 # preyect_final_django

1. Configuración del Entorno de Desarrollo:

Asegúrate de tener Python y Django instalados en tu sistema.

Crea un entorno virtual para el proyecto:

python -m venv myenv

Activa el entorno virtual:

En Windows: myenv\Scripts\activate

En macOS/Linux: source myenv/bin/activate

Instala los requisitos del proyecto: pip install django

2. Configuración del Proyecto Django:

Crea un nuevo proyecto Django: django-admin startproject ventas_web

Crea una nueva aplicación dentro del proyecto: cd ventas_web

python manage.py startapp ventas_web

Define los modelos necesarios en ventas_web/models.py y asegúrate de ejecutar las migraciones:

python manage.py makemigrations

python manage.py migrate

3. Implementación de las Vistas y URLs:

Define las vistas necesarias en ventas_web/views.py.

Configura las URLs en ventas_web/urls.py.

4. Creación de los Templates:

Crea los templates HTML en el directorio ventas_web/templates/.

Asegúrate de que los nombres de los templates coincidan con las vistas definidas y que contengan el código HTML necesario.

5. Estilos y Scripts:

Agrega archivos CSS y JavaScript si es necesario en el directorio ventas_web/static/.

Configuración Inicial del Proyecto

Para iniciar un proyecto Django, es esencial crear un entorno de desarrollo adecuado. Esto implica establecer una carpeta específica para el proyecto y utilizar un entorno virtual para instalar Django y gestionar las dependencias de manera aislada . Una vez configurado el entorno, se puede crear el proyecto con el comando django-admin startproject seguido del nombre del proyecto . Este comando genera una estructura de directorios y archivos necesarios para el proyecto .

Creación y Registro de Aplicaciones

Dentro del proyecto Django, se pueden crear aplicaciones específicas que contienen la lógica y las funcionalidades del sitio web . Cada aplicación se debe registrar en el archivo settings.py del proyecto para que Django la reconozca como parte del mismo .

Modelos y Administración

Los modelos en Django definen la estructura de la base de datos y permiten interactuar con ella. En este proyecto, se han creado modelos para Product, Transaction, Cart, y CartItem . Estos modelos incluyen campos para almacenar información relevante como el nombre, descripción y precio de los productos, así como las transacciones y los elementos del carrito de compras .

Para administrar estos modelos, se utilizan las funcionalidades de administración de Django, registrando los modelos en el archivo admin.py para que sean accesibles desde la interfaz de administración .

Formularios y Vistas

Los formularios en Django facilitan la recopilación y validación de datos de entrada del usuario. En este proyecto, se ha definido un ProductForm para manejar la creación y edición de productos .

Las vistas son funciones o clases que toman una solicitud web y devuelven una respuesta. Se han creado vistas para listar productos, ver detalles de productos, añadir al carrito, actualizar el carrito, eliminar del carrito, ver el carrito y realizar el proceso de pago .

Configuración de URLs

Las URLs en Django son patrones que se utilizan para dirigir las solicitudes web a las vistas correspondientes. Se han configurado varias URLs para manejar las diferentes páginas y acciones

del sitio web, como la lista de productos, el registro de usuarios, la adición de productos al carrito, entre otros .

Configuración de Archivos Estáticos y Multimedia

Para servir archivos estáticos y multimedia, se han configurado las URLs y las rutas en el archivo settings.py y urls.py del proyecto. Esto incluye la definición de STATIC_URL, STATICFILES_DIRS, MEDIA_URL, y MEDIA_ROOT .

Base de Datos y Migraciones

Django utiliza migraciones para aplicar cambios en la base de datos basados en las definiciones de los modelos. Se ha configurado una base de datos SQLite para el proyecto y se han ejecutado migraciones para crear las tablas correspondientes a los modelos .

Ejecución del Servidor de Desarrollo

Finalmente, para ver el proyecto en acción, se inicia el servidor de desarrollo con el comando python manage.py runserver y se accede a la aplicación a través del navegador .

Este informe proporciona una visión general de cómo se ha configurado y estructurado el proyecto Django. Cada paso es crucial para asegurar que el proyecto funcione correctamente y que las funcionalidades estén disponibles para los usuarios finales.