

Primera parte

Proyecto: Dulceria.

lenguaje: Python.

Framework: Django.

Editor: VS code.

1 Procedimiento para crear carpeta del Proyecto: UIII_Dulceria_0726

2 procedimiento para abrir vs code sobre la carpeta

UIII_Dulceria_0726

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_Dulceria sin duplicar carpeta.

9 procedimiento para ejecutar servidor en el puerto 0726

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

11 procedimiento para crear aplicacion app_Dulceria

12 Aqui el modelo models.py

==--==--==

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

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

```
# MODELO: Dulces
```

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

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

```
    nombre = models.CharField(max_length=150, help_text="Nombre del dulce")
```

```
    descripcion = models.TextField(blank=True, null=True, help_text="Descripción detallada del dulce")
```

```
    precio = models.DecimalField(max_digits=10, decimal_places=2, help_text="Precio del dulce")
```

```
    stock = models.PositiveIntegerField(help_text="Cantidad de unidades en stock")
```

```
    categoria = models.CharField(max_length=50, help_text="Categoría del dulce (ej. Chocolates, Gomas, Caramelos)")
```

```
    fecha_disponibilidad = models.DateField(help_text="Fecha a partir de la cual el dulce está disponible")
```

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

```
        return self.nombre
```

```
==--==--==
```

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

```
# MODELO: CLIENTES
```

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

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

```
    nombre = models.CharField(max_length=100, unique=True)
```

```
    apellido = models.CharField(max_length=100, blank=True, null=True)
```

```
    email = models.EmailField(max_length=254, unique=True)
```

```
    telefono = models.CharField(max_length=20, blank=True, null=True)
```

```
    direccion = models.CharField(max_length=255, blank=True, null=True)
```

```
    fecha_registro = models.DateTimeField(auto_now_add=True)
```

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

```

        return f"{self.nombre} {self.apellido}"
# =====
# MODELO: PEDIDOS
# =====

class Pedido(models.Model):
    cliente_id = models.ForeignKey(Cliente, on_delete=models.CASCADE, related_name='pedidos')
    direccion_envio = models.CharField(max_length=255)
    TIPO_PEDIDO_CHOICES = [
        ('estandar', 'Estándar'),
        ('urgente', 'Urgente'),
        ('personalizado', 'Personalizado'),
    ]
    tipo_pedido = models.CharField(max_length=20, choices=TIPO_PEDIDO_CHOICES,
default='estandar')
    fecha_pedido = models.DateTimeField(auto_now_add=True)
    ESTADO_CHOICES = [
        ('pendiente', 'Pendiente'),
        ('procesando', 'Procesando'),
        ('enviado', 'Enviado'),
        ('entregado', 'Entregado'),
        ('cancelado', 'Cancelado'),
    ]
    estado = models.CharField(max_length=20, choices=ESTADO_CHOICES, default='pendiente')

    METODO_PAGO_CHOICES = [
        ('tarjeta_credito', 'Tarjeta de Crédito'),
        ('paypal', 'PayPal'),
        ('transferencia', 'Transferencia Bancaria'),
        ('efectivo', 'Efectivo al Recibir'),
    ]
    metodo_pago = models.CharField(max_length=50, choices=METODO_PAGO_CHOICES)
    id_dulce = models.ManyToManyField(Dulce, related_name="pedidos")
    def __str__(self):
        return f"Pedido #{self.pk} de {self.cliente.nombre} - Estado: {self.estado}"

```

12.5 Procedimiento para realizar las migraciones(makemigrations y migrate.

13 primero trabajamos con el MODELO: DULCES

14 En view de app_dulceria crear las funciones con sus códigos correspondientes (inicio_dulceria, agregar_dulce,

actualizar_dulce, realizar_actualizacion_dulce, borrar_dulce)

15 Crear la carpeta “templates” dentro de “app_Dulceria”.

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 Dulceria”, “Inicio”, “dulce”,en submenu de dulces(Aregar Dulce,ver dulce, actualizar dulce,

borrar dulce), “Clientes” en submenu de Clientes(Agregar clientes,ver clientes, actualizar clientes, borrar clientes)

“Pedidos” en submenu de Pedidos(Agregar pedidos,ver pedidos,, actualiza pedidos,, borrar pedidos,), 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 Dominique Vaquera, 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 dulce dentro de app_Dulceria\templates.

22 crear los archivos html con su codigo correspondientes de (agregar_dulce.html, ver_dulces.html mostrar en tabla con los botones ver, editar y borrar, actualizar_dulce.html, borrar_dulce.html) dentro de app_Dulceria\templates\dulce.

23 No utilizar forms.py.

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

25 procedimiento para agregar app_Dulceria en settings.py de backend_Dulceria

26 realizar las configuraciones correspondiente a urls.py de backend_Dulceria para enlazar con app_Dulceria

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

27 por lo pronto solo trabajar con “dulce” dejar pendiente # MODELO: CLIENTES y # MODELO: PEDIDOS

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 0726