

Primera parte

Proyecto: **Empresa de Eventos**

lenguaje: **Python**.

Framework: **Django**.

Editor: **VS code**.

1 Procedimiento para crear carpeta del Proyecto: **UIII_Eventos_0301**

2 procedimiento para abrir vs code sobre la carpeta

UIII_Eventos_0301

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_eventos** sin duplicar carpeta.

9 procedimiento para ejecutar servidor **en el puerto 8026**

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

11 procedimiento para crear aplicacion **app_eventos**

12 Aqui el modelo **models.py**

==--==--==

from django.db import models

=====

MODELO: SERVICIO

=====

```
class Servicio(models.Model):
    nombre = models.CharField(max_length=100)
    descripcion = models.TextField()
    costo = models.DecimalField(max_digits=10, decimal_places=2)
    duracion_horas = models.PositiveIntegerField()
    tipo = models.CharField(max_length=50)
    disponible = models.BooleanField(default=True)
    fecha_creacion = models.DateTimeField(auto_now_add=True)

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

=====

MODELO: EVENTO

=====

```
class Evento(models.Model):
    nombre = models.CharField(max_length=150)
    descripcion = models.TextField()
    fecha_inicio = models.DateField()
    fecha_fin = models.DateField()
```

```

ubicacion = models.CharField(max_length=200)
capacidad = models.PositiveIntegerField()
precio_base = models.DecimalField(max_digits=10, decimal_places=2)

# Relación muchos a muchos con Servicio
servicios = models.ManyToManyField(Servicio,
related_name='eventos')

def __str__(self):
    return self.nombre

```

=====

MODELO: EMPLEADO

=====

```

class Empleado(models.Model):
    nombre = models.CharField(max_length=100)
    apellido = models.CharField(max_length=100)
    puesto = models.CharField(max_length=100)
    salario = models.DecimalField(max_digits=10, decimal_places=2)
    telefono = models.CharField(max_length=20)
    email = models.EmailField(unique=True)
    fecha_contratacion = models.DateField()

    # Relación uno a muchos: un evento tiene varios empleados
    evento = models.ForeignKey(Evento, on_delete=models.CASCADE,
related_name='empleados')

    def __str__(self):
        return f"{self.nombre} {self.apellido}"

```

=====

12.5 Procedimiento para realizar las migraciones(makemigrations y migrate.

13 primero trabajamos con el **MODELO: Servicio**

14 En view de **app_eventos** crear las funciones con sus códigos correspondientes (**inicio_evento**, **agregar_evento**, **actualizar_evento**, **realizar_actualizacion_evento**, **borrar_evento**)

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

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 Eventos”, “Inicio”, “Servicio”, en submenu de

servicios(Agregar servicio,ver servicio, actualizar servicio, borrar servicio), “servicio” en submenu de servicio (Agregar servicio,ver servicio, actualizar servicio, borrar servicio)
“Empleado” en submenu de Empleados (Agregar empleados,ver empleados, actualizar empleados, borrar empleados), 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 Paola Ortega, 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 empresa de eventos.

21 Crear la subcarpeta carpeta servicio dentro de app_eventos\templates.

22 crear los archivos html con su código correspondientes de (agregar_servicio.html, ver_servicio.html mostrar en tabla con los botones ver, editar y borrar, actualizar_servicio.html, borrar_servicio.html) dentro de app_eventos\templates\servicio.

23 No utilizar forms.py.

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

25 procedimiento para agregar app_eventos en settings.py de backend_eventos

26 realizar las configuraciones correspondiente a urls.py de backend_eventos para enlazar con app_evento

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

27 por lo pronto solo trabajar con “servicio” dejar pendiente #

MODELO: EVENTO y # MODELO: EMPLEADO

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 8026.