Proyecto de fin de grado del ciclo formativo de grado superior

Desarrollo de Aplicaciones Web

(DAW)

TechCore

Tecnología de vanguardia para todos los públicos

Alumno: Salvador García Olmedo

Curso: 2024-2025

Centro: IES Martínez Montañés

Tutor/a: José Luís Fernández Yagües

ÍNDICE DE CONTENIDO

RESUMEN DEL PROYECTO

Este proyecto de fin de grado tiene como objetivo desarrollar una aplicación web de comercio electrónico basada en el framework Django, con PostgreSQL como sistema de gestión de bases de datos e implementada mediante contenedores Docker. La tienda online permite a los usuarios explorar el catálogo de productos, registrarse, realizar pedidos y gestionar sus perfiles de forma sencilla e intuitiva.

La característica principal del proyecto es la personalización de productos, que permite a los usuarios añadir sus propias frases, logotipos o imágenes antes de la compra. Esta función añade valor al modelo tradicional de comercio electrónico y abre un sinfín de posibilidades en áreas como la personalización de regalos, el merchandising y la impresión bajo demanda.

El sistema adopta un diseño modular, escalable y seguro, integra las mejores prácticas de desarrollo web y garantiza la portabilidad y replicabilidad del entorno mediante Docker. La base de datos está modelada para gestionar eficientemente productos, usuarios, pedidos y personalizaciones.

El proyecto no solo demostró la aplicación práctica de los conocimientos adquiridos durante la formación, sino que también incorporó un enfoque práctico y pragmático aplicable al ámbito profesional.

INTRODUCCIÓN DEL PROYECTO

Actualmente, el comercio electrónico está en pleno desarrollo y se ha convertido en una de las principales formas de consumo. Plataformas como Amazon, Etsy y Shopify han liderado la tendencia, permitiendo a cualquier usuario comprar desde casa, disfrutar de una experiencia de compra personalizada y recibir pedidos rápidamente. En este contexto, los comercios que ofrecen productos personalizados están en auge, ya que aportan un valor emocional adicional a la compra.

El proyecto busca combinar las tecnologías existentes con las necesidades del mercado para ofrecer una solución de tienda online que, además de cubrir las funciones básicas de un sitio web de comercio electrónico, permita a los usuarios personalizar los productos con frases, logotipos o sus propias imágenes, mejorando así el atractivo del producto final.

OBJETIVOS

Desarrollar una tienda en línea personalizada con Django, PostgreSQL y Docker que permita a los usuarios modificar los elementos gráficos o de texto de un producto antes de comprarlo.

Objetivos específicos

- Implementar un sistema de registro, inicio de sesión y gestión de usuarios.
- Crear un catálogo dinámico de productos con funciones de ordenación y filtrado.
- Diseñar una función personalizada que permite añadir texto o imágenes a los productos.
- Integrar una base de datos relacional PostgreSQL para una gestión eficiente de la información.
- Contenerizar toda la aplicación con Docker para facilitar su implementación y portabilidad.
- Aplicar las mejores prácticas de diseño y desarrollo web responsivo, como HTML, CSS y Bootstrap.
- Garantizar la seguridad y la validación de formularios y procesos clave.

JUSTIFICACIÓN DEL PROYECTO

Elegí este proyecto porque me permitió aplicar diversas habilidades aprendidas durante la formación, como desarrollo backend, gestión de bases de datos, desarrollo frontend e implementación de contenedores. Además, el enfoque personalizado representó un desafío técnico interesante, ya que requería la gestión de imágenes, formularios dinámicos y vistas previas en vivo.

Además, el modelo de tienda personalizado tenía un gran valor práctico y comercial, lo que me motivó aún más, ya que sabía que era útil y viable en un entorno real. También me interesaba utilizar tecnologías actuales como Django y Docker, ya que son ampliamente utilizadas en el mercado laboral.

METODOLOGÍA GENERAL DE TRABAJO

La metodología que he seguido ha sido una combinación de desarrollo incremental y modelo iterativo, basada en el enfoque ágil. El proyecto se ha dividido en diferentes fases (análisis, diseño, desarrollo, pruebas y despliegue), permitiendo avanzar y realizar mejoras continuas tras cada fase.

Para la planificación y desarrollo de tareas, se han utilizado herramientas como Git, permitiendo un control de versiones y una organización clara del trabajo. Cada apartado(registro, catálogo, personalización, etc.) ha sido desarrollado de forma aislada y luego integrado al sistema principal. Asimismo, se han realizado pruebas funcionales al finalizar cada etapa para asegurar el correcto funcionamiento antes de continuar.

ANÁLISIS DEL SISTEMA

REQUISITOS FUNCIONALES

1. Registro y autenticación de usuarios
    - Los usuarios podrán registrarse mediante un formulario.
    - Se implementará un sistema de login y logout seguro.
    - Se ofrecerá un panel de usuario donde consultar pedidos.

2. Gestión de productos
    - Visualización de productos por categoría.
    - Filtro y búsqueda de productos por nombre o tipo.
    - Detalle del producto con descripción, precio y opciones de personalización.

3. Personalización de productos
    - El usuario podrá añadir texto, seleccionar tipo de letra y color.
    - Se permitirá subir imágenes o logos para integrarlos en el producto.
    - Se ofrecerá una previsualización del producto personalizado.

4. Carrito de compra y pedidos
    - Añadir productos al carrito (con o sin personalización).
    - Modificar o eliminar elementos del carrito.
    - Realizar el pedido final, con almacenamiento de los datos en la base de datos.

5. Panel de administración
    - Acceso para el administrador para gestionar productos, categorías y pedidos.
    - Validación de imágenes subidas y control del contenido personalizado.

REQUISITOS NO FUNCIONALES

1. Escalabilidad
    - Uso de Docker para desplegar la aplicación en distintos entornos.

2. Portabilidad
    - Al estar contenerizada, puede ejecutarse en cualquier máquina con Docker.

3. Seguridad
    - Se controlan formularios, autenticación y subida de imágenes.

4. Usabilidad
    - Diseño responsive con Bootstrap, interfaz intuitiva y limpia.

5. Mantenibilidad
    - Estructura modular de Django y uso de Git para control de versiones.

REQUISITOS DEL SISTEMA

- Lenguaje de programación: Python 3.

- Framework backend: Django

- Base de datos: PostgreSQL 15

- Contenedores: Docker y Docker Compose

- Frontend: HTML5, CSS3, JavaScript, Bootstrap

- Entorno de desarrollo: VSCode, GitHub

- Sistema operativo: Linux / Windows con Docker instalado

DISEÑO DEL SISTEMA

ARQUITECTURA GENERAL

Mi proyecto sigue una arquitectura *Modelo-Vista-Controlador*(**MVC**), utilizando el framework Django:

- **Modelos(Models)**: Representa las entidades del sistema como usuario, personalización o pedido, entre otros
- **Vistas(Views)**: Permiten manejar la lógica de negocio y los datos enviados a la plantilla
- **Plantillas(Templates)**: Crean la interfaz de usuario utilizando HTML y Bootstrap

DIAGRAMA GENERAL DEL SISTEMAS

User: Usuario base del sistema. Desde aquí se relacionan el perfil, dirección, tarjeta de pago, carrito y compras.

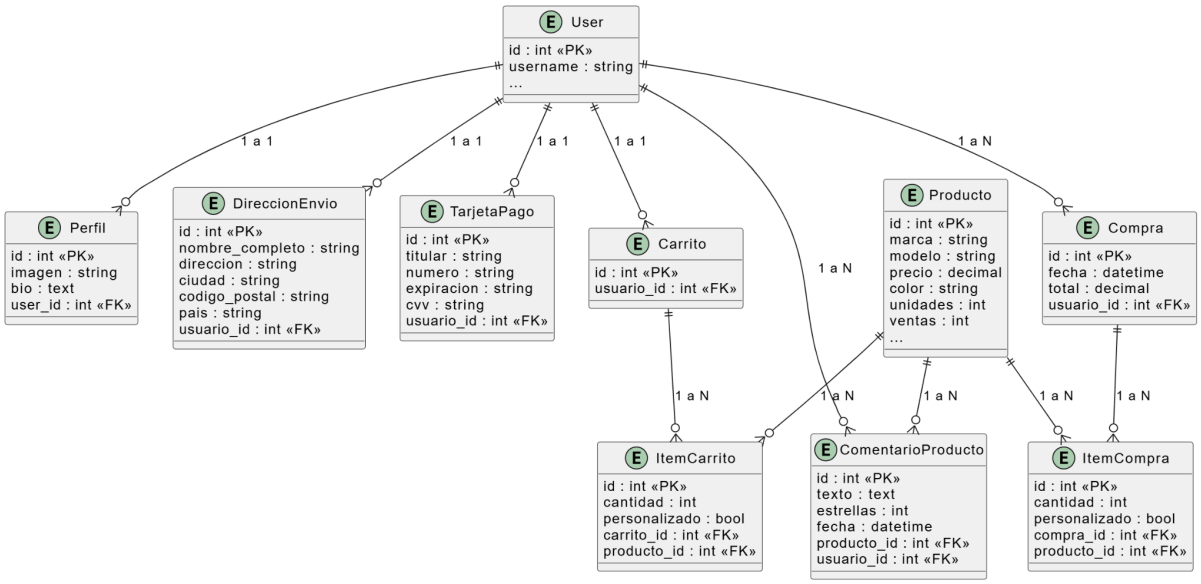
Producto: Representa los productos en venta, incluyendo color, precio, unidades disponibles y posibilidad de personalización.

Carrito / ItemCarrito: Relación entre los usuarios y los productos seleccionados para compra, incluyendo si son personalizados.

Compra / ItemCompra: Registro de compras realizadas, productos y cantidad.

ComentarioProducto: Sistema de valoración de productos.

Perfil / DireccionEnvio / TarjetaPago: Datos adicionales del usuario para gestionar identidad, envíos y pagos.


User
id : int «PK»
username : string
...
1 a 1
1 a 1
1 a 1
1 a 1
1 a N
Perfil
id : int «PK»
imagen : string
bio : text
user_id : int «FK»
DireccionEnvio
id : int «PK»
nombre_completo : string
direccion : string
ciudad : string
codigo_postal : string
pais : string
usuario_id : int «FK»
TarjetaPago
id : int «PK»
titular : string
numero : string
expiracion : string
cvv : string
usuario_id : int «FK»
Carrito
id : int «PK»
usuario_id : int «FK»
Producto
id : int «PK»
marca : string
modelo : string
precio : decimal
color : string
unidades : int
ventas : int
...
Compra
id : int «PK»
fecha : datetime
total : decimal
usuario_id : int «FK»
1 a N
1 a N
1 a N
1 a N
1 a N
1 a N
ItemCarrito
id : int «PK»
cantidad : int
personalizado : bool
carrito_id : int «FK»
producto_id : int «FK»
ComentarioProducto
id : int «PK»
texto : text
estrellas : int
fecha : datetime
producto_id : int «FK»
usuario_id : int «FK»
ItemCompra
id : int «PK»
cantidad : int
personalizado : bool
compra_id : int «FK»
producto_id : int «FK»

DESARROLLO DEL SISTEMA

1. Instalación, creación y configuración
   - Instalación del framework Django, PostgreSQL, y la configuración de Docker Compose.
   - Creación y configuración del entorno virtual (***.env***)

2. Estructura de un proyecto django
   - Creación del proyecto y posteriormente la app de django para poder trabajar.

3. Backend
   - Definición de modelos con relaciones adecuadas (ForeignKeys, OneToOne, etc).
   - Implementación de formularios de registro, login, personalización y pago simulado.
   - Vistas protegidas para usuarios registrados.

4. Frontend
   - Uso de Bootstrap para diseño responsive.
   - Previsualización en tiempo real mediante JavaScript del texto o imagen sobre el producto.
   - Formularios dinámicos y validaciones visuales.

5. Uso de Docker
   - Creación de DockerFile.
   - Creación y configuración del docker-compose.yml permitiendo levantar el contenedor con Django y PostgreSQL.
   - Manejo de volúmenes para persistencia de imágenes y base de datos.

6. Pruebas
   - Pruebas funcionales: navegación, creación de pedidos y personalización, entre otros