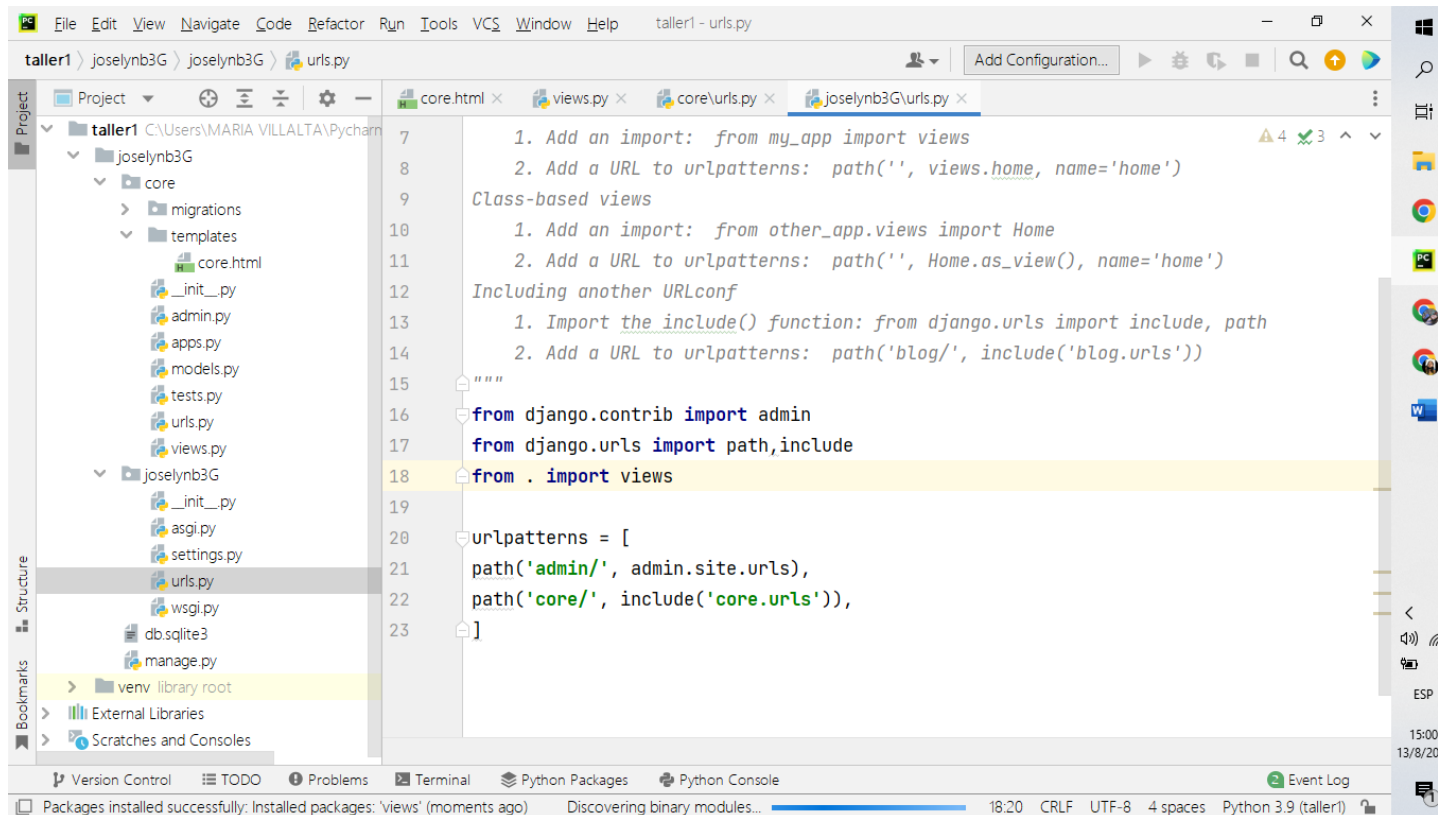




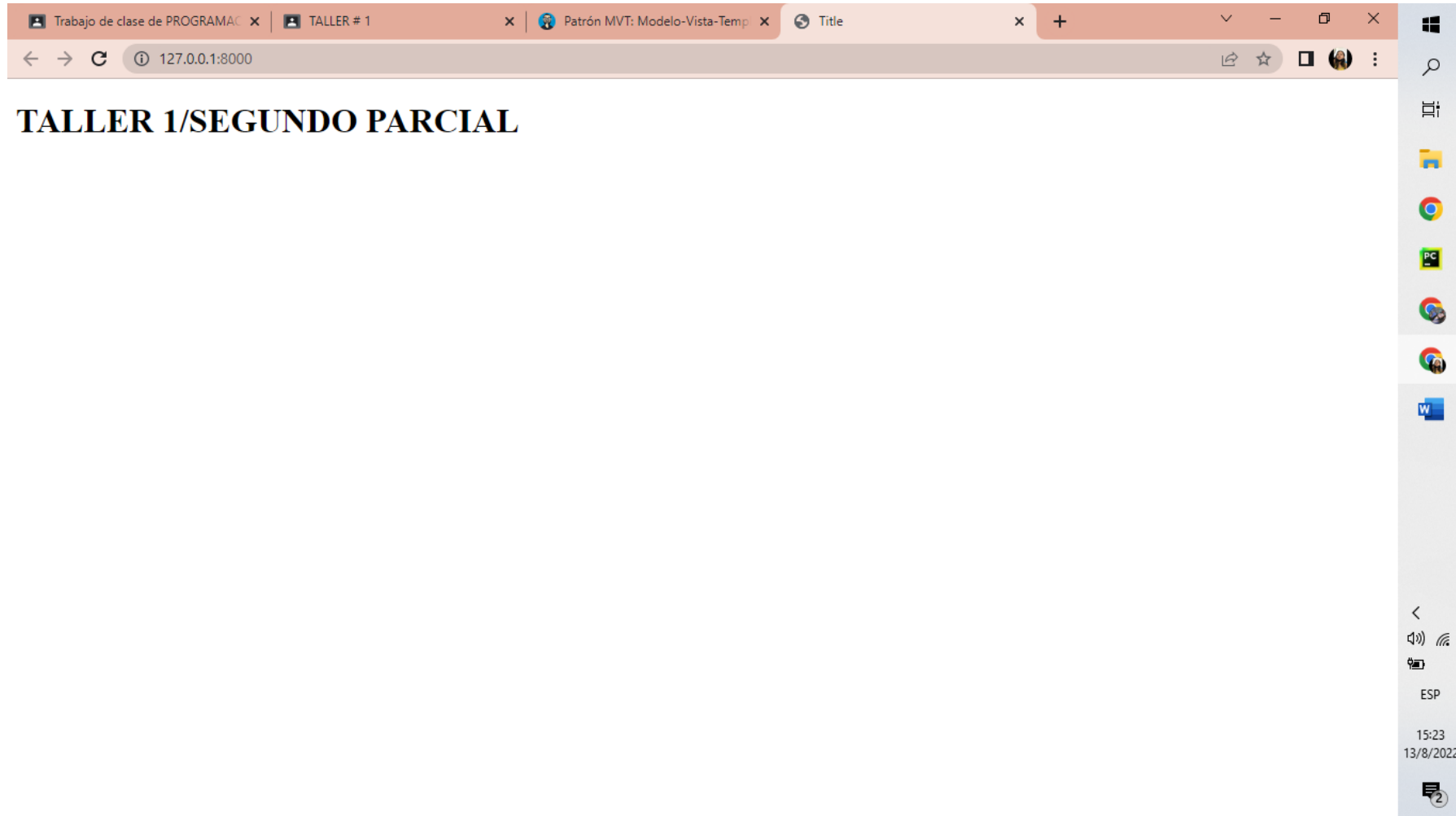
\* Dentro de nuestro proyecto principal(taller1) Urls.py debemos llamar al Urls.py de nuestra app creada core.

```
from django.contrib import admin
from django.urls import path,include
```

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('core.urls')),
]
```

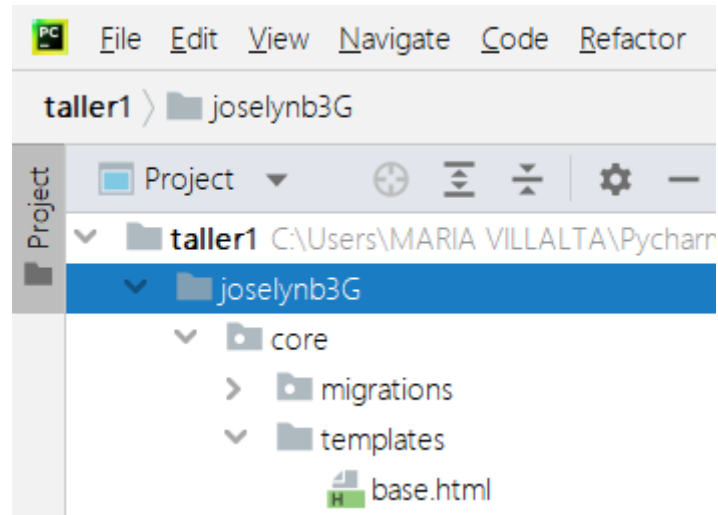


\*Volvemos a levantar el servidor con el comando **python manage.py runserver** y listo.



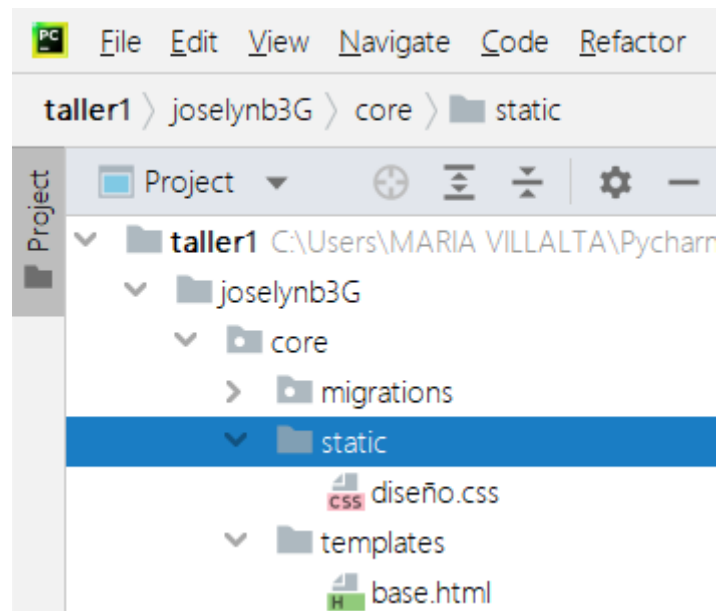
## 9) ¿Qué es la Carpeta Templates?

En un archivo de texto que determina la estructura o diseño de un archivo (como una página HTML), con marcadores usados para representar el contenido real. **Django** automáticamente buscará plantillas en un directorio llamado '**templates**' de su aplicación.

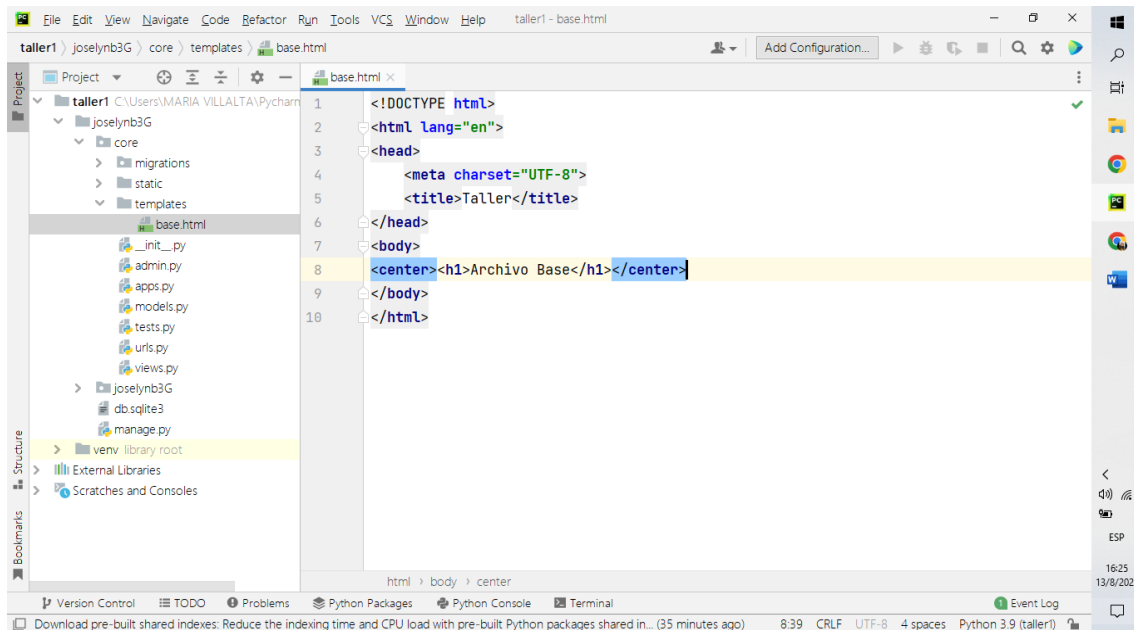


## 10) ¿Qué es la Carpeta static?

Esta carpeta contiene todos los archivos de estilos: css, etc.



## 11) Crear un archivo base html en la APPS core.



## 12) Como se llaman a los CSS desde el archivo base html.

Se lo llama con :

```
{%load static%}
```

```
<link rel="stylesheet" href="">
```

Y se lo ubica en el head:





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

El hijo recibe la herencia directa del archivo padre mediante el comando:

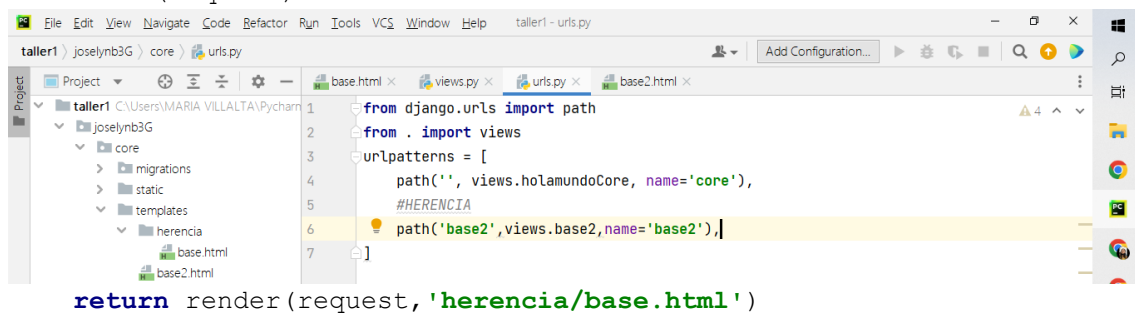
```
{% extends
'carpeta_de_la_herencia/archivo_padre_herencia.html' %}
```

```
{% extends 'herencia/base.html' %}
<!DOCTYPE html>
<html lang="es">
```

### 14) Crear un view que llame al html hijo.

Es llamado de la siguiente manera:

```
def base2(request):
```



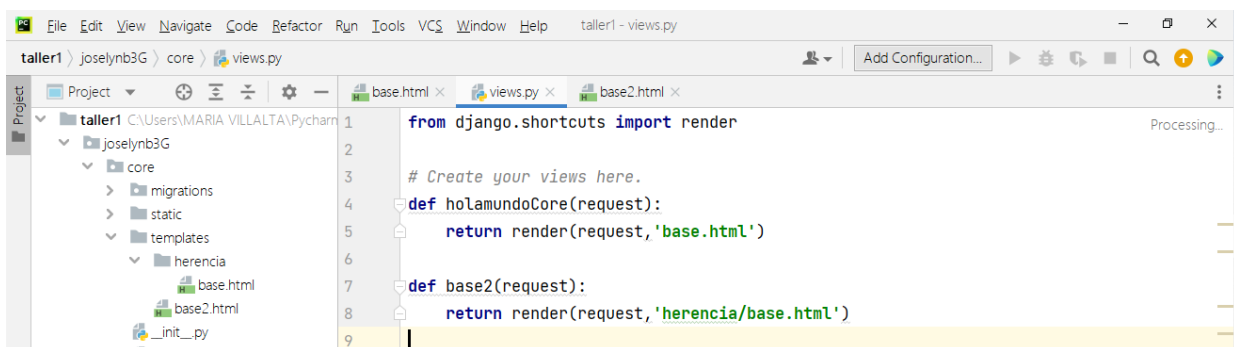
```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.holamundoCore, name='core'),
    #HERENCIA
    path('base2', views.base2, name='base2'),
]

return render(request, 'herencia/base.html')
```

### 15) Crear la urls que llame al views.

Llamado de la siguiente manera:

```
path('base2', views.base2, name='base2'),
```



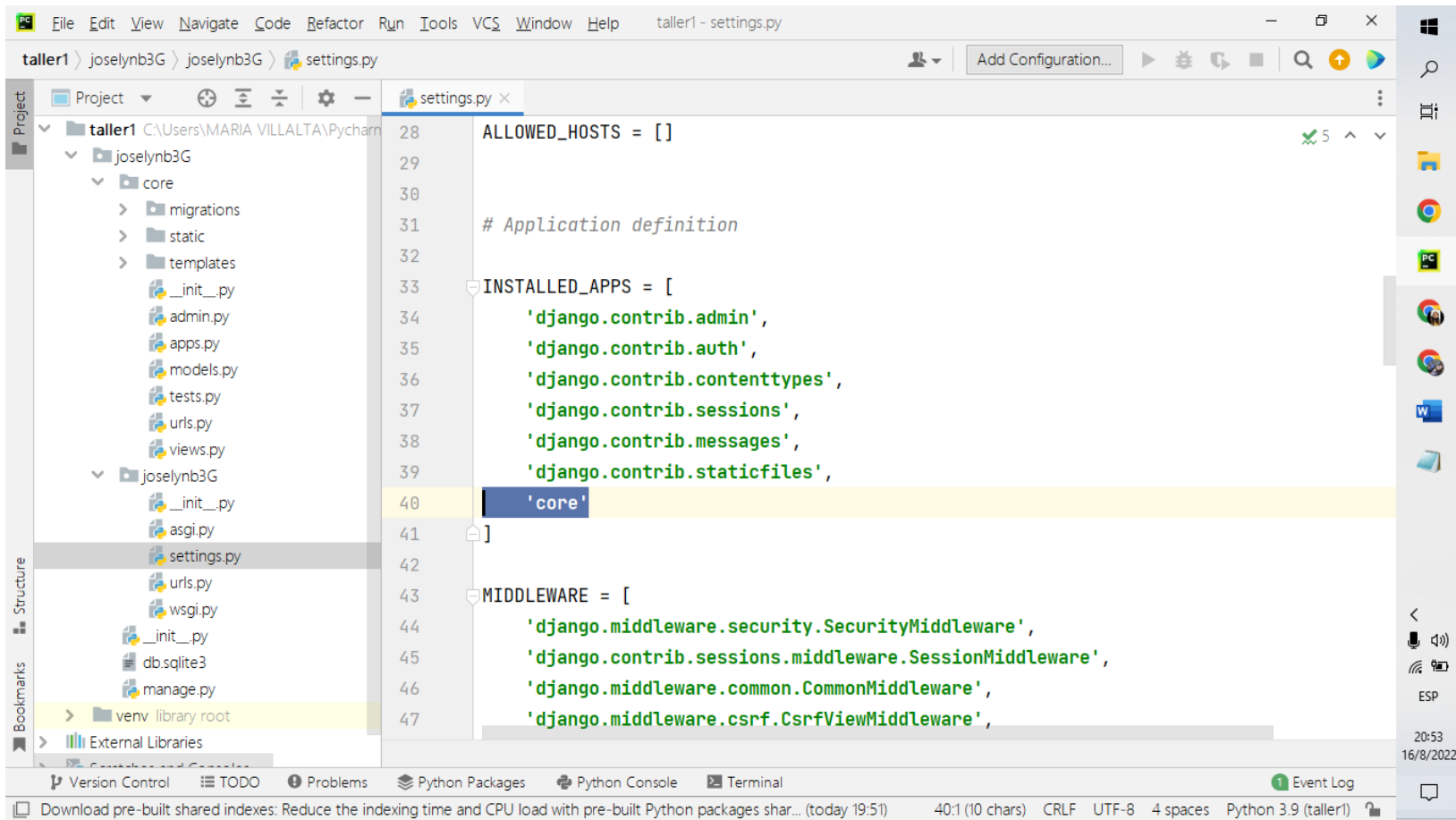
```
from django.shortcuts import render

# Create your views here.
def holamundoCore(request):
    return render(request, 'base.html')

def base2(request):
    return render(request, 'herencia/base.html')
```

## 16) Integrar la aplicación APPS core al proyecto principal.

\*Nos dirigimos al **settings.py** en la carpeta principal para poner nuestro core y que Django lo reconozca como app.

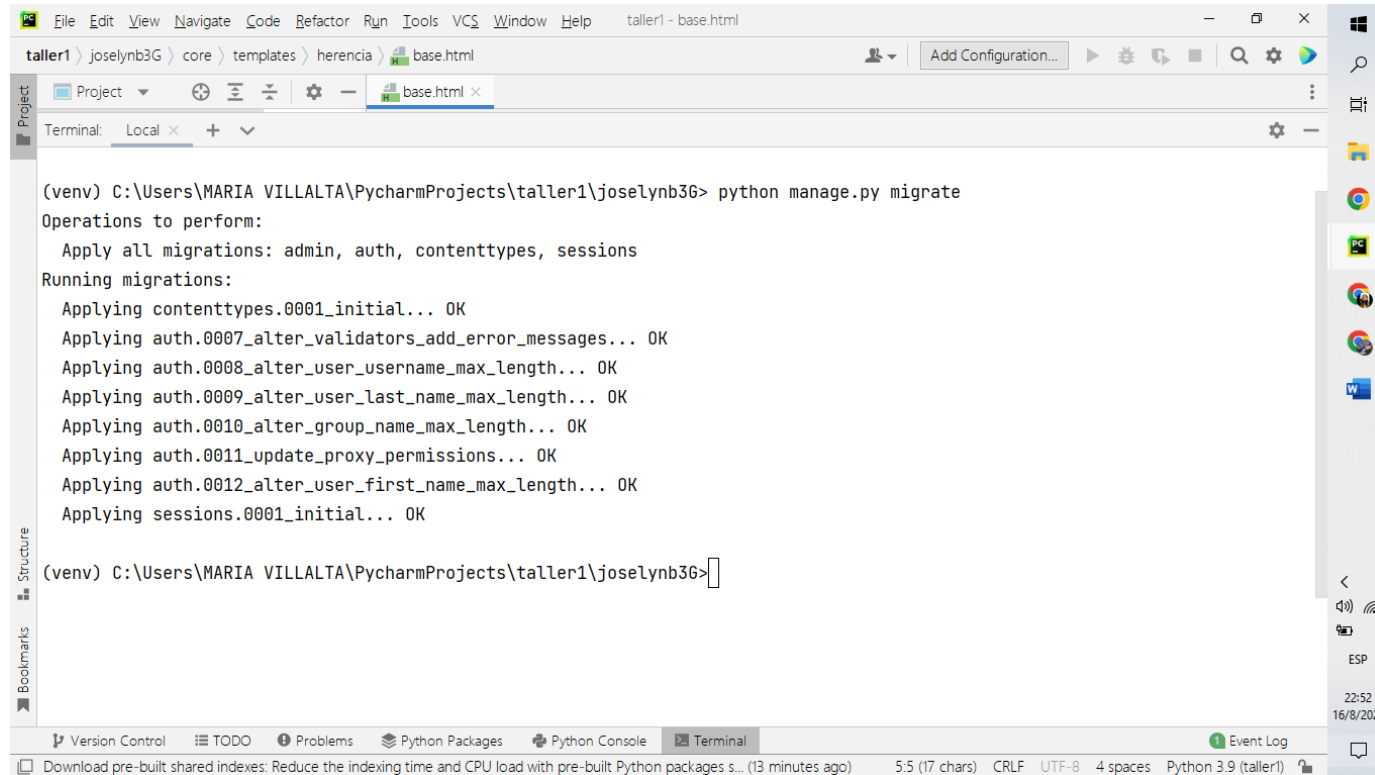


```
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'core'
41 ]
42
43 MIDDLEWARE = [
44     'django.middleware.security.SecurityMiddleware',
45     'django.contrib.sessions.middleware.SessionMiddleware',
46     'django.middleware.common.CommonMiddleware',
47     'django.middleware.csrf.CsrfViewMiddleware',
```

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

Primero que todo migramos las tablas que Django necesita para subir toda nuestra información a la base de datos.

El comando es: **python manage.py migrate**



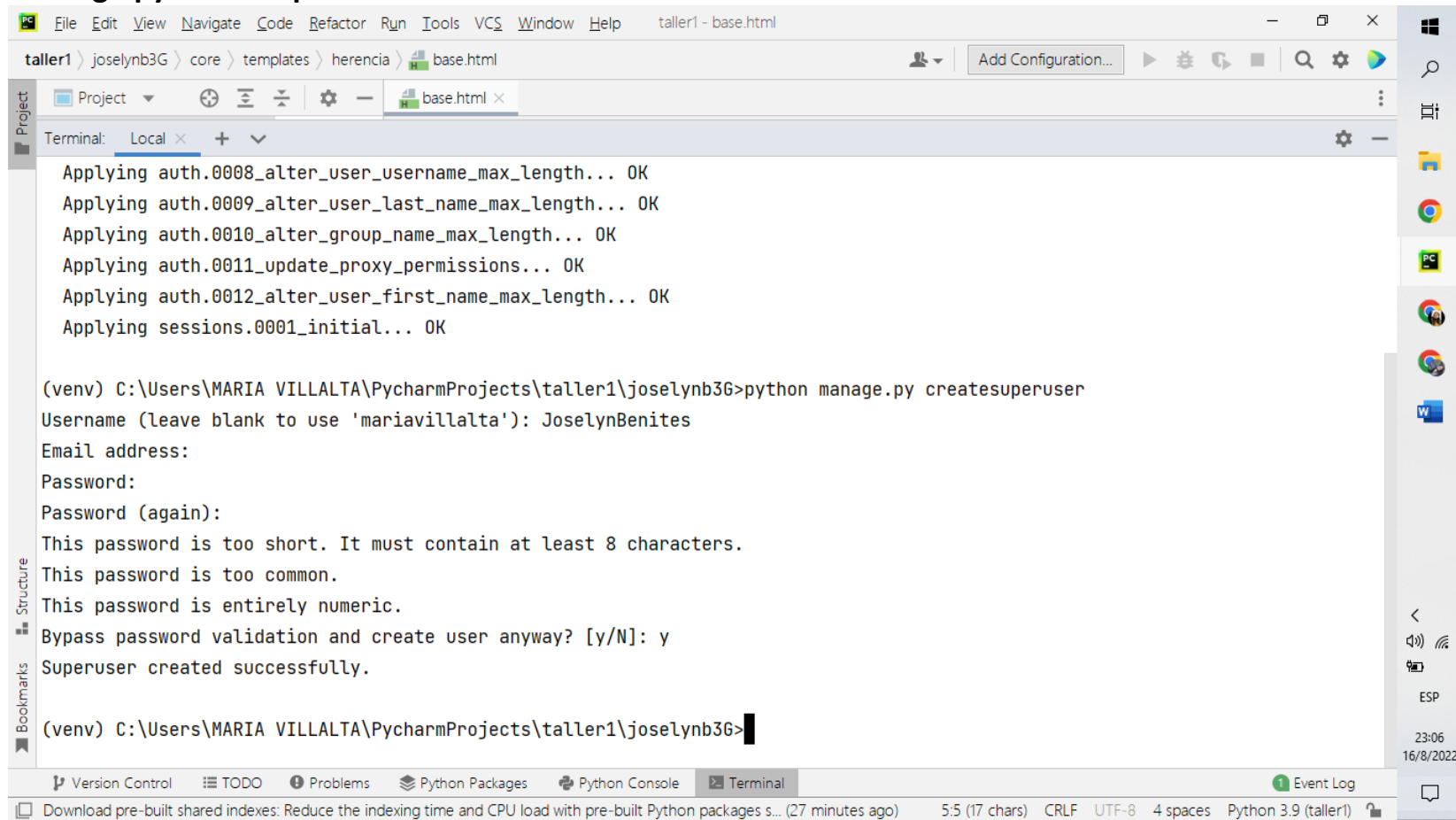
```
(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1\joselynb3G> python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK

(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1\joselynb3G>
```

## 18) Crear un usuario para poder ingresar al Panel de Administración.

Para la creación de usuarios utilizamos el siguiente comando en la terminal:

**python manage.py createsuperuser**



```
taller1 > joselynb3G > core > templates > herencia > base.html
Terminal: Local x + v
Applying auth.0008_alter_user_username_max_length... OK
Applying auth.0009_alter_user_last_name_max_length... OK
Applying auth.0010_alter_group_name_max_length... OK
Applying auth.0011_update_proxy_permissions... OK
Applying auth.0012_alter_user_first_name_max_length... OK
Applying sessions.0001_initial... OK

(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1\joselynb3G>python manage.py createsuperuser
Username (Leave blank to use 'mariavillalta'): JoselynBenites
Email address:
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.

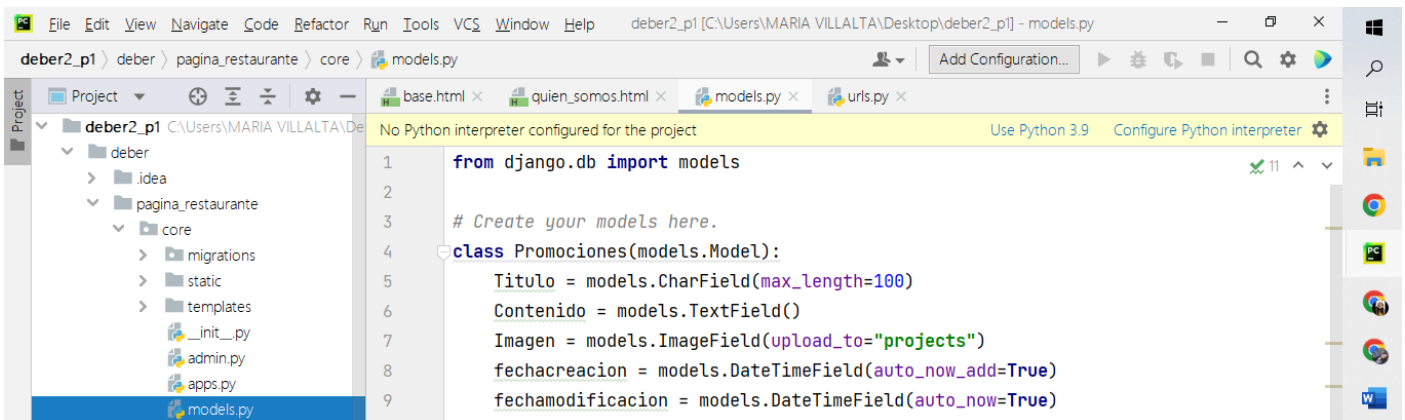
(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1\joselynb3G>
```

## 19) 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.

## 20) Crear un modelo en Django.

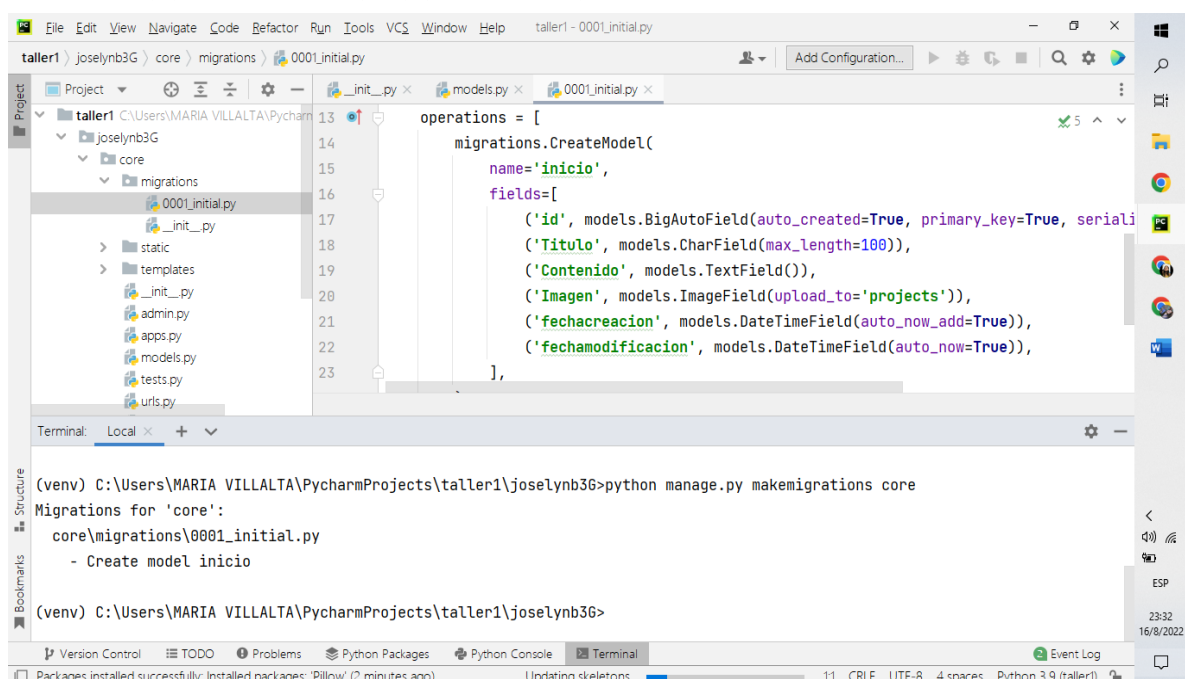
Necesitamos crear una clase y agregarles los atributos.



```
1 from django.db import models
2
3 # Create your models here.
4 class Promociones(models.Model):
5     Titulo = models.CharField(max_length=100)
6     Contenido = models.TextField()
7     Imagen = models.ImageField(upload_to="projects")
8     fechacreacion = models.DateTimeField(auto_now_add=True)
9     fechamodificacion = models.DateTimeField(auto_now=True)
```

## 21) Migrar el Modelo a la base del Panel de Administración.

Para migrar los modelos utilizamos el siguiente comando en la terminal: **python manage.py makemigrations core**



```
operations = [
    migrations.CreateModel(
        name='inicio',
        fields=[
            ('id', models.BigAutoField(auto_created=True, primary_key=True, seriali
            ('Titulo', models.CharField(max_length=100)),
            ('Contenido', models.TextField()),
            ('Imagen', models.ImageField(upload_to='projects')),
            ('fechacreacion', models.DateTimeField(auto_now_add=True)),
            ('fechamodificacion', models.DateTimeField(auto_now=True)),
        ],
    ),
]
```

```
(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1\joselynb3G>python manage.py makemigrations core
Migrations for 'core':
  core\migrations\0001_initial.py
    - Create model inicio

(venv) C:\Users\MARIA VILLALTA\PycharmProjects\taller1\joselynb3G>
```

## 22) Integrar el Modelo al Panel de Administración.

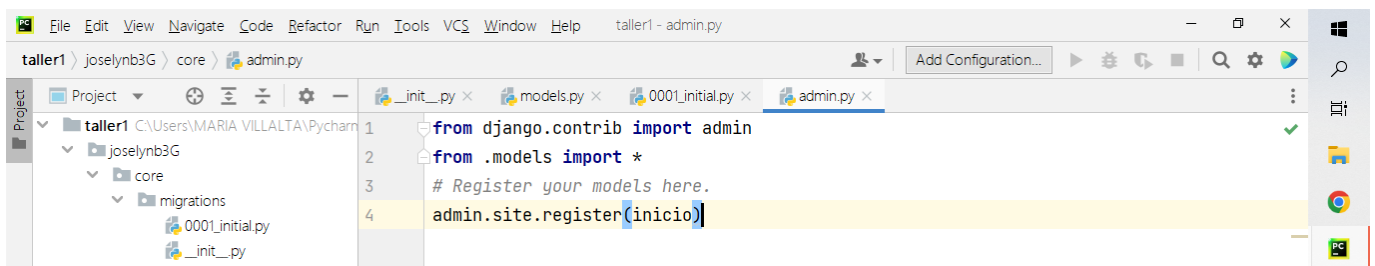
Para que el modelo lo reconozca nuestro panel de administración, debemos añadirlo a nuestro archivo admin.py

Primero importamos nuestro modelo con el comando:

**from .models import \***

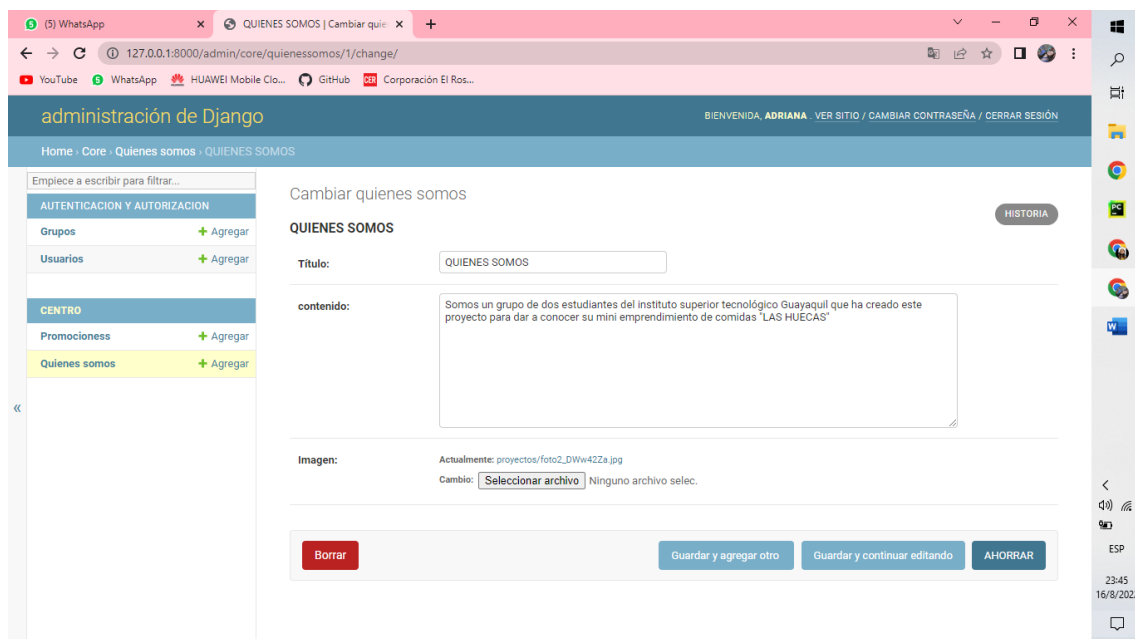
Luego utilizamos la siguiente sentencia y ponemos el nombre de nuestro modelo en el paréntesis:

**admin.site.register(inicio)**



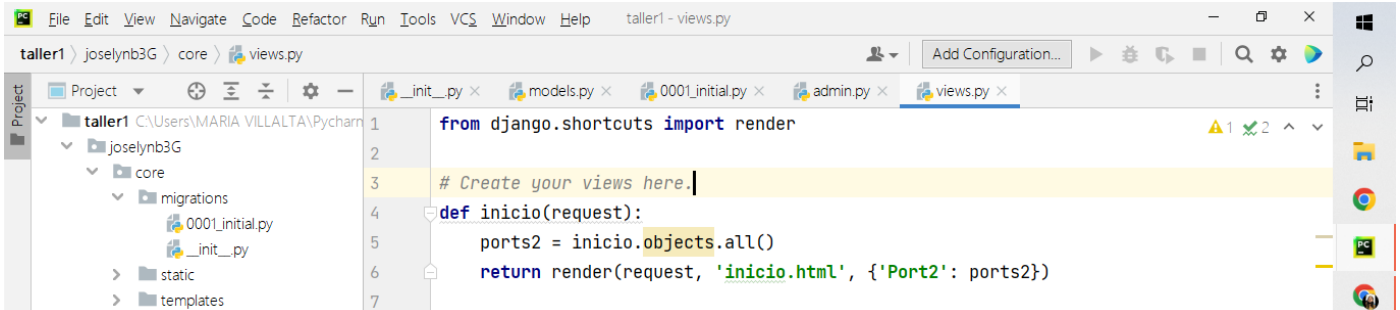
## 23) Ingresar información al modelo por el Panel de Administración.

Con el usuario y contraseña, ingresamos a nuestro admin y ingresamos información en el modelo.



## 24) Realizar la consulta de todo lo ingresado en el modelo desde el views.

Desde la views la podemos ver de la siguiente manera:

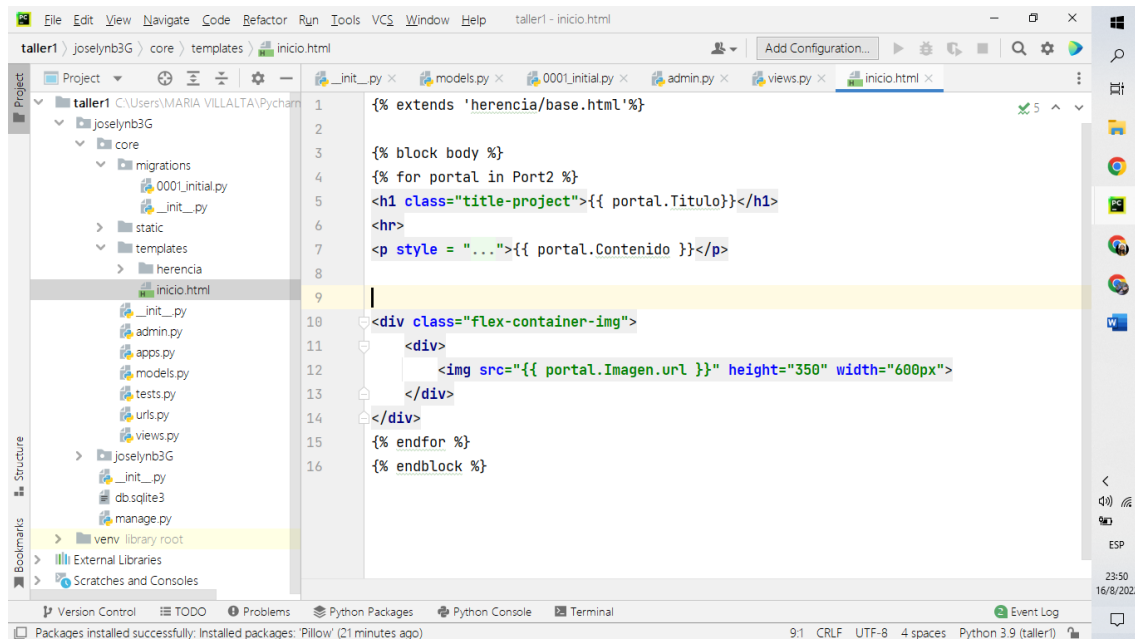


```

1 from django.shortcuts import render
2
3 # Create your views here.
4 def inicio(request):
5     ports2 = inicio.objects.all()
6     return render(request, 'inicio.html', {'Port2': ports2})
7

```

Para que se muestres en la página la agregamos también en el html:



```

1 {% extends 'herencia/base.html' %}
2
3 {% block body %}
4     {% for portal in Port2 %}
5     <h1 class="title-project">{{ portal.Titulo }}</h1>
6     <hr>
7     <p style = "...">{{ portal.Contenido }}</p>
8
9
10    <div class="flex-container-img">
11        <div>
12            
13        </div>
14    </div>
15    {% endfor %}
16    {% endblock %}

```



## 25)Mostrar los datos guardados en el modelo al html hijo.

Los datos ingresados se mostrarán de la siguiente manera:

