

UNIVERSIDAD DE EL SALVADOR
FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE
DEPARTAMENTO DE INGENIERÍA Y ARQUITECTURA



PRÁCTICA N° 3

ASIGNATURA:

Técnicas De Programación En Internet

DOCENTE:

Ing. Xenia Ivette Peñate Godoy

INTEGRANTES:

Calderón López, Fátima del Rosario Carnet: CL16006

Ponce Quijada, Jessica Lissette Carnet: PQ16001

PRÁCTICA 3

Creación de una aplicación haciendo uso de un framework

En la actualidad, se cuenta con múltiples herramientas que facilitan el desarrollo web, podemos disponer de gran cantidad de librerías, así como de plugins enmarcados dentro de lo que se denominan Frameworks.

Podemos definir un Framework como un conjunto de módulos que permiten, o tienen por objetivo, el desarrollo ágil de aplicaciones mediante la aportación de librerías y/o funcionalidades ya creadas para que nosotros las usemos directamente.

Por tal razón, un Framework se encarga de facilitar el desarrollo, estandarizar las páginas y son de mucha utilidad para el desarrollo de aplicaciones web más escalables y sencillas de mantener, lo que nos permite también ahorrar tiempo.

Para realizar esta práctica hemos utilizado las siguientes herramientas:

Framework Django:

Django es un framework web de código abierto, escrito en Python, que se ajusta en gran medida al conocido patrón de diseño MVC (Modelo Vista Controlador) el cual permite crear aplicaciones web extensibles, escalables y fáciles de mantener.

Entorno de desarrollo PyCharm:

Es un entorno de desarrollo integrado (IDE) utilizado en programación de computadoras, específicamente para el lenguaje Python. Está desarrollado por la empresa checa JetBrains. Proporciona análisis de código, un depurador gráfico, un probador de unidades integrado, integración con sistemas de control de versiones y admite el desarrollo web con Django.

Lenguaje de programación Python:

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.

PROCEDIMIENTO

Primero debemos instalar Django, para ello tenemos que tener instalado previamente Python 3 en nuestra máquina, para Instalar Django utilizamos el comando:

```
pip install Django==3.2.3
```

para ver la que lo tenemos instalado ejecutamos el comando:

```
py -m django --version
```

creamos el proyecto

Para ello tenemos que ubicarnos en el directorio donde tenemos nuestros proyectos con django y ejecutamos el comando

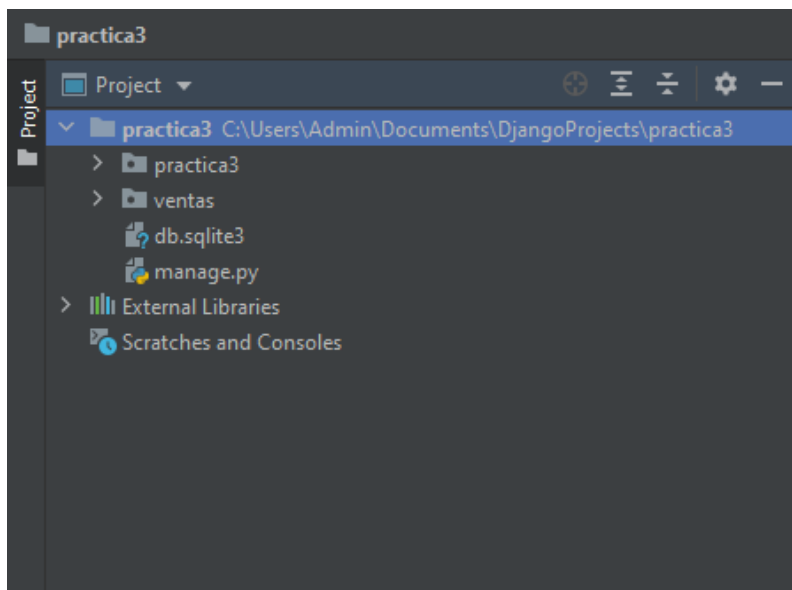
```
py -m django startproject practica3
```

Podemos abrir el proyecto en el IDE en nuestro caso usamos Pycharm.

En pycharm abrimos una terminal y ejecutamos el comando:

```
python manage.py startapp ventas
```

Esto nos crea una aplicación dentro de nuestro proyecto principal, cada aplicación que usted escribe en Django consiste en un paquete de Python que sigue una determinada convención. Django tiene una utilidad que genera automáticamente la estructura básica de directorios de una aplicación, por lo que usted puede centrarse en la escritura de código en lugar de crear directorios.



Agregamos nuestra aplicación al archivo practica3/urls.py

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

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

Vamos a empezar trabajando en el modelo, abrimos el archivo ventas\models.py

Y lo editamos como sigue:

```
from django.db import models

class categoria(models.Model):
    nombre=models.CharField(max_length=200)
    descripcion=models.CharField(max_length=200)

    def __str__(self):
        return self.nombre

class productos(models.Model):
    nombre = models.CharField(max_length=200)
    precio=models.DecimalField(max_digits=10,decimal_places=2)
    categoria=models.ForeignKey(categoria, on_delete=models.CASCADE)

    def __str__(self):
        return '{} ${} {}'.format(self.nombre,self.precio,self.categoria)

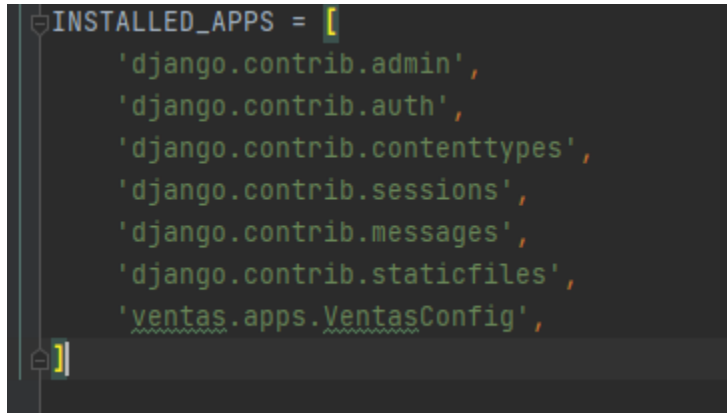
class ventas(models.Model):
    descripcion = models.CharField(max_length=200)
    producto=models.ForeignKey(productos,on_delete=models.CASCADE)
    fecha=models.DateTimeField()

    def __str__(self):
        return self.descripcion
```

Ahora que tenemos el modelo realizamos las migraciones con el comando:

```
python manage.py migrate
```

Para incluir la aplicación en nuestro proyecto necesitamos agregar una referencia a su clase de configuración en la configuración `INSTALLED_APPS`. La clase `VentasConfig` está en el archivo `ventas/apps.py` por lo que su ruta punteada es `'ventas.apps.VentasConfig'`. Edite el archivo `practica3/settings.py` y agregue la ruta punteada a la configuración `INSTALLED_APPS`. Se verá así:



```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'ventas.apps.VentasConfig',  
]
```

Ahora hay que hacer las migraciones con el comando:

```
python manage.py makemigrations ventas
```

Al ejecutar `makemigrations`, usted le indica a Django que ha realizado algunos cambios a sus modelos (en este caso, ha realizado cambios nuevos) y que le gustaría que los guarde como una *migración*.

Ahora creamos un superusuario para el panel de administración con el comando:

```
python manage.py createsuperuser
```

nos pide que ingresemos un nombre, un email, y que creamos una contraseña.

Iniciamos el servidor con el comando:

```
python manage.py runserver
```

Y nos dirigimos a la ruta <http://127.0.0.1:8000/admin/>

Tendremos que ingresar el usuario y contraseña en este caso el usuario es `"fatima"` y la contraseña es `"fatima97"`

Administración de Django

Nombre de usuario:

fatima

Contraseña:

●●●●●●●●

Iniciar sesión

Y ya estamos dentro del panel de administración de la aplicación

Administración de Django

Sitio administrativo

AUTENTICACIÓN Y AUTORIZACIÓN

Grupos

+ Añadir

✎ Modificar

Usuarios

+ Añadir

✎ Modificar

Podemos crear grupos y usuarios

Vamos a editar el archivo ventas/admin.py

```
from django.contrib import admin
from .models import productos, categoria, ventas
# Register your models here.

admin.site.register(productos)
admin.site.register(categoria)
admin.site.register(ventas)
|
```

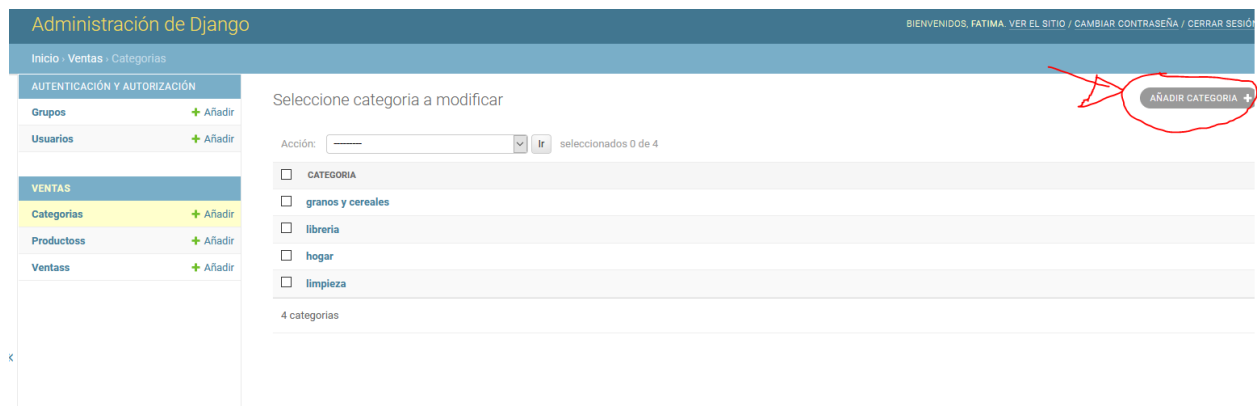
Recargamos el panel de administración y ahora ya podemos hacer las operaciones crud a nuestros modelos:

Administración de Django

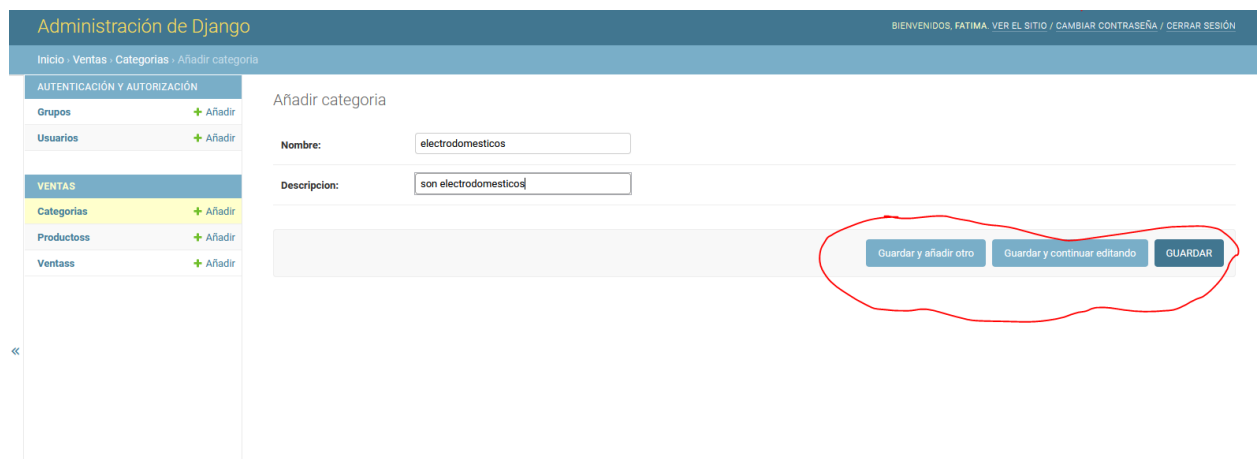
Sitio administrativo

AUTENTICACIÓN Y AUTORIZACIÓN		
Grupos	+ Añadir	 Modificar
Usuarios	+ Añadir	 Modificar
VENTAS		
Categorías	+ Añadir	 Modificar
Productoss	+ Añadir	 Modificar
Ventass	+ Añadir	 Modificar

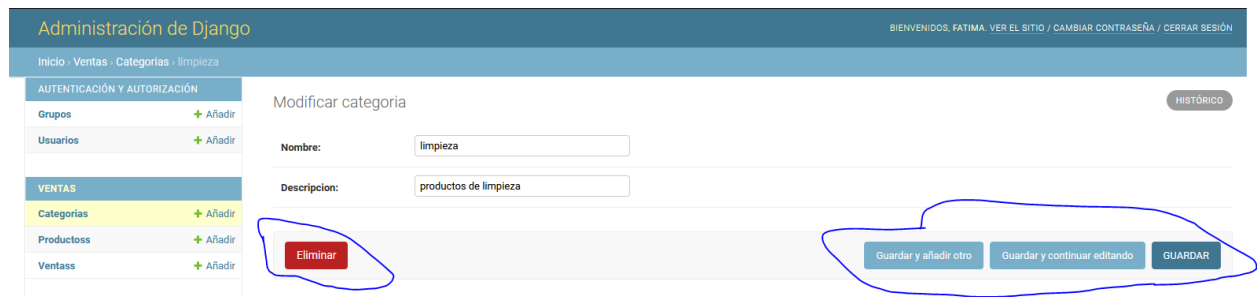
Si damos click en categorías podemos agregar categorías:



Podemos crear una nueva categoría



Podemos modificar o eliminar una categoría



Y así podemos también gestionar productos y ventas:

The screenshot shows the Django Admin interface for adding a product. The left sidebar contains a menu with 'AUTENTICACIÓN Y AUTORIZACIÓN' (Grupos, Usuarios) and 'VENTAS' (Categorías, Productoss, Ventass). The main area is titled 'Añadir productos' and contains form fields for 'Nombre:', 'Precio:', and 'Categoría:'. At the bottom, there are three buttons: 'Guardar y añadir otro', 'Guardar y continuar editando', and 'GUARDAR'. The 'GUARDAR' button is circled in red.

Ahora vamos a crear un usuario para que pueda solo hacer ventas

Sitio administrativo

The screenshot shows the Django Admin interface for user management. The left sidebar contains a menu with 'AUTENTICACIÓN Y AUTORIZACIÓN' (Grupos, Usuarios). The main area is titled 'AUTENTICACIÓN Y AUTORIZACIÓN' and contains a table with two rows: 'Grupos' and 'Usuarios'. Each row has a '+ Añadir' button and a 'Modificar' button. The '+ Añadir' button for 'Usuarios' is circled in red.

Añadir usuario

Primero, ingrese un nombre de usuario y contraseña. Luego, podrá editar más opciones del usuario.

The screenshot shows the Django Admin interface for adding a user. The form has three sections: 'Nombre de usuario:', 'Contraseña:', and 'Contraseña (confirmación):'. The 'Nombre de usuario' field contains 'vendedor1'. The 'Contraseña' field is masked with dots. The 'Contraseña (confirmación)' field is also masked with dots. At the bottom, there are three buttons: 'Guardar y añadir otro', 'Guardar y continuar editando', and 'GUARDAR'. The 'GUARDAR' button is circled in red.

Vamos a darle permisos a nuestro usuario

Seleccione usuario a modificar

Acción: seleccionados 0 de 2

<input type="checkbox"/>	NOMBRE DE USUARIO	DIRECCIÓN DE CORREO ELECTRÓNICO	NOMBRE	APELLIDOS	ES STAFF
<input type="checkbox"/>	fatima	fatima@gmail.com			✓
<input type="checkbox"/>	vendedor1	perez1@gmail.com	Juan	Perez	✓

2 usuarios

Debemos marcar las siguientes casillas

Permisos

☒ Activo
Indica si el usuario debe ser tratado como activo. Desmarque esta opción en lugar de borrar la cuenta.

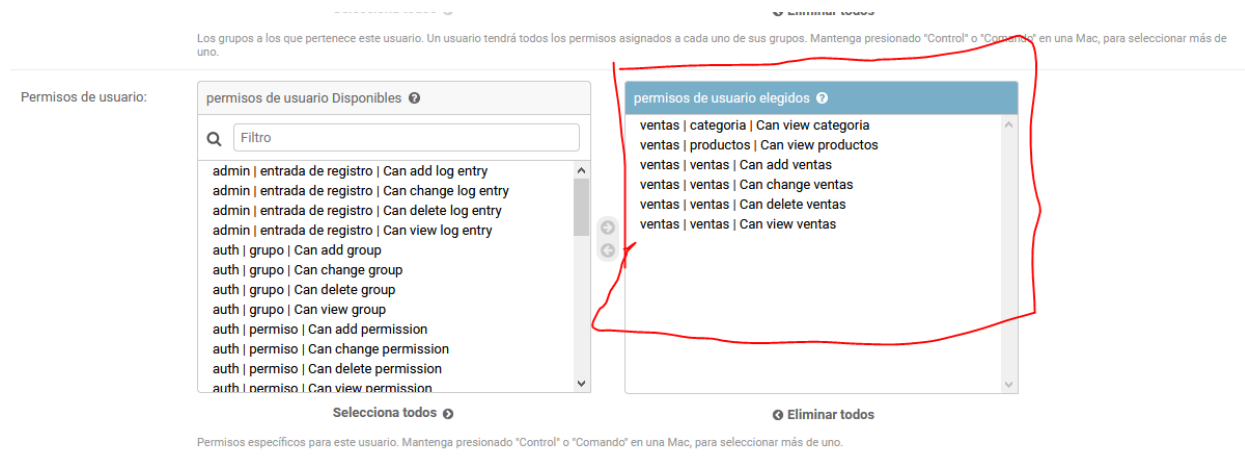
☒ Es staff
Indica si el usuario puede entrar en este sitio de administración.

☐ Estado de superusuario
Indica que este usuario tiene todos los permisos sin asignárselos explícitamente.

Grupos:

grupos Disponibles ?

Y dar los siguientes permisos



Guardamos todos los cambios.

Ahora cerramos sesión y vamos a iniciar sesión con el usuario **"vendedor1"** y con la contraseña **"13052021v1"**.

Administración de Django

Sitio administrativo

VENTAS

Categorías

Productos

Ventass

Vista

Vista

+ Añadir

✎ Modificar

Acciones recientes

Mis acciones

+ venta de detergente

Ventas

+ venta de borrador de goma

Ventas

+ venta de jabón 2

Ventas

Ahora nuestro usuario "vendedor1" puede hacer ventas, pero solo podrá ver las categorías y los productos.

Inicio › Ventas › Categorías

VENTAS

Categorías

Productoss

Ventass

+ Añadir

Selecione categoria para ver

CATEGORIA

electrodomesticos

granos y cereales

libreria

hogar

limpieza

5 categorias

Inicio › Ventas › Productoss

VENTAS

Categorías

Productoss

Ventass

+ Añadir

Selecione productos para ver

PRODUCTOS

detergente \$1.00 limpieza

borrador \$0.30 libreria

arroz \$0.50 granos y cereales

maiz \$7.00 granos y cereales

jabon \$10.00 limpieza

5 productoss

Inicio · Ventas · Ventass

VENTAS

Categorías

Productoss

Ventass + Añadir

Seleccione ventas a modificar

AÑADIR VENTAS +

Acción: Ir seleccionados 0 de 4

☐ VENTAS

☐ venta de detergente

☐ venta de borrador de goma

☐ venta de jabón 2

☐ venta de jabon

4 ventass

Hacemos una venta

Añadir ventas

Descripción:

venta de maiz

Producto:

maiz \$7.00 granos y cereales

Fecha:

Fecha: 24/05/2021 Hoy

Hora: 22:16:42 Ahora

Nota: Usted va 6 horas por detrás de la hora del servidor.

Guardar y añadir otro

Guardar y continuar editando

GUARDAR

Al darle en guardar nos aparecerá un mensaje que la venta fue hecha y aparecerá en la lista:

✓ El ventas "venta de maiz" fue agregado correctamente.

Seleccione ventas a modificar

Acción: Ir seleccionados 0 de 5

☐ VENTAS

☐ venta de maiz

☐ venta de detergente

☐ venta de borrador de goma

☐ venta de jabón 2

☐ venta de jabon

5 ventass

Enlace del repositorio de GitHub:

<https://github.com/CL16006/practica3-unidad3-tpi>