Primera parte
Proyecto: **Taller Mecanico.**
lenguaje: **Python.**
Framework: **Django.**
Editor: **VS code.**
1 Procedimiento para crear carpeta del Proyecto: **UIII_Taller_Mecanico_0446**
2 procedimiento para abrir vs code sobre la carpeta
UIII_Taller_Mecanico_0446
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_Taller sin duplicar
carpeta.
9 procedimiento para ejecutar servidor en el puerto 8446
10 procedimiento para copiar y pegar el link en el navegador.
11 procedimiento para crear aplicacion app_Taller
12 Aqui el modelo models.py
=-=-=-=-=-

```
from django.db import models

# ========================================
# MODELO: CLIENTE
# ========================================
class Cliente(models.Model):
    nombre = models.CharField(max_length=100)
    apellido = models.CharField(max_length=100)
    telefono = models.CharField(max_length=20)
    email = models.EmailField(unique=True)
    rfc = models.CharField(max_length=20, unique=True)
    direccion = models.CharField(max_length=200)
    fecha_registro = models.DateTimeField(auto_now_add=True)

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

# ========================================
# MODELO: SERVICIO
# ========================================
class Servicio(models.Model):
    nombre_servicio = models.CharField(max_length=100)
    descripcion = models.TextField(blank=True, null=True)
    precio_base = models.DecimalField(max_digits=10, decimal_places=2)
    tiempo_estimado_horas = models.DecimalField(max_digits=5, decimal_places=2)
    aplica_garantia = models.BooleanField(default=False)
    fecha_actualizacion = models.DateTimeField(auto_now=True)
```

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

# ========================================
# MODELO: VEHÍCULO
# ========================================
class Vehiculo(models.Model):
    cliente = models.ForeignKey(
        Cliente, on_delete=models.CASCADE, related_name="vehiculos"
    )
    servicios = models.ManyToManyField(
        Servicio, related_name="vehiculos", blank=True
    )
    matricula = models.CharField(max_length=20, unique=True)
    marca = models.CharField(max_length=50)
    modelo = models.CharField(max_length=50)
    anio = models.PositiveIntegerField()
    kilometraje = models.PositiveIntegerField()
    color = models.CharField(max_length=30)

    def __str__(self):
        return f"{self.marca} {self.modelo} ({self.matricula})"
```

=-=-=-=-=-

12.5 Procedimiento para realizar las migraciones(makemigrations y migrate.

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

14 En view de app_Taller crear las funciones con sus códigos correspondientes (inicio_taller, agregar_cliente, actualizar_cliente, realizar_actualizacion_cliente, borrar_cliente)

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

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 Taller Mecanico”, “Inicio”, “cliente”,en submenu de cliente(Agregar cliente,ver cliente, actualizar cliente, borrar cliente), “Vehiculo” en submenu de Vehiculo(Agregar vehiculo,ver vehiculo, actualizar vehiculo, borrar vehiculo)
“Servicios” en submenu de Servicios(Agregar servicios,ver servicios, actualizar servicios, borrar servicios), 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 Alumno Osvaldo Arellano De La Cruz, 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 cinepolis.
21 Crear la subcarpeta carpeta cliente dentro de
app_Taller\templates.
22 crear los archivos html con su codigo correspondientes de
(agregar_cliente.html, ver_cliente.html mostrar en tabla con
los botones ver, editar y borrar, actualizar_cliente.html,
borrar_cliente.html)
dentro de app_Taller\templates\cliente.
23 No utilizar forms.py.
24 procedimiento para crear el archivo urls.py en app_Taller con
el código correspondiente para acceder a las funciones de views.py
para operaciones de crud en clientes.
25 procedimiento para agregar app_Taller en settings.py de
backend_Taller
26 realizar las configuraciones correspondiente a urls.py de
backend_Taller para enlazar con app_Taller

27 procedimiento para registrar los modelos en admin.py y volver a
realizar las migraciones.
27 por lo pronto solo trabajar con “cliente” dejar pendiente #
MODELO: SERVICIO y # MODELO: VEHICULO
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 8446.