

Primera parte

Proyecto: **Caffenio**

Lenguaje: **Python**.

Framework: **Django**.

Editor: **VS code**.

1 Procedimiento para crear carpeta del Proyecto: **Ulll_Caffenio_0647**

2 procedimiento para abrir vs code sobre la carpeta

Ulll_Caffenio_0647

3 procedimiento para **abrir terminal en vs code**

4 Procedimiento para crear carpeta entorno virtual **".venv"** desde terminal de vs code

5 Procedimiento para **activar el entorno virtual**.

6 procedimiento para **activar intérprete de python**.

7 Procedimiento para instalar **Django**

8 procedimiento para crear proyecto **backend_Caffenio** sin duplicar carpeta.

9 procedimiento para ejecutar servidor en el **puerto 8030**

10 procedimiento para copiar y pegar el link en el navegador.

11 procedimiento para crear aplicacion **app_Caffenio**

12 Aqui el modelo models.py

==-----

```
from django.db import models
```

```
# =====
```

```
# TABLA: PROVEEDORES (7 CAMPOS)
```

```
# =====
```

```
class Proveedor(models.Model):
```

```
    nombre = models.CharField(max_length=100)
```

```
    empresa = models.CharField(max_length=100)
```

```
    telefono = models.CharField(max_length=20)
```

```
    correo = models.EmailField()
```

```
    direccion = models.CharField(max_length=200)
```

```
    pais = models.CharField(max_length=50)
```

```
    tipo_producto = models.CharField(
```

```
        max_length=50,
```

```
        choices=[
```

```
            ('bebidas', 'Bebidas'),
```

```
            ('alimentos', 'Alimentos'),
```

```
            ('complementos', 'Complementos')
```

```
        ]
```

```
    )
```

```
    def __str__(self):
```

```
        return f"{self.nombre} - {self.empresa}"
```

```
# =====
```

```
# TABLA: PRODUCTO (7 CAMPOS)
```

```
# =====
```

```

class Producto(models.Model):
    nombre = models.CharField(max_length=100)
    descripcion = models.TextField()
    precio = models.DecimalField(max_digits=10, decimal_places=2)
    stock = models.IntegerField()
    categoria = models.CharField(
        max_length=50,
        choices=[
            ('bebidas', 'Bebidas'),
            ('alimentos', 'Alimentos'),
            ('complementos', 'Complementos'),
            ('otro', 'Otro')
        ]
    )
    proveedor = models.ForeignKey(
        Proveedor,
        on_delete=models.CASCADE,
        related_name='productos'
    )
    sucursal = models.CharField(
        max_length=50,
        choices=[
            ('Sucursal 1', 'Sucursal 1'),
            ('Sucursal 2', 'Sucursal 2')
        ]
    )

    def __str__(self):
        return self.nombre

```

```

# =====
# TABLA: VENTAS (7 CAMPOS)
# =====

```

```

class Venta(models.Model):
    fecha = models.DateTimeField(auto_now_add=True)
    cliente = models.CharField(max_length=100)
    total = models.DecimalField(max_digits=10, decimal_places=2)
    metodo_pago = models.CharField(
        max_length=50,
        choices=[
            ('efectivo', 'Efectivo'),
            ('tarjeta', 'Tarjeta'),
            ('transferencia', 'Transferencia')
        ]
    )
    sucursal = models.CharField(
        max_length=50,

```

```

        choices=[
            ('Sucursal 1', 'Sucursal 1'),
            ('Sucursal 2', 'Sucursal 2')
        ]
    )
    empleado = models.CharField(max_length=100)
    productos = models.ManyToManyField(
        Producto,
        related_name='ventas'
    )

    def __str__(self):
        return f"Venta {self.id} - {self.cliente}"

```

==--==--==--

12.5 Procedimiento para realizar las migraciones(makemigrations y migrate.

13 primero trabajamos con el MODELO: PROVEEDOR

14 En view de **app_Caffenio** crear las funciones con sus códigos correspondientes (inicio_caffenio, agregar_proveedor,actualizar_proveedor, realizar_actualizacion_proveedor, borrar_proveedor)

15 Crear la carpeta “**templates**” dentro de “**app_Caffenio**”.

16 En la carpeta templates crear los archivos html (**base.html**, **header.html**, **navbar.html**, **footer.html**, **inicio.html**).

17 En el archivo base.html **agregar bootstrap** para css y js.

18 En el archivo navbar.html incluir las opciones (“Sistema de Administración Caffenio”, “Inicio”, “proveedor”,en submenu de proveedores (Agregar Proveedor, ver proveedor, actualizar proveedor, borrar proveedor), “Productos” en submenu de Productos(Agregar productos, ver productos, actualizar productos, borrar productos) “Ventas” en submenu de Ventas(Agregar ventas,ver ventas, actualizar ventas, borrar ventas), incluir iconos a las opciones principales, no en los submenu.

19 En el archivo **footer.html** incluir derechos de autor,fecha del sistema y “Creado por Andrea Montoya, Cbtis 128” y mantenerla fija al final de la página.

20 En el archivo inicio.html se usa para colocar información del sistema más una imagen tomada desde la red sobre cinopolis.

21 Crear la subcarpeta carpeta **proveedor** dentro de **app_Caffenio\templates**.

22 crear los archivos html con su código correspondientes de (agregar_categoria.html, ver_categorias.html mostrar en tabla con los botones ver, editar y borrar, actualizar_categoria.html, borrar_categoria.html) dentro de **app_Caffenio\templates\categoria**.

23 No utilizar forms.py.

24 procedimiento para crear el archivo urls.py en **app_Caffenio** con el código correspondiente para acceder a las funciones de views.py para operaciones de crud en proveedores.

25 procedimiento para agregar **app_Caffenio** en settings.py de **backend_Caffenio**

26 realizar las configuraciones correspondiente a urls.py de **backend_Caffenio** para enlazar con **app_Caffenio**

27 procedimiento para registrar los modelos en admin.py y volver a realizar las migraciones.

27 por lo pronto solo trabajar con “Proveedor” dejar pendiente # MODELO: PRODUCTO y # MODELO: VENTAS

28 Utilizar colores suaves, atractivos y modernos, el código de las páginas web sencillas.

28 No validar entrada de datos.

29 Al inicio crear la estructura completa de carpetas y archivos.

30 proyecto totalmente funcional.

31 finalmente ejecutar servidor en el puerto puerto 8030.