

Curso de Django

Módulo 1: Introducción a Django

Lcdo. Diego Medardo Saavedra García. Mgtr.

2023-07-19

Tabla de contenidos

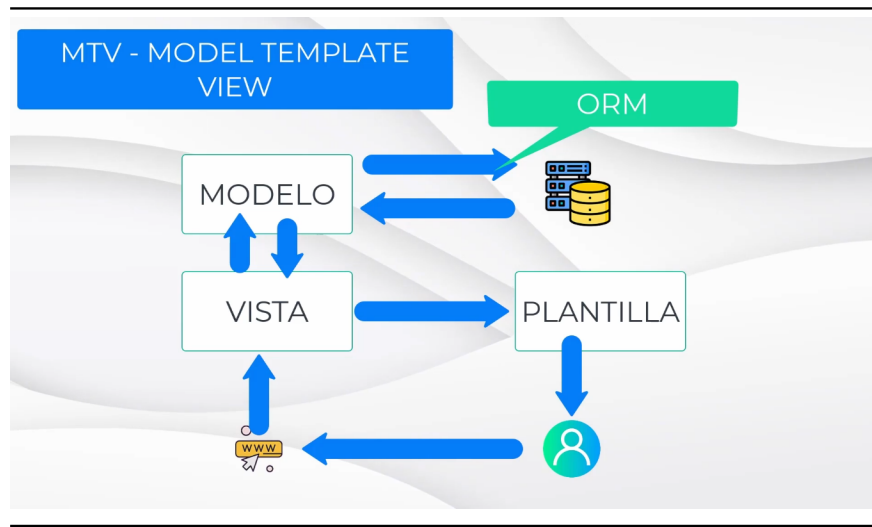
1	Módulo 1: Introducción a Django.	1
1.1	¿Qué es Django y por qué utilizarlo?	1
1.2	MVC vs MTV.	2
1.3	Instalación de Django 4.2.3 y Configuración del Entorno de Desarrollo	2
1.4	Creación de un Proyecto en Django.	3
1.4.1	Estructura de directorios generada:	3
1.4.2	Estructura de Directorios de un Proyecto Django.	3
1.5	Creación de una Aplicación en Django	4
2	Ejemplo Práctico:	5
2.1	Creación de un Proyecto de E-commerce en Django	5
3	Actividad Práctica.	7
3.1	Creación de Un Blog	7
3.2	Resolución de la Actividad Práctica:	7

1 Módulo 1: Introducción a Django.

1.1 ¿Qué es Django y por qué utilizarlo?

- Django es un framework web de alto nivel basado en Python.
- Facilita el desarrollo rápido de aplicaciones web robustas y seguras.
- Ventajas de Django: MVC (Modelo-Vista-Controlador), administrador de base de datos, seguridad integrada y comunidad activa.

1.2 MVC vs MTV.



1.3 Instalación de Django 4.2.3 y Configuración del Entorno de Desarrollo

Para instalar Django 4.2.3, se recomienda utilizar un entorno virtual (por ejemplo, con virtualenv o conda).

Comandos para instalar Django y crear un entorno virtual:

```
# Instalación de virtualenv

pip install virtualenv

# Creación del entorno virtual

python -m venv env

# Activación del Entorno Virtual

cd env/Scripts
activate

# Salir del directorio Scripts
cd ../../

# Instalación de Django usando pip
```

```
pip install django==4.2.3
```

1.4 Creación de un Proyecto en Django.

Para crear un nuevo proyecto Django, utilizamos el comando

```
django-admin startproject nombre_proyecto .
```

1.4.1 Estructura de directorios generada:

```
nombre_proyecto/  
    manage.py  
    __init__.py  
    settings.py  
    urls.py  
    asgi.py  
    wsgi.py
```

1.4.2 Estructura de Directorios de un Proyecto Django.

La estructura de directorios de un proyecto Django se organiza de la siguiente manera:

```
nombre_proyecto/  
    manage.py  
  
    nombre_proyecto/  
        __init__.py  
        settings.py  
        urls.py  
        asgi.py  
        wsgi.py  
  
    otras_aplicaciones/  
        ...
```

Directorio y/o Archivo	Descripción
nomb re_proyecto/	Es el directorio raíz del proyecto. Contiene el archivo “manage.py”, que es una herramienta para administrar el proyecto y ejecutar comandos de Django.
• *nombre_proyecto/nomb re_proyecto/** init.py	Es el directorio de la configuración del proyecto. Contiene varios archivos esenciales:
• *settings.py*	Este archivo indica a Python que el directorio es un paquete y permite la importación de módulos dentro de él.
urls.py	Aquí se encuentran todas las configuraciones del proyecto, como bases de datos, aplicaciones instaladas, rutas de plantillas, configuraciones de seguridad, etc.
asgi.py y wsgi.py	Contiene las configuraciones de las URLs del proyecto, es decir, cómo se manejan las solicitudes y se mapean a las vistas.
otras_aplicaciones/	Son archivos de configuración para el servidor ASGI (Asynchronous Server Gateway Interface) y WSGI (Web Server Gateway Interface), respectivamente. Estos archivos son utilizados por servidores web para servir la aplicación Django.
	Es un directorio opcional donde puedes organizar las aplicaciones adicionales que desarrolles para el proyecto. Cada aplicación puede tener su propia estructura de directorios.

1.5 Creación de una Aplicación en Django

Una aplicación es un componente reutilizable de un proyecto Django. Comando para crear una nueva aplicación:

```
python manage.py startapp nombre_app
```

Estructura de directorios de una aplicación:

```
nombre_app/
  migrations/
    ...
    __init__.py

  admin.py
  apps.py
```

```
models.py
tests.py
views.py
```

Directorio y/o Archivo	Descripción
nombre_app	Es el directorio raíz de la aplicación. Contiene los archivos esenciales y directorios para el funcionamiento de la aplicación.
migrations/	Es un directorio generado automáticamente por Django cuando se realizan cambios en los modelos de la aplicación. Contiene archivos de migración que representan los cambios en la base de datos.
init.py	Este archivo indica a Python que el directorio es un paquete y permite la importación de módulos dentro de él.
admin.py	En este archivo se pueden registrar los modelos de la aplicación para que aparezcan en el panel de administración de Django.
apps.py	Es el archivo donde se define la configuración de la aplicación, como su nombre y configuraciones adicionales.
models.py	Aquí se definen los modelos de la aplicación utilizando la clase “Model” de Django. Los modelos representan las tablas en la base de datos y definen los campos y relaciones de la aplicación.
tests.py	Es el archivo donde se pueden escribir pruebas unitarias y de integración para la aplicación.
views.py	Contiene las vistas de la aplicación, que son funciones o clases que manejan las solicitudes y generan las respuestas. Las vistas determinan qué se muestra en las páginas web de la aplicación.

2 Ejemplo Práctico:

2.1 Creación de un Proyecto de E-commerce en Django

En esta actividad práctica, crearás un nuevo proyecto de e-commerce en Django desde cero. Asegúrate de tener Django instalado en tu entorno de desarrollo antes de comenzar.

Paso 1: Crear un Proyecto de Django

Abre una terminal o línea de comandos en la ubicación donde desees crear tu proyecto de e-commerce.

Ejecuta el siguiente comando para crear un nuevo proyecto Django llamado “mi_ecommerce”:

```
django-admin startproject mi_ecommerce
```

Verás que se ha creado un nuevo directorio llamado “mi_ecommerce” que contiene la estructura inicial del proyecto.

Paso 2: Crear una Aplicación para el E-commerce

Cambia al directorio recién creado “mi_ecommerce”:

```
cd mi_ecommerce
```

Ahora, crea una nueva aplicación llamada “ecommerce” utilizando el siguiente comando:

```
python manage.py startapp ecommerce
```

Se creará un nuevo directorio “ecommerce” dentro de tu proyecto, que contendrá todos los archivos necesarios para la aplicación.

Paso 3: Configurar el Proyecto y la Aplicación

Abre el archivo “settings.py” ubicado en el directorio “mi_ecommerce/mi_ecommerce”.

Asegúrate de agregar la aplicación “ecommerce” en la lista de “INSTALLED_APPS” para que Django la reconozca:

```
INSTALLED_APPS = [  
    # Otras aplicaciones...  
    'ecommerce',  
]
```

Paso 4: Ejecutar el Servidor de Desarrollo

Ahora, ejecuta el servidor de desarrollo para ver el proyecto en acción:

```
python manage.py runserver
```

Abre tu navegador y visita la dirección “<http://localhost:8000/>”. Deberías ver la página de bienvenida de Django.

Si has llegado a este punto, ¡Felicidades! Has creado con éxito un nuevo proyecto de e-commerce en Django y una aplicación llamada “ecommerce”. Ahora puedes comenzar a desarrollar las funcionalidades del e-commerce y diseñar las diapositivas para cada uno de los módulos del curso.

¡Buena suerte!

3 Actividad Práctica.

3.1 Creación de Un Blog

En esta actividad práctica, crearás un nuevo proyecto de Blog en Django desde cero. Asegúrate de tener Django instalado en tu entorno de desarrollo antes de comenzar.

Paso 1: Crear un Proyecto de Django

Paso 2: Crear una Aplicación para el E-commerce

Paso 3: Configurar el Proyecto y la Aplicación

Paso 4: Ejecutar el Servidor de Desarrollo

3.2 Resolución de la Actividad Práctica:

En esta actividad práctica, crearás un nuevo proyecto de blog en Django desde cero. Asegúrate de tener Django instalado en tu entorno de desarrollo antes de comenzar.

Paso 1: Crear un Proyecto de Django

Abre una terminal o línea de comandos en la ubicación donde desees crear tu proyecto de blog.

Ejecuta el siguiente comando para crear un nuevo proyecto Django llamado “mi_blog”:

```
django-admin startproject mi_blog
```

Verás que se ha creado un nuevo directorio llamado “mi_blog” que contiene la estructura inicial del proyecto.

Paso 2: Crear una Aplicación para el Blog

Cambia al directorio recién creado “mi_blog”:

```
cd mi_blog
```

Ahora, crea una nueva aplicación llamada “blog” utilizando el siguiente comando:

```
python manage.py startapp blog
```

Se creará un nuevo directorio “blog” dentro de tu proyecto, que contendrá todos los archivos necesarios para la aplicación.

Paso 3: Configurar el Proyecto y la Aplicación

Abre el archivo “settings.py” ubicado en el directorio “mi_blog/mi_blog”.

Asegúrate de agregar la aplicación “blog” en la lista de “INSTALLED_APPS” para que Django la reconozca:

```
INSTALLED_APPS = [  
    # Otras aplicaciones...  
    'blog',  
]
```

Paso 4: Ejecutar el Servidor de Desarrollo

Ahora, ejecuta el servidor de desarrollo para ver el proyecto en acción:

```
python manage.py runserver
```

Abre tu navegador y visita la dirección “http://localhost:8000/”. Deberías ver la página de bienvenida de Django.

¡Felicidades! Has creado con éxito un nuevo proyecto de blog en Django y una aplicación llamada “blog”. Ahora puedes comenzar a desarrollar las funcionalidades del blog y diseñar las diapositivas para cada uno de los módulos del curso. ¡Buena suerte!