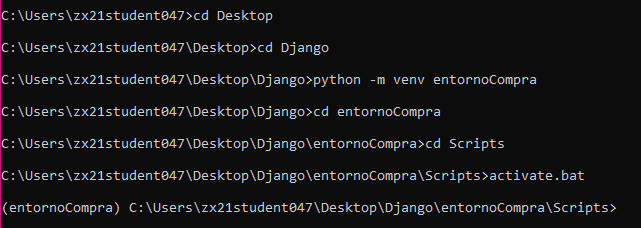
**Proyecto Django para dummies**

1. Creamos el entorno, para eso nos movemos a la carpeta en la que queremos que se cree.

**python -m venv *entornoCompra***

1. Ahora tenemos que activar el proyecto

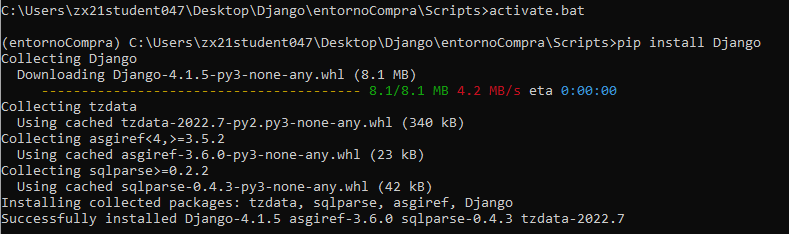
**miproyecto\Scripts\activate.bat**

****

1. Ahora tenemos que instalar Django en nuestro entorno.

**pip install Django**

**django-admin --version**: nos indica la versión de Django instalada



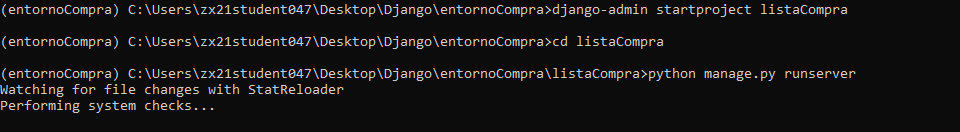
1. Ahora creamos un proyecto dentro de nuestro entorno

**django-admin startproject listaCompra**

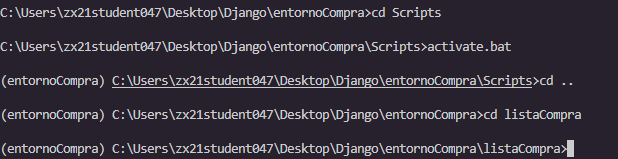
1. Ahora ponemos en marcha el servidor. Y con esto se nos abre la pagina web.

**python manage.py runserver**

Lo podemos ver en http://127.0.0.1:8000/

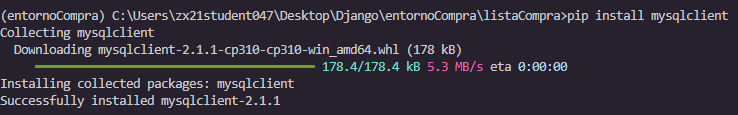
****

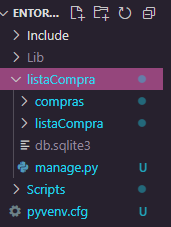
1. Abrimos Visual Studio, de ahora en adelante trabajaremos desde aquí. Abrimos la terminal, pero no usaremos powershell, hay que cambiar a cmd. **Siempre tenemos que entrar al entorno.**



1. Al querer conectarnos a una base de datos, tenemos que instalar mysql

**pip install mysqlclient**

****



1. Django, trabaja con aplicaciones (apps), es el siguiente paso a realizar.

**python manage.py startapp compras**

1. Dentro de la carpeta lista compras, tenemos un archivo settings.py, en el que iremos a la línea 76. Y Copiaremos lo siguiente:

**DATABASES = {**

**'default': {**

**'ENGINE': 'django.db.backends.mysql',**

**'NAME': '*nombreDB*',**

**'USER': '*nombreusuario*',**

**'PASSWORD': '*pass*',**

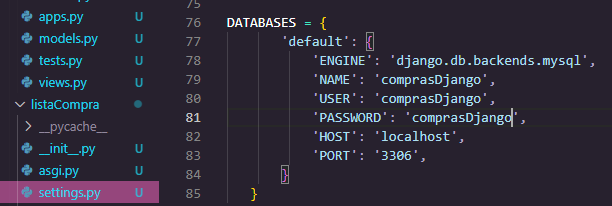
**'HOST': 'localhost',**

**'PORT': '3306',**

**}**

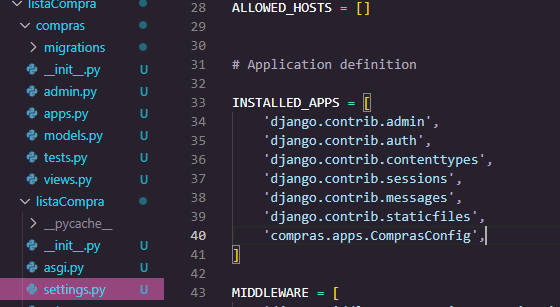
**}**

Esto sirve para configurar la conexión a la base de datos. Todo lo que está en cursiva lo cambiaremos a la configuración que nosotros tengamos en nuestra base de datos.

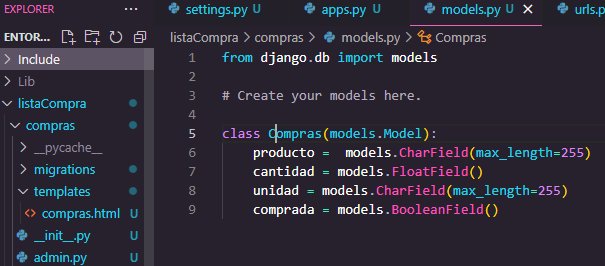


1. Ahora crearemos la base de datos en [phpMyAdmin](http://localhost/phpmyadmin/). Con los datos que hemos puesto en los settings.
2. En el mismo settings.py vamos a la linea 33, y añadimos:

compras.apps.ComprasConfigad



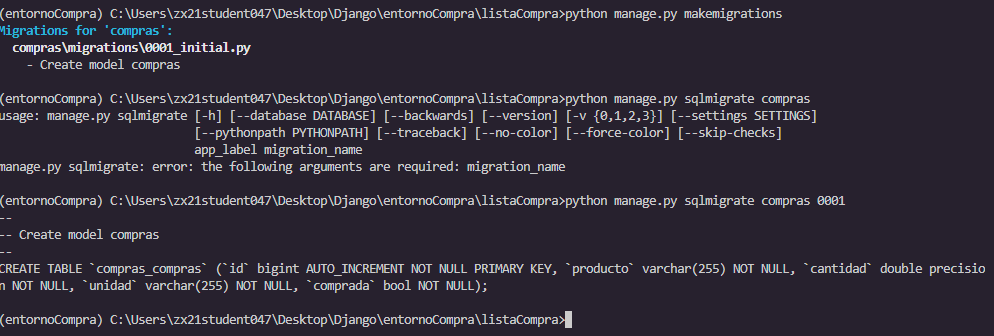
1. Antes de desarrollar debemos pensar que vamos a tener en nuestra BBDD. Es decir, creamos las tablas necesarias de nuestra app. Aquí se llaman modelos, por lo que tenemos que ir a la carpeta compras, y el archivo models.py.

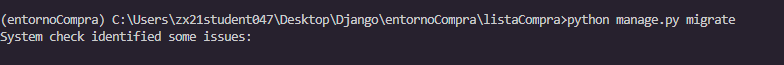


1. Una vez tenemos todo configurado, tenemos que crear las tablas, y migrarlas

**Importante debemos tener la base de datos creada en phpMyAdmin**

1. **python manage.py makemigrations**
2. **python manage.py sqlmigrate compras 0001**
3. **python manage.py migrate**

****

****

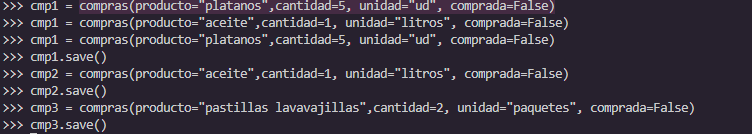
14. Ahora vamos a meter información dentro de la base de datos, para eso:

**python manage.py shell**

14.1 Ahora ya dentro:

**from compras.models import Compras**

**cmp1 = compras(producto="platanos",cantidad=5, unidad="ud", comprada=False)**

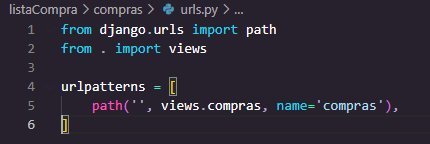


se sale con quit()

15. Vamos a realizar la funcionalidad de nuestra aplicación. Se necesitan varias cosas.

* Definir URL y enlazarla con la vista

Para esto en la carpeta compras tenemos que crear un urls.py

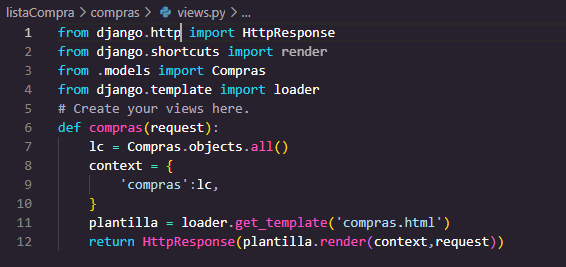


* Crear la vista

Dentro de la vista:

* + Recuperar los datos mediante modelos
  + Pasar los datos al template(html)

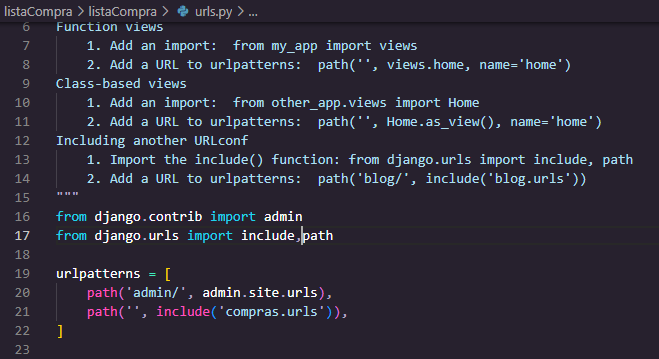
Nos vamos a views.py de la carpeta compras.

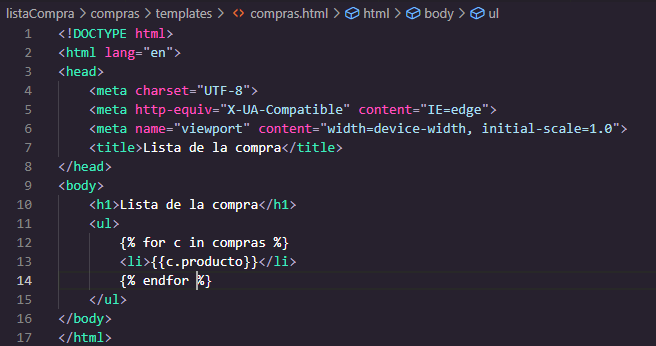


* Crear el template(html)

Creamos una carpeta templates dentro de la carpeta compras. y creamos el archivo compras.html

16. Hay que enlazar todas las urls, para eso, en urls.py de listaCompra, añadimos la url de urls.py de la carpeta compra, y ya ahi nos llevará a compra.html



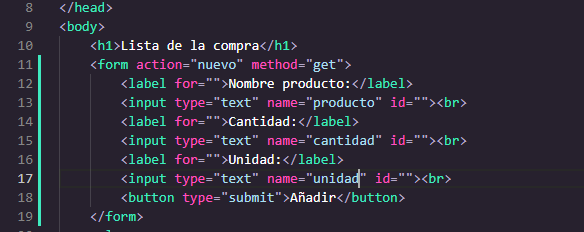


A partir de aquí lo que vamos a hacer es darle funcionalidad a nuestra pagina.

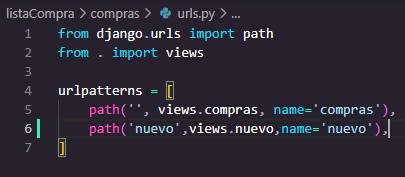
Tendremos:

* La ruta -> para acceder a la funcionalidad
* La vista -> que es el código
* El modelo -> Que es la representación de la información de la BBDD
* Los template -> Como se presenta la información

En este proyecto crearemos un formulario para acceder a la ruta ( para añadir a la lista de la compra ). Modificamos el html para esto.

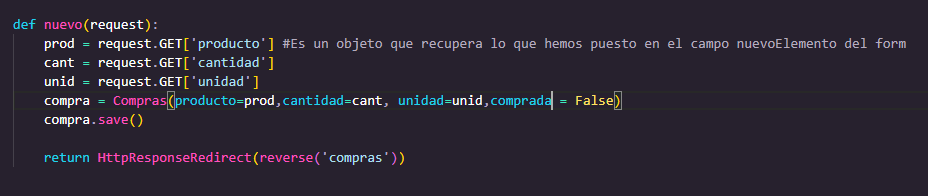


Para que esto nos lleve a algún lado, tenemos que añadir la en view, y esta se tiene que llamar igual que el action que pongas en el formulario. Por lo que en urls.py de la carpeta compras, añadimos nuestra nueva ruta.



Vale, pero con esto no es suficiente, porque al tu internet añadir algo, te va a llevar a una página que obviamente no existe. Por lo que tenemos que hacer es, modificar las views.

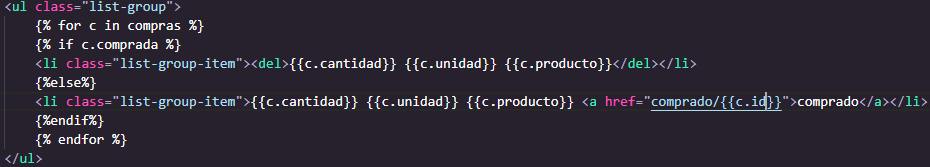
Lo que haremos será recuperar los datos del formulario ( para saber que tenemos que añadir )



Y con esto se añadiría a la base de datos y se modifica la lista.

Mas cosas que hemos hecho, en el html hacer que si estan compradas, salgan tachadas.



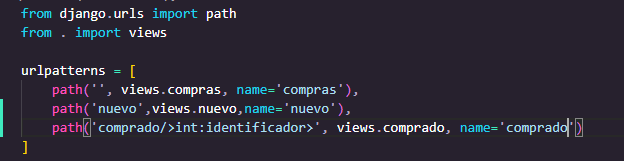


Ahora vamos a añadir un link, para que se modifique si ha sido comprado o no.

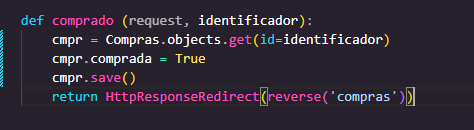
Añadimos un link



modificamos las urls.py para que nos lleve el link a donde queremos

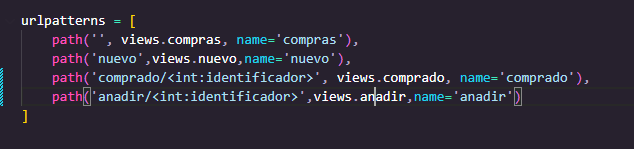


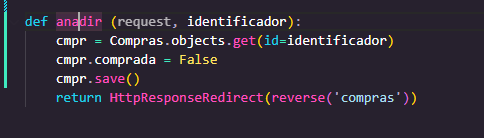
Por ultimo, añadimos a views lo que queremos hacer. En este caso marcar como que ha sido comprado.



Añadir de nuevo un elemento que ha sido tachado

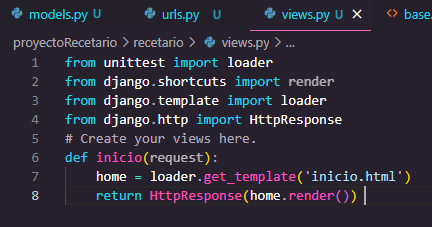




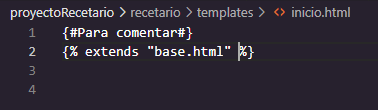


Jerarquía de Templates

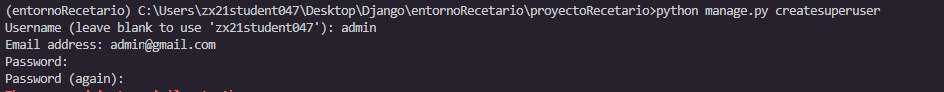
Cuando creamos la carpeta templates dentro de nuestra app, creamos un html base.html. Sera el html base que usaremos para otros htmls.



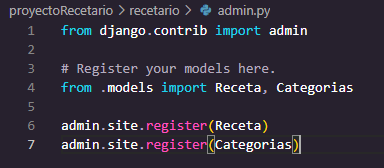
Creamos ahora otro html que heredara de base html. En este ejemplo creamos inicio.html



Crear Admin



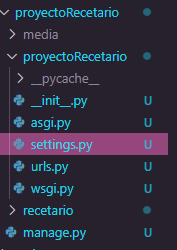
Despues de escribir el nombre de usuario y contraseña en el cmd, tenemos que modificar el archivo admin.py



Ahora desde la url/admin podrás modificar las tablas sin tener que entrar en phpadmin

AÑADIR MEDIA

Crear carpeta media en el entorno



Y añadimos en settings

#Configuracion de Media

MEDIA\_URL='/media/'

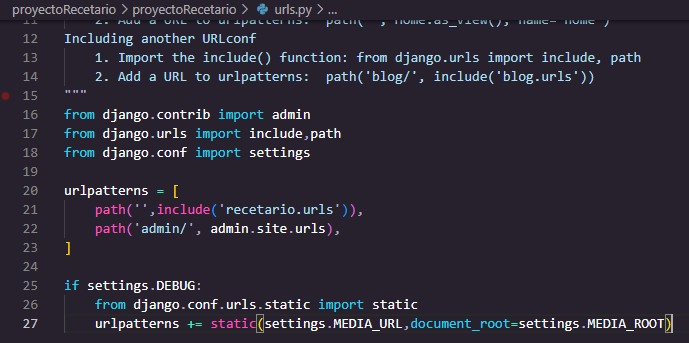
MEDIA\_ROOT=os.path.join(BASE\_DIR,'media')

Y cambiamos en urls.py del proyecto

if settings.DEBUG:

from django.conf.urls.static import static

urlpatterns += static(settings.MEDIA\_URL,document\_root=settings.MEDIA\_ROOT)



Modificamos models.py

Necesitamos instalar para el manejo de imágenes

python -m pip install Pillow

COPIAR UN PROYECTO A OTRO

Crear un nuevo proyecto. En este ejemplo recetarioV2

***python -m venv entornoRecetarioV2***

OJO! Activar el entorno, no seas como yo:

***Scripts\activate.bat***

Lo siguiente es hacer:

***pip install Django mysqlclient pillow***

Crear la base de datos ***recetarioV2***

Nota: Si se te joden las tablas, te vas a mysql (en el lateral izquierdo), seleccionas todas, y eliges reparar tablas.

Creamos el proyecto:

***django-admin startproject proyectoRecetario***

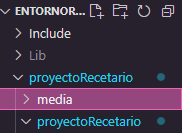
Cambiamos el Setting.py, No se puede copiar el de otro proyecto

linea 76, como se explica en el documento.

linea 110: LANGUAGE\_CODE = 'es'

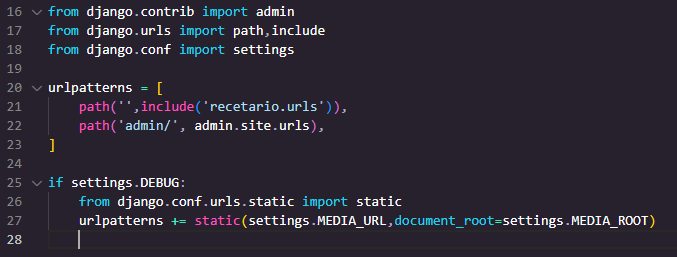
linea 41: 'recetario.apps.ReetarioConfig',

Creamos el directorio media.



Descargas el fichero .zip si la app ya la creó otra persona y descomprimirlo, o si es tuya, tienes que copiar la app, eliminar las migrations antes.

Modificar urls.py global:



Toca hacer las migraciones (Hay que estar dentro del proyecto) y crear superusuario.

***python manage.py makemigrations***

***python manage.py migrate***

***python manage.py createsuperuser***