
# Modelos en Django – AE3

**Aprendiendo a estructurar los datos de una aplicación web**

**Objetivo General: Comprender qué son los modelos en Django y cómo definirlos, incluso sin conexión a base de datos.**

# Objetivos de la Clase

**Entender qué son los modelos en Django**

Comprender su propósito fundamental en el desarrollo web

**Definir modelos sin base de datos**

Aprender a crear estructuras de datos incluso sin conexión a BD

**Explorar la integración con vistas y plantillas**

Descubrir cómo los modelos trabajan con otros componentes

**Crear un mini-proyecto práctico**

Desarrollar una aplicación de productos paso a paso

# ¿Qué son los modelos en Django?


**Clases de Python**

Representan tablas de la base de datos de forma elegante y estructurada

**Mapeo Directo**

Cada atributo de la clase se convierte en una columna en la tabla

**Lógica de Negocio**

Son la base fundamental de toda la aplicación web

# Ejemplo Básico de Modelo

```
from django.db import models

class Persona(models.Model):
    nombre = models.CharField(max_length=100)
    edad = models.IntegerField()
    email = models.EmailField()

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

Este modelo se traduce automáticamente en una tabla con las columnas: nombre, edad y email.

# Metodología Activa - Actividad 1

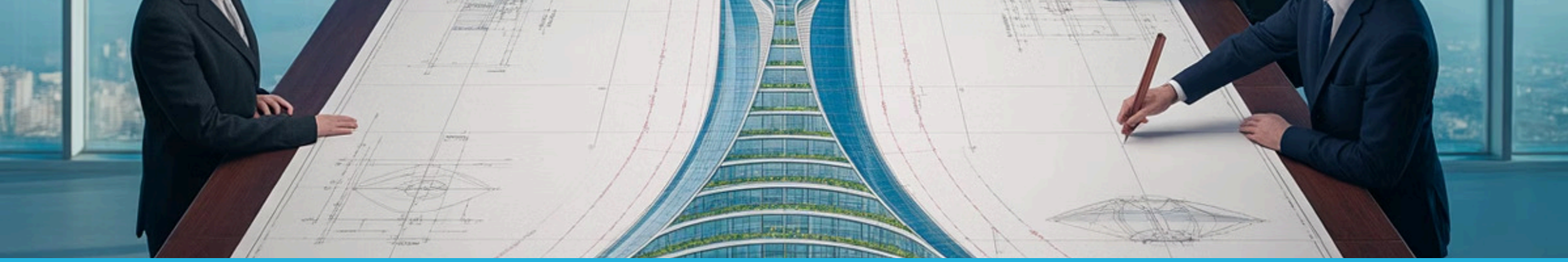
1

**Ejercicio Grupal en VS Code**

**Cada estudiante definirá un modelo sencillo de su interés personal**

- **Ejemplos: *Libro*, *Curso*, *Mascota***
- **Mínimo 3 campos obligatorios**
- **Usar CharField, IntegerField, BooleanField**

**Tiempo estimado: 10 minutos para definir y 5 minutos para compartir resultados**



# Definir Modelos sin Base de Datos

### Planificación Previa

Los modelos se pueden definir aunque aún no tengas BD configurada, permitiendo planificar la estructura antes de implementar

### Diseño de Estructura

Sirve para organizar ideas y definir la arquitectura de datos antes de crear migraciones

### Pruebas de Concepto

No se podrán guardar datos permanentemente, pero sí probar la estructura y lógica


# Ejemplo Práctico sin BD

```
from django.db import models

class Producto(models.Model):
    nombre = models.CharField(max_length=100)
    precio = models.DecimalField(max_digits=10, decimal_places=2)
    disponible = models.BooleanField(default=True)

    def __str__(self):
        return f"{self.nombre} - ${self.precio}"
```

**Este modelo define la estructura perfecta para manejar productos, incluso sin tener una base de datos real conectada.**

# ¿Qué pasa si no hay BD?


**Limitaciones**

- No hay tablas reales creadas
- No se pueden hacer migraciones
- No se guardan datos de forma persistente

**Posibilidades**

- Documentar modelos
- Simular datos temporales
- Probar vistas y plantillas

# Metodología Activa – Actividad 2

**Trabajo en Parejas**

**Definir un modelo Vehiculo con los siguientes campos:**

- **Marca (CharField)**
- **Modelo (CharField)**
- **Año (IntegerField)**
- **Disponible (BooleanField)**

**Luego compartir en clase y discutir las diferentes implementaciones**


# Conexión con Vistas

**Cliente**

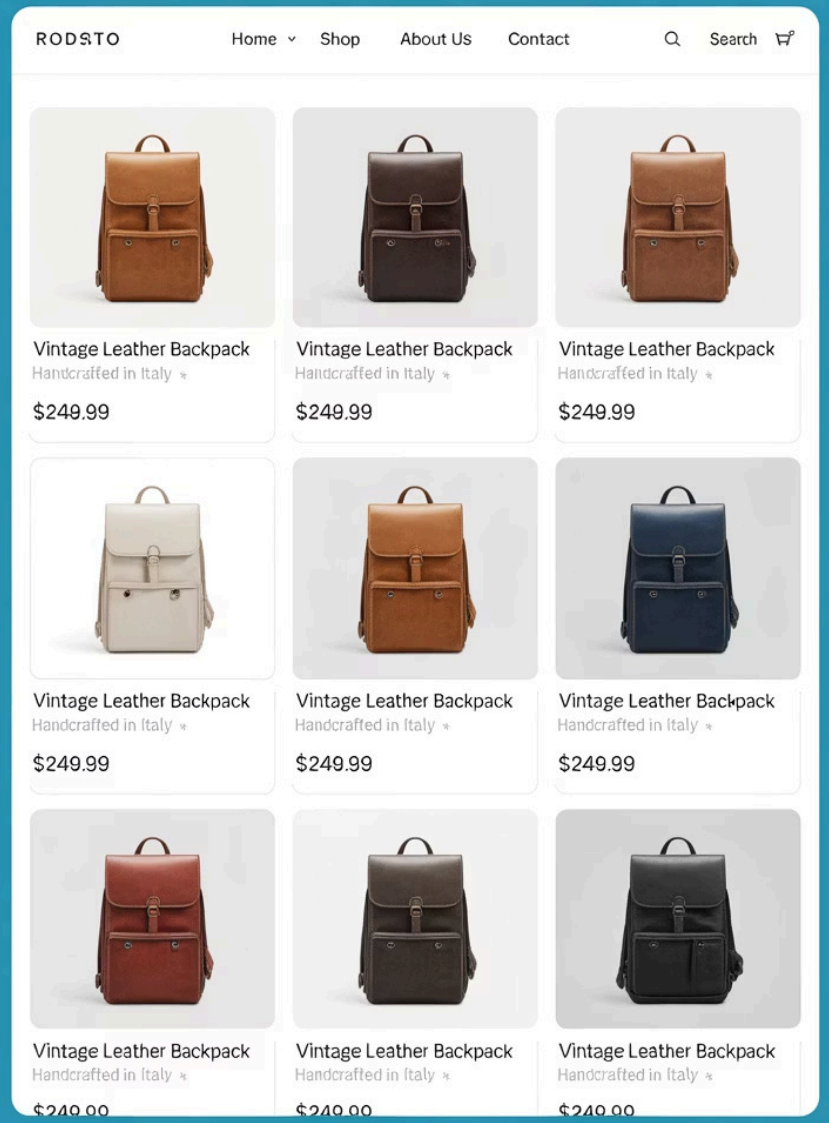
Envía solicitudes HTTP al servidor

**Vista**

Recibe solicitudes y usa datos de los modelos

**Plantilla**

Devuelve respuesta (HTML, JSON, etc.)

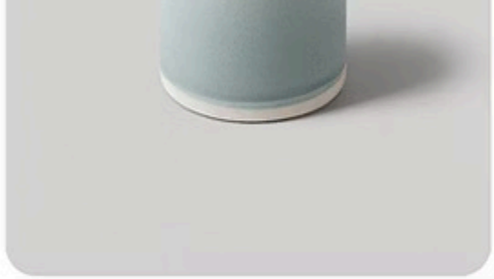



# Vista con Datos de Ejemplo

```
from django.shortcuts import render

def lista_productos(request):
    productos = [
        {"nombre": "Producto A", "precio": 100.50, "disponible": True},
        {"nombre": "Producto B", "precio": 200.75, "disponible": False},
        {"nombre": "Producto C", "precio": 50.99, "disponible": True},
    ]
    return render(request, 'productos/lista.html', {'productos': productos})
```

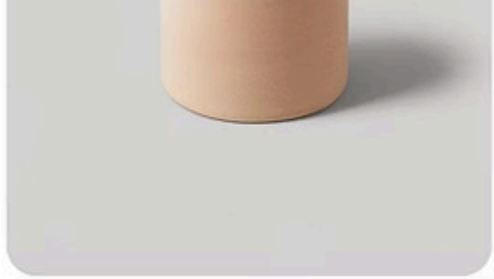
**Esta vista simula datos de productos y los pasa a la plantilla para su renderizado.**



Succulent Planter

Succulent Planter

Succulent Planter



Succulent Planter

# Plantilla HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Lista de Productos</title>
</head>
<body>
  <h1>Productos Disponibles</h1>
  <ul>
    {% for producto in productos %}
      <li>
        <strong>{{ producto.nombre }}</strong> - ${{ producto.precio }}
        {% if producto.disponible %}
          <span>(Disponible)</span>
        {% else %}
          <span>(No disponible)</span>
        {% endif %}
      </li>
    {% endfor %}
  </ul>
</body>
</html>
```

# Configuración de URL

### Definir URL

Configurar la ruta que conectará con nuestra vista

### Conectar Vista

Asociar la URL con la función de vista correspondiente

### Renderizar Plantilla

La vista procesa y devuelve la plantilla HTML

```
# urls.py
from django.urls import path
from . import views

urlpatterns = [
 path('productos/', views.lista_productos, name='lista_productos'),
]
```

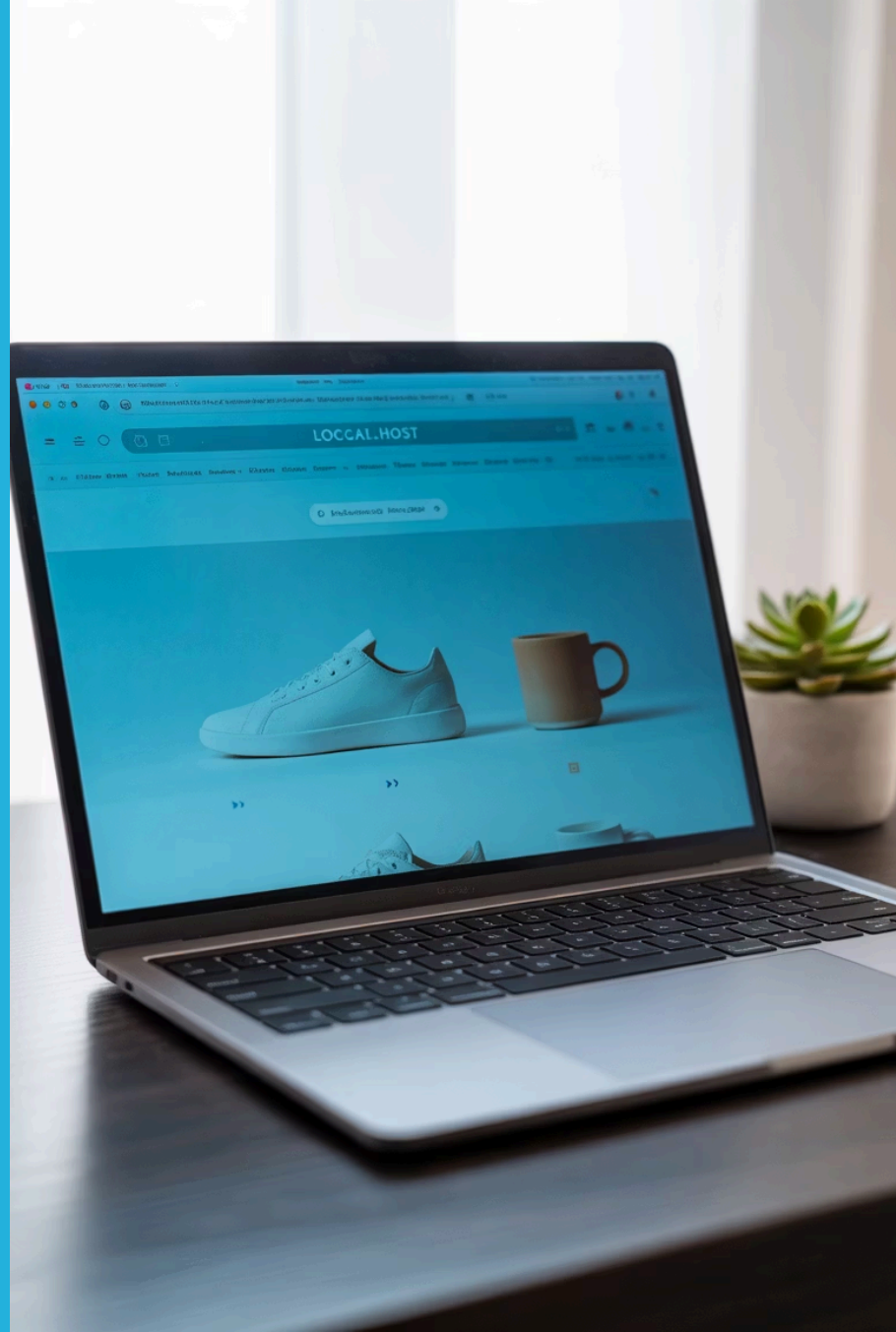
# Demostración en Navegador

**Abrir en el Navegador**

**Navegar a http://127.0.0.1:8000/productos/**

**Deberías ver la lista de productos simulada funcionando perfectamente**

**Resultado esperado: Una página web limpia mostrando los tres productos con sus precios y estado de disponibilidad**



# Metodología Activa - Actividad 3

### Crear Modelo Estudiante

Definir con campos: nombre, edad, curso y estado (activo/inactivo)

### Desarrollar Vista

Crear función que maneje lista de estudiantes ficticios

### Diseñar Plantilla

Mostrar lista de estudiantes con formato atractivo

**Ejercicio individual: 15 minutos para completar y probar en el navegador**

# Concepto de Migraciones (Teórico)

1. **Definición de Modelos**

   Crear las clases Python que representan nuestros datos

2. **makemigrations**

   Django genera archivos de migración basados en los modelos

3. **migrate**

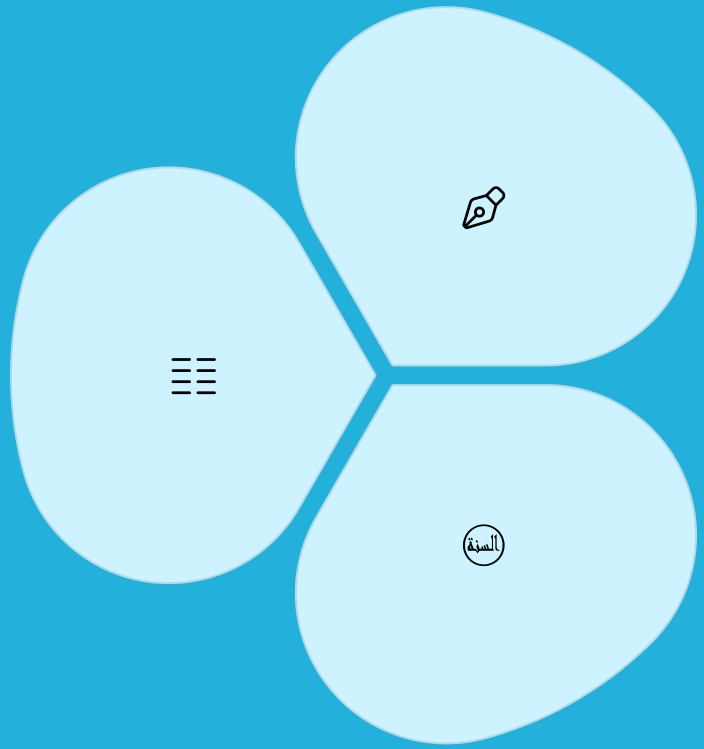
   Se ejecutan las migraciones para crear tablas reales en la BD

Las migraciones son la traducción automática de modelos Python a tablas de base de datos reales. Sin BD conectada, solo se queda en la definición conceptual.

# Reflexión Didáctica

**Organización**

Los modelos permiten ordenar ideas y datos de manera estructurada



**Planificación**

Aunque no tengamos BD, ya tenemos la estructura lista para implementar

**Fundamentos**

Es como armar el esqueleto de una casa antes de construirla completamente

# Aplicación Práctica Completa

**Definir modelo Producto**

Crear la clase con todos los campos necesarios

**Crear vista lista_productos**

Implementar la lógica para manejar los datos

**Crear plantilla lista.html**

Diseñar la interfaz de usuario

**Configurar URL**

Establecer las rutas de navegación

**Ver en navegador**

Probar el resultado final funcionando

# Cierre y Conclusiones

**Corazón de Django**

Los modelos son el núcleo fundamental de cualquier aplicación Django

**Flexibilidad de Desarrollo**

Se pueden definir desde el inicio, sin necesidad de BD configurada

**Herramientas de Planificación**

Sirven para planificar, estructurar y probar antes de la implementación final

**Próximos Pasos**

En la próxima clase veremos cómo conectar modelos con la BD real

# Preguntas y Retroalimentación

### Espacio para Dudas

Momento para resolver cualquier pregunta sobre los conceptos vistos

### Quiz Interactivo

Breve evaluación:

- ¿Qué es un modelo en Django?
- ¿Qué pasa si no hay BD conectada?
- ¿Cómo se pasan datos a una plantilla?

Participación activa: Todos los estudiantes pueden contribuir con sus experiencias y aprendizajes del día