

# Prompt Completo - Proyecto Libreria AJMG 1194

Primera parte

Proyecto: Libreria AJMG

Lenguaje: Python

Framework: Django

Editor: VS Code

1. Procedimiento para crear carpeta del Proyecto: UIII\_Libreria\_1194
2. Procedimiento para abrir VS Code sobre la carpeta UIII\_Libreria\_1194
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\_Libreria sin duplicar carpeta
9. Procedimiento para ejecutar servidor en el puerto 1194
10. Procedimiento para copiar y pegar el link en el navegador
11. Procedimiento para crear aplicación app\_Libreria
12. Aquí el modelo models.py

python

```
from django.db import models
from django.contrib.auth.models import User
from django.core.validators import MinValueValidator, MaxValueValidator
from django.utils import timezone

class Autor(models.Model):
    nombre = models.CharField(max_length=100, verbose_name="Nombre completo")
    nacionalidad = models.CharField(max_length=50, verbose_name="Nacionalidad")
    fecha_nacimiento = models.DateField(verbose_name="Fecha de nacimiento")
    fecha_fallecimiento = models.DateField(null=True, blank=True, verbose_name="Fecha de fallecimiento")
    biografia = models.TextField(verbose_name="Biografía", help_text="Breve biografía del autor")
    email = models.EmailField(verbose_name="Email de contacto", blank=True)
    activo = models.BooleanField(default=True, verbose_name="Autor activo")

    class Meta:
```

```

        verbose_name = "Autor"
        verbose_name_plural = "Autores"
        ordering = ['nombre']

    def __str__(self):
        return self.nombre

class Libro(models.Model):
    GENEROS = [
        ('FIC', 'Ficción'),
        ('ROM', 'Romance'),
        ('TER', 'Terror'),
        ('CIE', 'Ciencia Ficción'),
        ('FAN', 'Fantasía'),
        ('HIS', 'Histórico'),
        ('BIO', 'Biografía'),
        ('INF', 'Infantil'),
    ]

    titulo = models.CharField(max_length=200, verbose_name="Título del libro")
    isbn = models.CharField(max_length=13, unique=True, verbose_name="ISBN")
    genero = models.CharField(max_length=3, choices=GENEROS, verbose_name="Género")
    fecha_publicacion = models.DateField(verbose_name="Fecha de publicación")
    precio = models.DecimalField(max_digits=8, decimal_places=2, verbose_name="Precio")
    stock = models.PositiveIntegerField(default=0, verbose_name="Cantidad en stock")
    descripcion = models.TextField(verbose_name="Descripción", blank=True)

    autores = models.ManyToManyField(Autor, related_name='libros', verbose_name="Autores")

    class Meta:
        verbose_name = "Libro"
        verbose_name_plural = "Libros"
        ordering = ['titulo']

    def __str__(self):
        return self.titulo

    def disponible(self):
        return self.stock > 0

```

```

class Venta(models.Model):
    ESTADOS = [
        ('PEN', 'Pendiente'),
        ('COM', 'Completada'),
        ('CAN', 'Cancelada'),
        ('DEV', 'Devuelta'),
    ]

    fecha_venta = models.DateTimeField(default=timezone.now,
    verbose_name="Fecha de venta")
    cantidad = models.PositiveIntegerField(verbose_name="Cantidad vendida")
    precio_unitario = models.DecimalField(max_digits=8, decimal_places=2,
    verbose_name="Precio unitario")
    total = models.DecimalField(max_digits=10, decimal_places=2,
    verbose_name="Total de la venta")
    estado = models.CharField(max_length=3, choices=ESTADOS, default='PEN',
    verbose_name="Estado de la venta")
    cliente_nombre = models.CharField(max_length=100, verbose_name="Nombre
del cliente")
    cliente_email = models.EmailField(verbose_name="Email del cliente")

    libro = models.ForeignKey(
        Libro,
        on_delete=models.PROTECT,
        related_name='ventas',
        verbose_name="Libro vendido"
    )

class Meta:
    verbose_name = "Venta"
    verbose_name_plural = "Ventas"
    ordering = ['-fecha_venta']

    def __str__(self):
        return f"Venta #{self.id} - {self.libro.titulo}"

    def save(self, *args, **kwargs):
        self.total = self.cantidad * self.precio_unitario

        super().save(*args, **kwargs)

```

## 12.5 Procedimiento para realizar las migraciones (makemigrations y migrate)

### 13. Primero trabajamos con el MODELO: AUTOR

14. En views de app\_Libreria crear las funciones con sus códigos correspondientes (inicio\_libreria, agregar\_autor, actualizar\_autor, realizar\_actualizacion\_autor, borrar\_autor)
15. Crear la carpeta "templates" dentro de "app\_Libreria"
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 Libreria AJMG 1194"
  - "Inicio"
  - "Autores" en submenu (Agregar Autor, Ver Autores, Actualizar Autor, Borrar Autor)
  - "Libros" en submenu (Agregar Libro, Ver Libros, Actualizar Libro, Borrar Libro)
  - "Ventas" en submenu (Agregar Venta, Ver Ventas, Actualizar Venta, Borrar Venta)
  - 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 Alfredo Martinez" y mantenerla fija al final de la página
20. En el archivo inicio.html usar para colocar información del sistema más una imagen tomada desde la red sobre librerías
21. Crear la subcarpeta "autor" dentro de app\_Libreria\templates
22. Crear los archivos html con su código correspondiente de (agregar\_autor.html, ver\_autores.html mostrar en tabla con los botones ver, editar y borrar, actualizar\_autor.html, borrar\_autor.html) dentro de app\_Libreria\templates\autor
23. No utilizar forms.py
24. Procedimiento para crear el archivo urls.py en app\_Libreria con el código correspondiente para acceder a las funciones de views.py para operaciones de crud en autores
25. Procedimiento para agregar app\_Libreria en settings.py de backend\_Libreria
26. Realizar las configuraciones correspondientes a urls.py de backend\_Libreria para enlazar con app\_Libreria
27. Procedimiento para registrar los modelos en admin.py y volver a realizar las migraciones
28. Por lo pronto solo trabajar con "Autor" dejar pendiente MODELO: LIBRO y MODELO: VENTA
29. Utilizar colores suaves, atractivos y modernos, el código de las páginas web sencillas
30. No validar entrada de datos
31. Al inicio crear la estructura completa de carpetas y archivos
32. Proyecto totalmente funcional

33. Finalmente ejecutar servidor en el puerto 1194