Proyecto: Tienda de Ropa

Universidad Distrital Francisco José de Caldas

Facultad de Ingeniería

Proyecto: Ingeniería de Sistemas

Fundamentos de Ingeniería de Software

Docente: Santiago Salazar Fajardo

Integrantes: Diego Alejandro Parra Diaz 20212020091
Esteban Villalba 20212020064
Valentina Parra Ordóñez 20212020025
Jonathan Felipe Ochoa Silva – 20212020039

9 de Julio de 2025

Contenido

Introducción:

El presente documento reúne y organiza de manera integral todos los artefactos generados durante el desarrollo del proyecto *"Sistema de Tienda de Ropa Online"*, realizado como parte de la asignatura **Fundamentos de Ingeniería de Software**. Este proyecto tuvo como objetivo diseñar, planificar y documentar un sistema web que permita a los usuarios personalizar y adquirir camisetas de manera eficiente, aplicando las mejores prácticas de la ingeniería de software.

A lo largo del desarrollo del proyecto, se emplearon metodologías ágiles, particularmente **Scrum**, para gestionar el trabajo en equipo y garantizar entregas iterativas e incrementales. En este documento se incluyen los procesos de planificación, estimación, gestión de riesgos, definición de roles, arquitectura del sistema, diagramas UML, historias de usuario y los planes de prueba realizados.

El propósito de este compendio es presentar una visión clara y ordenada del proceso seguido, evidenciando las actividades realizadas y los resultados alcanzados en cada etapa. Esto permite reflejar no solo el producto final obtenido, sino también el proceso de trabajo colaborativo y de aprendizaje adquirido durante el desarrollo del proyecto.

Integrantes:

Diego Alejandro Parra Díaz – 20212020091

- Líder del equipo: Se encargará de la gestión del grupo, asegurando el cumplimiento de tareas, facilitando reuniones y manteniendo informado al instructor. Además, será el guía para sus compañeros.

Esteban Villalba – 20212020064

- Responsable de desarrollo: Sera el dirigente en la planificación técnica, desarrollo e implementación del producto, asegurando estándares de calidad.

Jonathan Felipe Ochoa Silva – 20212020039

- Responsable de planeación: Organizará el plan de trabajo del equipo, estableciendo cronogramas y seguimiento a la ejecución del proyecto.

Valentina Parra Ordoñez – 20212020025

- Responsable de calidad: Garantizara que se sigan procesos definidos y se mantenga la calidad del producto final mediante revisiones e inspecciones.

Planteamos el siguiente objetivo general con el fin de profundizar y mejorar en todo aspecto mientras cursemos la materia Fundamentos de Ingeniería de Software.

- Desarrollar un portal web para la personalización y compra de camisetas estampadas, que permita a usuarios, artistas y administradores interactuar con el sistema según sus roles,

integrando funcionalidades de gestión navegación y venta, aplicando principios de la ingeniería de software y metodologías ágiles como Scrum.

Objetivos específicos:

- Diseñar e implementar una interfaz para usuarios compradores, que permita navegar en un catálogo de estampas, donde pueda seleccionar tallas, colores, materiales y realizar las respectivas compras.
- Desarrollar una interfaz exclusiva para artistas, la cual permita subir, editar y administrar diseños de estampas, así como poder llevar una estadística de sus ventas y ratings.
- Construir un módulo administrativo para gestionar usuarios, tarifas, temas, estadísticas del portal y configuraciones generales del sistema.
- Aplicar técnicas de estimación como PROBE y Planning Poker, para determinar y analizar el esfuerzo requerido en las funcionalidades del sistema.
- Planificar el desarrollo mediante la metodología ágil Scrum, organizando el trabajo en sprints con entregables parciales de acuerdo a historias de usuario priorizadas.

Planeación del Proyecto:

Presenta la planeación del proyecto mediante la organización de actividades, asignación de responsables y las entregas previstas en cada sprint. Se ha divido el proyecto en tres ciclos de trabajo (sprints), priorizando historias de usuario clave para avanzar de forma iterativa en el desarrollo del portal.

Sprint 1: Configuración inicial y funcionalidades básicas (semana 1)

Actividad	Historia de Usuario	Responsable	Fecha Inicio	Fecha Fin	Entregable
Configuración de repositorio y entorno de trabajo	-	Diego	19-Mayo	23-Mayo	Repositorio en GitHub y estructura base
Diseño del prototipo visual (UI)	Personalizar Camisetas	Valentina	19-Mayo	23-Mayo	Prototipo en figma para personalización
Registro e inicio de sesión	Registrar cuenta/ inicio de sesión	Esteban	19-Mayo	23-Mayo	Módulo funcional de login y registro
Agregar al carrito (estructura básica)	Agregar al carrito	Jonathan	19-Mayo	23-Mayo	Funcionalidad de agregar al carrito básico

Sprint 2: Funcionalidades de compra y artista (Semana 2)

Actividad	Historia de Usuario	Responsable	Fecha Inicio	Fecha Fin	Entregable
Personalización de camiseta con estampa	Personalizar camisetas	Diego	26-may	30-may	Vista interactiva con estampa aplicada
Ver y eliminar productos del carrito	Eliminar camisetas, ver productos en el carrito	Valentina	26-may	30-may	Carrito funcional con productos listados
Subida de estampas por artista	Inicio de sesión como cliente/artista	Esteban	26-may	30-may	Módulo de carga de imágenes para artistas
Creación del catálogo del artista	Registrar cuenta como cliente/artista	Jonathan	26-may	30-may	Catálogo por usuario artista

Sprint 3: Integración, pago y pruebas (Semana 3)

Actividad	Historia de Usuario	Responsable	Fecha Inicio	Fecha Fin	Entregable
Implementar módulo de pago	Pagar camisetas	Esteban	2-jun	6-jun	Compra y comprobante funcional
Revisión y pruebas de funcionalidades	Todas	Diego	9-jun	13-jun	Informe de errores y validaciones
Aprobación de diseños por administrador	Filtrar datos del catálogo por categorias	Valentina	16-jun	20-jun	Módulo de control de diseños aprobado
Despliegue del sistema y demo final	-	Todos	23-jun	27-jun	Versión funcional del sistema publicada

Se presenta la estimación de puntos de historia realizada por el equipo usando la técnica Planning Poker. Las historias están clasificadas por área funcional del sistema.

Compras:

Historia de Usuario	Puntos
Registro cliente/artista	4
Inicio de sesión cliente/artista	6
Filtros del catálogo	3
Agregar camiseta al carrito	7
Personalizar camiseta	7
Eliminar Camisetas del carrito	7
Ver productos del carrito	5
Pagar camisetas	9

Artistas:

Historia de Usuario	Puntos
Crear catálogo	6
Añadir diseño	7
Eliminar diseño	6
Habilitar producto para venta	4
Ver estadísticas de ventas	7

Administrador:

Historia de Usuario	Puntos
Ver rating de productos	6
Consultar y actualizar tarifas	6
Aprobar diseños de artistas	5

Planning Poker y Probe

Se presenta la estimación con PROBE (basada en experiencias previos) debes temer tareas o módulos similares que se tenga antes. Como es un proyecto nuevo podemos usar versiones adaptadas basada en dificultad, dependencias y referencias comunes en proyectos similares.

Historia	Proxy anterior	Tiempo proxy	Ajuste	Estimación final	Notas
Personalizar camiseta	Interfaz gráfica en proyecto anterior	10 h	+30 % (complejidad visual alta)	13 h	Uso de preview dinámica requiere JS y CSS avanzados
Agregar al carrito	Formulario interactivo básico	6 h	+50%(muchas variables: color, talla, cantidad)	9 h	Requiere validaciones lógicas

Inicio de sesión	Módulo de login en proyecto anterior	4 h	0 %	4 h	Requiere control de sesiones

Justificación de la Nueva Estimación

Debido a las condiciones particulares de los integrantes del equipo (estudiantes de universidad pública con trabajos y otras obligaciones académicas), se reestructuró el proyecto para poder cumplir los objetivos mínimos viables manteniendo la calidad del producto.

Metodología Adoptada

Se mantuvo el uso de **Scrum**, con adaptaciones a sprints más ajustados en tiempo, reuniones virtuales y enfoque en entregables iterativos, funcionales y documentados.

Estimación Recalculada

Sprint	Duración	Objetivo Principal	Resultado
1	14 al 18 de junio	Backend completo (CRUD, conexión DB)	Sistema funcional para usuarios y pedidos
2	19 al 23 de junio	Frontend (HTML, CSS, formularios)	Interfaz operativa con vistas básicas
3	24 al 30 de junio	Integración, validación y pruebas	Flujo completo de compra funcionando

Impacto del Reajuste

Área	Planeado	Real ejecutado	Comentario breve
Requerimientos funcionales	100%	90%	Se priorizó MVP y funcionalidades clave
Reuniones presenciales	Semanales	Virtuales	Se adaptó por paro académico
Pruebas	Exhaustivas	Básicas + validaciones	Enfocadas en funcionamiento principal
Documentación	Completa y por fases	Consolidada al final	Alineada a exigencias del docente

Cronograma Actualizado

Actividad	Fecha inicio	Fecha fin	Responsable	Estado
Definición de requerimientos	10-jun	13-jun	Todo el equipo	Finalizado
Sprint 1 - Backend	14-jun	18-jun	Pedro / Andrés	Finalizado
Sprint 2 - Frontend	19-jun	23-jun	Esteban	Finalizado
Sprint 3 - Integración y pruebas	24-jun	30-jun	Todos	Finalizado
Documentación técnica	1-jul	4-jul	Todos	En curso
Preparación de presentación final	2-jul	7-jul	Valentina/Diego	En curso
Entrega final y sustentación	9-jul	9-jul	Todo el equipo	Por hacer

A pesar de los desafíos contextuales, se cumplió con los entregables funcionales. Esta nueva estimación muestra cómo el equipo reorganizó su trabajo para cumplir el cronograma adaptado y mantener una entrega de calidad.

CONFORMACIÓN DEL EQUIPO Y ROLES

Nombre del proyecto: Tienda de Ropa Online

Miembros del equipo y roles Scrum:

- **Scrum Master:** Diego Alejandro Parra Diaz
  - Facilita reuniones, gestiona bloqueos, asegura el cumplimiento de Scrum.
- **Product Owner:** Valentina Parra Ordóñez
  - Define funcionalidades clave, prioriza el backlog, mantiene contacto con el profesor.
- **Desarrolladores:**
  - Esteban Villalba (Backend)
  - Jonathan Felipe Ochoa Silva (Frontend)
  - Diego Alejandro Parra Díaz (Base de datos y pruebas)

Funciones del proyecto:

- Registro y login de usuarios
- Catálogo de productos (camisas)

- Carrito de compras
- Registro de pedidos

OBJETIVOS DEL EQUIPO Y DE CADA ROL

Objetivo general del equipo:

Desarrollar una aplicación funcional de tienda online usando Scrum, aplicando buenas prácticas de desarrollo ágil, trabajo colaborativo y entregables iterativos.

Objetivos por rol:

Scrum Master: Asegurar fluidez del proyecto, cumplimiento de sprints y mejora continua.
Product Owner: Representar los intereses del "cliente" (profesor), definir backlog con enfoque funcional.
Desarrolladores: Implementar las funcionalidades requeridas de manera modular, con integración continua y pruebas.

Identificación y Gestión de Riesgos

Este documento describe los riesgos anticipados durante el desarrollo del proyecto, su análisis y los planes definidos para mitigar su impacto, siguiendo los principios de gestión ágil y adaptativa en un entorno académico con limitaciones reales.

Tabla de Gestión de Riesgos

Riesgo identificado	Probabilidad	Impacto	Mitigación	Contingencia
Paro académico y suspensión de clases	2/5	5/5	Reuniones virtuales por WhatsApp/Meet, replanteo del cronograma	Reducir tareas menos críticas, trabajar de forma asíncrona
Carga académica y laboral del equipo	4/5	3/5	División clara de tareas por rol, priorización de funcionalidades clave	Reducción del alcance, entregas mínimas viables documentadas
Errores en integración del sistema	3/5	5/5	Uso de repositorio ordenado, pruebas de integración frecuentes	Corrección colaborativa y revisión cruzada entre los integrantes
Desconocimiento técnico puntual	3/5	3/5	Consulta activa de documentación, apoyo mutuo, prototipos de prueba	Consultas al profesor o simplificación del módulo

Riesgo identificado	Probabilidad	Impacto	Mitigación	Contingencia
Falta de tiempo para pruebas profundas	5/5	5/5	Planificación anticipada, definición de pruebas esenciales y funcionales	Pruebas manuales estructuradas, validaciones con usuarios sim

Notas adicionales

- Los riesgos fueron discutidos en reunión y priorizados por frecuencia e impacto.
- Las decisiones fueron tomadas en consenso, considerando la realidad académica y personal de los integrantes.

Informe de Seguimiento de Riesgos

Este informe detalla cómo los riesgos identificados se manifestaron durante el proyecto y qué medidas fueron implementadas en respuesta, según lo planificado en el documento de gestión de riesgos.

1. **Carga académica y laboral de los integrantes**
   Resultado: Cada integrante cumplió con su parte adaptando los tiempos. Las tareas se dividieron de forma efectiva.
   Impacto real: Medio. En algunos momentos se retrasaron entregas internas, pero no afectó la entrega general.
   Lección aprendida: Delegar tareas pequeñas y calendarizar microentregas.
2. **Errores en integración del sistema**
   Resultado: Se presentaron pequeños bugs durante la unión frontend-backend, resueltos mediante revisión en pareja.
   Impacto real: Bajo. No comprometió funcionalidades críticas.
   Lección aprendida: Revisar continuamente ramas y realizar pruebas simples desde el inicio.
3. **Desconocimiento técnico punctual**
   Resultado: Se resolvió mediante investigación y prototipos. El equipo apoyó en lógica y documentación.
   Impacto real: Bajo. Se logró completar incluso las funcionalidades nuevas (como CRUD de pedidos).
   Lección aprendida: La colaboración y prueba temprana reduce tiempos de bloqueo.
4. **Tiempo limitado para pruebas**
   Resultado: Se optó por pruebas funcionales y no automatizadas. Se validaron casos críticos del flujo de usuario.
   Impacto real: Medio. No se realizaron tests automatizados, pero el sistema funciona de forma estable.
   Lección aprendida: Asegurar tiempo para testing desde el primer sprint.

describe la arquitectura lógica y física del sistema desarrollado como parte del proyecto de la asignatura, una tienda en línea funcional orientada al modelo cliente-servidor. Esta arquitectura

fue diseñada para garantizar modularidad, facilidad de mantenimiento y claridad en las responsabilidades de cada componente.

El proyecto fue desarrollado en Python y HTML/CSS puro, con separación entre frontend y backend, siguiendo una estructura MVC simplificada.

Componentes Principales del Sistema

1. **Frontend (Interfaz de Usuario)**

- Implementado en HTML, CSS y recursos estáticos (imágenes).
- Páginas principales: index.html, login.html, register.html, carrito.html.
- Estilos definidos en style.css.
- Accede al backend mediante formularios HTML estándar (sin JavaScript dinámico).

2. **Backend (Lógica de Negocio y Controlador)**

- Lenguaje: Python
- Framework: Flask (o base mínima de Python para servidor web)
- Archivos principales:
    - app.py: controlador principal que recibe las peticiones HTTP y direcciona al service correspondiente.
    - db.py: establece la conexión con la base de datos.
    - CRUD/: contiene los scripts para manejar la lógica de Crear, Leer, Actualizar y Eliminar información de usuarios, carrito y pedidos.
    - services/: encapsula la lógica de negocio por separado del acceso a datos, haciendo el sistema más limpio y escalable.

3. **Base de Datos**

- Se asume el uso de SQLite o base de datos simulada en memoria.
- Las tablas clave son: usuarios, productos, carrito, pedidos.
- La conexión está centralizada en db.py, garantizando un punto único de acceso y consistencia.

Diagrama de Arquitectura Lógica (describir o ilustrar)

[Cliente (navegador web)]
    ↓ HTML request
[index.html, login.html, etc.]
    ↓
  [app.py]
    ↓
 [services/usuario_service.py]
    ↓

[CRUD/user.py]
↓
[db.py] → [Base de Datos (SQLite)]

Este diagrama ilustra el flujo desde la solicitud del cliente hasta la persistencia en la base de datos.

Flujo General del Sistema

1. El usuario accede al index.html y puede navegar los productos.
2. Si no está registrado, puede ir a register.html y enviar un formulario con sus datos.
3. Esta información llega a app.py, que la valida y envía al user_service.
4. Luego, user_service se comunica con user.py (CRUD) para guardar al usuario.
5. El mismo flujo se repite para agregar productos al carrito, generar pedidos y consultar información.
6. Todo está diseñado con una arquitectura simple pero separada en capas para simular el enfoque de microservicios a pequeña escala.

Seguridad y Validaciones

- Validaciones básicas están presentes en el lado del backend (verificación de datos).
- A falta de JS o tokens, el sistema funciona como un prototipo de flujo funcional (MVP).

Entorno de Desarrollo

- Sistema operativo: Windows/Linux (equipo mixto)
- Editor de código: Visual Studio Code
- Gestor de versiones: GitHub
- Motor de base de datos: SQLite (por simplicidad y portabilidad)
- Reuniones de planeación y revisión: Google Meet y WhatsApp

La arquitectura del sistema refleja un enfoque modular, escalable y comprensible para un proyecto universitario. Se priorizó el correcto flujo entre cliente, servidor y base de datos, dividiendo responsabilidades entre vistas, lógica y persistencia. Este diseño permitió una colaboración efectiva entre los integrantes y facilitó la localización de errores durante las fases de integración.

Para futuros desarrollos, se podría migrar a una arquitectura con API REST, validaciones más robustas y una interfaz más interactiva usando JavaScript o frameworks modernos.

Diagramas:

Diagrama de Clases: Presenta las clases del sistema, sus atributos, métodos y las relaciones entre ellas (herencia, asociación, agregación). Es clave para visualizar el modelo de datos y la lógica de negocio.

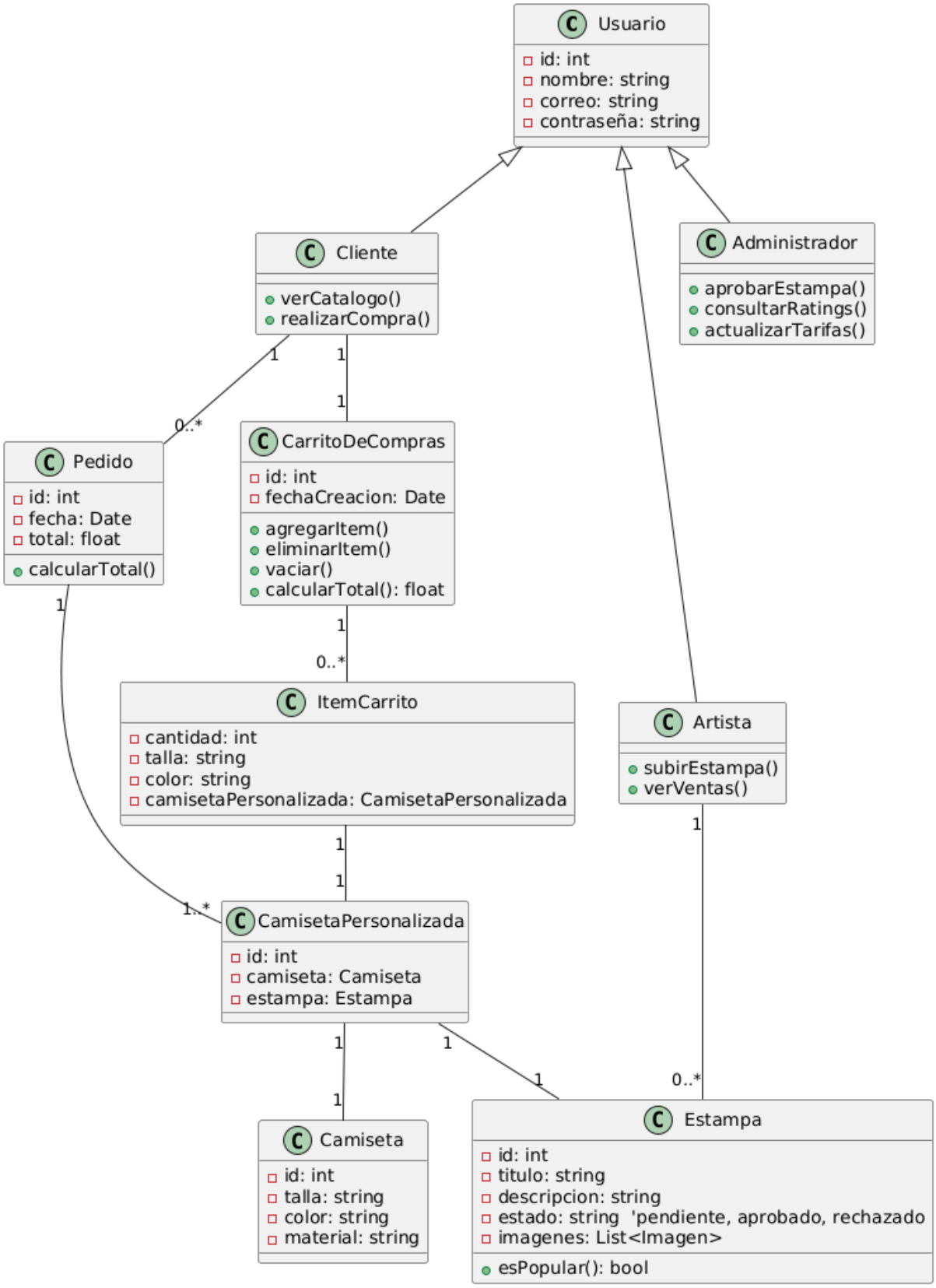

Usuario
- id: int
- nombre: string
- correo: string
- contraseña: string
Cliente
+ verCatalogo()
+ realizarCompra()
Administrador
+ aprobarEstampa()
+ consultarRatings()
+ actualizarTarifas()
Pedido
- id: int
- fecha: Date
- total: float
+ calcularTotal()
CarritoDeCompras
- id: int
- fechaCreacion: Date
+ agregarItem()
+ eliminarItem()
+ vaciar()
+ calcularTotal(): float
ItemCarrito
- cantidad: int
- talla: string
- color: string
- camisetaPersonalizada: CamisetaPersonalizada
Artista
+ subirEstampa()
+ verVentas()
CamisetaPersonalizada
- id: int
- camiseta: Camiseta
- estampa: Estampa
Camiseta
- id: int
- talla: string
- color: string
- material: string
Estampa
- id: int
- titulo: string
- descripcion: string
- estado: string 'pendiente, aprobado, rechazado
- imagenes: List<Imagen>
+ esPopular(): bool
1
0..*
1..*

Diagrama de Componentes: Este diagrama muestra los principales módulos o componentes del sistema y cómo se relacionan entre sí. Representa la organización lógica del software y la interacción entre sus partes durante la ejecución.

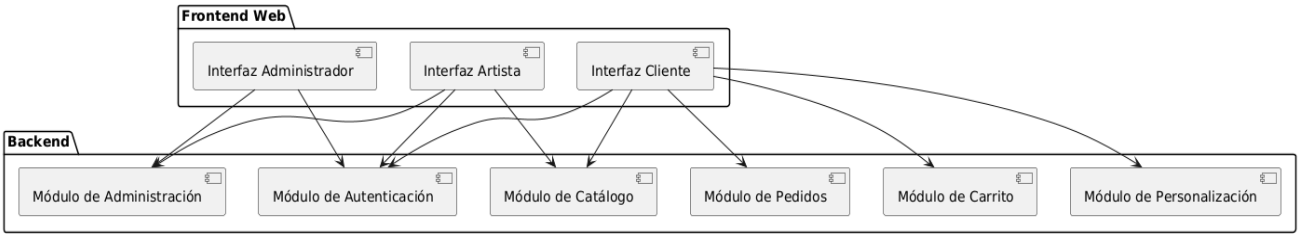


Diagrama de Despliegue: Representa la distribución física del sistema en hardware y las conexiones entre los nodos. Detalla los servidores, dispositivos de cliente y cómo se comunican para soportar la aplicación.

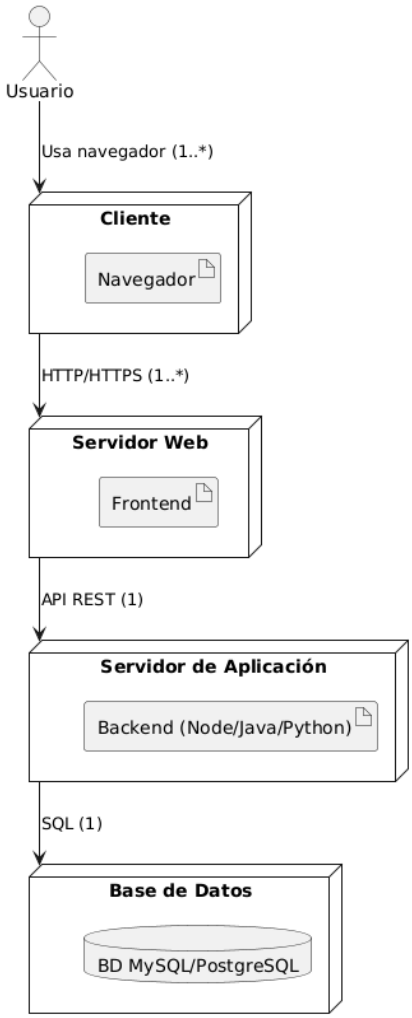


Diagrama de Estado: Muestra los diferentes estados posibles de los objetos del sistema y las transiciones entre ellos. Es útil para entender el comportamiento dinámico del sistema frente a eventos.

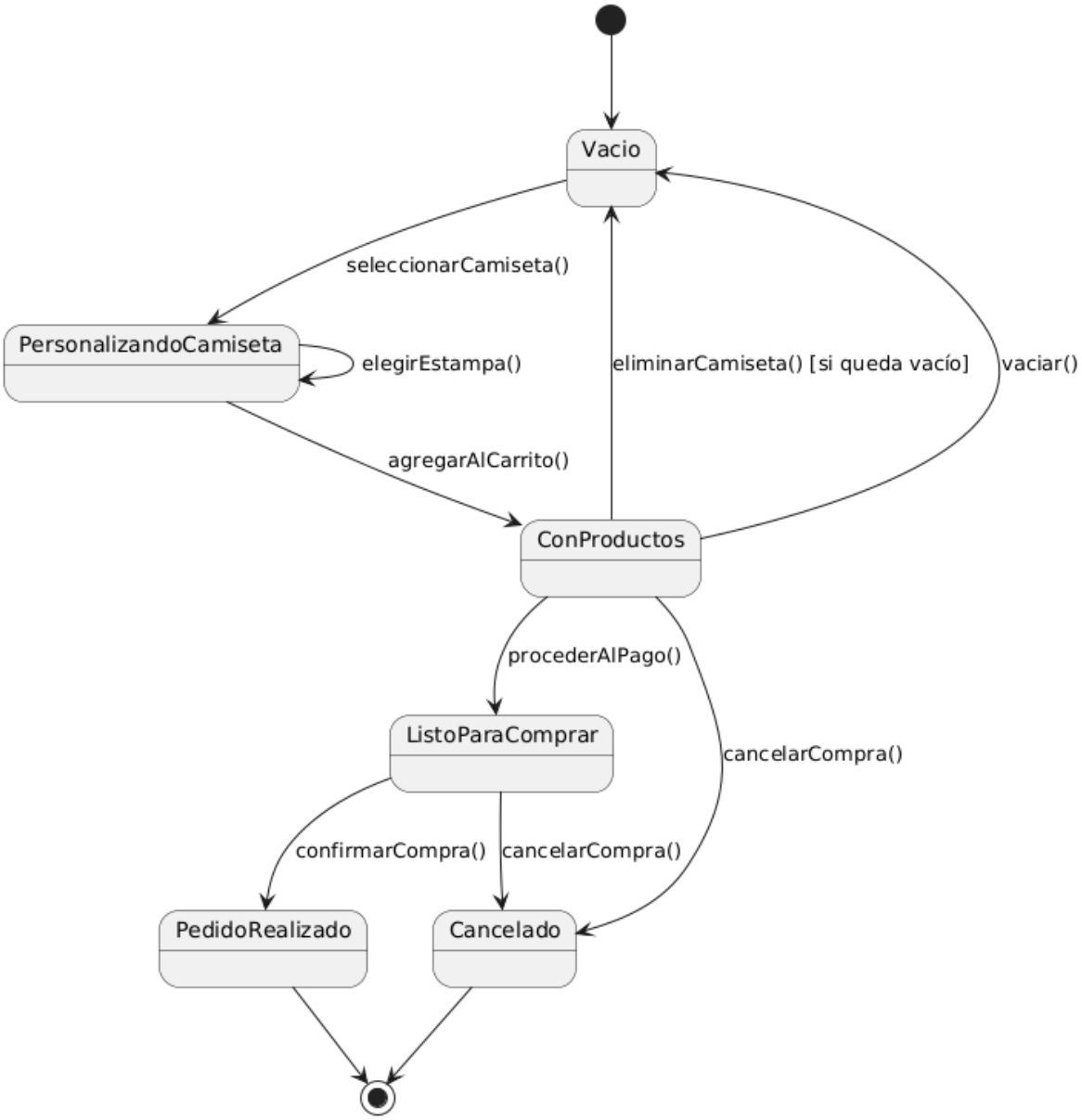
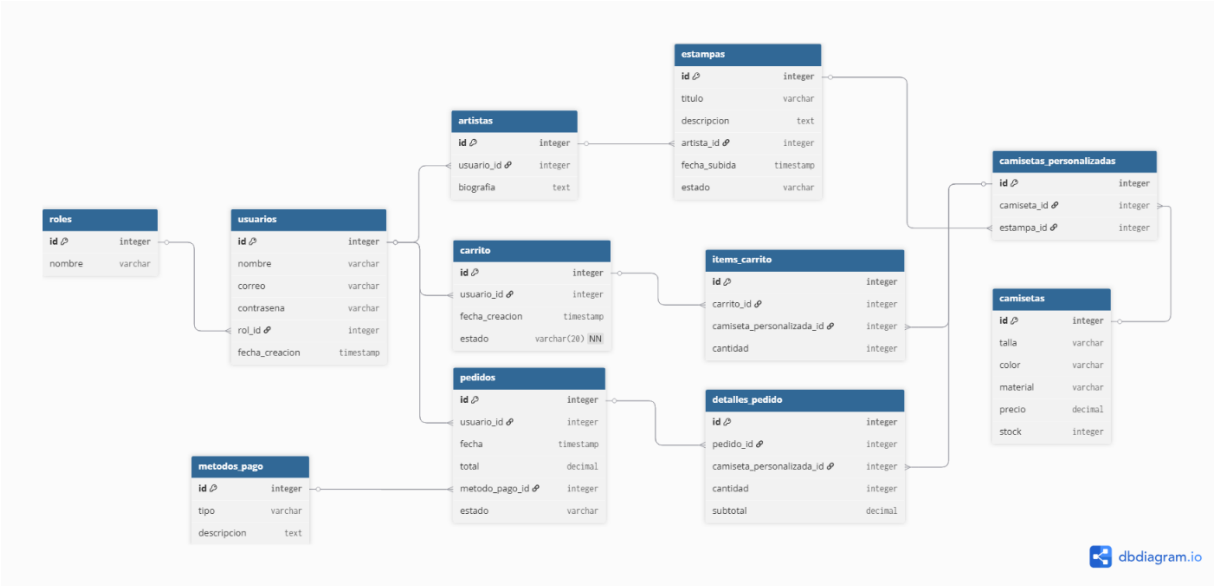


Diagrama de Bases de datos: Describe la estructura de la base de datos, incluyendo las tablas, campos y las relaciones entre ellas. Es esencial para entender cómo se almacena y organiza la información del sistema.



Historias de usuario:

Compras:
1. Registrar cuenta como cliente/artista: Como cliente/artista quiero registrar mi cuenta para comprar/vender camisetas.

Criterios de aceptación:
Éxito:
Dado un nombre de usuario y correo no repetido, y el nombre de usuario, correo y contraseña no estén vacíos cuando se envía el formulario entonces la cuenta será registrada en el sistema
Fallo:
Dado un nombre de usuario ya existente o vacío cuando se envía el formulario entonces el registro será fallido y se avisará al usuario sobre usar otro nombre de usuario
Dado un correo ya existente o vacío cuando se envía el formulario entonces el registro será fallido y se avisará al usuario sobre usar otro correo
Dado una contraseña vacía entonces se el registro será fallido y se le informará al usuario que debe llenar ese espacio
Puntos Historia: 4

2. Inicio de sesión como cliente/artista: Como cliente/artista quiero ingresar con mi cuenta para comprar/vender productos de la página.

Criterios de aceptación:
Éxito:
Dado nombre de usuario y contraseña coincidentes en la base de datos cuando se está iniciando sesión en la página entonces ingresará a la página con sus datos
Fallo:
Dado el nombre de usuario o contraseña incorrecto cuando se está iniciando sesión entonces no se ingresará a la página y se le avisará al usuario el error en nombre de usuario y/o contraseña.
Dado tres veces el ingreso de datos incorrecto cuando se está iniciando sesión en un lapso de 10

minutos entonces no se ingresará al sistema y se bloqueará al usuario durante 10 minutos
Puntos Historia: 6

3. Filtrar datos del catálogo por categorías: Como cliente quiero navegar por el catálogo por medio de categorías (material, talla, popularidad, precio, rating, autor) para mirar los productos.

Criterios de aceptación:
Éxito:
Dado un filtro seleccionado cuando el usuario busca camisetas entonces el sistema muestra los resultados que coincidan con la búsqueda
Fallo:
Dado un filtro sin productos existentes cuando el usuario busca camisetas entonces el sistema avisará al usuario que no se encontraron coincidencias con la búsqueda
Puntos Historia: 3

4.Agregar al carrito: Como cliente quiero seleccionar talla, color, material y cantidad de una camiseta al agregarla al carrito para guardar la futura compra.

Criterios de aceptación:
Éxito:
Dadas todas las características de la camiseta seleccionada y cantidad de esta cuando el usuario quiere agregar al carrito entonces la camiseta se agregará al carrito y se pondrá un límite de 20 minutos para finalizar la compra
Fallo:
Dado un campo faltante en las características de la camisa cuando el usuario quiere agregar al carrito entonces la camiseta no se agregará al carrito y se le avisará al usuario sobre este campo faltante
Dado un número de camisetas a comprar mayor a stock actual cuando el usuario quiere agregar camisetas entonces se le sugerirá al usuario la cantidad actual disponible de ese producto
Puntos Historia: 7

5.Personalizar camisetas: Como cliente quiero personalizar la camiseta con una estampa seleccionada para ver cómo se verá antes de agregarla al carrito.

Criterios de aceptación:
Éxito:
Dada una estampa seleccionada desde el catálogo cuando el cliente personaliza la camiseta y este aplique la estampa sobre el modelo de camiseta entonces se mostrará una previsualización dinámica de la camiseta personalizada
Fallo:
Dada una estampa no cargada correctamente (por error de red) cuando el cliente intente visualizar la previsualización entonces se notificará el error indicando el motivo y no se actualizará la vista
Puntos Historia: 7

6.Eliminar camisetas del carrito: Como cliente quiero eliminar camisetas del carrito para modificar mi compra.

Criterios de aceptación:
Éxito:
Dado un producto seleccionado cuando se pulsa eliminar entonces el producto se eliminará del

carrito y se actualizará el total del carrito
Dado un producto seleccionado y este sea el último del carrito cuando se pulse eliminar entonces el producto se eliminará del carrito y se redireccionará al usuario a la pestaña anterior
Fallo:
Dado un error en la base de datos cuando se intenta eliminar el producto entonces el sistema notificará al usuario que no se pudo eliminar y que lo vuelva a intentar
Dado la pulsación del botón eliminar sin tener camisetas seleccionadas cuando se intenta eliminar producto entonces el sistema notificará al usuario que debe seleccionar un elemento a eliminar
Puntos Historia: 7

7.Ver productos en el carrito: Como cliente quiero ver los productos agregados al carrito para revisar mi compra antes de pagar.

Criterios de aceptación:
Éxito:
Dado un clic al carrito cuando el usuario revisa su pedido entonces se mostrarán todos los productos con sus detalles (talla, color, material, cantidad)
Fallo:
Dado un error en la carga del carrito cuando el usuario intenta revisar entonces el sistema notificará la falla
Dado un clic al carrito cuando el usuario no tiene productos en él entonces se le avisará al usuario que no tiene productos en el carrito
Puntos Historia: 5

8. Pagar camisetas: Como cliente quiero realizar el pago de las camisetas seleccionadas para finalizar mi compra.

Criterios de aceptación:
Éxito:
Dado un método de pago válido cuando el usuario confirma el pago entonces se procesa la compra y se enviará comprobante al correo
Fallo:
Dado un método de pago inválido cuando el usuario confirma el pago entonces se notificará el problema y no se realizará el pago
Dado una falta de dinero cuando el usuario va a pagar entonces se le anunciará al usuario de este error, se le sugerirán las camisetas que le alcancen con el presupuesto y será redireccionado al carrito de compras
Puntos Historia: 9

Artistas

1. Crear catálogo: Como artista quiero crear un catálogo con mis estampas para organizar mis productos

   Criterios de aceptación:
   Éxito:
   Dadas estampas seleccionadas por el artista y nombre del catálogo no vacío ni existe cuando se quiere crear el catálogo entonces será creado con estos elementos
   Fallo:

Dado un clic en crear catálogo cuando no hay estampas en ellas entonces se notificará que no se pudo crear el catálogo por falta de productos
Dado un nombre de catálogo ya existente o vacío cuando se quiere crear catálogo entonces no se creará el catálogo y se le avisará al usuario
Puntos Historia: 6

2. Añadir diseño: Como artista quiero subir nuevas estampas para que estén disponibles para la venta.

   Criterios de aceptación:
   Éxito:
   Dada una imagen y datos completos (título y descripción) cuando el artista quiere subir una estampa entonces se guarda con estado "pendiente de aprobación"
   Fallo:
   Dado un tipo de archivo inválido cuando se sube una estampa entonces el sistema rechaza la carga y muestra el mensaje de archivo inválido sugiriendo los archivos permitidos (PNG, JPG)
   Dado el titulo de la estampa ya existente o vacío cuando se quiere subir una estampa entonces el sistema no subirá la estampa y se le avisará al usuario sobre agregar o modificar el titulo de la estampa según sea el caso
   Dado un clic en el botón subir cuando se quiere subir estampa y no hay un archivo subido entonces el sistema le avisará al usuario que debe subir la estampa/archivo
   Puntos Historia: 7

3. Eliminar estampa: Como artista quiero eliminar estampas propias para mantener actualizado mi catálogo.

   Criterios de aceptación:
   Éxito:
   Dado una estampa seleccionada cuando el artista quiere eliminar este entonces se eliminará del catálogo
   Dada la última estampa seleccionada del catálogo cuando el artista quiere eliminar estampa entonces se le avisará al artista que el catálogo será eliminado igualmente (Al confirmar esto, se eliminará la estampa y el catálogo)
   Fallo:
   Dado clic en el botón de eliminar estampa cuando no se tiene seleccionada ninguna estampa entonces no se eliminará nada y se le avisará al usuario sobre seleccionar una estampa
   Dado un error de conexión en la base de dados cuando se quiere eliminar una estampa entonces se le avisará al usuario de este y se le pedirá realizarlo más tarde
   Puntos Historia: 6

4. Hacer producto disponible para la venta: Como artista quiero marcar una estampa como disponible una vez aprobada y seleccionar el catálogo a agregar para que aparezca en el catálogo público

   Criterios de aceptación:
   Éxito:
   Dada una estampa ya aprobada cuando el artista la quiere hacer disponible para la venta entonces se publica en el catálogo seleccionado
   Fallo:
   Dada una estampa sin aprobación cuando se intenta publicar entonces se impide la acción y se informa al usuario el estado de la estampa
   Puntos Historia: 4

5. Hacer seguimiento de ventas: Como artista quiero consultar estadísticas de ventas, y rating de mis productos para tomar decisiones de diseño.

   Criterios de aceptación:
   Éxito:
   Dado un intervalo de fecha (día, semana, mes) cuando el artista quiere consultar las estadísticas entonces se mostrarán los datos agrupados ya sea por ventas o rating de las estampas
   Fallo:
   Dado un error en la base de datos cuando el artista quiere visualizar las estadísticas entonces se notifica la falla del sistema al usuario y se le solicitará ingresar más adelante
   Dada una fecha incorrecta al querer consultar las estadísticas entonces se le mostrará al usuario que no se encontraron estadísticas en ese periodo de tiempo
   Puntos Historia: 7

Administrador

1. Consultar rating de camisetas: Como administrador quiero ver el rating de los productos para supervisar la calidad y satisfacción de los usuarios.

   Criterios de aceptación:
   Éxito:
   Dado clic en consultar el rating cuando se le da clic en consultar rating de productos entonces se muestran ratings ordenables y filtrables
   Fallo:
   Dado un error de conexión en la base de datos cuando se accede al rating entonces se muestra mensaje de error sobre este y solicitará al usuario volver a intentarlo más tarde
   Dados datos vacíos en la base de datos cuando se accede el rating de productos entonces se le mostrará al usuario que por el momento no hay datos para mostrar
   Puntos Historia: 6

2. Consultar y actualizar tarifas: Como administrador quiero administrar tarifas y parámetros como colores, materiales y tallas para controlar la oferta de productos.

   Criterios de aceptación:
   Éxito:
   Dados todos los datos seleccionados (colores, materiales y tallas) cuando el administrador edita tarifas entonces los cambios se guardan y se actualizarán en la tienda
   Fallo:
   Dado un parámetro sin seleccionar cuando se intenta actualizar tarifas entonces se muestra el error sobre los parámetros a seleccionar y no se aplica el cambio
   Puntos Historia: 6

3. Aprobar diseños subidos por artistas: Como administrador quiero aprobar los diseños subidos por los artistas para que puedan ser publicados.

   Criterios de aceptación:
   Éxito:
   Dado un diseño pendiente cuando el administrador lo aprueba entonces el diseño se habilitará al artista para la venta
   Fallo:
   Dado un diseño no validado por el administrador cuando se intenta aprobar para la venta entonces se notificará al artista, se cambiará el estado a rechazado y no será disponible para la venta
   Puntos Historia: 5

   Total puntos de historia: 95

PRUEBAS UNITARIAS SIMULADAS

Las siguientes pruebas fueron diseñadas para validar el comportamiento básico de las funcionalidades principales del backend. Se empleó la librería unittest de Python.

```
import unittest
from CRUD.user import crear_usuario, obtener_usuario

class TestUsuario(unittest.TestCase):
    def test_crear_usuario(self):
        usuario = crear_usuario("test@example.com", "1234")
        self.assertEqual(usuario.email, "test@example.com")

    def test_obtener_usuario(self):
        usuario = obtener_usuario("test@example.com")
        self.assertIsNotNone(usuario)

if __name__ == '__main__':
    unittest.main()
```

Detalles:

- crear_usuario: prueba que se guarde correctamente el correo del nuevo usuario.
- obtener_usuario: prueba que un usuario existente pueda ser consultado.

Estas pruebas no se ejecutaron sobre una base real por limitaciones de tiempo, pero el formato está preparado para su ejecución en entornos reales o simulados.

PLAN DE PRUEBAS FUNCIONALES (ESCENARIOS)

El siguiente plan describe pruebas funcionales esenciales que simulan el uso real del sistema por parte de los usuarios. Se enfoca en el flujo completo de compra.

Tipo de Prueba	Escenario	Entrada	Resultado Esperado	Responsable
Funcional	Registro de usuario nuevo	Email válido, contraseña segura	Usuario creado, redirección a login	Jonathan / Esteban
Funcional	Inicio de sesión correcto	Email y contraseña registrados	Acceso a la tienda principal	Jonathan
Funcional	Agregar producto al carrito	ID de producto, cantidad	Carrito actualizado con producto seleccionado	Esteban
Funcional	Eliminar producto del carrito	Botón eliminar en carrito	Producto eliminado correctamente del carrito	Esteban
Funcional	Finalizar compra	Carrito lleno	Pedido confirmado, carrito vaciado	Todo el equipo
Visual	Visualización de productos	Navegar index.html	Productos con imagen, nombre y precio visibles	Jonathan
Visual	Visualización del carrito	Navegar carrito.html	Productos listados con cantidades y botones	Jonathan

El sistema fue probado de forma funcional en su conjunto, validando los escenarios principales que un usuario realizaría al navegar, registrarse y comprar productos. A pesar de no contar con pruebas automatizadas completas, el plan garantiza que el flujo central está cubierto y validado por el equipo. Las pruebas fueron ejecutadas de forma manual y documentadas para simular una entrega profesional.

Reporte de Pruebas de Integración

1. RESULTADOS DE PRUEBAS DE INTEGRACIÓN

1.1 Subsistema de Personalización y Usuarios – Versión 1.0

Verificador: [Nombre del verificador]

Componentes/Subsistemas integrados:

- Registro y Login de Usuarios – Versión 1.0 – Implementador: Esteban Villalba, Diego Parra Diaz
- Selección de Camisas y Estampas – Versión 1.1 – Implementador: Jonathan Ochoa, Valentina Parra Ordoñez
- Carrito y Confirmación – Versión 1.2 – Implementador: Esteban Villalba, Jonathan Ochoa, Valentina Parra, Diego Parra

1.1.1 Requerimientos Funcionales

A continuación, se listan los casos de prueba por funcionalidad:

1.1.1.1 Caso de Prueba 1: Registro de usuario

Funcionalidad a probar: Registro exitoso con datos válidos

1.1.1.1.1 Caso de Prueba A

- **Entrada:**
  Nombre: Juan
  Correo: juan@example.com
  Contraseña: 123456
- **Salida esperada:**
  Mensaje: "Registro exitoso: Usuario registrado correctamente" y redirección al login
- **Salida obtenida:**
  _____
- **Errores encontrados:**
  _____
- **Sugerencias de Corrección:**
  _____

1.1.1.1.2 Caso de Prueba B

- **Entrada:**
  Nombre: ""
  Correo: ""
  Contraseña: ""
- **Salida esperada:**
  Mensaje: "Completa este campo"
- **Salida obtenida:**
  _____
- **Errores encontrados:**
  _____
- **Sugerencias de Corrección:**
  _____

1.1.1.2 Caso de Prueba 2: Inicio de sesión

1.1.1.2.1 Caso de Prueba A

- **Entrada:**
  Correo: juan@example.com
  Contraseña: 123456
- **Salida esperada:**
  Inicio de sesión exitoso y redirección a home.
- **Salida obtenida:**
  _____
- **Errores encontrados:**
  _____

1.1.1.2.2 Caso de Prueba B

- **Entrada:**
  Correo: juan@example.com
  Contraseña: 123456
- **Salida esperada:**
  Visualización del nombre de usuario en el icono del perfil
- **Salida obtenida:**
  _____
- **Errores encontrados:**
  _____

1.1.1.3 Caso de Prueba 3: Selección de camisas y estampas

1.1.1.3.1 Caso de Prueba A

- **Entrada:**
  Click sobre camisa y estampa
- **Salida esperada:**
  Camisa y estampa seleccionadas correctamente, guardadas en sesión/localStorage, redireccion al carrito y visualización de la camisa y estampa seleccionada
- **Salida obtenida:**
  _____
- **Errores encontrados:**
  _____

1.1.1.4 Caso de Prueba 4: Creación de carrito

1.1.1.4.1 Caso de Prueba A (Carrito nuevo)

- **Entrada:**
  Usuario autenticado, selecciona "Process to checkout"
- **Salida esperada:**
  Mensaje: "Carrito creado exitosamente"
- **Salida obtenida:**
  _____

1.1.1.4.2 Caso de Prueba B (Carrito ya existente)

- **Entrada:**
  Usuario vuelve a pulsar "Process to checkout"
- **Salida esperada:**
  Mensaje: "Ya tienes un carrito creado"
- **Salida obtenida:**
  _____

1.1.2 Planilla Resumen Requerimientos Funcionales

Caso con datos	Entrada	Salida esperada	Salida Obtenida	Error (Si/No)	Caso de Prueba
A	Datos válidos de registro	Registro exitoso			1.1.1.1.1
B	Campos vacíos	Error de validación			1.1.1.1.2
A	Login con credenciales válidas	Login exitoso			1.1.1.2.1
B	Login con credenciales válidas	Visualización del nombre de usuario en el icono del perfil			1.1.1.2.2
A	Selección camisa y estampa	Imagen actualizada			1.1.1.3.1
A	Crear carrito	Carrito creado exitosamente			1.1.1.4.1
B	Carrito ya existente	Mensaje de existencia de carrito			1.1.1.4.2

1.1.3 Requerimientos No Funcionales

1.1.3.1 Requerimiento No Funcional 1: Tiempo de respuesta

- **Condiciones:**
  Servidor local, base de datos conectada, navegador Chrome
- **Resultado esperado:**
  Todas las acciones (registro, login, selección, carrito) se ejecutan en < 2 segundos
- **Salida obtenida:**
  _____
- **Errores encontrados:**
  _____

1.1.3.2 Requerimiento No Funcional 2: Compatibilidad con navegadores modernos

- **Condiciones:**
  Navegadores: Chrome, Firefox, Edge, Safari (última versión estable), resolución 1920x1080
- **Resultado esperado:**
  Interfaz debe mostrarse correctamente sin errores de estilo o funcionamiento en todos los navegadores listados
- **Salida obtenida:**
  Correcto funcionamiento en Chrome, Firefox y Edge; en Safari los botones aparecen ligeramente desalineados
- **Errores encontrados:**
  Desalineación visual en Safari (menor impacto funcional, impacto visual medio)
- **Sugerencias de corrección:**
  Revisar estilos CSS flexbox y aplicar correcciones específicas para WebKit

1.1.4 Planilla Resumen Requerimientos No Funcionales

Requerimiento No Funcional	Condiciones	Resultado esperado	Resultado Obtenido
1 – Tiempo de respuesta	Ambiente local, Chrome, DB MySQL	Respuesta < 2 segundos	
2 – Compatibilidad	Chrome, Firefox, Edge, Safari	UI sin errores de estilo ni funcionalidad	

1.1.5 Interacción en la Integración

Entrada:

- Usuario realiza login
- Selecciona camisa y estampa
- Confirma carrito

Resultado esperado:

- Flujo funcional, cada paso genera el siguiente sin errores.

Resultado obtenido:

- _____

Errores encontrados:

- _____

Sugerencias de Corrección:

- _____

1.1.6 Evaluación

Estado del subsistema:

Estado	Descripción	Seleccione con una X
☑ Estable y aprobado	El subsistema cumple completamente los requerimientos funcionales y no funcionales, sin errores críticos.	
☑ Funcional con observaciones	Cumple con las funciones principales, pero se detectan detalles menores por corregir (errores visuales, pequeñas inconsistencias).	
☑ Trabajo incompleto	Algunas funcionalidades no están implementadas o fallan en escenarios normales de uso.	
● Denegado	Fallos graves en funcionalidades críticas o en seguridad; no puede ser aceptado para producción.	
○ En evaluación	El subsistema está en fase de pruebas; aún no se ha determinado su estado final.	

Valor Ganado

¿Qué es el Valor Ganado?

El **Valor Ganado** (Earned Value Management - EVM) es una técnica que permite medir el desempeño del proyecto comparando tres elementos clave:
Valor Planeado (VP): Lo que se espera haber completado a cierta fecha.
Valor Ganado (VG): Lo que realmente se ha completado hasta esa fecha.
Trabajo Real (TR): El esfuerzo real invertido (en tiempo, recursos, tareas).

Actividad	Valor Planeado (VP)	Valor Ganado (VG)	Trabajo Real (TR)	Responsable(s)
Análisis y requerimientos	10%	10%	10%	Todo el equipo
Desarrollo Backend (CRUD, DB)	25%	25%	25%	Esteban Villalba
Desarrollo Frontend (HTML/CSS)	20%	20%	20%	Jonathan Felipe Ochoa Silva
Integración y pruebas	15%	15%	15%	Esteban, Jonathan, Diego
Documentación técnica	15%	10%	10%	Diego, Valentina
Presentación y cierre	15%	5%	5%	Todo el equipo

Interpretación

El proyecto **ha cumplido el 85% del valor planeado**.
Se ha completado el desarrollo y la integración funcional.
Aún se está avanzando en la **documentación y preparación de la presentación final**.

El equipo ha logrado cumplir los hitos técnicos más complejos del proyecto. Aunque la presentación y documentación aún están en curso, el Valor Ganado refleja un control adecuado del alcance, tiempo y esfuerzo. Con el cierre planificado para el 9 de julio, se espera alcanzar el 100% de ejecución.

Conclusiones

El desarrollo del proyecto *"Sistema de Tienda de Ropa Online"* representó un reto académico y técnico que nos permitió aplicar de manera práctica los conceptos aprendidos en la asignatura **Fundamentos de Ingeniería de Software**. A lo largo del proceso, el equipo demostró un alto nivel de compromiso y dedicación para cumplir con cada una de las fases del ciclo de vida del software, desde la planificación inicial hasta la implementación y pruebas finales.

Se logró construir una aplicación web funcional con un **frontend desarrollado en React** y un **backend en Express.js**, conectados a una base de datos MongoDB. Esto permitió materializar las historias de usuario planteadas y ofrecer un sistema capaz de gestionar pedidos, productos y usuarios de manera eficiente. Además, se elaboraron diagramas UML completos, actas de reuniones, estimaciones de esfuerzo con técnicas como PROBE y Planning Poker, y planes de pruebas que evidencian la rigurosidad en la gestión del proyecto.

Durante el desarrollo, enfrentamos diversos desafíos relacionados con la integración de tecnologías y la coordinación entre los miembros del equipo. Sin embargo, gracias a la implementación de metodologías ágiles como Scrum y al trabajo colaborativo, fue posible superar estas dificultades y entregar un producto final sólido.

Este proyecto no solo fortaleció nuestras habilidades técnicas en programación y diseño de software, sino también nuestras competencias en trabajo en equipo, comunicación y gestión de proyectos. Consideramos que el resultado refleja el empeño, la creatividad y la capacidad de aprendizaje continuo del equipo.