

**“AÑO DEL BICENTENARIO, DE LA CONSOLIDACIÓN DE NUESTRA  
INDEPENDENCIA, Y DE LA CONMEMORACIÓN DE LAS HEROICAS  
BATALLAS DE JUNÍN Y AYACUCHO”**



**Avance de Proyecto Final 3**

**Sistema de Comercio digital para la floristería “El y Ella Detalles” en Java**

Trabajo que como parte del curso Integrador I: Sistemas – Software presentan los  
alumnos

**DOCENTE:**

Mg. Ing. Edwin Palomino Iriarte.

**INTEGRANTES:**

- |                                       |                |
|---------------------------------------|----------------|
| 1. Sandro Cadmo De La Cruz Asencios.  | Cod. U22202494 |
| 2. Joaquín Alfonso Loa Denegri.       | Cod. U22234069 |
| 3. Jorge Brandy Membrillo Gaitán.     | Cod. U22229151 |
| 4. Jefferson Eudes Pernia De La Cruz. | Cod. U22220722 |
| 5. Kiara Mishell Santi Saavedra.      | Cod. U22201636 |

**Lima, 28 de octubre del 2024**

<b>Índice</b>	
<b>1    Introducción .....</b>	1
1.1    Objetivos del proyecto.....	3
1.1.1    Objetivo General .....	3
1.1.2    Objetivos específicos.....	3
1.2    Alcance del proyecto.....	3
1.3    Objetivos del aplicativo.....	4
1.4    Limitaciones y restricciones.....	4
<b>2    Análisis del contexto .....</b>	6
2.1    Descripción de la empresa.....	6
2.2    Visión y misión de la empresa .....	6
2.3    Análisis del entorno.....	7
2.4    Estrategias y planes de la empresa.....	7
2.5    Impacto en los procesos de la empresa.....	8
2.6    Lienzo del modelo de negocio .....	9
2.7    Identificación y descripción del problema.....	10
2.8    Definición del alcance de la solución .....	10
<b>3    Planteamiento de alternativas de solución .....</b>	13
3.1    Alternativa 1 .....	13
3.1.1    Descripción general.....	13
3.1.2    Pantallas propuestas .....	13
3.1.3    Tecnologías por utilizar.....	17
3.2    Alternativa 2 .....	17
3.2.1    Descripción general.....	17
3.2.2    Pantallas propuestas .....	18
3.2.3    Tecnologías por utilizar.....	21
3.3    Alternativa 3 .....	21
3.3.1    Descripción general.....	21
3.3.2    Pantallas propuestas .....	21
3.3.3    Tecnologías por utilizar.....	25
<b>4    Análisis de la solución .....</b>	25
4.1    Levantamiento de información .....	25
4.2    Matriz de requerimientos .....	26
4.2.1    Requisitos funcionales .....	26
4.2.2    Requisitos no funcionales .....	29
4.2.3    Especificación de Requerimientos de Software (SRS) .....	32
<b>5    Diseño de la solución .....</b>	35
5.1    Arquitectura del proyecto.....	35
5.1.1    Aplicación del patrón MVC (Modelo, Vista, Controlador).....	36
5.1.2    Implementación de DAO (Data Access Object) para gestión de datos.....	39
5.1.3    Pruebas orientadas a TDD (Test-Driven Development) .....	40
5.1.3.1    Principios de TDD .....	41
5.1.3.2    Mejora de la Seguridad del Proyecto .....	44
5.1.3.3    Uso de JUnit para manejo de errores y validación de métodos .....	45
5.1.3.3.1    Implementación de Pruebas en Detalle_PedidoDAO.....	45
5.1.3.3.2    Identificación de Problemas con Junit .....	45

5.1.3.3.3	Error Inicial Detectado.....	46
5.1.3.3.4	Solución Implementada.....	46
5.1.3.3.5	Resultados Finales de la Prueba .....	47
5.1.4	Operaciones CRUD.....	48
5.1.4.1	Método update() para Actualización de Registros.....	48
5.1.4.2	Método get() para Recuperación de Registros.....	49
5.1.4.3	Método eliminar() para Eliminación de Registros.....	49
5.1.4.4	Método agregar() para Inserción de Registros.....	50
5.1.5	Medidas de seguridad integradas.....	50
5.1.5.1	Medidas de Seguridad contra Inyección SQL .....	50
<b>6</b>	<b>Uso de Recursos Java .....</b>	<b>51</b>
6.1	Librerías utilizadas en el proyecto .....	51
6.1.1	Google Guava – Optimización de estructuras de datos .....	52
6.1.1.1	guava-33.2.1-jre.jar.....	52
6.1.2	Apache POI – Manipulación de archivos Excel.....	53
6.1.2.1	poi-5.2.3.jar .....	53
6.1.3	Apache Commons – Funciones comunes para manipulación de datos .....	53
6.1.3.1	commons-io-2.16.1.jar.....	53
6.1.4	Logback – Sistema de registro de logs .....	53
6.1.4.1	logback-classic-1.5.6.jar .....	54
<b>7</b>	<b>Uso de Control de Versiones .....</b>	<b>54</b>
7.1	Estructura del Repositorio .....	54
7.2	Integración de GitHub con NetBeans .....	56
7.3	Flujo de Trabajo Colaborativo con Ramas Individuales.....	56
7.4	Documentación y Trazabilidad de Cambios.....	57
7.5	Evidencia de Uso .....	58
7.5.1	División del Trabajo en el Proyecto .....	59
<b>4.</b>	<b>Kiara Santti - Implementación del Modelo DAO .....</b>	<b>62</b>
<b>8</b>	<b>Implementación de las Interfaces Gráficas .....</b>	<b>63</b>
8.1	Diseño UX/UI del sistema.....	63
8.1.1	Interfaces gráficas.....	63
8.1.1.1	Página de Inicio.....	64
8.1.1.2	Formulario de Inicio de Sesión .....	64
8.1.1.3	Formulario de Registro.....	65
8.1.1.4	Catálogo de Productos .....	66
8.1.1.5	Detalle del Producto .....	66
8.1.1.6	Formulario de Reseñas .....	67
8.1.1.7	Perfil de Usuario .....	68
8.1.1.8	Carrito de Compras.....	68
8.1.1.9	Formulario de Pago.....	69
8.1.1.10	Pedidos .....	70
8.1.1.11	Detalle de Pedido .....	70
8.1.1.12	Interfaz Administrativa .....	71
8.1.1.12.1	Usuarios Registrados.....	72
8.1.1.12.2	Flores Disponibles .....	73
8.1.1.12.3	Carrito de Compras.....	73

8.1.1.12.4	Historial de Pedidos.....	74
8.1.1.12.5	Detalles de Pedidos.....	74
8.1.1.12.6	Categorías .....	75
8.1.1.12.7	Relación entre Flores y Categorías .....	75
8.1.1.12.8	Comentarios de Usuarios .....	76
8.1.1.12.9	Pagos Realizados .....	76
8.1.1.13	Ejemplo de Administración de Tabla.....	76
<b>9</b>	<b>Construcción del Producto Final.....</b>	<b>77</b>
9.1	Alcance cubierto en el avance.....	78
9.1.1	Gestión de Inventario .....	78
9.1.2	Carrito de Compras .....	79
9.1.3	Historial de Pedidos .....	80
9.1.4	Registro y Gestión de Usuarios.....	81
9.1.5	Comentarios de Usuarios sobre Productos .....	82
9.1.6	Pagos Realizados .....	83
9.1.7	Administración de Categorías y Relación con Flores .....	84
9.1.8	Formulario de Pago y Procesamiento de Compras .....	85
9.2	Coherencia entre documentación y código.....	85
9.2.1	Informe Técnico: Decisiones de Diseño del Proyecto .....	86
9.2.2	Revisión de Pull Request en GitHub .....	99
9.3	Buenas prácticas aplicadas en el desarrollo.....	101
9.3.1	Uso de librerías adecuadas .....	101
9.3.2	Uso de patrones de diseño.....	103
9.3.3	Software de control de versiones .....	106
9.4	Autoría del proyecto y dominio del código .....	108
9.4.1	Justificación de las elecciones de diseño .....	108
9.4.2	Evidencia de autoría y conocimiento del sistema .....	108
<b>10</b>	<b>Conclusiones.....</b>	<b>109</b>
<b>11</b>	<b>Recomendaciones.....</b>	<b>110</b>
<b>12</b>	<b>Referencias .....</b>	<b>111</b>
<b>13</b>	<b>Anexos .....</b>	<b>113</b>
13.1	Evidencias del trabajo en la empresa .....	113
13.2	Otros documentos relevantes .....	115
13.2.1	Formato de una encuesta .....	115
13.2.2	Formato de una entrevista. ....	119
13.2.3	Listado de requerimientos funcionales del proyecto .....	126
13.2.4	Listado de requerimientos no funcionales del proyecto .....	132
13.2.5	Procedimiento de Gestión de Inventario - Buena Práctica de Documentación.....	137
13.3	Diagramas de Arquitectura.....	149
13.4	Documentación de Código .....	151

## Índice de figuras

<b>Figura 1:</b> Lean Canvas del proyecto.....	9
<b>Figura 2:</b> Página de inicio de la alternativa de solución 1 .....	14
<b>Figura 3:</b> Catálogo de productos para la alternativa de solución 1 .....	14
<b>Figura 4:</b> Detalle de productos para la alternativa de solución 1 .....	15
<b>Figura 5:</b> Formulario de acceso a la plataforma digital de la floristería para la alternativa de solución 1 .....	16
<b>Figura 6:</b> Carrito de compras para la alternativa de solución 1 .....	16
<b>Figura 7:</b> Formulario de pago para la alternativa de solución 1 .....	17
<b>Figura 8:</b> Página de inicio de la alternativa de solución 2 .....	18
<b>Figura 9:</b> Formulario para la personalización de arreglos florales de la alternativa de solución 2.....	19
<b>Figura 10:</b> Página de testimonios de clientes potenciales de la alternativa de solución 2.....	19
<b>Figura 11:</b> Apartado de plantillas para inspirar en la personalización de arreglos florales de los usuarios en la alternativa de solución 2 .....	20
<b>Figura 12:</b> Formulario de contacto de la alternativa de solución 2 .....	21
<b>Figura 13:</b> Página de inicio de la alternativa de solución 3 .....	22
<b>Figura 14:</b> Información detallada necesaria sobre la suscripción en la floristería para la alternativa de solución 3 .....	22
<b>Figura 15:</b> Apartado de los tipos de suscripción en la floristería para la alternativa de solución 3.....	23
<b>Figura 16:</b> Formulario para agregar los detalles de envío de arreglos florales para la alternativa de solución 3 .....	24
<b>Figura 17:</b> Formulario de pago para la alternativa de solución 3.....	24
<b>Figura 18:</b> Estructura del Proyecto del Sistema de Comercio Digital para la Floristería "El y Ella Detalles".....	37
<b>Figura 19:</b> Organización de las Clases del Proyecto en Paquetes Java .....	38
<b>Figura 20:</b> Clases del Paquete Modelo.dao.....	40
<b>Figura 21:</b> Prueba Unitaria con Mockito para CarritoDAO .....	42
<b>Figura 22:</b> Implementación del Método getCarritosById() en CarritoDAO .....	43
<b>Figura 23:</b> Implementación Mejorada del Método getCarritosById() en CarritoDAO .....	44
<b>Figura 24:</b> Pruebas unitarias de errores con JUnit .....	46
<b>Figura 25:</b> Resultado de Ejecución de Pruebas Unitarias en JUnit .....	47
<b>Figura 26:</b> Implementación del Método update() en FloresDAO .....	48
<b>Figura 27:</b> Implementación del Método get() en FloresDAO.....	49
<b>Figura 28:</b> Implementación del Método eliminar() en FloresDAO.....	49
<b>Figura 29:</b> Implementación del Método agregar() en FloresDAO .....	50
<b>Figura 30:</b> Implementación del Método save() en RegisterDAO .....	51
<b>Figura 31:</b> Bibliotecas Importadas en el Proyecto .....	52
<b>Figura 32:</b> Organización del Repositorio en GitHub.....	55
<b>Figura 33:</b> Archivos del Primer Avance del Proyecto en GitHub.....	55
<b>Figura 34:</b> Archivos del Segundo Avance en GitHub .....	56
<b>Figura 35:</b> Historial de commits en repositorio colaborativo .....	58
<b>Figura 36:</b> Detalle de Commit: 'Detalles de pedidos, productos y edición form .....	59
<b>Figura 37:</b> Commit: Implementación del paquete servicio y clase de conexión .....	60
<b>Figura 38:</b> Commit: Implementación de interfaces de usuario y estilos .....	61
<b>Figura 39:</b> Commit: Implementación del modelo DTO .....	61
<b>Figura 40:</b> Commit: Implementación del modelo DAO .....	62
<b>Figura 41:</b> Commit: Implementación del Controlador .....	63
<b>Figura 42:</b> Página de Inicio del Sistema de Comercio Digital.....	64
<b>Figura 43:</b> Página de Inicio de Sesión y Registro del Usuario .....	65
<b>Figura 44:</b> Formulario de Registro en "El y Ella Detalles" .....	65
<b>Figura 45:</b> Visualización de Categorías y Productos en "El y Ella Detalles" .....	66
<b>Figura 46:</b> Detalle de Producto en "El y Ella Detalles".....	67
<b>Figura 47:</b> Área de Comentarios y Footer en "El y Ella Detalles".....	67
<b>Figura 48:</b> Perfil de Usuario en "El y Ella Detalles" .....	68
<b>Figura 49:</b> Vista del Carrito de Compras en "El y Ella Detalles" .....	69
<b>Figura 50:</b> Vista del Formulario de Pago en "El y Ella Detalles" .....	69
<b>Figura 51:</b> Vista del Historial de Pedidos en "El y Ella Detalles" .....	70
<b>Figura 52:</b> Vista de Detalle de un Pedido en "El y Ella Detalles" .....	71
<b>Figura 53:</b> Interfaz para la Gestión de Tablas en "El y Ella Detalles" .....	72

<b>Figura 54:</b> Administración de Usuarios en "El y Ella Detalles" .....	72
<b>Figura 55:</b> Administración de Flores en "El y Ella Detalles" .....	73
<b>Figura 56:</b> Administración del Carrito de Compras en "El y Ella Detalles" .....	73
<b>Figura 57:</b> Administración del Historial de Pedidos en "El y Ella Detalles" .....	74
<b>Figura 58:</b> Administración de los Detalles de Pedidos en "El y Ella Detalles" .....	74
<b>Figura 59:</b> Administración de Categorías en "El y Ella Detalles" .....	75
<b>Figura 60:</b> Administración de la Relación entre Flores y Categorías en "El y Ella Detalles" .....	75
<b>Figura 61:</b> Administración de Comentarios en "El y Ella Detalles" .....	76
<b>Figura 62:</b> Administración de Pagos en "El y Ella Detalles" .....	76
<b>Figura 63:</b> Administración de Usuarios en "El y Ella Detalles" .....	77
<b>Figura 64:</b> Gestión de Flores Disponibles .....	78
<b>Figura 65:</b> Relación entre Flores y Categorías.....	79
<b>Figura 66:</b> Carrito de Compras.....	79
<b>Figura 67:</b> Formulario de Pago .....	80
<b>Figura 68:</b> Detalles de Pedidos.....	81
<b>Figura 69:</b> Gestión de Categorías.....	81
<b>Figura 70:</b> Usuarios Registrados .....	82
<b>Figura 71:</b> Comentarios de Usuarios .....	83
<b>Figura 72:</b> Pagos Realizados .....	83
<b>Figura 73:</b> Gestión de Categorías.....	84
<b>Figura 74:</b> Relación entre Flores y Categorías.....	84
<b>Figura 75:</b> Formulario de Pago .....	85
<b>Figura 76:</b> Implementación del método save para registro de usuarios .....	87
<b>Figura 77:</b> Método get para obtener usuario por ID.....	87
<b>Figura 78:</b> Método update para actualizar usuario .....	88
<b>Figura 79:</b> Método eliminar para eliminar un usuario por ID.....	88
<b>Figura 80:</b> Método get para obtener un pedido por ID.....	89
<b>Figura 81:</b> Método update para actualizar un pedido .....	89
<b>Figura 82:</b> Método eliminar para borrar un pedido por ID .....	90
<b>Figura 83:</b> Método agregar para insertar flores en la base de datos.....	90
<b>Figura 84:</b> Método get para obtener una flor por ID .....	91
<b>Figura 85:</b> Método update para actualizar una flor existente.....	91
<b>Figura 86:</b> Método eliminar para borrar una flor por ID .....	92
<b>Figura 87:</b> Modelo Relacional de la Base de Datos del Sistema .....	92
<b>Figura 88:</b> Creación de la Tabla "Register" (Usuarios Registrados).....	93
<b>Figura 89:</b> Creación de la Tabla "Flores" (Productos Disponibles).....	94
<b>Figura 90:</b> Creación de la Tabla "Carrito" (Productos en el Carrito).....	95
<b>Figura 91:</b> Creación de la Tabla "Pedidos" (Historial de Pedidos).....	95
<b>Figura 92:</b> Creación de la Tabla "Detalle Pedido" (Detalle de cada pedido) .....	96
<b>Figura 93:</b> Creación de la Tabla "Pago" (Información de pago).....	97
<b>Figura 94:</b> Creación de la Tabla "Comentarios" (Comentarios y valoraciones de productos) .....	98
<b>Figura 95:</b> Estado anterior del archivo carshop.html .....	99
<b>Figura 96:</b> Estado modificado del archivo carshop.jsp .....	100
<b>Figura 97:</b> Implementación final del archivo carshop.jsp .....	100
<b>Figura 98:</b> Librerías utilizadas en el proyecto.....	102
<b>Figura 99:</b> Estructura del Proyecto Java - Capa Modelo .....	104
<b>Figura 100:</b> Estructura del Proyecto - Capa Vista.....	105
<b>Figura 101:</b> Estructura del Proyecto - Capa Controlador .....	106
<b>Figura 102:</b> Árbol de avances en GitHub .....	107

## **1 Introducción**

El comercio electrónico se ha convertido en un componente crucial para el éxito empresarial a nivel global. Según eMarketer, las ventas minoristas de comercio electrónico en todo el mundo alcanzaron los 4,9 billones de dólares en 2021, con proyecciones de superar los 7,4 billones de dólares para 2025. Este crecimiento exponencial ha sido impulsado por la creciente penetración de internet, la adopción de dispositivos móviles y los cambios en los hábitos de consumo, acelerados por la pandemia de COVID-19.

En Perú, el panorama del comercio electrónico ha evolucionado rápidamente. La Cámara Peruana de Comercio Electrónico (CAPECE) reporta que el comercio electrónico en el país creció un 50% en 2020, alcanzando los 6 mil millones de dólares. Este crecimiento ha sido impulsado por una mayor confianza de los consumidores en las compras en línea y por la necesidad de las empresas de adaptarse a las nuevas demandas del mercado.

En el contexto local de Lima, específicamente en el distrito de Carabayllo, las pequeñas empresas como las floristerías se enfrentan al desafío de adaptarse a esta nueva realidad digital. La floristería "El y Ella Detalles" ha identificado la necesidad de modernizar sus operaciones y ampliar su alcance mediante la implementación de un sistema de comercio digital. La problemática central radica en cómo una floristería tradicional puede hacer la transición al mundo digital de manera efectiva, manteniendo la calidad de sus productos y servicios. La pregunta que guía este proyecto es: ¿Cómo puede un sistema de comercio digital mejorar las operaciones y ampliar el alcance de la floristería "El y Ella Detalles"? La hipótesis es que la implementación de un sistema de comercio digital aumentará las ventas, mejorará la eficiencia operativa y ampliará la base de clientes de la floristería. El objetivo principal es desarrollar un Sistema de Comercio Digital para la floristería "El y Ella Detalles" utilizando tecnología Java.

Para abordar este desafío, el presente trabajo se ha estructurado en varias secciones clave. Comienza con un análisis del contexto, donde se describe detalladamente la empresa "El y Ella Detalles", su visión, misión y el problema identificado. Luego, se presentan tres alternativas de solución, cada una con sus respectivas pantallas y tecnologías propuestas. Se realiza un análisis exhaustivo de los requerimientos, tanto funcionales como no funcionales. El diseño de la solución se aborda de manera integral, incluyendo un Project Charter, diagramas de procesos, un diagrama de clases y un diseño detallado de la base de datos. Se presenta el diseño del prototipo, que incluye maquetas de la interfaz de usuario y una descripción detallada de las funcionalidades principales. Finalmente, se abordan aspectos como el diseño de la interfaz de usuario (UX/UI) y se proporciona una guía de implementación para desarrolladores. El trabajo concluye con recomendaciones y anexos que incluyen evidencias del trabajo realizado en la empresa.

Este avance del proyecto cubre aspectos esenciales, desde el análisis del contexto hasta la implementación de las primeras funcionalidades. Se detalla la estructura organizacional de la empresa, sus valores y objetivos, así como el problema detectado. Además, se incluyen los avances en la arquitectura del sistema siguiendo el patrón MVC, el diseño de la base de datos, los diagramas de procesos, y las interfaces de usuario preliminares.

Se han utilizado librerías clave como **JSTL**, **MySQL Connector**, y **JUnit**, además de herramientas como **GitHub** para control de versiones, lo que garantiza una gestión eficiente y colaborativa del desarrollo. Este avance también muestra las revisiones realizadas mediante Pull Requests en GitHub y las pruebas iniciales para asegurar la calidad del sistema. El trabajo concluye con un análisis del progreso logrado, identificando los próximos pasos a seguir para completar el desarrollo y garantizar el éxito de la implementación.

## 1.1 Objetivos del proyecto

### 1.1.1 Objetivo General

Desarrollar un Sistema de Comercio Digital para la floristería "El y Ella Detalles" utilizando tecnología Java.

### 1.1.2 Objetivos específicos

1. Diseñar e implementar una plataforma de comercio electrónico que permita a los clientes navegar, personalizar y comprar productos florales y detalles en línea.
2. Crear una interfaz de usuario intuitiva y atractiva que refleje la identidad de la marca y mejore la experiencia del cliente.
3. Implementar un sistema de pedidos y pagos en línea seguro y eficiente.
4. Asegurar que al menos el 50% del desarrollo del sistema se realice utilizando tecnología Java.
5. Mejorar la visibilidad en línea de "El y Ella Detalles" y aumentar su base de clientes.
6. Incrementar las ventas y la eficiencia operativa de la floristería mediante la digitalización de sus procesos de venta y gestión.

## 1.2 Alcance del proyecto

El alcance del proyecto define los límites y expectativas del sistema de comercio digital para "El y Ella Detalles". Esta sección detalla los objetivos específicos que el aplicativo busca alcanzar, así como las limitaciones y restricciones que deben considerarse durante su desarrollo e implementación. La clara definición del alcance es crucial para asegurar que el proyecto cumpla con las necesidades de la floristería y sus clientes, mientras se mantiene dentro de los parámetros establecidos de tiempo, presupuesto y recursos.

### 1.3 Objetivos del aplicativo

Los objetivos del aplicativo representan las metas concretas que el sistema de comercio digital debe lograr para considerarse exitoso. Estos objetivos están alineados con la visión general del proyecto y las necesidades específicas identificadas en el análisis del contexto de "El y Ella Detalles":

1. Desarrollar una plataforma de comercio electrónico intuitiva y atractiva que permita a los clientes navegar, personalizar y comprar arreglos florales en línea.
2. Establecer un sistema de procesamiento de pagos seguro y eficiente que soporte múltiples métodos de pago.
3. Implementar herramientas de análisis de datos para obtener insights sobre el comportamiento del cliente y las tendencias de compra.
4. Integrar la plataforma digital con los procesos existentes de la floristería física para una operación unificada.

### 1.4 Limitaciones y restricciones

Las limitaciones y restricciones del proyecto definen los factores que pueden afectar el desarrollo, implementación o funcionamiento del sistema de comercio digital. Es esencial identificar y comprender estas limitaciones para gestionar eficazmente los riesgos y expectativas del proyecto.

A continuación, a través de la siguiente tabla se describen dichas limitaciones y/o restricciones:

**Tabla 1:**  
*Limitaciones y restricciones del proyecto*

<b>Limitación/Restricción</b>	<b>Descripción</b>
Presupuesto	El desarrollo e implementación del sistema deben ajustarse al presupuesto asignado por "El y Ella Detalles".

Tiempo	El proyecto debe completarse dentro del plazo establecido para coincidir con la temporada alta de ventas de la floristería.
Recursos humanos	La disponibilidad limitada de personal técnico especializado en la empresa puede afectar el desarrollo y mantenimiento del sistema.
Infraestructura tecnológica	La solución debe ser compatible con la infraestructura de TI existente en la floristería.
Regulaciones	El sistema debe cumplir con las normativas locales de comercio electrónico y protección de datos personales.
Integración de sistemas	La nueva plataforma debe integrarse sin problemas con los sistemas de gestión existentes de la floristería.
Capacitación	Se debe considerar el tiempo y recursos necesarios para capacitar al personal en el uso del nuevo sistema.
Escalabilidad	El sistema debe diseñarse para manejar un aumento significativo en el volumen de transacciones durante períodos pico.
Conectividad	La dependencia de una conexión a internet estable puede afectar la operatividad del sistema en ciertas áreas geográficas.
Mantenimiento	Se deben considerar los requisitos de mantenimiento continuo y actualizaciones del sistema a largo plazo.

**Nota:** Esta tabla proporciona una visión general de las principales limitaciones y restricciones que el equipo de proyecto deberá tener en cuenta durante todas las fases del desarrollo e implementación del Sistema de Comercio Digital para "El y Ella Detalles". **Fuente:** Elaboración Propia.

## **2 Análisis del contexto**

Para comprender a fondo la situación actual de la floristería "El y Ella Detalles S.A.C." y el contexto en el que opera, es fundamental realizar un análisis exhaustivo de diversos aspectos de la empresa y su entorno. Este análisis permitirá identificar las fortalezas, debilidades, oportunidades y amenazas que enfrenta la floristería, sentando así las bases para el desarrollo de una solución de comercio digital efectiva y adaptada a sus necesidades específicas.

### **2.1 Descripción de la empresa**

Para entender el alcance del proyecto, es esencial conocer la historia y características principales de la empresa en cuestión.

"El y Ella Detalles S.A.C." es una floristería local fundada en 2022 por Melody Loa Denegri en el distrito de Carabayllo, Lima, Perú. La empresa se especializa en la creación y venta de arreglos florales personalizados para diversas ocasiones, como bodas, cumpleaños, aniversarios y eventos corporativos. A pesar de su joven trayectoria, la floristería ha logrado establecer una sólida reputación basada en la calidad de sus productos y el servicio personalizado que ofrece a sus clientes.

### **2.2 Visión y misión de la empresa**

La visión y misión de una empresa definen su dirección y propósito. En el caso de "El y Ella Detalles S.A.C.", estas declaraciones reflejan sus aspiraciones y valores fundamentales.

**Visión:** Ser la floristería líder en Lima, reconocida por la innovación en diseños florales y la excelencia en el servicio al cliente, expandiendo su presencia en el mercado digital.

**Misión:** Ofrecer arreglos florales y detalles personalizados de alta calidad que transmitan emociones y embellezcan los momentos especiales de los clientes, utilizando tecnología para mejorar la experiencia de compra y la eficiencia operativa.

## 2.3 Análisis del entorno

Comprender el entorno en el que opera la empresa es crucial para identificar oportunidades y amenazas. Esta sección examina las tendencias y cambios en el sector florístico de Lima.

El sector florístico en Lima ha experimentado cambios significativos en los últimos años. La creciente demanda de servicios personalizados y la tendencia hacia las compras en línea han transformado el mercado. La competencia se ha intensificado con la entrada de nuevos actores que ofrecen servicios de entrega rápida y opciones de personalización en línea. Además, la pandemia de COVID-19 ha acelerado la adopción del comercio electrónico en el sector.

## 2.4 Estrategias y planes de la empresa

Las estrategias y planes de una empresa reflejan cómo pretende adaptarse a los cambios del entorno y lograr sus objetivos. "Él y Ella Detalles S.A.C." ha desarrollado las siguientes estrategias:

**Tabla 2:**  
Estrategias y planes de la empresa

ESTRATEGIAS Y PLANES		
Descripción	Explicación	Justificación
Digitalización de operaciones de venta y gestión	La empresa planea implementar sistemas digitales para gestionar ventas, inventario y procesos administrativos.	Esto permitirá a "El y Ella Detalles" mejorar la eficiencia operativa, reducir errores y tener un mejor control sobre sus procesos internos.
Desarrollo de una plataforma de comercio electrónico	Se creará una tienda en línea donde los clientes puedan ver, personalizar y comprar productos florales.	Esta estrategia ampliará el alcance de la floristería, permitiéndole llegar a nuevos clientes y ofrecer servicios 24/7, aumentando así las oportunidades de venta.
Implementación de un sistema de personalización en línea	Se desarrollará una herramienta que permita a los clientes diseñar sus propios arreglos florales en línea.	Esta función diferenciará a "El y Ella Detalles" de la competencia, ofreciendo una experiencia única al cliente y potencialmente aumentando el valor promedio de las compras.
Mejora de los procesos de logística y entrega	Se optimizarán las rutas de entrega y se implementará un sistema de seguimiento en tiempo real para los pedidos.	Esto mejorará la satisfacción del cliente al garantizar entregas puntuales y permitir a los clientes rastrear sus pedidos, lo que puede resultar en una mayor fidelización.
Capacitación del personal en habilidades digitales y servicio al cliente en línea	Se proporcionará formación al personal en el uso de nuevas tecnologías y en la atención al cliente a través de canales digitales.	Esta estrategia asegurará una transición suave hacia el modelo de negocio digital y mantendrá la calidad del servicio al cliente, que ha sido una fortaleza de la empresa.

*Nota: Estas estrategias están diseñadas para trabajar en conjunto, permitiendo a "El y Ella Detalles" transformar su modelo de negocio y adaptarse a las demandas del mercado digital actual, manteniendo al mismo tiempo la calidad y el servicio personalizado que han caracterizado a la floristería.*

*Fuente: Elaboración Propia.*

## 2.5 Impacto en los procesos de la empresa

Es importante entender cómo el problema identificado afecta a los diferentes procesos de la empresa. Esta sección detalla esos impactos:

La falta de un sistema de comercio digital afecta varios procesos clave de la empresa:

**Tabla 3:**  
Procesos afectados por la problemática identificada

Proceso	Impacto
Ventas	Limita el alcance geográfico y temporal de las ventas.
Gestión de inventario	Dificulta el control preciso del stock y la previsión de demanda.

Atención al cliente	Restringe la capacidad de atender consultas y pedidos fuera del horario de atención física.
Personalización de productos	Limita las opciones de personalización que se pueden ofrecer a los clientes.
Logística y entrega	Reduce la eficiencia en la planificación y ejecución de entregas.
Marketing	Limita la capacidad de realizar campañas personalizadas y medir su efectividad.

**Nota:** La implementación de un sistema de comercio digital se presenta como una solución integral para abordar estos desafíos y mejorar la competitividad de "El y Ella Detalles" en el mercado actual.

**Fuente:** Elaboración Propia.

## 2.6 Lienzo del modelo de negocio

El Lean Canvas que se presenta a continuación refleja la visión estratégica de este proyecto, detallando cómo se abordarán los desafíos actuales de la floristería y cómo se aprovechará la tecnología para crear una ventaja competitiva en el mercado digital de flores y regalos personalizados.

**Figura 1:**  
Lean Canvas del proyecto



**Nota:** Este Lean Canvas refleja el modelo de negocio para el sistema de comercio digital de "El y Ella Detalles", abordando los principales problemas identificados y proponiendo soluciones que aprovechan las oportunidades del mercado digital. **Fuente:** Elaboración Propia.

## 2.7 Identificación y descripción del problema

La identificación precisa del problema es esencial para desarrollar soluciones efectivas. En el caso de "El y Ella Detalles", el principal desafío es el siguiente:

El principal problema identificado es la limitación del modelo de negocio actual para competir en el mercado digital y satisfacer las nuevas demandas de los clientes. La falta de presencia en línea y de un sistema de comercio electrónico eficiente ha resultado en:

1. Pérdida de potenciales clientes que prefieren comprar en línea.
2. Dificultades para gestionar el inventario y los pedidos de manera eficiente.
3. Limitaciones en la capacidad de personalización de productos.
4. Procesos de venta y entrega menos eficientes en comparación con competidores digitalizados.

## 2.8 Definición del alcance de la solución

El Sistema de Comercio Digital para la floristería "El y Ella Detalles" se ha diseñado con un alcance específico para abordar las necesidades identificadas y facilitar la transición de la empresa hacia el comercio electrónico. A continuación, se detallan los aspectos clave del alcance de la solución:

1. Funcionalidades principales:
  - Catálogo en línea: Presentación digital de todos los productos florales y accesorios ofrecidos por la floristería.
  - Sistema de carrito de compras: Permitirá a los clientes seleccionar y gestionar sus pedidos antes de la compra final.
  - Procesamiento de pagos en línea: Integración con pasarelas de pago seguras para transacciones electrónicas.

- Gestión de pedidos: Interfaz para que el personal de la floristería administre y procese los pedidos recibidos.
  - Seguimiento de pedidos: Funcionalidad para que los clientes puedan rastrear el estado de sus compras.
  - Gestión de inventario: Módulo para controlar y actualizar el stock de productos en tiempo real.
2. Usuarios del sistema:
- Clientes: Podrán navegar por el catálogo, realizar compras y dar seguimiento a sus pedidos.
  - Administradores: Tendrán acceso completo para gestionar productos, pedidos e inventario.
  - Floristas: Podrán ver y procesar pedidos, actualizando su estado.
  - Repartidores: Accederán a la información de entrega y actualizarán el estado de los pedidos.
3. Procesos de negocio abarcados:
- Venta en línea: Digitalización completa del proceso de venta, desde la selección del producto hasta el pago.
  - Gestión de inventario: Automatización del control de stock, incluyendo alertas de reposición.
  - Procesamiento de pedidos: Flujo digital desde la recepción del pedido hasta su entrega.
  - Atención al cliente: Sistema de mensajería integrado para comunicación con los clientes.

4. Resultados esperados:

- Aumento en el volumen de ventas a través del canal digital.
- Mejora en la eficiencia operativa de la floristería.
- Expansión de la base de clientes más allá del área geográfica inmediata.
- Incremento en la satisfacción del cliente mediante un servicio más rápido y conveniente.

5. Aspectos fuera del alcance:

- Diseño floral personalizado en línea: Esta función quedará fuera del alcance inicial.
- Integración con sistemas de proveedores: No se incluirá en esta fase del proyecto.
- Gestión de recursos humanos: Se mantendrá el sistema actual de la floristería.

Relación con el problema identificado:

El alcance definido aborda directamente los desafíos identificados para "El y Ella Detalles":

1. Limitación geográfica: El sistema de comercio digital permitirá a la floristería alcanzar clientes más allá de su ubicación física.
2. Ineficiencias operativas: La automatización de procesos como la gestión de inventario y el procesamiento de pedidos mejorará significativamente la eficiencia operativa.
3. Competencia digital: Al proporcionar una plataforma de comercio electrónico robusta, la floristería podrá competir efectivamente en el mercado digital.
4. Experiencia del cliente: Las funcionalidades de seguimiento de pedidos y atención al cliente mejorarán la experiencia general de compra.

5. Adaptación a nuevas tendencias de consumo: El sistema permitirá a la floristería satisfacer la creciente demanda de compras en línea y servicios digitales.

### **3 Planteamiento de alternativas de solución**

El análisis de múltiples alternativas permite a la empresa evaluar diferentes enfoques para abordar sus necesidades de digitalización, considerando diversos aspectos técnicos, funcionales y estratégicos. Cada propuesta alternativa ofrece una visión única de cómo se puede implementar el sistema, permitiendo a los responsables de la toma de decisiones comparar y contrastar las opciones disponibles. A continuación, se presentan tres alternativas distintas, cada una detallada con su descripción, pantallas propuestas y tecnologías a utilizar.

#### **3.1 Alternativa 1**

##### **Plataforma de E-commerce Integrada**

###### **3.1.1 Descripción general**

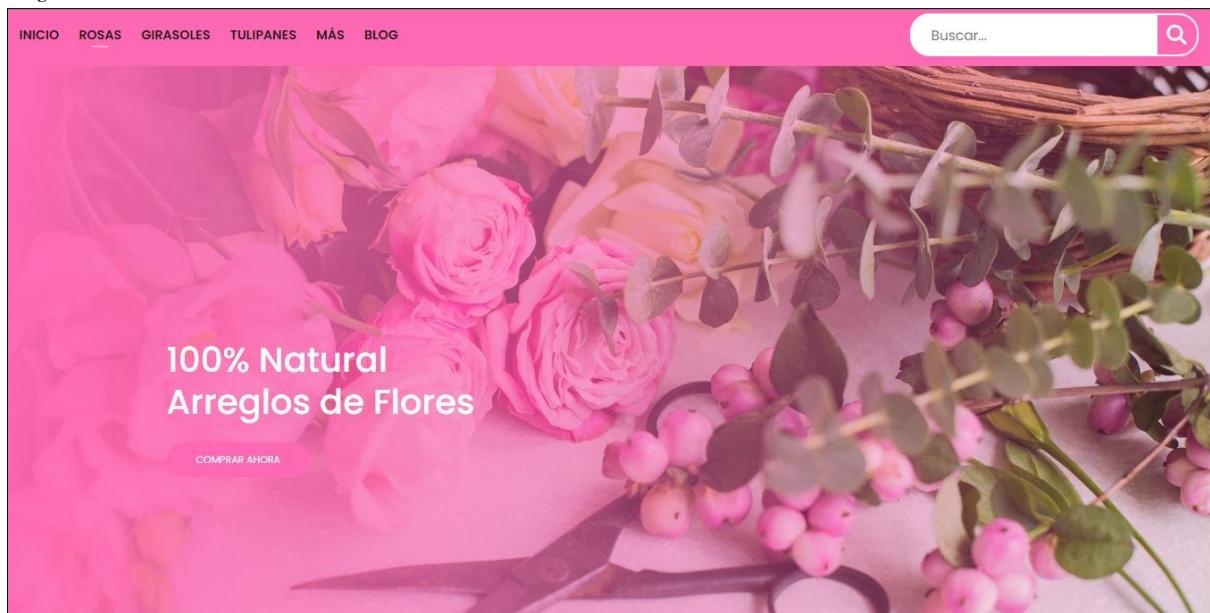
Desarrollo de una página web completa de e-commerce con un catálogo digital, carrito de compras, y procesamiento de pagos online.

###### **3.1.2 Pantallas propuestas**

- 1. Página de inicio:** La página de inicio es la puerta de entrada a la "Floristería". Presenta una visión general de la tienda, con imágenes atractivas de los productos más populares, promociones destacadas, y enlaces rápidos a las secciones más importantes del sitio, como el catálogo de productos y el carrito de compras. Esta pantalla también puede incluir una breve presentación sobre la filosofía de la tienda, destacando la calidad y frescura de sus flores.

**Figura 2:**

Página de inicio de la alternativa de solución 1



**Nota:** La página de inicio presenta un diseño atractivo y femenino, dominado por tonos rosados que complementan perfectamente la imagen de fondo de un arreglo floral. **Fuente:** Elaboración Propia.

**2. Catálogo de productos:** El catálogo de productos muestra todas las flores y arreglos disponibles para la venta, organizados en categorías. Los usuarios pueden filtrar los productos por precio, tipo de flor, o disponibilidad, y cada ítem incluye una miniatura con la opción de hacer clic para ver más detalles. Esta pantalla es clave para ofrecer una experiencia de compra intuitiva y eficiente.

**Figura 3:**

Catálogo de productos para la alternativa de solución 1

Producto	Imagen	Oferta	Precio
Box Tulipanes		-13%	\$50
Box Rosas Gigante		-22%	\$200
Box Amor			\$120
Box Girasoles			\$135

**Nota:** Esta sección del catálogo muestra una presentación efectiva de los productos destacados, con un diseño limpio y atractivo. Las imágenes de alta calidad de los arreglos florales son el punto focal, acompañadas de información clara sobre precios y descuentos. **Fuente:** Elaboración Propia.

**3. Detalle de producto:** Al hacer clic en un producto del catálogo, los usuarios son llevados a la pantalla de detalle de producto, donde pueden ver imágenes en alta resolución, una descripción completa del artículo, opciones de personalización (como seleccionar colores o añadir una tarjeta), y el precio. También se incluyen reseñas de otros clientes y la opción de añadir el producto al carrito de compras.

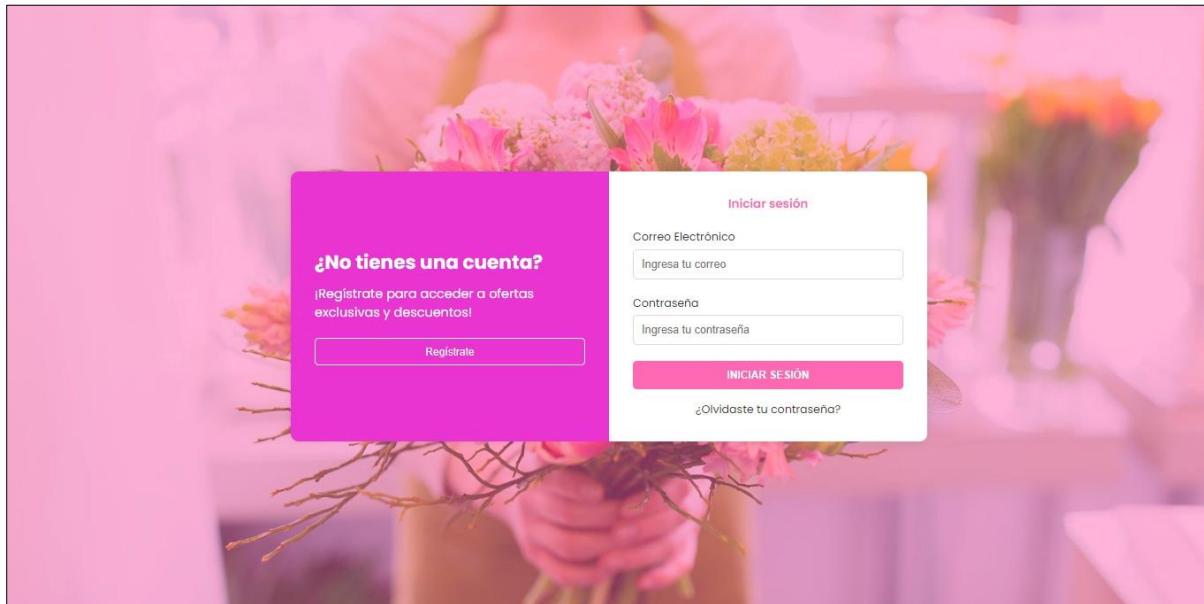
**Figura 4:**  
Detalle de productos para la alternativa de solución 1

The screenshot shows a product detail page for a sunflower arrangement. At the top, there's a navigation bar with links for 'INICIO', 'ROSAS', 'GIRASOLES', 'TULIPANES', 'MÁS', and 'BLOG'. A search bar is also present. The main content area features a large image of a sunflower arrangement in a pink box, with three smaller images below it showing different angles. To the right of the image, the product title 'Caja de Girasoles' is displayed, along with its price '\$135' and a '20% OFF' discount. Below this, there's a list of product details: '12 Girasoles', 'Follaje de temporada', 'Topper Te amo', 'Feliz dí', 'Tarjeta dedicatoria', 'Box Corazón', and 'Colores Disponibles'. A dropdown menu for quantity is set to '2', and a 'Añadir al carrito' button is visible. Further down, sections for 'Información de Entrega' and 'Políticas de Devoluciones' provide shipping and return information.

**Nota:** La página de detalle del producto "Caja de Girasoles" está bien estructurada y ofrece información completa. La imagen principal del producto es grande y clara, con miniaturas adicionales para ver diferentes ángulos. El precio y el descuento son prominentes, y la calificación de los clientes añade credibilidad. **Fuente:** Elaboración Propia.

**4. Login:** La pantalla de login permite a los usuarios registrarse o iniciar sesión en sus cuentas. Esto es fundamental para gestionar pedidos, guardar preferencias de compra, y acceder a beneficios exclusivos como descuentos o el seguimiento de pedidos. Los usuarios pueden ingresar utilizando su correo electrónico y contraseña, o iniciar sesión a través de plataformas de terceros como Google o Facebook.

**Figura 5:**  
Formulario de acceso a la plataforma digital de la floristería para la alternativa de solución 1



**Nota:** La pantalla de inicio de sesión está bien diseñada, combinando funcionalidad con atractivo visual. El fondo con imágenes de flores en tonos rosados crea una atmósfera acogedora y coherente con la temática de la floristería.

**Fuente:** Elaboración Propia.

**5. Carrito de compras:** El carrito de compras es donde los usuarios pueden revisar los productos que han seleccionado antes de proceder al pago. Incluye un resumen de los artículos, cantidades, precios unitarios y total acumulado. Los usuarios pueden actualizar las cantidades, eliminar productos, o guardar el carrito para una compra posterior. Además, el carrito proporciona enlaces rápidos para volver al catálogo o proceder al pago, facilitando una navegación fluida.

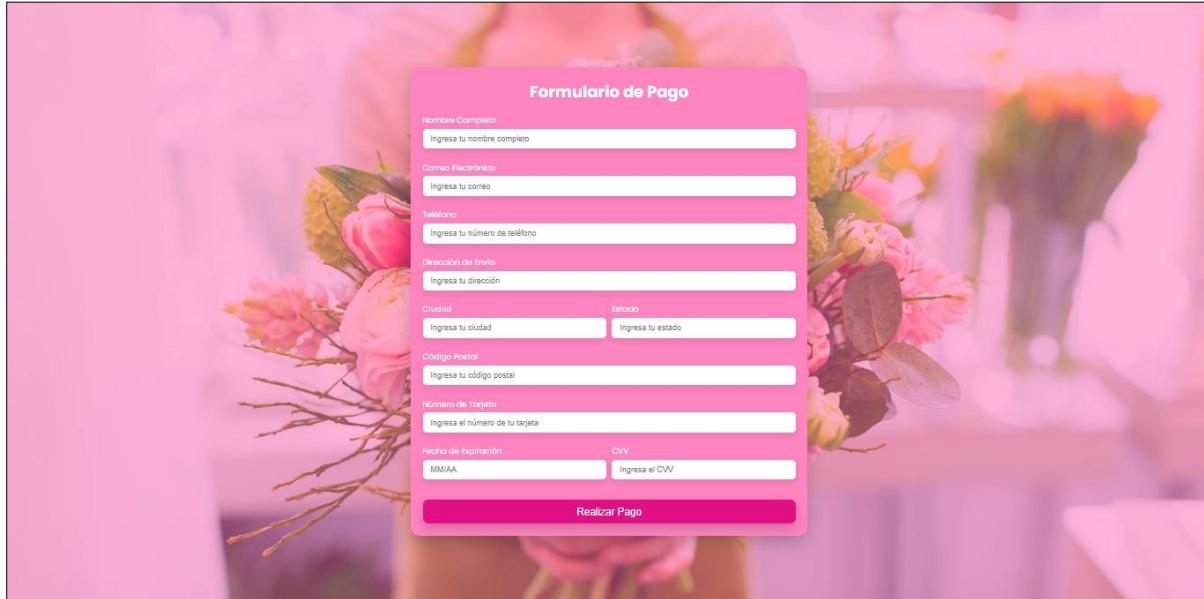
**Figura 6:**  
Carrito de compras para la alternativa de solución 1

**Nota:** La página del carrito de compras está bien organizada, mostrando claramente el producto seleccionado (Box Girasoles) con su imagen, cantidad y precio. La sección de resumen del pedido a la derecha ofrece un desglose claro del subtotal, gastos de envío y total. El botón "Realizar compra" es prominente y fácil de localizar.

**Fuente:** Elaboración Propia.

**6. Procesamiento de pago:** Una vez que los usuarios han terminado de seleccionar sus productos, son dirigidos a la pantalla de procesamiento de pago. Aquí ingresan su información de envío y eligen un método de pago.

**Figura 7:**  
Formulario de pago para la alternativa de solución 1



*Nota: El formulario de pago está bien estructurado y presenta un diseño limpio y coherente con la estética floral del sitio. Los campos requeridos están claramente definidos, cubriendo toda la información necesaria para el envío y el pago. La disposición en una sola columna facilita el llenado secuencial de la información.*

*Fuente: Elaboración Propia.*

### 3.1.3 Tecnologías por utilizar

Java será utilizado en la gestión del backend para manejar las bases de datos, procesos de pago y la lógica de negocios. Frontend desarrollado en HTML, CSS y JavaScript.

## 3.2 Alternativa 2

### Aplicación Web con Gestión de Pedidos Personalizados

#### 3.2.1 Descripción general

Desarrollo de una aplicación web que permite a los usuarios crear arreglos florales personalizados y gestionar sus pedidos desde la plataforma.

### 3.2.2 Pantallas propuestas

**1. Página de inicio:** La página de inicio introduce a los usuarios a la posibilidad de personalizar arreglos florales. Muestra ejemplos visuales de arreglos personalizados, testimonios de clientes satisfechos, y enlaces directos al constructor de arreglos y otras secciones clave. También destaca la facilidad de uso de la plataforma y la calidad de los productos ofrecidos, invitando a los usuarios a explorar y crear sus propios diseños.

*Figura 8:  
Página de inicio de la alternativa de solución 2*

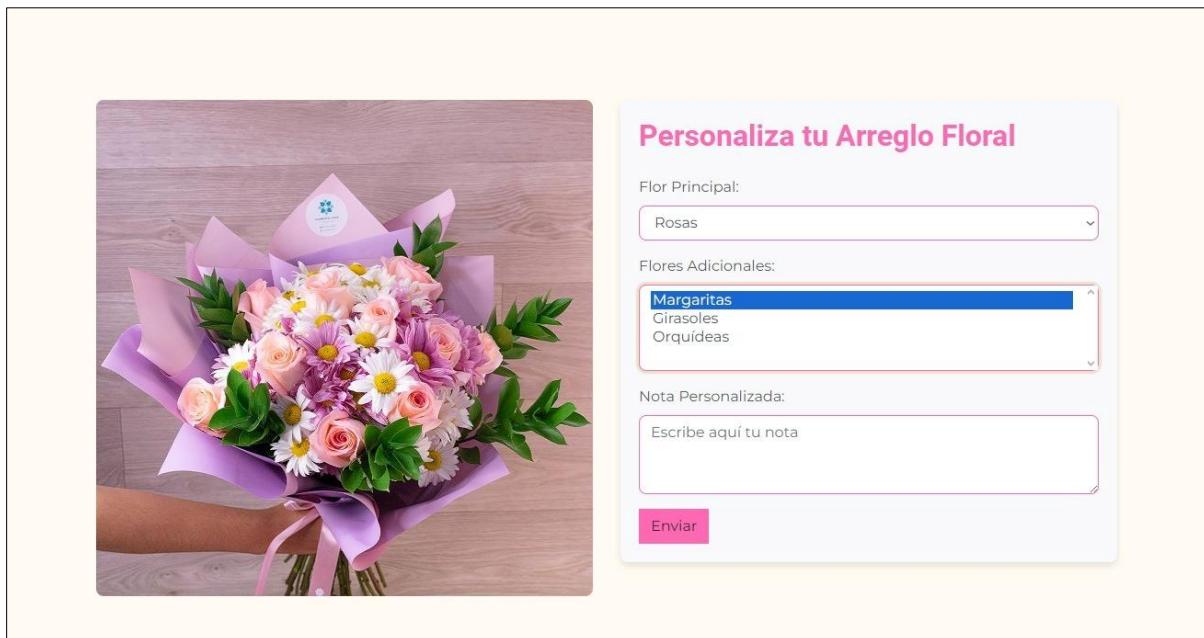


*Nota: Esta primera pantalla resalta las funcionalidades de la página de inicio para "El y Ella Detalles.*

*Fuente: Elaboración Propia.*

**2. Constructor de arreglos:** El constructor de arreglos es una herramienta interactiva que permite a los usuarios seleccionar flores, colores, y estilos para crear un arreglo floral personalizado. La pantalla ofrece una vista previa en tiempo real del arreglo mientras los usuarios realizan sus elecciones, proporcionando opciones de personalización detalladas como el tipo de envoltura, el tamaño del ramo, y mensajes personalizados.

**Figura 9:**  
Formulario para la personalización de arreglos florales de la alternativa de solución 2



**Nota:** A través de este formulario, el usuario podrá personalizar sus arreglos florales y agregar notas personalizadas.

**Fuente:** Elaboración Propia.

**3. Testimonio de clientes:** Esta pantalla muestra testimonios de clientes que han utilizado la plataforma para personalizar y comprar arreglos florales. Incluye reseñas, calificaciones, y fotografías de los arreglos adquiridos, proporcionando una visión confiable de la satisfacción del cliente. Es una sección que refuerza la credibilidad y calidad del servicio.

**Figura 10:**  
Página de testimonios de clientes potenciales de la alternativa de solución 2



**Nota:** Esta página de testimonios es fundamental para captar más clientes y continuar con el crecimiento en el mercado de la floristería. **Fuente:** Elaboración Propia.

**4. Plantillas de arreglos:** Para aquellos que necesitan inspiración, esta pantalla ofrece una variedad de plantillas predefinidas de arreglos florales. Los usuarios pueden elegir una plantilla y luego personalizarla aún más, lo que facilita el proceso de creación para usuarios que prefieren comenzar con un diseño base.

**Figura 11:**

Apartado de plantillas para inspirar en la personalización de arreglos florales de los usuarios en la alternativa de solución 2

INSPIRÁTE CON NUESTRAS PLANTILLAS

## Comienza tu personalización

**Plantilla 1**



**Elegante**

Una opción sofisticada para ocasiones especiales, con una combinación de flores clásicas y elegantes.

**Plantilla 3**



**Romántica**

Perfecta para expresar tus sentimientos más profundos, con flores delicadas y románticas.

**Plantilla 2**



**Colorida**

Un diseño vibrante y alegre, ideal para celebrar momentos felices y llenos de color.

**Plantilla 4**



**Moderna**

Un estilo contemporáneo y fresco, ideal para quienes buscan algo diferente y original.

*Nota: Al implementar estas plantillas, será posible que los usuarios tengan una idea más detallada de sus pedidos.*

*Fuente: Elaboración Propia.*

**5. Formulario de atención al cliente:** Esta pantalla proporciona un formulario donde los usuarios pueden enviar consultas, solicitar asistencia, o realizar pedidos especiales. El formulario incluye campos para ingresar el nombre, correo electrónico, tipo de consulta, y un mensaje detallado. Además, puede ofrecer opciones de contacto directo como chat en vivo o llamadas.

**Figura 12:**

Formulario de contacto de la alternativa de solución 2

**ATENCIÓN AL CLIENTE**  
**No dudes en contactarnos**

  
**Dirección**  
Av.Caudivila 455 Carabayllo  
15318

  
**Teléfono**  
+51 924 137 543

  
**Correo Electrónico**  
ElyEllaDetalles@gmail.com



Tu Nombre

Tu Correo Electrónico

Asunto

Mensaje

*Nota: A través de este formulario de atención al cliente, se podrá resolver las dudas e inquietudes de los usuarios de manera organizada y eficiente. Fuente: Elaboración Propia.*

### 3.2.3 Tecnologías por utilizar

Java se utilizará para implementar la lógica del constructor de arreglos y la gestión de pedidos.

La interfaz de usuario se desarrollará con tecnologías web estándar.

## 3.3 Alternativa 3

### Plataforma de Suscripción para Entrega de Flores

#### 3.3.1 Descripción general

Creación de un sistema de suscripción donde los usuarios pueden elegir recibir arreglos florales periódicamente, con opciones de personalización y gestión de suscripciones.

#### 3.3.2 Pantallas propuestas

- 1. Página de inicio:** La página de inicio presenta la propuesta de valor del servicio de suscripción de flores, destacando los beneficios de recibir arreglos florales frescos de manera periódica. Incluye imágenes atractivas de arreglos recientes, una breve explicación del servicio, y enlaces directos a la información sobre la suscripción y los tipos de suscripción disponibles.

**Figura 13:**  
Página de inicio de la alternativa de solución 3



**Nota:** Esta primera pantalla resalta las funcionalidades de la página de inicio para "El y Ella Detalles, con una variación en la paleta de colores tradicional de una floristería. **Fuente:** Elaboración Propia.

**2. Información sobre la suscripción:** Esta pantalla proporciona detalles completos sobre cómo funciona el servicio de suscripción. Explica las ventajas de suscribirse, la flexibilidad del servicio, y cómo se personalizan los arreglos para cada entrega. También se incluyen testimonios de clientes y una sección de preguntas frecuentes para resolver dudas comunes.

**Figura 14:**  
Información detallada necesaria sobre la suscripción en la floristería para la alternativa de solución 3

The screenshot shows a detailed subscription information page. The left side features a large, vibrant photograph of a hand holding a lush bouquet of pink roses, green hydrangeas, and yellow tulips. The right side contains text and bullet points. At the top right is a section titled 'Sobre Nuestra Suscripción Floral' with a sub-section about the benefits of their service. Below this is a bulleted list of advantages: 'Variedad de Arreglos', 'Comodidad', 'Beneficios Exclusivos', and 'Personalización'. Further down is a question and answer section about the subscription, followed by a paragraph about the service's unique selling proposition. The footer includes standard navigation links.

**Nota:** Este apartado de información detallada acerca la suscripción en la plataforma digital de la floristería es importante para la seguridad de los usuarios. **Fuente:** Elaboración Propia.

**3. Tipos de suscripción:** En esta pantalla, los usuarios pueden explorar y seleccionar entre diferentes planes de suscripción. Los planes pueden variar en contenido (básica, estándar y premium), tamaño de los arreglos, y tipo de flores. Cada opción se presenta con detalles y precios, permitiendo a los usuarios comparar fácilmente y elegir la que mejor se ajuste a sus necesidades.

**Figura 15:**

Apartado de los tipos de suscripción en la floristería para la alternativa de solución 3

The screenshot displays a dark-themed website section titled "Nuestras Suscripciones de Arreglos Florales". At the top, there are three tabs: "Básica" (highlighted in red), "Estándar", and "Premium". Below each tab is a photograph of a woman surrounded by flowers. The first image shows a woman in a pink hat with a variety of flowers. The second image shows a woman with long blonde hair in a blue dress. The third image shows a woman in a straw hat holding large pink flowers. Each image has a price tag below it: "\$50", "\$100", and "\$150" respectively. Below the price tags are the subscription names: "Suscripción Básica", "Suscripción Estándar", and "Suscripción Premium". Underneath each name is a brief description in small text.

Tipo de Suscripción	Precio	Descripción
Suscripción Básica	\$50	Incluye un arreglo floral mensual con flores de temporada. Ideal para mantener un ambiente fresco y alegre en tu hogar.
Suscripción Estándar	\$100	Ofrece dos arreglos florales al mes, incluyendo flores exóticas y opciones más variadas para alegrar tu espacio.
Suscripción Premium	\$150	Disfruta de cuatro arreglos florales mensuales con una selección exclusiva de flores y diseño personalizado para cada ocasión especial.

**Nota:** Este apartado resume el contenido de cada tipo de suscripción en la plataforma digital de la floristería, cada uno con sus respectivos costos asequibles para los usuarios. **Fuente:** Elaboración Propia.

**4. Formulario para los detalles de envío:** Despues de seleccionar un plan de suscripción, los usuarios son dirigidos al formulario de detalles de envío. Aquí ingresan su dirección, preferencias de entrega (por ejemplo, días específicos o instrucciones especiales), y cualquier otra información relevante para la entrega de sus arreglos florales.

**Figura 16:**

Formulario para agregar los detalles de envío de arreglos florales para la alternativa de solución 3

Correo Electrónico

Dirección de Entrega

dd/mm/aaaa

Número de Arreglos

Información Adicional

Enviar Elección

**Nota:** A través de este formulario se podrá recolectar información clave para el envío de arreglos florales de los usuarios. **Fuente:** Elaboración Propia.

**5. Formulario de pago:** Finalmente, los usuarios completan el proceso en la pantalla de formulario de pago, donde ingresan los datos de su tarjeta de crédito u otra forma de pago.

**Figura 17:**

Formulario de pago para la alternativa de solución 3

Nombre Completo

Correo Electrónico

Número de Tarjeta

Fecha de Expiración

Código de Seguridad (CVV)

MM/AA

CVV

Pagar Ahora

**Nota:** A través de este formulario se podrá acceder al canal de pago para los suscriptores de la floristería digital “El y Ella Detalles”. **Fuente:** Elaboración Propia.

### 3.3.3 Tecnologías por utilizar

El backend en Java gestionará las suscripciones, el historial de entregas y las preferencias del usuario, mientras que el frontend utilizará HTML, CSS, y JavaScript para la experiencia de usuario.

## 4 Análisis de la solución

### 4.1 Levantamiento de información

El levantamiento de información es el proceso inicial y crucial en el análisis de requerimientos. Consiste en recopilar datos relevantes sobre las necesidades del cliente, los procesos de negocio existentes y las expectativas de los usuarios finales. Esta fase sienta las bases para la definición precisa de los requerimientos del sistema, asegurando que el producto final se alinee con las necesidades reales del negocio y sus usuarios.

Para el levantamiento de información, el equipo ha empleado las siguientes técnicas y herramientas:

**a) Entrevista estructurada:** Se ha diseñado un formato de entrevista dirigido al dueño(a) de la floristería "El y Ella Detalles". Esta entrevista tiene como objetivo obtener información detallada sobre los procesos actuales del negocio, las necesidades específicas de la empresa y las expectativas respecto al sistema de comercio digital.

**b) Encuesta a usuarios:** Se ha preparado un formato de encuesta dirigido a los clientes o usuarios potenciales del sistema. Esta encuesta busca recopilar datos sobre las preferencias de los clientes, sus hábitos de compra en línea y sus expectativas en cuanto a la experiencia de usuario en una plataforma de comercio electrónico para floristerías.

**c) Análisis de sistemas de comercio electrónico existentes:** El equipo ha realizado un estudio comparativo de plataformas de comercio electrónico utilizadas por otras floristerías,

identificando características comunes, mejores prácticas y áreas de oportunidad para el desarrollo del sistema.

**d) Revisión de literatura académica:** Se ha llevado a cabo una investigación de artículos académicos y publicaciones especializadas sobre comercio electrónico y sistemas de gestión de inventarios, con el fin de incorporar conocimientos actualizados y tendencias relevantes en el diseño del sistema.

**Nota:** Los formatos detallados de la entrevista dirigida al dueño(a) de la empresa y de la encuesta dirigida a los clientes o usuarios se encuentran disponibles en la sección "Anexos" de este documento. Estos instrumentos han sido cuidadosamente diseñados para capturar información crucial para el desarrollo del proyecto y pueden ser consultados para obtener una comprensión más profunda de la metodología de recolección de datos empleada.

## 4.2 Matriz de requerimientos

El análisis de requerimientos es una fase crucial en el desarrollo de software que permite identificar y documentar las necesidades específicas del sistema a desarrollar. En el contexto de este proyecto académico para la floristería "El y Ella Detalles", este análisis se enfoca en definir las funcionalidades y características esenciales del sistema de comercio digital, asegurando que cumpla con las expectativas del cliente y los objetivos de aprendizaje del curso.

### 4.2.1 Requisitos funcionales

Los requisitos funcionales describen las capacidades y funciones específicas que el sistema debe proporcionar. Estos requerimientos definen lo que el sistema debe hacer, detallando las interacciones entre el sistema y su entorno, así como los comportamientos esperados en diferentes situaciones. Los requerimientos funcionales son esenciales para establecer las características operativas del sistema de comercio digital para "El y Ella Detalles".

A continuación, se muestra el listado de los requerimientos funcionales del proyecto:

**Tabla 4:**  
Lista de requerimientos funcionales del proyecto

REQUERIMIENTOS FUNCIONALES													
Introducción													
Nombre del Proyecto		Sistema de Comercio Digital para "El y Ella Detalles"											
Fecha		28 de agosto del 2024											
Versión del Documento		2.0											
Autor		Equipo de estudiantes del curso Integrador 1: Sistemas – Software. Grupo 01											
Revisión		01 de setiembre del 2024											
Objetivo del Documento													
Este documento describe los requerimientos funcionales para el Sistema de Comercio Digital de la floristería "El y Ella Detalles". Los requerimientos funcionales especifican las funciones que el sistema debe realizar para satisfacer las necesidades del negocio y los criterios de éxito del proyecto académico.													
Matriz de actividades y requisitos del Sistema de Comercio Digital para "El y Ella Detalles"													
Proceso del negocio	Actividad del negocio	Responsable del negocio	Requerimiento funcional		Caso de Uso		Actores/Stakeholders	Prioridad					
Gestión de Usuarios	Registrar usuarios	Sistema	RF01	Permitir a los nuevos usuarios registrarse proporcionando información como nombre, correo electrónico y contraseña	CU01	Registrar nuevo usuario	Cliente, Administrador	Alta					
	Autenticar usuarios		RF02	Permitir a los usuarios iniciar sesión con sus credenciales (correo electrónico y contraseña)	CU02	Autenticar usuario		Alta					
	Actualizar perfil de usuario	Cliente	RF03	Permitir a los usuarios actualizar su información personal, dirección de envío y preferencias de compra	CU03	Gestionar perfil de usuario	Cliente	Media					
	Restablecer contraseña	Sistema	RF04	Permitir a los usuarios recuperar o restablecer su contraseña mediante un correo electrónico de recuperación	CU04	Restablecer contraseña		Media					
Gestión de Catálogo	Mostrar catálogo	Sistema	RF05	Presentar un catálogo de productos con imágenes, descripciones y precios.	CU05	Mostrar catálogo de productos	Cliente, Administrador	Alta					
	Categorizar productos	Administrador	RF06	Organizar los productos en categorías (por ejemplo, ramos, arreglos florales, plantas)	CU06	Categorizar productos	Administrador	Media					

Gestión de Compras	Buscar y filtrar productos	Sistema	RF07	Permitir a los usuarios buscar productos por nombre, categoría o tipo de ocasión	CU07	Buscar y filtrar productos	Cliente	Alta
			RF08	Mostrar información detallada al seleccionar un producto, incluyendo opciones de personalización disponibles.				
	Administrar productos	Administrador	RF09	Proporcionar una vista previa visual de las personalizaciones realizadas.	CU09	Gestionar productos	Administrador	Alta
Gestión de Compras	Administrar carrito de compras	Cliente	RF10	Permitir a los usuarios añadir productos al carrito de compras.	CU10	Agregar productos al carrito	Cliente	Alta
			RF11	Permitir a los usuarios modificar la cantidad de productos o eliminarlos del carrito.	CU11	Modificar carrito		Alta
	Mostrar resumen de compra	Sistema	RF12	Presentar un resumen del carrito con el total de la compra, incluyendo	CU12	Mostrar resumen del carrito		Alta
	Seleccionar método de entrega	Cliente	RF13	Permitir a los usuarios elegir entre recoger en tienda o envío a domicilio.	CU13	Seleccionar método de entrega		Alta
	Definir entrega		RF14	Permitir seleccionar fecha y franja horaria de entrega para envíos a domicilio	CU14	Programar entrega		Alta
	Procesar pago	Sistema	RF15	Simular el proceso de pago, incluyendo la selección de método de pago (por ejemplo, tarjeta de crédito, PayPal).	CU15	Procesar pago	Cliente, Sistema	Alta
	Generar confirmación de pedido		RF16	Generar una confirmación de pedido con un número de seguimiento tras completar la compra	CU16	Generar confirmación de pedido		Alta
Gestión de inventario	Actualizar inventario automáticamente	Sistema	RF17	Actualizar el inventario automáticamente después de cada venta	CU18	Actualizar inventario automáticamente	Sistema	Alta
	Generar alertas de stock		RF18	Generar alertas cuando el stock de un producto esté por debajo de un umbral predefinido	CU19	Generar alertas de stock bajo	Sistema, Administrador	Media
Personalización de Productos	Personalizar arreglos florales	Cliente	RF19	Permitir a los usuarios personalizar ciertos aspectos de los arreglos florales (por ejemplo, color de las flores, tipo de florero)	CU20	Personalizar productos	Cliente	Media
Administración de Pedidos	Visualizar pedidos	Administrador	RF20	Permitir a los administradores ver y gestionar los pedidos recibidos.	CU21	Ver pedidos recibidos		Alta

	Actualizar estado de pedidos				CU22	Actualizar estado de pedidos			Alta				
Reporte y Análisis	Generar reportes de ventas	Administrador	RF21	Generar reportes básicos de ventas y productos más vendidos.	CU23	Generar reportes de ventas	Administrador	Media	Media				
	Analizar productos más vendidos				CU24	Generar reporte de productos más vendidos			Media				
	<b>Notas adicionales</b>												
Limitaciones Conocidas		<ul style="list-style-type: none"> <li>• El sistema es un prototipo académico y puede no incluir todas las funcionalidades de un sistema comercial completo.</li> </ul>											
Consideraciones Especiales		<ul style="list-style-type: none"> <li>• El sistema debe ser desarrollado utilizando Java para al menos el 50% del backend.</li> <li>• Se debe priorizar la usabilidad y la experiencia de usuario en el diseño de la interfaz.</li> </ul>											

*Nota:* Esta lista de requerimientos funcionales proporciona una base sólida para el desarrollo del Sistema de Comercio Digital para "El y Ella Detalles", considerando las necesidades específicas de una floristería en línea y el contexto académico del proyecto. **Fuente:** Elaboración Propia.

#### 4.2.2 Requisitos no funcionales

Los requisitos no funcionales especifican los criterios que pueden usarse para juzgar la operación de un sistema, en lugar de sus comportamientos específicos. Estos requisitos se centran en aspectos como la usabilidad, rendimiento, seguridad y mantenibilidad del sistema. Aunque no se describen funciones específicas, son cruciales para garantizar la calidad, eficiencia y robustez del sistema de comercio digital.

A continuación, se muestra el listado de los requerimientos no funcionales del proyecto:

**Tabla 5:**

Lista de requerimientos no funcionales del proyecto

REQUERIMIENTOS NO FUNCIONALES														
Introducción														
Nombre del Proyecto	Sistema de Comercio Digital para "El y Ella Detalles"													
Fecha	28 de agosto del 2024													
Versión del Documento	2.0													
Autor	Equipo de estudiantes del curso Integrador 1: Sistemas – Software. Grupo 01													
Revisión	01 de setiembre del 2024													
Objetivo del Documento														
Este documento describe los requerimientos no funcionales para el Sistema de Comercio Digital de la floristería "El y Ella Detalles". Los requerimientos no funcionales definen los atributos cualitativos del sistema y cómo debe comportarse en diversos contextos.														
Matriz de actividades y requisitos del Sistema de Comercio Digital para "El y Ella Detalles"														
Proceso del negocio	Actividad del negocio	Responsable del negocio	Requerimiento funcional		Caso de Uso		Actores/Stakeholders	Prioridad						
Gestión de Usuarios	Registrar usuarios	Sistema	RNF01	Seguridad, Usabilidad	CU01	Registrar nuevo usuario	Cliente, Administrador, sistema	Alta						
	Autenticar usuarios		RNF02	Seguridad, Rendimiento	CU02	Autenticar usuario		Alta						
	Actualizar perfil de usuario	Cliente	RNF03	Usabilidad, Seguridad	CU03	Gestionar perfil de usuario	Cliente	Media						
	Restablecer contraseña	Sistema	RNF04	Seguridad, Fiabilidad	CU04	Restablecer contraseña		Media						
Gestión de Catálogo	Mostrar catálogo	Sistema	RNF05	Rendimiento, Usabilidad	CU05	Mostrar catálogo de productos	Cliente, Administrador	Alta						
	Categorizar productos	Administrador	RNF06	Usabilidad, Mantenibilidad	CU06	Categorizar productos	Administrador	Media						
	Buscar y filtrar productos	Sistema	RNF07	Rendimiento, Usabilidad	CU07	Buscar y filtrar productos	Cliente	Alta						
	Administrar productos	Administrador	RNF08	Usabilidad, Mantenibilidad	CU09	Gestionar productos	Administrador	Alta						
Gestión de Compras	Administrar carrito de compras	Cliente	RNF9	Usabilidad, Rendimiento	CU10	Agregar productos al carrito	Cliente	Alta						
	Mostrar resumen de compra				CU11	Modificar carrito		Alta						
		Sistema			CU12	Mostrar resumen del carrito		Alta						

	Seleccionar método de entrega	Cliente	RNF10	Usabilidad	CU13	Seleccionar método de entrega		Alta			
	Definir entrega		RNF11	Usabilidad, Fiabilidad	CU14	Programar entrega		Alta			
Gestión de inventario	Procesar pago	Sistema	RNF12	Seguridad, Rendimiento	CU15	Procesar pago	Cliente, Sistema	Alta			
	Generar confirmación de pedido		RNF13	Fiabilidad, Rendimiento	CU16	Generar confirmación de pedido		Alta			
	Actualizar inventario automáticamente		RNF14		CU17			Alta			
Personalización de Productos	Actualizar inventario automáticamente	Sistema	RNF14	Fiabilidad, Rendimiento	CU18	Actualizar inventario automáticamente	Sistema	Alta			
	Generar alertas de stock				CU19	Generar alertas de stock bajo	Sistema, Administrador	Media			
Personalización de Productos	Personalizar arreglos florales	Cliente	RNF15	Usabilidad, Rendimiento	CU20	Personalizar productos	Cliente	Media			
Administración de Pedidos	Visualizar pedidos	Administrador			CU21	Ver pedidos recibidos		Alta			
	Actualizar estado de pedidos				CU22	Actualizar estado de pedidos		Alta			
Reporte y Análisis	Generar reportes de ventas	Administrador	RF16	Usabilidad, Rendimiento	CU23	Generar reportes de ventas	Administrador	Media			
	Analizar productos más vendidos				CU24	Generar reporte de productos más vendidos		Media			
<b>Notas adicionales</b>											
Limitaciones Conocidas			<ul style="list-style-type: none"> <li>La escalabilidad y el rendimiento pueden estar limitados por el entorno de hosting utilizado para el proyecto académico.</li> </ul>								
Consideraciones Especiales			<ul style="list-style-type: none"> <li>El sistema es un prototipo académico y puede no incluir todas las medidas de seguridad de un sistema comercial.</li> </ul>								

*Nota:* Esta tabla proporciona una visión general de los requerimientos no funcionales para el Sistema de Comercio Digital de "El y Ella Detalles", adaptado al contexto de un proyecto académico.

*Fuente:* Elaboración Propia.

#### 4.2.3 Especificación de Requerimientos de Software (SRS)

La Especificación de Requerimientos de Software (SRS) es un documento comprensivo que describe de manera detallada y estructurada todos los aspectos del sistema a desarrollar. Sirve como un contrato entre el equipo de desarrollo y el cliente, en este caso, el contexto académico del proyecto. El SRS combina los requisitos funcionales y no funcionales, junto con otros detalles importantes, para proporcionar una visión completa y clara del sistema propuesto.

A continuación, se muestra la Especificación de Requerimientos de Software (SRS) del proyecto a través de la siguiente tabla:

**Tabla 6:**  
*Especificación de requerimientos de software del proyecto*

ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE	
Introducción	
Propósito	El propósito de este documento es definir los requerimientos específicos para el desarrollo del Sistema de Comercio Digital para la floristería "El y Ella Detalles". Este documento servirá como guía para el equipo de desarrollo y como acuerdo entre el equipo y las partes interesadas del proyecto.
Alcance	El sistema por desarrollar es una plataforma de comercio electrónico que permitirá a "El y Ella Detalles" vender sus productos en línea. El sistema incluye un catálogo de productos, carrito de compras, proceso de pago, gestión de inventario y una interfaz de administración.
Definiciones, acrónimos y abreviaturas	<ul style="list-style-type: none"><li>• SRS: Especificación de requisitos de software</li><li>• RF: Requerimiento Funcional</li><li>• RNF: Requerimiento No Funcional</li><li>• UI: Interfaz de usuario</li><li>• API: Interfaz de Programación de Aplicaciones</li></ul>
Referencias	Norma IEEE 830-1998, Práctica recomendada por IEEE para especificaciones de requisitos de software.
Visión general del documento	Este documento contiene una descripción general del sistema, los requisitos específicos, tanto funcionales como no funcionales, y los modelos del sistema.
Descripción general	
Perspectiva del producto	El Sistema de Comercio Digital para "El y Ella Detalles" es un producto independiente que busca modernizar las operaciones de la floristería, permitiéndole alcanzar un mercado más amplio a través de Internet.
Funciones del producto	Las principales funciones del sistema incluyen: <ul style="list-style-type: none"><li>• Gestión de catálogo de productos</li><li>• Carrito de compras</li><li>• Proceso de compra y pago</li><li>• Gestión de inventario</li><li>• Personalización de productos</li><li>• Interfaz de administración</li></ul>
Características de los usuarios	<ul style="list-style-type: none"><li>• Clientes: Usuarios que buscan comprar productos florales en línea.</li><li>• Administradores: Personal de la floristería que gestionará el catálogo, inventario y pedidos.</li></ul>
Restricciones	<ul style="list-style-type: none"><li>• El sistema debe desarrollarse utilizando Java para al menos el 50% del backend.</li><li>• El proyecto debe completarse en un plazo de 16 semanas.</li><li>• El sistema debe ser compatible con los navegadores web más comunes.</li></ul>

Suposiciones y dependencia	<ul style="list-style-type: none"> <li>Se supone que los usuarios tendrán acceso a Internet y dispositivos compatibles.</li> <li>El sistema dependerá de servicios de terceros para el procesamiento de pagos.</li> </ul>					
<b>Requisitos específicos</b>						
<b>Interfaces externas</b>						
Interfaces de usuario	El sistema tendrá una interfaz web responsive, accesible desde navegadores en dispositivos móviles y de escritorio.					
Interfaces de hardware	No se requieren interfaces de hardware específicas.					
Interfaces de software	El sistema se integrará con una pasarela de pago (simulada para este proyecto académico).					
Interfaces de comunicación	El sistema utilizará protocolos estándar de Internet (HTTP/HTTPS).					
<b>Requisitos funcionales</b>						
RF-1	Registrar nuevos usuarios	Permitir a los nuevos usuarios registrarse proporcionando información como nombre, correo electrónico y contraseña.	Considerar la validación de correo electrónico.			
RF-2	Autenticar usuarios	Permitir a los usuarios iniciar sesión con sus credenciales (correo electrónico y contraseña).	Implementar medidas de seguridad como bloqueo tras múltiples intentos fallidos.			
RF-3	Gestionar perfil de usuario	Permitir a los usuarios actualizar su información personal, dirección de envío y preferencias de compra.	Incluir opción para guardar múltiples direcciones de envío.			
RF-4	Restablecer contraseña	Permitir a los usuarios recuperar o restablecer su contraseña mediante un correo electrónico de recuperación.	Asegurar que el enlace de restablecimiento tenga un tiempo de vencimiento.			
RF-5	Mostrar catálogo de productos	Presentar un catálogo de productos con imágenes, descripciones y precios.	Considerar implementar paginación para mejorar el rendimiento.			
RF-6	Categorizar productos	Organizar los productos en categorías (por ejemplo, ramos, arreglos florales, plantas).	Permitir que un producto pertenezca a múltiples categorías si es necesario.			
RF-7	Buscar y filtrar productos	Permitir a los usuarios buscar productos por nombre, categoría o tipo de ocasión.	Implementar búsqueda predictiva para mejorar la experiencia del usuario.			
RF-8	Visualizar detalles del producto	Mostrar información detallada al seleccionar un producto, incluyendo opciones de personalización disponibles.	Incluir galería de imágenes para cada producto.			
RF-9	Agregar productos al carrito	Permitir a los usuarios añadir productos al carrito de compras.	Mostrar confirmación visual al agregar un producto.			
RF-10	Modificar carrito	Permitir a los usuarios modificar la cantidad de productos o eliminarlos del carrito.	Actualizar automáticamente el total al realizar cambios.			
RF-11	Mostrar resumen del carrito	Presentar un resumen del carrito con el total de la compra, incluyendo impuestos y costos de envío.	Considerar mostrar productos recomendados en esta vista.			
RF-12	Seleccionar método de entrega	Permitir a los usuarios elegir entre recoger en tienda o envío a domicilio.	Mostrar costos diferentes según el método seleccionado.			
RF-13	Programación de entrega	Permitir seleccionar fecha y franja horaria de entrega para envíos a domicilio.	Considerar restricciones de horarios y fechas especiales.			
RF-14	Procesar pago	Simular el proceso de pago, incluyendo la selección de método de pago (por ejemplo, tarjeta de crédito, PayPal).	Implementar medidas de seguridad para la información de pago.			
RF-15	Generar confirmación de pedido	Generar una confirmación de pedido con un número de seguimiento tras completar la compra.	Enviar confirmación por correo electrónico además de mostrarla en pantalla.			
RF-16	Actualizar inventario automáticamente	Actualizar el inventario automáticamente después de cada venta.	Considerar manejar casos de concurrencia en ventas simultáneas.			

RF-17	Generar alertas de stock bajo	Generar alertas cuando el stock de un producto esté por debajo de un umbral predefinido.	Permitir configurar el umbral por producto o categoría.
RF-18	Personalizar productos	Permitir a los usuarios personalizar ciertos aspectos de los arreglos florales (por ejemplo, color de las flores, tipo de florero).	Asegurar que las opciones de personalización afectarán el precio final.
RF-19	Gestionar productos	Permitir a los administradores agregar, editar y eliminar productos del catálogo.	Incluir opción para desactivar temporalmente productos sin eliminarlos.
RF-20	Gestionar pedidos	Permitir a los administradores ver y gestionar los pedidos recibidos.	Implementar sistema de estados para seguimiento de pedidos.
RF-21	Generar informes básicos	Generar informes básicos de ventas y productos más vendidos.	Considerar opciones de exportación a diferentes formatos (PDF, Excel).
<b>Requisitos no funcionales</b>			
RNF-1	Garantizar tiempo de respuesta	El sistema debe cargar las páginas en menos de 3 segundos en condiciones normales de red.	Se utilizarán técnicas de optimización de carga y caché para cumplir este requisito.
RNF-2	Asegurar escalabilidad	El sistema debe soportar hasta 100 usuarios simultáneos sin degradar el rendimiento.	Se implementará un sistema de balanceo de carga para manejar picos de tráfico.
RNF-3	Mantener capacidad de procesamiento	El sistema debe ser capaz de procesar 50 transacciones por minuto.	Se optimizarán las consultas a la base de datos y se utilizará un sistema de cola para gestionar transacciones.
RNF-4	Mantener disponibilidad	El sistema debe asegurar una disponibilidad del 99% durante el horario operativo de la floristería.	Se implementará un sistema de monitoreo continuo para detectar y resolver problemas rápidamente.
RNF-5	Implementar recuperación ante fallos	El sistema debe ser capaz de recuperarse en menos de 30 minutos después de una interrupción.	Se establecerá un plan de recuperación ante desastres y se realizarán copias de seguridad regulares.
RNF-6	Implementar autenticación segura	El sistema debe proteger las cuentas de usuario con métodos de autenticación robustos.	Se utilizará autenticación de dos factores para cuentas de administrador.
RNF-7	Establecer autorización basada en roles	El sistema debe controlar el acceso a funciones según roles de usuario (cliente, administrador).	Se implementará un sistema de gestión de acceso basado en roles (RBAC).
RNF-8	Diseñar interfaz intuitiva	El sistema debe tener una interfaz fácil de usar con diseño responsive para diferentes dispositivos.	Se realizarán pruebas de usabilidad con usuarios reales durante el desarrollo.
RNF-9	Cumplir estándares de accesibilidad	El sistema debe seguir las pautas básicas de accesibilidad web WCAG 2.1 nivel A.	Se utilizarán herramientas de evaluación de accesibilidad durante el desarrollo.
RNF-10	Facilitar actualizaciones	El sistema debe ser modular para permitir actualizaciones sin interrupciones generales.	Se implementará una arquitectura de microservicios para facilitar las actualizaciones parciales.
RNF-11	Proporcionar documentación	El sistema debe contar con documentación clara del código y manuales de usuario.	Se utilizará un sistema de documentación automática para mantener la documentación actualizada.
RNF-12	Asegurar compatibilidad con navegadores	El sistema debe funcionar correctamente en las versiones más recientes de Chrome, Firefox y Safari.	Se realizarán pruebas de compatibilidad cruzada regularmente durante el desarrollo.
RNF-13	Garantizar compatibilidad con dispositivos	El sistema debe operar adecuadamente en dispositivos móviles y computadoras de escritorio.	Se implementará un diseño responsive utilizando frameworks modernos como Bootstrap.

RNF-14	Permitir implementación flexible	El sistema debe poder implementarse en un entorno de hosting compartido estándar.	Se utilizarán tecnologías ampliamente soportadas para asegurar la compatibilidad con diferentes entornos de hosting.
RNF-15	Cumplir normativas de privacidad	El sistema debe adherirse a las regulaciones básicas de protección de datos personales.	Se implementará un sistema de gestión de consentimiento para el manejo de datos personales.
<b>Modelos del sistema</b>			
Casos de uso	Caso de uso: Realizar compra Actor principal: Cliente Flujo básico: <ul style="list-style-type: none"><li>• El cliente navega por el catálogo.</li><li>• El cliente agrega productos al carrito.</li><li>• El cliente procede al pago</li><li>• El cliente ingresa información de envío y pago.</li><li>• El sistema procesa el pago y confirma la orden.</li></ul>		
Diagramas de secuencia	El diagrama de secuencias del proyecto se muestra en la sección <a href="#">Estructura de la solución</a> para mayor detalle.		
Diagrama de clases	El diagrama de clases del proyecto también se encuentra en la sección <a href="#">Estructura de la solución</a> para mayor detalle.		
<b>Apéndices</b>			
Cronograma tentativo	<ul style="list-style-type: none"><li>• Semana 1-2: Análisis y diseño</li><li>• Semana 3-10: Desarrollo</li><li>• Semana 11-12: Pruebas</li><li>• Semana 13-14: Documentación</li><li>• Semana 15-16: Presentación y entrega</li></ul>		
Glosario	<ul style="list-style-type: none"><li>• Comercio electrónico: Comercio electrónico, compra y venta de productos o servicios a través de Internet.</li><li>• Frontend: Parte del software que interactúa con los usuarios.</li><li>• Backend: Parte del software que procesa la entrada recibida desde el frontend.</li></ul>		
Análisis de problemas pendientes	<ul style="list-style-type: none"><li>• Definir la integración exacta con sistemas de pago (simulada para este proyecto).</li><li>• Establecer los detalles específicos del diseño de la interfaz de usuario.</li></ul>		

*Nota:* Este SRS proporciona una visión general y detallada del Sistema de Comercio Digital para "El y Ella Detalles".

*Fuente: Elaboración Propia.*

## 5 Diseño de la solución

El diseño de la solución es fundamental para asegurar el correcto desarrollo del sistema de comercio digital para la floristería "El y Ella Detalles". En esta sección, se aborda cómo la arquitectura propuesta, siguiendo principios y patrones modernos, permite cumplir con los requerimientos de funcionalidad, eficiencia y seguridad. A continuación, se presenta un desglose detallado de los componentes principales del diseño.

### 5.1 Arquitectura del proyecto

La arquitectura define la estructura del sistema, asegurando su modularidad, escalabilidad y facilidad de mantenimiento. El uso de patrones bien definidos permite

mantener la separación de responsabilidades y garantizar un desarrollo más eficiente. A continuación, se explican los elementos clave de esta arquitectura.

### 5.1.1 Aplicación del patrón MVC (Modelo, Vista, Controlador)

El patrón MVC se implementa para separar las responsabilidades del sistema en capas independientes: la gestión de datos, la interfaz gráfica y el control de la lógica. Esta estructura favorece la organización del código y facilita futuras ampliaciones o mantenimientos del sistema. A continuación, se describen tres principios clave que se han implementado:

#### 1. Separación de la Lógica de Negocio y la Presentación

- En la carpeta de vistas, las páginas JSP (login.jsp, register.jsp, carshop.jsp, etc.) están diseñadas exclusivamente para mostrar la interfaz al usuario y manejar la interacción, sin incorporar lógica de negocio directa.
- Los controladores (MainController.java, ControlLogin.java, ControlCarrito.java, etc.) actúan como intermediarios entre la vista y el modelo, manejando la lógica de negocio y la manipulación de datos antes de enviarlos a la vista.
- Este principio se asegura al dividir el proyecto en tres carpetas principales: Vistas, Controladores y Modelos, cada una con responsabilidades bien definidas.

#### 2. Control Centralizado de las Peticiones

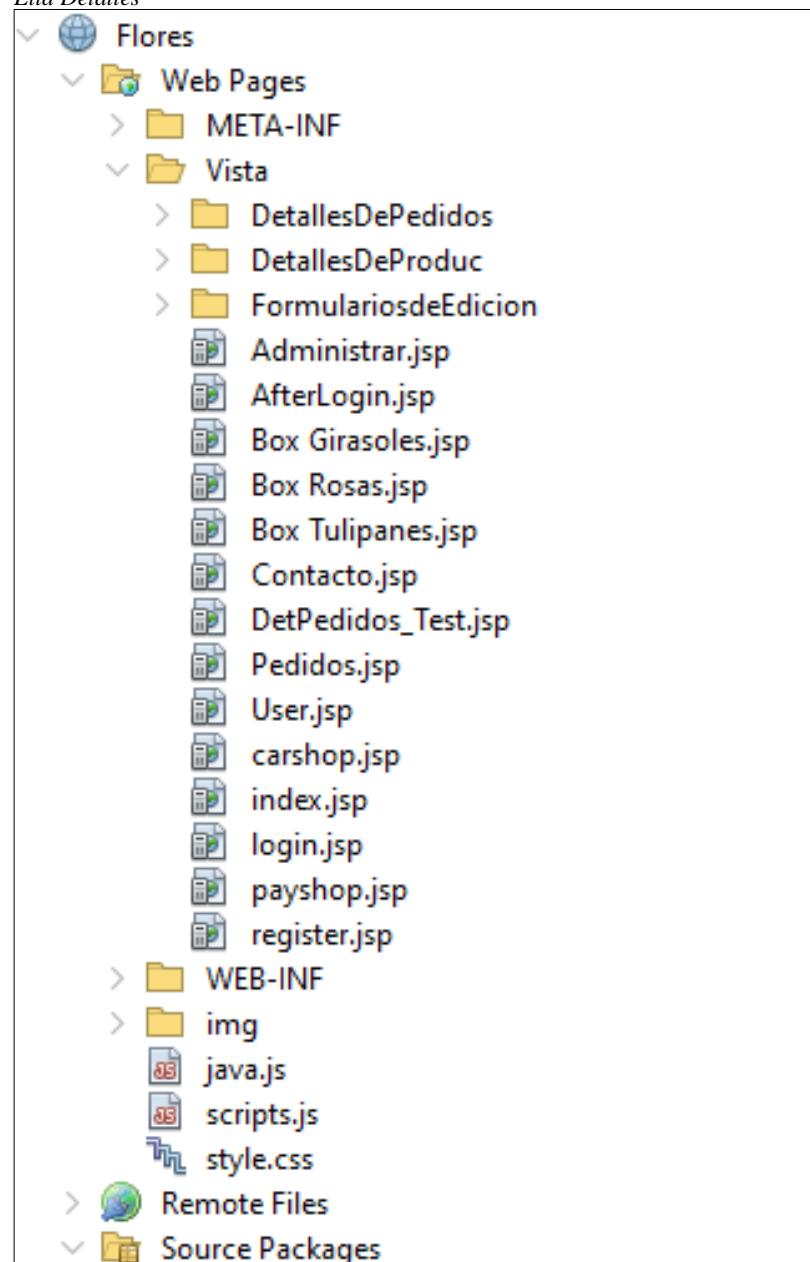
- En la carpeta de controladores, clases como MainController.java y ControlLogin.java manejan las peticiones del usuario desde las vistas, determinando qué acciones tomar en función de los datos recibidos y luego enviando la respuesta apropiada.
- El controlador centralizado permite gestionar el flujo de información de manera ordenada y consistente, garantizando que los datos se validen antes de ser enviados al modelo para su procesamiento o almacenamiento.

#### 3. Modularidad en la Vista y Reutilización de Componentes

- Las páginas JSP en la carpeta de vistas están organizadas para permitir la reutilización de componentes visuales y formularios (e.g., formularios de login, registros y edición de productos).
- Al mantener las vistas separadas de la lógica del controlador, es más fácil realizar cambios en la interfaz sin afectar la lógica de negocio, mejorando la mantenibilidad del sistema.

**Figura 18:**

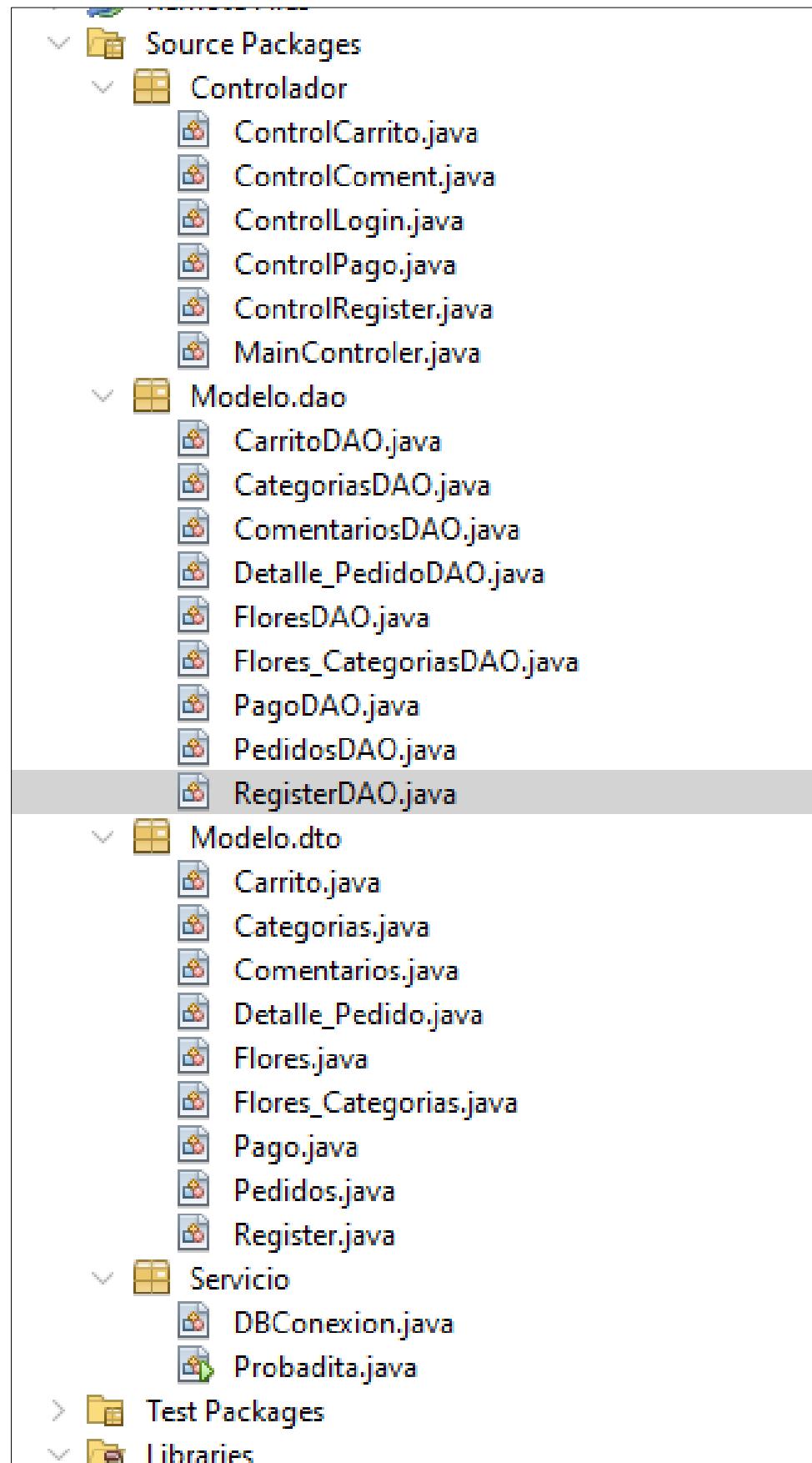
Estructura del Proyecto del Sistema de Comercio Digital para la Floristería "El y Ella Detalles"



*Nota: La figura muestra la organización del proyecto en Java, con interfaces JSP en la carpeta Vista, recursos en img y archivos esenciales como style.css, scripts.js, y WEB-INF para configuración y estilos.*

*Fuente: Elaboración Propia.*

*Figura 19:  
Organización de las Clases del Proyecto en Paquetes Java*



*Nota: La figura muestra la estructura del proyecto con paquetes organizados por roles: Controlador para la lógica de control, Modelo.dao para acceso a datos, Modelo.dto para transferencia de datos y Servicio para gestión de servicios como la conexión a la base de datos.*

*Fuente: Elaboración Propia.*

### 5.1.2 Implementación de DAO (Data Access Object) para gestión de datos

Para garantizar la integridad y eficiencia en la gestión de los datos del sistema, se utiliza el patrón DAO, el cual permite una interacción directa y segura con las fuentes de datos de la floristería, como productos, clientes y pedidos. A continuación, se destacan tres principios clave de esta implementación:

#### 1. Encapsulamiento de la Lógica de Acceso a Datos

- Cada clase DAO, como RegisterDAO.java, CarritoDAO.java y PedidosDAO.java, encapsula las operaciones específicas para cada entidad de la base de datos.
- Por ejemplo, RegisterDAO.java contiene métodos para gestionar las operaciones de inserción, como el método save() para registrar un usuario. Este principio asegura que la lógica de acceso a datos esté separada del resto de la aplicación, permitiendo cambios en la base de datos sin afectar la lógica de negocio.

#### 2. Uso de Conexiones Seguras y Gestión de Recursos

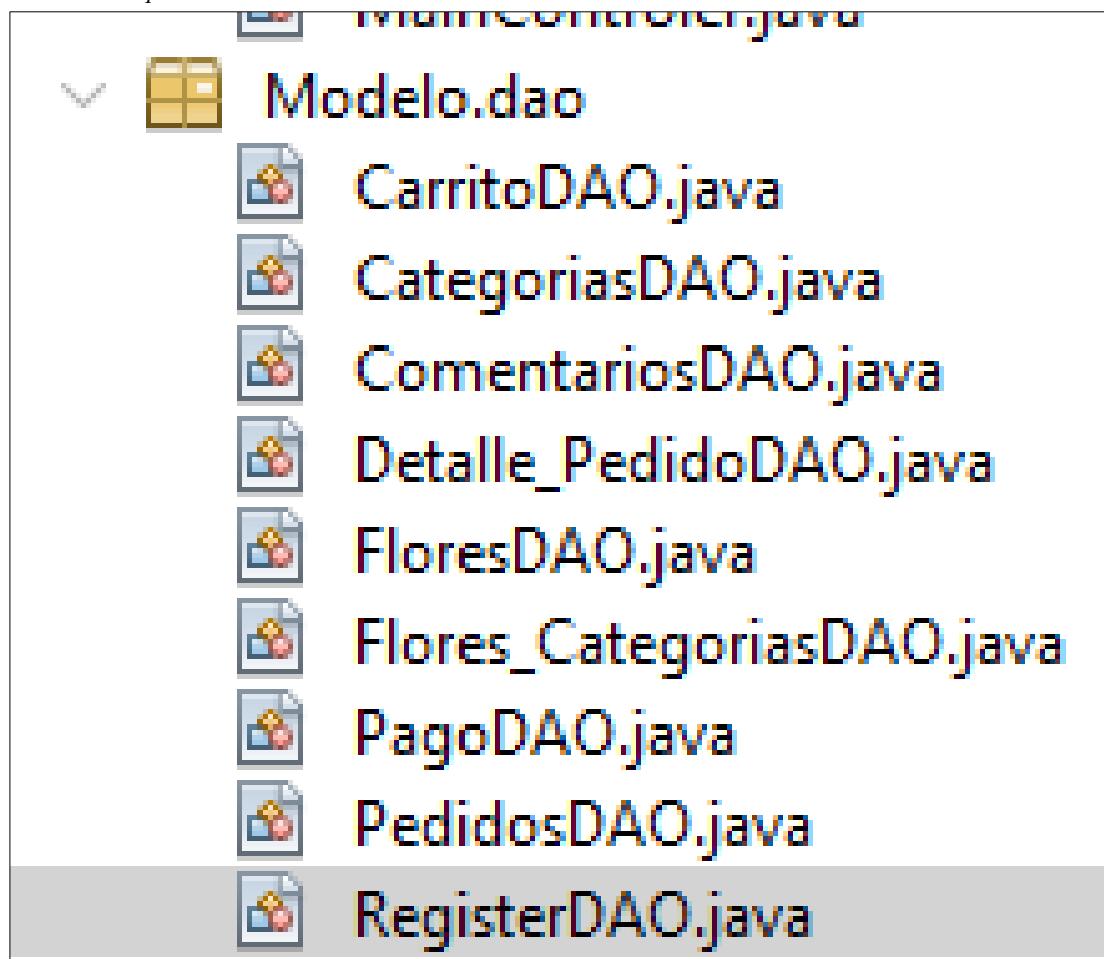
- En las clases DAO, se implementa la obtención de conexiones mediante una clase de servicio (DBConexion.java), lo cual asegura que todas las interacciones con la base de datos se realicen de manera controlada y eficiente.
- Se usa la estructura try-with-resources para manejar las conexiones, garantizando que los recursos se cierren automáticamente después de su uso, como se observa en el método save() de RegisterDAO.java.

#### 3. Reducción de Redundancia y Mantenibilidad

- La implementación de DAO promueve la reducción de código redundante, ya que las operaciones de acceso a datos comunes están centralizadas en clases DAO específicas.

- Por ejemplo, operaciones como la inserción, actualización, eliminación y búsqueda de datos se manejan dentro de sus respectivas clases DAO, lo que facilita la mantenibilidad del código y permite agregar nuevas operaciones sin modificar la estructura existente.

**Figura 20:**  
Clases del Paquete Modelo.dao



*Nota: La figura muestra las clases del paquete Modelo.dao, encargadas de gestionar el acceso a datos del sistema. Cada clase, como CarritoDAO.java y PedidosDAO.java, se encarga de la interacción con la base de datos para sus respectivas entidades, garantizando una separación clara entre la lógica de negocio y el manejo de datos. Fuente: Elaboración Propia.*

### 5.1.3 Pruebas orientadas a TDD (Test-Driven Development)

El Desarrollo Guiado por Pruebas o Test-Driven Development (TDD) es una metodología de desarrollo de software utilizada en todo este proyecto de e-commerce para una floristería, asegurando que cada funcionalidad sea implementada de manera correcta y eficiente desde el inicio. El ejemplo aquí mostrado se centra en el método

getCarritosById(int userId), pero el TDD se aplicó de manera similar en todo el proyecto para garantizar la calidad de cada módulo. A continuación, se describe cómo se aplicó el ciclo de TDD en este método, destacando tres principios fundamentales: Red, Green y Refactor.

#### 5.1.3.1 Principios de TDD

El desarrollo guiado por pruebas, conocido como TDD (Test-Driven Development), es una metodología de programación que prioriza la creación de pruebas antes de escribir el código funcional. Esta técnica asegura que cada componente desarrollado cumpla con los requisitos desde el inicio, garantizando así un código más fiable, mantenable y menos propenso a errores.

A continuación, se explicarán los principios fundamentales de TDD, su proceso y las ventajas que aporta al desarrollo del sistema de comercio digital para la floristería "El y Ella Detalles".

1. **Red:** Es la primera etapa del ciclo de TDD, donde se escribe una prueba que inicialmente falla, estableciendo el comportamiento esperado del método o componente antes de implementar cualquier código funcional.
  - En este caso, se escribió una prueba para el método getCarritosById(int userId), diseñada para verificar que el método devolviera una lista de carritos correcta para un usuario específico.
  - Al ejecutar la prueba, falló, ya que el método aún no estaba implementado. Esto es lo esperado en la fase "Red", que define el punto de partida para el desarrollo.

**Figura 21:**  
Prueba Unitaria con Mockito para CarritoDAO

```

public class CarritoDAOTest {
    private CarritoDAO carritoDAO;
    private Connection mockConnection;
    private PreparedStatement mockPreparedStatement;
    private ResultSet mockResultSet;

    private CarritoDAOTest(Connection mockConnection) {
    }

    @BeforeEach
    public void setUp(CarritoDAOTest CarritoDAOTest) throws Exception {
        mockConnection = Mockito.mock(Connection.class);
        mockPreparedStatement = Mockito.mock(PreparedStatement.class);
        mockResultSet = Mockito.mock	ResultSet.class);
        CarritoDAOTest = new CarritoDAOTest(mockConnection);
    }

    @Test
    public void testGetCarritosByUserId() throws Exception {
        // Configurar el comportamiento del ResultSet simulado
        Mockito.when(methodCall: mockConnection.prepareStatement(string: Mockito.anyString())).thenReturn(:mockPreparedStatement);
        Mockito.when(methodCall: mockPreparedStatement.executeQuery()).thenReturn(:mockResultSet);
        Mockito.when(methodCall: mockResultSet.next()).thenReturn(:true).thenReturn(:false); // Solo un resultado
        Mockito.when(methodCall: mockResultSet.getInt(string: "carrito_id")).thenReturn(1);
        Mockito.when(methodCall: mockResultSet.getInt(string: "flor_id")).thenReturn(101);
        Mockito.when(methodCall: mockResultSet.getInt(string: "cantidad")).thenReturn(2);
        Mockito.when(methodCall: mockResultSet.getDouble(string: "precio")).thenReturn(15.99);

        // Llamar al método a probar
        List<Carrito> carritos = carritoDAO.getCarritosByUserId(userId: 1);

        // Verificar el resultado
        assertNotNull(actual: carritos, message:"La lista de carritos no deberia ser nula");
        assertEquals(expected:1, actual: carritos.size(), message:"La lista de carritos deberia tener un elemento");
        assertEquals(expected:1, actual: carritos.get(index: 0).getIdCarrito(), message:"El ID del carrito no coincide");
        assertEquals(expected:101, actual: carritos.get(index: 0).getFlorId(), message:"El ID de la flor no coincide");
        assertEquals(expected:2, actual: carritos.get(index: 0).getCantidad(), message:"La cantidad no coincide");
        assertEquals(expected:15.99, actual: carritos.get(index: 0).getPrecio(), delta: 0.01, message:"El precio no coincide");
    }
}

```

**Nota:** La figura muestra una prueba unitaria realizada con Mockito para validar el comportamiento del método `getCarritosByUserId()` de la clase `CarritoDAO`. La prueba utiliza objetos simulados (mocks) para la conexión a la base de datos y las operaciones SQL, asegurando que el método funcione correctamente sin depender de recursos externos. Además, se verifican los resultados mediante aserciones, garantizando que los datos obtenidos cumplan con los valores esperados. Fuente: Elaboración Propia.

**2. Green:** En esta etapa, se implementa el código mínimo necesario para que la prueba pase, desarrollando solo la funcionalidad suficiente para hacer que la prueba deje de fallar.

- Para este método, se desarrolló la funcionalidad necesaria para recuperar la lista de carritos del usuario desde la base de datos y hacer que la prueba pasara.
- La prueba pasó con éxito, confirmando que la funcionalidad básica del método fue implementada correctamente.

**Figura 22:**  
Implementación del Método `getCarritosByUserId()` en `CarritoDAO`

```

public class CarritoDAOTest {

    // Declarar la conexión
    private Connection cnx;

    // Constructor que recibe la conexión
    public CarritoDAOTest(Connection connection) {
        this.cnx = connection;
    }

    public List<Carrito> getCarritosByUserId(int userId) {
        List<Carrito> carritos = new ArrayList<>();
        PreparedStatement ps;
        ResultSet rs;

        String sql = "SELECT ca.carrito_id, ca.cantidad, "
                    + "f.flor_id, f.precio "
                    + "FROM carrito ca "
                    + "INNER JOIN flores f ON ca.flor_id = f.flor_id "
                    + "WHERE ca.user_id = ?";

        try {
            ps = cnx.prepareStatement(string: sql);
            ps.setInt(i:1, il:userId);
            rs = ps.executeQuery();

            while (rs.next()) {
                Carrito carrito = new Carrito();
                carrito.setIdCarrito(idCarrito: rs.getInt(string: "carrito_id"));
                carrito.setFlorId(florId: rs.getInt(string: "flor_id"));
                carrito.setCantidad(cantidad: rs.getInt(string: "cantidad"));
                carrito.setPrecio(precio: rs.getDouble(string: "precio"));

                carritos.add(e: carrito);
            }
            rs.close();
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
        return carritos;
    }
}

```

**Nota:** La figura muestra la implementación del método `getCarritosByUserId()` en la clase `CarritoDAO`. Este método ejecuta una consulta SQL que obtiene los detalles de los carritos asociados a un usuario específico, utilizando una conexión a la base de datos. Los resultados se almacenan en una lista de objetos `Carrito`, la cual es retornada al final del proceso. La implementación incluye manejo de excepciones para evitar interrupciones en caso de errores de SQL. **Fuente:** Elaboración Propia.

**3. Refactor:** La última etapa del ciclo de TDD se centra en mejorar el código sin cambiar su comportamiento externo, limpiando la estructura y optimizando la lógica del código.

- En esta etapa, el código del método `getCarritosByUserId(int userId)` fue refactorizado para mejorar su legibilidad, garantizar el cierre adecuado de recursos y manejar errores de manera más eficiente.

- Se mantuvo la funcionalidad y la prueba continuó pasando, lo cual garantiza que la calidad del código se mejoró sin afectar su funcionamiento.

**Figura 23:**  
Implementación Mejorada del Método `getCarritosById()` en `CarritoDAO`

```

public List<Carrito> getCarritosById(int userId) {
    List<Carrito> carritos = new ArrayList<>();
    PreparedStatement ps;
    ResultSet rs;

    String sql = "SELECT ca.carrito_id, ca.cantidad, "
        + "f.flor_id, f.precio "
        + "FROM carrito ca "
        + "INNER JOIN flores f ON ca.flor_id = f.flor_id "
        + "WHERE ca.user_id = ?";

    try {
        ps = cnx.prepareStatement(string: sql);
        ps.setInt(1, id: userId);
        rs = ps.executeQuery();

        while (rs.next()) {
            Carrito carrito = new Carrito();
            carrito.setIdCarrito(idCarrito: rs.getInt(string: "carrito_id")); // Actualiza el nombre de la columna aquí
            carrito.setFlorId(florId: rs.getInt(string: "flor_id"));
            carrito.setCantidad(cantidad: rs.getInt(string: "cantidad"));
            carrito.setPrecio(precio: rs.getDouble(string: "precio"));

            carritos.add(carrito);
        }
        rs.close();
    } catch (SQLException ex) {
        System.err.println("Error al obtener los carritos: " + ex.getMessage());
    }
    return carritos;
}

```

*Nota: La figura muestra una versión actualizada del método `getCarritosById()` de la clase `CarritoDAO`.*

*Este método consulta los carritos asociados a un usuario específico mediante una consulta SQL con una conexión preparada. Además, el manejo de excepciones se mejora con mensajes de error más claros usando `System.err.println`. Los resultados se almacenan en una lista de objetos `Carrito`, asegurando que la información relevante del carrito, como la cantidad y el precio, se recupere correctamente. Fuente: Elaboración Propia.*

#### 5.1.3.2 Mejora de la Seguridad del Proyecto

El uso de TDD en todo el proyecto mejora la seguridad de varias formas:

- **Verificación constante de la lógica de negocio:** Las pruebas escritas antes de implementar la funcionalidad aseguran que cada parte del sistema cumpla con los requisitos desde el inicio, lo cual minimiza la posibilidad de errores y vulnerabilidades.
- **Manejo de excepciones:** TDD fomenta la inclusión de pruebas para casos extremos, lo cual garantiza un manejo adecuado de excepciones, protegiendo al sistema de fallos o ataques de inyección de datos.

- **Modularidad y menos errores:** La refactorización guiada por pruebas promueve un código más limpio y modular, lo que reduce la probabilidad de errores y mejora la robustez del sistema.

#### *5.1.3.3 Uso de JUnit para manejo de errores y validación de métodos*

JUnit es una herramienta de pruebas unitarias ampliamente utilizada en Java para verificar el funcionamiento del código de manera automatizada. Permite identificar problemas de lógica y errores en el código mediante la ejecución de casos de prueba que validan la funcionalidad de métodos específicos. El uso de JUnit en el desarrollo facilita la identificación y corrección de errores, asegurando la calidad del código y su correcto comportamiento

##### **5.1.3.3.1 Implementación de Pruebas en Detalle\_PedidoDAO**

En este caso, se utilizó JUnit para probar el método `getDetallePedidosByPedidoId` de la clase `Detalle_PedidoDAO`, que tiene como objetivo obtener una lista de detalles de pedido de la base de datos en función del ID del pedido y del usuario. Al implementar esta prueba unitaria, se esperaba garantizar que el método devolviera resultados correctos y manejara adecuadamente los errores.

##### **5.1.3.3.2 Identificación de Problemas con Junit**

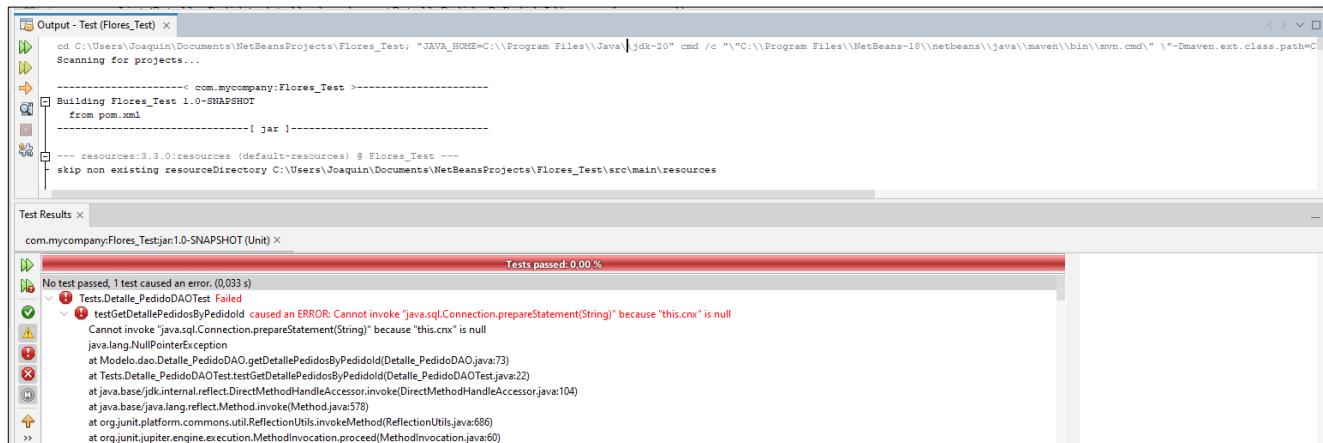
Durante la primera ejecución de la prueba con JUnit, se produjo un error relacionado con la conexión a la base de datos. El mensaje de error indicaba que la variable de conexión (`cnx`) era `null`, lo que causaba un fallo al intentar ejecutar el método `prepareStatement()`. Esto reflejó que la conexión no se estaba inicializando correctamente, lo cual era un problema crítico, ya que impedía la ejecución normal del método y, por lo tanto, la recuperación de los datos requeridos.

### 5.1.3.3.3 Error Inicial Detectado

- **Error:** La conexión a la base de datos (`Connection`) era `null` en el método `getDetallePedidosByPedidoId`.
- **Motivo del Error:** La conexión no estaba establecida antes de invocar el método, lo que generaba un `NullPointerException` al intentar crear un `PreparedStatement`.
- **Información proporcionada por JUnit:**

JUnit proporcionó un informe claro del error, señalando la línea específica del código donde se produjo el fallo y mostrando la traza completa del error, lo que facilitó la identificación precisa del problema.

**Figura 24:**  
Pruebas unitarias de errores con JUnit



**Nota:** La imagen muestra los resultados de una prueba fallida realizada con JUnit, enfocada en el método `getDetallePedidosByPedidoId`. El error principal se debe a que la conexión a la base de datos (`cnx`) no fue inicializada correctamente, lo que causó un `NullPointerException` al intentar preparar un `PreparedStatement`. Esta evidencia refleja la importancia de manejar adecuadamente las conexiones en las pruebas unitarias para evitar interrupciones en la ejecución.

Fuente: Elaboración Propia.

### 5.1.3.3.4 Solución Implementada

Basándonos en la información detallada proporcionada por JUnit, se realizaron ajustes en la clase para garantizar que la conexión se inicializara de manera correcta antes de ejecutar el método `getDetallePedidosByPedidoId`. Esto implicó asegurar que la conexión a la base de datos estuviera disponible en el momento de la ejecución de la prueba, lo cual resolvió el problema de la conexión `null` y permitió la creación del `PreparedStatement` de manera exitosa.

Para garantizar la correcta inicialización de la conexión:

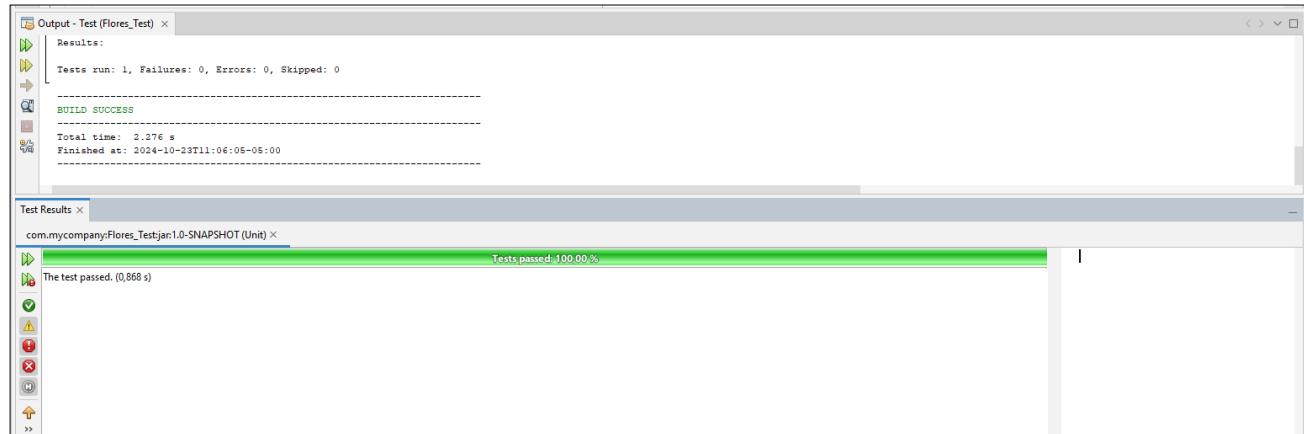
1. Se revisó y ajustó el proceso de configuración de la conexión, asegurando que se estableciera adecuadamente antes de la ejecución del método.
2. Se verificó la lógica de inicialización para evitar posibles conflictos que pudieran llevar a un estado `null` de la conexión.

#### 5.1.3.3.5 Resultados Finales de la Prueba

Después de aplicar las correcciones, se volvió a ejecutar la prueba con JUnit, y esta vez el resultado fue exitoso:

- **Pruebas Pasadas:** JUnit mostró que todas las pruebas unitarias se ejecutaron correctamente, logrando una tasa de éxito del 100%.
- **Validación Completa:** La prueba verificó que el método devolviera los resultados esperados y que la conexión funcionara adecuadamente durante toda la ejecución del método.

**Figura 25:**  
Resultado de Ejecución de Pruebas Unitarias en JUnit



**Nota:** La imagen muestra la ejecución exitosa de pruebas unitarias con Maven. En los resultados se observa que se ejecutó 1 prueba sin fallos, errores ni pruebas omitidas. La salida del proceso indica "BUILD SUCCESS", confirmando que la compilación y las pruebas fueron exitosas en 2.276 segundos. La barra verde en los resultados indica un 100% de éxito en la ejecución de la prueba. **Fuente:** Elaboración Propia.

## 5.1.4 Operaciones CRUD

Las operaciones CRUD son fundamentales en cualquier sistema que gestione datos, ya que representan las acciones básicas para interactuar con una base de datos. Cada operación permite crear, leer, actualizar y eliminar registros, asegurando el control completo sobre la información almacenada.

A continuación, se explicarán las implementaciones de las operaciones CRUD en el sistema de comercio digital para la floristería "El y Ella Detalles", destacando su importancia en la gestión eficiente de productos, pedidos y usuarios.

### 5.1.4.1 Método update() para Actualización de Registros

**Figura 26:**

Implementación del Método update() en FloresDAO

```
// Método para actualizar una flor existente
public boolean update(Flores f) {
    String cadSQL = "UPDATE flores SET nombre = ?, descripcion = ?, precio = ?, stock = ?, imagen_url = ? WHERE flor_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setString(i: 1, string: f.getNombre());
        ps.setString(i: 2, string: f.getDescripcion());
        ps.setDouble(i: 3, d: f.getPrecio());
        ps.setInt(i: 4, il: f.getStock());
        ps.setString(i: 5, string: f.getImagenUrl());
        ps.setInt(i: 6, il: f.getFlorId());
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al actualizar flor: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** La figura muestra la implementación del método update() para actualizar los datos de una flor en la base de datos. Este método utiliza una consulta SQL preparada para modificar los atributos de una flor específica, como su nombre, descripción, precio, stock e imagen. En caso de error, se captura la excepción y se muestra un mensaje informativo con System.out.println(), además de imprimir la traza del error para facilitar la depuración. El método retorna un valor booleano que indica si la actualización fue exitosa o no.

Fuente: Elaboración Pronia.

#### 5.1.4.2 Método get() para Recuperación de Registros

Figura 27:

Implementación del Método get() en FloresDAO

```
public Flores get(int id) {
    Flores f = null;
    PreparedStatement ps;
    ResultSet rs;
    String cadSQL = "SELECT * FROM flores WHERE flor_id = ?";
    try {
        ps = cnx.prepareStatement(string: cadSQL);
        ps.setInt(i: 1, il: id);
        rs = ps.executeQuery();

        if (rs.next()) {
            f = new Flores(
                florId: rs.getInt(string: "flor_id"),
                nombre: rs.getString(string: "nombre"),
                descripcion: rs.getString(string: "descripcion"),
                precio: rs.getDouble(string: "precio"),
                stock: rs.getInt(string: "stock"),
                imagenUrl: rs.getString(string: "imagen_url")
            );
        }
        rs.close();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return f;
}
```

**Nota:** La figura muestra la implementación del método get() para recuperar los datos de una flor específica desde la base de datos utilizando su ID. Este método realiza una consulta SQL preparada y, si encuentra el registro correspondiente, crea un objeto Flores con los valores obtenidos de la base de datos, como su nombre, descripción, precio, stock y URL de la imagen.

Fuente: Elaboración Propia.

#### 5.1.4.3 Método eliminar() para Eliminación de Registros

Figura 28:

Implementación del Método eliminar() en FloresDAO

```
// Método para eliminar una flor por ID
public boolean eliminar(int id) {
    String cadSQL = "DELETE FROM flores WHERE flor_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setInt(i: 1, il: id);
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al eliminar flor: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** La figura muestra la implementación del método eliminar() para borrar una flor de la base de datos mediante su ID. El método utiliza una consulta SQL preparada para asegurar la correcta eliminación del registro. Si la operación es exitosa, retorna true; en caso de error, captura la excepción y muestra un mensaje informativo utilizando System.out.println(), además de imprimir la traza del error para facilitar su depuración. Retorna false si la eliminación no se completa.

Fuente: Elaboración Propia.

#### 5.1.4.4 Método agregar() para Inserción de Registros

**Figura 29:**  
Implementación del Método agregar() en FloresDAO

```
public boolean agregar(Flores f) {
    String cadSQL = "INSERT INTO flores (nombre, descripcion, precio, stock, imagen_url) VALUES (?, ?, ?, ?, ?)";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setString(i: 1, string: f.getNombre());
        ps.setString(i: 2, string: f.getDescripcion());
        ps.setDouble(i: 3, d: f.getPrecio());
        ps.setInt(i: 4, i: f.getStock());
        ps.setString(i: 5, string: f.getImagenUrl());
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al agregar flor: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** La figura muestra la implementación del método agregar() para insertar un nuevo registro de flor en la base de datos. Utiliza una consulta SQL preparada para añadir los datos proporcionados, como nombre, descripción, precio, stock y URL de imagen. Si la inserción se realiza correctamente, el método retorna true; en caso de error, se captura la excepción, se imprime un mensaje informativo con System.out.println() y la traza del error para facilitar la depuración, retornando false si la operación no se completa.

Fuente: Elaboración Propia.

#### 5.1.5 Medidas de seguridad integradas

Se han incorporado medidas de seguridad en cada etapa del desarrollo para proteger la información sensible y garantizar la confiabilidad del sistema. Esto incluye manejo seguro de datos y controles para prevenir accesos no autorizados.

##### 5.1.5.1 Medidas de Seguridad contra Inyección SQL

Para proteger el acceso a la base de datos, se utilizan *Prepared Statements*, evitando que los datos de entrada se interpreten como parte de la consulta SQL, como se observa en el método save() de RegisterDAO.java:

```
String cadSQL = "INSERT INTO register (nombre, email,
contrasena, direccion, telefono) VALUES (?, ?, ?, ?, ?)";
try (PreparedStatement ps = cnx.prepareStatement(cadSQL)) {
    ps.setString(1, r.getNombre());
    ps.setString(2, r.getEmail());
    ps.setString(3, r.getContrasena());
    ps.setString(4, r.getDireccion());
    ps.setString(5, r.getTelefono());
    ps.executeUpdate();
}
```

**Figura 30:**  
Implementación del Método save() en RegisterDAO

```
public RegisterDAO() {
    cnx = new DBCexion().getConnection();
}

public boolean save(Register r) {
    String cadSQL = "INSERT INTO register (nombre, email, contrasena, direccion, telefono) VALUES (?, ?, ?, ?, ?)";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) { // try-with-resources para cerrar ps automáticamente
        ps.setString(i: 1, string: r.getNombre());
        ps.setString(i: 2, string: r.getEmail());
        ps.setString(i: 3, string: r.getContrasena());
        ps.setString(i: 4, string: r.getDireccion());
        ps.setString(i: 5, string: r.getTelefono());
        ps.executeUpdate();
        return true; // Registro exitoso
    } catch (SQLException ex) {
        System.out.println("Error al registrar usuario: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** La figura muestra la implementación del método save() en la clase RegisterDAO, encargado de registrar nuevos usuarios en la base de datos. Utiliza una consulta SQL preparada para insertar datos como nombre, email, contraseña, dirección y teléfono. Si la operación es exitosa, retorna true; en caso de error, se captura la excepción y se muestra un mensaje informativo junto con la traza del error para facilitar la depuración. **Fuente: Elaboración Propia.**

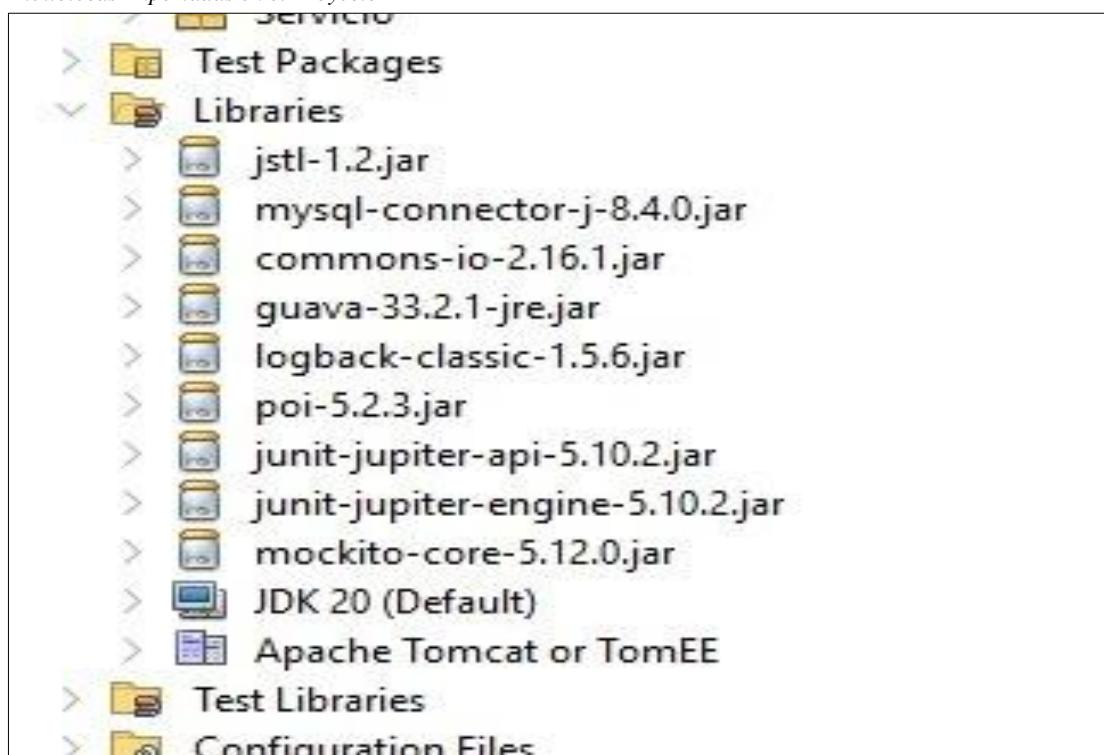
## 6 Uso de Recursos Java

En esta sección, se detalla el uso de diversas librerías de apoyo, las cuales mejoran la eficiencia, funcionalidad y seguridad del sistema. Estas herramientas permiten simplificar tareas complejas y garantizar un desempeño óptimo. A continuación, se describe el uso de las principales librerías utilizadas.

### 6.1 Librerías utilizadas en el proyecto

Cada librería seleccionada aporta funcionalidades específicas que optimizan distintas áreas del desarrollo del proyecto. A continuación, se presentan las más relevantes.

**Figura 31:**  
Bibliotecas Importadas en el Proyecto



**Nota:** La figura muestra las librerías utilizadas, como JSTL para JSP, MySQL Connector para la conexión con la base de datos, Guava y Commons IO para optimización, Logback para registros y POI para manejo de archivos Office, junto con JDK 20 y Apache Tomcat para la ejecución del proyecto.

Fuente: Elaboración Propia.

### 6.1.1 Google Guava – Optimización de estructuras de datos

Google Guava proporciona estructuras de datos avanzadas y algoritmos optimizados, facilitando la manipulación eficiente de la información del sistema.

#### 6.1.1.1 guava-33.2.1-jre.jar

- **Propósito:** Google Guava es una librería que ofrece utilidades avanzadas para manejar colecciones, cadenas, concurrencia y más.
- **Uso en el proyecto:** Se está utilizando para optimizar el manejo de colecciones y estructuras de datos en la capa de lógica de negocio, permitiendo una mejor organización y procesamiento de datos antes de enviarlos a la capa de vista o modelo. Además, sus herramientas de validación se usan para verificar datos de entrada del usuario en el controlador, asegurando que sean correctos antes de procesarlos.

## 6.1.2 Apache POI – Manipulación de archivos Excel

Esta librería permite generar y leer archivos Excel, lo que resulta útil para la gestión de inventarios y reportes de ventas en la floristería.

### 6.1.2.1 *poi-5.2.3.jar*

- **Propósito:** Apache POI es una librería para trabajar con documentos de Microsoft Office, especialmente hojas de cálculo de Excel (.xls y .xlsx).
- **Uso en el proyecto:** Se está utilizando para generar reportes de ventas y pedidos en formato Excel, lo que permite a los administradores exportar datos de manera fácil y efectiva. Además, la librería facilita la carga de datos desde archivos Excel para actualizar información masiva en la base de datos.

## 6.1.3 Apache Commons – Funciones comunes para manipulación de datos

Apache Commons ofrece herramientas reutilizables para operaciones frecuentes como validaciones y conversiones de datos, optimizando el código.

### 6.1.3.1 *commons-io-2.16.1.jar*

- **Propósito:** Apache Commons IO es una librería de utilidades para trabajar con operaciones de entrada/salida (I/O) en Java, proporcionando métodos para manipular y leer archivos, flujos de datos, y sistemas de archivos de manera eficiente.
- **Uso en el proyecto:** Se está utilizando para manejar la carga y descarga de archivos en el sistema, permitiendo a los usuarios subir imágenes de productos o descargar reportes en formato PDF desde la aplicación. La librería simplifica la manipulación de flujos de entrada y salida de datos.

## 6.1.4 Logback – Sistema de registro de logs

Logback permite registrar eventos importantes durante la ejecución del sistema, facilitando la monitorización y resolución de errores.

#### *6.1.4.1 logback-classic-1.5.6.jar*

- **Propósito:** Logback es una librería de logging (registro de eventos) para Java que permite registrar mensajes en la consola, archivos o sistemas de gestión de registros.
- **Uso en el proyecto:** Se utiliza para registrar eventos del sistema, como accesos de usuarios, errores en el manejo de excepciones, y operaciones críticas, facilitando la auditoría y depuración del sistema. La configuración de Logback permite ajustar los niveles de registro (e.g., debug, info, warn, error) según el entorno (desarrollo o producción).

## 7 Uso de Control de Versiones

El uso de un sistema de control de versiones como GitHub ha sido esencial para gestionar los cambios y avances entre el Avance 2 y el Avance 3 del proyecto. A continuación, se detalla cómo GitHub, junto con el entorno de desarrollo integrado (IDE) NetBeans, ha facilitado la colaboración entre los miembros del equipo, manteniendo una documentación clara y organizada de los cambios realizados en cada fase del desarrollo. A continuación, se detalla cómo se ha implementado el control de versiones en este proyecto.

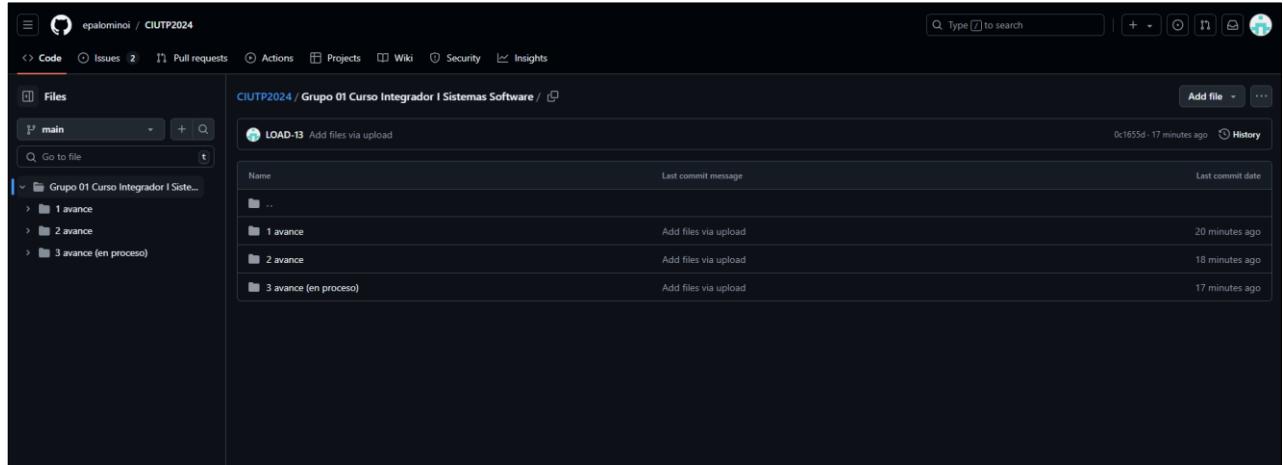
### 7.1 Estructura del Repositorio

El repositorio en GitHub se ha estructurado en carpetas organizadas por etapas de avance, como se puede ver en a continuación:

- **1er Avance:** Incluye archivos y documentación inicial del proyecto.
- **2do Avance:** Contiene documentación más detallada y archivos relacionados con la segunda etapa del desarrollo, tales como presentaciones, archivos de base de datos y otros documentos relevantes.
- **3er Avance (en proceso):** Actualmente, esta carpeta está en proceso de actualización con nuevos entregables.

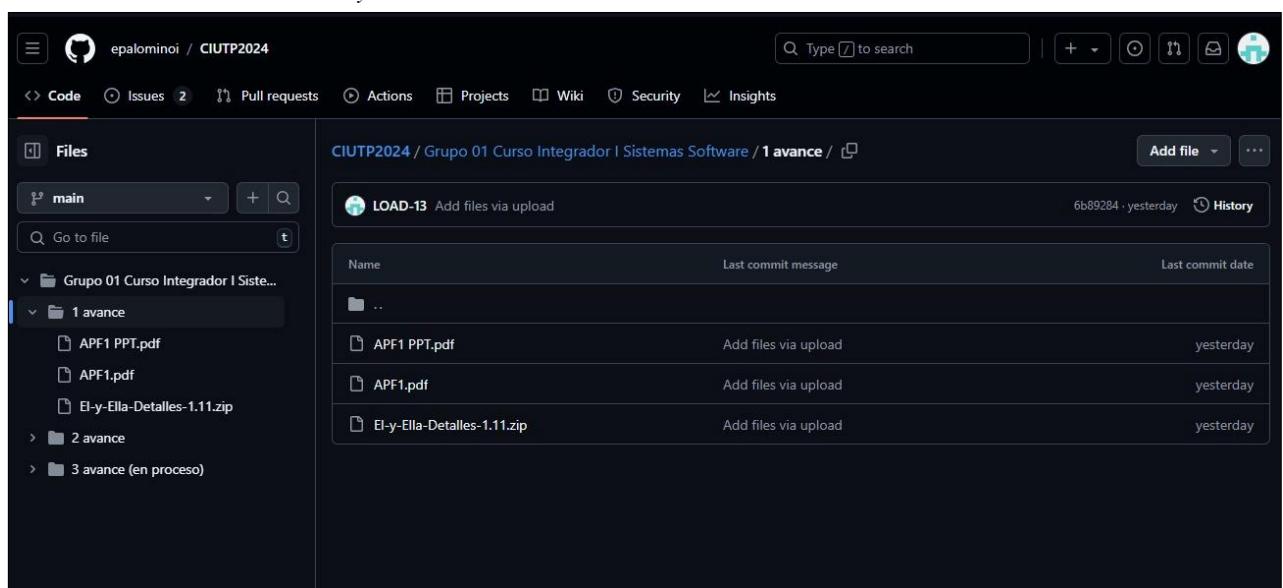
Cada carpeta representa una fase de desarrollo en la que se almacena la evidencia correspondiente, lo que permite un seguimiento claro y detallado de la evolución del proyecto.

**Figura 32:**  
Organización del Repositorio en GitHub



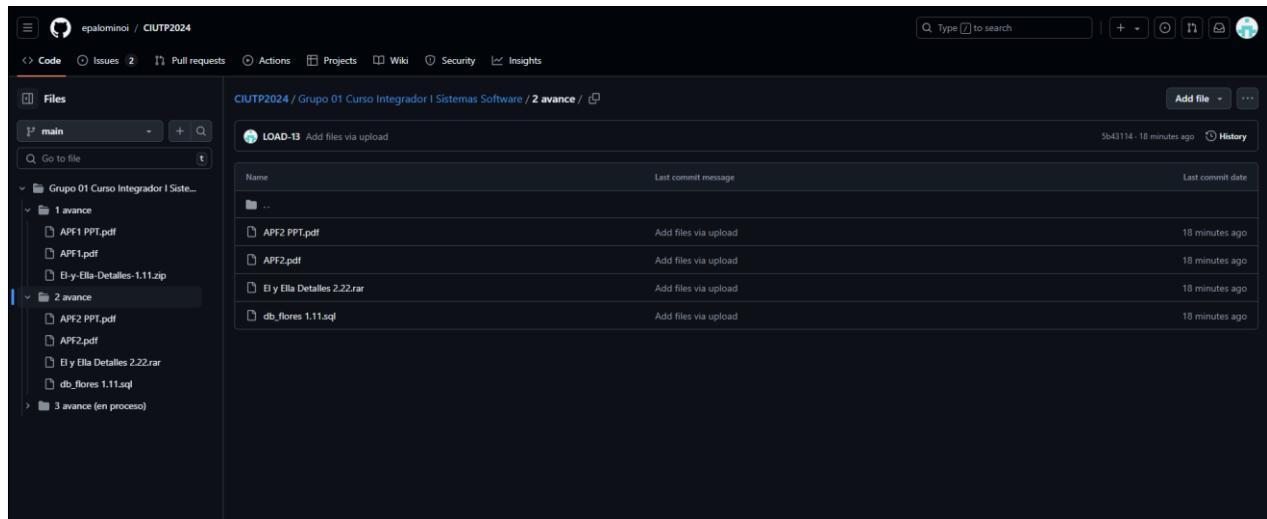
**Nota:** La figura muestra la estructura del repositorio en GitHub utilizado para gestionar el proyecto del curso integrador. El repositorio está organizado en carpetas correspondientes a los avances del proyecto, con cada carpeta conteniendo los archivos relevantes para esa fase. Esta organización permite un seguimiento claro del progreso y facilita la colaboración entre los miembros del equipo mediante commits detallados. **Fuente: Elaboración Propia.**

**Figura 33:**  
Archivos del Primer Avance del Proyecto en GitHub



**Nota:** La figura muestra los archivos correspondientes al primer avance del proyecto subidos al repositorio de GitHub. Incluye documentos en PDF, como presentaciones y reportes, además de un archivo comprimido con recursos adicionales. Esta organización permite un control eficiente del progreso y facilita la colaboración mediante la gestión de versiones en GitHub.  
**Fuente: Elaboración Propia.**

**Figura 34:**  
Archivos del Segundo Avance en GitHub



**Nota:** La figura muestra los archivos subidos para el segundo avance del proyecto en el repositorio de GitHub. Los documentos incluyen presentaciones en PDF, archivos comprimidos con el desarrollo del proyecto, y scripts SQL para la gestión de la base de datos. Esta estructura facilita el control de versiones y la trazabilidad de cada fase del proyecto, permitiendo que los miembros del equipo colaboren eficientemente. **Fuente: Elaboración Propia.**

## 7.2 Integración de GitHub con NetBeans

Una de las principales ventajas de usar GitHub en conjunto con NetBeans es la integración directa que ofrece este IDE. Esta conexión permite a los desarrolladores realizar acciones de control de versiones (como commits, pushes y merges) directamente desde NetBeans, lo que hace que el proceso sea más eficiente y menos propenso a errores. De esta manera, cada desarrollador puede trabajar en su rama individual y sincronizar sus cambios con el repositorio remoto en GitHub sin salir del entorno de desarrollo.

## 7.3 Flujo de Trabajo Colaborativo con Ramas Individuales

En la imagen presentada, se puede observar el uso de ramas individuales para cada miembro del equipo. Esta estrategia de trabajo ofrece varias ventajas:

- **Trabajo independiente:** Cada integrante puede desarrollar y probar sus características en una rama propia sin interferir con el código de otros.

- **Evitación de conflictos:** Al trabajar en ramas separadas, se minimiza el riesgo de conflictos en el código principal (master), manteniendo la integridad del proyecto.
- **Facilidad de fusión:** Una vez que los cambios han sido verificados, se fusionan con la rama principal mediante pull requests. Este proceso garantiza que el código agregado haya sido revisado y cumpla con los estándares de calidad del proyecto.

#### 7.4 Documentación y Trazabilidad de Cambios

El historial de commits en GitHub, como se muestra en la imagen, proporciona una visión clara y detallada de cada modificación realizada en el proyecto. Cada commit está acompañado de un mensaje descriptivo, lo que facilita entender qué cambios se han realizado, quién los ha hecho y en qué momento. Esto es clave para:

- **Auditar el progreso:** La capacidad de revisar los cambios realizados a lo largo del tiempo ayuda a identificar posibles problemas o áreas de mejora.
- **Responsabilidad individual:** Al documentar cada cambio con el usuario correspondiente, se mantiene la transparencia en el desarrollo y se puede atribuir el trabajo a cada miembro del equipo.

**Figura 35:**  
Historial de commits en repositorio colaborativo

The screenshot shows a GitHub repository's commit history. The interface includes filters for 'All branches', 'All activity', 'All users', and 'All time'. The commits are listed in chronological order, with the most recent at the top. Each commit entry contains the author's name, the branch it was pushed to, the commit hash, and a timestamp. The commits are categorized into several sections, such as 'Implementación del paquete servicio con la clase DBConexion y otra cl...', 'Implementación del modelo dto', and 'Implementación del Controlador'. Some commits include descriptive messages like 'Primer implementacion de una interfaz grafica para el perfil de usuario'.

Commit Message	Author	Branch	Hash	Timestamp
Implementación del paquete servicio con la clase DBConexion y otra cl...	LOAD-13	master	b0db9d8...3746de3	14 seconds ago
Implementación del paquete servicio con la clase DBConexion y otra cl...	LOAD-13	Joaquin_Loa	0b34c79...3746de3	47 seconds ago
implementación del modelo dto	Borderline06	master	5b521fb...b0db9d8	6 minutes ago
Implementación del modelo dto	Borderline06	Jorge_Membrillo	b0db9d8	6 minutes ago
Implementación del modelo dao	Kiaraz1052001	master	b1d4538...5b521fb	10 minutes ago
Implementación del modelo dao	Kiaraz1052001	Kiana_Santti	5b521fb	11 minutes ago
Implementación del Controlador	Sandrotic22	master	0b34c79...b1d4538	19 minutes ago
Implementación del Controlador	Sandrotic22	Sandro_Delacruz	b1d4538	20 minutes ago
Implementación de las Interfaces de Usuario Principales y su codigo c...	LOAD-13	master	d5560e8...0b34c79	25 minutes ago
Implementación de las Interfaces de Usuario Principales y su codigo c...	LOAD-13	Joaquin_Loa	4d20f1b...0b34c79	26 minutes ago
*****Detalles de pedidos, productos y edición form*****	iTaichi-bit	master	4d20f1b...d5560e8	41 minutes ago
*****Detalles de pedidos, productos y edición form*****	iTaichi-bit	Jefferson_Penia	d5560e8	42 minutes ago
Primera implemetacion de una interfaz grafica para el perfil de usuario	LOAD-13	master	2d3a93e...4d20f1b	57 minutes ago
Primera implemetacion de una interfaz grafica para el perfil de usuario	LOAD-13	Joaquin_Loa	4d20f1b	1 hour ago
Iniciando repositorio	Borderline06	master	2d3a93e	1 hour ago

**Nota:** La imagen refleja las contribuciones de los desarrolladores del proyecto colaborativo, evidenciando la implementación de múltiples componentes como controladores, interfaces de usuario, y modelos DAO y DTO. Cada commit está asociado a un autor, un mensaje descriptivo del cambio realizado y un timestamp, lo que permite el seguimiento detallado del progreso del desarrollo. **Fuente:** Elaboración Propia.

## 7.5 Evidencia de Uso

En las capturas de pantalla proporcionadas, se observa el repositorio del proyecto en GitHub, el cual contiene carpetas organizadas por avances y archivos asociados a cada etapa de desarrollo. Esta estructura permite demostrar de manera clara el cumplimiento del estándar esperado en la rúbrica, ya que cada carpeta representa una etapa específica del proyecto con la evidencia correspondiente.

### 7.5.1 División del Trabajo en el Proyecto

Para mantener una estructura organizada y eficiente en el desarrollo del proyecto, se dividieron las tareas entre los miembros del equipo, asignando a cada uno un área específica del proyecto. La colaboración se llevó a cabo a través de ramas individuales en GitHub, lo que permitió un desarrollo simultáneo y autónomo de las funcionalidades asignadas a cada integrante.

#### 1. Jefferson Pernía - Implementación de Detalles de Pedidos, Productos y Edición

- **Descripción:** Jefferson fue responsable de desarrollar la funcionalidad relacionada con los detalles de pedidos, productos y la edición de formularios.
- **Contribuciones Visualizadas:** En la imagen correspondiente, se observa un commit que muestra la implementación de la vista para la gestión de pedidos y productos en la aplicación, con un total de 17 archivos modificados y un incremento significativo en el número de líneas de código.

**Figura 36:**

Detalle de Commit: 'Detalles de pedidos, productos y edición form'

The screenshot shows a GitHub commit page for a repository named 'Borderline06 / El-y-Ella-Detalles'. The commit was made by 'Jefferson Pernia' 11 hours ago. It has 17 files changed, with 3649 additions and 0 deletions. The main file shown is 'Vista/DetallesDePedidos/DetallePedidoRosa.jsp'. The code editor displays the following snippet from the JSP file:

```
*****Detalles de pedidos, productos y edición form*****  
@@ -0,0 +1,392 @@  
1 + <!--  
2 + Document : DetallePedidoRosa  
3 + Created on : 23 sept 2024, 13:51:51  
4 + Author : Joaquin  
5 + --%>  
6 +  
7 + <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  
8 + <!DOCTYPE html>  
9 + <html lang="es">  
10 + <head>  
11 + <meta charset="UTF-8">
```

At the bottom right of the code editor, there is a note: 'Activar Windows' and 'Ve a Configuración para activar Windows.'

**Nota:** La imagen muestra un commit específico en un proyecto alojado en GitHub, realizado por Jefferson Pernia. El commit corresponde a la edición y creación de vistas en el proyecto, como se observa en el archivo Vista/DetallesDePedidos/DetallePedidoRosa.jsp. Este archivo, escrito en JSP (JavaServer Pages), incluye el uso de bibliotecas como JSTL para la generación dinámica de contenido HTML. Fuente: Elaboración Propia.

## 2. Joaquín Loa - Implementación del Paquete de Conexión y las Interfaces de Usuario

- **Conexión a la Base de Datos:** Joaquín se encargó de implementar la clase DBConexion y las pruebas para la conexión, asegurando que la base de datos funcione correctamente con el sistema.
- **Contribuciones Visualizadas:** En la imagen de su commit se muestra la creación de la clase de conexión con un total de 2 archivos modificados.

**Figura 37:**  
Commit: Implementación del paquete servicio y clase de conexión

The screenshot shows a GitHub commit page for a repository named 'Borderline06 / El-y-Ella-Detalles'. The commit message is 'Implementacion del paquete servicio con la clase DBConexion y otra clase para hacer el Test de Conexion'. It was authored and committed by 'Joaquin Loa' 10 hours ago. The commit has 2 files changed (+58 -0 lines) and 1 parent commit 'b0db9d8'. The changes are shown in a diff view for 'Servicio/DBConexion.java', which contains Java code for database connection management. A note on the right says 'Activar Windows' and 'Ve a Configuración para activar Windows.'

**Nota:** Commit realizado por Joaquín Loa, agregando 41 líneas de código. Incluye la implementación de la clase DBConexion.java para gestionar la conexión a base de datos y una clase de prueba para validarla.

**Fuente:** Elaboración Propia.

- **Interfaces de Usuario:** También fue responsable de diseñar las interfaces de usuario y de agregar el código CSS y JavaScript asociado.
- **Contribuciones Visualizadas:** La imagen muestra la implementación de 79 archivos de la vista, con un enfoque en la interfaz de administración y las funcionalidades de usuario.

**Figura 38:**

*Commit: Implementación de interfaces de usuario y estilos*

The screenshot shows a GitHub commit page for a repository named 'Borderline06 / El-y-Ella-Detalles'. The commit was made by 'Joaquin Loa' 10 hours ago. It has 79 files changed, with +5800 -0 lines changed. The main file shown is 'Vista/Administrar.jsp', which contains JSP code. A note on the right side of the commit details says 'Activar Windows' and 'Ve a Configuración para activar Windows.'

```

    1 + <%-- 
    2 + Document : Administrar.jsp
    3 + Created on : 17 sept 2024, 18:25:02
    4 + Author : Joaquin
    5 + --%>
    6 +
    7 + <%@page import="java.sql.Connection"%>
    8 + <%@page import="java.sql.Statement"%>
    9 + <%@page import="java.sql.ResultSet"%>
   10 + <%@page import="Servicio.DBConexion"%>
   11 + <%@page contentType="text/html" pageEncoding="UTF-8"%>
  
```

**Nota:** Commit de Joaquín Loa que añade 570 líneas en la vista Administrar.jsp y archivos CSS y JS, implementando interfaces principales para gestionar la aplicación.

**Fuente:** Elaboración Propia.

### 3. Jorge Membrillo - Implementación del Modelo DTO

- Descripción:** Jorge desarrolló las clases del modelo DTO (Data Transfer Object), que sirven para gestionar la transferencia de datos entre las capas de la aplicación.
- Contribuciones Visualizadas:** En su commit se observa la creación y modificación de 9 archivos relacionados con las clases de transferencia de datos.

**Figura 39:**

*Commit: Implementación del modelo DTO*

The screenshot shows a GitHub commit page for the same repository. The commit was made by 'Membrillo Gaitan Jorge' 10 hours ago. It has 9 files changed, with +721 -0 lines changed. The main file shown is 'dto/Carrito.java', which contains Java code. A note on the right side of the commit details says 'Activar Windows' and 'Ve a Configuración para activar Windows.'

```

    1 + /*
    2 + * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
    3 + * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
    4 + */
    5 + package Modelo.dto;
    6 +
    7 + /**
    8 + *
    9 + * @author Joaquin
   10 + */
   11 +
  
```

**Nota:** Commit realizado por Jorge Membrillo, añadiendo 721 líneas en múltiples archivos del paquete dto. Define objetos de transferencia de datos como Carrito.java, Pedidos.java y Pagos.java para estructurar la información entre capas del sistema. **Fuente:** Elaboración Propia.

#### 4. Kiara Santti - Implementación del Modelo DAO

- **Descripción:** Kiara se enfocó en el desarrollo del modelo DAO (Data Access Object), que permite la interacción con la base de datos mediante las operaciones CRUD (Crear, Leer, Actualizar, Eliminar).
- **Contribuciones Visualizadas:** La imagen muestra la implementación de 10 archivos en el paquete DAO, donde se añaden funcionalidades de acceso y manejo de datos.

Figura 40:

Commit: Implementación del modelo DAO

The screenshot shows a GitHub commit page for a repository named 'Borderline06 / El-y-Ella-Detalles'. The commit was made by 'Kiara Santti' 10 hours ago. It is a 'master' branch commit with 1 parent, commit ID '5b521fb'. The commit message is 'Implementación del modelo dao'. The commit details show '10 files changed +1049 -0 lines changed'. One of the files shown is 'dao/CarritoDAO.java', which has 106 additions and 0 deletions. The code snippet for this file is as follows:

```
1 + /*
2 + * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3 + * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4 + */
5 + package Modelo.dao;
6 +
7 + import Servicio.DBConexion;
8 + import Modelo.dto.Carrito;
9 + import java.sql.*;
10 + import java.util.ArrayList;
```

A note on the right side of the code editor says 'Activar Windows' and 'Ve a Configuración para activar Windows.'

*Nota: Commit realizado por Kiara Santti, agregando 1049 líneas en varias clases del paquete dao. Implementa la lógica de acceso a datos con clases como CarritoDAO.java y PedidosDAO.java, conectando la base de datos con la capa de negocio. Fuente: Elaboración Propia.*

#### 5. Sandro De la Cruz - Implementación del Controlador

- **Descripción:** Sandro fue el encargado de desarrollar el controlador, que maneja la lógica de negocio de la aplicación y coordina la comunicación entre el modelo y la vista.
- **Contribuciones Visualizadas:** En la imagen, se puede ver que el commit incluyó 8 archivos modificados, representando la implementación de la lógica de control para distintas funcionalidades del sistema.

**Figura 41:**  
Commit: Implementación del Controlador

The screenshot shows a GitHub commit page for the repository "Borderline06 / El-y-Ella-Detalles". The commit was made by "Sandro De la Cruz" 10 hours ago on the "master" branch. The commit message is "Implementacion del Controlador". The commit details show 8 files changed, with 861 additions and 95 modifications. The main file shown is "Controlador/ControlCarrito.java". The code snippet for this file is as follows:

```

@@ -0,0 +1,101 @@
+
+ * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
+ * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
+
+ package Controlador;
+
+ import Modelo.dao.CarritoDAO;
+ import Modelo.dto.Carrito;
+ import javax.servlet.ServletException;
+ import javax.servlet.http.HttpServlet;

```

A tooltip on the right side of the code editor says "Activar Windows" and "Ve a Configuración para activar Windows."

**Nota:** Commit realizado por Sandro De la Cruz, añadiendo 861 líneas y modificando 95. Incluye clases como ControlCarrito.java y ControlPago.java, encargadas de gestionar las operaciones entre la vista y el modelo, siguiendo el patrón MVC. Fuente: *Elaboración Propia*.

## 8 Implementación de las Interfaces Gráficas

La tienda en línea "El y Ella Detalles" es un proyecto de comercio electrónico dedicado a la venta de arreglos florales. La implementación de la interfaz gráfica (UI/UX) de la plataforma busca ofrecer una experiencia de usuario intuitiva y amigable, cumpliendo con el 100% del alcance comprometido según los estándares del proyecto.

A continuación, se documenta cada sección de la interfaz gráfica con sus respectivas funcionalidades y conexión con la base de datos:

### 8.1 Diseño UX/UI del sistema

El diseño UX/UI se ha enfocado en facilitar la interacción del usuario, asegurando que las interfaces sean intuitivas y eficientes.

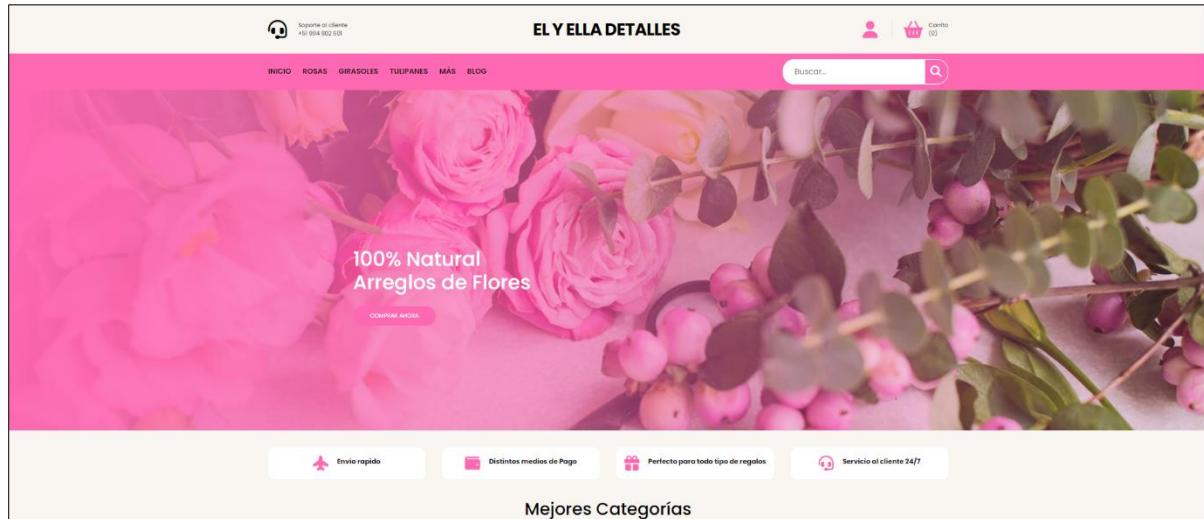
#### 8.1.1 Interfaces gráficas

Se han desarrollado pantallas que permiten al usuario gestionar productos, realizar pedidos y consultar reportes.

### 8.1.1.1 Página de Inicio

La primera vista que el usuario tiene al ingresar a la plataforma es la página de inicio. Esta interfaz presenta una imagen atractiva de bienvenida y permite al usuario acceder de manera rápida a las principales secciones del sitio. No tiene características funcionales específicas más allá de proporcionar una navegación inicial sencilla para el usuario.

**Figura 42:**  
Página de Inicio del Sistema de Comercio Digital



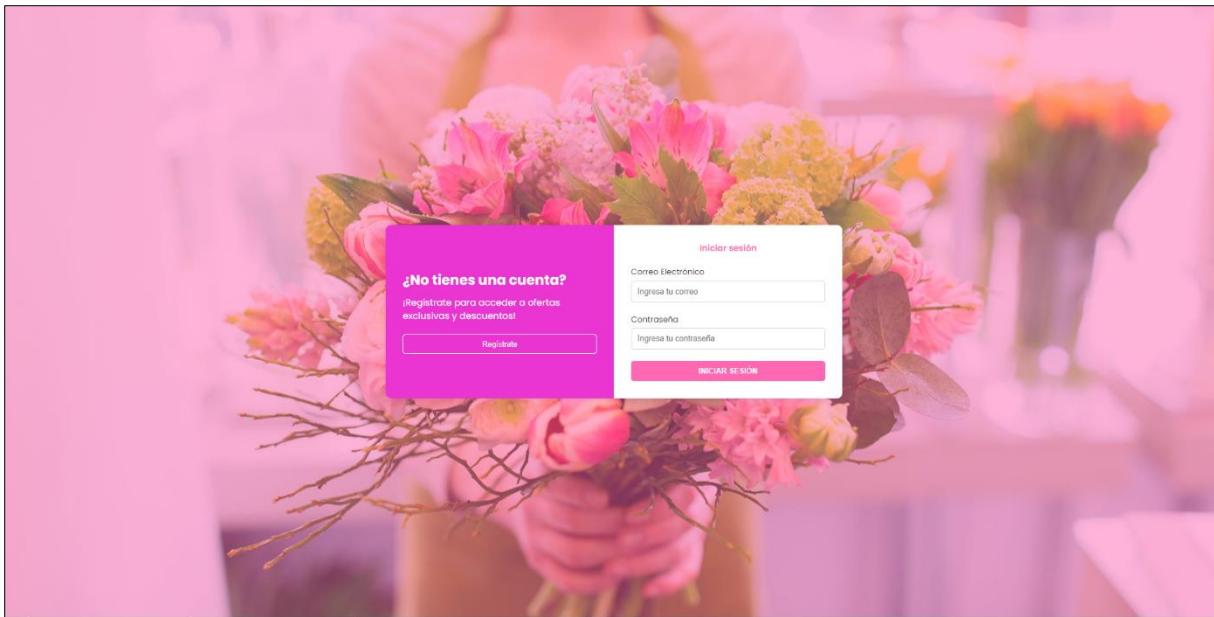
**Nota:** La figura muestra la página principal del sitio web "El y Ella Detalles", diseñada para ofrecer una experiencia visual atractiva y funcional. El encabezado contiene opciones de navegación hacia diferentes categorías de flores y funcionalidades del sitio, como el carrito de compras y el perfil del usuario. La interfaz destaca la promoción de arreglos florales naturales y facilita la búsqueda mediante una barra dedicada. Además, presenta íconos informativos sobre las ventajas del servicio, como envío rápido y atención al cliente 24/7, mejorando la experiencia del usuario.

Fuente: Elaboración Propia.

### 8.1.1.2 Formulario de Inicio de Sesión

El formulario de inicio de sesión permite a los usuarios autenticarse ingresando su correo electrónico y contraseña. Este formulario realiza una consulta a la base de datos para verificar la validez de las credenciales proporcionadas. Si los datos coinciden con un usuario registrado, se activa una sesión que almacena el `user_id` del usuario autenticado, permitiendo un acceso seguro y personalizado.

**Figura 43:**  
Página de Inicio de Sesión y Registro del Usuario

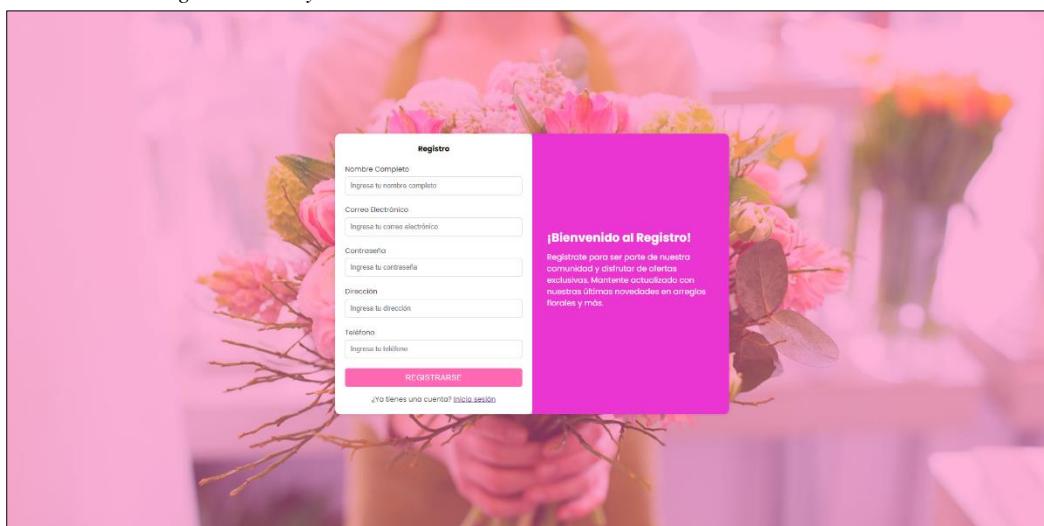


**Nota:** La figura muestra la interfaz de acceso del sistema "El y Ella Detalles", donde los usuarios pueden iniciar sesión o registrarse. La sección de registro invita a nuevos usuarios a unirse para recibir ofertas exclusivas, mientras que el área de inicio de sesión permite a los usuarios existentes ingresar sus credenciales. Esta interfaz está diseñada para ser intuitiva y atractiva, facilitando el acceso rápido a las funcionalidades del sitio y mejorando la experiencia del cliente desde el primer contacto. **Fuente:** Elaboración Propia.

#### 8.1.1.3 Formulario de Registro

El formulario de registro se utiliza para crear una nueva cuenta de usuario. Los campos solicitados incluyen nombre completo, correo electrónico, contraseña, dirección y teléfono. Al completar el formulario y enviarlo, se registran los datos en la base de datos, creando un nuevo usuario. Esta función permite a los usuarios acceder a ofertas exclusivas y realizar compras en la tienda.

**Figura 44:**  
Formulario de Registro en "El y Ella Detalles"

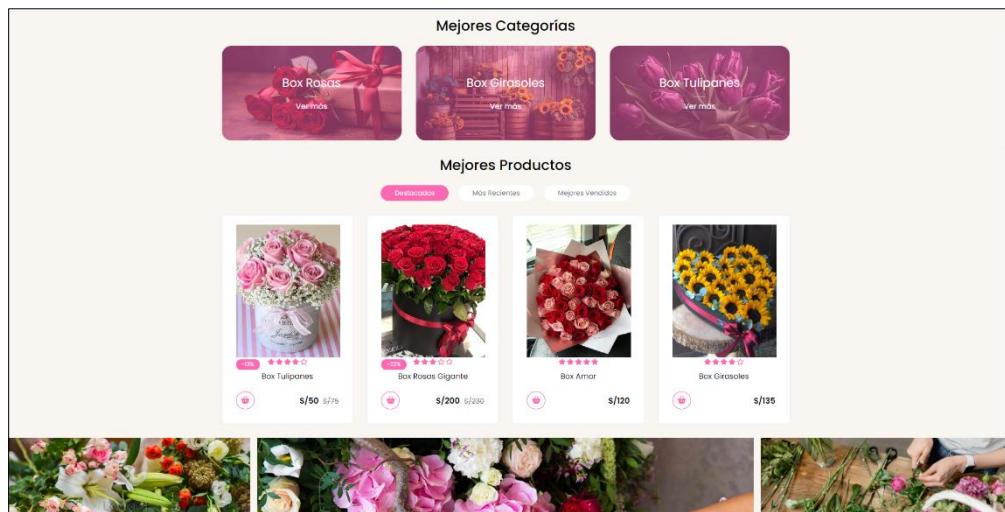


**Nota:** La figura muestra el formulario de registro del sistema "El y Ella Detalles". Los campos solicitados incluyen nombre completo, correo electrónico, contraseña, dirección y teléfono, lo que permite recopilar la información necesaria para gestionar las cuentas de los usuarios.  
**Fuente:** Elaboración Propia.

#### 8.1.1.4 Catálogo de Productos

El catálogo de productos muestra todos los arreglos florales disponibles para la venta, organizados en categorías como rosas, girasoles y tulipanes. Esta vista permite a los usuarios explorar las diferentes opciones de productos, con información básica como el nombre del producto, precio y una imagen. Cada producto está conectado a la base de datos, lo que permite que se muestren las últimas actualizaciones de stock y precios en tiempo real.

**Figura 45:**  
Visualización de Categorías y Productos en "El y Ella Detalles"



**Nota:** La figura muestra la sección de categorías y productos destacados del sitio "El y Ella Detalles". La interfaz ofrece una vista clara de las mejores categorías, como Box Rosas, Box Girasoles y Box Tulipanes, junto con opciones para explorar más detalles. Debajo, se presenta un catálogo de productos organizados por destacados, más recientes y vendidos, con precios visibles y opciones para descuentos. Este diseño permite a los usuarios navegar fácilmente por el inventario y seleccionar productos según sus preferencias. **Fuente:** Elaboración Propia.

#### 8.1.1.5 Detalle del Producto

Al hacer clic en un producto en el catálogo, el usuario es redirigido a una página de detalles del producto. En esta sección se muestra información detallada sobre el producto seleccionado, incluyendo descripción, precio, imágenes adicionales y políticas de entrega y devolución. Un elemento clave de esta interfaz es el botón "Añadir al carrito", que inserta el `product_id` y el `user_id` en la tabla del carrito en la base de datos, registrando así la selección del usuario para futuras compras.

**Figura 46:**  
Detalle de Producto en "El y Ella Detalles"

**Nota:** La figura muestra la página de detalle de un producto específico, en este caso, la "Caja de Amor". La interfaz presenta imágenes del producto junto con su descripción, precio, y opciones para seleccionar cantidad y añadir al carrito. También se incluyen secciones informativas sobre la entrega y las políticas de devoluciones, ofreciendo al usuario claridad en la compra. Este diseño optimiza la experiencia de usuario al brindar toda la información necesaria para tomar decisiones de compra informadas.

Fuente: Elaboración Propia.

#### 8.1.1.6 Formulario de Reseñas

Debajo de la sección de detalles del producto se encuentra el formulario de reseñas, donde los usuarios pueden enviar comentarios y calificaciones sobre el producto. El formulario recoge el texto de la reseña, la calificación (de 1 a 5 estrellas), el `user_id` y el `product_id`, almacenando toda la información en la base de datos. Esto permite generar un feedback valioso para otros usuarios interesados en el producto.

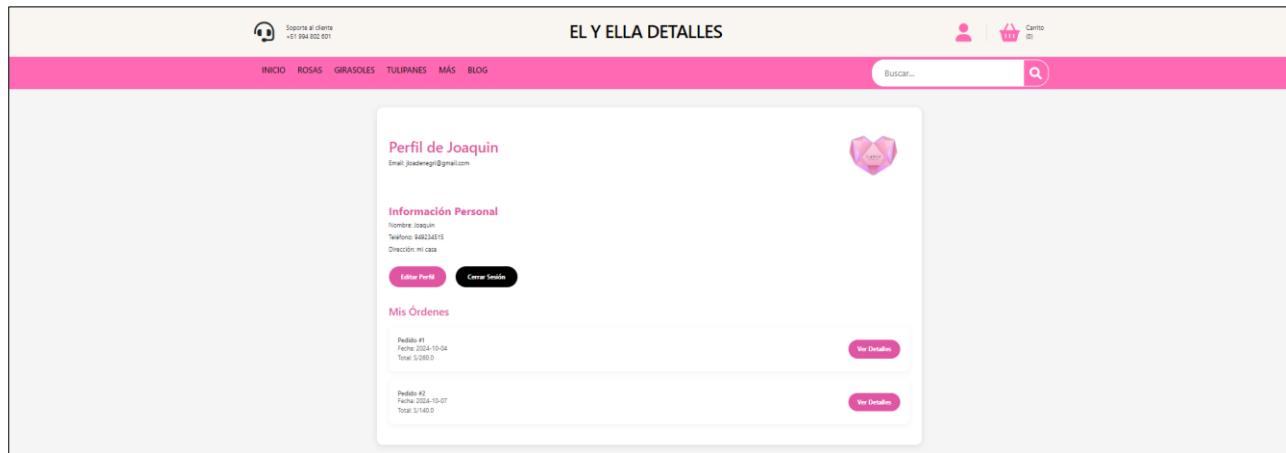
**Figura 47:**  
Área de Comentarios y Footer en "El y Ella Detalles"

**Nota:** La figura muestra la sección de comentarios y el pie de página del sitio web. Los usuarios pueden dejar reseñas calificando productos con estrellas y añadiendo comentarios, fomentando la interacción y retroalimentación. El pie de página proporciona información de contacto, enlaces a secciones importantes como políticas de privacidad, términos y condiciones, así como accesos rápidos a las cuentas de usuario y opciones de registro. Además, se incluyen íconos de redes sociales y métodos de pago aceptados, mejorando la accesibilidad y transparencia del sitio. Fuente: Elaboración Propia.

#### 8.1.1.7 Perfil de Usuario

El perfil de usuario muestra la información personal del usuario registrado, utilizando el `user_id` almacenado en la sesión para obtener los datos de la base de datos. Se incluyen detalles como nombre, correo electrónico, dirección y teléfono, además de un historial de pedidos realizados. La interfaz permite a los usuarios editar su información y ver el estado de sus pedidos.

**Figura 48:**  
Perfil de Usuario en "El y Ella Detalles"



**Nota:** La figura muestra la interfaz del perfil de usuario en el sitio web "El y Ella Detalles". En esta vista, el usuario puede acceder a su información personal, como teléfono, correo y dirección, con opciones para editar el perfil o cerrar sesión. También se presenta un historial de órdenes realizadas con detalles básicos y botones para ver más información. Esta interfaz proporciona una experiencia personalizada, facilitando la gestión de datos y el seguimiento de pedidos de manera intuitiva y organizada. **Fuente:** Elaboración Propia.

#### 8.1.1.8 Carrito de Compras

El carrito de compras muestra los productos que el usuario ha añadido para su compra. La interfaz realiza una consulta a la base de datos para obtener los productos vinculados al `user_id` del usuario autenticado. Los usuarios pueden modificar la cantidad de productos en el carrito o eliminarlos antes de proceder al pago.

**Figura 49:**  
Vista del Carrito de Compras en "El y Ella Detalles"

**Nota:** La figura muestra la interfaz del carrito de compras del sitio web "El y Ella Detalles". Los usuarios pueden ver los productos añadidos al carrito, modificar la cantidad de cada artículo y actualizar el total. En el panel lateral derecho, se presenta un resumen del pedido con el subtotal, costos de envío y total. Además, se proporciona la opción de "Realizar compra" para proceder al pago. Esta vista facilita la gestión de las compras, brindando claridad sobre los productos seleccionados y el costo total de la transacción. **Fuente: Elaboración Propia.**

#### 8.1.1.9 Formulario de Pago

El formulario de pago permite a los usuarios ingresar los datos de la tarjeta para procesar la compra. Los datos incluyen nombre del titular, número de tarjeta, fecha de expiración y CVV. Esta información se guarda de manera segura en la base de datos, cumpliendo con las normativas de seguridad requeridas para el manejo de información sensible. Al completar el formulario, se confirma la compra y se emite un recibo digital.

**Figura 50:**  
Vista del Formulario de Pago en "El y Ella Detalles"

**Nota:** La figura muestra el formulario de pago donde los usuarios ingresan los datos necesarios para completar su compra. Los campos incluyen información personal, dirección de envío y detalles de la tarjeta de crédito, como número, fecha de expiración y CVV. Esta interfaz facilita el proceso de pago al ofrecer un diseño claro y amigable, asegurando que los usuarios puedan completar sus transacciones de forma rápida y segura. **Fuente: Elaboración Propia.**

### 8.1.1.10 Pedidos

La vista de pedidos muestra un historial de todos los pedidos realizados por el usuario. Esta interfaz recoge los datos de la base de datos y muestra los pedidos que tienen el mismo `user_id` que el usuario autenticado, agrupando así los productos del carrito de compras en un solo pedido después de que se ha completado el proceso de pago.

Cada entrada en la lista de pedidos incluye información como la fecha de realización, el total pagado y botones para acceder a los detalles del pedido o para eliminar el pedido seleccionado. También hay una barra lateral de navegación que permite al usuario moverse fácilmente entre diferentes secciones relacionadas con sus pedidos, como detalles de pago, reembolsos y devoluciones, entre otras.

**Figura 51:**  
Vista del Historial de Pedidos en "El y Ella Detalles"

**Nota:** La figura muestra la interfaz de historial de pedidos del usuario, donde se detallan las compras realizadas con su fecha, descripción y total de cada pedido. Los usuarios pueden acceder a los detalles del pedido o eliminar registros específicos. La barra lateral ofrece opciones adicionales para gestionar pedidos, realizar pagos, ajustar direcciones y más, proporcionando una experiencia organizada y eficiente para el seguimiento de las transacciones.

Fuente: Elaboración Propia.

### 8.1.1.11 Detalle de Pedido

La interfaz de detalle de pedido proporciona una vista más específica de un pedido individual. En esta vista, se muestran los siguientes elementos:

- **Estado del Pedido:** Se indica si el pedido está finalizado o en proceso.
- **Fecha y Hora del Pedido:** Se muestra la fecha y la hora en que se realizó el pedido.

- **Dirección de Envío:** Se presentan los detalles de la dirección a la que se enviará el pedido.
- **Lista de Productos del Pedido:** Se despliega una tabla con los productos incluidos en el pedido, mostrando la cantidad y el precio de cada uno.
- **Opciones de Acciones:** Hay botones para volver a comprar (redirigiendo al detalle del producto) o rastrear el pedido.

Esta información detallada permite a los usuarios hacer un seguimiento preciso de cada pedido, lo que mejora la transparencia y la experiencia del usuario en la plataforma.

**Figura 52:**  
Vista de Detalle de un Pedido en "El y Ella Detalles"

**Nota:** La figura muestra la interfaz de detalle de un pedido finalizado. Se presenta la información completa del pedido, incluyendo el número de referencia, la fecha, el método de pago y la dirección de envío. También se detalla el contenido del pedido, con el nombre del producto, cantidad y precio. En la barra lateral, se muestra una imagen del producto adquirido. Esta interfaz permite al usuario revisar su compra en profundidad, brindando opciones adicionales como realizar una nueva compra o ver la política de devoluciones.

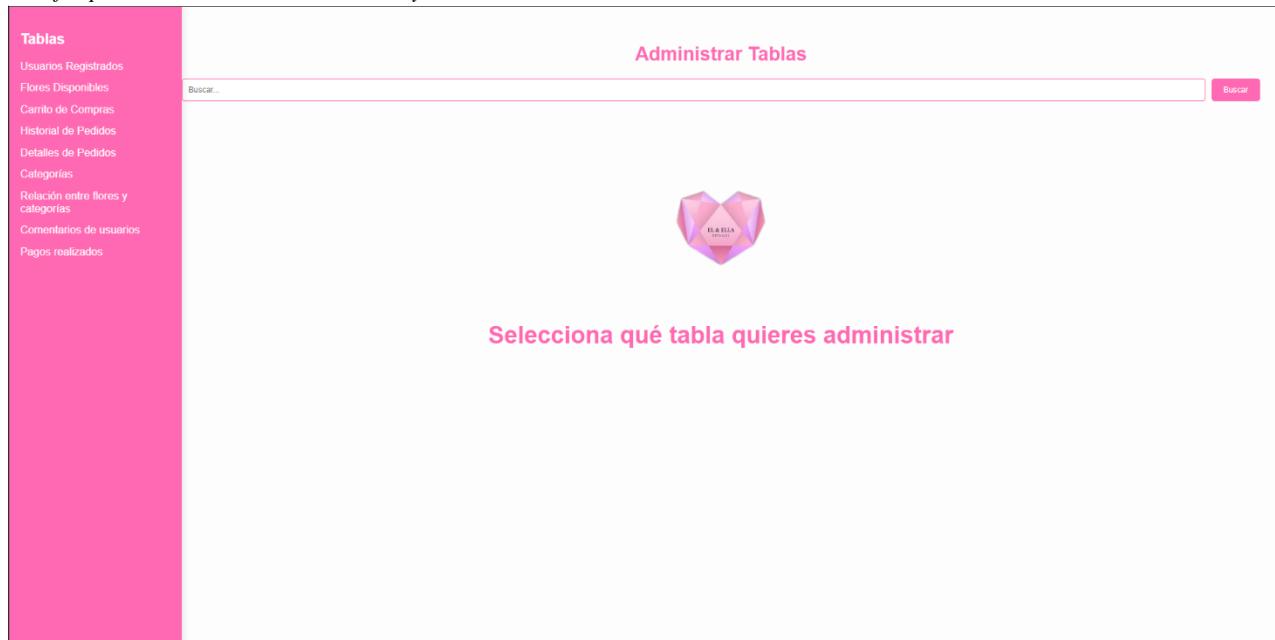
Fuente: Elaboración Propia.

#### 8.1.1.12 Interfaz Administrativa

La interfaz administrativa está diseñada para permitir el manejo completo de las tablas que interactúan con el sistema del proyecto. El acceso a esta interfaz está restringido a un usuario con privilegios de administrador, cuyas credenciales están predefinidas en el sistema. Una vez autenticado, el administrador es redirigido automáticamente a esta interfaz, donde puede ver una lista de tablas en una barra lateral para una navegación más sencilla.

El administrador puede seleccionar cualquier tabla para ver y gestionar los datos, lo que facilita el mantenimiento y la supervisión del sistema de manera centralizada.

**Figura 53:**  
Interfaz para la Gestión de Tablas en "El y Ella Detalles"



**Nota:** La figura muestra la pantalla de administración de tablas del sistema "El y Ella Detalles". En la barra lateral se listan las diferentes tablas disponibles para gestionar, como usuarios registrados, flores disponibles, historial de pedidos, categorías, comentarios de usuarios y pagos realizados. Esta interfaz permite a los administradores seleccionar y gestionar cada tabla de manera individual, facilitando la administración eficiente del sistema. **Fuente: Elaboración Propia.**

Las tablas disponibles para su administración incluyen:

#### 8.1.1.12.1 Usuarios Registrados

**Figura 54:**  
Administración de Usuarios en "El y Ella Detalles"

Administrador Tablas							
Usuarios Registrados							
ID	Nombre	Email	Dirección	Teléfono	Contraseña	Acciones	
1	Joaquín	joaquin@gmail.com	mi casa	949234515	12345	<button>Editar</button>	<button>Eliminar</button>
2	Klara	klara_santi@gmail.com	Otro lio	999999222	Klara	<button>Editar</button>	<button>Eliminar</button>
3	Raul	raul_escalaya@gmail.com	casa de raul	112345678	raul	<button>Editar</button>	<button>Eliminar</button>
27	Pedro Castillo	petercastile@gmail.com	carcel	080012200	pedro	<button>Editar</button>	<button>Eliminar</button>
28	Dina Pauar	DinaBoluarte@gmail.com	Palacio de Gobierno	99922333	Dina	<button>Editar</button>	<button>Eliminar</button>

**Nota:** La figura presenta una tabla con los usuarios registrados en el sistema "El y Ella Detalles". Cada fila muestra el ID, nombre, correo electrónico, dirección, teléfono y contraseña del usuario. En la columna de acciones, se incluyen botones para editar o eliminar cada registro, permitiendo una gestión eficaz de la información de los usuarios desde una interfaz clara y funcional. **Fuente: Elaboración Propia.**

### 8.1.1.12.2 Flores Disponibles

**Figura 55:**  
Administración de Flores en "El y Ella Detalles"

Tablas	
Usuarios Registrados	
Flores Disponibles	
Carrito de Compras	
Historial de Pedidos	
Detalles de Pedidos	
Categorías	
Relación entre flores y categorías	
Comentarios de usuarios	
Pagos realizados	

**Administrar Tablas**

Buscar... Buscar

**Flores Disponibles**

[Agregar Flor](#)

ID Flor	Nombre	Descripción	Precio	Stock	Imagen URL	Acciones
1	CGirasoles	Caja de Girasoles	135.0	20	/img/Box 14.png	<a href="#">Editar</a> <a href="#">Eliminar</a>
2	CTulipanes	Caja de Tulipanes	50.0	35	Box 4.jpg	<a href="#">Editar</a> <a href="#">Eliminar</a>
3	CRosas	Caja de rosas	120.0	15	Box 3.jpg	<a href="#">Editar</a> <a href="#">Eliminar</a>
4	CAmor	Caja de día de San Valentín contiene Rosas de distintos colores	120.0	10	Box 8.jpeg	<a href="#">Editar</a> <a href="#">Eliminar</a>

**Nota:** La figura muestra una tabla con las flores disponibles en el sistema "El y Ella Detalles". Cada registro incluye el ID de la flor, nombre, descripción, precio, stock y URL de la imagen correspondiente. En la columna de acciones, se ofrecen botones para editar o eliminar los registros, lo que permite a los administradores mantener actualizada la información del inventario de flores de manera sencilla y eficiente. **Fuente: Elaboración Propia.**

### 8.1.1.12.3 Carrito de Compras

**Figura 56:**  
Administración del Carrito de Compras en "El y Ella Detalles"

Tablas	
Usuarios Registrados	
Flores Disponibles	
Carrito de Compras	
Historial de Pedidos	
Detalles de Pedidos	
Categorías	
Relación entre flores y categorías	
Comentarios de usuarios	
Pagos realizados	

**Administrar Tablas**

Buscar... Buscar

**Carrito de Compras**

ID Carrito	ID Usuario	ID Flor	Cantidad	Precio	Fecha Agregado	Acciones
3	2	4	7	120.0	2024-09-28 16:11:04.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
4	2	4	3	120.0	2024-09-28 16:36:15.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
5	3	4	4	120.0	2024-09-28 17:19:17.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
6	2	4	1	120.0	2024-09-28 17:30:50.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
7	2	4	1	120.0	2024-09-28 17:31:07.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
8	2	4	1	120.0	2024-09-28 17:31:09.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
9	2	4	1	120.0	2024-09-28 17:31:09.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
10	2	4	1	120.0	2024-09-28 17:31:10.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
11	2	4	1	120.0	2024-09-28 17:31:12.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
13	1	4	1	120.0	2024-10-07 17:51:54.0	<a href="#">Editar</a> <a href="#">Eliminar</a>

**Nota:** La figura muestra la interfaz de administración del carrito de compras en el sistema "El y Ella Detalles". Cada registro incluye el ID del carrito, ID del usuario, ID de la flor, cantidad, precio, y fecha de agregado. En la columna de acciones, se ofrecen opciones para editar o eliminar los registros, permitiendo un control eficiente del contenido del carrito de compras. Esta vista facilita la supervisión y gestión de las compras realizadas por los usuarios.

**Fuente: Elaboración Propia.**

#### 8.1.1.12.4 Historial de Pedidos

**Figura 57:**  
Administración del Historial de Pedidos en "El y Ella Detalles"

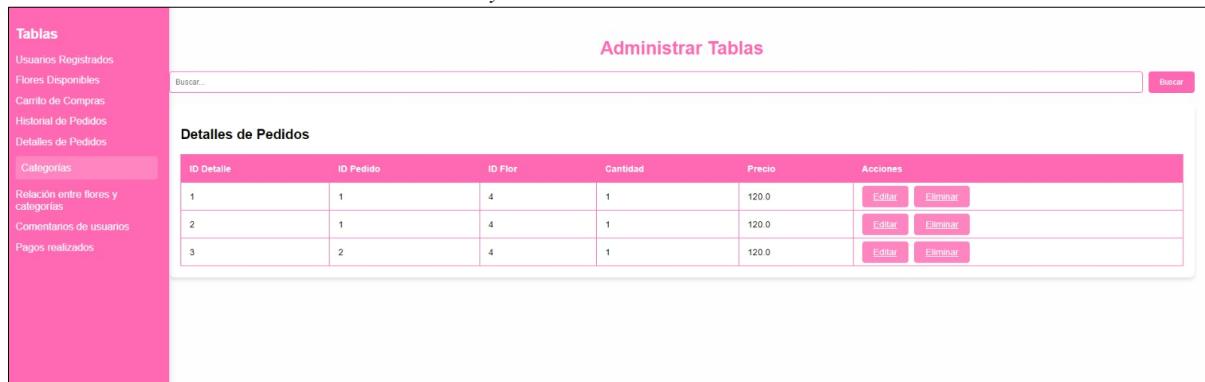


ID Pedido	ID Usuario	Total	Fecha Pedido	Estado	Acciones
1	1	260.0	2024-10-04 18:34:22.0	Pendiente	<button>Editar</button> <button>Eliminar</button>
2	1	140.0	2024-10-07 15:28:26.0	Pendiente	<button>Editar</button> <button>Eliminar</button>

**Nota:** La figura muestra la interfaz de administración del historial de pedidos. Cada registro incluye el ID del pedido, ID del usuario, total, fecha del pedido, y estado del mismo. En la columna de acciones, se ofrecen opciones para editar o eliminar los pedidos. Esta vista facilita el seguimiento de las órdenes realizadas por los usuarios y permite una gestión eficiente de los pedidos pendientes o completados. **Fuente: Elaboración Propia.**

#### 8.1.1.12.5 Detalles de Pedidos

**Figura 58:**  
Administración de los Detalles de Pedidos en "El y Ella Detalles"



ID Detalle	ID Pedido	ID Flor	Cantidad	Precio	Acciones
1	1	4	1	120.0	<button>Editar</button> <button>Eliminar</button>
2	1	4	1	120.0	<button>Editar</button> <button>Eliminar</button>
3	2	4	1	120.0	<button>Editar</button> <button>Eliminar</button>

**Nota:** La figura muestra la interfaz de administración de los detalles de pedidos del sistema. Cada registro incluye el ID del detalle, ID del pedido, ID de la flor, cantidad y precio. En la columna de acciones, se presentan opciones para editar o eliminar los detalles. Esta vista permite gestionar de manera precisa los productos asociados a cada pedido, asegurando un control adecuado sobre las transacciones realizadas. **Fuente: Elaboración Propia.**

### 8.1.1.12.6 Categorías

**Figura 59:**  
Administración de Categorías en "El y Ella Detalles"

ID Categoría	Nombre	Acciones
1	Girasoles	Editar Eliminar
2	Tulipanes	Editar Eliminar
3	Rosas	Editar Eliminar

**Nota:** La figura muestra la interfaz para la gestión de categorías en el sistema. Cada registro presenta el ID de la categoría y su nombre, con opciones en la columna de acciones para editar o eliminar las categorías existentes. Además, se incluye un botón para agregar nuevas categorías, permitiendo a los administradores mantener actualizado el inventario de productos según su clasificación. **Fuente: Elaboración Propia.**

### 8.1.1.12.7 Relación entre Flores y Categorías

**Figura 60:**  
Administración de la Relación entre Flores y Categorías en "El y Ella Detalles"



ID Flor	ID Categoría	Acciones
1	1	Editar Eliminar
2	2	Editar Eliminar
3	3	Editar Eliminar
4	3	Editar Eliminar

**Nota:** La figura muestra la interfaz para gestionar la relación entre flores y sus categorías en el sistema. Cada registro incluye el ID de la flor y el ID de la categoría correspondiente. En la columna de acciones, se ofrecen opciones para editar o eliminar las relaciones existentes, permitiendo mantener la organización y clasificación adecuada de los productos dentro del inventario del sistema. **Fuente: Elaboración Propia.**

### 8.1.1.12.8 Comentarios de Usuarios

**Figura 61:**

Administración de Comentarios en "El y Ella Detalles"

Comentarios de Usuarios										
ID Comentario	ID Usuario	ID Flor	Comentario	Calificación	Fecha	Acciones				
1	2	1	Muy bueno le gusto mucho a mi tia	5	2024-09-20 13:35:30.0	<button>Edit</button>	<button>Eliminar</button>			
2	27	1	Muy bonito le gusto a mi novia	5	2024-09-20 18:46:38.0	<button>Edit</button>	<button>Eliminar</button>			
3	1	1	a	3	2024-09-22 12:37:33.0	<button>Edit</button>	<button>Eliminar</button>			
4	1	1	God	5	2024-09-22 12:43:32.0	<button>Edit</button>	<button>Eliminar</button>			
7	28	3	Soy Dina muy Bonito	4	2024-09-22 15:11:14.0	<button>Edit</button>	<button>Eliminar</button>			
8	2	4	hola	5	2024-09-28 15:54:22.0	<button>Edit</button>	<button>Eliminar</button>			
9	2	1	buenas	5	2024-09-28 15:55:44.0	<button>Edit</button>	<button>Eliminar</button>			
10	2	2	adios	3	2024-09-28 15:56:01.0	<button>Edit</button>	<button>Eliminar</button>			

**Nota:** Esta interfaz permite gestionar los comentarios realizados por los usuarios sobre los productos. Cada registro muestra el ID del comentario, ID del usuario, ID de la flor, el contenido del comentario, la calificación otorgada, y la fecha en que fue realizado. Además, en la columna de acciones, se pueden editar o eliminar comentarios para mantener el control y la calidad del contenido visible en el sitio. **Fuente: Elaboración Propia.**

### 8.1.1.12.9 Pagos Realizados

**Figura 62:**

Administración de Pagos en "El y Ella Detalles"

Pagos Realizados														
ID Pago	ID Usuario	Nombre Completo	Correo	Teléfono	Dirección de Envío	Ciudad	Estado	Código Postal	Número de Tarjeta	Fecha de Expiración	CVV	Monto	Fecha Pago	Acciones
1	2	Kiara Marshall Santi Saavedra	kiara_santi@gmail.com	999222333	casa de Kiara	Lima	S M P	012332	4534534534534543	07/12	324	1940.0	2024-09-28 18:09:04.0	<button>Edit</button> <button>Eliminar</button>
2	1	Joaquín Alfonso Loa Denegri	jloadenegri@gmail.com	949234515	mi casa	Lima	Comas	12312312	4324234234234	07/14	432	260.0	2024-10-04 18:34:22.0	<button>Edit</button> <button>Eliminar</button>
3	1	Joaquín Loa	jloadenegri@gmail.com	949234515	Av Felipe Pinglo Alva 308	Lima	Comas	01504	345643543534543	07/12	432	140.0	2024-10-07 15:28:26.0	<button>Edit</button> <button>Eliminar</button>
4	1	Joaquín Loa	jloadenegri@gmail.com	949234515	Av Felipe Pinglo Alva 308	Lima	Comas	01504	345643543534543	07/12	432	140.0	2024-10-07 15:31:53.0	<button>Edit</button> <button>Eliminar</button>

**Nota:** Esta interfaz presenta los detalles de los pagos realizados por los usuarios. Cada registro incluye información como ID del pago, ID del usuario, nombre completo, correo, teléfono, dirección de envío, ciudad, estado, código postal, número de tarjeta, fecha de expiración, CVV, monto y fecha del pago. En la columna de acciones, se ofrecen opciones para editar o eliminar registros, facilitando una gestión clara y precisa de los pagos efectuados dentro del sistema.

**Fuente: Elaboración Propia.**

### 8.1.1.13 Ejemplo de Administración de Tabla

La vista de administración de una tabla específica muestra los datos de la tabla

seleccionada de manera detallada. En el ejemplo de la tabla de "Usuarios Registrados", se muestran columnas con el ID, nombre, correo electrónico, dirección, teléfono y contraseña de

cada usuario. Además, hay botones de acción para "Editar" o "Eliminar" cada registro, permitiendo al administrador modificar los datos o eliminar usuarios de acuerdo a las necesidades de gestión del sistema.

Esta funcionalidad ofrece un control completo sobre los datos del proyecto, permitiendo realizar ajustes, correcciones o eliminación de registros cuando sea necesario para mantener la integridad y actualización de la información almacenada en la base de datos.

**Figura 63:**  
Administración de Usuarios en "El y Ella Detalles"

**Administrador Tablas**

Buscar...

**Usuarios Registrados**

ID	Nombre	Email	Dirección	Teléfono	Contraseña	Acciones
1	Joaquin	jioadenegri@gmail.com	mi casa	949234515	12345	<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
2	Kiara	kiara_santi@gmail.com	Otro lio	999999222	Kiara	<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
3	Raul	raul_escuayaya@gmail.com	casa de raul	112345678	raul	<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
27	Pedro Castillo	petercastle@gmail.com	carcel	080012200	pedro	<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
28	Dina Paucar	DinaBoluarte@gmail.com	Palacio de Gobierno	999222333	Dina	<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>

**Nota:** La figura muestra la pantalla de administración de tablas del sistema "El y Ella Detalles". En la barra lateral se listan las diferentes tablas disponibles para gestionar, como usuarios registrados, flores disponibles, historial de pedidos, categorías, comentarios de usuarios y pagos realizados. Esta interfaz permite a los administradores seleccionar y gestionar cada tabla de manera individual, facilitando la administración eficiente del sistema. **Fuente: Elaboración Propia.**

## 9 Construcción del Producto Final

La construcción del producto final es un proceso integral que abarca la implementación de las funcionalidades definidas, la alineación entre la documentación y el código, y la aplicación de buenas prácticas de desarrollo. Cada fase del proyecto se ha diseñado cuidadosamente para asegurar que el sistema cumpla con los objetivos establecidos.

A continuación, se detallan los avances logrados y los criterios considerados para garantizar la calidad del desarrollo.

### 9.1 Alcance cubierto en el avance

El avance desarrollado incluye funcionalidades esenciales que permiten gestionar los aspectos clave del sistema, como el inventario y los pedidos, cumpliendo con los objetivos establecidos para esta etapa del proyecto. Estas funcionalidades son fundamentales para asegurar una base sólida sobre la cual se continuará desarrollando las siguientes fases del sistema. A continuación, se describen las funcionalidades implementadas hasta el momento:

### 9.1.1 Gestión de Inventario

Este módulo permite **registrar, actualizar y consultar las flores disponibles** en el sistema. La gestión del inventario es fundamental para mantener un control adecuado del stock, evitando faltantes o excesos de productos. A través de esta funcionalidad, el administrador puede gestionar las flores disponibles en el catálogo de la tienda, asignándoles nombre, descripción, precio, stock e imagen.

Asegura la disponibilidad de productos para los clientes y permite llevar un control eficiente del inventario, garantizando que la tienda pueda responder a la demanda de manera oportuna.

**Figura 64:**  
*Gestión de Flores Disponibles*

Administrador de Flores Disponibles							
Administrar Tablas							
Flores Disponibles							
<input type="text" value="Buscar..."/> <span style="float: right;">Buscar</span>							
ID Flor	Nombre	Descripción	Precio	Stock	Imagen URL	Acciones	
1	CGirasoles	Caja de Girasoles	135.0	20	Box 14.png	<button>Editar</button>	<button>Eliminar</button>
2	CTulipanes	Caja de Tulipanes	50.0	35	Box 4.jpg	<button>Editar</button>	<button>Eliminar</button>
3	CRosas	Caja de rosas	120.0	15	Box 3.jpg	<button>Editar</button>	<button>Eliminar</button>
4	CAmor	Caja de dia de San Valentin contiene Rosas de distintos colores	120.0	10	Box 8.jpeg	<button>Editar</button>	<button>Eliminar</button>

**Nota:** Esta figura muestra la interfaz del módulo de gestión de flores, donde se presenta una lista de las flores disponibles en el sistema con sus respectivos datos, como nombre, descripción, precio, stock e imagen asociada. Además, se ofrecen acciones para editar o eliminar cada flor, permitiendo una administración eficiente del inventario.

*Fuente: Elaboración Propia.*

**Figura 65:**  
Relación entre Flores y Categorías

ID Flor	ID Categoría	Acciones
1	1	[Edit] [Delete]
2	2	[Edit] [Delete]
3	3	[Edit] [Delete]
4	3	[Edit] [Delete]

**Nota:** Esta figura muestra el módulo que administra la relación entre flores y sus categorías correspondientes. Cada entrada indica el ID de la flor y su ID de categoría asignada, permitiendo al administrador editar o eliminar estas relaciones según sea necesario. Esta funcionalidad es fundamental para mantener una organización coherente del catálogo de productos y facilitar la navegación por categorías a los usuarios del sistema. **Fuente: Elaboración Propia.**

### 9.1.2 Carrito de Compras

Este módulo permite a los usuarios seleccionar productos del catálogo y agregarlos al carrito de compras. La funcionalidad del carrito permite **modificar las cantidades de los productos seleccionados y proceder al pago**, ofreciendo al usuario una experiencia de compra sencilla y eficiente.

Mejora la experiencia de usuario al facilitar la gestión de los productos antes de realizar la compra, permitiendo cambios de última hora y un proceso de pago claro y ordenado.

**Figura 66:**  
Carrito de Compras

EL Y ELLA DETALLES

INICIO ROSAS GIRASOLES TULIPANES MÁS BLOG

Continuar comprando

Tienes 1 productos en tu carrito

Flor ID: 4  
1 unidades

S/120.00

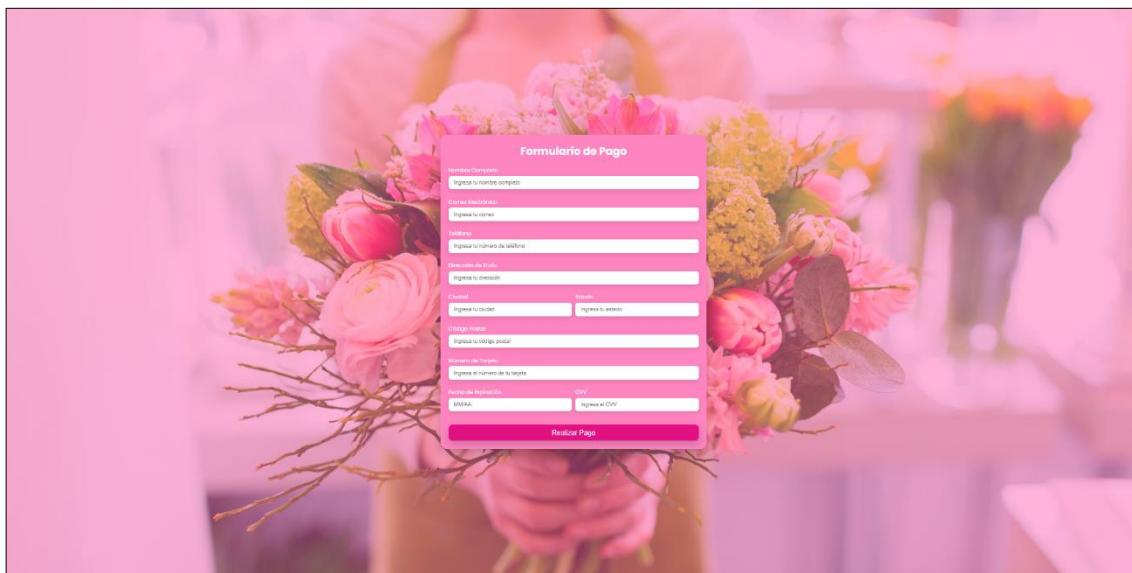
Joaquin

Subtotal S/120.00  
Envío S/20.00  
Total S/140.00

Realizar compra

**Nota:** Esta figura ilustra la interfaz del carrito de compras del sistema. Los usuarios pueden visualizar los productos añadidos, modificar la cantidad seleccionada y observar el subtotal, costo de envío y el total de la compra. Además, se incluye una opción para continuar comprando o realizar la compra, brindando una experiencia intuitiva y ágil para completar el proceso de adquisición de productos. **Fuente: Elaboración Propia.**

**Figura 67:**  
*Formulario de Pago*



**Nota:** Esta figura presenta el formulario de pago, donde los usuarios ingresan la información requerida para completar su compra. Los campos incluyen datos personales como nombre, dirección y correo electrónico, así como información de pago, como número de tarjeta, fecha de expiración y CVV. El diseño simplifica el proceso de pago, garantizando una experiencia rápida y segura para el usuario. **Fuente:** Elaboración Propia.

#### 9.1.3 Historial de Pedidos

El historial de pedidos permite tanto a los usuarios como a los administradores **consultar los pedidos realizados**, junto con información sobre la fecha, estado y monto total. Los usuarios pueden ver un resumen de sus compras anteriores, mientras que los administradores pueden gestionar los pedidos, confirmarlos o cancelarlos según sea necesario.

Facilita el seguimiento de los pedidos y proporciona una mayor transparencia al cliente, mejorando su confianza en el sistema. Además, permite a los administradores organizar y priorizar los envíos.

**Figura 68:**  
Detalles de Pedidos

**Administrar Tablas**

Buscar... | Buscar

**Detalles de Pedidos**

ID Detalle	ID Pedido	ID Flor	Cantidad	Precio	Acciones
1	1	4	1	120.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
2	1	4	1	120.0	<a href="#">Editar</a> <a href="#">Eliminar</a>
3	2	4	1	120.0	<a href="#">Editar</a> <a href="#">Eliminar</a>

**Nota:** En esta vista se presentan los detalles de los pedidos registrados en el sistema. Cada fila muestra el ID del detalle, ID del pedido, ID de la flor, la cantidad solicitada, y el precio total por producto. Además, se ofrecen opciones para editar o eliminar cada entrada, lo que permite al administrador gestionar de manera eficiente la información de los pedidos y garantizar la precisión en el seguimiento de las órdenes. **Fuente: Elaboración Propia.**

**Figura 69:**  
Gestión de Categorías

**Administrar Tablas**

Buscar... | Buscar

**Categorías**

[Agregar Categoría](#)

ID Categoría	Nombre	Acciones
1	Girasoles	<a href="#">Editar</a> <a href="#">Eliminar</a>
2	Tulipanes	<a href="#">Editar</a> <a href="#">Eliminar</a>
3	Rosas	<a href="#">Editar</a> <a href="#">Eliminar</a>

**Nota:** Esta figura muestra el módulo de administración de categorías, donde se listan las categorías existentes en el sistema, como Girasoles, Tulipanes y Rosas. Además, se ofrecen opciones para agregar, editar o eliminar categorías, facilitando la organización del inventario en grupos específicos para mejorar la experiencia del usuario y la gestión del administrador. **Fuente: Elaboración Propia.**

#### 9.1.4 Registro y Gestión de Usuarios

Se ha implementado un módulo que permite **registrar nuevos usuarios** en la plataforma, así como gestionar y administrar la información de los clientes. El administrador puede **editar o eliminar usuarios registrados**, asegurando que la base de datos de usuarios esté siempre actualizada y libre de inconsistencias.

Este módulo permite personalizar la experiencia del cliente, facilitando el acceso a su historial de compras y comentarios. Además, mejora la seguridad del sistema al llevar un control claro de los usuarios registrados.

**Figura 70:**  
*Usuarios Registrados*

**Administrador Tablas**

**Usuarios Registrados**

ID	Nombre	Email	Dirección	Teléfono	Contraseña	Acciones
1	Joaquin	joaquin.negri@gmail.com	mi casa	949234515	12345	<button>Editar</button> <button>Eliminar</button>
2	Kiara	kiara_santi@gmail.com	Otro lio	999999222	Kiara	<button>Editar</button> <button>Eliminar</button>
3	Raul	raul_escalalaya@gmail.com	casa de raul	112345678	raul	<button>Editar</button> <button>Eliminar</button>
27	Pedro Castillo	petercastile@gmail.com	carcel	080012200	pedro	<button>Editar</button> <button>Eliminar</button>
28	Dina Paucar	DinaBoluarte@gmail.com	Palacio de Gobierno	999222333	Dina	<button>Editar</button> <button>Eliminar</button>

**Nota:** Esta vista muestra la lista de usuarios registrados en el sistema. Cada registro contiene información relevante, como ID, nombre, correo electrónico, dirección, teléfono y contraseña. Además, se incluyen opciones para editar o eliminar usuarios, proporcionando al administrador control total sobre los datos de los usuarios y facilitando la gestión de cuentas en caso de modificaciones o bajas. **Fuente: Elaboración Propia.**

### 9.1.5 Comentarios de Usuarios sobre Productos

Los usuarios pueden **dejar comentarios y calificaciones** sobre los productos adquiridos, lo que permite evaluar la satisfacción del cliente y ofrecer retroalimentación directa a la tienda. Estos comentarios también funcionan como referencias para otros usuarios al momento de realizar sus compras.

Fomenta la interacción y fidelización de los clientes, al darles la posibilidad de expresar su opinión sobre los productos. También ayuda al administrador a identificar áreas de mejora.

**Figura 71:**  
Comentarios de Usuarios

Tablas	Administrar Tablas						
	Comentarios de Usuarios						
ID Comentario	ID Usuario	ID Flor	Comentario	Calificación	Fecha	Acciones	
1	2	1	Muy bueno le gusto mucho a mi tia	5	2024-09-20 13:35:30.0	<button>Editar</button>	<button>Eliminar</button>
2	27	1	Muy bonito le gusto a mi novia	5	2024-09-20 18:46:38.0	<button>Editar</button>	<button>Eliminar</button>
3	1	1	a	3	2024-09-22 12:37:33.0	<button>Editar</button>	<button>Eliminar</button>
4	1	1	God	5	2024-09-22 12:43:32.0	<button>Editar</button>	<button>Eliminar</button>
7	28	3	Soy Dina muy Bonito	4	2024-09-22 15:11:14.0	<button>Editar</button>	<button>Eliminar</button>
8	2	4	hola	5	2024-09-28 15:54:22.0	<button>Editar</button>	<button>Eliminar</button>
9	2	1	buenas	5	2024-09-28 15:55:44.0	<button>Editar</button>	<button>Eliminar</button>
10	2	2	adios	3	2024-09-28 15:56:01.0	<button>Editar</button>	<button>Eliminar</button>

**Nota:** Esta sección muestra los comentarios y valoraciones que los usuarios han dejado sobre los productos adquiridos.

Cada comentario se encuentra vinculado a un ID de usuario y un ID de flor, junto con una calificación numérica, la fecha en la que se realizó el comentario y las opciones para editar o eliminar. Esta funcionalidad permite a los administradores gestionar las opiniones de los clientes, fomentando un entorno de retroalimentación constante.

Fuente: Elaboración Propia.

### 9.1.6 Pagos Realizados

El sistema registra todos los pagos realizados por los clientes, junto con información relevante como el monto, el método de pago, el número de tarjeta y la fecha de pago. Esta funcionalidad permite al administrador llevar un control preciso de las transacciones efectuadas.

Asegura la trazabilidad de las transacciones, facilita la resolución de problemas relacionados con los pagos y genera confianza en los usuarios al proporcionar transparencia en el proceso de compra.

**Figura 72:**  
Pagos Realizados

Tablas	Administrar Tablas														
	Pagos Realizados														
ID Pago	ID Usuario	Nombre Completo	Correo	Teléfono	Dirección de Envío	Ciudad	Estado	Código Postal	Número de Tarjeta	Fecha de Expiración	CVV	Monto	Fecha Pago	Acciones	
1	2	Kara Mishell Santi Saavedra	kiara_santi@gmail.com	999222333	casa de kiara	Lima	S M P	012332	4534534534534543	07/12	324	1940.0	2024-09-28 15:09:04.0	<button>Editar</button>	<button>Eliminar</button>
2	1	Joaquín Alfonso Loa Denegri	jloadenegri@gmail.com	949234515	mi casa	Lima	Comas	12312312	4324234234234	07/14	432	260.0	2024-10-04 18:34:22.0	<button>Editar</button>	<button>Eliminar</button>
3	1	Joaquín Loa	jloadenegri@gmail.com	949234515	Av Felipe Pinglo Alva 308	Lima	Comas	01504	34564354354543	07/12	432	140.0	2024-10-07 15:26:26.0	<button>Editar</button>	<button>Eliminar</button>
4	1	Joaquín Loa	jloadenegri@gmail.com	949234515	Av Felipe Pinglo Alva 308	Lima	Comas	01504	34564354354543	07/12	432	140.0	2024-10-07 15:31:53.0	<button>Editar</button>	<button>Eliminar</button>

**Nota:** Esta sección presenta un registro detallado de los pagos efectuados por los usuarios, incluyendo información relevante como el nombre completo, correo, teléfono, y la dirección de envío. Además, se muestran los datos de pago, tales como el monto, fecha del pago, y opciones para editar o eliminar los registros. Fuente: Elaboración Propia.

### 9.1.7 Administración de Categorías y Relación con Flores

El módulo de categorías permite agrupar las flores en distintas categorías, facilitando la organización del catálogo. Además, se ha implementado una funcionalidad que permite relacionar flores con categorías, asegurando una navegación más ordenada y estructurada para los usuarios.

Mejora la experiencia del usuario al simplificar la búsqueda de productos específicos y optimiza la gestión del inventario por parte del administrador.

**Figura 73:**  
Gestión de Categorías

ID Categoría	Nombre	Acciones
1	Girasoles	Editar Eliminar
2	Tulipanes	Editar Eliminar
3	Rosas	Editar Eliminar

**Nota:** Esta interfaz permite administrar las categorías del sistema, proporcionando opciones para agregar nuevas categorías y editar o eliminar las existentes. La sección garantiza una organización eficiente de los productos, facilitando la clasificación y búsqueda rápida por parte de los usuarios. **Fuente:** Elaboración Propia.

**Figura 74:**  
Relación entre Flores y Categorías

ID Flor	ID Categoría	Acciones
1	1	Editar Eliminar
2	2	Editar Eliminar
3	3	Editar Eliminar
4	3	Editar Eliminar

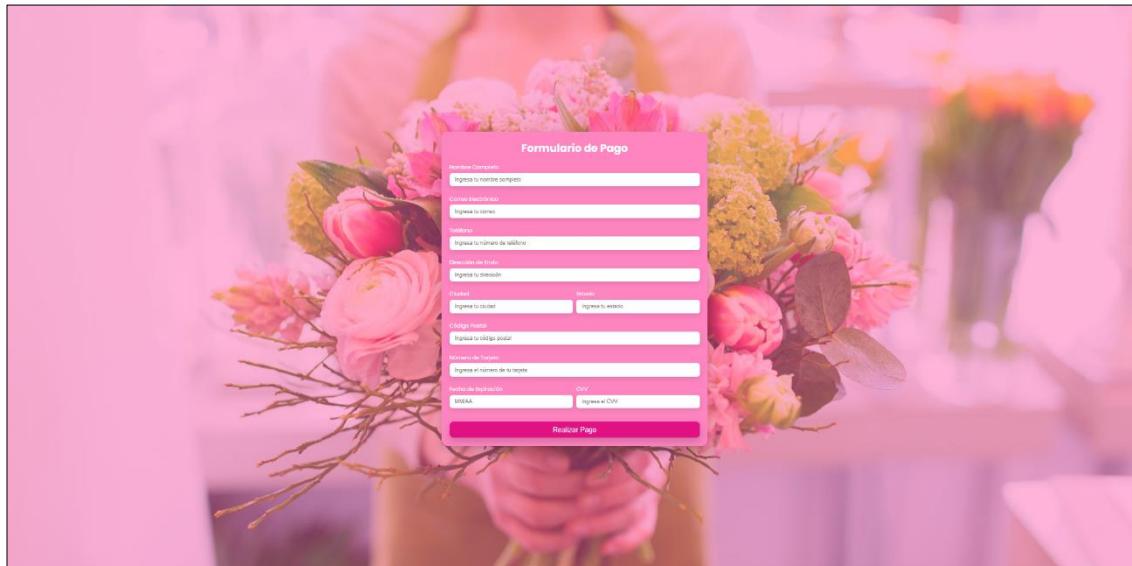
**Nota:** Esta sección permite gestionar la asociación entre las flores disponibles y sus respectivas categorías, facilitando la organización y clasificación de los productos. Proveer esta estructura asegura que los usuarios puedan encontrar productos específicos de manera más eficiente mediante la navegación por categorías relevantes. Las opciones de editar y eliminar garantizan flexibilidad para mantener actualizada esta relación. **Fuente:** Elaboración Propia.

### 9.1.8 Formulario de Pago y Procesamiento de Compras

Se ha diseñado e implementado un formulario de pago completo, donde los usuarios ingresan sus datos de facturación para finalizar la compra. El formulario incluye campos para datos personales y de la tarjeta de crédito, asegurando que el proceso de pago sea claro y eficiente.

Garantiza un proceso de pago sencillo y seguro, minimizando errores y facilitando la experiencia del cliente durante la compra.

**Figura 75:**  
*Formulario de Pago*



**Nota:** Este formulario captura la información necesaria para procesar el pago de un pedido. Incluye campos para el nombre, dirección, ciudad, estado y datos de la tarjeta de crédito, como el número, fecha de expiración y CVV. El diseño intuitivo asegura que los usuarios puedan completar su compra de manera rápida y segura. El botón "Realizar Pago" inicia el proceso de transacción, garantizando una experiencia de compra eficiente y sencilla. **Fuente:** Elaboración Propia.

### 9.2 Coherencia entre documentación y código

La documentación es un elemento crítico del proyecto, ya que proporciona una referencia clara sobre la estructura y funcionamiento del sistema. Para garantizar coherencia, cada módulo y funcionalidad desarrollada ha sido descrito de manera detallada en la documentación técnica, asegurando que las decisiones de diseño y los procesos de desarrollo estén alineados con los objetivos iniciales.

Para garantizar la coherencia entre la documentación y el código, cada componente ha sido cuidadosamente descrito y documentado.

- **Descripciones de diseño:** Las decisiones tomadas en cada fase del desarrollo se documentaron en informes técnicos, alineando la implementación con los objetivos del proyecto.
- **Validación cruzada:** Se realizaron revisiones continuas para asegurar que cada parte del código desarrollado corresponda a lo descrito en la documentación. Esto permite que el sistema sea fácilmente auditável, reduciendo el riesgo de inconsistencias.

A continuación, se detallan estos componentes clave:

#### 9.2.1 Informe Técnico: Decisiones de Diseño del Proyecto

##### 1. Uso de Patrones de Diseño

Para garantizar una estructura sólida y mantenible en el sistema, se optó por implementar dos patrones de diseño fundamentales:

- **Patrón DAO (Data Access Object):**

Este patrón se utilizó para separar la lógica de acceso a la base de datos del resto del sistema. Cada tabla de la base de datos tiene su clase DAO correspondiente (por ejemplo, RegisterDAO, PedidosDAO, FlorDAO), lo que permite realizar las operaciones CRUD de forma modular y reutilizable. Esta elección mejora la mantenibilidad y facilita futuros cambios en la base de datos sin afectar la lógica del negocio:

- a) **CRUD de RegisterDAO**

- **Create:**

**Figura 76:**

Implementación del método save para registro de usuarios

```

public boolean save(Register r) {
    String cadSQL = "INSERT INTO register (nombre, email, contrasena, direccion, telefono) VALUES (?, ?, ?, ?, ?)";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) { // try-with-resources para cerrar ps automáticamente
        ps.setString(i: 1, string: r.getNombre());
        ps.setString(i: 2, string: r.getEmail());
        ps.setString(i: 3, string: r.getContrasena());
        ps.setString(i: 4, string: r.getDireccion());
        ps.setString(i: 5, string: r.getTelefono());
        ps.executeUpdate();
        return true; // Registro exitoso
    } catch (SQLException ex) {
        System.out.println("Error al registrar usuario: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}

```

**Nota:** El método save permite almacenar nuevos usuarios en la base de datos mediante una sentencia SQL INSERT. Se utiliza la clase PreparedStatement para evitar inyecciones SQL y gestionar los parámetros dinámicamente. Cada atributo del objeto Register es asignado a su respectivo campo en la tabla register. En caso de éxito, el método devuelve true, indicando que el registro fue exitoso. **Fuente:** Elaboración Propia.

- **Read:**

**Figura 77:**

Método get para obtener usuario por ID

```

public Register get(int id) {
    Register r = null;
    PreparedStatement ps;
    ResultSet rs;
    String cadSQL = "SELECT * FROM register WHERE user_id = ?";
    try {
        ps = cnx.prepareStatement(string: cadSQL);
        ps.setInt(i: 1, ii: id);
        rs = ps.executeQuery();

        if (rs.next()) {
            r = new Register(
                userId: rs.getInt(string: "user_id"),
                nombre: rs.getString(string: "nombre"),
                email: rs.getString(string: "email"),
                contrasena: rs.getString(string: "contrasena"),
                direccion: rs.getString(string: "direccion"),
                telefono: rs.getString(string: "telefono")
            );
        }
        rs.close();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return r;
}

```

**Nota:** Este método realiza una consulta en la base de datos utilizando el ID del usuario como parámetro. A través de un PreparedStatement, se ejecuta la consulta SQL para seleccionar los datos del usuario correspondiente. Si se encuentra un registro, se crea y devuelve un objeto Register con la información obtenida (ID, nombre, email, contraseña, dirección y teléfono). Además, el método implementa manejo de excepciones para capturar errores de SQL y asegura el cierre adecuado de recursos mediante el uso de try-catch. **Fuente:** Elaboración Propia.

- **Actualizar:**

**Figura 78:**

Método update para actualizar usuario

```
// Método para actualizar un usuario existente
public boolean update(Register r) {
    String cadSQL = "UPDATE register SET nombre = ?, email = ?, contrasena = ?, direccion = ?, telefono = ? WHERE user_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setString(i: 1, string: r.getNombre());
        ps.setString(i: 2, string: r.getEmail());
        ps.setString(i: 3, string: r.getContrasena());
        ps.setString(i: 4, string: r.getDireccion());
        ps.setString(i: 5, string: r.getTelefono());
        ps.setInt(i: 6, ii: r.getUserId()); // Asegúrate de tener el ID en el objeto Register
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al actualizar usuario: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** Este método permite actualizar los datos de un usuario existente en la base de datos. Utiliza un `PreparedStatement` para preparar la consulta SQL con los nuevos valores de nombre, email, contraseña, dirección y teléfono, basándose en el `user_id` del usuario. La ejecución se realiza dentro de un bloque `try-catch` para manejar posibles excepciones SQL y asegurar la integridad del proceso. **Fuente: Elaboración Propia.**

- **Delete:**

**Figura 79:**

Método eliminar para eliminar un usuario por ID

```
// Método para eliminar un usuario por ID
public boolean eliminar(int id) {
    String cadSQL = "DELETE FROM register WHERE user_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setInt(i: 1, ii: id);
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al eliminar usuario: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** Este método elimina un usuario específico de la base de datos utilizando su `user_id`. Se realiza mediante un `PreparedStatement` con la consulta SQL `DELETE`. El bloque `try-catch` garantiza el manejo seguro de excepciones SQL, devolviendo `true` si la eliminación es exitosa o `false` en caso de error. **Fuente: Elaboración Propia.**

## b) CRUD de PedidosDAO

- **Read:**

Figura 80:

Método get para obtener un pedido por ID

```
public Pedidos get(int id) {
    Pedidos pedido = null;
    PreparedStatement ps;
    ResultSet rs;
    String sql = "SELECT * FROM pedidos WHERE pedido_id = ?";
    try {
        ps = cnx.prepareStatement(string: sql);
        ps.setInt(i: 1, ii: id);
        rs = ps.executeQuery();

        if (rs.next()) {
            pedido = new Pedidos(
                pedidoId: rs.getInt(string: "pedido_id"),
                userId: rs.getInt(string: "user_id"),
                total: rs.getDouble(string: "total"),
                estado: rs.getString(string: "estado"),
                fechaPedido:rs.getDate(string: "fecha_pedido") // Asegúrate de usar el nombre correcto de la columna
            );
        }
        rs.close();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return pedido;
}
```

**Nota:** Este método permite recuperar un pedido específico desde la base de datos utilizando su `pedido_id`. Emplea un `PreparedStatement` para ejecutar la consulta SQL `SELECT`. Si el pedido existe, se instancia un nuevo objeto `Pedidos` con los valores recuperados. El bloque `try-catch` maneja excepciones SQL, asegurando un control adecuado de errores.

Fuente: Elaboración Propia.

- **Update:**

Figura 81:

Método update para actualizar un pedido

```
// Método para actualizar un pedido existente
public boolean update(Pedidos pedido) {
    String sql = "UPDATE pedidos SET user_id = ?, total = ?, estado = ? WHERE pedido_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: sql)) {
        ps.setInt(i: 1, ii: pedido.getUserId());
        ps.setDouble(i: 2, d: pedido.getTotal());
        ps.setString(i: 3, string: pedido.getEstado());
        ps.setInt(i: 4, ii: pedido.getPedidoId()); // Asignar el ID del pedido a actualizar
        ps.executeUpdate();
        return true; // Actualización exitosa
    } catch (SQLException ex) {
        System.out.println("Error al actualizar el pedido: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** Este método permite modificar la información de un pedido existente en la base de datos. Utiliza una sentencia SQL `UPDATE` para cambiar los datos del usuario, total y estado del pedido. La ejecución del bloque `try` garantiza la correcta preparación y ejecución de la sentencia, mientras que el bloque `catch` gestiona posibles excepciones SQL, mostrando un mensaje de error en caso de fallo. Fuente: Elaboración Propia.

- **Delete:**

**Figura 82:**  
Método eliminar para borrar un pedido por ID

```
// Método para eliminar un pedido por ID
public boolean eliminar(int id) {
    String sql = "DELETE FROM pedidos WHERE pedido_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: sql)) {
        ps.setInt(i: 1, i1: id);
        ps.executeUpdate();
        return true; // Eliminación exitosa
    } catch (SQLException ex) {
        System.out.println("Error al eliminar el pedido: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** Este método realiza la eliminación de un pedido específico en la base de datos utilizando una sentencia SQL DELETE. El try asegura la ejecución correcta de la consulta preparada, eliminando el pedido correspondiente al ID proporcionado. Si ocurre un error, el bloque catch captura la excepción y muestra un mensaje para identificar la causa del fallo. **Fuente: Elaboración Propia.**

### c) CRUD de FloresDAO

- **Create:**

**Figura 83:**  
Método agregar para insertar flores en la base de datos

```
public boolean agregar(Flores f) {
    String cadSQL = "INSERT INTO flores (nombre, descripcion, precio, stock, imagen_url) VALUES (?, ?, ?, ?, ?)";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setString(i: 1, string: f.getNombre());
        ps.setString(i: 2, string: f.getDescripcion());
        ps.setDouble(i: 3, d: f.getPrecio());
        ps.setInt(i: 4, i1: f.getStock());
        ps.setString(i: 5, string: f.getImagenUrl());
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al agregar flor: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

**Nota:** Este método utiliza una sentencia SQL INSERT para añadir un nuevo registro de flor con sus atributos (nombre, descripción, precio, stock e imagen). La consulta se realiza mediante un PreparedStatement para asegurar seguridad contra inyecciones SQL. Si la operación es exitosa, devuelve true; en caso de error, se captura la excepción y se imprime un mensaje con el detalle del fallo. **Fuente: Elaboración Propia.**

- **Read:**

**Figura 84:**  
Método get para obtener una flor por ID

```

public Flores get(int id) {
    Flores f = null;
    PreparedStatement ps;
    ResultSet rs;
    String cadSQL = "SELECT * FROM flores WHERE flor_id = ?";
    try {
        ps = cnx.prepareStatement(string: cadSQL);
        ps.setInt(i: 1, il: id);
        rs = ps.executeQuery();

        if (rs.next()) {
            f = new Flores(
                florId: rs.getInt(string: "flor_id"),
                nombre: rs.getString(string: "nombre"),
                descripcion: rs.getString(string: "descripcion"),
                precio: rs.getDouble(string: "precio"),
                stock: rs.getInt(string: "stock"),
                imagenUrl: rs.getString(string: "imagen_url")
            );
        }
        rs.close();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return f;
}

```

**Nota:** Este método recupera la información de una flor específica de la base de datos utilizando su `flor_id`. Mediante una sentencia SQL `SELECT`, el resultado se almacena en un objeto `Flores` si se encuentra una coincidencia. El `try-catch` garantiza que cualquier error en la consulta sea capturado, permitiendo manejar excepciones SQL y asegurando que el recurso se cierre correctamente. **Fuente:** Elaboración Propia.

- **Update:**

**Figura 85:**  
Método update para actualizar una flor existente

```

// Método para actualizar una flor existente
public boolean update(Flores f) {
    String cadSQL = "UPDATE flores SET nombre = ?, descripcion = ?, precio = ?, stock = ?, imagen_url = ? WHERE flor_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setString(i: 1, string: f.getNombre());
        ps.setString(i: 2, string: f.getDescripcion());
        ps.setDouble(i: 3, d: f.getPrecio());
        ps.setInt(i: 4, il: f.getStock());
        ps.setString(i: 5, string: f.getImagenUrl());
        ps.setInt(i: 6, il: f.getFlorId());
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al actualizar flor: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}

```

**Nota:** Este método permite actualizar la información de una flor en la base de datos mediante su `flor_id`. Utiliza una sentencia SQL `UPDATE` para modificar los atributos como `nombre`, `descripción`, `precio`, `stock` e `imagen`. La instrucción preparada garantiza seguridad frente a inyecciones SQL y eficiencia en la ejecución. En caso de error, se captura la excepción SQL para su gestión, asegurando que el proceso falle de manera controlada. **Fuente:** Elaboración Propia.

- **Delete:**

**Figura 86:**

Método eliminar para borrar una flor por ID

```
// Método para eliminar una flor por ID
public boolean eliminar(int id) {
    String cadSQL = "DELETE FROM flores WHERE flor_id = ?";
    try (PreparedStatement ps = cnx.prepareStatement(string: cadSQL)) {
        ps.setInt(i: 1, ii: id);
        ps.executeUpdate();
        return true;
    } catch (SQLException ex) {
        System.out.println("Error al eliminar flor: " + ex.getMessage());
        ex.printStackTrace();
        return false;
    }
}
```

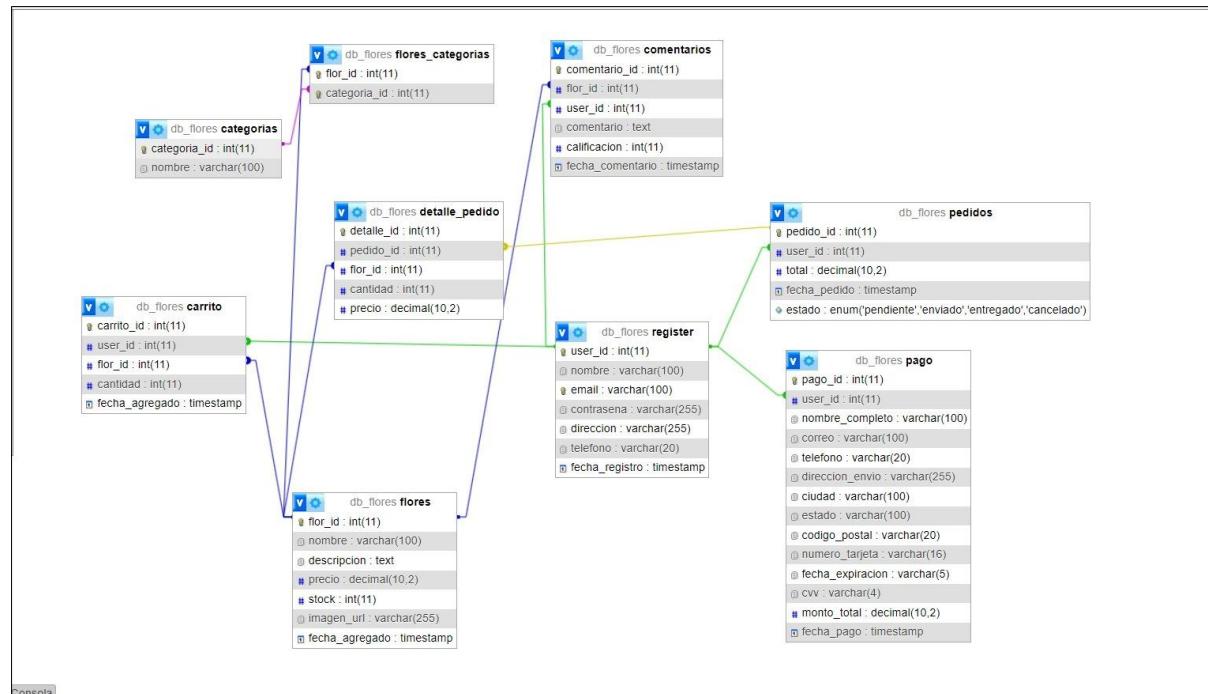
**Nota:** Este método utiliza una sentencia SQL DELETE para eliminar una flor específica de la base de datos según su flor\_id. La instrucción preparada evita riesgos de inyección SQL al establecer el parámetro del ID. En caso de éxito, devuelve true; si ocurre un error, se captura la excepción SQL, mostrando un mensaje de error detallado y devolviendo false. Este enfoque asegura la estabilidad del sistema ante posibles fallos en la eliminación. **Fuente:** Elaboración

## 2. Justificación del Modelo de Base de Datos

El diseño del modelo de datos se llevó a cabo considerando las necesidades funcionales del sistema y asegurando la correcta relación entre las entidades. A continuación, se detalla la lógica detrás de cada módulo y la estructura de las relaciones:

**Figura 87:**

Modelo Relacional de la Base de Datos del Sistema



**Nota:** El diagrama muestra la estructura y relaciones entre las tablas principales del sistema, incluyendo usuarios, flores, pedidos, pagos, y categorías. Las conexiones ilustran cómo se organiza y gestiona la información para asegurar integridad y trazabilidad en cada transacción realizada dentro de la aplicación. **Fuente:** Elaboración Propia.

### a) Módulo de Usuarios y Registro

- **Entidad:** register
- **Descripción:** Esta tabla permite gestionar a los usuarios registrados, incluyendo su información personal y datos de contacto (nombre, email, dirección, teléfono).
- **Decisión:** Se optó por incluir campos adicionales, como fecha de registro, para facilitar la gestión de usuarios y auditorías internas. Cada usuario tiene un identificador único que permite relacionarse con otros módulos como pedidos y pagos.

Figura 88:

Creación de la Tabla "Register" (Usuarios Registrados)

```
4      -- Crear tabla register (Usuarios registrados)
5 • CREATE TABLE register (
6      user_id INT AUTO_INCREMENT PRIMARY KEY,
7      nombre VARCHAR(100) NOT NULL,
8      email VARCHAR(100) UNIQUE NOT NULL,
9      contraseña VARCHAR(255) NOT NULL,
10     dirección VARCHAR(255),
11     teléfono VARCHAR(20),
12     fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
13 );
```

*Nota: El fragmento de código SQL muestra la creación de la tabla register, que almacena la información de los usuarios registrados en el sistema. Cada usuario cuenta con un identificador único (user\_id) como clave primaria, además de campos esenciales como nombre, correo electrónico, contraseña, dirección, teléfono y la fecha de registro. Esta estructura permite un control organizado y seguro de los datos de los usuarios. Fuente: Elaboración Propia.*

### b) Inventario de Productos (Flores)

- **Entidad:** flores
- **Descripción:** Esta tabla centraliza la información de los productos disponibles, como nombre, descripción, precio y stock, junto con la imagen URL correspondiente.

- **Relación:** A través de la tabla `flores_categorias`, se establece una relación muchos a muchos entre flores y categorías, permitiendo una categorización flexible.
- **Decisión:** La estructura modular asegura que cada flor pueda pertenecer a varias categorías y que las categorías puedan reutilizarse para distintos productos, optimizando la organización del catálogo.

**Figura 89:**

Creación de la Tabla "Flores" (Productos Disponibles)

```

15      -- Crear tabla flores (Productos disponibles)
16 • CREATE TABLE flores (
17     flor_id INT AUTO_INCREMENT PRIMARY KEY,
18     nombre VARCHAR(100) NOT NULL,
19     descripcion TEXT,
20     precio DECIMAL(10, 2) NOT NULL,
21     stock INT NOT NULL,
22     imagen_url VARCHAR(255),
23     fecha_agregado TIMESTAMP DEFAULT CURRENT_TIMESTAMP
24 );

```

*Nota: El código SQL ilustra la creación de la tabla flores, destinada a gestionar los productos disponibles en el sistema. Esta tabla incluye campos como el identificador único del producto (flor\_id), nombre, descripción, precio, stock disponible, URL de la imagen y la fecha de agregado. Fuente: Elaboración Propia.*

### c) Carrito de Compras

- **Entidad:** `carrito`
- **Descripción:** Esta entidad registra los productos agregados por los usuarios al carrito de compras, junto con la cantidad seleccionada y la fecha de adición.
- **Decisión:** Esta tabla permite que los usuarios gestionen sus compras antes de finalizar un pedido. La información detallada facilita la experiencia del cliente y permite un análisis de los productos más demandados.

**Figura 90:**

Creación de la Tabla "Carrito" (Productos en el Carrito)

```
-- Crear tabla carrito (Productos en el carrito)
• CREATE TABLE carrito (
    carrito_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    flor_id INT,
    cantidad INT NOT NULL,
    fecha_agregado TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES register(user_id),
    FOREIGN KEY (flor_id) REFERENCES flores(flor_id)
);
```

*Nota: Este fragmento de código SQL muestra la estructura de la tabla carrito, la cual permite gestionar los productos que los usuarios agregan a su carrito de compras. Fuente: Elaboración Propia.*

#### d) Gestión de Pedidos y Detalles

- **Entidades:** pedidos y detalle\_pedido
- **Descripción:** La entidad pedidos contiene información general del pedido (fecha, estado, total), mientras que detalle\_pedido desglosa cada producto incluido en el pedido.
- **Decisión:** La separación en dos entidades permite mayor flexibilidad al gestionar pedidos complejos con múltiples productos. Esta estructura asegura una trazabilidad precisa de cada transacción.

**Figura 91:**

Creación de la Tabla "Pedidos" (Historial de Pedidos)

```
-- Crear tabla pedidos (Historial de pedidos)
• CREATE TABLE pedidos (
    pedido_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    total DECIMAL(10, 2) NOT NULL,
    fecha_pedido TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    estado ENUM('pendiente', 'enviado', 'entregado', 'cancelado') DEFAULT 'pendiente',
    FOREIGN KEY (user_id) REFERENCES register(user_id)
);
```

*Nota: Este fragmento de código SQL muestra la definición de la tabla pedidos, que almacena los registros de pedidos realizados por los usuarios. La tabla incluye un identificador único de pedido (pedido\_id), el identificador del usuario (user\_id), el total del pedido, la fecha en que se realizó y un estado que indica el progreso del pedido (pendiente, enviado, entregado o cancelado). Además, se establece una clave foránea que asegura la integridad referencial con la tabla de usuarios. Fuente: Elaboración Propia.*

**Figura 92:**  
Creación de la Tabla "Detalle Pedido" (Detalle de cada pedido)

```
-- Crear tabla detalle_pedido (Detalle de cada pedido)
• CREATE TABLE detalle_pedido (
    detalle_id INT AUTO_INCREMENT PRIMARY KEY,
    pedido_id INT,
    flor_id INT,
    cantidad INT NOT NULL,
    precio DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (pedido_id) REFERENCES pedidos(pedido_id),
    FOREIGN KEY (flor_id) REFERENCES flores(flor_id)
);
```

**Nota:** El código SQL define la tabla detalle\_pedido, que almacena información específica sobre los productos incluidos en cada pedido. Cada registro contiene un identificador único (detalle\_id), el identificador del pedido al que pertenece (pedido\_id), el identificador del producto (flor\_id), la cantidad comprada y el precio unitario. Las claves foráneas aseguran la integridad referencial con las tablas pedidos y flores, garantizando que los detalles estén correctamente asociados. **Fuente: Elaboración Propia.**

#### e) Procesamiento de Pagos

- **Entidad:** pago
- **Descripción:** Almacena información sobre los pagos realizados, incluyendo datos de la tarjeta, monto pagado, y fecha de transacción.
- **Decisión:** La entidad de pagos está relacionada con los usuarios, permitiendo llevar un control detallado de las transacciones. Esto mejora la gestión financiera y asegura que cada pago esté vinculado a un usuario y pedido específico.

**Figura 93:**  
Creación de la Tabla "Pago" (Información de pago)

```
-- Crear tabla pago (Información de pago)
• CREATE TABLE pago (
    pago_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    nombre_completo VARCHAR(100) NOT NULL,
    correo VARCHAR(100) NOT NULL,
    telefono VARCHAR(20),
    direccion_envio VARCHAR(255) NOT NULL,
    ciudad VARCHAR(100),
    estado VARCHAR(100),
    codigo_postal VARCHAR(20),
    numero_tarjeta VARCHAR(16) NOT NULL,
    fecha_expiracion VARCHAR(5) NOT NULL, -- MM/AA
    cvv VARCHAR(4) NOT NULL,
    monto_total DECIMAL(10, 2) NOT NULL,
    fecha_pago TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES register(user_id)
);
```

*Nota: La tabla pago almacena los detalles de las transacciones realizadas por los usuarios. Cada registro incluye información esencial como el nombre completo, correo, dirección de envío, datos de la tarjeta, y el monto total pagado. Se utiliza una clave primaria pago\_id para identificar cada pago de manera única, y una clave foránea user\_id para asociar cada transacción con un usuario registrado en la tabla register. Este diseño asegura la integridad de los datos financieros y facilita el seguimiento de pagos dentro del sistema.*

#### f) Comentarios y Opiniones de Usuarios

- **Entidad:** comentarios
- **Descripción:** Permite a los usuarios dejar reseñas y calificaciones sobre los productos adquiridos.
- **Decisión:** La relación entre comentarios, usuarios y flores permite analizar el rendimiento de cada producto basado en las opiniones de los clientes, lo que es útil para optimizar la oferta de productos.

**Figura 94:**  
Creación de la Tabla "Comentarios" (Comentarios y valoraciones de productos)

```
-- Crear tabla comentarios (Comentarios y valoraciones de productos)
• CREATE TABLE comentarios (
    comentario_id INT AUTO_INCREMENT PRIMARY KEY,
    flor_id INT,
    user_id INT,
    comentario TEXT,
    calificacion INT CHECK (calificacion BETWEEN 1 AND 5),
    fecha_comentario TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (flor_id) REFERENCES flores(flor_id),
    FOREIGN KEY (user_id) REFERENCES register(user_id)
);
```

**Nota:** La tabla comentarios almacena las opiniones y valoraciones de los usuarios sobre los productos adquiridos. Cada comentario incluye una calificación en una escala de 1 a 5 y un mensaje opcional, con las fechas de publicación automáticamente registradas mediante CURRENT\_TIMESTAMP. Se utilizan claves foráneas para enlazar cada comentario con el producto correspondiente (flor\_id) y con el usuario que lo realizó (user\_id). Esta estructura permite analizar la satisfacción de los clientes y mejorar los servicios basándose en sus opiniones. Fuente: Elaboración Propia.

## 2. Relaciones entre Entidades

- **Usuarios-Pedidos-Pagos:** Cada usuario puede realizar múltiples pedidos, y cada pedido está asociado con un pago específico, lo que garantiza la trazabilidad.
- **Flores-Categorías:** La relación **muchos a muchos** permiten clasificar los productos de forma eficiente, facilitando la búsqueda y organización del catálogo.
- **Carrito-Detalle-Pedido:** El flujo entre el carrito y el detalle del pedido asegura que los datos se reutilicen sin redundancias durante la compra.

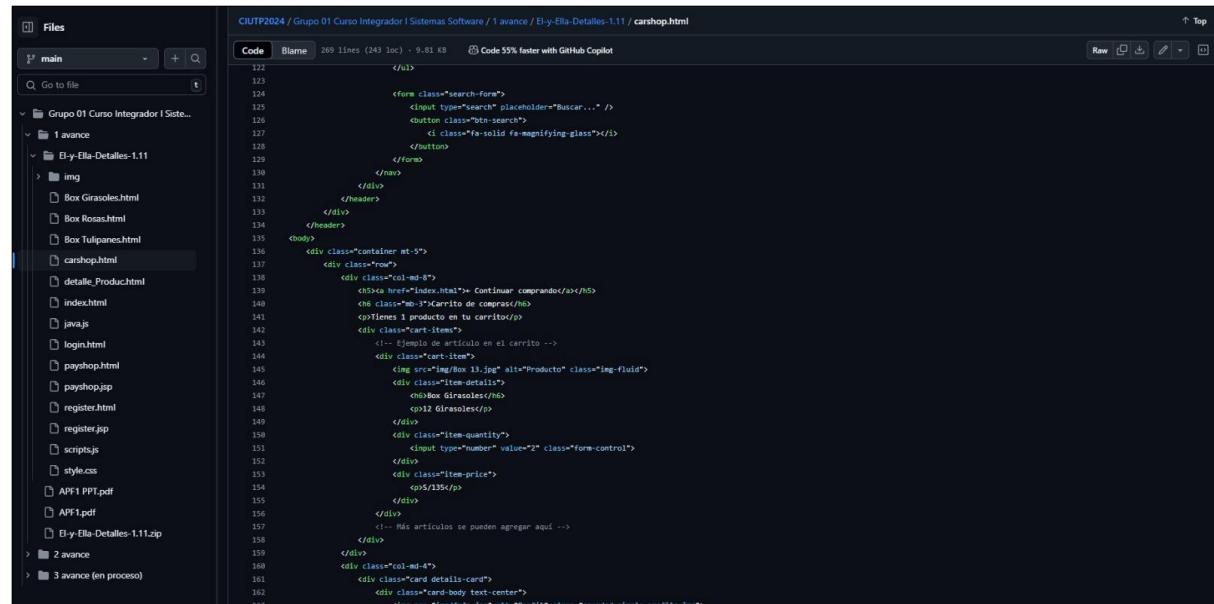
Este modelo de base de datos garantiza escalabilidad, integridad referencial y flexibilidad para futuras expansiones del sistema. Además, la implementación de relaciones **uno a muchos** y **muchos a muchos** optimiza la gestión de información, minimizando redundancias y asegurando que los datos se mantengan consistentes a lo largo del tiempo.

## 9.2.2 Revisión de Pull Request en GitHub

En esta sección se presenta un ejemplo de una revisión de código a través de un Pull Request (PR) en GitHub, donde se validan los cambios realizados en el proyecto con respecto a la documentación técnica y los informes. La revisión de PR es una práctica esencial en el desarrollo colaborativo, ya que permite asegurar la coherencia entre el código implementado y las especificaciones del proyecto.

A continuación, se muestran las líneas agregadas y/o eliminadas en los archivos para validar que las modificaciones cumplen con las directrices establecidas en la documentación técnica:

**Figura 95:**  
Estado anterior del archivo carshop.html



```
</ul>
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
617
618
619
619
620
621
622
623
623
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1
```

**Figura 96:**  
Estado modificado del archivo carshop.jsp

```

142     </header>
143     <body>
144         <div class="container mt-5">
145             <div class="row">
146                 <div class="col-md-8">
147                     <a href="AfterLogin.jsp">Continuar comprando</a></div>
148                     <h3 class="mb-3">Carrito de compras</h3>
149                 </div>
150             </div>
151             <%>
152             Integer userId = (Integer) session.getAttribute("user_id");
153             double subtotal = 0;
154             double envio = 20.00;
155             double total = 0;
156
157             if (userId != null) {
158                 CarritoDAO carritoDAO = new CarritoDAO();
159                 List carritos = carritoDAO.getCarritosByUserId(userId);
160
161                 if (carritos != null && !carritos.isEmpty()) {
162                     for (Carrito carrito : carritos) {
163                         subtotal += carrito.getPrecio() * carrito.getCantidad();
164                     }
165                     total = subtotal + envio; // Calculo total
166                     session.setAttribute("montoTotal", total); // Almacena el total en la sesión
167
168                 }
169             }
170
171             <p>Tienes <%= carritos.size()%> productos en tu carrito</p>
172             <div class="cart-items">
173                 <%>
174                 for (Carrito carrito : carritos) {
175                     <div class="cart-item d-flex align-items-center mb-3">
176                         <div class="item-image">
177                             
178                         <div class="item-details ms-3">
179                             <small class="mb-1">Flor ID: <%= carrito.getFlorId()%></small> <-- Se muestra el ID de la flor en lugar del nombre -->
180                             <p class="text-muted"><%= carrito.getCantidad()%> unidades</p>
181                         </div>
182                         <div class="item-quantity ms-3">
183                             <form action="ActualizarCarrito" method="post">
184                                 <input type="hidden" name="idCarrito" value="<%= carrito.getIdCarrito()%>">
185                                 <input type="number" name="cantidad" value="<%= carrito.getCantidad()%>" class="form-control" style="width: 70px;">
186                                 <button type="submit" class="btn btn-sm btn-primary mt-1">Actualizar</button>
187                             </form>
188                         </div>
189                         <div class="item-price ms-auto">
190                             <p class="mb-0"><%= String.format("%.2f", carrito.getPrecio() * carrito.getCantidad())%></p>
191                         </div>
192                     </div>
193                 }
194             <%>
195             <p>No tienes productos en tu carrito.</p>
196         <%>
197     </div>
198
199     <%>
200     </div>
201
202     <%>
203     <%>
204     <%>
205     <%>
206     <%>
207     </div>
208     <div class="col-md-4">
209         <div class="card card-details card7">

```

**Nota:** Esta imagen muestra la versión actualizada del archivo carshop.jsp, que incluye nuevas líneas de código para optimizar la funcionalidad del carrito de compras. En esta versión se han agregado cálculos dinámicos del subtotal y el total, utilizando datos extraídos de la base de datos a través del objeto CarritoDAO. Además, se implementa la gestión de sesiones para almacenar el monto total y se refina la presentación de los productos en el carrito, proporcionando una experiencia de usuario más eficiente y completa. **Fuente: Elaboración Propia.**

**Figura 97:**  
Implementación final del archivo carshop.jsp

```

166     <%>
167
168     <p>Tienes <%= carritos.size()%> productos en tu carrito</p>
169     <div class="cart-items">
170         <%>
171         for (Carrito carrito : carritos) {
172             <div class="cart-item d-flex align-items-center mb-3">
173                 <div class="item-image">
174                     
175                 <div class="item-details ms-3">
176                     <small class="mb-1">Flor ID: <%= carrito.getFlorId()%></small> <-- Se muestra el ID de la flor en lugar del nombre -->
177                     <p class="text-muted"><%= carrito.getCantidad()%> unidades</p>
178                 </div>
179                 <div class="item-quantity ms-3">
180                     <form action="ActualizarCarrito" method="post">
181                         <input type="hidden" name="idCarrito" value="<%= carrito.getIdCarrito()%>">
182                         <input type="number" name="cantidad" value="<%= carrito.getCantidad()%>" class="form-control" style="width: 70px;">
183                         <button type="submit" class="btn btn-sm btn-primary mt-1">Actualizar</button>
184                     </form>
185                 </div>
186                 <div class="item-price ms-auto">
187                     <p class="mb-0"><%= String.format("%.2f", carrito.getPrecio() * carrito.getCantidad())%></p>
188                 </div>
189             </div>
190         }
191     </div>
192
193     <%>
194     <%>
195     <%>
196     <%>
197     <%>
198     <%>
199     <%>
200     </div>
201
202     <%>
203     <%>
204     <%>
205     <%>
206     <%>
207     </div>
208     <div class="col-md-4">
209         <div class="card card-details card7">

```

**Nota:** Esta captura muestra la versión definitiva del archivo carshop.jsp. En ella se observa la estructura del carrito de compras con elementos dinámicos extraídos del backend. El formulario permite actualizar la cantidad de productos directamente desde el carrito, y se utiliza lógica adicional para verificar si el carrito contiene artículos o si está vacío. Esta integración asegura que los datos del carrito se sincronicen correctamente con la interfaz del usuario, mejorando así la usabilidad y funcionalidad del sistema. **Fuente: Elaboración Propia.**

La Figura 86 refleja el estado anterior del archivo `carshop.html`, donde se muestra el código base del carrito de compras antes de implementar modificaciones. Esta versión inicial sirvió como punto de partida para la revisión de los cambios solicitados.

Por otro lado, las figuras 87 y 88 corresponden al estado del archivo `carshop.jsp` después de la integración de nuevas líneas de código. En estas capturas se evidencia cómo se han añadido elementos para optimizar la funcionalidad del carrito, como la gestión dinámica de productos y cálculos automáticos de precios, asegurando una experiencia más robusta para los usuarios.

Este proceso de revisión mediante Pull Request (PR) en GitHub es esencial para mantener la calidad y coherencia del proyecto. Las revisiones permiten validar que los cambios estén alineados con los objetivos y la documentación técnica. Además, este flujo garantiza que cualquier contribución se realice de manera controlada y colaborativa, evitando conflictos en el código y asegurando que el sistema evolucione de forma ordenada.

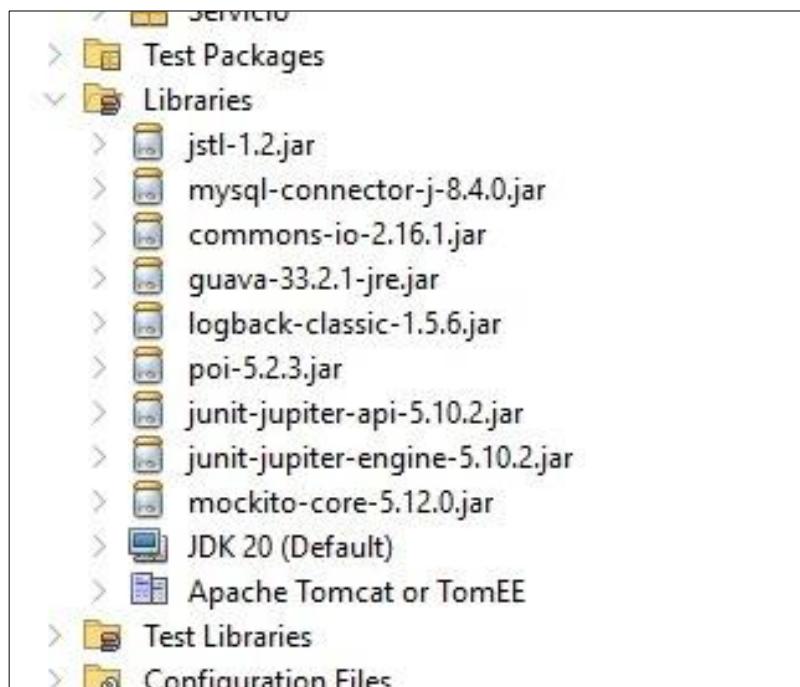
### 9.3 Buenas prácticas aplicadas en el desarrollo

Para garantizar la calidad y sostenibilidad del desarrollo, se han seguido estándares de la industria que aseguran un código limpio, eficiente y fácil de mantener. A continuación, se describen las principales buenas prácticas aplicadas en este proyecto.

#### 9.3.1 Uso de librerías adecuadas

En el desarrollo de este proyecto se utilizaron varias librerías esenciales que garantizan un código eficiente, mantenible y alineado con las mejores prácticas de la industria. La integración de estas librerías ha permitido optimizar tareas complejas, facilitar la conexión con servicios externos y mejorar el flujo de trabajo general. A continuación, se detalla el propósito de cada una de las librerías utilizadas:

**Figura 98:**  
Librerías utilizadas en el proyecto



**Nota:** La figura muestra las librerías integradas en el proyecto. Entre ellas se incluyen mysql-connector-j para la conexión con la base de datos MySQL, junit y mockito para la realización de pruebas unitarias, poi para el manejo de archivos Excel, logback para la gestión de logs, y jstl para facilitar la creación de páginas JSP dinámicas. **Fuente:** Elaboración Propia.

**a) Conexión a Bases de Datos – mysql-connector-j:**

Facilita la conexión entre la aplicación y la base de datos MySQL, permitiendo ejecutar consultas y gestionar datos de manera eficiente. Esta librería garantiza una comunicación estable y segura con el servidor de datos.

**b) Manejo de Archivos – commons-io y poi:**

- **commons-io:** Ofrece utilidades para trabajar con archivos y flujos de datos, simplificando operaciones comunes como la lectura, escritura y copiado de archivos.
- **poi:** Específicamente utilizada para trabajar con archivos de Microsoft Excel. Ha sido fundamental en la generación de reportes personalizados.

**c) Registro y Monitoreo – logback:**

Proporciona un sistema robusto de logging, lo que permite registrar eventos relevantes

durante la ejecución del sistema. Esto facilita la depuración y ayuda en la identificación de errores para asegurar la estabilidad de la aplicación.

**d) Pruebas Unitarias y Simulación – `junit` y `mockito`:**

- **JUnit:** Framework utilizado para realizar pruebas unitarias que garantizan la correcta funcionalidad de los módulos desarrollados.
- **Mockito:** Permite la simulación de dependencias, facilitando la creación de pruebas aisladas que evalúan el comportamiento del sistema sin necesidad de acceder a recursos externos.

**e) Dependencias y Gestión de Versiones – `guava`:**

Proporciona estructuras de datos avanzadas, utilidades comunes y mejoras de rendimiento, simplificando la gestión de dependencias y ayudando a escribir código más limpio y eficiente.

Cada una de estas librerías juega un papel clave en el desarrollo del proyecto. Gracias a su integración, se asegura un rendimiento óptimo, la reducción de errores y una fácil adaptación del sistema a futuras modificaciones o ampliaciones.

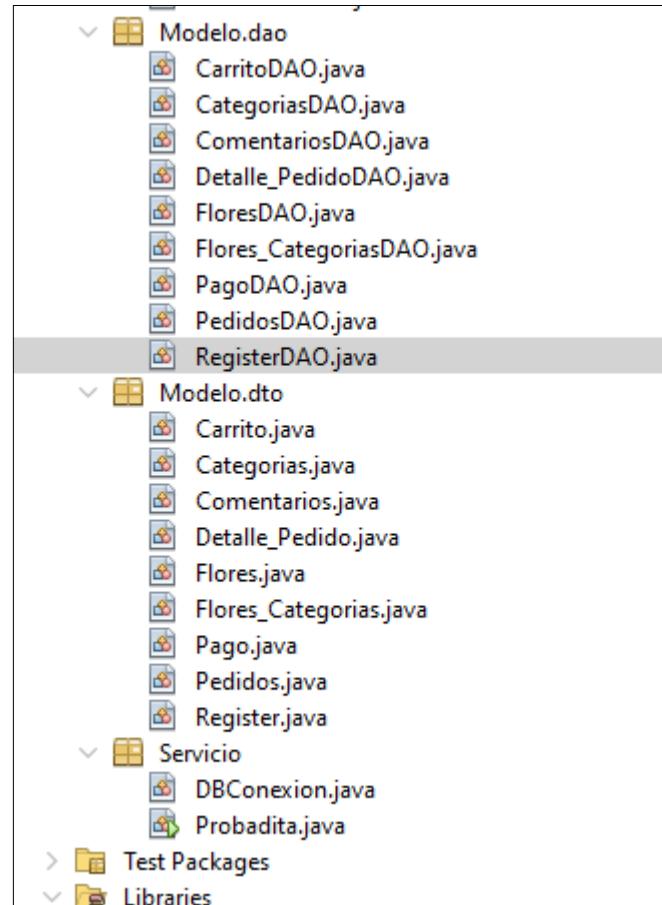
### 9.3.2 Uso de patrones de diseño

Se ha implementado el patrón **MVC (Modelo-Vista-Controlador)** para estructurar el sistema de manera eficiente y modular. Este patrón divide el desarrollo en tres componentes principales:

- a) **Capa Modelo (dao y dto):** Gestiona los datos y la lógica del negocio. Es responsable de acceder a la base de datos, realizar cálculos y gestionar la información.
- **DAO (Data Access Object):** En la carpeta `Modelo.dao` se encuentran las clases que gestionan las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) sobre la base de datos, como `FloresDAO.java` y `PedidosDAO.java`. Estas clases son intermedias entre la aplicación y la base de datos, permitiendo el acceso controlado a los datos.

- **DTO (Data Transfer Object):** En la carpeta `Modelo.dto` se agrupan las clases que representan las entidades del sistema (como `Flores.java` y `Pedidos.java`). Estas clases encapsulan los datos que se transfieren entre la aplicación y las diferentes capas del sistema.
- **Servicio:**  
Contiene clases auxiliares, como `DBConexion.java`, que gestionan la conexión con la base de datos, facilitando la integración del sistema con el almacenamiento de datos.

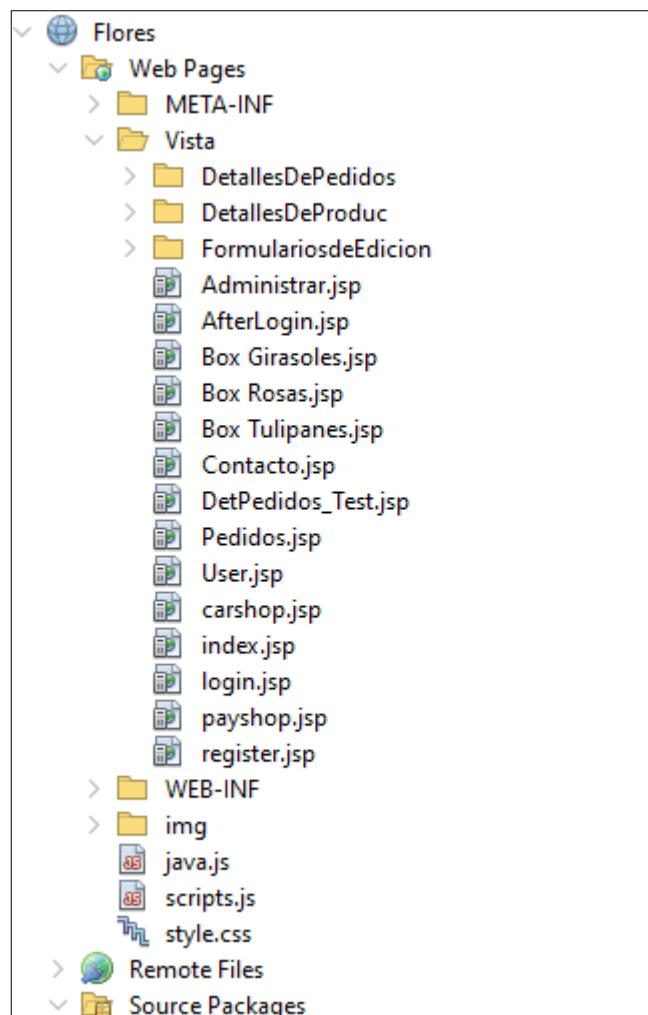
**Figura 99:**  
Estructura del Proyecto Java - Capa Modelo



*Nota: Muestra las clases DAO y DTO del modelo, encargadas de gestionar la lógica de acceso y transferencia de datos del sistema. Fuente: Elaboración Propia.*

- b) Vista:** Controla la interfaz de usuario, mostrando los datos al cliente y recogiendo la entrada del usuario mediante formularios u otros elementos visuales.

**Figura 100:**  
Estructura del Proyecto - Capa Vista



*Nota: Muestra las páginas JSP que componen la interfaz de usuario, permitiendo la interacción directa con el sistema a través de formularios y pantallas de navegación.*

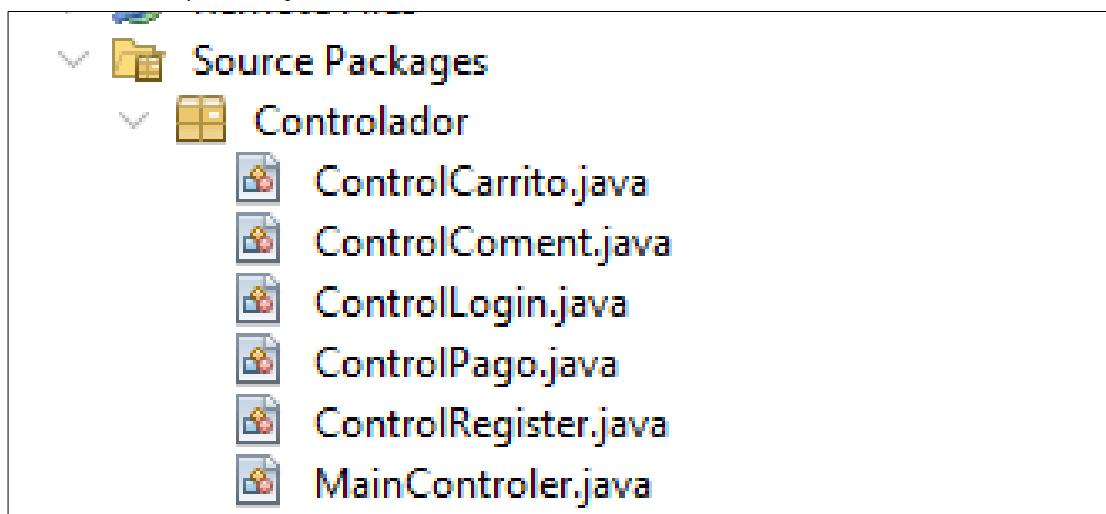
*Fuente: Elaboración Propia.*

- c) Controlador:** Actúa como intermediario entre el Modelo y la Vista. Recibe las peticiones del usuario desde la Vista, procesa los datos utilizando el Modelo y devuelve los resultados a la Vista para ser presentados.

Este enfoque mejora la mantenibilidad del sistema, permitiendo que las actualizaciones en la interfaz de usuario o en la lógica del negocio se realicen sin afectar el

resto del sistema. Además, facilita la escalabilidad, ya que cada componente es independiente y puede ser desarrollado o modificado sin impactar negativamente a los demás.

**Figura 101:**  
Estructura del Proyecto - Capa Controlador



**Nota:** Esta capa gestiona la lógica de control del sistema, conectando la interfaz de usuario con la capa de modelo. Los controladores aquí presentes manejan las solicitudes del usuario y gestionan el flujo de datos entre la vista y el modelo. **Fuente:** Elaboración Propia.

### 9.3.3 Software de control de versiones

Para el desarrollo de este proyecto se utilizó GitHub como software de control de versiones, el cual permitió llevar una gestión eficiente del código fuente y un control riguroso sobre cada etapa del desarrollo. A través de GitHub, fue posible organizar el trabajo en ramas, documentar cambios con mensajes claros en los commits y asegurar una integración continua que minimizara errores en las versiones finales.

El uso de ramas específicas por avance y funcionalidades facilitó que los desarrolladores pudieran trabajar en paralelo sin conflictos, realizando integraciones controladas solo cuando cada parte del sistema estaba completamente funcional. GitHub no solo sirvió como repositorio del código, sino que también permitió llevar un historial detallado del progreso, lo cual es fundamental para la trazabilidad del proyecto y la corrección de errores.

**Figura 102:**  
Árbol de avances en GitHub

The screenshot shows a GitHub repository interface. On the left, a sidebar titled 'Files' displays a file tree under 'main'. The tree includes a folder 'Grupo 01 Curso Integrador I Siste...' which contains three sub-folders: '1 avance', '2 avance', and '3 avance (en proceso)'. '1 avance' contains files like 'APF1 PPT.pdf', 'APF1.pdf', and 'El-y-Ella-Detalles-1.11.zip'. '2 avance' contains 'APF2 PPT.pdf', 'APF2.pdf', 'El y Ella Detalles 2.22.rar', and 'db\_flores 1.11.sql'. The right side of the screen shows a table titled 'LOAD-13 Add files via upload' with four rows of commit history. Each row has a file name, the message 'Add files via upload', and the date 'yesterday'. The columns are 'Name', 'Last commit message', and 'Last commit date'.

Name	Last commit message	Last commit date
..	Add files via upload	yesterday
APF1 PPT.pdf	Add files via upload	yesterday
APF1.pdf	Add files via upload	yesterday
El-y-Ella-Detalles-1.11.zip	Add files via upload	yesterday

**Nota:** Esta figura muestra la estructura del repositorio del proyecto en GitHub, organizada en avances específicos. Se destacan tres fases: "1 avance", "2 avance" y "3 avance (en proceso)", cada una con documentos relevantes como archivos PDF, scripts SQL y versiones empaquetadas del proyecto. Esta organización permite mantener un control eficiente del progreso del desarrollo y facilita la trazabilidad de los cambios realizados. **Fuente: Elaboración Propia.**

La figura presentada muestra la estructura del proyecto en GitHub, organizada en tres fases de avance:

- **1 avance:** Contiene las primeras versiones del sistema, junto con documentos preliminares como archivos PDF y versiones empaquetadas en ZIP.
- **2 avance:** Abarca actualizaciones significativas del sistema, como nuevas versiones del proyecto en RAR y archivos SQL para la base de datos.
- **3 avance (en proceso):** Indica que se encuentran en desarrollo las fases finales del proyecto, con nuevas funcionalidades aún en implementación.

Esta estructura organizada por avances permite que los miembros del equipo tengan una visión clara del estado del proyecto en cada momento, facilitando las revisiones de código y la validación de tareas pendientes. Además, GitHub ofrece herramientas de colaboración, como los pull requests y las revisiones de cambios, permitiendo que los desarrolladores puedan comentar y validar los aportes de sus compañeros antes de integrar definitivamente el código a la rama principal.

## 9.4 Autoría del proyecto y dominio del código

El proyecto ha sido desarrollado íntegramente por el equipo, lo que demuestra el compromiso y dominio de las herramientas y conceptos estudiados. Cada integrante ha trabajado en áreas específicas del sistema, garantizando que todos comprendan en profundidad cada componente del código.

### 9.4.1 Justificación de las elecciones de diseño

Las decisiones de diseño se han tomado basadas en la eficiencia y facilidad de mantenimiento del sistema. Por ejemplo, se eligió **MVC** como patrón de arquitectura porque permite separar la lógica del negocio de la interfaz gráfica, facilitando futuras modificaciones o ampliaciones del sistema. Además, se optó por implementar **DAO** para aislar la lógica de acceso a datos, lo que permite que el sistema sea más flexible frente a cambios en la base de datos o el modelo de datos.

### 9.4.2 Evidencia de autoría y conocimiento del sistema

El dominio del sistema se evidencia en la capacidad del equipo para explicar y justificar cada componente desarrollado. Cada miembro ha trabajado de forma autónoma en la implementación de funcionalidades específicas, y se ha realizado una revisión continua del trabajo mediante sesiones de codificación compartida para asegurar que todos los miembros comprendan el funcionamiento del sistema en su totalidad.

Durante las revisiones de avance, el equipo ha demostrado conocimiento profundo del sistema, explicando cada funcionalidad desarrollada y mostrando cómo las decisiones de diseño se alinean con los objetivos del proyecto. Esto no solo respalda la autoría del trabajo, sino que también asegura la capacidad del equipo para realizar futuras mejoras o adaptaciones en función de las necesidades del negocio.

## 10 Conclusiones

El desarrollo de este proyecto integró diversas buenas prácticas de ingeniería de software para asegurar la calidad, eficiencia y mantenimiento del sistema. A lo largo del proceso, se implementaron patrones de diseño adecuados, un control de versiones riguroso y una estructura modular que facilita futuras actualizaciones. A continuación, se presentan las conclusiones derivadas del proyecto:

1. El sistema desarrollado bajo el patrón MVC facilitó la separación de responsabilidades, permitiendo que cada componente (modelo, vista y controlador) se mantuviera modular y fácil de gestionar, lo que asegura una mayor escalabilidad y mantenimiento a futuro.
2. La utilización de GitHub como plataforma de control de versiones fomentó una colaboración efectiva dentro del equipo, permitiendo realizar un seguimiento detallado de los cambios, así como facilitar las revisiones continuas y correcciones en tiempo real.
3. Las decisiones de diseño tomadas en la creación del modelo de datos garantizaron una estructura clara y funcional para la gestión de información, lo que permitió una integración eficiente entre los diferentes módulos del sistema, como los pedidos, usuarios y productos.
4. El uso de librerías especializadas como MySQL Connector y JUnit contribuyó a optimizar tanto la conectividad con la base de datos como las pruebas unitarias del sistema, asegurando la fiabilidad del software desarrollado.
5. Finalmente, la documentación detallada y las revisiones de Pull Request garantizaron que el código entregado cumpliera con las especificaciones iniciales del proyecto, minimizando errores y permitiendo que el equipo mantuviera un alineamiento constante con los objetivos planteados desde el inicio.

## 11 Recomendaciones

A partir de las conclusiones obtenidas, es esencial delinear una serie de recomendaciones que orienten el desarrollo futuro del proyecto y su mantenimiento eficiente. Estas sugerencias buscan asegurar la evolución del sistema, preservando la calidad del software, optimizando su desempeño, y facilitando la colaboración entre los miembros del equipo. A continuación, se presentan las recomendaciones más relevantes:

1. Para mantener y mejorar el sistema en el futuro, se recomienda seguir una estructura modular basada en el patrón MVC, promoviendo así la escalabilidad y facilitando el mantenimiento continuo del software.
2. Se sugiere continuar utilizando GitHub como herramienta de control de versiones, fomentando prácticas de integración continua y revisiones colaborativas del código para garantizar la calidad del proyecto a lo largo del tiempo.
3. Es recomendable llevar a cabo sesiones periódicas de revisión y optimización del modelo de datos, asegurando que cualquier expansión del sistema mantenga la eficiencia y coherencia de la gestión de información.
4. Se aconseja incorporar nuevas librerías y frameworks según las necesidades futuras del proyecto, así como realizar pruebas automatizadas más extensivas para detectar posibles errores antes de pasar a producción.
5. Finalmente, se recomienda mantener la documentación siempre actualizada y alineada con los cambios realizados en el código, asegurando así una fácil comprensión por parte de los futuros desarrolladores o equipos de soporte.

## 12 Referencias

- Alonso-Dos-Santos, M., Soto-Fuentes, Y., y Valderrama-Palma, VA (2020). Determinantes de la lealtad de los usuarios de banca móvil. *Journal of Promotion Management*, 26(5), 615-633. <https://doi.org/10.1080/10496491.2020.1729315>
- Atlassian. (2022). Confluence: Documentación de proyectos y colaboración en equipo. <https://www.atlassian.com/es/software/confluence>
- Bass, L., Clements, P., & Kazman, R. (2020). Software architecture in practice (4th ed.). Addison-Wesley Professional.
- Behfar, K., & Okhuysen, G. A. (2020). Perspective—Integrating Micro and Macro Organizational Behavior: A Multilevel Framework for Management Scholarship. *Organization Science*, 31(3), 617-638. <https://doi.org/10.1287/orsc.2019.1333>
- Bhatti, A., Akram, H., Basit, HM, Khan, AU, Raza, SM y Naqvi, MB (2020). Tendencias del comercio electrónico durante la pandemia de COVID-19. *Revista internacional de comunicación y redes de las generaciones futuras*, 13(2), 1449-1452.
- CCOviedo. (2023). *Pruebas unitarias en MVC*. GitHub. [https://github.com/CCOviedo/UNIT\\_TEST](https://github.com/CCOviedo/UNIT_TEST)
- Chacon, S., & Straub, B. (2021). Pro Git (2nd ed.). Apress. <https://git-scm.com/book/en/v2>
- Choshin, M., y Ghaffari, A. (2023). El impacto del comercio electrónico en el desempeño de las pymes en un país en desarrollo. *Journal of Organizational Change Management*, 36(3), 477-491. <https://doi.org/10.1108/JOCM-07-2022-0200>
- Dingsøyr, T., Moe, N. B., & Seim, E. A. (2018). Coordinating Knowledge Work in Multiteam Programs: Findings From a Large-Scale Agile Development Program. *Project Management Journal*, 49(6), 64–77. <https://doi.org/10.1177/8756972818798980>

FRRe. (2024). *Pruebas unitarias y control de versiones en el desarrollo de software*.

<https://frre-ds.github.io/programa-analitico/unidad-05/>

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (2019). Design patterns: Elements of reusable object-oriented software. Addison-Wesley Professional.

García-Madariaga, J., Moreno-Gavara, C., & Virto, NR (2019). La importancia del marketing digital en el comercio electrónico. Revista de Estudios Económicos y Empresariales, 31, 149-170.

Garousi, V., Felderer, M., Karapıçak, Ç. M., & Yılmaz, U. (2021). Testing embedded software: A survey of the literature. Information and Software Technology, 131, 106487.

<https://doi.org/10.1016/j.infsof.2020.106487>

Hasan, R., Shams, R. y Rahman, M. (2021). Adopción de servicios bancarios móviles por parte de los consumidores: un examen empírico de los factores según las etapas de adopción. Journal of Retailing and Consumer Services, 59, 102372.

<https://doi.org/10.1016/j.jretconser.2020.102372>

Scribbr. (2020). APA citation guide (7th edition). Recuperado de <https://www.scribbr.com/apa-citation-generator/>

uo283706. (2024). *Ejemplo de pruebas unitarias y aplicación Java con Swing y MVC*. GitHub.

<https://github.com/uo283706/proyecto-inicial>

ychavez. (2023). *Taller de pruebas unitarias con XUnit y arquitectura limpia*. GitHub.

<https://github.com/ychavez/Taller-pruebas-unitarias>

## **13 Anexos**

Los anexos proporcionarán material complementario que respalda y amplía los contenidos presentados en este informe. Aquí se incluirán documentos relevantes, como diagramas, tablas, fragmentos de código y otros recursos que no forman parte del cuerpo principal, pero que son esenciales para una mejor comprensión del "Sistema de Comercio digital para la floristería El y Ella Detalles". A continuación, se presentarán los anexos, organizados de manera que el lector pueda acceder a información detallada y técnica que apoya los puntos abordados en las secciones anteriores.

### **13.1 Evidencias del trabajo en la empresa**

Esta sección presenta una recopilación de evidencias que documentan el trabajo realizado en colaboración con la floristería "El y Ella Detalles" durante el desarrollo del proyecto de Sistema de Comercio Digital. Las siguientes imágenes son evidencia de las operaciones de la floristería "El Y Ella Detalles". Muestran las entregas reales de arreglos florales a sus clientes:



## 13.2 Otros documentos relevantes

Esta sección contiene documentos adicionales que son fundamentales para comprender el proceso de desarrollo y los requisitos del Sistema de Comercio Digital para la floristería "El y Ella Detalles". Estos documentos proporcionan información valiosa sobre las necesidades de los usuarios, las expectativas del negocio y los requisitos técnicos del sistema.

### 13.2.1 Formato de una encuesta

Este documento presenta el formato de la encuesta diseñado para recopilar información directa de los usuarios actuales y potenciales de "El y Ella Detalles". La encuesta busca entender las preferencias de compra, la experiencia del usuario y las expectativas respecto a una plataforma de comercio electrónico para la floristería. A continuación, se muestra el formulario del cuestionario mediante varias capturas de pantalla:

Enlace de la encuesta:

[https://docs.google.com/forms/d/e/1FAIpQLSd1OviGVgKnTQThN9TaFwkYV-MvIkaQSU2-\\_95JnmmDyFWDOQ/viewform?usp=sf\\_link](https://docs.google.com/forms/d/e/1FAIpQLSd1OviGVgKnTQThN9TaFwkYV-MvIkaQSU2-_95JnmmDyFWDOQ/viewform?usp=sf_link)



**Encuesta para la floristería El y Ella detalles**

¡Gracias por participar en nuestra encuesta! Su opinión es muy valiosa para nosotros y nos ayudará a mejorar nuestro sistema de comercio digital para floristerías. Queremos asegurarnos de que nuestra plataforma cumpla con sus necesidades y expectativas. Por favor, responda las siguientes preguntas con la mayor sinceridad posible. La encuesta es anónima y confidencial.

kiarasanti@gmail.com [Cambiar de cuenta](#) 

 No compartido

\* Indica que la pregunta es obligatoria

Cual es tu nombre? \*

Kiara Santti

Cual es tu correo electrónico?

kiarasantti@gmail.com

¿Cuál es su edad? \*

- Menos de 18 años
- 18-24 años
- 25-34 años
- 35-44 años
- 45-54 años

¿Con qué frecuencia realiza compras en línea de productos de floristería? \*

- Diario
- Semanal
- Mensual
- Raramente
- Nunca

¿Cómo calificaría su satisfacción con el sistema actual de comercio digital para floristerías? \*

- Muy insatisfecho/a
- Insatisfecho/a
- Neutral
- Satisfecho/a
- Muy satisfecho/a

¿Qué aspectos de la plataforma de comercio digital le gustan más? \*

- Variedad de productos
- Facilidad de uso
- Seguridad en el pago
- Atención al cliente

¿Qué aspectos de la plataforma de comercio digital cree que podrían mejorarse? \*

- Variedad de productos
- Facilidad de uso
- Seguridad en el pago
- Atención al cliente
- Otros (especifique): [Espacio para respuesta]

¿Tiene algún comentario adicional o sugerencia sobre nuestra plataforma de comercio digital para floristerías?

Tu respuesta

### Agradecimientos

*¡Gracias por su tiempo y participación! Sus respuestas son muy importantes para nosotros. Si desea recibir actualizaciones sobre los resultados de la encuesta o participar en futuras investigaciones, por favor proporcione su correo electrónico a continuación (opcional)*

**Enviar**

**Borrar formulario**

Nunca envíes contraseñas a través de Formularios de Google.

Este contenido no ha sido creado ni aprobado por Google. [Denunciar abuso](#) - [Términos del Servicio](#) - [Política de Privacidad](#)

**Google Formularios**

#### 13.2.2 Formato de una entrevista.

Este apartado contiene el guión de la entrevista realizada a la dueña de "El y Ella Detalles". La entrevista está diseñada para obtener información detallada sobre la visión del negocio, los procesos actuales, las necesidades específicas y las expectativas para el nuevo sistema de comercio digital. A continuación, se muestran los documentos que contiene el formato de la entrevista:

**Formato de Entrevista para el proyecto Sistema de Comercio digital para la floristería "El y Ella Detalles" en Java**

**1. Información Básica**

**Fecha de la Entrevista:** 30 de agosto de 2024

**Hora de Inicio:** 2:00 PM

**Duración Estimada:** 1 hora y media

**Entrevistador/a:** Joaquín Loa Denegri

**Entrevistado/a:** Melody Loa Denegri

**Puesto/Posición del Entrevistado/a:** Dueña y Creadora

**Empresa/Organización:** El y Ella Detalles

**Lugar:** Carabayllo

**2. Objetivo de la Entrevista**

Conocer la historia, visión y experiencias de la dueña y creadora de "El y Ella Detalles," así como obtener una comprensión más profunda de los retos y logros en la gestión de una floristería.

**3. Preguntas**

**Introducción:**

**1. ¿Podrías contarnos un poco sobre ti y cómo surgió la idea de crear "El y Ella Detalles"?**

Hace relativamente poco empecé unos hobbies por la jardinería con algunas plantas en mi hogar. Además, siempre tuve una inclinación por las manualidades y disfrutaba creando regalos hechos a mano para mi familia y amigos. Fue esta combinación de amor por las plantas y la creatividad en las manualidades lo que me inspiró a fundar 'El y Ella Detalles'. Quería compartir mi pasión por la naturaleza y la habilidad de crear algo único y significativo para los demás.

**2. ¿Qué te motivó a especializarte en el negocio de la floristería?**

La floristería es el perfecto equilibrio entre mi amor por la jardinería y mi habilidad para las manualidades. Me encanta la idea de transformar flores frescas en arreglos hermosos que puedan alegrar el día de alguien o celebrar momentos especiales. Además, ver la satisfacción en los rostros de mis clientes cuando reciben un arreglo personalizado es increíblemente gratificante. Especializarme en floristería me permite combinar mis pasiones y ofrecer algo que realmente aporta belleza y alegría a la vida de las personas.

**Historia y Desarrollo de la Empresa:**

**3. ¿Cuáles fueron los principales retos que enfrentaste al iniciar "El y Ella Detalles"?**

Uno de los mayores retos fue establecer una marca en un mercado competitivo. Al principio, era difícil hacer que la gente conociera y confiara en 'El y Ella Detalles'.

También hubo desafíos logísticos, como encontrar proveedores confiables y gestionar las operaciones diarias mientras mantenía un alto estándar de calidad.

**4. ¿Cómo ha evolucionado la empresa desde su creación hasta el día de hoy?**

La empresa ha crecido de ser un pequeño emprendimiento local a una marca reconocida en la comunidad. Hemos ampliado nuestra gama de productos, mejorado nuestros servicios, y adoptado tecnologías que nos permiten llegar a más personas y ofrecer una experiencia de compra más fluida.

**5. ¿Podrías compartir alguna anécdota o momento significativo que hayas vivido durante el crecimiento de la empresa?**

Recuerdo claramente una ocasión en la que un cliente nos pidió un arreglo muy especial para pedirle matrimonio a su pareja. Diseñamos un arreglo personalizado con sus flores favoritas y un mensaje oculto en él. Fue un momento muy emotivo cuando nos contó que su pareja había dicho que sí. Esos momentos son los que realmente me recuerdan por qué hago lo que hago.

**Productos y Servicios:**

**6. ¿Qué consideras que distingue a "El y Ella Detalles" de otras floristerías en el mercado?**

Nos distinguimos por la personalización y la atención al detalle. Cada arreglo que sale de 'El y Ella Detalles' es único y hecho a mano con mucho cuidado. Nos tomamos el tiempo para entender las necesidades de nuestros clientes y crear algo que realmente resuene con ellos.

**7. ¿Cómo decides qué tipos de arreglos florales o productos ofrecer a tus clientes?**

Es una combinación de escuchar a nuestros clientes, estar al tanto de las tendencias, y mantener un catálogo de opciones que refleje nuestra identidad. También me inspiro en la temporada y en eventos especiales. Me gusta mantener un equilibrio entre lo clásico y lo moderno.

**8. ¿Qué herramientas o estrategias digitales has implementado para mejorar el servicio al cliente y las ventas?**

Estamos queriendo implementar un sistema de comercio digital que facilita las compras en línea, permitiendo a los clientes personalizar sus pedidos y realizar pagos de forma segura. También usamos las redes sociales para interactuar con nuestros clientes y compartir ideas e inspiraciones.

**Visión y Futuro:**

**9. ¿Qué visión tienes para el futuro de "El y Ella Detalles"?**

Me gustaría ver a 'El y Ella Detalles' expandiéndose a nuevas ubicaciones y ofreciendo una gama más amplia de productos, como decoraciones para eventos y servicios de diseño floral personalizado para bodas. También quiero seguir innovando en la experiencia del cliente, integrando más tecnología para hacer las compras aún más convenientes.

**10. ¿Hay algún proyecto o iniciativa nueva en la que estés trabajando actualmente?**

Actualmente, estamos desarrollando una línea de arreglos ecológicos que utilizan materiales sostenibles y flores cultivadas de manera ética. También estamos trabajando en mejorar nuestra plataforma en línea para ofrecer una experiencia de usuario aún más fluida y personalizada.

**Cultura y Valores:**

**11. ¿Cómo te aseguras de que "El y Ella Detalles" refleje los valores y la cultura que consideras importantes?**

Es fundamental para mí que cada miembro del equipo entienda y comparta la visión de la empresa. Nos enfocamos en la calidad, la atención al detalle, y el cuidado por nuestros clientes. También promovemos un ambiente de trabajo inclusivo y respetuoso, donde cada idea es valorada.

**12. ¿Qué es lo que más valoras en tu equipo de trabajo?**

Valoro la pasión y el compromiso. Nuestro equipo no solo es talentoso, sino que también comparte un amor genuino por lo que hacemos. Es inspirador trabajar con personas que están dispuestas a ir más allá para hacer que cada detalle cuente.

**Preguntas Finales:**

**13. ¿Tienes algún consejo para aquellos que están pensando en emprender en el negocio de la floristería?**

Mi consejo sería que sigan su pasión, pero también que estén preparados para trabajar duro y ser resilientes. La floristería puede ser un negocio desafiante, pero es

increíblemente gratificante cuando amas lo que haces. No tengan miedo de ser creativos y de buscar siempre maneras de mejorar.

**14. ¿Hay algo más que te gustaría añadir o destacar sobre "El y Ella Detalles"?**

Solo quiero agradecer a todos nuestros clientes y al equipo que ha estado conmigo desde el principio. 'El y Ella Detalles' es más que una floristería; es un reflejo de amor, arte y comunidad. Estoy emocionada por lo que el futuro tiene reservado para nosotros.

**Notas del Entrevistador/a**

**Observaciones sobre el Entrevistado/a:** Melody muestra una gran pasión y compromiso con su empresa. Su visión clara y su enfoque en la calidad y el servicio al cliente son notables. Se percibe una profunda conexión personal con su negocio, lo cual es evidente en cómo habla de sus experiencias y desafíos.

**Fortalezas Identificadas:** Liderazgo fuerte, creatividad, enfoque en la personalización y el detalle, capacidad para adaptarse y crecer en un mercado competitivo.

**Áreas de Mejora:** Continuar explorando nuevas tecnologías y estrategias de marketing para expandir el alcance de la floristería.

**Recomendación:** Sí

**Cierre de la Entrevista**

**Hora de Finalización:** 3:30 PM

**Próximos Pasos:** Revisar las oportunidades de expansión digital mencionadas en la entrevista y considerar la implementación de la nueva línea de productos ecológicos.

**Comentarios Finales del Entrevistador/a:** Melody demostró una sólida comprensión del mercado y una visión clara para el futuro. Su enfoque en la calidad y la personalización la distingue en el sector, y su capacidad para innovar asegura que "El y Ella Detalles" siga siendo una floristería líder en la comunidad.

### 13.2.3 Listado de requerimientos funcionales del proyecto

En esta sección se presenta un listado detallado de los requisitos funcionales del sistema. Estos requisitos describen las funcionalidades específicas que el Sistema de Comercio Digital debe proporcionar, definiendo qué debe hacer el sistema desde la perspectiva del usuario y del negocio.

A continuación, se muestra el documento que contiene los requerimientos funcionales del proyecto:

# **Listado de Requerimientos Funcionales para el proyecto Sistema de Comercio digital para la floristería "El y Ella Detalles" en Java**

## **1. Introducción**

**Nombre del Proyecto:** Sistema de Comercio Digital para "El y Ella Detalles"

**Fecha:** 28 de agosto del 2024

**Versión del documento:** 1.0

**Autor:** Equipo de estudiantes del curso Integrador 1: Sistemas - Software

**Revisión:** 01 de setiembre del 2024

## **2. Objetivo del Documento**

Este documento describe los requerimientos funcionales para el Sistema de Comercio Digital de la floristería "El y Ella Detalles". Los requerimientos funcionales especifican las funciones que el sistema debe realizar para satisfacer las necesidades del negocio y los criterios de éxito del proyecto académico.

## **3. Requisitos Funcionales**

### **3.1. Gestión de usuarios**

#### **1. Registro de usuario**

El sistema debe permitir a los nuevos usuarios registrarse proporcionando información como nombre, correo electrónico y contraseña.

#### **2. Inicio de Sesión**

Los usuarios deben poder iniciar sesión con sus credenciales (correo electrónico y contraseña).

### **3. Gestión de Perfil**

Los usuarios deben poder actualizar su información personal, dirección de envío y preferencias de compra.

### **4. Recuperación de Contraseña**

El sistema debe permitir a los usuarios recuperar o restablecer su contraseña mediante un correo electrónico de recuperación.

## **3.2 Catálogo de productos**

### **1. Visualización de productos**

El sistema debe mostrar un catálogo de productos con imágenes, descripciones y precios.

### **2. Categorización**

Los productos deben estar organizados en categorías (por ejemplo, ramos, arreglos florales, plantas).

### **3. Búsqueda y filtrado**

Los usuarios deben poder buscar productos por nombre, categoría o tipo de ocasión.

### **4. Detalles del Producto**

Al seleccionar un producto, el sistema debe mostrar información detallada, incluyendo opciones de personalización disponibles.

### **3.3. Carrito de compras**

#### **1. Agregar al carrito**

Los usuarios deben poder añadir productos al carrito de compras.

#### **2. Modificar Carrito**

El sistema debe permitir a los usuarios modificar la cantidad de productos o eliminarlos del carrito.

#### **3. Resumen del Carrito**

El sistema debe mostrar un resumen del carrito con el total de la compra, incluyendo impuestos y costos de envío.

### **3.4. Proceso de compra**

#### **1. Selección de entrega**

Los usuarios deben poder elegir entre recoger en tienda o envío a domicilio.

#### **2. Programación de entrega**

Para envíos a domicilio, el sistema debe permitir seleccionar fecha y franja horaria de entrega.

#### **3. Proceso de pago**

El sistema debe simular el proceso de pago, incluyendo la selección de método de pago (por ejemplo, tarjeta de crédito, PayPal).

#### **4. Confirmación de pedido**

Después de completar la compra, el sistema debe generar una confirmación de pedido con un número de seguimiento.

### **3.5. Gestión de inventario**

#### **1. Actualización de stock**

El sistema debe actualizar automáticamente el inventario después de cada venta.

#### **2. Alertas de Stock Bajo**

El sistema debe generar alertas cuando el stock de un producto esté por debajo de un umbral predefinido.

### **3.6. Personalización de Productos**

#### **1. Opciones de personalización**

Los usuarios deben poder personalizar ciertos aspectos de los arreglos florales (por ejemplo, color de las flores, tipo de florero).

### **3.7. Interfaz de administración**

#### **1. Gestión de productos**

Los administradores deben poder agregar, editar y eliminar productos del catálogo.

#### **2. Gestión de pedidos**

Los administradores deben poder ver y gestionar los pedidos recibidos.

#### **3. Informes Básicos**

El sistema debe generar informes básicos de ventas y productos más vendidos.

#### **4. Criterios de aceptación**

##### **Pruebas de Funcionalidad**

El sistema debe ser probado para asegurar que cumple con todos los requisitos funcionales especificados.

##### **Revisión de usuario**

Los usuarios finales (representados por el profesor y compañeros de clase) deben revisar y aprobar que el sistema cumple con las necesidades y expectativas establecidas.

#### **5. Notas adicionales**

##### **Limitaciones conocidas:**

El sistema es un prototipo académico y no puede incluir todas las funcionalidades de un sistema comercial completo.

##### **Consideraciones especiales:**

El sistema debe ser desarrollado utilizando Java para al menos el 50% del backend.

Se debe priorizar la usabilidad y la experiencia de usuario en el diseño de la interfaz.

#### 13.2.4 Listado de requerimientos no funcionales del proyecto

Este apartado contiene un listado de los requisitos no funcionales del sistema. Estos requisitos especifican criterios que pueden usarse para juzgar la operación de un sistema, en lugar de sus comportamientos específicos. Incluyen aspectos como rendimiento, seguridad, usabilidad y escalabilidad del Sistema de Comercio Digital.

A continuación, se muestra el documento que contiene los requerimientos no funcionales del proyecto:

**Listado de Requerimientos Funcionales para el proyecto Sistema de Comercio digital para la floristería "El y Ella Detalles" en Java**

**1. Introducción**

**Nombre del Proyecto:** Sistema de Comercio Digital para "El y Ella Detalles"

**Fecha:** 28 de agosto del 2024

**Versión del documento:** 1.0

**Autor:** Equipo de estudiantes del curso Integrador 1: Sistemas - Software

**Revisión:** 01 de setiembre del 2024

**2. Objetivo del Documento**

Este documento describe los requerimientos no funcionales para el Sistema de Comercio Digital de la floristería "El y Ella Detalles". Los requerimientos no funcionales definen los atributos cualitativos del sistema y cómo debe comportarse en diversos contextos.

**3. Requisitos No Funcionales**

**3.1. Rendimiento**

**1. Tiempo de respuesta**

El sistema debe cargar las páginas en menos de 3 segundos en condiciones normales de red.

**2. Escalabilidad**

El sistema debe soportar hasta 100 usuarios simultáneos sin degradar el rendimiento.

### **3. Capacidad de Procesamiento**

El sistema debe ser capaz de procesar 50 transacciones por minuto.

#### **3.2. Disponibilidad y Fiabilidad**

##### **1. Tiempo de disponibilidad**

El sistema debe tener una disponibilidad del 99% durante el horario operativo de la floristería.

##### **2. Recuperación ante Fallos**

El sistema debe ser capaz de recuperarse en menos de 30 minutos después de una interrupción.

#### **3.3. Seguridad**

##### **1. Autenticación**

El sistema debe implementar autenticación segura para proteger las cuentas de usuario.

##### **2. Autorización**

Los permisos de acceso deben estar basados en roles (cliente, administrador).

##### **3. Protección de datos**

Los datos sensibles (como información de pago) deben ser cifrados usando algoritmos estándar.

#### **3.4 Usabilidad**

##### **1. Interfaz de usuario**

La interfaz debe ser intuitiva y fácil de usar, con un diseño responsive para diferentes dispositivos.

## **2. Accesibilidad**

El sistema debe cumplir con las pautas básicas de accesibilidad web WCAG 2.1 nivel A.

## **3.5 Mantenimiento**

### **1. Facilidad de actualización**

El sistema debe ser modular para permitir actualizaciones y mantenimiento sin interrumpir todo el servicio.

### **2. Documentación**

Se debe proporcionar documentación clara del código y manuales de usuario.

## **3.6 Compatibilidad**

### **1. Compatibilidad con Navegadores**

El sistema debe ser compatible con las versiones más recientes de Chrome, Firefox y Safari.

### **2. Compatibilidad con Dispositivos**

El sistema debe funcionar correctamente en dispositivos móviles y computadoras de escritorio.

## **3.7 Portabilidad**

### **1. Implementación**

El sistema debe poder estar desplegado en un entorno de hosting compartido estándar.

### **3.8. Legalidad y Cumplimiento**

#### **1. Privacidad**

El sistema debe cumplir con las regulaciones básicas de protección de datos personales.

#### **2. Criterios de aceptación**

##### **Pruebas de rendimiento**

Se realizarán pruebas para asegurar que el sistema cumple con los requisitos de tiempo de respuesta y capacidad.

##### **Evaluación de usabilidad**

Se realizarán pruebas de usuario para verificar la facilidad de uso de la interfaz.

#### **5. Notas adicionales**

##### **Consideraciones especiales:**

El sistema es un prototipo académico y no puede incluir todas las medidas de seguridad de un sistema comercial.

##### **Limitaciones conocidas:**

La escalabilidad y el rendimiento pueden estar limitados por el entorno de hosting utilizado para el proyecto académico.

### 13.2.5 Procedimiento de Gestión de Inventario - Buena Práctica de Documentación

A continuación, presenta un ejemplo destacado de buena práctica de documentación en el contexto del Sistema de Comercio Digital para la floristería 'El y Ella Detalles'. El documento 'Procedimiento de Gestión de Inventario', desarrollado por el Equipo de Desarrollo 01, ilustra cómo una documentación bien estructurada y detallada puede mejorar significativamente la eficiencia operativa y la comprensión de los procesos clave del sistema.

#### **Protocolo de Documentación y Buenas Prácticas para el Sistema de Comercio Digital de la Floristería “El y Ella Detalles”**

##### **Procedimiento de Gestión de Inventario**

Versión: 1.0

Fecha de Creación: 23/09/2024

Última Modificación: 25/09/2024

Elaborado por: Equipo 01 de estudiantes del Curso Integrador I: Sistemas Software-Sección 19502 – Sistema de Comercio Digital El y Ella Detalles

##### **Descripción General**

Este documento detalla el procedimiento para gestionar el inventario dentro del sistema de comercio digital de la floristería El y Ella Detalles. El inventario es actualizado en tiempo real para reflejar la disponibilidad de los productos, y se genera una alerta cuando el stock de un producto llega al nivel mínimo establecido. También se incluyen pasos para añadir nuevos productos, modificar el stock de productos existentes y eliminar productos que ya no estén disponibles.

##### **Contexto**

El presente documento está diseñado para ser una guía exhaustiva que cubra las **buenas prácticas de documentación** aplicadas al **Sistema de Comercio Digital para la Floristería “El y Ella Detalles” en Java**. Estas prácticas están orientadas a establecer estándares claros y eficientes para la gestión de la información y los procesos del sistema. A través de una documentación estructurada, buscamos garantizar la transparencia, accesibilidad y capacidad de mantenimiento del sistema, facilitando la implementación de mejoras continuas y la resolución eficiente de problemas.

La floristería “Él y Ella Detalles S.A.C.”, fundada en 2022, se especializa en la creación y venta de arreglos florales personalizados para diversas ocasiones. Debido a la necesidad de adaptarse al entorno digital y mejorar sus procesos, se ha

desarrollado este proyecto de comercio digital, el cual integra funciones como la gestión de inventarios, la personalización de productos, la administración de pedidos y la automatización de la facturación y entregas.

Este documento define el protocolo de documentación que todos los involucrados en el proyecto deben seguir, estableciendo un formato claro y estructurado para facilitar la colaboración entre los equipos y asegurar que cada proceso sea comprensible y replicable. La implementación de buenas prácticas permitirá mejorar la eficiencia operativa, reducir errores, y garantizar que cualquier modificación o mejora futura sea debidamente registrada.

## Alcance

El sistema abarca desde la **gestión de inventarios** hasta la **personalización de productos**, pasando por el **procesamiento de pedidos y pagos**. Este documento describe los procedimientos operativos de estos procesos, con el objetivo de que todo el personal involucrado tenga una referencia clara y precisa para llevar a cabo sus tareas de manera eficiente.

El alcance también incluye:

- La elaboración de un **manual de procedimientos** que describa cada proceso clave.
- La creación de **plantillas de documentación** para garantizar uniformidad en la presentación de casos de uso, diagramas de flujo y cualquier otro tipo de documento.
- La implementación de un **sistema de control de versiones** para asegurar que cualquier cambio en el sistema sea rastreable y pueda revertirse si es necesario.
- Un **protocolo de retroalimentación** para mejorar continuamente tanto el sistema como la documentación.

## **Objetivo**

El objetivo principal de este documento es establecer un protocolo de documentación estandarizado y buenas prácticas que garanticen la correcta documentación, actualización y gestión de los procesos y requerimientos del **Sistema de Comercio Digital de la Floristería “El y Ella Detalles”**. A través de estas directrices, se busca asegurar que la información sea clara, accesible y eficiente, facilitando la comprensión y el uso del sistema por todos los involucrados, además de promover la mejora continua y la adaptación a cambios en el negocio o en la tecnología.

## **Entradas y Salidas**

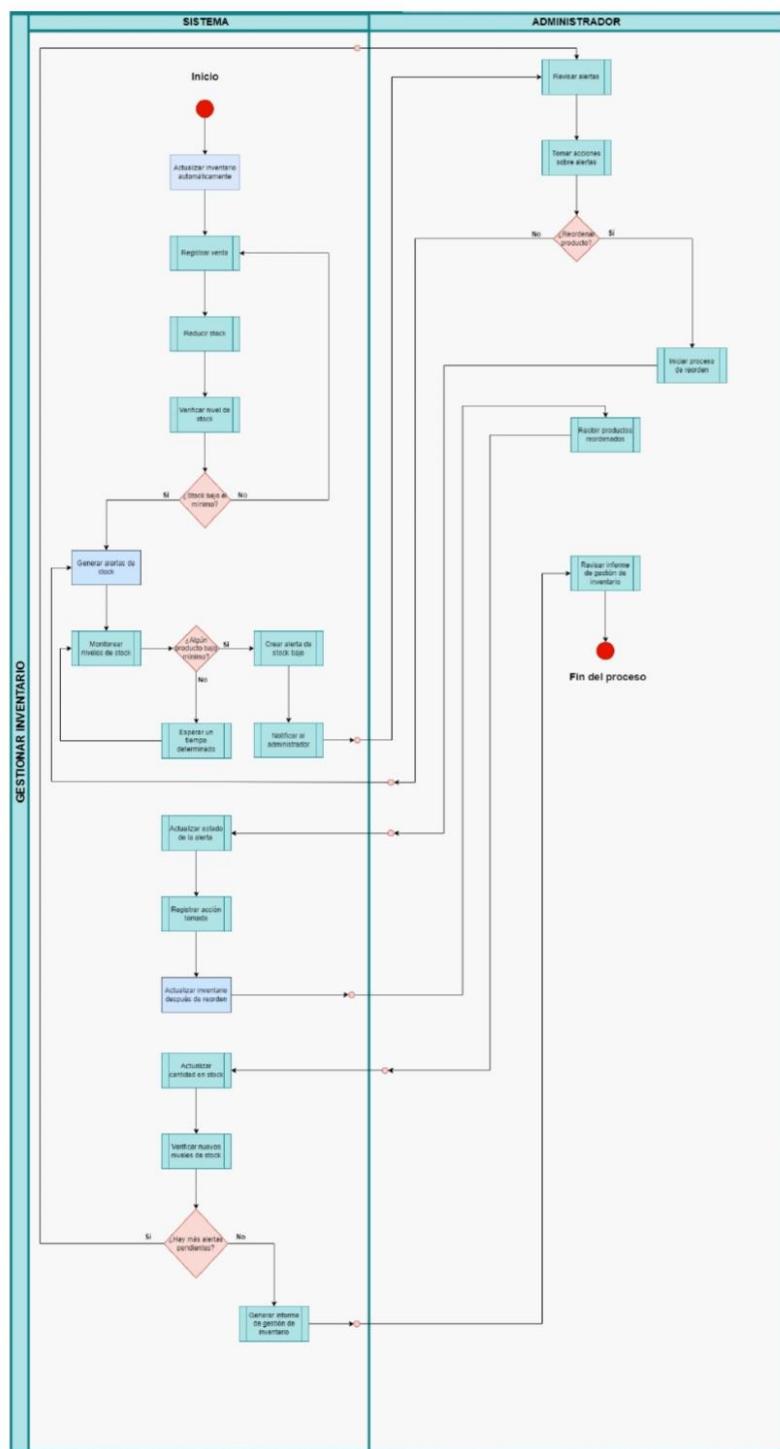
### **Entradas:**

Datos del producto (nombre, tipo de flor, cantidad en stock, precio).  
Información sobre las ventas (cantidad vendida, cliente, fecha de venta).  
Stock mínimo permitido (definido por el administrador del sistema).

### **Salidas:**

Confirmación de la actualización del stock.  
Notificación de necesidad de reabastecimiento.  
Informe de productos fuera de stock.  
Registro de historial de inventario para análisis y toma de decisiones.

## Diagrama de Flujo



4

## Roles y Responsabilidades

Rol	Responsabilidad
Administrador del sistema	Verificar y gestionar los módulos de inventario y pedidos.
Encargado del inventario	Actualizar manualmente los productos disponibles en caso de cambios.
Cliente	Realizar pedidos y revisar productos personalizados.
Responsable de envíos	Procesar la entrega y confirmar la salida del pedido al cliente.

## Indicadores de Desempeño

Indicador	Descripción	Meta
Tiempo de actualización de stock	Tiempo que toma actualizar el stock de un producto una vez realizada una venta.	Inmediato (< 1 min)
Tiempo de respuesta a una alerta de reabastecimiento	Tiempo desde que se emite una alerta de stock bajo hasta que se solicita reabastecimiento.	< 2 horas
Nivel de cumplimiento de stock mínimo	Porcentaje de veces que el inventario se encuentra por encima del nivel mínimo permitido.	100%
Nivel de satisfacción del cliente	Porcentaje de clientes que expresan satisfacción respecto a la disponibilidad de productos.	> 95%

## **Casos de Uso**

### **Caso de Uso 1: Actualización de Inventario Automática Tras Venta**

**Actor principal:** Administrador del Sistema

#### **Acciones:**

1. El sistema recibe la notificación de una nueva venta.
2. El módulo de inventario se actualiza automáticamente, descontando la cantidad vendida del stock disponible.
3. El sistema verifica si el stock restante está por debajo del nivel mínimo permitido.
4. Si el stock está por debajo del nivel mínimo, el sistema genera una alerta para reabastecimiento.

#### **Flujo principal:**

- El sistema actualiza el inventario automáticamente tras cada venta.

#### **Flujos alternativos:**

- Si no se puede actualizar el inventario debido a un error en la base de datos, se genera una alerta para el administrador del sistema.

#### **Excepciones:**

- Si el sistema no tiene acceso al inventario (por ejemplo, por un fallo en la conexión con la base de datos), el pedido se coloca en cola y se reintenta actualizar más tarde.

## Caso de Uso 2: Adición de Nuevos Productos al Inventario

**Actor principal:** Encargado de Inventario

### Acciones:

1. El encargado del inventario accede al módulo de inventario del sistema.
2. Selecciona la opción "Añadir nuevo producto".
3. Ingresa la información del producto (nombre, tipo, cantidad inicial, precio, stock mínimo).
4. El sistema valida la información y guarda el nuevo producto en la base de datos.

### Flujo principal:

- El encargado añade un nuevo producto, el sistema lo valida y lo añade al inventario.

### Flujos alternativos:

- Si la información del producto está incompleta o contiene errores, el sistema muestra un mensaje de advertencia y solicita la corrección.

### Excepciones:

- Si ocurre un error de conexión durante la actualización, el producto no se guarda y el encargado debe reintentarla más tarde.

## Inventario del Sistema

El inventario del sistema consiste en los siguientes módulos que interactúan entre sí para garantizar el correcto funcionamiento de la gestión de pedidos y stock:

1. **Módulo de Inventario:** Responsable de registrar el stock disponible, actualizar las cantidades después de cada venta y generar alertas de reabastecimiento.

2. **Módulo de Pedidos:** Procesa las órdenes de los clientes, descuenta el stock en tiempo real y notifica al cliente sobre la disponibilidad de los productos.
3. **Módulo de Pagos:** Maneja la confirmación del pago y activa la actualización del inventario una vez que la transacción se completa.
4. **Módulo de Envíos:** Controla el estado del pedido y notifica al cliente cuando el producto ha sido enviado.

## **Implementar un Manual de Procedimientos Digitales**

### **Contenido del Manual:**

El manual será una guía detallada que permita a los empleados de la floristería manejar el sistema de comercio digital de manera eficiente. Este manual debe estar disponible en formato digital para fácil acceso y actualizaciones constantes.

### **1. Cómo realizar pedidos personalizados:**

#### **• Descripción del Proceso:**

Los clientes podrán elegir entre arreglos florales preconfigurados o personalizar su propio pedido. El sistema verificará en tiempo real la disponibilidad de las flores o productos solicitados.

#### **Pasos:**

1. El cliente accede al sitio web y selecciona un arreglo predefinido o ingresa a la sección de personalización.
2. El cliente selecciona el tipo de flores, el color, la cantidad y el estilo de envoltura.
3. El sistema verifica la disponibilidad del inventario.
4. Se calcula el precio en tiempo real.
5. El cliente confirma y procede al pago.
6. El sistema genera un número de orden y notifica al cliente sobre la confirmación del pedido.

## **2. Gestión de inventarios:**

- **Descripción del Proceso:**

El sistema actualizará automáticamente el inventario con cada venta o ingreso de productos nuevos.

**Pasos:**

1. Cuando se recibe un nuevo lote de flores, el administrador de inventario actualiza el sistema.
2. Durante la venta, el sistema verifica el stock en tiempo real y ajusta la cantidad disponible.
3. Si el stock es bajo, el sistema genera una alerta para notificar al encargado del inventario y se sugiere la reposición.

## **3. Procesamiento de pagos y seguimiento de entregas:**

- **Descripción del Proceso:**

El sistema aceptará diferentes métodos de pago y generará recibos digitales, a la vez que mantendrá al cliente informado del estado de su pedido.

**Pasos:**

1. El cliente selecciona el método de pago (tarjeta de crédito, débito, PayPal, etc.).
2. El sistema verifica y confirma el pago.
3. Se genera una factura digital y se envía por correo electrónico al cliente.
4. El pedido se marca como "En preparación" y el cliente puede realizar un seguimiento de la entrega en tiempo real.

#### **Actualización Trimestral:**

- Cada tres meses, el manual debe ser revisado y actualizado para reflejar nuevos procedimientos, cambios en el sistema o mejoras tecnológicas implementadas en la tienda.

#### **Capacitación Continua**

##### **Plan de Capacitación:**

- **Frecuencia:** Capacitación mensual para el personal administrativo, personal de ventas y equipo de atención al cliente.
- **Contenido:**

Uso del sistema para gestionar inventarios.

Cómo procesar y monitorear pedidos.

Herramientas de atención al cliente en línea.

- **Evaluaciones:** Al finalizar cada capacitación, se realizarán evaluaciones para asegurar que el equipo comprenda el funcionamiento del sistema.

#### **Historial de Cambios**

Versión	Fecha	Descripción del cambio	Autor
1.0	23/09/2024	Creación inicial del documento de gestión de inventario.	Equipo de Desarrollo
1.1	25/09/2024	Actualización del proceso de reabastecimiento de stock.	Administrador del sistema

## **Sistema de Control de Versiones**

Para asegurar que el código y la documentación estén correctamente gestionados, se implementará el uso de GitHub para control de versiones.

### **Flujo de Trabajo:**

1. Los desarrolladores crearán ramas (branches) para cada nueva característica o corrección.
2. Los cambios serán revisados antes de ser fusionados en la rama principal (main).
3. Cada versión del software o documento será etiquetada con un número de versión específico.

## **Retroalimentación y Mejora Continua**

### **Sistema de Feedback:**

Se establecerá un sistema de retroalimentación que permita a los usuarios (empleados del sistema) enviar comentarios, sugerencias y reportar errores o dificultades en el uso del módulo de inventario. Esta retroalimentación será revisada mensualmente y las mejoras se implementarán de manera continua para optimizar los procesos.

## **Tabla de Contactos y Responsabilidades**

En caso de que surjan problemas técnicos u operativos durante el uso del sistema de comercio digital, la siguiente tabla proporciona información sobre quién debe ser contactado según el tipo de inconveniente:

Rol/ Responsabilidad	Nombre	Teléfono	Correo Electrónico	Inconvenientes Comunes
Gerente General	Melody Loa Denegri	+51 999 123 456	<a href="mailto:melody.loa@elyelladetalles.com">melody.loa@elyelladetalles.com</a>	Problemas estratégicos o decisiones de alto nivel que impactan el funcionamiento general del negocio.
Encargado de Inventario	Carlos Fernández	+51 988 234 567	<a href="mailto:carlos.fernandez@elyelladetalles.com">carlos.fernandez@elyelladetalles.com</a>	Problemas con la gestión de inventario, como productos agotados que no se reflejan en el sistema o discrepancias de stock.
Encargado de Ventas	Diana Chávez	+51 977 345 678	<a href="mailto:diana.chavez@elyelladetalles.com">diana.chavez@elyelladetalles.com</a>	Problemas relacionados con la confirmación de pedidos, atención al cliente o coordinación de entregas.
Soporte Técnico	Juan Ramírez	+51 966 456 789	<a href="mailto:support@elyelladetalles.com">support@elyelladetalles.com</a>	Fallos técnicos del sistema, problemas con el servidor, base de datos o pasarela de pagos.
Desarrollador de Software	Equipo 01 de estudiantes del Curso Integrador I: Sistemas Software- Sección 19502	+51 944 567 890	<a href="mailto:desarrolladores@elyelladetalles.com">desarrolladores@elyelladetalles.com</a>	Ajustes o modificaciones en el sistema, desarrollo de nuevas funcionalidades o resolución de bugs críticos.
Contabilidad y Finanzas	María Espinoza	+51 933 678 901	<a href="mailto:maria.espinoza@elyelladetalles.com">maria.espinoza@elyelladetalles.com</a>	Problemas con la facturación, pagos o reportes financieros.

### Descripción de Inconvenientes y Acciones Recomendadas:

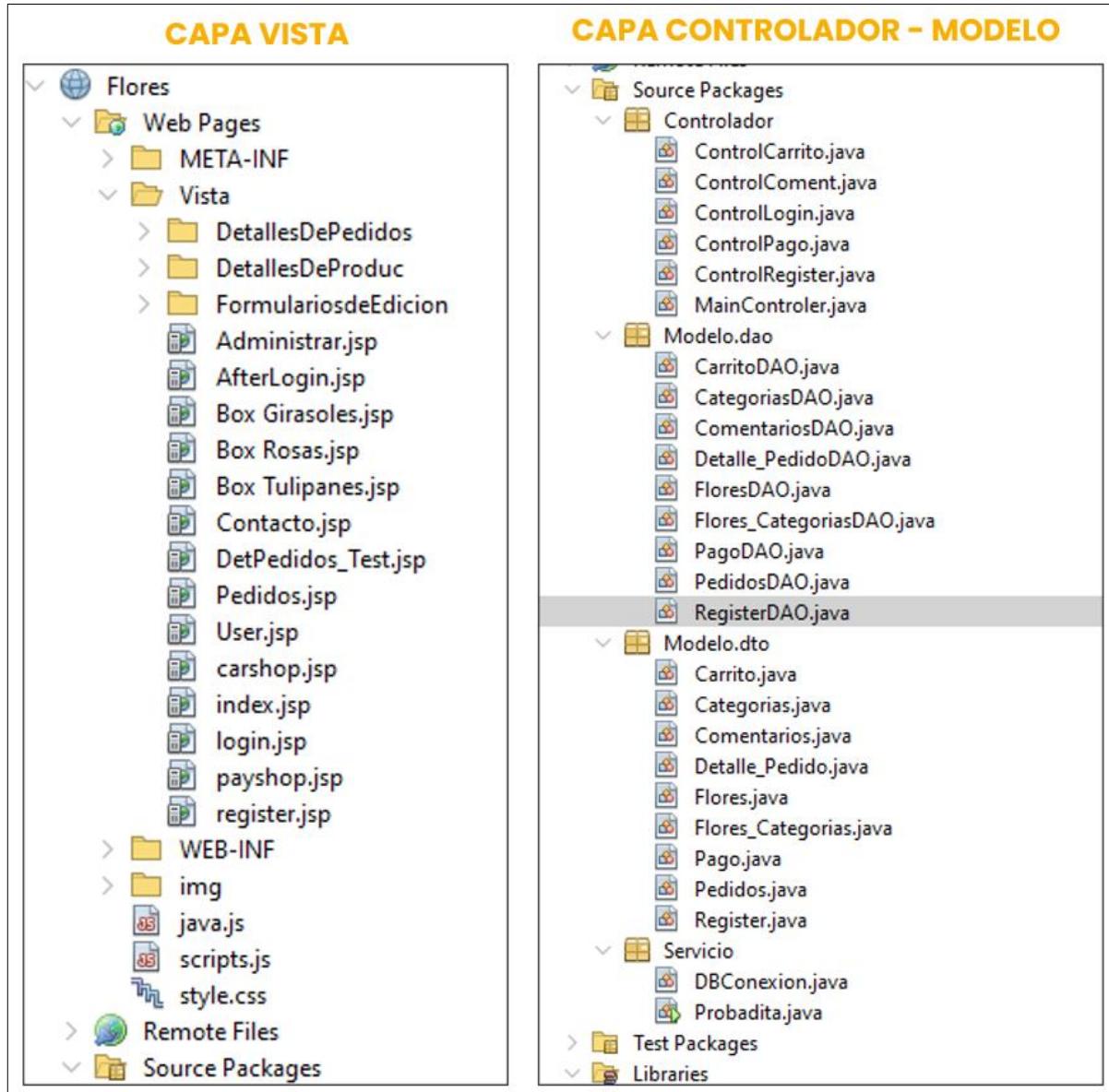
- **Problemas Estratégicos:** Contactar a **Melody Loa Denegri** si es necesario tomar decisiones estratégicas que afecten el negocio, como cambios en el modelo de ventas o expansiones.
- **Gestión de Inventario:** Cualquier error en los niveles de stock o productos que no se actualizan correctamente debe ser resuelto por **Carlos Fernández**.
- **Problemas de Ventas o Atención al Cliente:** Si existen dificultades con pedidos, confirmaciones o atención al cliente, contactar a **Diana Chávez**.
- **Problemas Técnicos:** Errores técnicos en el sistema deben ser dirigidos al equipo de **Soporte Técnico**, y si el problema persiste, se debe escalar al **Desarrollador de Software**.
- **Inconvenientes Financieros:** Errores en facturas o pagos deben ser tratados con **María Espinoza** en el área de contabilidad.

### 13.3 Diagramas de Arquitectura

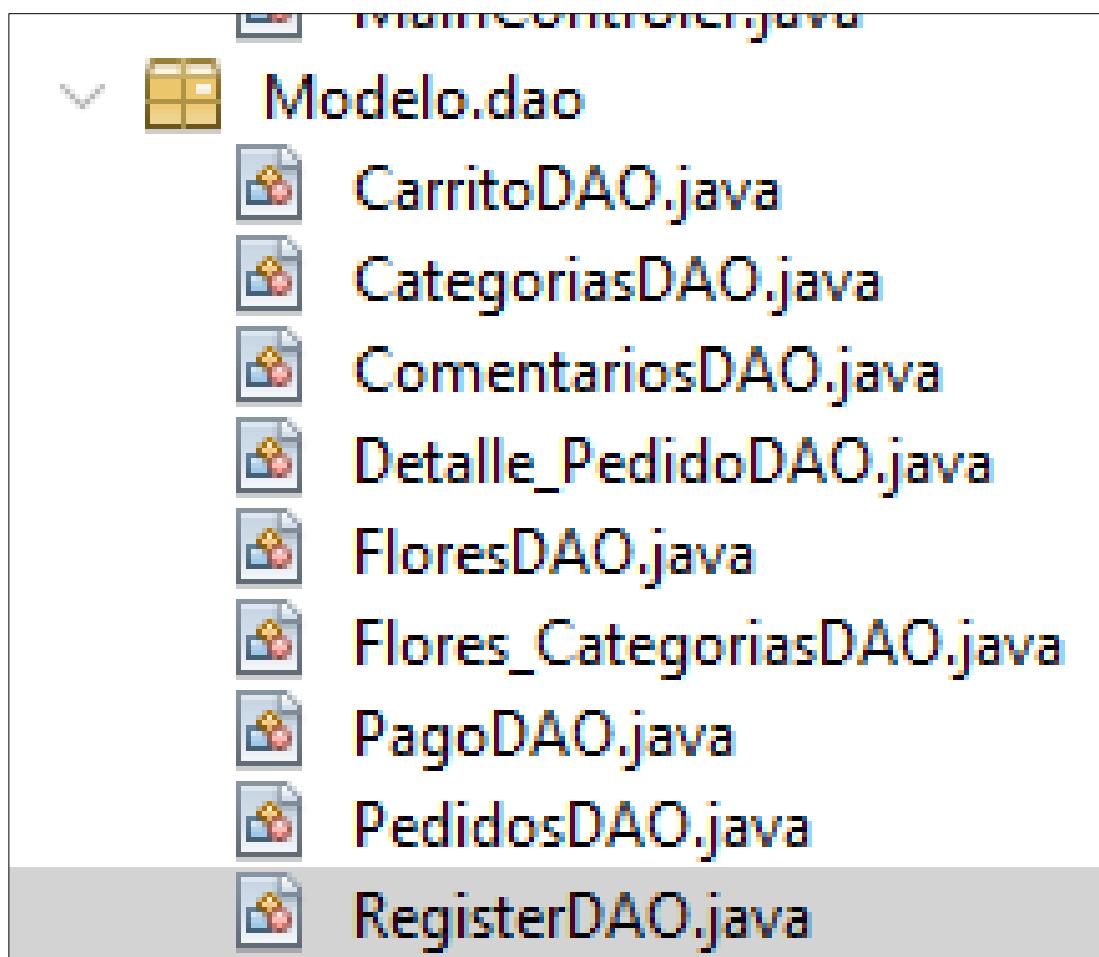
En esta sección se presentan los diagramas que ilustran la arquitectura del proyecto.

Se incluye una representación clara de:

- **Estructura MVC (Modelo-Vista-Controlador):** Diagrama de cómo se organizan y relacionan las capas del proyecto.



- **Integración de Capas DAO:** Diagrama que muestra la interacción entre las capas DAO (Data Access Object) y el resto del sistema.



## 13.4 Documentación de Código

Esta sección contiene la documentación técnica del código:

- 1. Javadoc:** A continuación, se presenta la documentación generada en Javadoc que describe las principales clases, métodos y sus funcionalidades:

# Documentación del Proyecto con Javadoc

## ¿Qué es Javadoc y para qué sirve?

Javadoc es una herramienta incluida en el JDK (Java Development Kit) de Java que se utiliza para generar documentación en formato HTML a partir de comentarios especiales en el código fuente de Java. Estos comentarios describen el comportamiento y la funcionalidad de clases, métodos y atributos, proporcionando una referencia clara y organizada para los desarrolladores.

## ¿Cómo se genera la documentación con Javadoc?

La documentación se genera añadiendo comentarios Javadoc en el código fuente antes de las declaraciones de clases, métodos y atributos. Una vez añadidos estos comentarios, se utiliza una herramienta como NetBeans para generar el HTML de la documentación. En NetBeans, esto se hace seleccionando el proyecto y utilizando la opción "Generate Javadoc..." en el menú contextual.

## Organización de la Documentación Javadoc

### Visión General de los Paquetes

La documentación generada incluye una página principal que muestra todos los paquetes del proyecto, con una breve descripción de cada uno de ellos. Esto proporciona una visión general de la estructura del proyecto y facilita la navegación entre los distintos componentes. Como se observa en la imagen, cada paquete tiene su propia descripción que detalla su propósito en la aplicación.

The screenshot shows a web-based Java Javadoc interface. At the top, there's a navigation bar with links for OVERVIEW, PACKAGE, CLASS, USE, TREE, INDEX, and HELP. On the right side of the header is a search bar labeled 'SEARCH' with a magnifying glass icon and a close button 'X'. Below the header, there's a table titled 'Packages'. The table has two columns: 'Package' and 'Description'. There are four rows in the table, each representing a package: 'Controlador', 'Modelo.dao', 'Modelo.dto', and 'Servicio'. The 'Controlador' row describes classes managing user requests and coordinating operations between the user interface and business logic. The 'Modelo.dao' row describes classes implementing the DAO pattern for data access. The 'Modelo.dto' row describes DTO classes for data transfer between application layers. The 'Servicio' row describes service classes implementing business logic and orchestrating communication between the controller and data access layer.

Package	Description
Controlador	Este paquete contiene las clases del controlador, encargadas de manejar las solicitudes de los usuarios y coordinar las operaciones entre la interfaz de usuario y la lógica de negocio de la aplicación.
Modelo.dao	Este paquete contiene las clases que implementan el patrón DAO (Data Access Object), encargadas de la gestión de acceso a los datos de la aplicación.
Modelo.dto	Este paquete contiene las clases DTO (Data Transfer Object), utilizadas para transferir datos entre las diferentes capas de la aplicación de manera eficiente.
Servicio	Este paquete contiene las clases de servicios de la aplicación, encargadas de implementar la lógica de negocio y orquestar la comunicación entre el controlador y la capa de acceso a datos.

## Detalles de las Clases en un Paquete

Dentro de cada paquete, la documentación incluye una tabla con todas las clases que contiene, acompañada de una breve descripción de cada una. Esta tabla permite identificar rápidamente la funcionalidad de cada clase y entender cómo se relacionan entre sí dentro del paquete. En la imagen correspondiente, se puede ver cómo se presentan estas clases y sus descripciones, lo cual facilita la exploración de la lógica del proyecto.

The screenshot shows a Java package documentation interface. At the top, there are tabs for OVERVIEW, PACKAGE, CLASS, USE, TREE, INDEX, and HELP. The PACKAGE tab is selected. Below the tabs, there are links for PACKAGE DESCRIPTION, RELATED PACKAGES, and CLASSES AND INTERFACES. A search bar is at the top right. The main content area is titled "Package Modelo.dao". It contains a table with a header row "Clases". The table has two columns: "Class" and "Description". The "Class" column lists various DAO classes: CarritoDAO, CategoriasDAO, ComentariosDAO, Detalle\_PedidoDAO, Flores\_CategoriasDAO, FloresDAO, PagoDAO, and PedidosDAO. The "Description" column provides a brief description for each class, such as "La clase CarritoDAO se encarga de gestionar las operaciones de acceso a datos relacionadas con el carrito de compras, como agregar, eliminar y listar productos en el carrito." and "La clase PedidosDAO se encarga de gestionar el acceso a datos para los pedidos de los clientes, permitiendo operaciones como crear, actualizar y consultar pedidos."

## Documentación de Métodos y Atributos

La documentación Javadoc también proporciona detalles más específicos dentro de cada clase, incluyendo la lista de constructores, métodos y atributos. Para cada método, se ofrece una descripción detallada junto con los parámetros que acepta y los valores que devuelve, si corresponde. Este nivel de detalle permite a los desarrolladores comprender el uso adecuado de cada método y atributo de las clases. En la imagen se muestra un ejemplo de cómo se presenta esta información en el HTML generado por Javadoc.

The screenshot shows a Java class documentation interface. At the top, there are tabs for OVERVIEW, PACKAGE, CLASS, USE, TREE, INDEX, and HELP. The CLASS tab is selected. Below the tabs, there are links for SUMMARY, NESTED, FIELD, CONSTR, METHOD, DETAIL, FIELD, CONSTR, and METHOD. A search bar is at the top right. The main content area is titled "Package Modelo.dao". It contains a table with a header row "Constructors". The table has two columns: "Constructor" and "Description". The "Constructor" column lists "RegisterDAO". The "Description" column states "La clase RegisterDAO gestiona las operaciones de acceso a datos para el registro de nuevos usuarios en la aplicación, permitiendo la creación de nuevos registros de usuario y la verificación de datos de registro." Below this, there is a "Method Summary" section. It includes tabs for ALL METHODS, INSTANCE METHODS, and CONCRETE METHODS. The ALL METHODS tab is selected. It shows a table with columns: "Modifier and Type", "Method", and "Description". The methods listed are boolean eliminar(int id), get(int id), Register, and Register(String email, String contrasena). The "Description" column provides a brief description for each method, such as "elimina un usuario por su ID" and "Registra un nuevo usuario en la aplicación".

# Descripción Técnica del Proyecto

## 1. Estructura General del Proyecto

El proyecto está organizado en una arquitectura multicapa que sigue un modelo típico de aplicaciones web en Java. La estructura se divide en las siguientes capas:

- **Vista:** Contiene las páginas JSP que forman la interfaz de usuario.
- **Controlador:** Maneja la lógica de control y la interacción con el usuario.
- **Modelo:** Gestiona la lógica de negocio y el acceso a la base de datos, incluyendo los paquetes DAO y DTO.
- **Servicio:** Proporciona la conexión a la base de datos y otras funcionalidades de soporte.

## 2. Capas del Proyecto

### a. Vista

La carpeta "Web Pages" incluye las páginas JSP que conforman la interfaz de usuario de la aplicación. Estas páginas están organizadas en subcarpetas según la funcionalidad que representan:

- **DetallesDePedidos y DetallesDeProduc:** Páginas relacionadas con la visualización de detalles de pedidos y productos.
- **FormulariosdeEdicion:** Contiene formularios para editar información de productos o pedidos.
- Otras páginas JSP como `Administrar.jsp`, `Pedidos.jsp`, `User.jsp`, etc., manejan la visualización y gestión de diferentes aspectos de la aplicación, como usuarios, pedidos, y productos.

### b. Controlador

El paquete `Controlador` contiene las clases Java que manejan la lógica de control de la aplicación. Cada clase de esta capa actúa como un puente entre la vista y la capa de modelo:

- **ControlCarrito.java:** Gestiona las solicitudes relacionadas con el carrito de compras, como agregar o eliminar productos.
- **ControlLogin.java:** Maneja las solicitudes de inicio de sesión y autenticación de usuarios.
- **MainController.java:** Coordina la navegación general de la aplicación y la integración con otros controladores.
- Otras clases, como `ControlPago.java` y `ControlDetallePedido.java`, manejan aspectos específicos como pagos y detalles de pedidos.

### c. Modelo

El paquete `Modelo` está dividido en dos subpaquetes:

1. **Modelo.dao:** Implementa el patrón DAO (Data Access Object) para interactuar con la base de datos. Cada clase DAO se encarga de una entidad específica, como `CarritoDAO`, `CategoriasDAO`, `PedidosDAO`, etc., manejando la creación, lectura, actualización y eliminación de registros en la base de datos.
2. **Modelo.dto:** Contiene clases DTO (Data Transfer Object) que representan las entidades de la base de datos. Estas clases se utilizan para transferir datos entre la capa de modelo y la capa de controlador, facilitando la comunicación y el intercambio de información.

### d. Servicio

El paquete `Servicio` contiene clases que proporcionan funcionalidades de soporte para la aplicación:

- **DBConexion.java:** Administra la conexión a la base de datos, asegurando una comunicación estable y segura.
- **Probadita.java:** Parece estar destinada a pruebas y validación de funciones específicas de la aplicación.

## 3. Bibliotecas y Dependencias

El proyecto utiliza diversas bibliotecas para el soporte de funcionalidades adicionales, tales como:

- **JSTL:** Para el manejo de etiquetas en JSP y la presentación de datos.
- **MySQL Connector:** Para la conexión con la base de datos MySQL.
- **Logback y JUnit:** Para el registro de eventos y pruebas unitarias, respectivamente.
- **Apache POI:** Posiblemente para la manipulación de archivos Excel, lo cual es común en aplicaciones de gestión.

## 4. Flujo de Trabajo

El flujo de trabajo en la aplicación sigue un patrón MVC (Modelo-Vista-Controlador), donde:

1. El usuario interactúa con la interfaz web (Vista).
2. Las solicitudes se envían al controlador correspondiente (Controlador), que procesa la lógica de la solicitud y solicita datos al modelo si es necesario.
3. El modelo realiza las operaciones de acceso a datos (Modelo.dao) o de transferencia de datos (Modelo.dto).
4. Finalmente, el controlador devuelve una respuesta a la vista, mostrando los resultados al usuario.