



---

**Universidad de Valladolid**

Escuela de Ingeniería Informática

**TRABAJO FIN DE GRADO**

**Grado en Ingeniería Informática  
(Mención Tecnologías de la Información)**

**SISTEMA SOFTWARE DE GESTIÓN  
COMERCIAL CON MÓDULO TPV**

**Director:** Miguel Ángel Laguna Serrano

**Autor:** Alejandro Mediavilla Benito



## **Resumen**

El Trabajo Fin de Grado que se presenta en este documento supone el fin de un proceso de aprendizaje de los distintos aspectos, disciplinas y metodologías que constituyen el ámbito de conocimiento de la Ingeniería Informática.

El objetivo del siguiente proyecto es el desarrollo de un sistema software para el control y gestión de un comercio.

El sistema contará con una aplicación de gestión, desde la que el administrador podrá gestionar todos los elementos del sistema (productos, categorías, proveedores, clientes, etc), y una aplicación de venta TPV, desde la que se podrán realizar las ventas.

En el siguiente documento se detallan las diferentes fases llevadas a cabo para la realización del proyecto (análisis, diseño, implementación, etc), así como un manual de uso para el usuario.





# Índice general

Resumen .....	3
1. Introducción .....	16
1.1 Descripción del proyecto .....	16
1.2 Tecnologías a utilizar para el desarrollo del proyecto .....	17
1.2.1 JavaFX .....	17
1.2.2 Spring .....	17
1.2.4 JUnit .....	19
1.2.5 Mockito .....	19
1.2.6 Maven .....	19
1.2.6 Git .....	20
1.2.6 Balsamiq Mockups .....	20
1.2.7 SonarQube .....	20
1.3 Estructura de la memoria .....	21
2. Gestión del proyecto .....	24
2.1 Proceso Unificado .....	24
2.2 Test Driven Development (TDD) .....	25
2.3 Planificación Inicial .....	26
2.4 Análisis de Riesgos .....	27
2.4.1 Tabla de riesgos .....	28
2.5 Recursos necesarios .....	29
2.5.1 Recursos Humanos .....	29
2.5.2 Recursos Hardware .....	30
2.5.3 Recursos Software .....	30
2.6 Desarrollo efectivo .....	31
3. Análisis .....	34
3.1 Descripción general del sistema .....	34
3.1.1 Visión global del sistema .....	34
3.1.2 Características de los usuarios .....	35
3.2 Requisitos .....	35

3.2.1 Requisitos funcionales.....	35
3.2.2 Requisitos no funcionales.....	39
3.3 Casos de Uso.....	40
3.3.1 Actores.....	40
3.3.2 Diagrama de Casos de Uso.....	41
3.4 Modelo de dominio.....	41
3.5 Descripción detallada de Casos de Uso.....	42
3.5.1 UC-001: Acceder al sistema.....	42
3.5.2 UC-002: Realizar venta.....	43
3.5.3 UC-003: Consultar productos.....	43
3.5.4 UC-004: Buscar productos.....	44
3.5.5 UC-005: Añadir producto.....	45
3.5.6 UC-006: Modificar producto.....	46
3.5.7 UC-007: Eliminar producto.....	46
3.5.8 UC-008: Consultar categorías.....	47
3.5.9 UC-009: Añadir categoría.....	48
3.5.10 UC-010: Modificar categoría.....	49
3.5.11 UC-011: Eliminar categoría.....	49
3.5.12 UC-012: Consultar clientes.....	50
3.5.13 UC-013: Buscar clientes.....	51
3.5.14 UC-014: Añadir cliente.....	52
3.5.15 UC-015: Modificar cliente.....	52
3.5.16 UC-016: Eliminar cliente.....	53
3.5.17 UC-017: Consultar proveedores.....	54
3.5.18 UC-018: Buscar proveedores.....	55
3.5.19 UC-019: Añadir proveedor.....	55
3.5.20 UC-020: Modificar proveedor.....	56
3.5.21 UC-021: Eliminar proveedor.....	57
3.5.22 UC-022: Consultar empleados.....	58
3.5.23 UC-023: Buscar empleados.....	58
3.5.24 UC-024: Añadir empleado.....	59
3.5.25 UC-025: Modificar empleado.....	60

3.5.26 UC-026: Eliminar empleado.....	61
3.5.27 UC-027: Consultar usuarios .....	62
3.5.28 UC-028: Añadir usuario.....	62
3.5.29 UC-029: Modificar usuario .....	63
3.5.30 UC-030: Eliminar usuario .....	64
4. Diseño.....	67
4.1 Patrones de diseño utilizados.....	67
4.1.1 Patrón arquitectónico de capas .....	67
4.1.2 Patrón MVC.....	68
4.1.3 Patrón DAO .....	68
4.2 Descomposición en subsistemas.....	69
4.3 Módulo tfg-model.....	70
4.3.1 Diagrama de clases .....	70
4.4 Módulo tfg-backoffice .....	70
4.4.1 Diagrama de clases .....	70
4.4.2 Capa de vista.....	71
4.4.3 Capa de servicios .....	71
4.4.4 Capa de persistencia .....	71
4.5 Módulo tfg-tpv.....	72
4.5.1 Diagrama de clases .....	72
4.5.2 Capa de vista.....	73
4.5.3 Capa de servicios .....	73
4.5.4 Capa de persistencia .....	73
4.6 Interfaz de usuario .....	74
4.6.1 Pantalla para añadir nuevo elemento .....	74
4.6.2 Pantalla para listado de elementos.....	75
4.6.3 Pantalla para búsqueda de elementos .....	75
4.6.2 Pantalla para eliminar elemento .....	76
4.6.2 Pantalla para editar elemento.....	76
4.5.4 Pantalla de acceso a la aplicación.....	77
4.5.4 Pantalla de venta TPV .....	77

4.7 Modelo de datos.....	78
4.8 Diagramas de secuencia.....	78
4.8.1 SD-001: Añadir producto.....	79
4.8.2 SD-002: Consultar productos .....	79
4.8.3 SD-003: Eliminar Producto .....	80
4.8.4 SD-004: Buscar productos.....	80
4.8.5 SD-005: Editar Producto .....	81
4.8.6 SD-006: Añadir categoría.....	82
4.8.7 SD-007: Consultar categorías.....	82
4.8.8 SD-008: Eliminar categoría .....	83
4.8.9 SD-009: Editar categoría .....	84
4.8.10 SD-010: Añadir cliente .....	85
4.8.11 SD-011: Consultar clientes .....	85
4.8.12 SD-012: Eliminar cliente.....	86
4.8.13 SD-013: Buscar clientes .....	86
4.8.14 SD-014: Editar cliente .....	87
4.8.15 SD-015: Añadir proveedor.....	88
4.8.16 SD-016: Consultar proveedores.....	88
4.8.17 SD-017: Eliminar proveedor .....	89
4.8.18 SD-018: Buscar proveedores .....	89
4.8.19 SD-019: Editar proveedor.....	90
4.8.20 SD-020: Añadir empleado .....	91
4.8.21 SD-021: Consultar empleados.....	91
4.8.22 SD-022: Eliminar empleado .....	92
4.8.23 SD-023: Buscar empleados .....	92
4.8.24 SD-024: Editar empleado .....	93
4.8.25 SD-025: Añadir usuario .....	94
4.8.26 SD-026: Consultar Usuarios.....	94
4.8.27 SD-027: Eliminar usuario.....	95
4.8.28 SD-028: Buscar ventas .....	95
5. Pruebas.....	99

5.1	Introducción.....	99
5.2	Pruebas unitarias.....	99
5.2.1	Introducción a JUnit 4 .....	99
5.2.2	Metodología.....	100
5.2.3	Estructura de las pruebas .....	101
5.2.4	Pruebas de la capa de persistencia.....	102
5.2.5	Pruebas de la capa de servicios (lógica de negocio).....	103
5.2.6	Listado de pruebas unitarias .....	104
5.3	Pruebas funcionales .....	108
6.	Implementación .....	115
6.1	Diagrama de despliegue.....	115
6.2	Spring MVC.....	116
6.2.1	Introducción .....	116
6.2.2	Anotaciones.....	117
6.2.3	View Resolver.....	117
6.2.4	JSP.....	118
6.2.5	JSTL.....	118
6.2.6	Formularios.....	119
6.2.7	Validadores.....	119
6.3	Spring Security .....	121
6.3.1.	Introducción.....	121
6.3.2.	Configuración .....	121
6.4	MyBatis.....	123
6.4.1	Introducción.....	123
6.4.2	Integración con Spring.....	123
7.	Conclusiones.....	127
7.1	Trabajo futuro .....	127
	Bibliografía.....	131
	Anexo I - Manual de usuario .....	134
	Anexo II - Contenido del CD.....	155



## Índice de figuras

Figura 1 - Módulos de Spring.....	18
Figura 2 - Diagrama de desarrollo dirigido por pruebas.....	25
Figura 3 - Fase de inicio.....	26
Figura 4 - Fase de elaboración.....	26
Figura 5 - Fase de construcción.....	27
Figura 6 - Fase de transición.....	27
Figura 7 - Tabla de riesgos del proyecto.....	29
Figura 8 - Fase de elaboración replanificada.....	31
Figura 9 - Fase de construcción replanificada.....	31
Figura 10 - Fase de transición replanificada.....	31
Figura 11 - Diagrama de arquitectura del sistema.....	34
Figura 12 - Diagrama de Casos de Uso.....	41
Figura 13 - Modelo de dominio.....	41
Figura 14 - Detalle UC-001.....	42
Figura 15 - Detalle UC-002.....	43
Figura 16 - Detalle UC-003.....	44
Figura 17 - Detalle UC-004.....	45
Figura 18 - Detalle UC-005.....	45
Figura 19 - Detalle UC-006.....	46
Figura 20 - Detalle UC-007.....	47
Figura 21 - Detalle UC-008.....	48
Figura 22 - Detalle UC-009.....	48
Figura 23 - Detalle UC-010.....	49
Figura 24 - Detalle UC-011.....	50
Figura 25 - Detalle UC-012.....	51
Figura 26 - Detalle UC-013.....	51
Figura 27 - Detalle UC-014.....	52
Figura 28 - Detalle UC-015.....	53
Figura 29 - Detalle UC-016.....	54
Figura 30 - Detalle UC-017.....	54
Figura 31 - Detalle UC-018.....	55
Figura 32 - Detalle UC-019.....	56

Figura 33 - Detalle UC-020 .....	57
Figura 34 - Detalle UC-021 .....	58
Figura 35 - Detalle UC-022 .....	58
Figura 36 - Detalle UC-023 .....	59
Figura 37 - Detalle UC-024 .....	60
Figura 38 - Detalle UC-025 .....	61
Figura 39 - Detalle UC-026 .....	61
Figura 40 - Detalle UC-027 .....	62
Figura 41 - Detalle UC-028 .....	63
Figura 42 - Detalle UC-029 .....	64
Figura 43 - Detalle UC-030 .....	64
Figura 44 - Diagrama del patron arquitectónico de capas .....	67
Figura 45 - Diagrama de comportamiento Patrón MVC.....	68
Figura 46 - Diagrama del patrón DAO .....	69
Figura 47 - Descomposición en subsistemas del Sistema Software .....	69
Figura 48 - Diagrama de clases del módulo tfg-model.....	70
Figura 49 - Diagrama de clases.....	70
Figura 50 - Capa de vista del módulo de backoffice.....	71
Figura 51 - Capa de servicios del módulo de backoffice.....	71
Figura 52 - Capa de persistencia del módulo de backoffice .....	71
Figura 53 - Diagrama de clases.....	72
Figura 54 - Capa de vista del módulo de TPV.....	73
Figura 55 - Capa de servicios del módulo de TPV .....	73
Figura 56 - Capa de persistencia del módulo de TPV.....	73
Figura 57 - Mockup para añadir nuevo elemento .....	74
Figura 58 - Mockup para listado de elementos.....	75
Figura 59 - Mockup para búsqueda de elementos.....	75
Figura 60 - Mockup para eliminar elemento.....	76
Figura 61 - Mockup para editar elemento.....	76
Figura 62 - Mockup de acceso a la aplicación.....	77
Figura 63 - Mockup de pantalla de venta TPV .....	77
Figura 64 - Esquema relacional del modelo de datos del sistema .....	78
Figura 65 - Diagrama secuencia SD-001 .....	79
Figura 66 - Diagrama secuencia SD-003 .....	79



Figura 67 - Diagrama secuencia SD-002 .....	80
Figura 68 - Diagrama secuencia SD-004 .....	80
Figura 69 - Diagrama secuencia SD-005 .....	81
Figura 70 - Diagrama secuencia SD-006 .....	82
Figura 71 - Diagrama secuencia SD-008 .....	82
Figura 72 - Diagrama secuencia SD-007 .....	83
Figura 73 - Diagrama secuencia SD-009 .....	84
Figura 74 - Diagrama secuencia SD-010 .....	85
Figura 75 - Diagrama secuencia SD-012 .....	85
Figura 76 - Diagrama secuencia SD-011 .....	86
Figura 77 - Diagrama secuencia SD-013 .....	86
Figura 71 - Diagrama secuencia SD-014 .....	87
Figura 79 - Diagrama secuencia SD-015 .....	88
Figura 80 - Diagrama secuencia SD-016 .....	88
Figura 81 - Diagrama secuencia SD-017 .....	89
Figura 82 - Diagrama secuencia SD-018 .....	89
Figura 83 - Diagrama secuencia SD-019 .....	90
Figura 84 - Diagrama secuencia SD-020 .....	91
Figura 85 - Diagrama secuencia SD-022 .....	91
Figura 86 - Diagrama secuencia SD-021 .....	92
Figura 87 - Diagrama secuencia SD-023 .....	92
Figura 80 - Diagrama secuencia SD-023 .....	93
Figura 89 - Diagrama secuencia SD-024 .....	94
Figura 90 - Diagrama secuencia SD-025 .....	94
Figura 91 - Diagrama secuencia SD-026 .....	95
Figura 92 - Diagrama secuencia SD-028 .....	95
Figura 93 - Ejecución fallida de pruebas .....	100
Figura 94 - Ejecución de pruebas con éxito .....	101
Figura 95 - Diagrama de despliegue .....	115
Figura 96 - Diagrama de comunicación entre elementos .....	116
Figura 97 - Diagrama funcionamiento DispatcherServlet .....	117

# **1. Introducción**



# 1. Introducción

Hoy en día son muchas las pequeñas y medianas empresas necesitadas de sistemas de control, que les permitan la realización de ventas, así como su gestión, de una forma rápida y sencilla.

El objetivo de este Trabajo Fin de Grado es dar soporte a un comercio, desarrollando un sistema software, capaz de proporcionar tanto un módulo de punto de venta (TPV) como un módulo de gestión (backoffice).

## 1.1 Descripción del proyecto

El Sistema Software estará formado por dos módulos o aplicaciones:

- **Aplicación TPV** (*Terminal Punto de Venta*) para la realización de las ventas:

Esta aplicación será una aplicación de escritorio, que permitirá llevar a cabo todo el proceso de realización de ventas.

A través de esta aplicación, el usuario podrá proceder con la tramitación de la venta, añadiendo productos a una lista previa o “carrito de compra”.

Una vez realizada la venta, ésta quedará registrada en la base de datos y se podrán consultar a través de la aplicación de gestión.

- **Aplicación Web** para la gestión del sistema (productos, clientes, empleados, ventas...):

A través de esta aplicación el usuario podrá realizar las tareas de mantenimiento o gestión de las diferentes entidades de negocio de la aplicación (productos, clientes, proveedores, familias de productos, empleados, etc).

Entre estas tareas destacan la gestión de usuarios del sistema: altas, bajas y modificaciones de todos aquellos usuarios que van a tener permisos para el manejo de la aplicación (empleados del comercio).

De manera similar al caso de los empleados, también se permitirá gestionar los datos de los clientes, permitiendo dar de alta un nuevo cliente, modificar sus datos o borrarlos de la base de datos del sistema.

Por otro lado, el usuario podrá gestionar todo el catálogo de productos, así como crear, modificar y eliminar las familias que categorizan los productos que se añaden a la base de datos.

Por último, la aplicación permitirá la consulta de ventas realizadas, pudiéndose aplicar distintos filtros (día, mes, periodo, empleado, producto, etc).

## 1.2 Tecnologías a utilizar para el desarrollo del proyecto

Los lenguajes de programación utilizados serán:

- *Java* para el desarrollo.
- *Groovy* para la creación de tests unitarios.

Para el desarrollo de la aplicación de escritorio la tecnología utilizada será *JavaFX*.

Por su parte, para el desarrollo de la aplicación web, se utilizará *Spring Framework*.

El desarrollo del proyecto será un desarrollo guiado por pruebas (*Test Driven Development*).

De esta forma se consigue generar un código limpio que funcione correctamente, traduciendo los requisitos a pruebas unitarias, de modo que cuando las pruebas pasen, se puede garantizar que el software funciona correctamente, y que cumple los requisitos que se han establecido.

Para la ejecución de las pruebas unitarias se utilizará *JUnit*.

Para la administración y gestión de dependencias se utilizará *Maven*.

Se utilizará *Git* como herramienta para el control de versiones del proyecto.

Durante la fase de construcción del proyecto se utilizará la plataforma *SonarQube*, para el análisis de la calidad del código fuente que vayamos generando. De esta forma se conseguirá llegar a un producto final de mayor calidad, y menos expuesto a posibles bugs y errores.

### 1.2.1 JavaFX



JavaFX es una familia de productos y tecnologías de Oracle Corporation, para la creación de aplicaciones web que tienen las características y capacidades de aplicaciones de escritorio.

Las aplicaciones JavaFX pueden ser ejecutadas en una amplia variedad de dispositivos.

### 1.2.2 Spring



Spring es una plataforma Java que provee una infraestructura que sirve de soporte para desarrollar aplicaciones empresariales Java.

Entre los servicios que proporciona cabe destacar la autenticación, la autorización y la gestión de pruebas. Su característica principal es la inyección de dependencias a través de la inversión de control.

Spring está compuesto por varios módulos, sin embargo solo es necesario utilizar los que nos sirvan para la aplicación que estemos desarrollando.

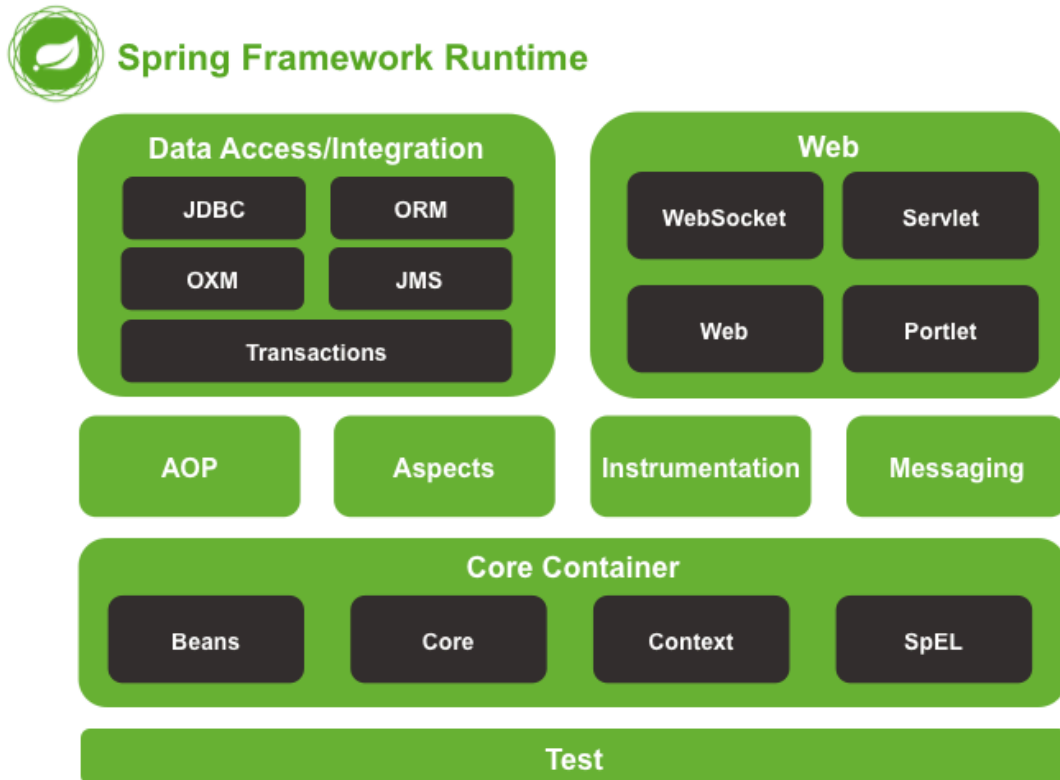


Figura 1 - Módulos de Spring

### **Contenedor de Inversión de Control (inyección de dependencia)**

El contenedor de inversión de control proporciona una forma consistente de configuración y administración de objetos Java usando la reflexión.

Se encarga de gestionar los ciclos de vida de objetos de los objetos específicos: la creación de estos objetos, llamando a sus métodos de inicialización, y configurando estos objetos.

Los objetos creados por el contenedor también se denominan objetos gestionados o beans.

Se puede configurar mediante la carga de archivos XML o la detección de anotaciones Java específicas sobre la configuración de las clases. Estas fuentes de datos contienen las definiciones que proporcionan la información necesaria para la creación de las beans.

## 1.2.4 JUnit



Se trata de un framework que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta tal y como se espera.

El concepto fundamental es el caso de prueba (*test case*), y la suite de prueba (*test suite*).

Los casos de prueba son clases o módulos que disponen de métodos para probar los métodos de una clase o módulo concreta/o. Así, para cada clase que quisiéramos probar definiríamos su correspondiente clase de caso de prueba.

Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente.

## 1.2.5 Mockito



Framework Java que permite simular objetos y configurar su comportamiento y estado deseado para realizar pruebas de caja blanca en aislamiento.

Mediante Mockito podemos aislar diversos componentes de nuestra aplicación para llevar a cabo las pruebas unitarias sin depender del funcionamiento de otras partes o componentes de dicha aplicación. Para verificar que el comportamiento de los objetos simulados ha sido el correcto, podremos especificar una serie de expectativas o resultados a obtener, o simplemente comprobar que no ha producido error alguno en el caso de prueba.

## 1.2.6 Maven



Maven es una herramienta de software para la gestión y construcción de proyectos.

Aparece ante la necesidad de modelar el concepto de "proyecto" y artefacto en forma estándar independientemente del IDE de desarrollo.

Se utiliza como herramienta de SCM (Software Configuration Management) para gestionar:

- El versionado de nuestros proyectos.
- Las dependencias con proyectos nuestros y librerías de terceros.
- La automatización de tareas relativas al ambiente de desarrollo y construcción del artefacto (compilación, generación de código, empaquetado, etc) .

## 1.2.6 Git



Se trata de una herramienta de control de versiones, cuyo principal objetivo es controlar los cambios en el desarrollo de cualquier tipo de software, permitiendo conocer el estado actual de un proyecto, los cambios que se le han realizado a cualquiera de sus piezas, las personas que intervinieron en ellos, etc.

Algunas de las ventajas que permite el uso de control de versiones son:

- Comparación del código de un archivo, pudiendo ver las diferencias entre versiones.
- Restauración de versiones antiguas.
- Fusión de cambios entre distintas versiones.
- Posibilidad de trabajar con distintas ramas de un mismo proyecto.

## 1.2.6 Balsamiq Mockups



Prototipado y creación de interfaces gráficas de usuario. Mediante una herramienta de prototipado o mocking, podemos llevar a cabo el diseño de la interfaz gráfica de nuestras aplicaciones. De esta manera podemos tener una visión aproximada del producto final que queremos lograr, lo cuál nos aportará una visión de objetivos y de posibles fallos de diseño antes de llevar a cabo el desarrollo.

Balsamiq Mockups nos facilita ese proceso de prototipado y agiliza la creación de bocetos. Su gran variedad de elementos a añadir, y su flexibilidad a la hora de personalizarlos y moldearlos al estilo de tu aplicación, hace que sea todo más sencillo y permite un diseño muy detallado de los mockups.

## 1.2.7 SonarQube



SonarQube es una plataforma de código abierto para el análisis de la calidad de código.

Esta plataforma recibe el código fuente como entrada de datos.

En base a esa entrada de datos, comienza a aplicar reglas predefinidas y a controlar si se cumplen. Como salida de datos de análisis, se brinda mucha información útil y propuestas de mejoras.

Surge ante la importancia de mantener un cierto nivel de calidad y legibilidad de código para un desarrollo exitoso en el ambiente de desarrollo dinámico actual, en el que múltiples equipos trabajan en el mismo código, y se realizan cambios con frecuencia.

Estos ambientes requieren seguir ciertas convenciones de código para lograr que el código sea entendible para todos los involucrados en el proceso.



### 1.3 Estructura de la memoria

A continuación, se proporciona un pequeño esquema de los temas principales que van a ser tratados en los siguientes capítulos, así como una breve descripción de cada uno de ellos.

- **Introducción.** Es el punto de partida de cualquier proyecto. Antes de iniciar el proyecto es importante realizar una explicación sobre el trabajo que va a realizarse, así como de analizar las diferentes tecnologías que van a emplearse para su desarrollo.
- **Gestión del proyecto.** Desglose del tiempo y esfuerzo que se estiman necesarios para llevar a cabo cada una de las fases de las que se compone el proyecto, así como la pertinente estimación de recursos.
- **Análisis.** Detalle de los aspectos correspondientes a la fase de análisis de la Ingeniería del Software.
- **Diseño.** Detalle de los aspectos correspondientes a la fase de diseño de la Ingeniería del Software.
- **Pruebas.** Detalle de los aspectos correspondientes a la fase de pruebas de la Ingeniería del Software.
- **Implementación.** Detalle de los aspectos correspondientes a la fase de implementación de la Ingeniería del Software.
- **Conclusiones.** Apartado dedicado a las conclusiones obtenidas a lo largo del proyecto.
- **Bibliografía.** Fuentes, tanto físicas como digitales, consultadas a lo largo de la realización del proyecto.
- **Anexos.** Documentación relacionada o que pueda aportar información para el proyecto .

## **2. Gestión del proyecto**



## 2. Gestión del proyecto

A continuación, se realiza el correspondiente desglose de tiempo y esfuerzo necesario para llevar a cabo satisfactoriamente este proyecto.

En la estimación de costes y tiempos de este proyecto se ha de tener en cuenta que el alumno es un estudiante del Grado en Ingeniería Informática, por lo tanto, no existirá un coste ya que no va a recibir retribución económica alguna.

### 2.1 Proceso Unificado

El Proceso Unificado es una metodología de desarrollo de software, caracterizada por estar dirigida por los casos de uso, centrada en la arquitectura, y tratarse de un proceso iterativo e incremental.

Las fases del Proceso Unificado son las siguientes:

- **Fase de inicio:** durante esta fase se establece el ámbito y límites del proyecto. Se localizan los casos de uso críticos y se realiza la planificación temporal del proyecto.
- **Fase de elaboración:** esta fase tiene como objetivo analizar el dominio del proyecto y establecer una arquitectura base sólida. Es la parte más crítica del proceso, ya que a partir de ella la arquitectura, los requisitos y los planes de desarrollo son estables.
- **Fase de construcción:** durante esta fase se lleva a cabo la implementación de los casos de uso descritos durante la fase de elaboración.
- **Fase de transición:** se trata de la última fase del proceso, y tiene como objetivo conseguir la aceptación por el usuario de que lo entregado es completo.

Cada una de las fases descritas anteriormente se dividen en iteraciones.

Las iteraciones pueden considerarse como pequeños proyectos dentro del proyecto general.

Su cometido es la evolución incremental del desarrollo de la fase de la que forman parte.

## 2.2 Test Driven Development (TDD)

TDD o Test-Driven Development (desarrollo dirigido por pruebas) es una práctica de programación que consiste en escribir primero las pruebas (generalmente unitarias), después escribir el código fuente que pase la prueba satisfactoriamente y, por último, refactorizar el código escrito.

Con esta práctica se consigue entre otras cosas: un código más robusto, más seguro, más mantenible y una mayor rapidez en el desarrollo.

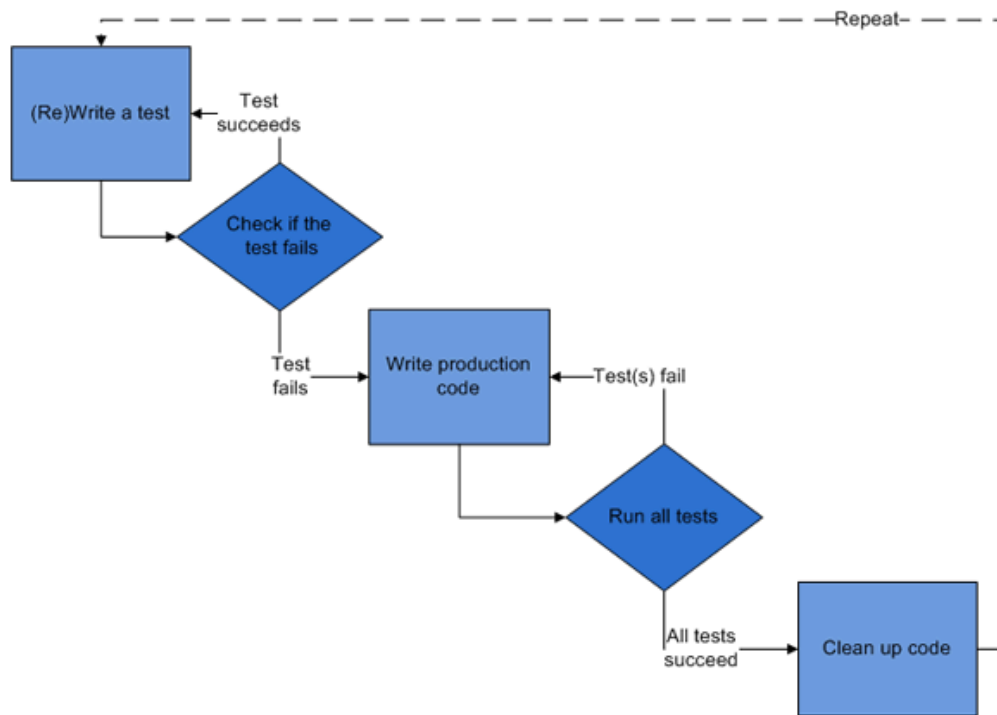


Figura 2 - Diagrama de desarrollo dirigido por pruebas

El propósito del desarrollo guiado por pruebas es lograr un código limpio que funcione.

La idea es que los requisitos sean traducidos a pruebas, de este modo, cuando las pruebas pasen se garantizará que el software cumple con los requisitos que se han establecido.

En este proyecto se combinará el proceso unificado con el desarrollo guiado por pruebas, de forma que a partir de los diagramas de secuencia puedan determinarse las pruebas a realizar.

## 2.3 Planificación Inicial

En la planificación inicial del proyecto se distinguen tres etapas:

1. Fase de Inicio: primera etapa, consistente en la investigación sobre las tecnologías a emplear, así como la planificación y gestión, tanto de riesgos como de recursos.

▼ ● Inicio	21/09/17	28/09/17
● Tecnologías a utilizar	21/09/17	22/09/17
● Planificación temporal	23/09/17	24/09/17
● Gestión de riesgos	25/09/17	26/09/17
● Gestión de recursos	27/09/17	28/09/17

Figura 3 - Fase de inicio

2. Fase de Elaboración: durante esta etapa se llevan a cabo el análisis y diseño del sistema. Se trata de una fase muy importante, ya que es base para la implementación del sistema.

▼ ● Elaboración	29/09/17	31/10/17
▼ ● Análisis del sistema	29/09/17	12/10/17
● Requisitos	29/09/17	3/10/17
● Casos de uso	4/10/17	7/10/17
● Clases del modelo de dominio	8/10/17	12/10/17
▼ ● Diseño del sistema	13/10/17	31/10/17
● Diagrama de clases	13/10/17	22/10/17
● Patrones de diseño	13/10/17	22/10/17
● Diagramas de secuencia	23/10/17	28/10/17
● Modelo de datos	29/10/17	31/10/17

Figura 4 - Fase de elaboración

3. Fase de Construcción: durante esta etapa se lleva a cabo la implementación de la aplicación. Se divide en dos iteraciones, una para cada uno de los módulos.

▼ ● Construcción	1/11/17	20/12/17
▼ ● Iteración 1	1/11/17	30/11/17
● Implementación de la aplicación web	1/11/17	30/11/17
● Pruebas	1/11/17	30/11/17
▼ ● Iteración 2	1/12/17	20/12/17
● Implementación del módulo TPV	1/12/17	20/12/17
● Pruebas	1/12/17	20/12/17

Figura 5 - Fase de construcción

4. Fase de Transición: en esta última etapa se finaliza con la elaboración de las baterías de pruebas, así como la revisión de las fases anteriores.

▼ ● Transición	21/12/17	29/12/17
● Revisión de fases anteriores	21/12/17	25/12/17
● Revisión de documentación	25/12/17	29/12/17

Figura 6 - Fase de transición

## 2.4 Análisis de Riesgos

En este apartado se describen los principales riesgos que se consideran a tener en cuenta, así como una descripción detallada y el estudio de su probabilidad, impacto y el plan de actuación a seguir en caso de que se produzcan.

Para llevar a cabo el pertinente análisis de riesgos, se ha confeccionado una tabla con los aspectos a considerar en cada uno de los casos.

- Problema.
- Probabilidad de ocurrencia.
- Impacto.
- Plan de actuación.

## 2.4.1 Tabla de riesgos

Problema	Descripción	Probabilidad	Impacto	Plan de Actuación
<b>Estimación temporal errónea</b>	Estimación demasiado optimista, lo que imposibilita el cumplimiento de los plazos fijados inicialmente.	Media	Alto	Replanificación del proyecto
	<u>Punto de comprobación:</u>		A lo largo del desarrollo	
<b>Retraso en el proyecto</b>	Retraso debido a la imposibilidad de trabajar en el proyecto por motivos laborales, educativos, o personales	Media	Muy Alto	Replanificación del proyecto
	<u>Punto de comprobación:</u>		A lo largo del desarrollo	
<b>Indisponibilidad de recursos necesarios</b>	No se cuenta con los recursos hardware necesarios para el correcto desarrollo del proyecto, por pérdida o avería.	Baja	Muy Alto	Reposición de recursos a la mayor brevedad
	<u>Punto de comprobación:</u>		Al inicio de cada fase	
<b>La fecha de entrega planteada está muy ajustada</b>	El plazo de presentación podría haber sido mal planificado, dando lugar a retrasos en la entrega.	Media	Alto	Comprobación periódica y replanificación
	<u>Punto de comprobación:</u>		A lo largo del desarrollo	
<b>El alumno carece de conocimientos suficientes</b>	Debido a la inexperiencia del alumno es posible que se produzca algún fallo durante su desarrollo.	Baja	Muy Alto	Búsqueda intensiva de información
	<u>Punto de comprobación:</u>		Al inicio de cada fase	



Problema	Descripción	Probabilidad	Impacto	Plan de Actuación
<b>El alumno no está familiarizado con la tecnología utilizada</b>	Desconocimiento por parte del alumno desarrollando aplicaciones con las tecnologías elegidas	Baja	Alto	Búsqueda intensiva de información
	<u>Punto de comprobación:</u>		Al inicio de cada fase	
<b>Fallo de alguna de las herramientas usadas</b>	Posibilidad de que las herramientas usadas en el desarrollo del proyecto fallen o surjan inconvenientes con ellas.	Baja	Medio	Elección de herramientas actualizadas y estables
	<u>Punto de comprobación:</u>		A lo largo del desarrollo	

Figura 7 - Tabla de riesgos del proyecto

## 2.5 Recursos necesarios

Para llevar a cabo el desarrollo del proyecto serán necesarios recursos humanos, así como una serie de elementos hardware y software. En las siguientes líneas serán definidos.

### 2.5.1 Recursos Humanos

Los recursos humanos disponibles a lo largo de todo el proceso de desarrollo del proyecto son, el autor del presente documento, Alejandro Mediavilla Benito, alumno del Grado en Ingeniería Informática; y el tutor Miguel Ángel Laguna Serrano, encargado de la supervisión y seguimiento del proyecto.

## 2.5.2 Recursos Hardware

- **Ordenador.** Es una de las herramientas fundamentales del proyecto.

En la siguiente tabla pueden verse las características del sistema utilizado:

Modelo	<i>MacBook Pro (Retina, 13 pulgadas)</i>
Procesador	<i>2,6 GHz Intel Core i5</i>
Memoria RAM	<i>8 GB 1600 MHz DDR3</i>
Sistema Operativo	<i>macOS Sierra 10.12.2</i>

## 2.5.3 Recursos Software

- **Eclipse.** IDE utilizado para el desarrollo del proyecto. Se trata de una plataforma de software compuesto por un conjunto de herramientas de programación de código abierto multiplataforma. Proporciona herramientas para la gestión de espacios de trabajo, escribir, desplegar, ejecutar y depurar aplicaciones.
- **Microsoft Project 2016.** Software de administración de proyectos, que se utilizará para realizar la planificación y el correspondiente seguimiento del proyecto. La licencia de Microsoft Project Professional 2016, que será la versión utilizada, no supondrá un coste adicional para el proyecto, ya se dispone de una licencia gratuita.
- **Astah Professional.** Herramienta de modelado UML. Se dispone de una licencia proporcionada por la Escuela, luego no supondrá coste adicional alguno.
- **Git.** Herramienta de control de versiones utilizada para el desarrollo del proyecto.
- **Apache Maven.** Herramienta de software para la gestión y construcción de proyectos Java. Es una manera fácil y gratuita para la gestión de las dependencias de un proyecto.
- **Apache Tomcat 8.** Se trata de un servidor web de código abierto. Con este servidor se pueden desplegar los WAR generados.
- **Balsamic Mockups.** Prototipado y creación de interfaces gráficas (mockups) de usuario.
- **SonarQube.** Herramienta de código abierto para el análisis de la calidad de código.

## 2.6 Desarrollo efectivo

Por compatibilidad con la jornada laboral, se observa la necesidad de realizar una replanificación del proyecto para las fases de elaboración, construcción y transición:

### Fase de elaboración

▼ ● Elaboración	1/12/17	26/01/18
▼ ● Análisis del sistema	1/12/17	19/12/17
● Requisitos	1/12/17	7/12/17
● Casos de uso	8/12/17	14/12/17
● Clases del modelo de dominio	15/12/17	19/12/17
▼ ● Diseño del sistema	20/12/17	26/01/18
● Diagrama de clases	20/12/17	8/01/18
● Patrones de diseño	20/12/17	8/01/18
● Diagramas de secuencia	9/01/18	23/01/18
● Modelo de datos	24/01/18	26/01/18

Figura 8 - Fase de elaboración replanificada

### Fase de construcción

▼ ● Construcción	27/01/18	26/04/18
▼ ● Iteración 1	27/01/18	27/03/18
● Implementación de la aplicación web	27/01/18	27/03/18
● Pruebas	27/01/18	27/03/18
▼ ● Iteración 2	28/03/18	26/04/18
● Implementación del módulo TPV	28/03/18	26/04/18
● Pruebas	28/03/18	26/04/18

Figura 9 - Fase de construcción replanificada

### Fase de transición

▼ ● Transición	27/04/18	11/05/18
● Revisión de fases anteriores	27/04/18	11/05/18
● Revisión de documentación	27/04/18	11/05/18

Figura 10 - Fase de transición replanificada

## **3. Análisis**



### 3. Análisis

El objetivo del análisis en el desarrollo de software es adquirir una visión clara y bien definida del proyecto que se desea realizar. Uno de los puntos principales es establecer una visión global del sistema, junto con la identificación de todos los requisitos, tanto funcionales como no funcionales.

#### 3.1 Descripción general del sistema

##### 3.1.1 Visión global del sistema

La aplicación desarrollada en este Trabajo Fin de Grado está orientada hacia una arquitectura cliente-servidor, donde toda la información será almacenada en una base de datos.

A continuación se muestra un diagrama de esta arquitectura:

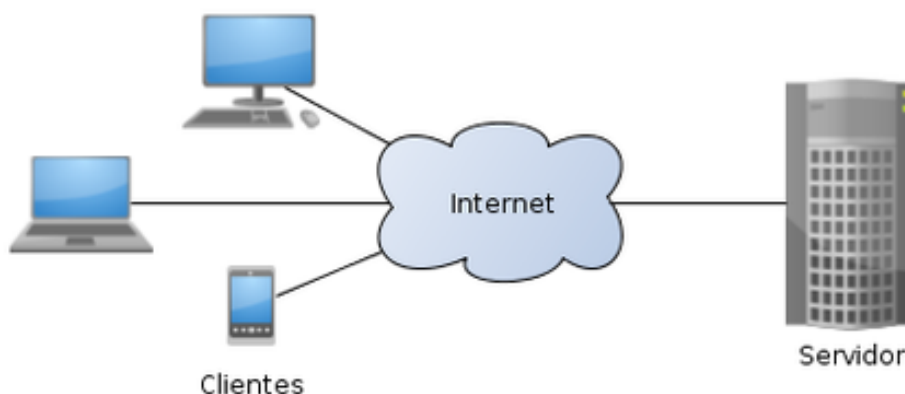


Figura 11 - Diagrama de arquitectura del sistema

El sistema constará de dos módulos: aplicación de escritorio y aplicación web.

En el caso de la aplicación de escritorio, se utilizará una arquitectura cliente-servidor con **cliente pesado**, donde toda la parte de procesamiento y cómputo recaerá en el cliente (en este caso, la aplicación de escritorio), y donde el servidor se limitará a la gestión de transacciones contra la base de datos, es decir, al almacenamiento de la información.

Por su parte, en el caso de la aplicación web, la arquitectura empleada será cliente-servidor con **cliente delgado**, ya que la mayor parte de procesamiento y cómputo va a recaer en el lado del servidor. En este caso el cliente se limitará a recibir los datos del usuario a través de la interfaz gráfica, y enviarlos al servidor para su procesamiento.

### 3.1.2 Características de los usuarios

El nivel de conocimiento de los usuarios de la aplicación deberá ser un nivel intermedio, ya que la interfaz de la aplicación será sencilla e intuitiva para facilitar el uso y la navegación.

Uno de los requisitos no funcionales es la sencillez de uso de la aplicación, por lo que no serán necesarios grandes conocimientos por parte de los usuarios.

## 3.2 Requisitos

### 3.2.1 Requisitos funcionales

Estos requisitos definen la funcionalidad del sistema y de sus componentes.

RF 1	Acceso de usuarios
Descripción	El sistema deberá permitir acceder a los usuarios a través de un login mediante un nombre de usuario y clave.

RF 2	Realizar venta
Descripción	El sistema deberá permitir realizar la tramitación de una venta.

RF 3	Generar ticket
Descripción	El sistema deberá permitir la generación de tickets para cada una de las ventas realizadas.

RF 4	Búsqueda de productos
Descripción	El sistema deberá permitir realizar la búsqueda de productos, aplicando una serie de filtros.

RF 5	Consulta de ventas
Descripción	El sistema deberá permitir consultar las ventas realizadas, pudiendo aplicar una serie de filtros.

RF 6	Añadir nuevo producto
Descripción	El sistema deberá permitir a los administradores dar de alta un nuevo producto en el sistema.

RF 7	Modificar producto existente
<b>Descripción</b>	El sistema deberá permitir a los administradores modificar un producto existente en el sistema.

RF 8	Eliminar producto existente
<b>Descripción</b>	El sistema deberá permitir a los administradores eliminar un producto existente en el sistema.

RF 9	Consulta de productos
<b>Descripción</b>	El sistema deberá permitir a los administradores la consulta y visualización de los productos registrados en el sistema.

RF 10	Validación de datos de producto
<b>Descripción</b>	El sistema deberá validar los datos introducidos en la creación o modificación de un producto.

RF 11	Añadir nueva familia de productos
<b>Descripción</b>	El sistema deberá permitir a los administradores dar de alta una nueva familia de productos en el sistema.

RF 12	Modificar familia de productos existente
<b>Descripción</b>	El sistema deberá permitir a los administradores modificar una familia de productos existente en el sistema.

RF 13	Eliminar familia de productos existente
<b>Descripción</b>	El sistema deberá permitir a los administradores eliminar un producto existente en el sistema.

RF 14	Consulta de familias de productos
<b>Descripción</b>	El sistema deberá permitir a los administradores la consulta y visualización de los productos registrados en el sistema.

RF 15	Validación de datos de familia de producto
<b>Descripción</b>	El sistema deberá validar los datos introducidos en la creación o modificación de una familia de producto.



<b>RF 16</b>	<b>Búsqueda de clientes</b>
<b>Descripción</b>	El sistema deberá permitir realizar la búsqueda de clientes, aplicando una serie de filtros.

<b>RF 17</b>	<b>Añadir nuevo cliente</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores dar de alta un nuevo cliente en el sistema.

<b>RF 18</b>	<b>Modificar cliente existente</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores modificar un cliente existente en el sistema.

<b>RF 19</b>	<b>Eliminar cliente existente</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores eliminar un cliente existente en el sistema.

<b>RF 20</b>	<b>Consulta de clientes</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores la consulta y visualización de los clientes registrados en el sistema.

<b>RF 21</b>	<b>Validación de datos de cliente</b>
<b>Descripción</b>	El sistema deberá validar los datos introducidos en la creación o modificación de un cliente.

<b>RF 22</b>	<b>Búsqueda de proveedores</b>
<b>Descripción</b>	El sistema deberá permitir realizar la búsqueda de proveedores, aplicando una serie de filtros.

<b>RF 23</b>	<b>Añadir nuevo proveedor</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores dar de alta un nuevo proveedor en el sistema.

<b>RF 24</b>	<b>Modificar proveedor existente</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores modificar un proveedor existente en el sistema.

<b>RF 25</b>	<b>Eliminar proveedor existente</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores eliminar un proveedor existente en el sistema.

<b>RF 26</b>	<b>Consulta de proveedores</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores la consulta y visualización de los proveedores registrados en el sistema.

<b>RF 27</b>	<b>Validación de datos de proveedor</b>
<b>Descripción</b>	El sistema deberá validar los datos introducidos en la creación o modificación de un proveedor.

<b>RF 28</b>	<b>Añadir nuevo empleado</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores dar de alta un nuevo empleado en el sistema.

<b>RF 29</b>	<b>Modificar empleado existente</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores modificar un empleado existente en el sistema.

<b>RF 30</b>	<b>Eliminar empleado existente</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores eliminar un empleado existente en el sistema.

<b>RF 31</b>	<b>Consulta de empleados</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores la consulta y visualización de los empleados registrados en el sistema.

<b>RF 32</b>	<b>Validación de datos de empleado</b>
<b>Descripción</b>	El sistema deberá validar los datos introducidos en la creación o modificación de un empleado.

<b>RF 33</b>	<b>Añadir nuevo usuario</b>
<b>Descripción</b>	El sistema deberá permitir a los administradores dar de alta un nuevo usuario en el sistema.

RF 34	Modificar usuario existente
<b>Descripción</b>	El sistema deberá permitir a los administradores modificar un usuario existente en el sistema.

RF 35	Eliminar usuario existente
<b>Descripción</b>	El sistema deberá permitir a los administradores eliminar un usuario existente en el sistema.

RF 36	Consulta de usuarios
<b>Descripción</b>	El sistema deberá permitir a los administradores la consulta y visualización de los usuarios registrados en el sistema.

RF 37	Validación de datos de usuario
<b>Descripción</b>	El sistema deberá validar los datos introducidos en la creación o modificación de un usuario.

RF 38	Mensajes de error
<b>Descripción</b>	El sistema deberá mostrar mensajes explicativos en los casos en que se produzca algún error de validación en los datos.

### 3.2.2 Requisitos no funcionales

Estos requisitos sirven para describir propiedades o cualidades que el sistema debe tener:

RNF 1	Accesibilidad
<b>Descripción</b>	El sistema backoffice debe ser accesible desde cualquier navegador web.

RNF 2	Accesibilidad
<b>Descripción</b>	El sistema de venta debe ser accesible desde una aplicación de escritorio, independiente del sistema operativo.

RNF 3	Autenticación de usuarios
<b>Descripción</b>	El sistema debe disponer de autenticación de usuarios.

RNF 4	Encriptación de contraseñas
<b>Descripción</b>	Las contraseñas serán almacenadas bajo algún tipo de cifrado.

RNF 5	Sistema escalable
<b>Descripción</b>	El sistema debe permitir añadir nuevas funcionalidades de forma sencilla, sin que la funcionalidad existente se vea afectada ni perjudicada.

RNF 6	Interfaz sencilla
<b>Descripción</b>	El sistema debe poseer una interfaz sencilla e intuitiva para los usuarios, con un tiempo de aprendizaje no superior a 2 - 3 días .

RNF 7	Tiempos de respuesta
<b>Descripción</b>	El sistema debe proporcionar unos tiempos de respuesta rápidos, no superiores a 5 segundos para cada funcionalidad.

RNF 8	Disponibilidad
<b>Descripción</b>	El sistema debe estar disponible todos los días del año, 24 horas al día.

### 3.3 Casos de Uso

Los casos de uso del sistema definen las secuencias de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema.

Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas.

#### 3.3.1 Actores

Se trata de cualquier entidad externa al sistema que guarda una relación con éste y que le demanda una funcionalidad.

A continuación se detallan los actores del sistema:

ACT-001	Empleado
<b>Descripción</b>	Empleado del comercio que accederá a la aplicación.
<b>Comentarios</b>	Accederá al sistema para la realización de las ventas.

ACT-002	Administrador
<b>Descripción</b>	Usuario administrador, encargado del mantenimiento del sistema
<b>Comentarios</b>	Encargado de la gestión de las entidades de negocio de la aplicación.

### 3.3.2 Diagrama de Casos de Uso

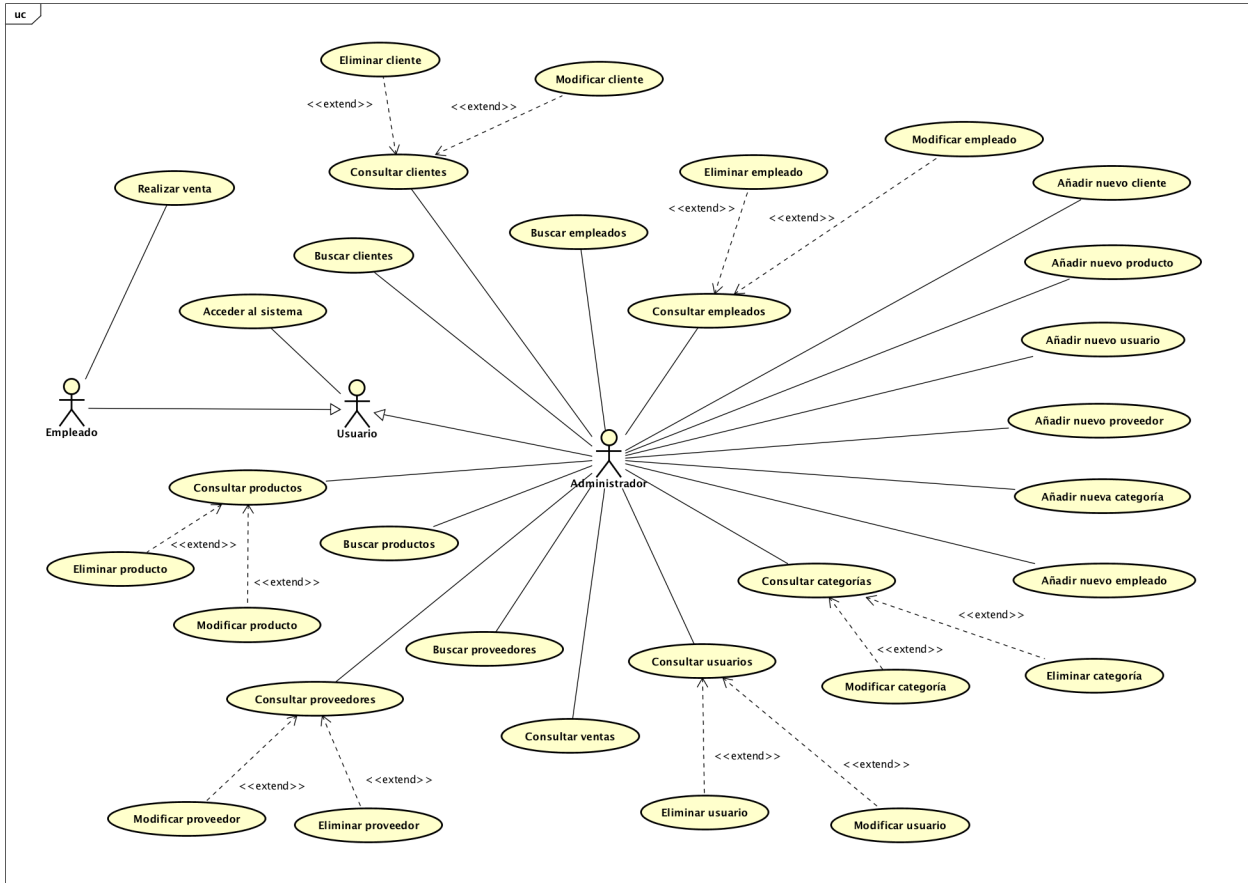


Figura 12 - Diagrama de Casos de Uso

### 3.4 Modelo de dominio

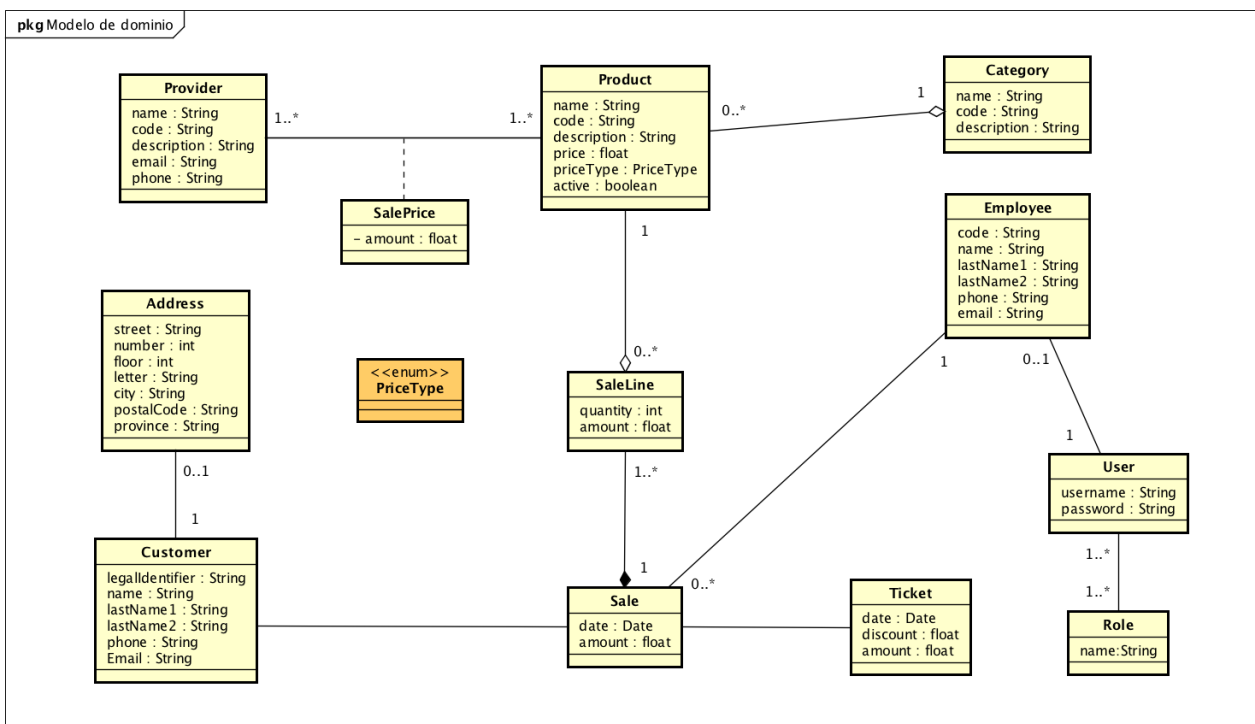


Figura 13 - Modelo de dominio

### 3.5 Descripción detallada de Casos de Uso

A continuación se presentan con detalle los distintos casos de uso del sistema, con toda su información asociada, precondiciones, postcondiciones y demás información importante.

Para evitar tener duplicada la descripción de los casos de uso (casos de uso esenciales y caso de uso detallados), se muestran ya con detalles a nivel de diseño (botones, enlaces, etc).

#### 3.5.1 UC-001: Acceder al sistema

UC-001	Acceder al sistema	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Empleado	
	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar el login de usuario	
<b>Precondición</b>	El usuario dispone de un nombre de usuario	
	El usuario dispone de una clave de acceso	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario inicia la aplicación
	2	El sistema muestra al usuario el formulario para introducir su nombre de usuario y clave
	3	El usuario introduce sus datos y solicita acceder al sistema
	4	El sistema valida los datos del usuario
<b>Postcondición</b>	El usuario ha accedido al sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	4.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario

Figura 14 - Detalle UC-001

### 3.5.2 UC-002: Realizar venta

UC-002	Realizar venta	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Empleado	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar el login de usuario	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario pulsa sobre el producto que desea añadir a la venta
	2	El producto se añade al carrito de venta
	3	Se actualiza el importe de la venta
	4	El usuario solicita realizar venta
	5	Se vacía el carrito de venta
<b>Postcondición</b>	La venta ha sido añadida al sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
	Añadir más productos	Paso 3 - Si el usuario pulsa sobre más productos, se añaden a la venta actual
	Imprimir ticket	Paso 4 - Si el usuario selecciona la opción de imprimir ticket, se imprimirá el ticket de la venta
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	—	—

Figura 15 - Detalle UC-002

### 3.5.3 UC-003: Consultar productos

UC-003	Consultar productos	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la consulta de productos	

<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para mostrar el listado de productos
	2	El sistema muestra al usuario una tabla con el listado de todos los productos del sistema
	3	<u>Punto de extensión</u> (acción sobre un producto)
<b>Postcondición</b>	—	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	2.1	En caso de no existir ningún producto, el sistema muestra un mensaje al usuario

Figura 16 - Detalle UC-003

### 3.5.4 UC-004: Buscar productos

<b>UC-004</b>	<b>Buscar productos</b>	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la búsqueda de productos	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para realizar la búsqueda de productos
	2	El sistema muestra al usuario un formulario con los campos de búsqueda para los productos
	3	El usuario completa uno o varios de los campos de búsqueda y solicita la búsqueda de productos
	4	El sistema muestra al usuario el listado de productos que cumplen los criterios de búsqueda
<b>Postcondición</b>	—	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza



Excepciones	Paso	Acción
	4.1	En caso de no existir ningún producto, el sistema muestra un mensaje al usuario

Figura 17 - Detalle UC-004

### 3.5.5 UC-005: Añadir producto

UC-005	Añadir producto	
Versión	1.0	
Autores	Alejandro Mediavilla	
Actores	Actor Administrador	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera dar de alta un nuevo producto	
Precondición	El usuario ha realizado login en el sistema	
Secuencia normal	Paso	Acción
	1	El usuario selecciona la opción para añadir un nuevo producto en el sistema
	2	El sistema muestra al usuario el formulario para introducir los datos del producto
	3	El usuario completa el formulario introduciendo los datos del producto
	4	El usuario solicita añadir el producto
	5	El sistema valida los datos del producto introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el producto ha sido añadido correctamente
Postcondición	Se ha añadido un nuevo producto al sistema	
Flujo alternativo	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
Excepciones	Paso	Acción
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 18 - Detalle UC-005

### 3.5.6 UC-006: Modificar producto

UC-006	Modificar producto	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quieran modificar los datos de un producto existente ( <u>extiende el caso de uso Consultar Productos</u> )	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de productos	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para modificar un producto
	2	El sistema muestra al usuario la pantalla con el formulario cargado con los datos del producto
	3	El usuario modifica los datos del producto
	4	El usuario solicita guardar el producto
	5	El sistema valida los datos del producto introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el producto ha sido modificado correctamente
<b>Postcondición</b>	Se han modificado los datos del producto en el sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 19 - Detalle UC-006

### 3.5.7 UC-007: Eliminar producto

UC-007	Eliminar producto	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	

<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera eliminar un producto existente ( <u>extiende el caso de uso Consultar Productos</u> )	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de productos	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para eliminar un producto
	2	El sistema muestra al usuario un diálogo de confirmación, indicándole que está a punto de eliminar un producto
	3	El usuario confirma la eliminación del producto
	4	El sistema muestra un mensaje de éxito al usuario, para indicarle que el producto ha sido eliminado correctamente
<b>Postcondición</b>	Se ha eliminado el producto del sistema	
<b>Flujo alternativo</b>	3.1	El usuario solicita cancelar, y el caso de uso finaliza

Figura 20 - Detalle UC-007

### 3.5.8 UC-008: Consultar categorías

UC-008	Consultar categorías	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la consulta de categorías	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para mostrar el listado de categorías
	2	El sistema muestra al usuario una tabla con el listado de todas las categorías de productos
	3	<u>Punto de extensión</u> (acción sobre una categoría)
<b>Postcondición</b>		

<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	2.1	En caso de no existir ninguna categoría, el sistema muestra un mensaje al usuario

Figura 21 - Detalle UC-008

### 3.5.9 UC-009: Añadir categoría

UC-009	Añadir categoría	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera dar de alta una nueva categoría	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para añadir una nueva categoría en el sistema
	2	El sistema muestra al usuario el formulario para introducir los datos de categoría
	3	El usuario completa el formulario introduciendo los datos de la categoría
	4	El usuario solicita añadir la categoría
	5	El sistema valida los datos de la categoría introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que la categoría ha sido añadida correctamente
<b>Postcondición</b>	Se ha añadido una nueva categoría al sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 22 - Detalle UC-009

### 3.5.10 UC-010: Modificar categoría

UC-010	Modificar categoría	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quieran modificar los datos de una categoría existente <u>(extiende el caso de uso Consultar Categorías)</u>	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de categorías	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para modificar una categoría
	2	El sistema muestra al usuario el formulario cargado con los datos de la categoría
	3	El usuario modifica los datos de la categoría
	4	El usuario solicita guardar la categoría
	5	El sistema valida los datos de la categoría introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que la categoría ha sido modificada correctamente
<b>Postcondición</b>	Se han modificado los datos de la categoría en el sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 23 - Detalle UC-010

### 3.5.11 UC-011: Eliminar categoría

UC-011	Eliminar categoría	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	

<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera eliminar una categoría existente ( <u>extiende el caso de uso Consultar Categorías</u> )	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de categorías	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para eliminar una categoría
	2	El sistema muestra al usuario un diálogo de confirmación, indicándole que está a punto de eliminar una categoría
	3	El usuario pulsa sobre el botón de confirmar
	4	El sistema muestra un mensaje de éxito al usuario, para indicarle que la categoría ha sido eliminada correctamente
<b>Postcondición</b>	Se ha eliminado la categoría del sistema	
<b>Flujo alternativo</b>	3.1	El usuario solicita cancelar, y el caso de uso finaliza

Figura 24 - Detalle UC-011

### 3.5.12 UC-012: Consultar clientes

UC-012	Consultar clientes	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la consulta de clientes	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selección la opción para mostrar el listado de clientes
	2	El sistema muestra al usuario una tabla con el listado de todos los clientes del sistema
	3	<u>Punto de extensión</u> (acción sobre un cliente)
<b>Postcondición</b>		

<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	2.1	En caso de no existir ningún cliente, el sistema muestra un mensaje al usuario

Figura 25 - Detalle UC-012

### 3.5.13 UC-013: Buscar clientes

UC-013	Buscar clientes	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la búsqueda de clientes	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para realizar la búsqueda de clientes
	2	El sistema muestra al usuario el formulario con los campos de búsqueda para los clientes
	3	El usuario completa uno o varios de los campos de búsqueda y solicita la búsqueda de clientes
	4	El sistema muestra al usuario el listado de clientes que cumplen los criterios de búsqueda
<b>Postcondición</b>		
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	4.1	En caso de no existir ningún cliente, el sistema muestra un mensaje al usuario

Figura 26 - Detalle UC-013

### 3.5.14 UC-014: Añadir cliente

UC-014	Añadir cliente	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador / Actor Empleado	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera dar de alta un nuevo cliente	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para añadir un nuevo cliente en el sistema
	2	El sistema muestra al usuario el formulario para introducir los datos del cliente
	3	El usuario completa el formulario introduciendo los datos del cliente
	4	El usuario solicita añadir el cliente
	5	El sistema valida los datos del producto introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el cliente ha sido añadido correctamente
<b>Postcondición</b>	Se ha añadido un nuevo cliente al sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 27 - Detalle UC-014

### 3.5.15 UC-015: Modificar cliente

UC-015	Modificar cliente	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	



<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quieran modificar los datos de un cliente existente <u>(extiende el caso de uso Consultar Clientes)</u>	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de clientes	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para modificar un cliente
	2	El sistema muestra al usuario el formulario cargado con los datos del cliente
	3	El usuario modifica los datos del cliente
	4	El usuario solicita guardar el cliente
	5	El sistema valida los datos del cliente introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el cliente ha sido modificado correctamente
<b>Postcondición</b>	Se han modificado los datos del cliente en el sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 28 - Detalle UC-015

### 3.5.16 UC-016: Eliminar cliente

UC-016	Eliminar cliente	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera eliminar un cliente existente <u>(extiende el caso de uso Consultar Clientes)</u>	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de clientes	

<b>Secuencia normal</b>	1	El usuario selecciona la opción para eliminar un cliente
	2	El sistema muestra al usuario un diálogo de confirmación, indicándole que está a punto de eliminar un cliente
	3	El usuario confirma la eliminación del cliente
	4	El sistema muestra un mensaje de éxito al usuario, para indicarle que el cliente ha sido eliminado correctamente
<b>Postcondición</b>	Se ha eliminado el cliente del sistema	
<b>Flujo alternativo</b>	3.1	El usuario solicita cancelar, y el caso de uso finaliza

Figura 29 - Detalle UC-016

### 3.5.17 UC-017: Consultar proveedores

UC-017	Consultar proveedores	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la consulta de proveedores	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para mostrar el listado de proveedores
	2	El sistema muestra al usuario una tabla con el listado de todos los proveedores del sistema
	3	<u>Punto de extensión</u> (acción sobre un proveedor)
<b>Postcondición</b>		
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	2.1	En caso de no existir ningún proveedor, el sistema muestra un mensaje al usuario

Figura 30 - Detalle UC-017

### 3.5.18 UC-018: Buscar proveedores

UC-018	Buscar proveedores	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la búsqueda de proveedores	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para realizar la búsqueda de proveedores
	2	El sistema muestra al usuario un formulario con los campos de búsqueda para los proveedores
	3	El usuario completa uno o varios de los campos de búsqueda y solicita la búsqueda de proveedores
	4	El sistema muestra al usuario el listado de proveedores que cumplen los criterios de búsqueda
<b>Postcondición</b>		
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	4.1	En caso de no existir ningún proveedor, el sistema muestra un mensaje al usuario

Figura 31 - Detalle UC-018

### 3.5.19 UC-019: Añadir proveedor

UC-019	Añadir proveedor	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera dar de alta un nuevo proveedor	

<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para añadir un nuevo proveedor en el sistema
	2	El sistema muestra al usuario el formulario para introducir los datos del proveedor
	3	El usuario completa el formulario introduciendo los datos del proveedor
	4	El usuario solicita añadir el proveedor
	5	El sistema valida los datos del producto introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el proveedor ha sido añadido correctamente
<b>Postcondición</b>	Se ha añadido un nuevo proveedor al sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 32 - Detalle UC-019

### 3.5.20 UC-020: Modificar proveedor

UC-020	Modificar proveedor	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quieran modificar los datos de un proveedor existente <u>(extiende el caso de uso Consultar Proveedores)</u>	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de proveedores	
	1	El usuario selecciona la opción para modificar un proveedor

<b>Secuencia normal</b>	2	El sistema muestra al usuario el formulario cargado con los datos del proveedor
	3	El usuario modifica los datos del proveedor
	4	El usuario solicita guardar el proveedor
	5	El sistema valida los datos del proveedor introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el proveedor ha sido modificado correctamente
<b>Postcondición</b>	Se han modificado los datos del proveedor en el sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 33 - Detalle UC-020

### 3.5.21 UC-021: Eliminar proveedor

UC-021	Eliminar proveedor	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera eliminar un proveedor existente ( <u>extiende el caso de uso Consultar Proveedores</u> )	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de proveedores	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para eliminar un proveedor
	2	El sistema muestra al usuario un diálogo de confirmación, indicándole que está a punto de eliminar un proveedor
	3	El usuario confirma la eliminación del proveedor
	4	El sistema muestra un mensaje de éxito al usuario, para indicarle que el proveedor ha sido eliminado correctamente

<b>Postcondición</b>	Se ha eliminado el proveedor del sistema	
<b>Flujo alternativo</b>	3.1	El usuario solicita cancelar, y el caso de uso finaliza

Figura 34 - Detalle UC-021

### 3.5.22 UC-022: Consultar empleados

UC-022	Consultar empleados	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la consulta de empleados	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para mostrar el listado de empleados
	2	El sistema muestra al usuario una tabla con el listado de todos los empleados del sistema
	3	<u>Punto de extensión</u> (acción sobre un empleado)
<b>Postcondición</b>		
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	2.1	En caso de no existir ningún empleado, el sistema muestra un mensaje al usuario

Figura 35 - Detalle UC-022

### 3.5.23 UC-023: Buscar empleados

UC-023	Buscar empleados	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	

<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la búsqueda de empleados	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para realizar la búsqueda de empleados
	2	El sistema muestra al usuario un formulario con los campos de búsqueda para los empleados
	3	El usuario completa uno o varios de los campos de búsqueda y solicita la búsqueda de empleados
	4	El sistema muestra al usuario el listado de empleados que cumplen los criterios de búsqueda
<b>Postcondición</b>		
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	4.1	En caso de no existir ningún empleado, el sistema muestra un mensaje al usuario

Figura 36 - Detalle UC-023

### 3.5.24 UC-024: Añadir empleado

<b>UC-024</b>	<b>Alta de nuevo empleado</b>	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera dar de alta un nuevo empleado	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para añadir un nuevo empleado en el sistema
	2	El sistema muestra al usuario el formulario para introducir los datos del empleado

<b>Secuencia normal</b>	3	El usuario completa el formulario introduciendo los datos del empleado
	4	El usuario solicita añadir el empleado
	5	El sistema valida los datos del producto introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el empleado ha sido añadido correctamente
<b>Postcondición</b>	Se ha añadido un nuevo empleado al sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 37 - Detalle UC-024

### 3.5.25 UC-025: Modificar empleado

UC-025	Modificar empleado	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quieran modificar los datos de un empleado existente <u>(extiende el caso de uso Consultar Empleados)</u>	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de empleados	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para modificar un empleado
	2	El sistema muestra al usuario el formulario cargado con los datos del empleado
	3	El usuario modifica los datos del empleado
	4	El usuario solicita guardar el empleado
	5	El sistema valida los datos del empleado introducidos en el formulario



	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el empleado ha sido modificado correctamente
<b>Postcondición</b>	Se han modificado los datos del empleado en el sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 38 - Detalle UC-025

### 3.5.26 UC-026: Eliminar empleado

UC-026	Eliminar empleado	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera eliminar un empleado existente ( <u>extiende el caso de uso Consultar Empleados</u> )	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de empleados	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para eliminar un empleado
	2	El sistema muestra al usuario un diálogo de confirmación, indicándole que está a punto de eliminar un empleado
	3	El usuario confirma la eliminación del empleado
	4	El sistema muestra un mensaje de éxito al usuario, para indicarle que el empleado ha sido eliminado correctamente
<b>Postcondición</b>	Se ha eliminado el empleado del sistema	
<b>Flujo alternativo</b>	3.1	El usuario solicita cancelar, y el caso de uso finaliza

Figura 39 - Detalle UC-026

### 3.5.27 UC-027: Consultar usuarios

UC-027	Consultar usuarios	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera realizar la consulta de usuarios	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para mostrar el listado de usuarios
	2	El sistema muestra al usuario una tabla con el listado de todos los usuarios del sistema
	3	<u>Punto de extensión</u> (acción sobre un usuario)
<b>Postcondición</b>		
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita salir de la aplicación, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	2.1	En caso de no existir ningún usuario, el sistema muestra un mensaje al usuario

Figura 40 - Detalle UC-027

### 3.5.28 UC-028: Añadir usuario

UC-028	Añadir usuario	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera dar de alta un nuevo usuario	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	<b>Paso</b>	<b>Acción</b>
	1	El usuario selecciona la opción para añadir un nuevo usuario en el sistema

<b>Secuencia normal</b>	2	El sistema muestra al usuario el formulario para introducir los datos del usuario
	3	El usuario completa el formulario introduciendo los datos del usuario
	4	El usuario solicita añadir el usuario
	5	El sistema valida los datos del producto introducidos en el formulario
	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el usuario ha sido añadido correctamente
<b>Postcondición</b>	Se ha añadido un nuevo usuario al sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 41 - Detalle UC-028

### 3.5.29 UC-029: Modificar usuario

UC-029	Modificar usuario	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quieran modificar los datos de un usuario existente ( <u>extiende el caso de uso Consultar Usuarios</u> )	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de usuarios	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para modificar un usuario
	2	El sistema muestra al usuario el formulario cargado con los datos del usuario
	3	El usuario modifica los datos del usuario
	4	El usuario solicita guardar el usuario
	5	El sistema valida los datos del usuario introducidos en el formulario

	6	El sistema muestra un mensaje de éxito al usuario, para indicarle que el usuario ha sido modificado correctamente
<b>Postcondición</b>	Se han modificado los datos del usuario en el sistema	
<b>Flujo alternativo</b>	Cancelación	En cualquier momento el usuario solicita cancelar, y el caso de uso finaliza
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	5.1	En caso de no ser validos los datos, el sistema muestra un mensaje de error al usuario, indicando los errores existentes en el formulario

Figura 42 - Detalle UC-029

### 3.5.30 UC-030: Eliminar usuario

UC-030	Eliminar usuario	
<b>Versión</b>	1.0	
<b>Autores</b>	Alejandro Mediavilla	
<b>Actores</b>	Actor Administrador	
<b>Descripción</b>	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando se quiera eliminar un usuario existente ( <u>extiende el caso de uso Consultar Usuarios</u> )	
<b>Precondición</b>	El usuario ha realizado login en el sistema	
	El usuario se encuentra en el listado de usuarios	
<b>Secuencia normal</b>	1	El usuario selecciona la opción para eliminar un usuario
	2	El sistema muestra al usuario un diálogo de confirmación, indicándole que está a punto de eliminar un usuario
	3	El usuario confirma la eliminación del usuario
	4	El sistema muestra un mensaje de éxito al usuario, para indicarle que el usuario ha sido eliminado correctamente
<b>Postcondición</b>	Se ha eliminado el usuario del sistema	
<b>Flujo alternativo</b>	3.1	El usuario solicita cancelar, y el caso de uso finaliza

Figura 43 - Detalle UC-030

## **4. Diseño**



## 4. Diseño

### 4.1 Patrones de diseño utilizados

#### 4.1.1 Patrón arquitectónico de capas

El objetivo primordial es la separación (desacoplamiento) de las partes que componen el sistema. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sea necesario algún cambio, solo afectará al nivel requerido, sin necesidad de modificar el resto.

Las capas en que se divide el sistema son las siguientes:

- **Capa de presentación:** presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso. Esta capa se comunica únicamente con la capa de negocio.
- **Capa de lógica de negocio:** se denomina así porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos (persistencia), para solicitar almacenar o recuperar datos de ella.
- **Capa de datos:** es donde residen los datos y desde donde se accede a ellos. Se encarga de realizar todo el almacenamiento de datos, recibir las solicitudes de almacenamiento o recuperar la información solicitada desde la capa de negocio.

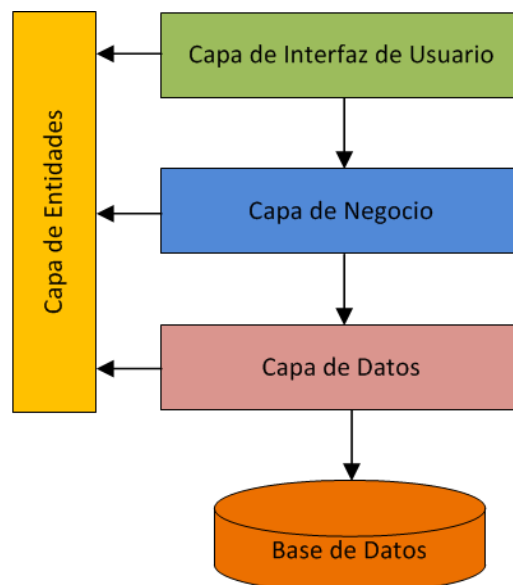


Figura 44 - Diagrama del patrón arquitectónico de capas

### 4.1.2 Patrón MVC

Se trata de un patrón de arquitectura de software, cuyo objetivo es separar los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Los componentes son los siguientes:

- **Modelo**: representación la información con la cual el sistema opera. Las peticiones de acceso o manipulación de información llegan al modelo a través del controlador.
- **Vista**: Presenta el modelo (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario).
- **Controlador**: responde a eventos (usualmente acciones del usuario) e invoca peticiones al modelo cuando se hace alguna solicitud sobre la información. Se puede decir que el controlador hace de intermediario entre la vista y el modelo.

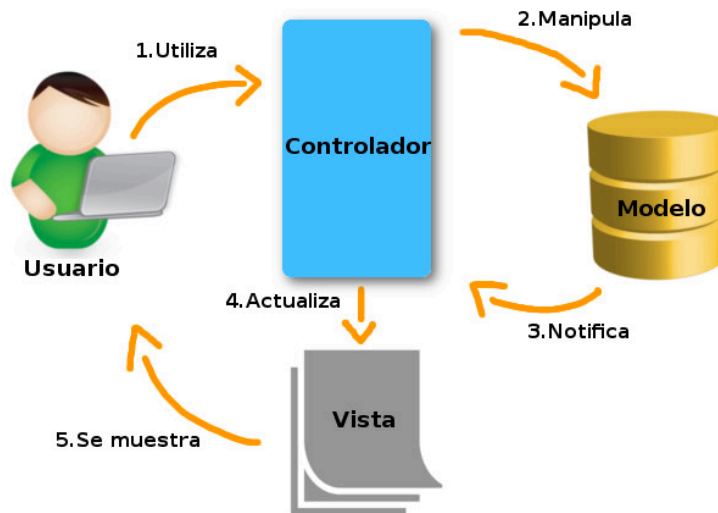


Figura 45 - Diagrama de comportamiento Patrón MVC

### 4.1.3 Patrón DAO

Se trata de un patrón utilizado en la capa de persistencia, cuyo principal objetivo es encapsular el acceso a los datos.

La capa de negocio accede a los datos a partir de la interfaz que este patrón ofrece.

La utilización de este patrón permite abstraer la aplicación de la fuente de datos, de forma que cualquier cambio en la fuente de datos de la aplicación no tendrá ningún impacto sobre el resto de capas



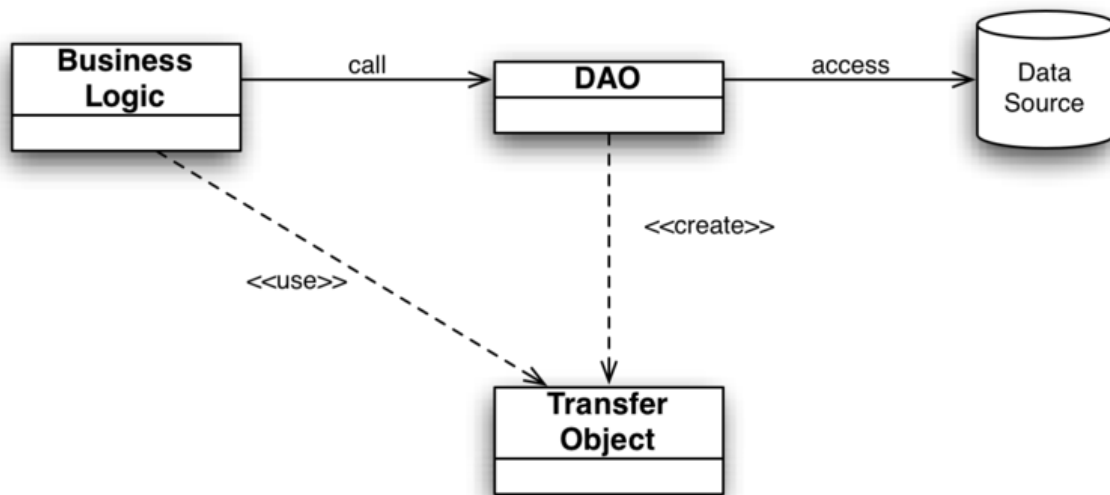


Figura 46 - Diagrama del patrón DAO

## 4.2 Descomposición en subsistemas

El sistema estará compuesto por tres módulos: tfg-model, tfg-backoffice y t

En la siguiente imagen puede apreciarse la descomposición del sistema:

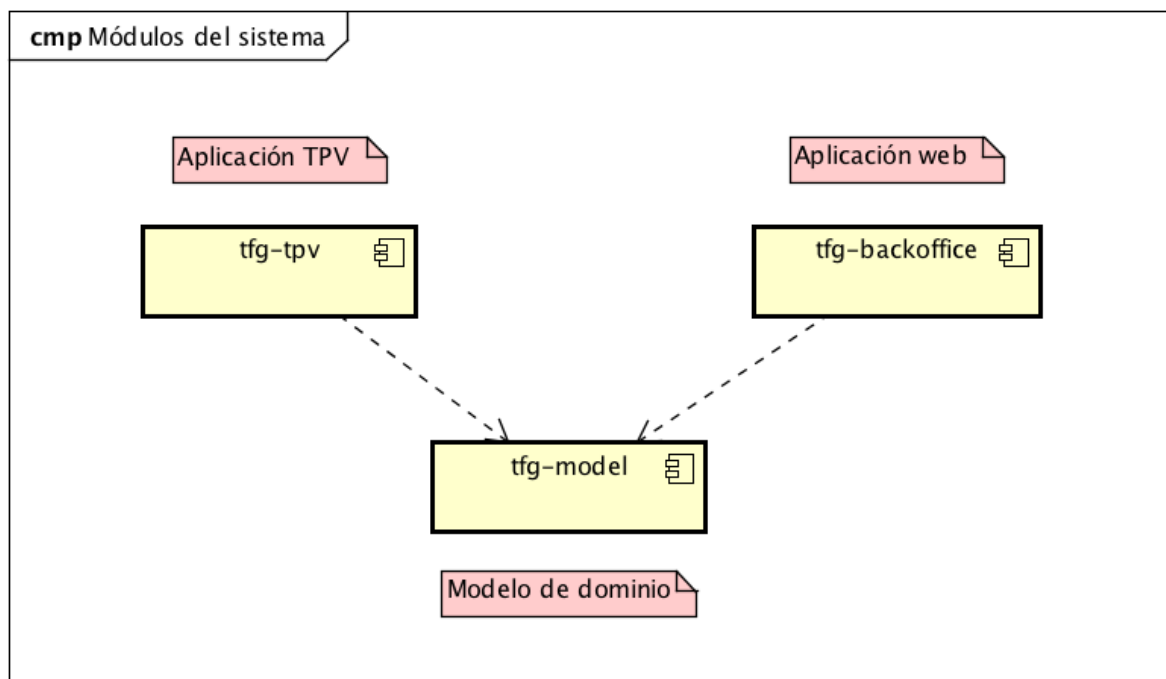


Figura 47 - Descomposición en subsistemas del Sistema Software

### 4.3 Módulo tfg-model

Este módulo contiene las entidades de dominio de la aplicación, comunes a los módulos de TPV y backoffice.

#### 4.3.1 Diagrama de clases

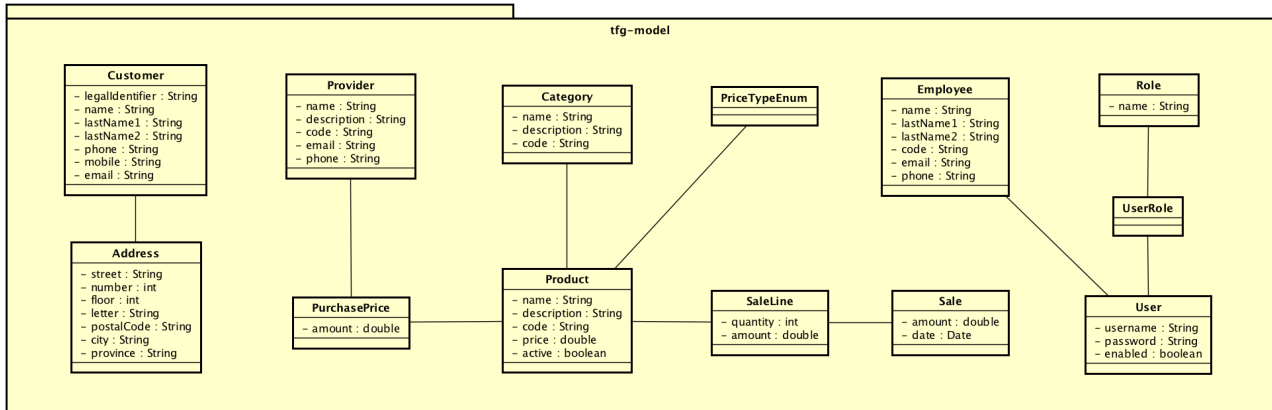


Figura 48 - Diagrama de clases del módulo tfg-model

### 4.4 Módulo tfg-backoffice

Este módulo contiene la aplicación web para la gestión y administración de las distintas entidades de dominio.

#### 4.4.1 Diagrama de clases

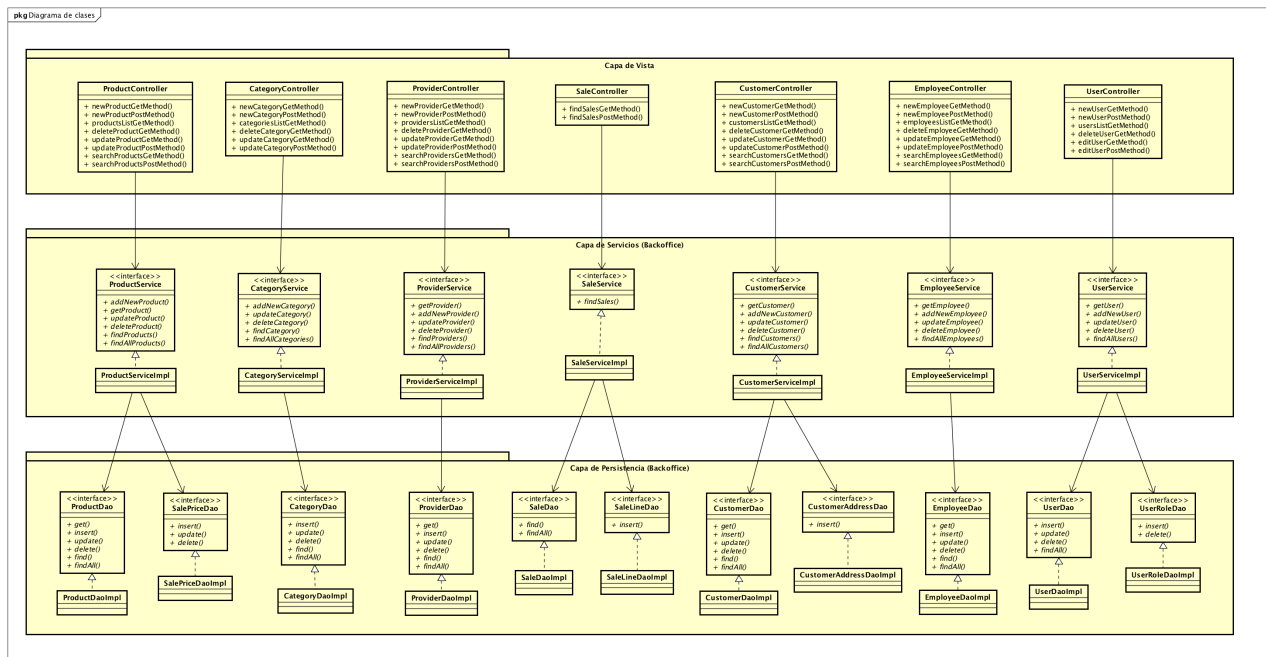


Figura 49 - Diagrama de clases

## 4.4.2 Capa de vista

Esta capa es la encargada de la comunicación del sistema con el usuario.

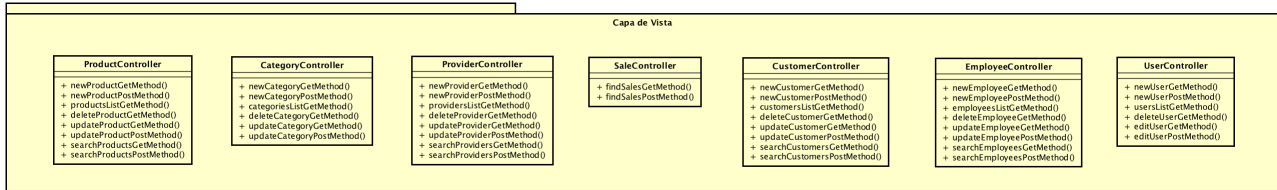


Figura 50 - Capa de vista del módulo de backoffice

## 4.4.3 Capa de servicios

En ella reside la lógica de negocio de la aplicación.

Se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de persistencia, para solicitar el almacenamiento o recuperación de datos.

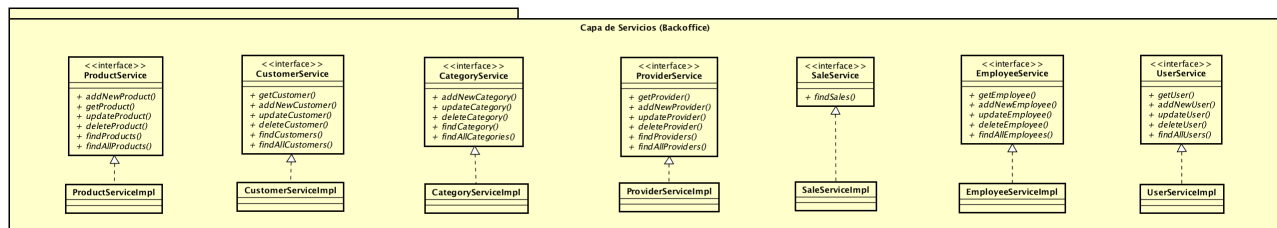


Figura 51 - Capa de servicios del módulo de backoffice

## 4.4.4 Capa de persistencia

Esta capa es la encargada de realizar todas las operaciones referentes al acceso y/o almacenamiento de los datos. Se encarga de recibir las solicitudes de almacenamiento o recuperar la información solicitada desde la capa de negocio.

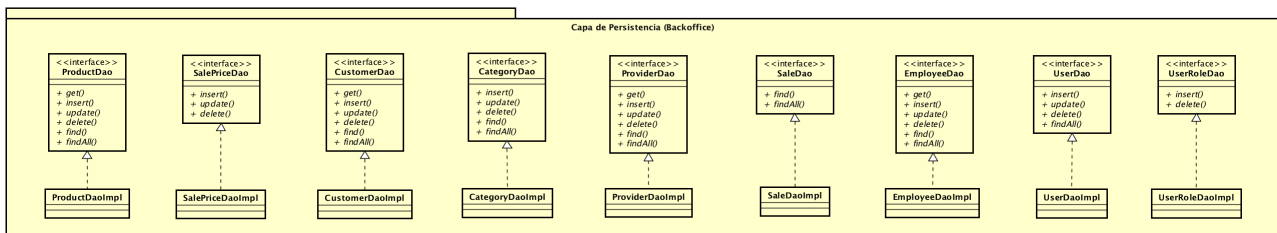


Figura 52 - Capa de persistencia del módulo de backoffice

## 4.5 Módulo tfg-tpv

Este módulo contiene la aplicación TPV de escritorio para la realización de las ventas.

### 4.5.1 Diagrama de clases

A continuación se muestra el diagrama de clases del módulo, con la especificación de cada una de las capas del módulo:

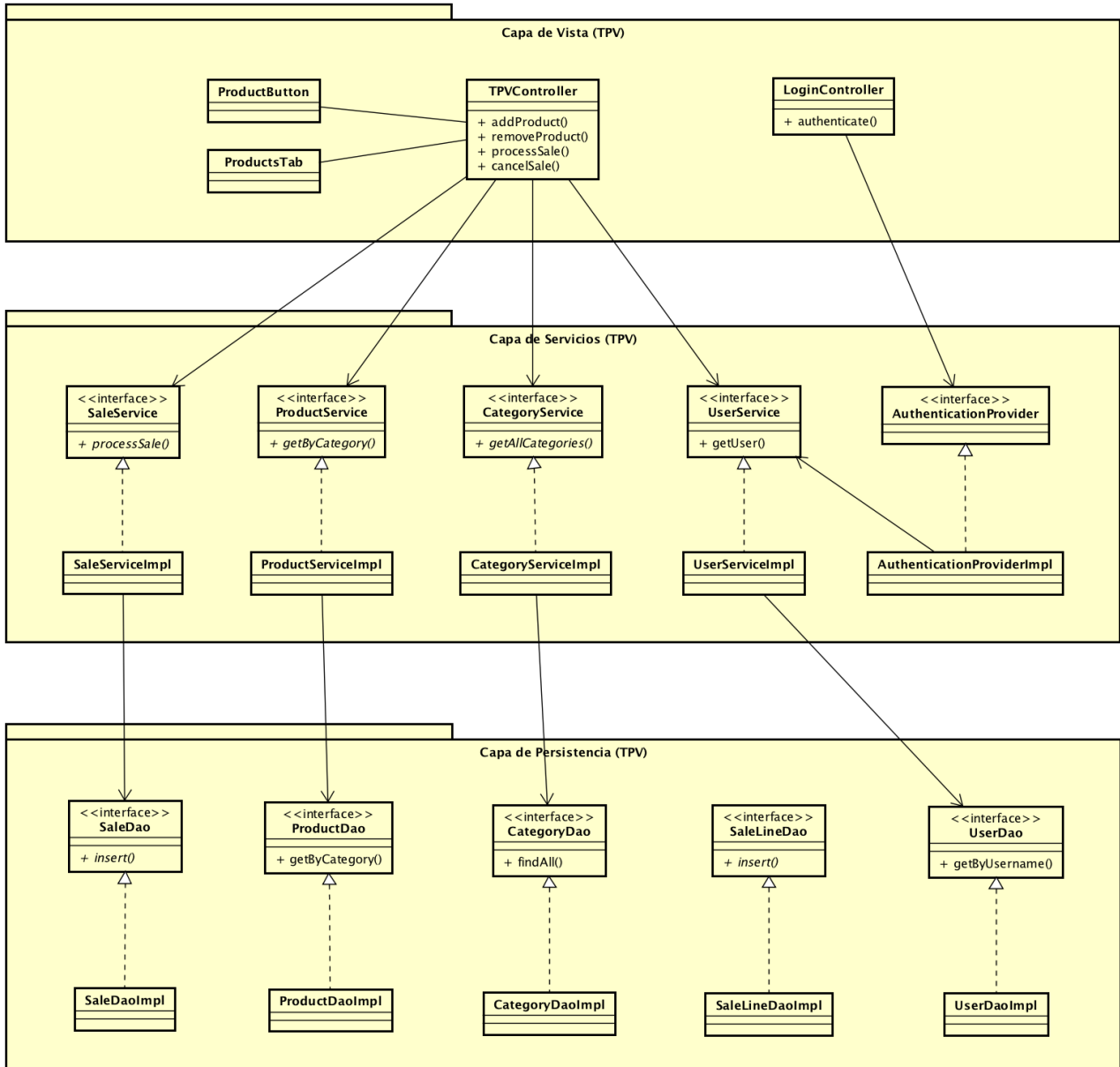


Figura 53 - Diagrama de clases

## 4.5.2 Capa de vista

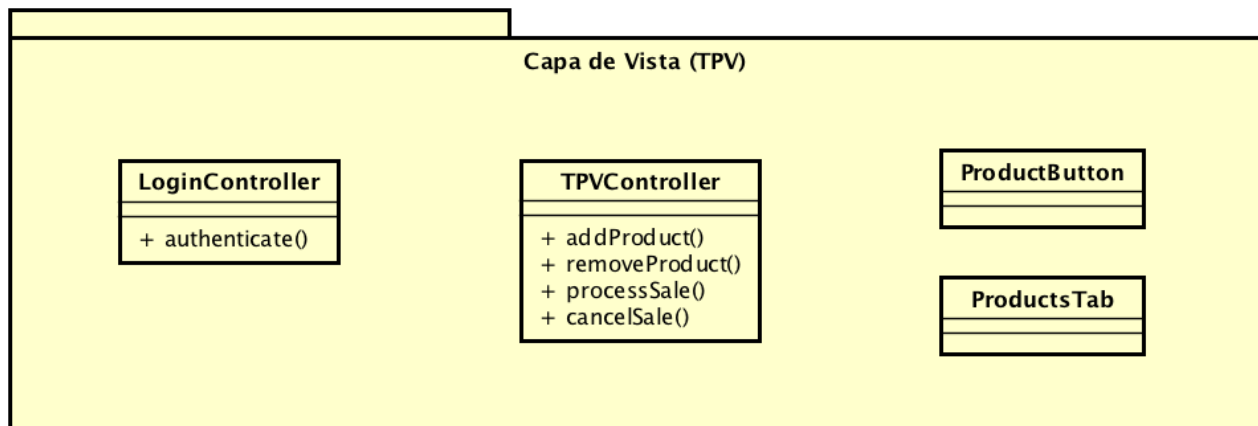


Figura 54 - Capa de vista del módulo de TPV

## 4.5.3 Capa de servicios

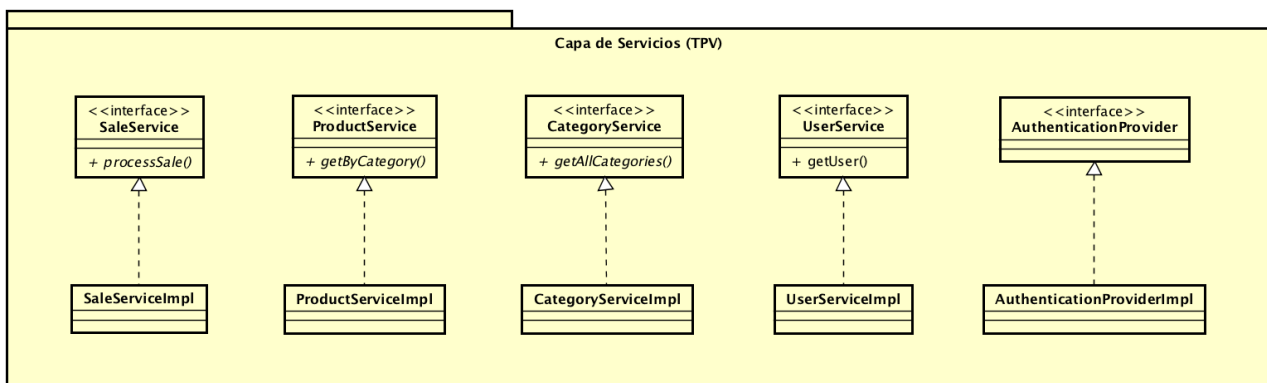


Figura 55 - Capa de servicios del módulo de TPV

## 4.5.4 Capa de persistencia

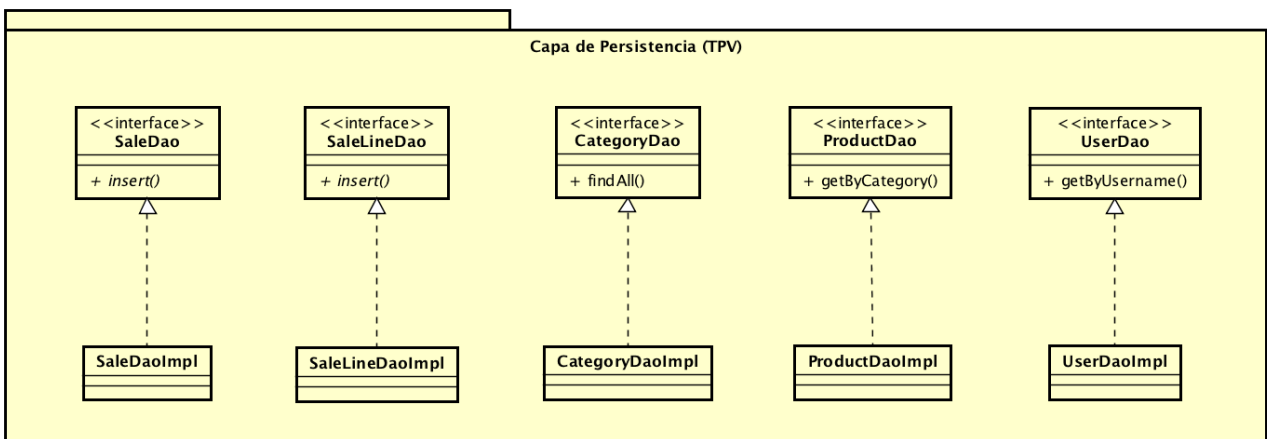


Figura 56 - Capa de persistencia del módulo de TPV

## 4.6 Interfaz de usuario

Para el diseño de la interfaz de usuario se ha priorizado la sencillez de uso, de forma que cualquier usuario con un nivel medio pueda familiarizarse con la aplicación en un corto periodo de tiempo.

La aplicación de backoffice se divide en las siguientes funcionalidades: añadir, editar, eliminar, listado y búsqueda.

Por su parte, la aplicación de TPV constará únicamente de dos pantallas: la pantalla de acceso a la aplicación y la pantalla de venta.

Debido a que la interfaz es muy similar sea cual sea el 'objeto' en cuestión (clientes, productos, proveedores, etc), se presentan unos mockups genéricos, que muestran el diseño para cada funcionalidad.

A continuación se presentan los mockups con el diseño de la interfaz de usuario para cada una de las funcionalidades.

### 4.6.1 Pantalla para añadir nuevo elemento

El mockup muestra una interfaz de usuario dentro de un navegador web. El navegador tiene una barra de direcciones con "http://" y un botón de búsqueda. El título de la página es "A Web Page".

El contenido principal de la página está dividido en una barra lateral de navegación y un área de formulario principal.

La barra lateral de navegación contiene los siguientes elementos:

- Ventas
- Productos
  - Añadir nuevo producto
  - Listado de productos
  - Búsqueda de productos
- Clientes
- Proveedores
- Empleados
- Usuarios

El área de formulario principal tiene el título "Añadir nuevo elemento" y un menú de navegación con "Inicio" y "Menu".

El formulario contiene ocho campos de entrada:

- Campo 1: Campo de texto simple.
- Campo 2: Campo de texto simple.
- Campo 3: Campo de texto simple.
- Campo 4: Menú desplegable con opciones "Opción 1", "Opción 2" y "Opción 3".
- Campo 5: Campo de texto simple.
- Campo 6: Campo de texto simple.
- Campo 7: Campo de texto simple.
- Campo 8: Campo de texto simple.

En la parte inferior derecha del formulario hay dos botones: "Limpiar" y "Guardar".

Figura 57 - Mockup para añadir nuevo elemento

Como puede apreciarse en el mockup, la aplicación constará de un menú lateral de navegación, desde el que se podrá acceder a cada una de las funcionalidades.

## 4.6.2 Pantalla para listado de elementos

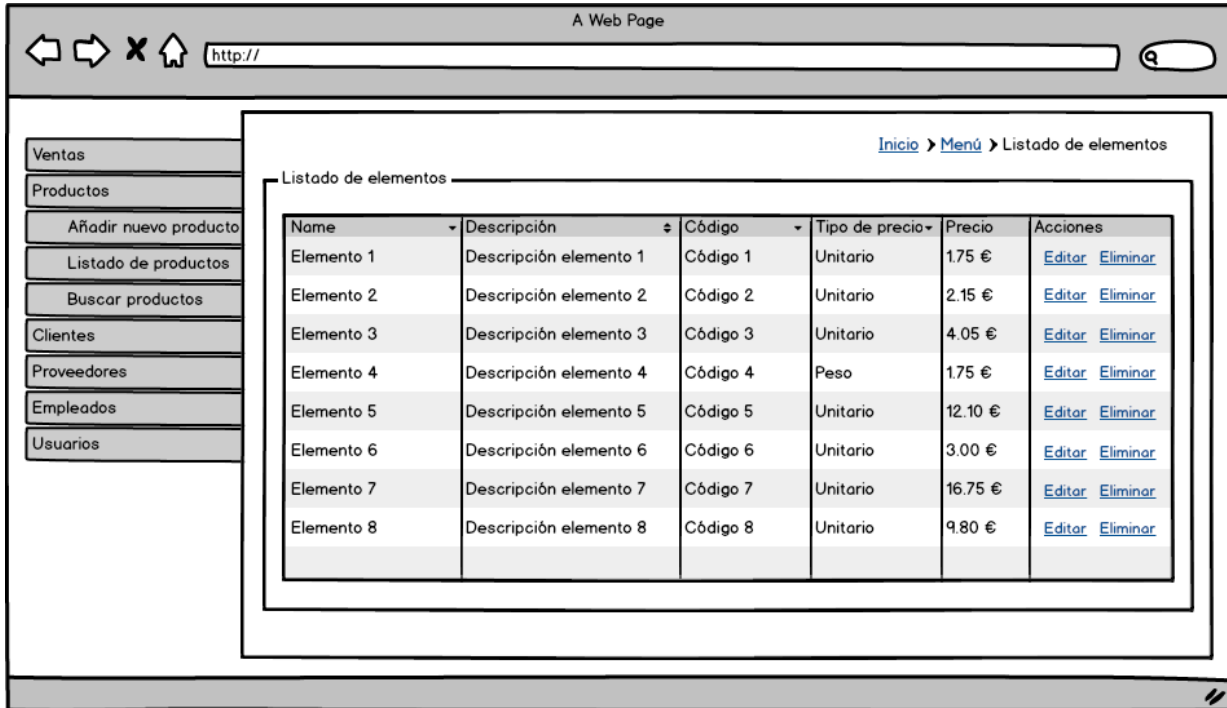


Figura 58 - Mockup para listado de elementos

## 4.6.3 Pantalla para búsqueda de elementos

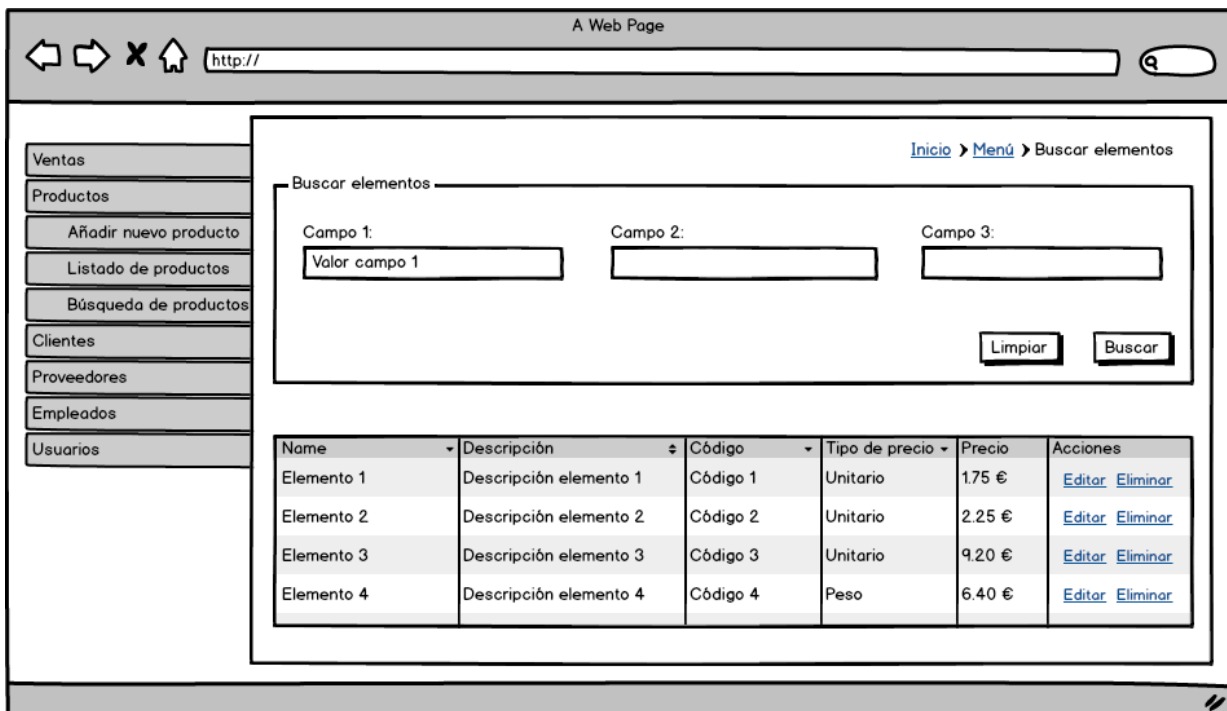


Figura 59 - Mockup para búsqueda de elementos

## 4.6.2 Pantalla para eliminar elemento

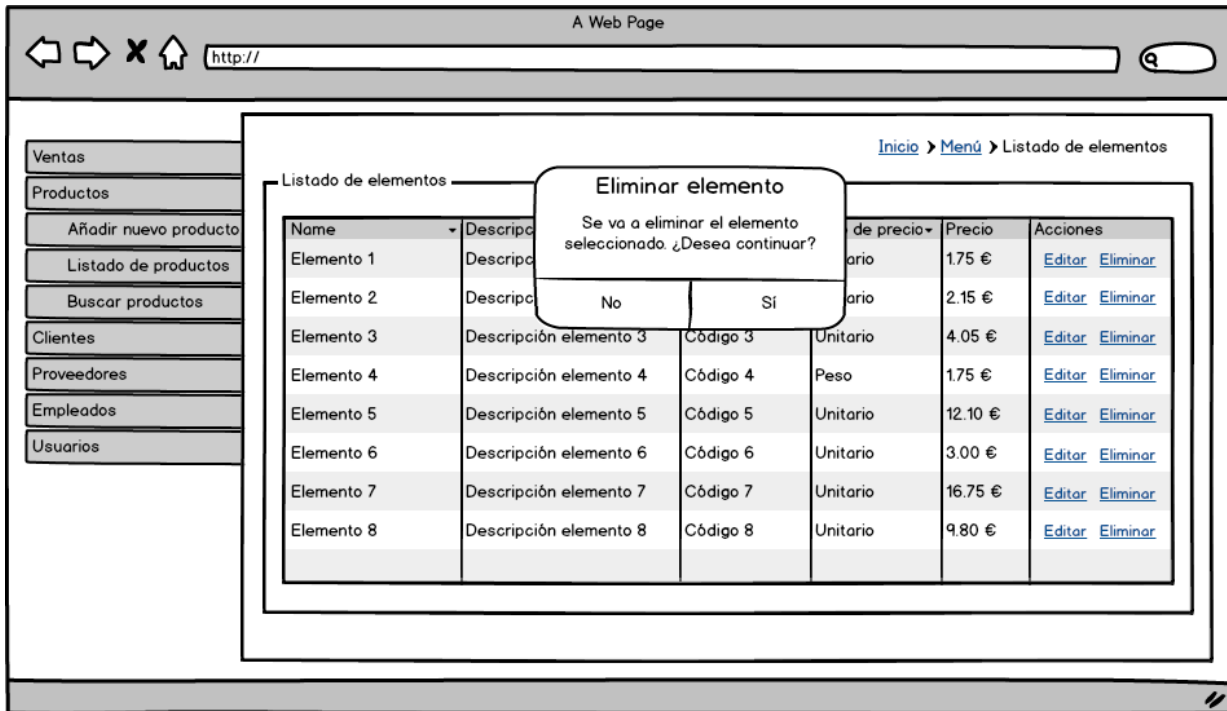


Figura 60 - Mockup para eliminar elemento

## 4.6.2 Pantalla para editar elemento

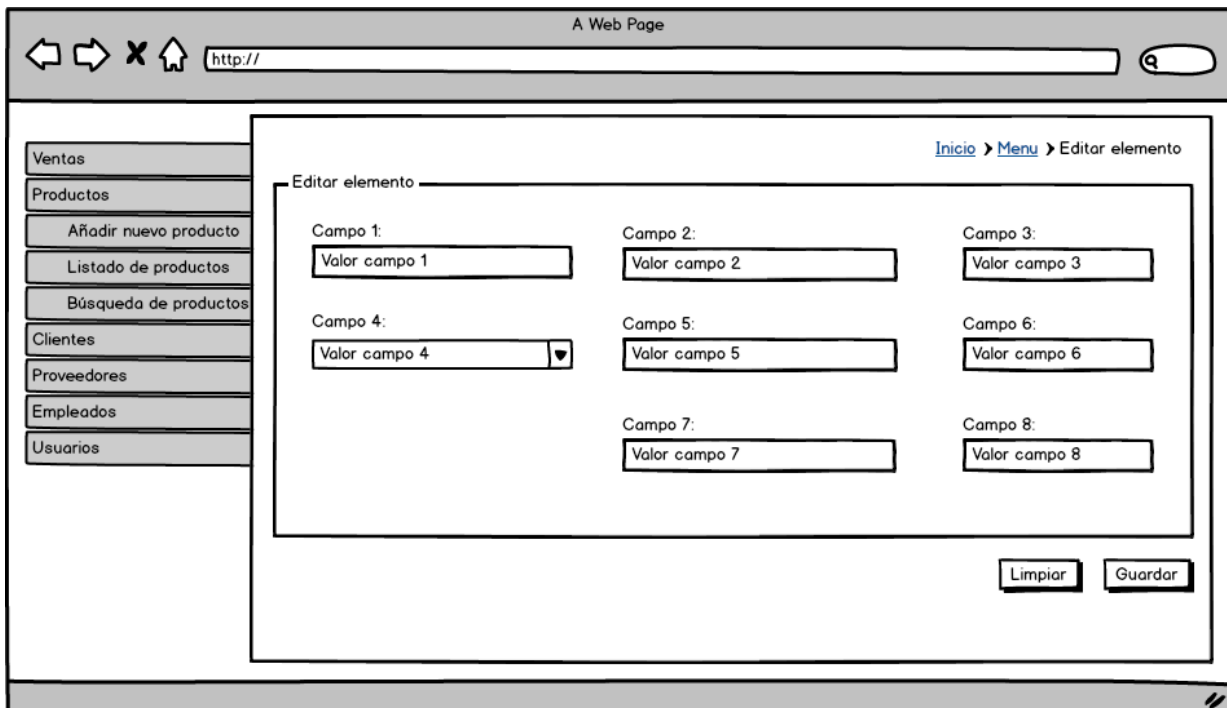


Figura 61 - Mockup para editar elemento



#### 4.5.4 Pantalla de acceso a la aplicación

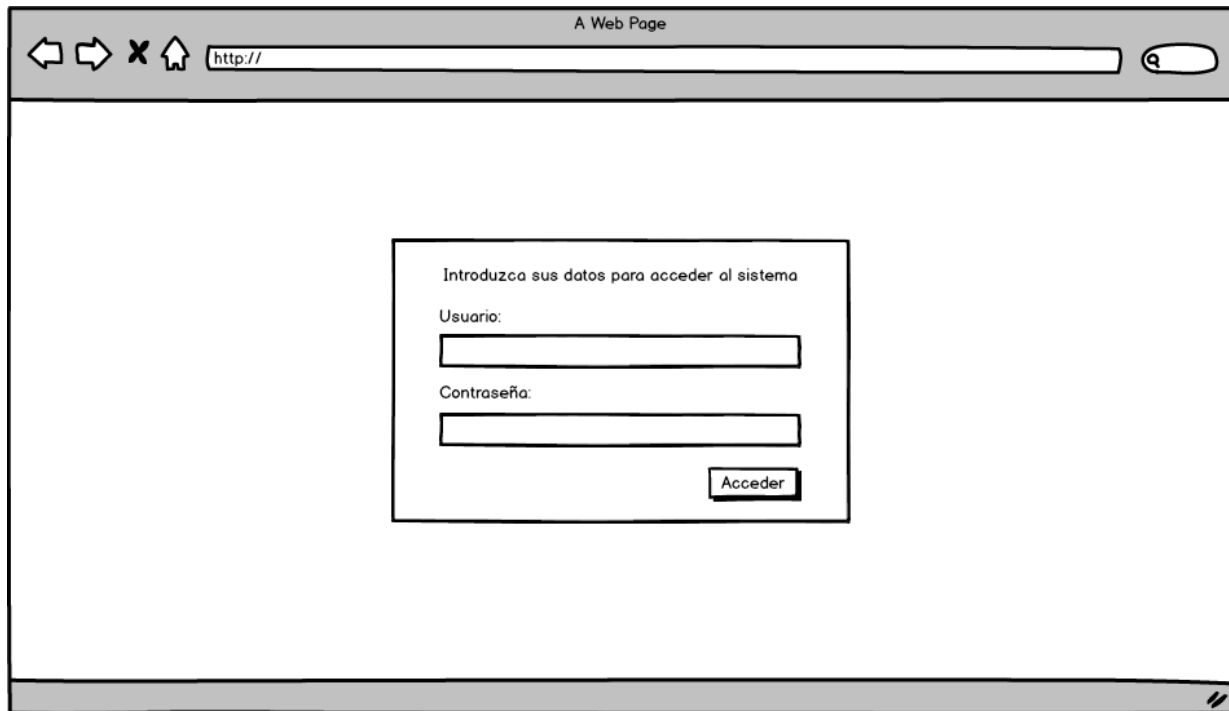


Figura 62 - Mockup de acceso a la aplicación

#### 4.5.4 Pantalla de venta TPV

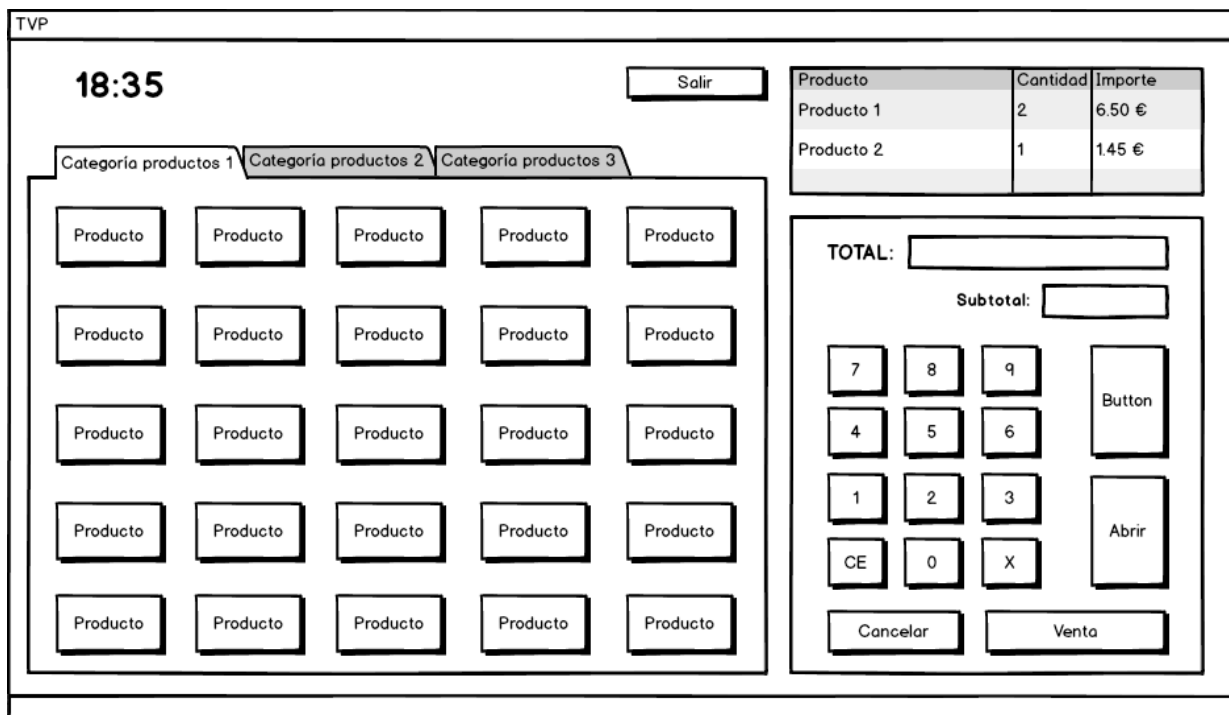


Figura 63 - Mockup de pantalla de venta TPV

## 4.7 Modelo de datos

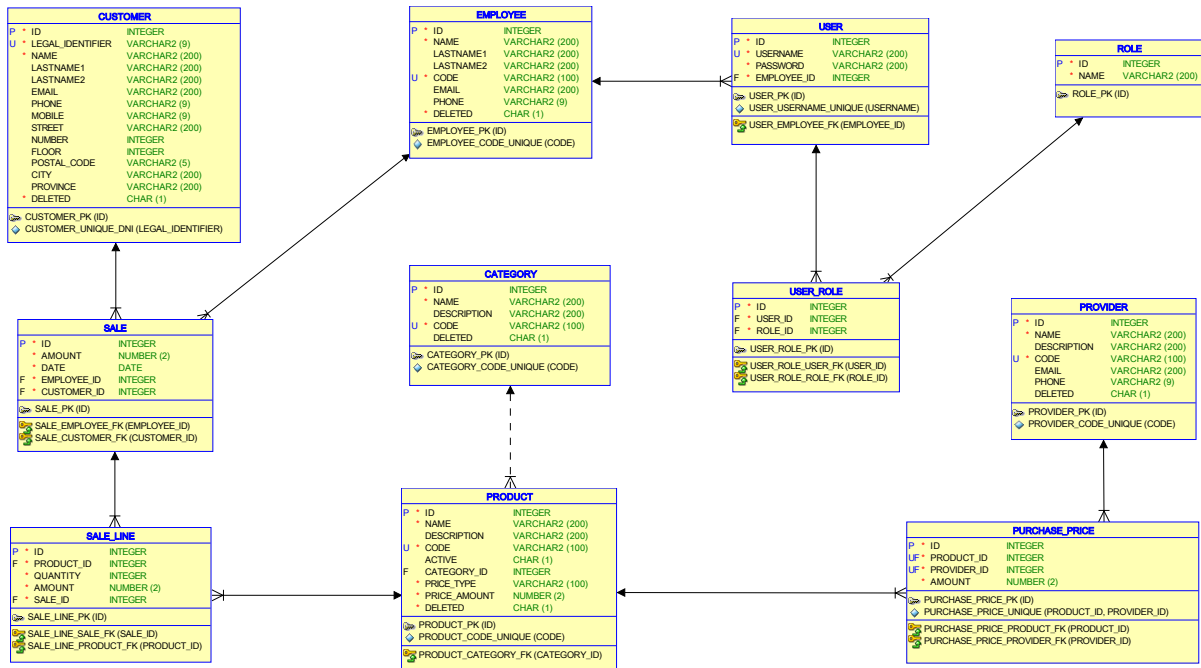


Figura 64 - Esquema relacional del modelo de datos del sistema

El borrado ó eliminación de datos se realizará mediante **borrado lógico**, para preservar la información en el sistema, evitando situaciones de inconsistencia de datos.

La realización de un borrado físico podría provocar situaciones de inconsistencia de datos. Por ejemplo, podrían darse situaciones en las que las ventas realizadas quedaran desligadas del cliente que las realizó, de forma que no sería posible recuperar datos de ventas anteriores.

Para llevar a cabo este borrado lógico, se añade una columna 'DELETED' en las tablas que almacenan datos susceptibles de ser borrados, y será esta columna la que indique si el registro ha sido o no borrado.

## 4.8 Diagramas de secuencia

A continuación se presentan los diagramas de secuencia de los casos de uso especificados anteriormente.

Las anotaciones 'Test' en cada uno de los diagramas indican la existencia de un test unitario que prueba el correcto funcionamiento de ese método.

## 4.8.1 SD-001: Añadir producto

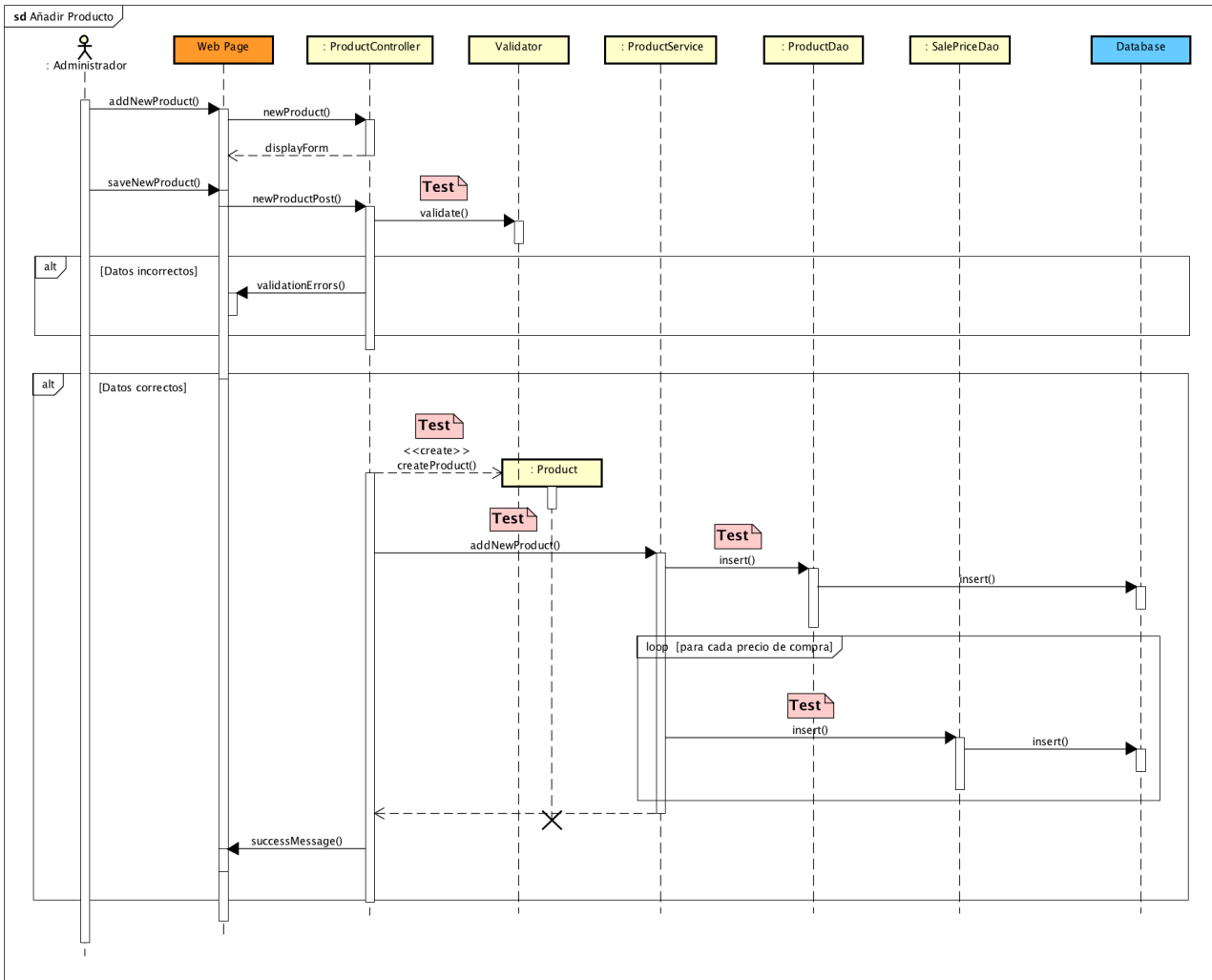


Figura 65 - Diagrama secuencia SD-001

## 4.8.2 SD-002: Consultar productos

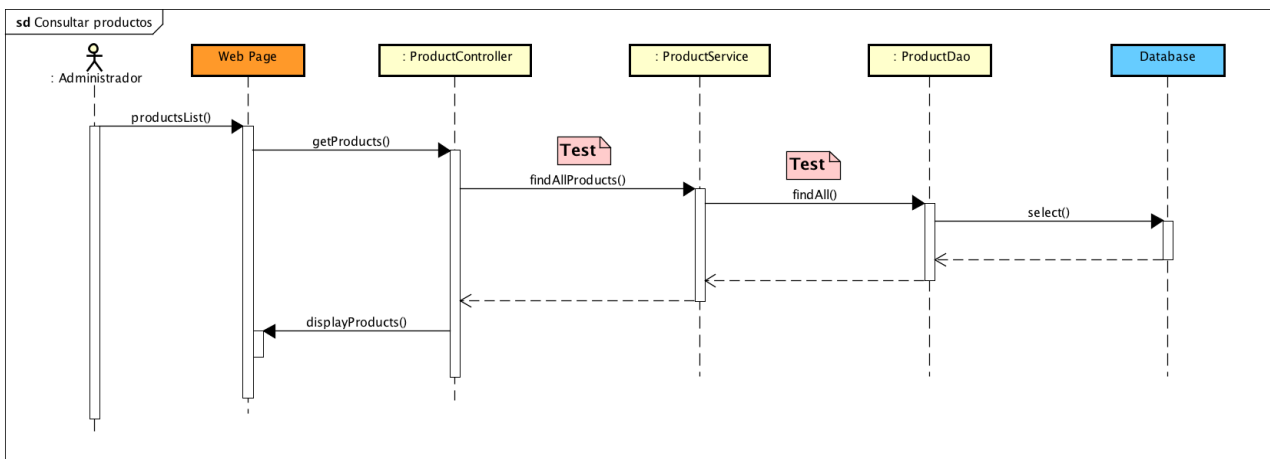


Figura 66 - Diagrama secuencia SD-003

### 4.8.3 SD-003: Eliminar Producto

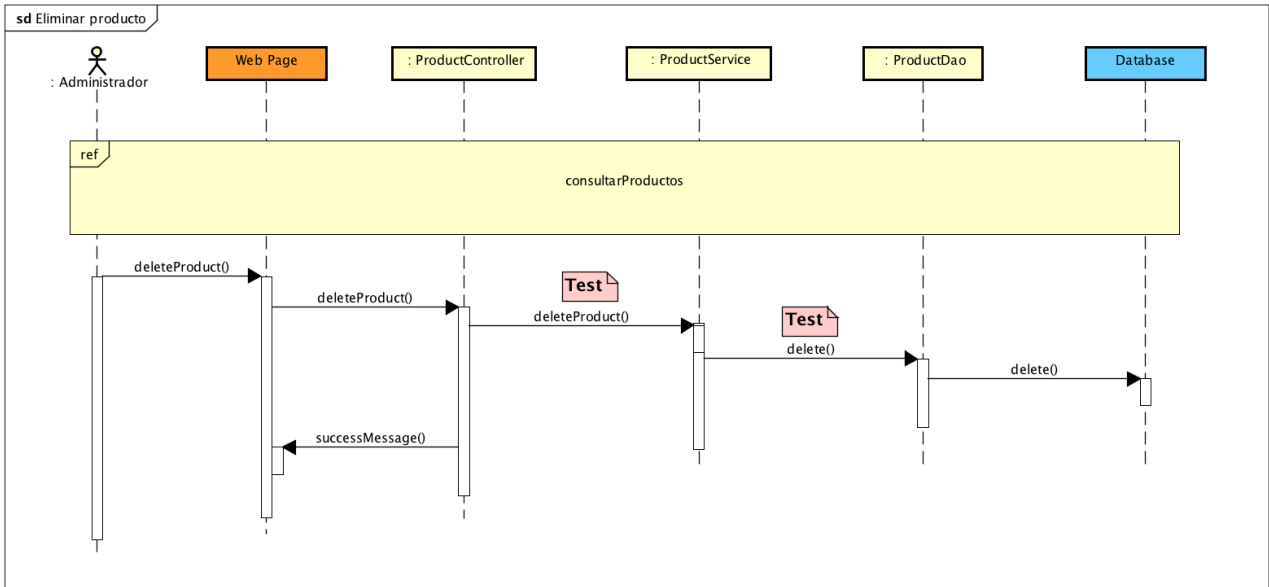


Figura 67 - Diagrama secuencia SD-002

### 4.8.4 SD-004: Buscar productos

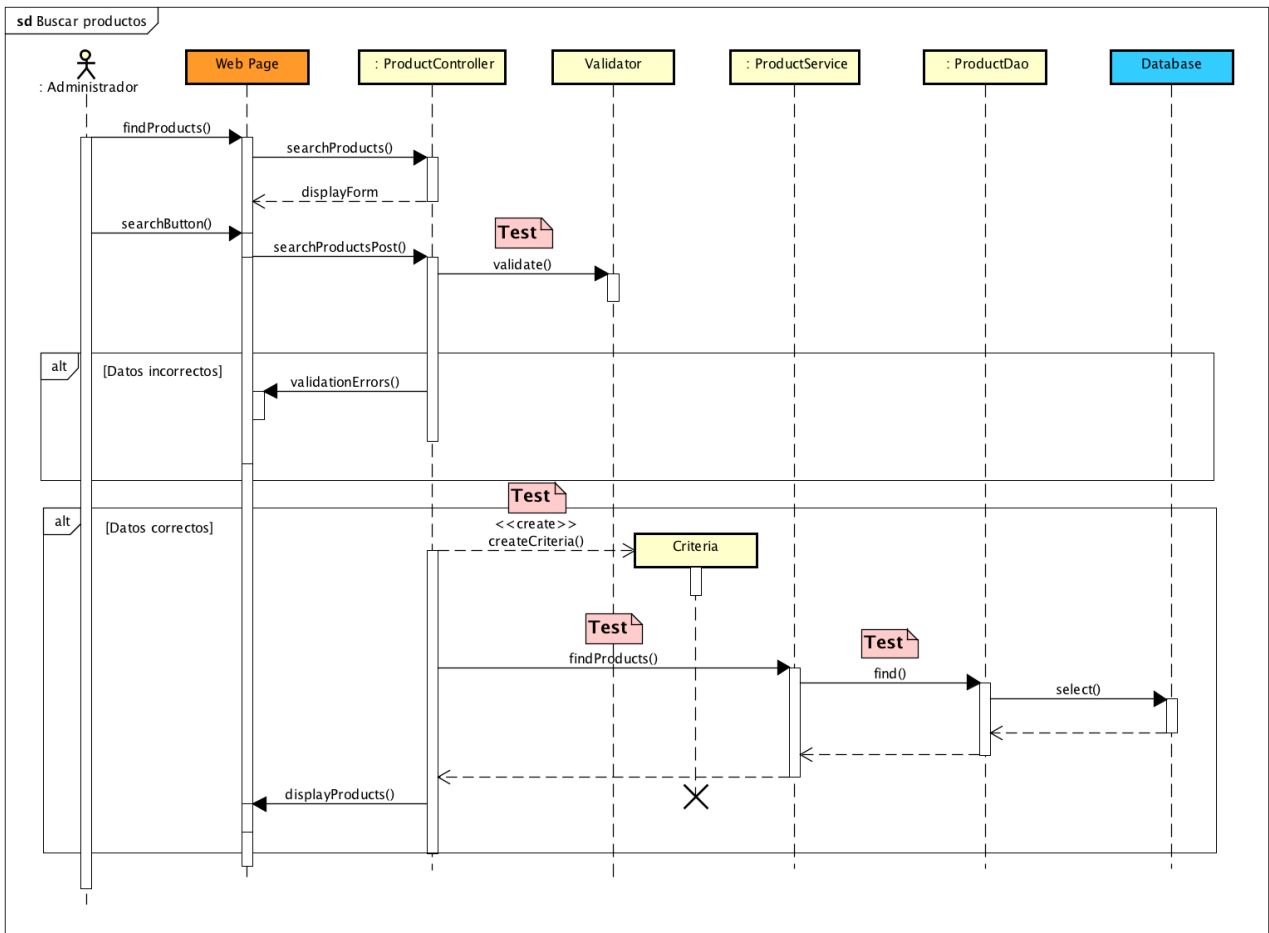


Figura 68 - Diagrama secuencia SD-004

### 4.8.5 SD-005: Editar Producto

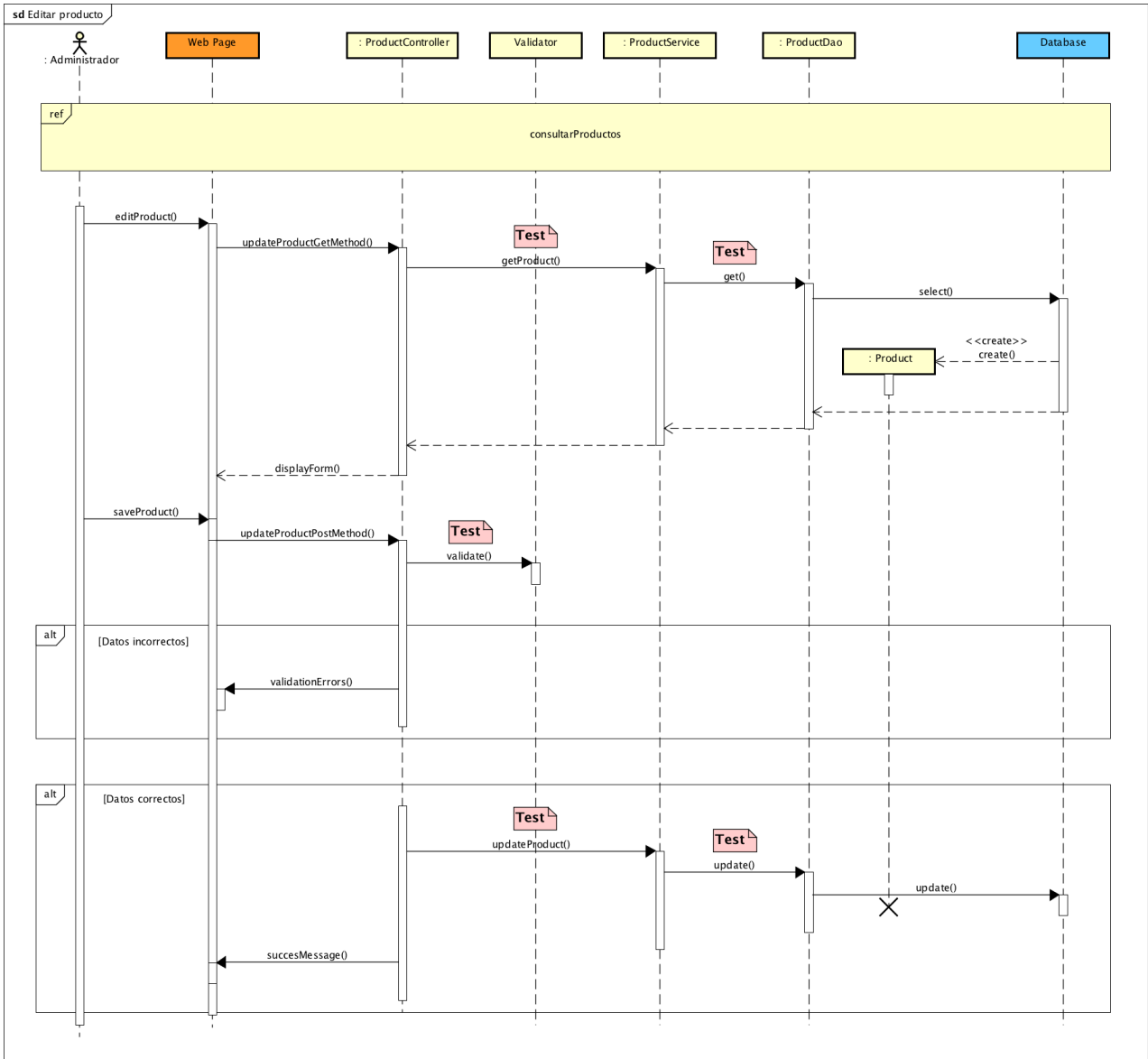


Figura 69 - Diagrama secuencia SD-005

### 4.8.6 SD-006: Añadir categoría

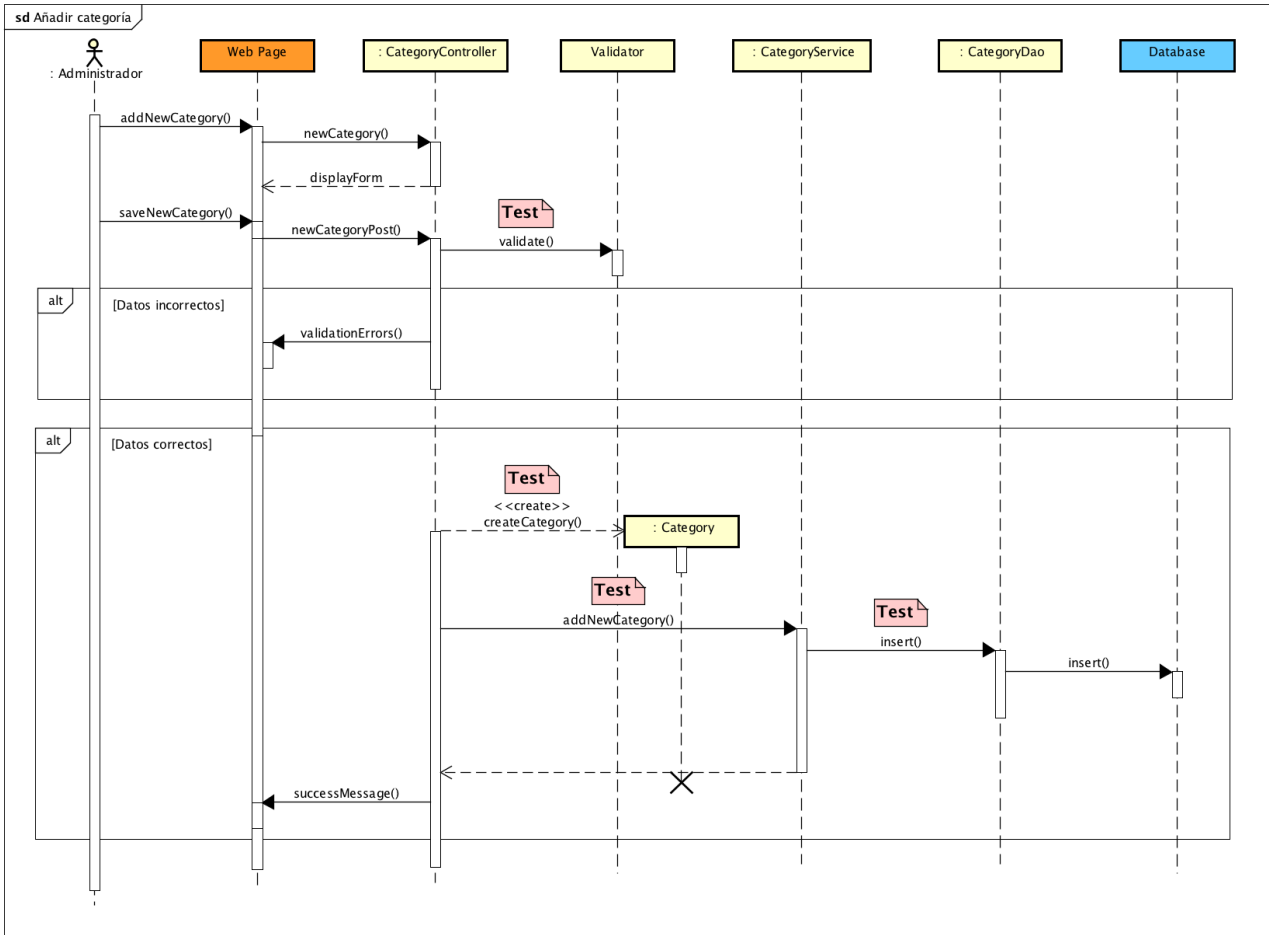


Figura 70 - Diagrama secuencia SD-006

### 4.8.7 SD-007: Consultar categorías

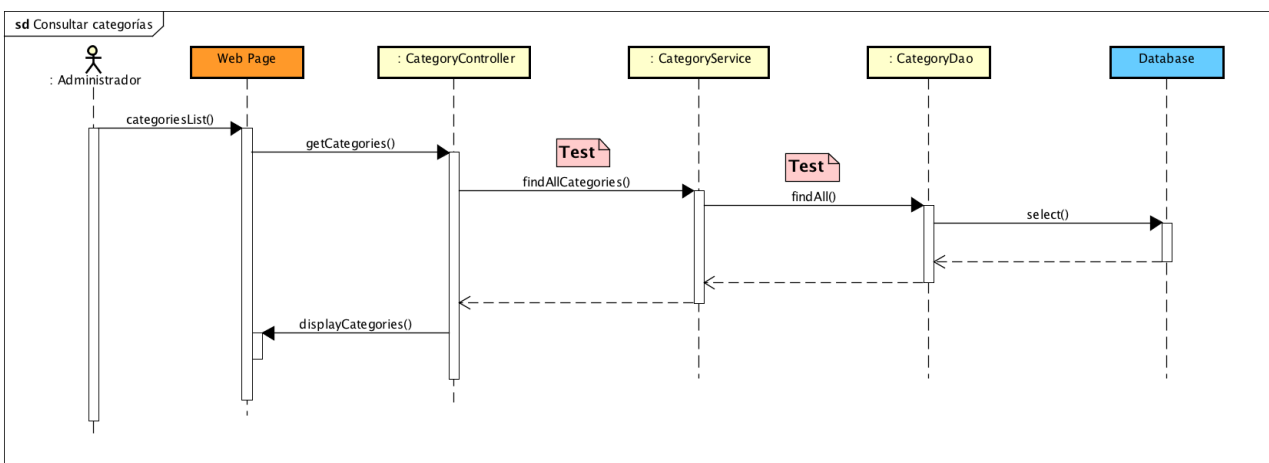


Figura 71 - Diagrama secuencia SD-008

#### 4.8.8 SD-008: Eliminar categoría

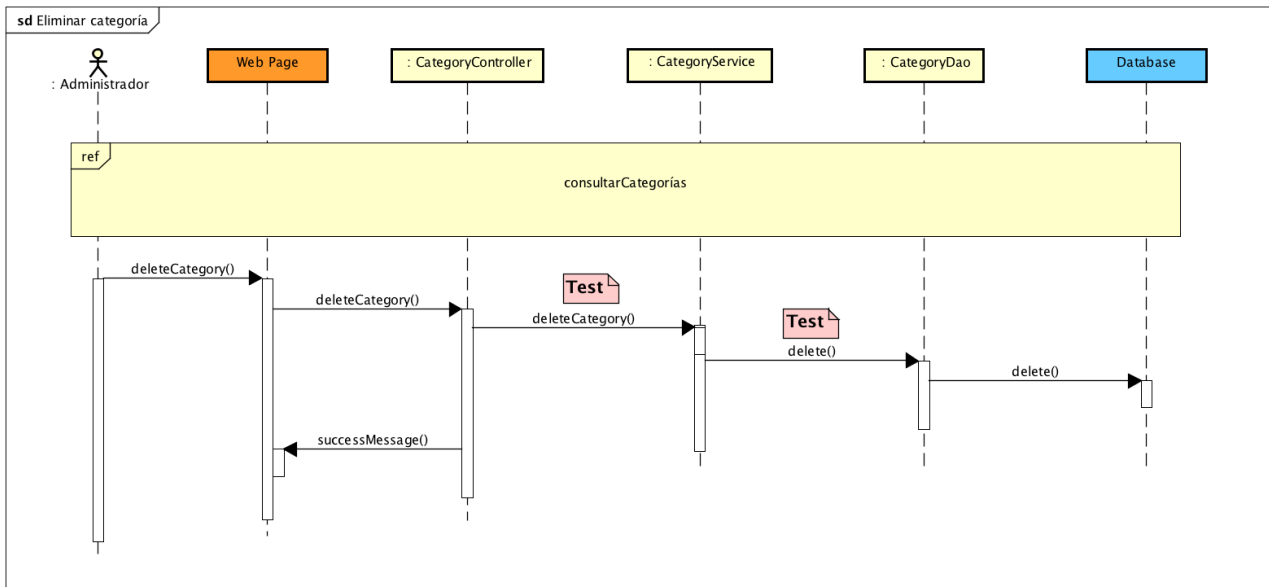


Figura 72 - Diagrama secuencia SD-007

## 4.8.9 SD-009: Editar categoría

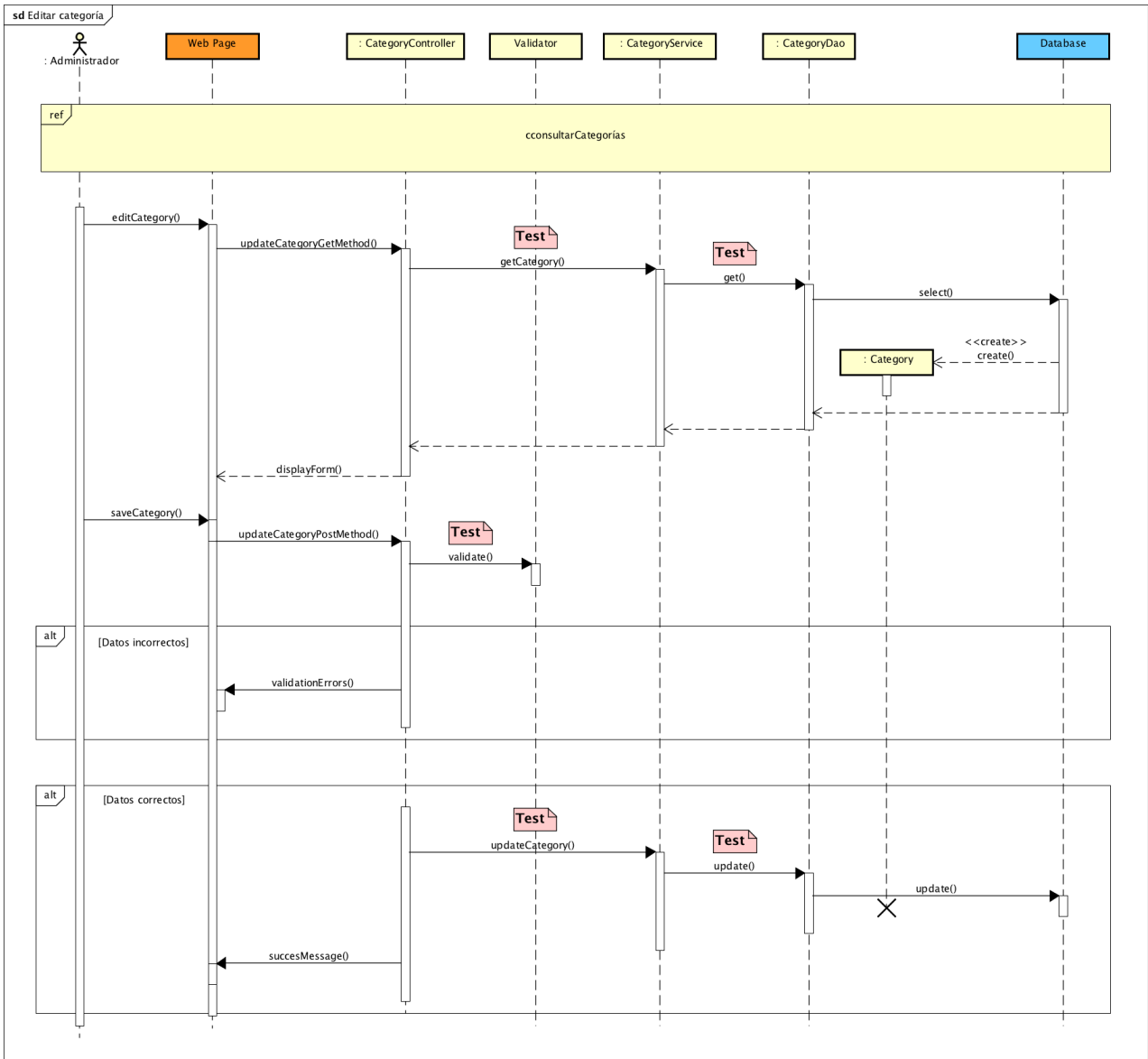


Figura 73 - Diagrama secuencia SD-009



### 4.8.10 SD-010: Añadir cliente

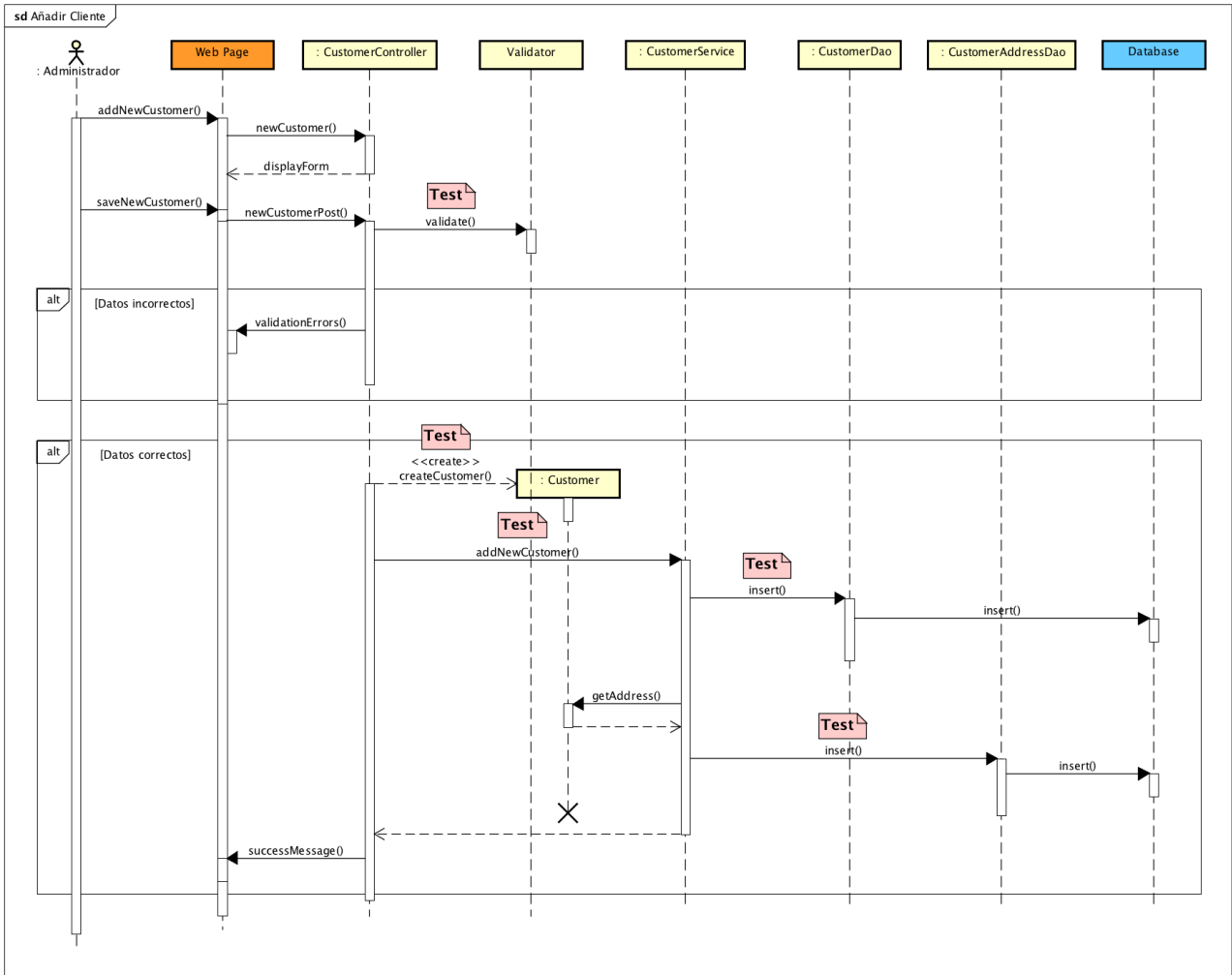


Figura 74 - Diagrama secuencia SD-010

### 4.8.11 SD-011: Consultar clientes

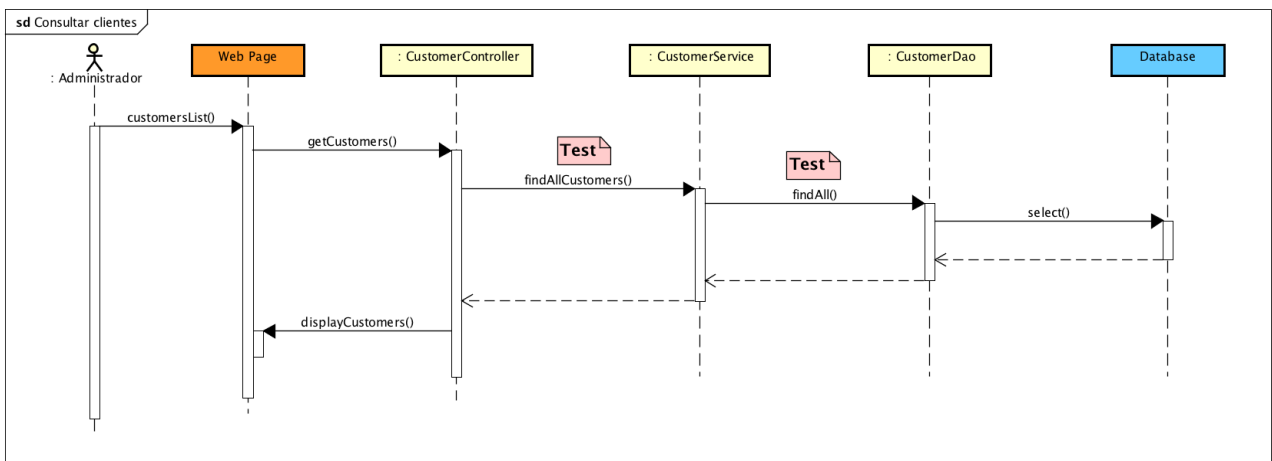


Figura 75 - Diagrama secuencia SD-012

### 4.8.12 SD-012: Eliminar cliente

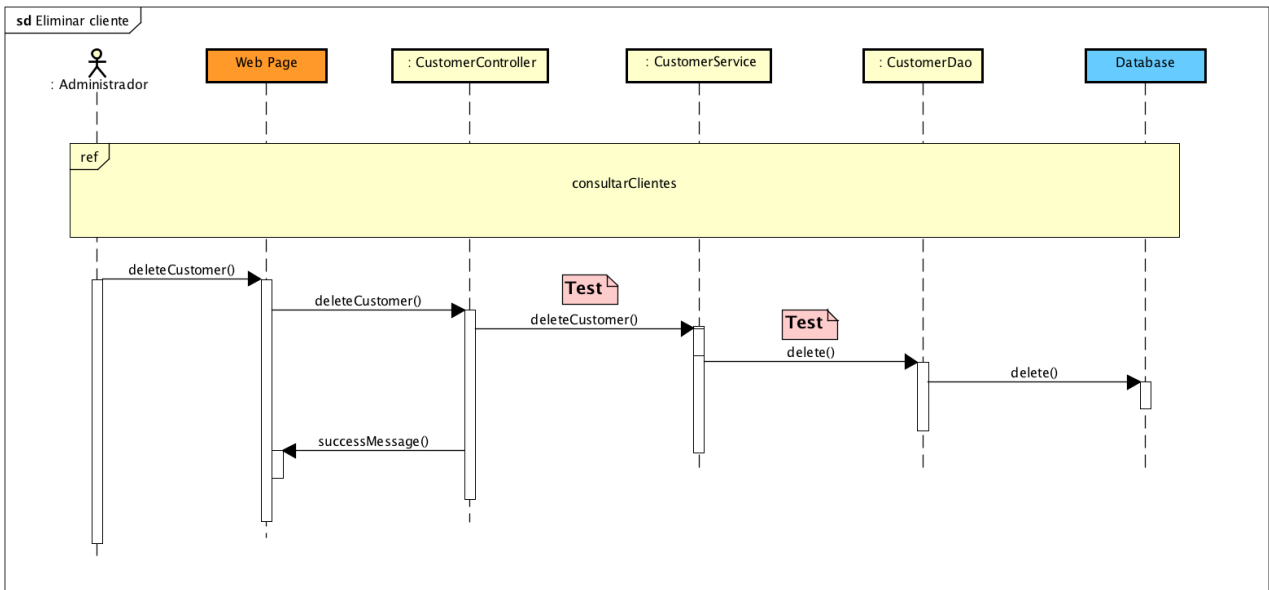


Figura 76 - Diagrama secuencia SD-011

### 4.8.13 SD-013: Buscar clientes

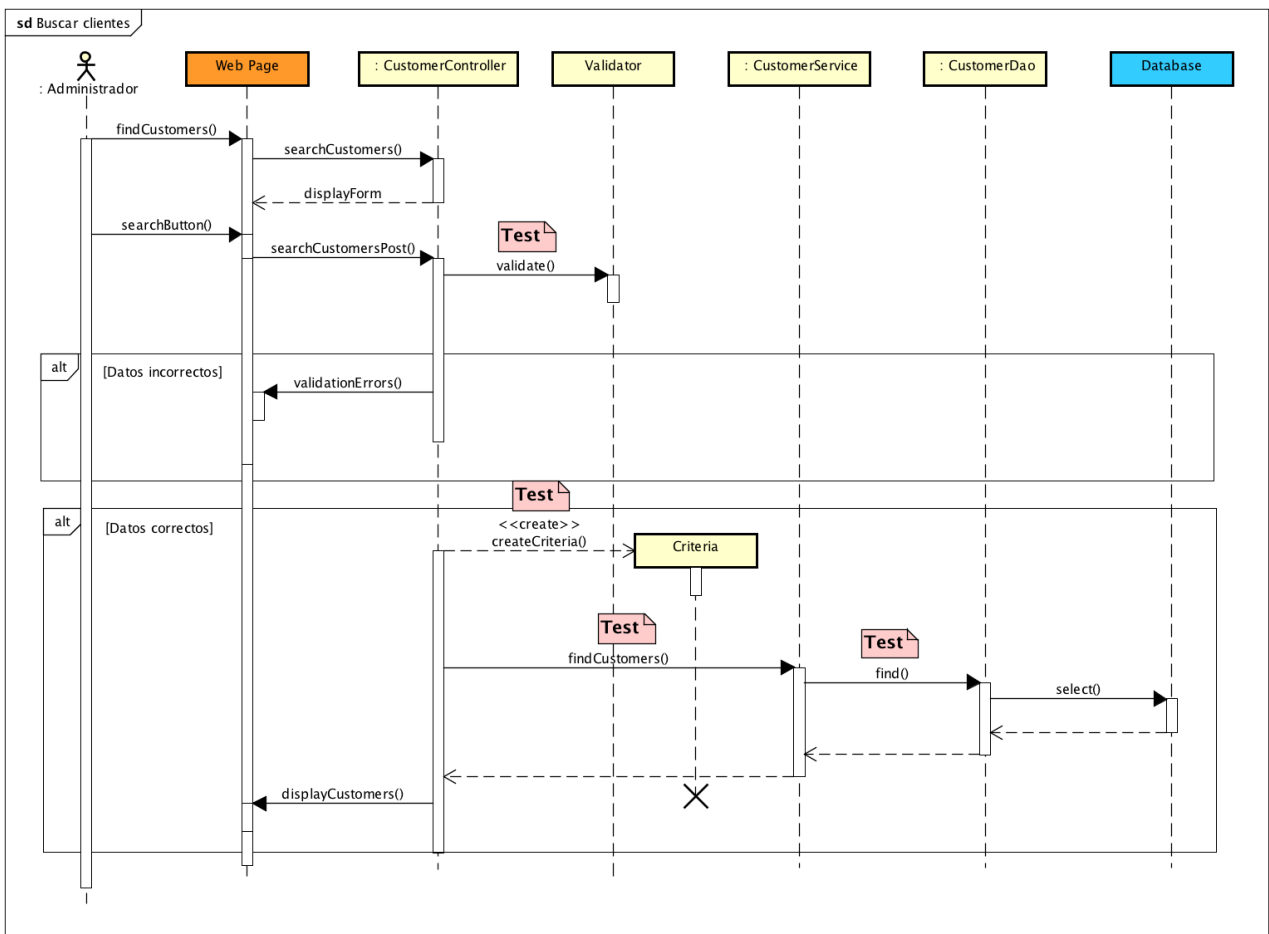


Figura 77 - Diagrama secuencia SD-013

## 4.8.14 SD-014: Editar cliente

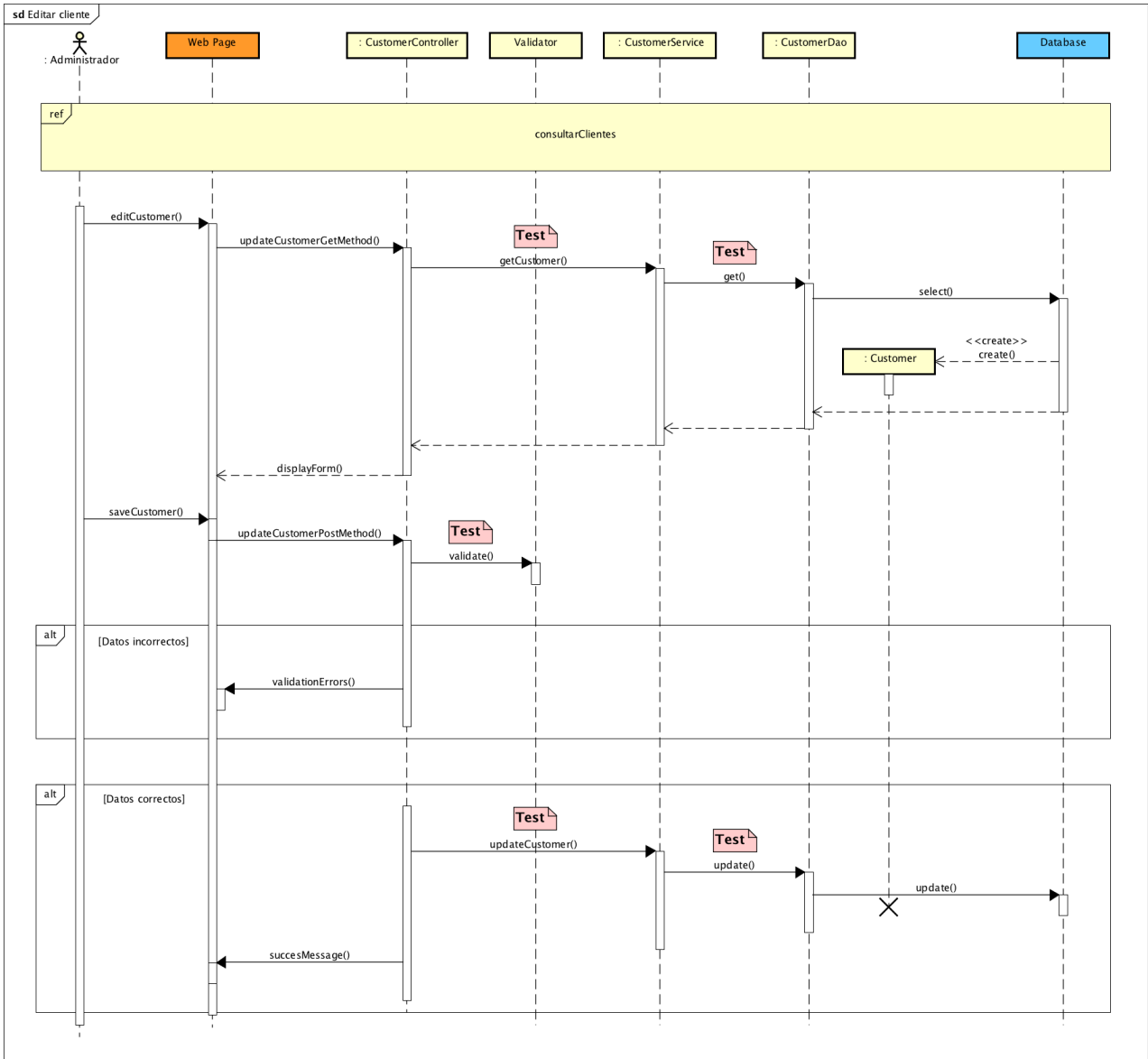


Figura 71 - Diagrama secuencia SD-014

### 4.8.15 SD-015: Añadir proveedor

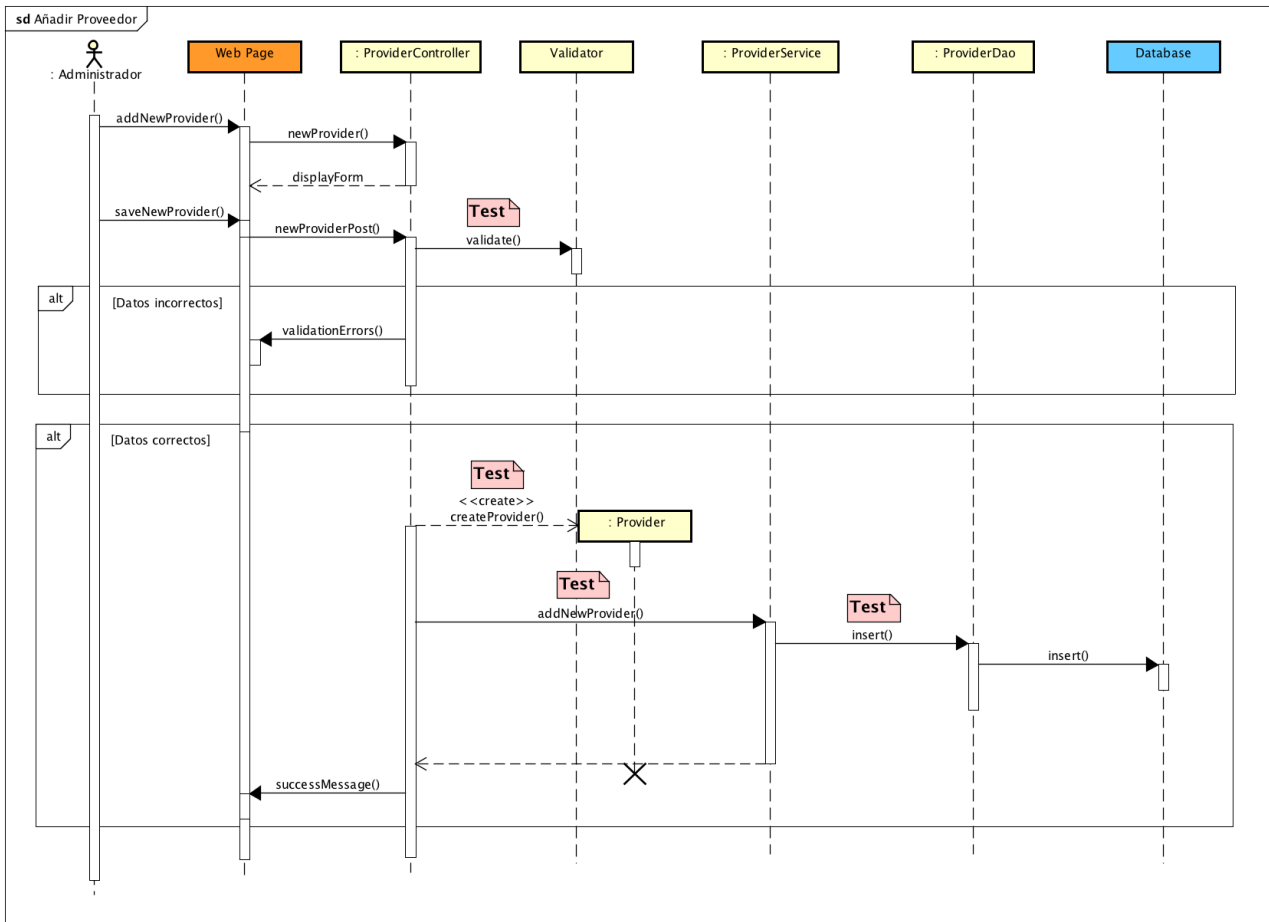


Figura 79 - Diagrama secuencia SD-015

### 4.8.16 SD-016: Consultar proveedores

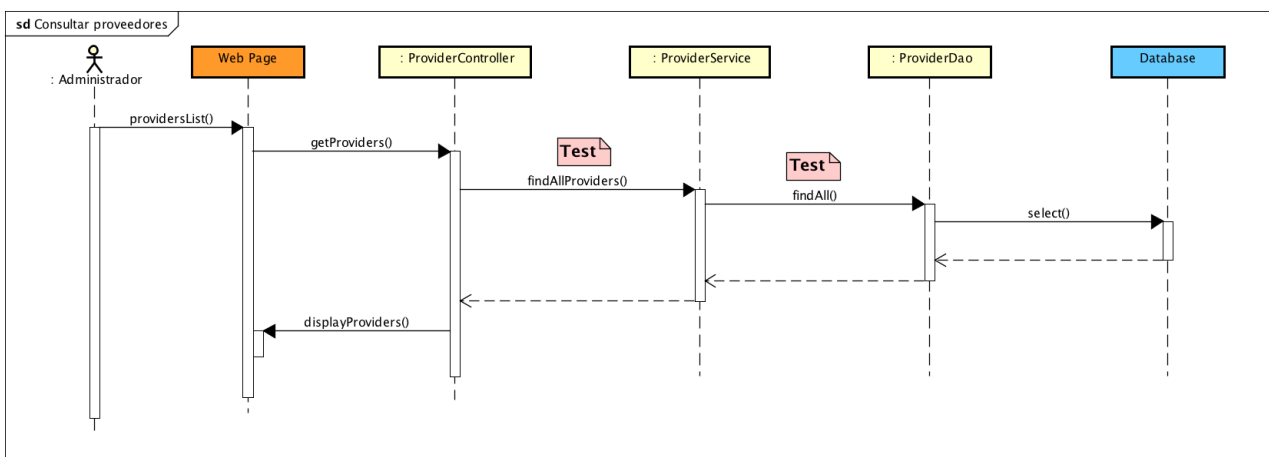


Figura 80 - Diagrama secuencia SD-016

### 4.8.17 SD-017: Eliminar proveedor

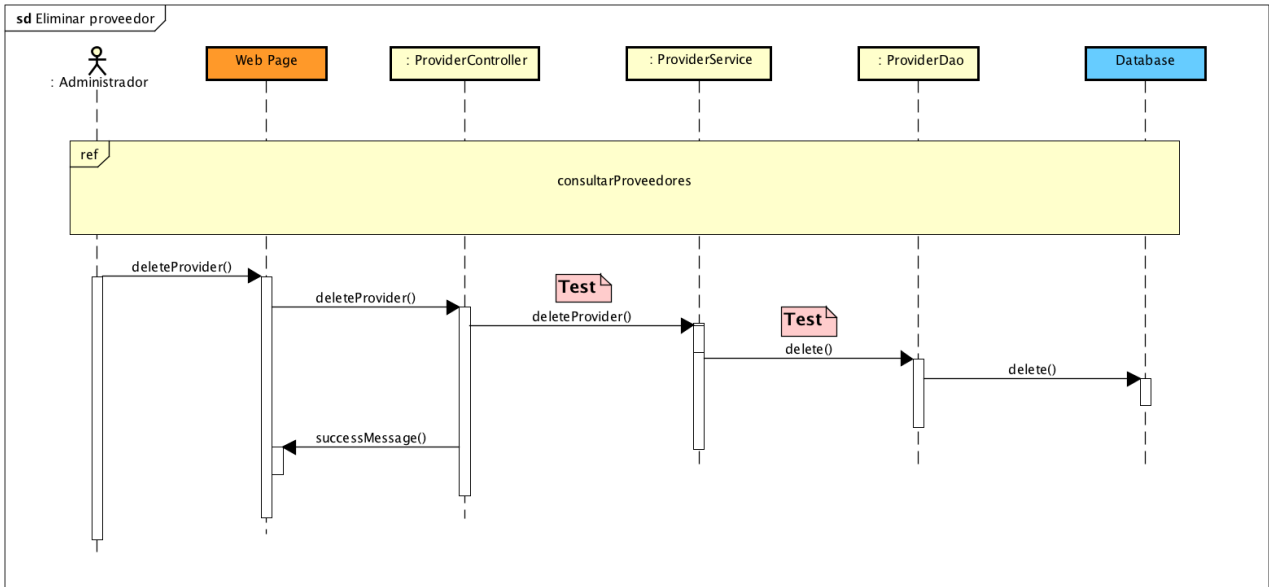


Figura 81 - Diagrama secuencia SD-017

### 4.8.18 SD-018: Buscar proveedores

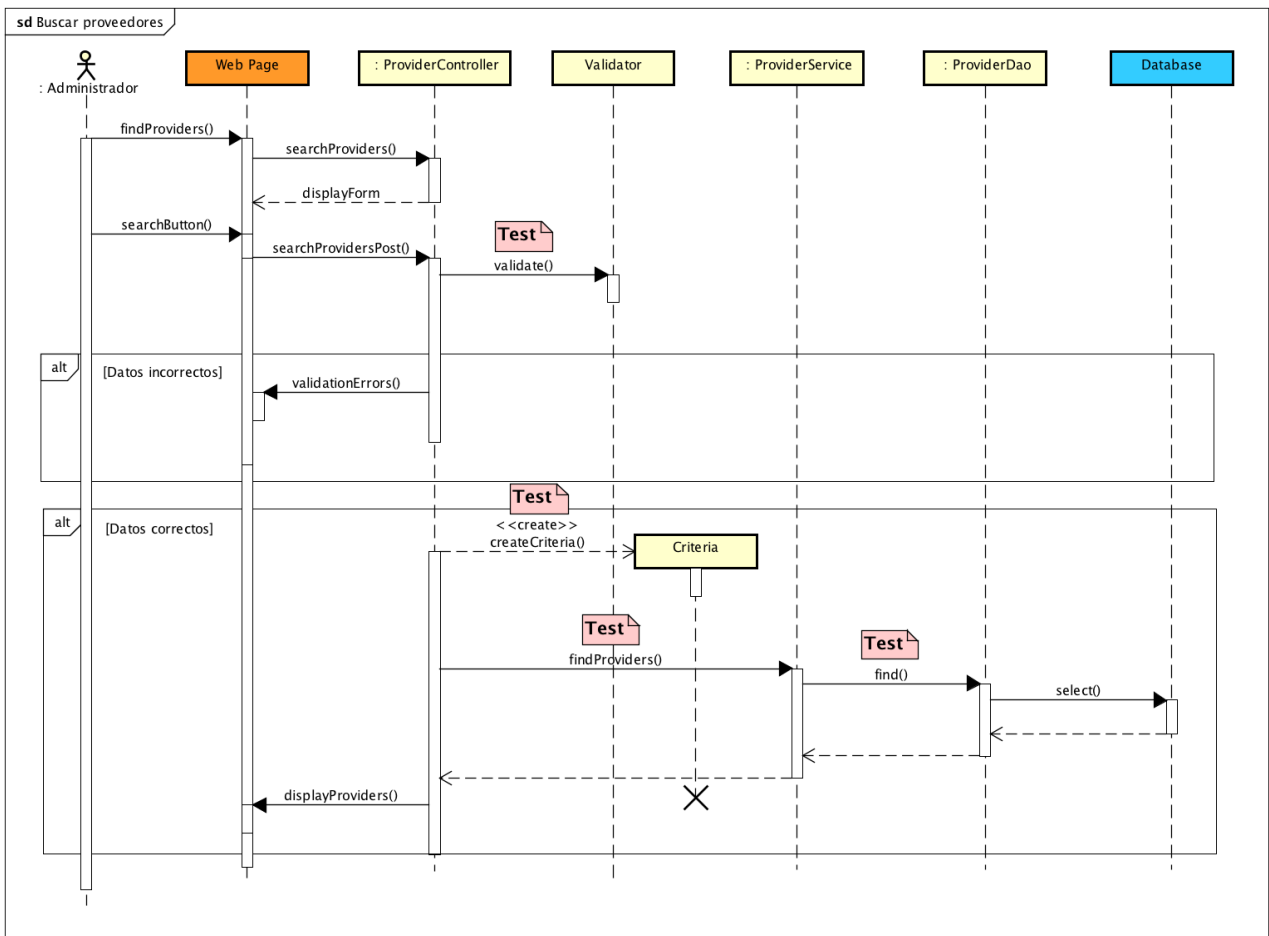


Figura 82 - Diagrama secuencia SD-018

## 4.8.19 SD-019: Editar proveedor

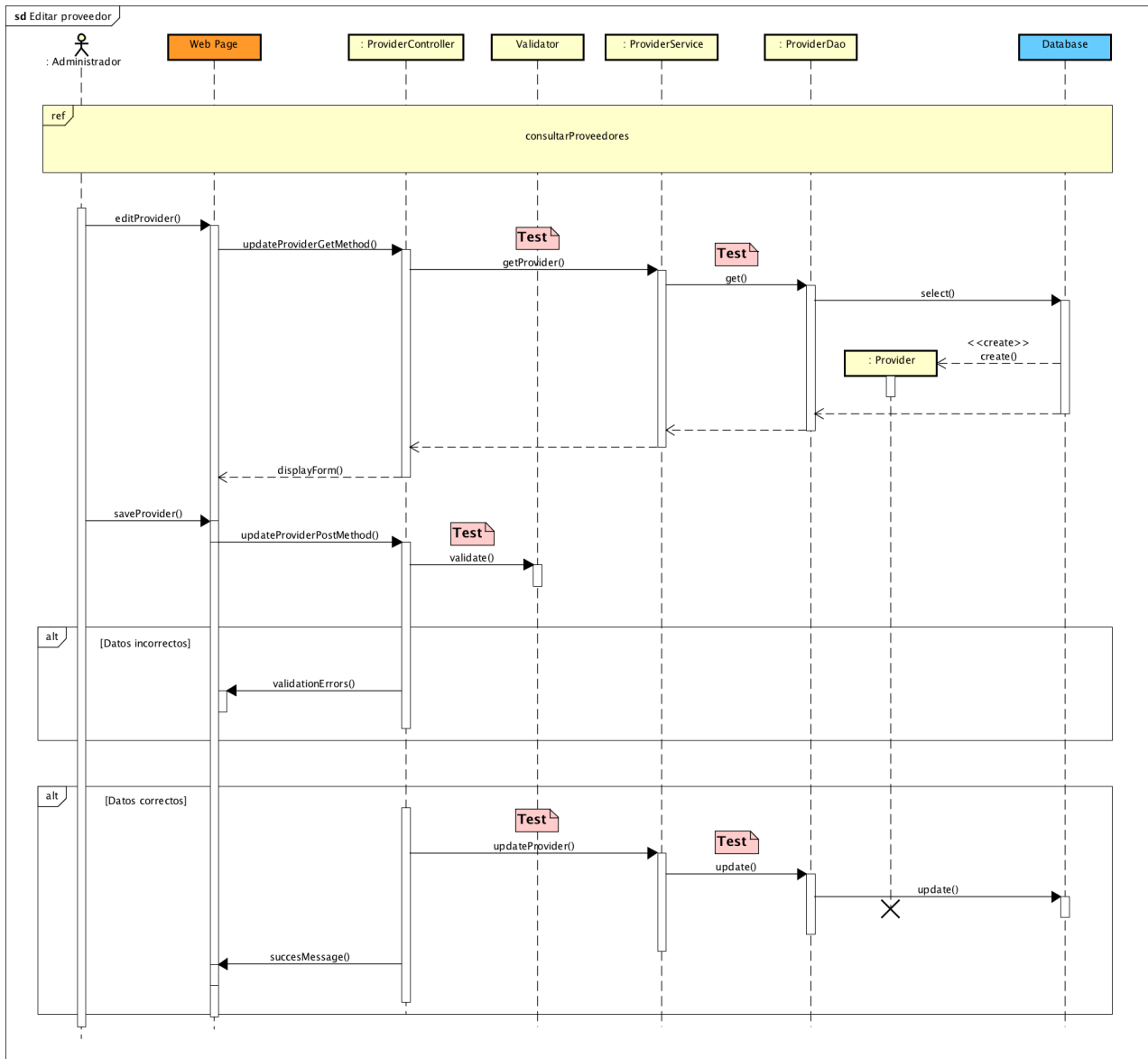


Figura 83 - Diagrama secuencia SD-019

## 4.8.20 SD-020: Añadir empleado

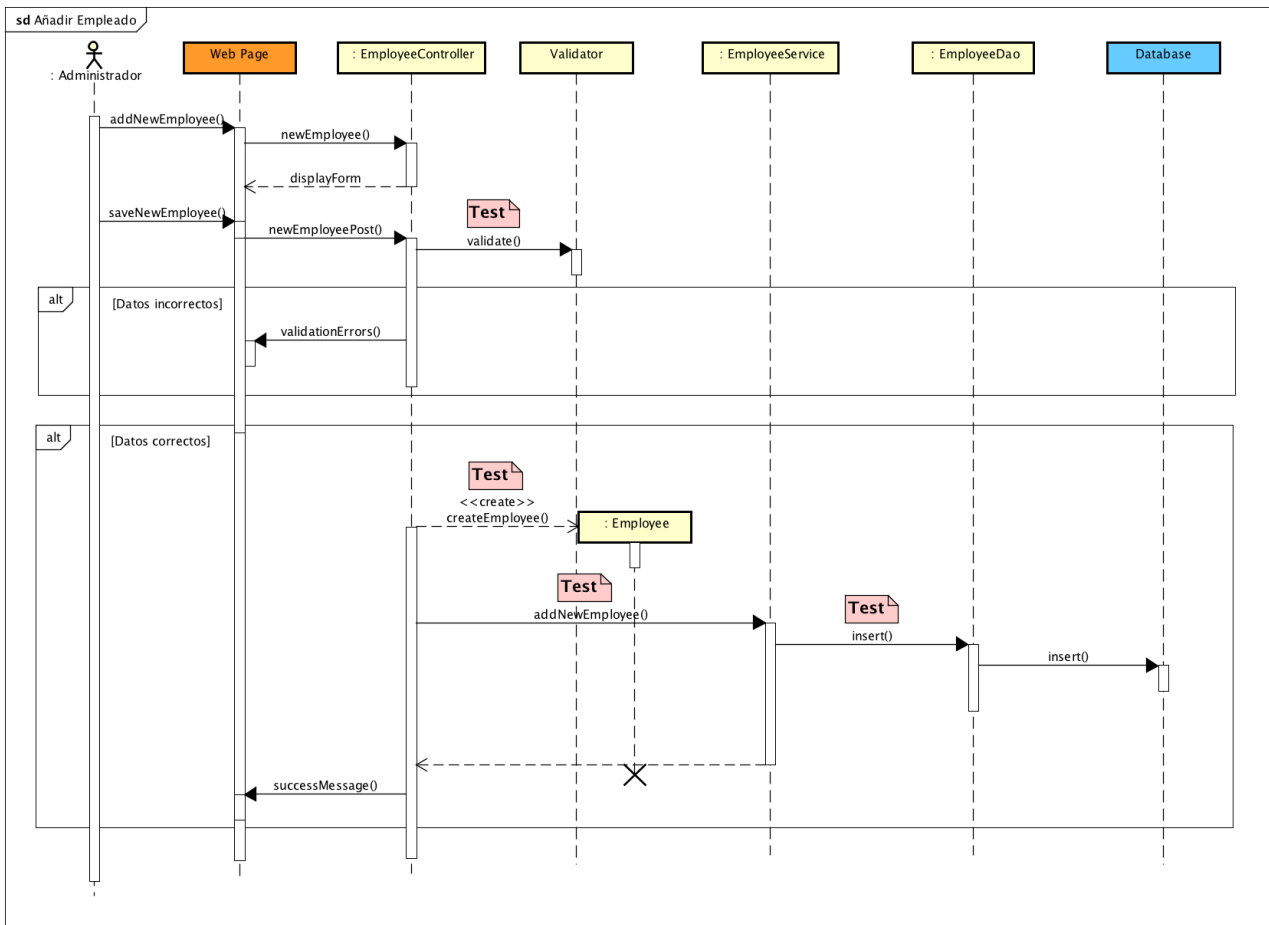


Figura 84 - Diagrama secuencia SD-020

## 4.8.21 SD-021: Consultar empleados

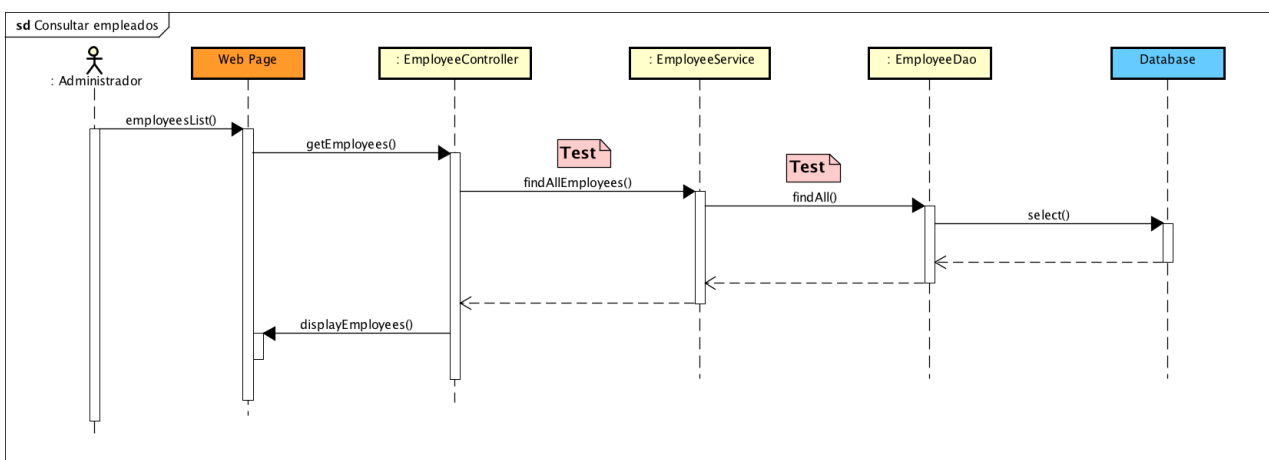


Figura 85 - Diagrama secuencia SD-022

### 4.8.22 SD-022: Eliminar empleado

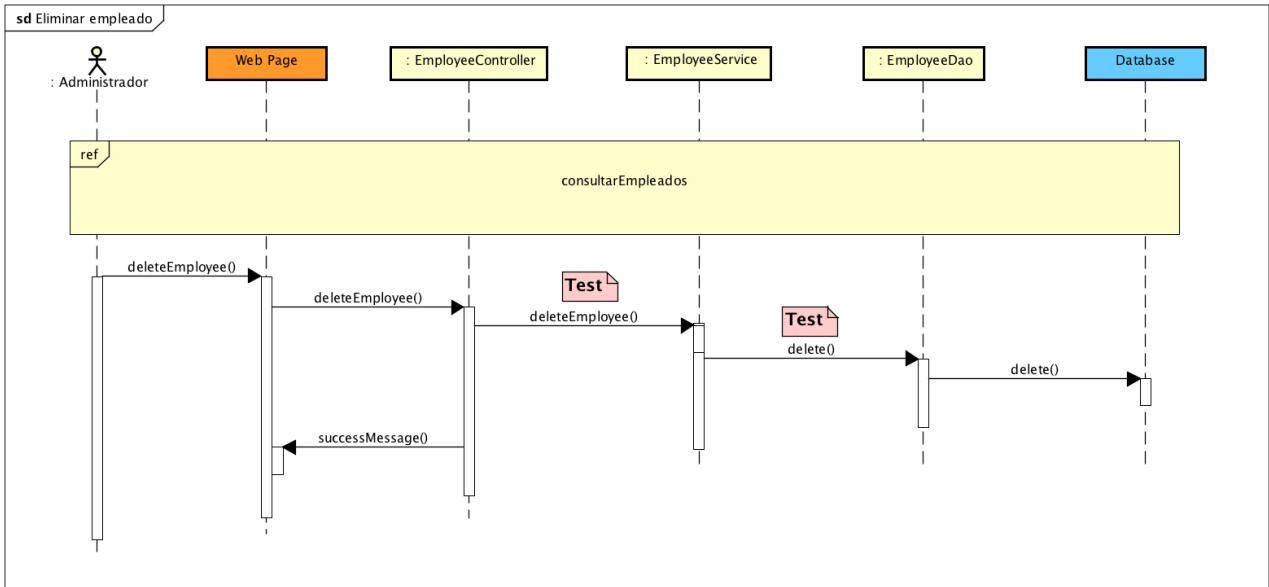


Figura 86 - Diagrama secuencia SD-021

### 4.8.23 SD-023: Buscar empleados

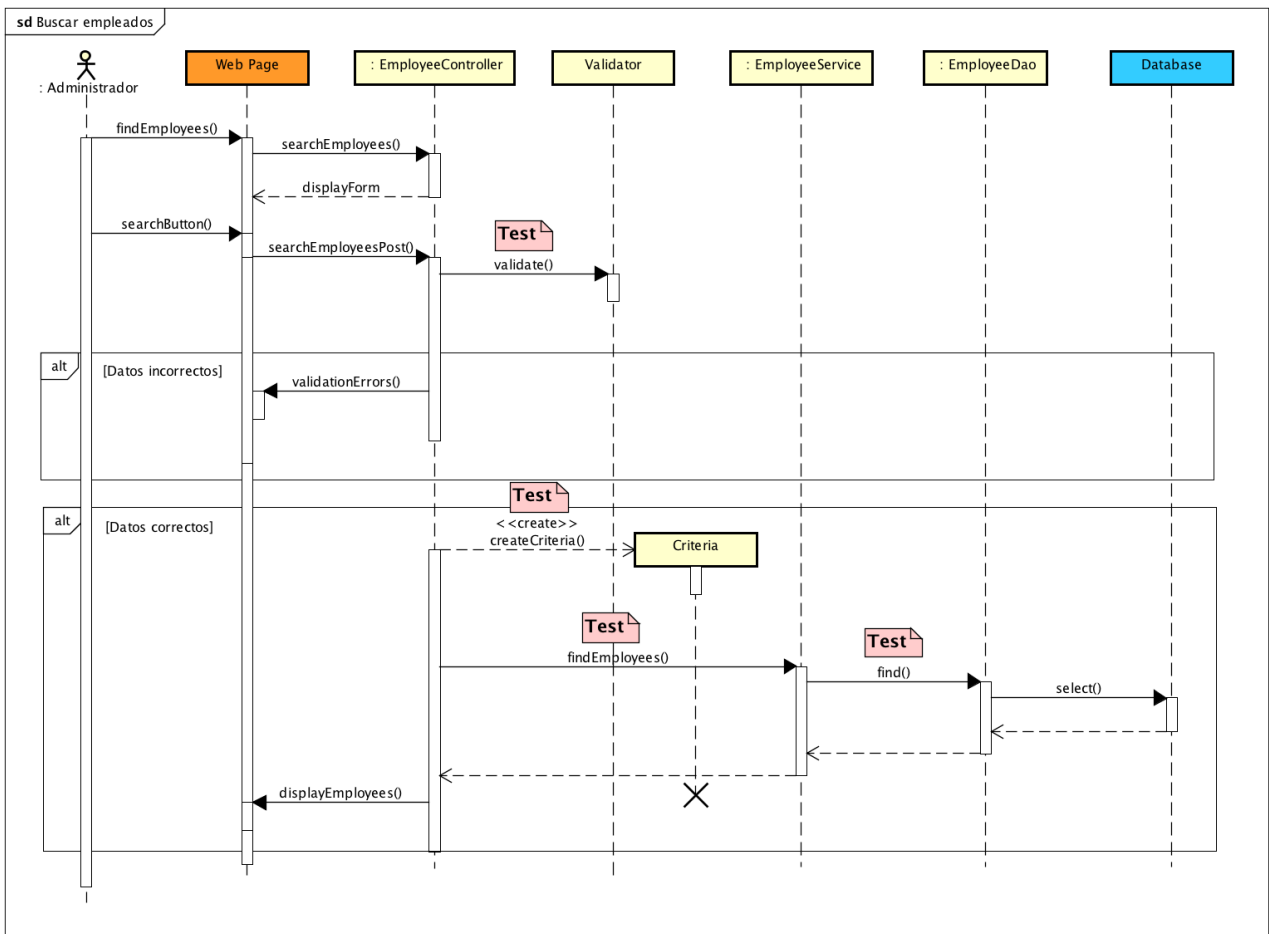


Figura 87 - Diagrama secuencia SD-023



## 4.8.24 SD-024: Editar empleado

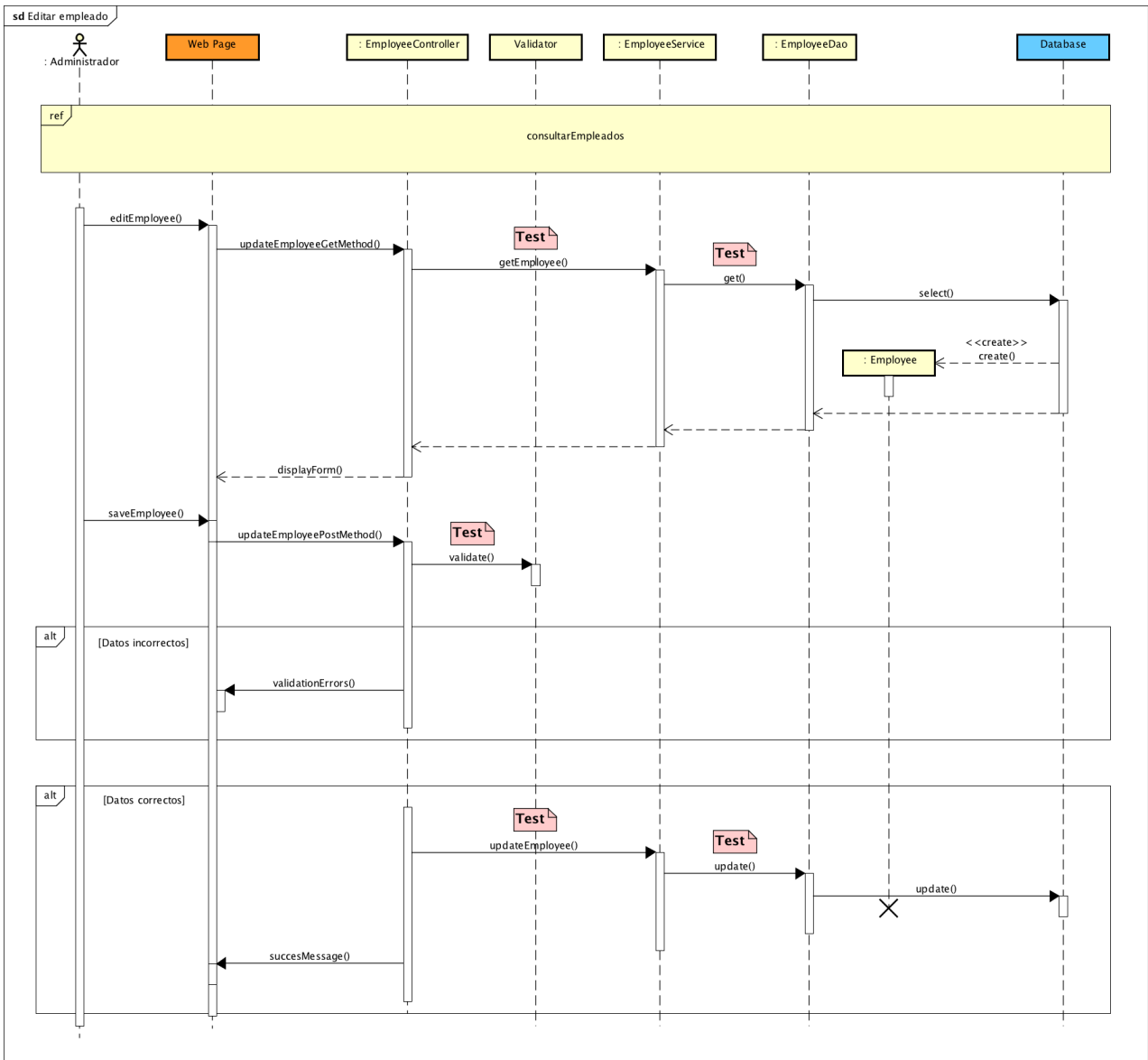


Figura 80 - Diagrama secuencia SD-023

### 4.8.25 SD-025: Añadir usuario

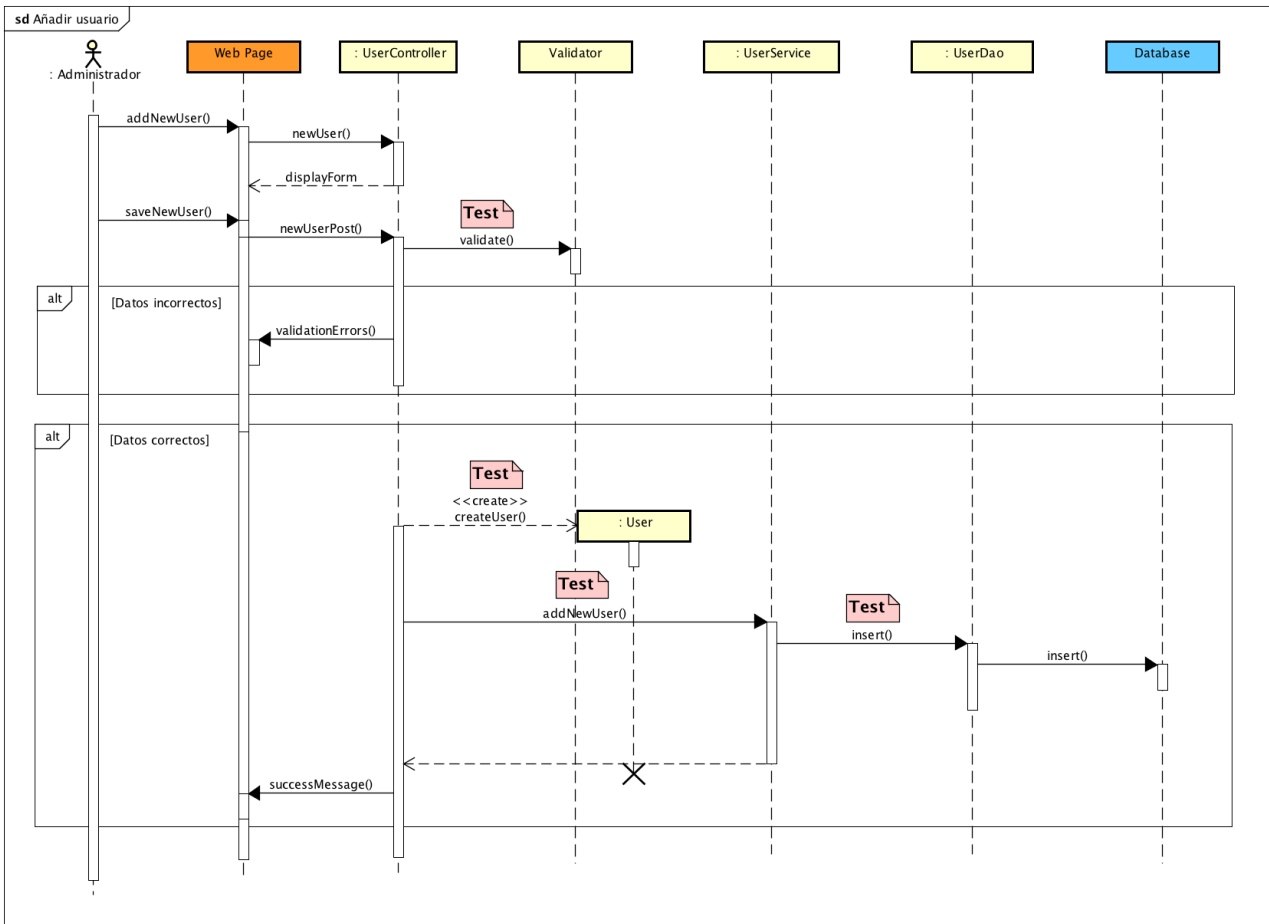


Figura 89 - Diagrama secuencia SD-024

### 4.8.26 SD-026: Consultar Usuarios

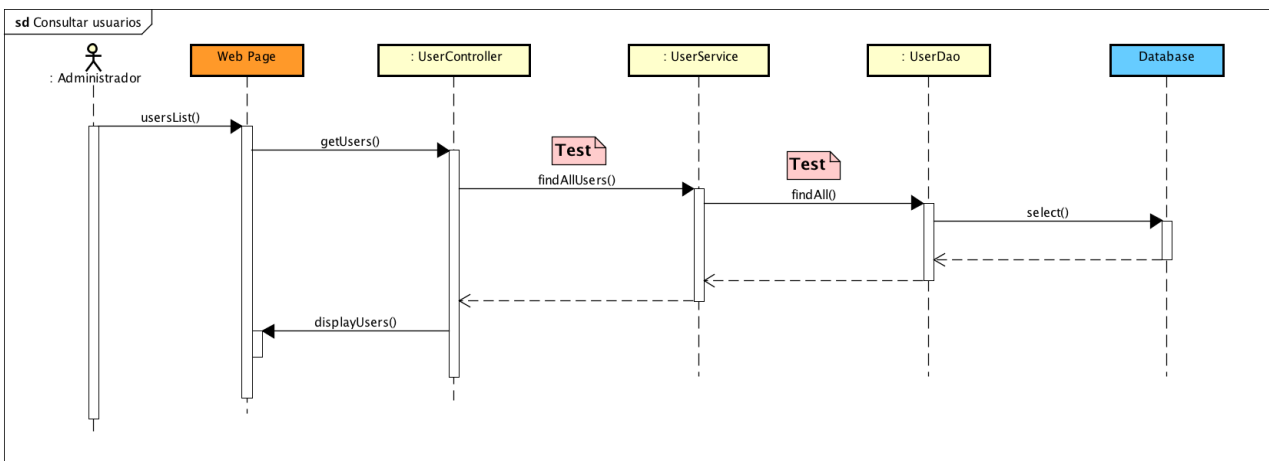


Figura 90 - Diagrama secuencia SD-025

#### 4.8.27 SD-027: Eliminar usuario

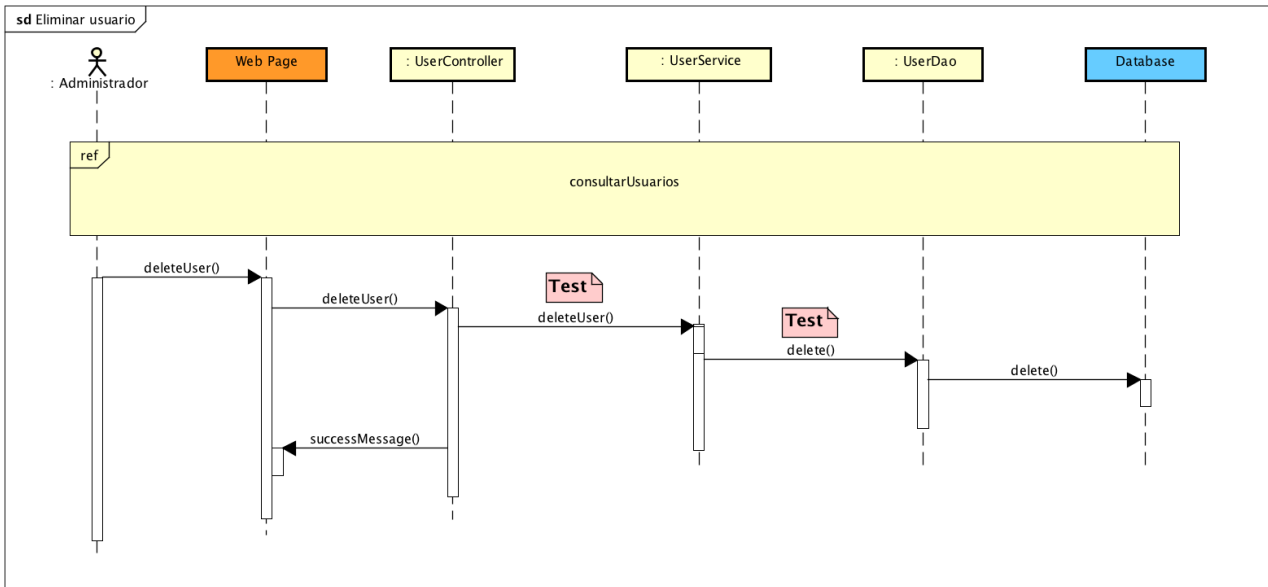


Figura 91 - Diagrama secuencia SD-026

#### 4.8.28 SD-028: Buscar ventas

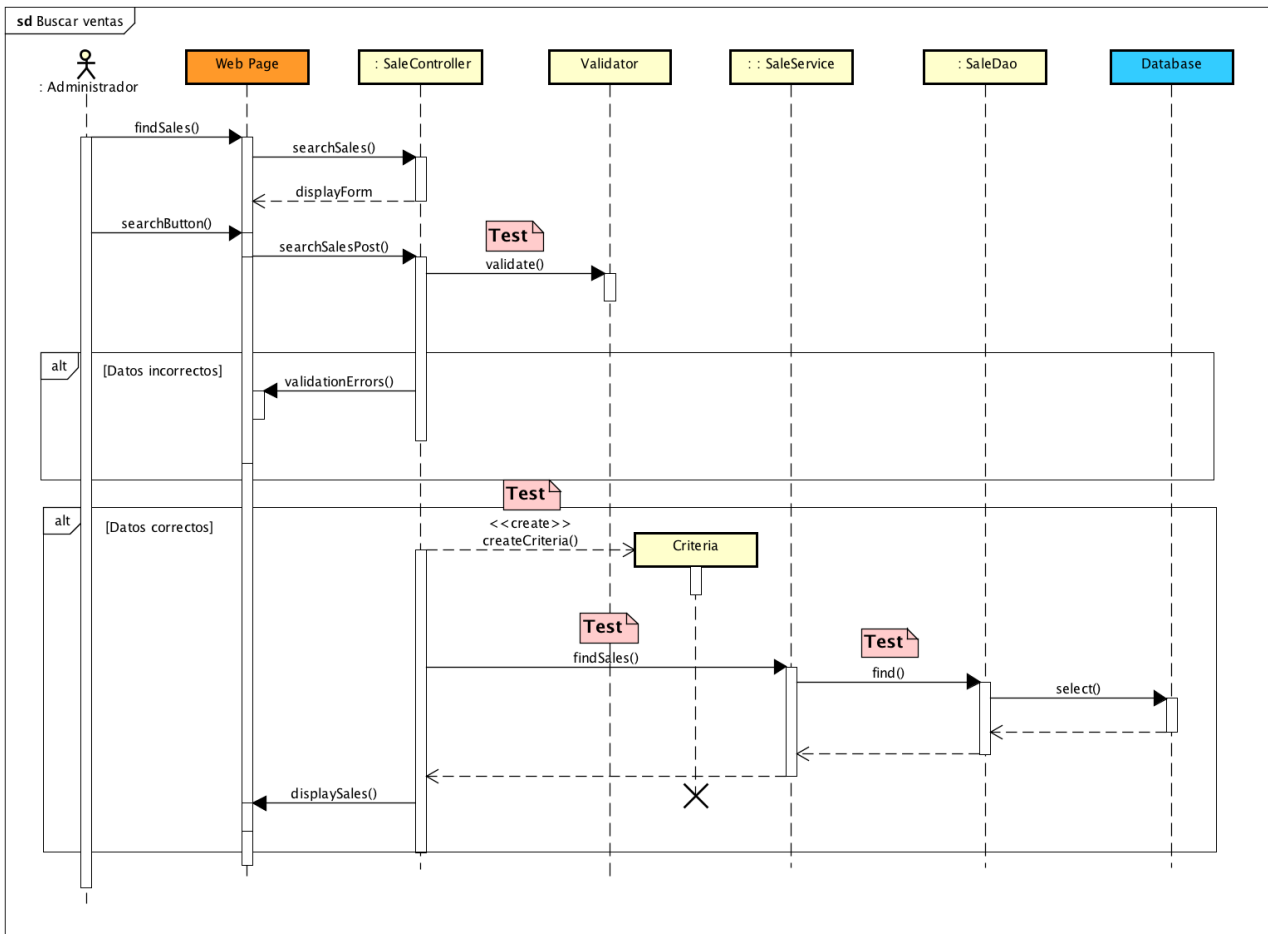


Figura 92 - Diagrama secuencia SD-028



## **5. Pruebas**



## 5. Pruebas

### 5.1 Introducción

El papel que juegan las pruebas en el desarrollo de software es el de detectar los fallos y errores que se pueden producir en un sistema.

Estas pruebas han de empezar a realizarse ya en las primeras etapas del desarrollo de software, pues cuanto antes se detecte la presencia de un posible fallo y/o error, menor será el esfuerzo empleado para corregirlo.

El proceso de pruebas realizado para el desarrollo del proyecto se divide en dos grupos:

- El conjunto de **pruebas unitarias**: realizadas sobre la capa de negocio y capa de persistencia de la aplicación. Gracias a estas pruebas se puede garantizar que cada unidad funcional, cada 'pieza', tiene el comportamiento esperado.
- El conjunto de **pruebas funcionales**: realizadas sobre la aplicación en su conjunto, como si de un usuario final se tratara. El objetivo de estas pruebas es comprobar que la aplicación se comporta como se espera, y cada funcionalidad se ejecuta sin fallos y/o errores.

### 5.2 Pruebas unitarias

#### 5.2.1 Introducción a JUnit 4

Las pruebas se implementan en clases POJO sin ninguna particularidad.

En el caso de Maven, estas clases se ubicarán en `/src/test/java` y los recursos que sean específicos de las pruebas estarán en el directorio `/src/test/resources`.

Al ejecutarse una clase con JUnit se invocarán los métodos con las siguientes anotaciones:

- **@Test**: cada método que implemente una prueba.
- **@Before**: se invoca siempre antes de la ejecución de cada `@Test`.
- **@After**: se invoca siempre después de la ejecución de cada `@Test`.
- **@BeforeClass**: método estático. Se invoca una única vez al ejecutarse la clase antes de cualquier método `@Test` o `@Before`. Por ejemplo, para arrancar una base de datos embebida como HSQLDB, evitando el coste de realizar esta operación cada vez que se ejecute un test.
- **@AfterClass**: método estático. Equivalente al anterior, pero ahora el método es el último que se invoca al ejecutarse la clase.
- **@Ignore**: los tests marcados con esta anotación no se ejecutarán.

## 5.2.2 Metodología

Las pruebas unitarias permiten la validación del correcto funcionamiento de cada 'pieza' del código de forma aislada.

En este proyecto, se han limitado este tipo de pruebas a las clases que implementan la capa de negocio y la capa de persistencia.

La idea reside en especificar varios casos de prueba por cada método, de forma que podamos garantizar que se cumple el comportamiento deseado, independientemente del resto de métodos.

Mediante las pruebas sobre consultas de lectura de datos, se han intentado cubrir dos tipos de situaciones:

En primer lugar, la búsqueda a partir de datos nulos o inválidos, es decir, datos que no se encuentran en la base de datos. Para este tipos de casos se comprueba que las operaciones devuelvan un resultado vacío, y sin ningún tipo de error.

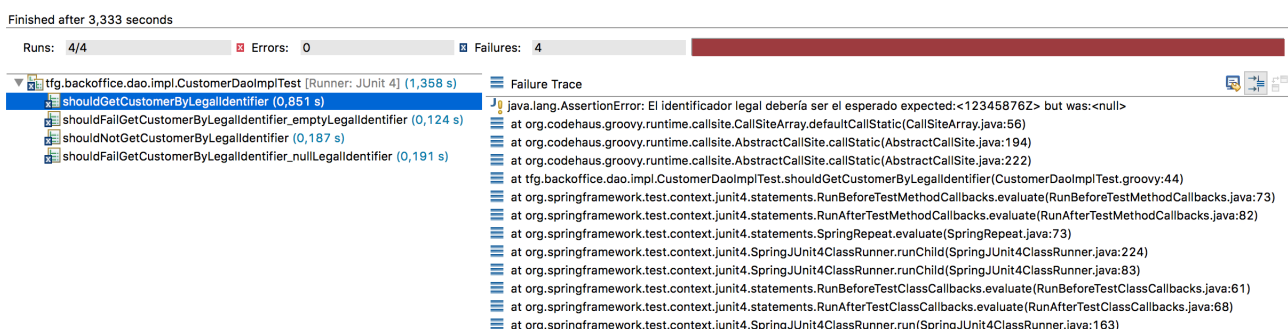
En segundo lugar, la búsqueda a partir de datos correctos. En este caso se debe comprobar que el resultado obtenido es el esperado, y se ajusta tanto a la cantidad de objetos deseados como a la información contenida en cada uno de estos.

Lo primero que hacemos a la hora de empezar a desarrollar la 'pieza', es escribir una prueba.

Esto nos obliga a pensar acerca de cómo se desea que el código se use, se comporte, e interactúe con otras piezas, en caso de que sea necesario.

Tras una primera definición de las nuevas pruebas a implementar, se ejecutan las pruebas.

Puesto que la funcionalidad no está aún implementada, se obtendrá como resultado un fallo para cada una de las pruebas (pruebas en rojo).



```
Finished after 3,333 seconds
Runs: 4/4 Errors: 0 Failures: 4
tfg.backoffice.dao.impl.CustomerDaoImplTest [Runner: JUnit 4] (1,358 s)
  shouldGetCustomerByLegalIdentifier (0,851 s)
  shouldFailGetCustomerByLegalIdentifier_emptyLegalIdentifier (0,124 s)
  shouldNotGetCustomerByLegalIdentifier (0,187 s)
  shouldFailGetCustomerByLegalIdentifier_nullLegalIdentifier (0,191 s)
Failure Trace
  java.lang.AssertionError: El identificador legal debería ser el esperado expected:<-12345876Z> but was:<null>
  at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallStatic(CallSiteArray.java:56)
  at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callStatic(AbstractCallSite.java:194)
  at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callStatic(AbstractCallSite.java:222)
  at tfg.backoffice.dao.impl.CustomerDaoImplTest.shouldGetCustomerByLegalIdentifier(CustomerDaoImplTest.groovy:44)
  at org.springframework.test.context.junit4.statements.RunBeforeTestMethodCallbacks.evaluate(RunBeforeTestMethodCallbacks.java:73)
  at org.springframework.test.context.junit4.statements.RunAfterTestMethodCallbacks.evaluate(RunAfterTestMethodCallbacks.java:82)
  at org.springframework.test.context.junit4.statements.SpringRepeat.evaluate(SpringRepeat.java:73)
  at org.springframework.test.context.junit4.SpringJUnit4ClassRunner.runChild(SpringJUnit4ClassRunner.java:224)
  at org.springframework.test.context.junit4.SpringJUnit4ClassRunner.runChild(SpringJUnit4ClassRunner.java:83)
  at org.springframework.test.context.junit4.statements.RunBeforeTestClassCallbacks.evaluate(RunBeforeTestClassCallbacks.java:61)
  at org.springframework.test.context.junit4.statements.RunAfterTestClassCallbacks.evaluate(RunAfterTestClassCallbacks.java:68)
  at org.springframework.test.context.junit4.SpringJUnit4ClassRunner.run(SpringJUnit4ClassRunner.java:163)
```

Figura 93 - Ejecución fallida de pruebas



El siguiente paso, según el ciclo de desarrollo guiado por pruebas (TDD), consiste en desarrollar el código necesario para que la validación definida en cada una de ellas se cumpla (pruebas en verde).

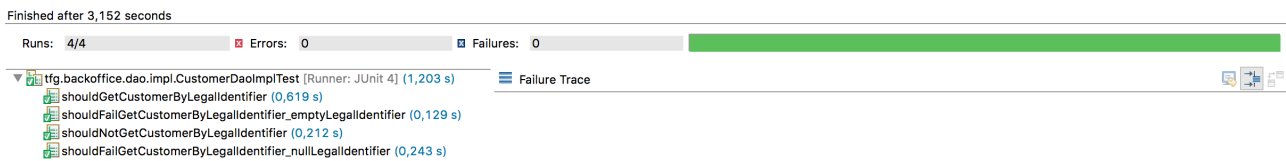


Figura 94 - Ejecución de pruebas con éxito

Una vez implementada la funcionalidad necesaria, la ejecución de las pruebas pasa en verde, lo que indica que el comportamiento deseado se cumple y las condiciones se validan de forma correcta.

El último proceso en este ciclo TDD se trata de la refactorización del código implementado, lo que nos permite mejorar la estructura del código para eliminar redundancia y bad-smells.

### 5.2.3 Estructura de las pruebas

Se define una estructura para cada una de las pruebas unitarias de la aplicación.

```
@Test
void unitTest() {

    // GIVEN

    // WHEN

    // THEN

}
```

- **GIVEN:** se definen los objetos, variables o constantes que se utilizarán en la prueba.
- **WHEN:** se realiza la llamada al método que se quiere probar.
- **THEN:** se realizan las comprobaciones (mediante asertos) para asegurar que los datos o el resultado obtenido es el esperado.

## 5.2.4 Pruebas de la capa de persistencia

A continuación se muestra un ejemplo de prueba unitaria en la capa de persistencia para probar el método de inserción de un nuevo producto:

```
@Test
void shouldInsertProduct() {

    // GIVEN
    def previousRows = countRowsInTable("PRODUCT");
    def product = new Product(
        name: "Producto",
        description: "Descripcion de producto",
        category: new tfg.model.Category(id:101L),
        code: "PROD_CODE",
        price: 15.25,
        priceType: PriceTypeEnum.PRECIO_UNITARIO,
        active: Boolean.TRUE
    )

    // WHEN
    productDao.insert(product)

    // THEN
    assertEquals('Debería haberse insertado un nuevo producto',
        previousRows+1, countRowsInTable('PRODUCT'))
    def productInserted = jdbcTemplate.queryForMap("SELECT * FROM PRODUCT
        WHERE ID = ?", product.id)
    assertEquals("El nombre del producto debería ser el esperado",
        product.name, productInserted["NAME"])
    assertEquals("La descripción del producto debería ser la esperada",
        product.description, productInserted["DESCRIPTION"])
    assertEquals("La categoría del producto debería ser la esperada",
        product.category.id, productInserted["CATEGORY_ID"])
    assertEquals("El código del producto debería ser el esperado",
        product.code, productInserted["CODE"])
    assertEquals("El tipo de precio de venta del producto debería ser el
        esperado", product.priceType.toString(),
        productInserted["PRICE_TYPE"])
    assertEquals("El importe de precio de venta del producto debería ser el
        esperado", product.price, productInserted["PRICE"], 0)
    assertEquals("El flag activo del producto debería ser el esperado",
        product.active, productInserted["ACTIVE"])
    assertEquals("El flag deleted del producto debería ser el esperado",
        Boolean.FALSE, productInserted["DELETED"])
}
```

Mediante los asertos (assertEquals) se comprueba que los valores añadidos a la tabla PRODUCT son los valores esperados, comparándolos con el objeto en cuestión que se quiere insertar.

### 5.2.5 Pruebas de la capa de servicios (lógica de negocio)

Para las pruebas de la capa de servicio es necesaria la utilización de mocks, para simular el comportamiento del resto de ‘piezas’ que interactúan en el método que se quiere probar.

En el desarrollo de este proyecto se utiliza el framework Mockito, que permite realizar estas operaciones de forma muy sencilla y rápida.

A continuación se muestra un ejemplo de prueba unitaria en la capa de servicio para probar el método de inserción de un nuevo producto:

```
@Before
void before() {
    customerService = new CustomerServiceImpl(customerDao:
        Mockito.mock(CustomerDao.class))
}

@Test
void shouldAddNewCustomer() {

    // GIVEN
    def customer = new Customer()

    // WHEN
    customerService.addNewCustomer(customer)

    // THEN
    verify(customerService.customerDao).insert(customer)
}
```

Mediante la anotación ‘verify’ comprobamos que la ejecución del método realiza la llamada al método insert del DAO correspondiente (customerDao), y con el parámetro esperado (customer).

Pueden existir situaciones en las que se quiera probar que la ejecución de un método con determinados parámetros o una determinada casuística provoca el lanzamiento de una excepción.

Un ejemplo de esta situación puede verse en el siguiente fragmento:

```
@Test(expected = IllegalArgumentException)
void shouldFailAddNewCustomer_nullCustomer() {

    // GIVEN
    def customer = null

    // WHEN
    customerService.addNewCustomer(customer)
}
```

Mediante la etiqueta ‘expected’ le indicamos a la prueba que para esa ejecución se espera el lanzamiento de una determinada excepción, especificada a continuación.

## 5.2.6 Listado de pruebas unitarias

Clase	Método	Descripción	Resultado
CustomerService	addNewCustomer	Se llama al DAO con los parámetros esperados	OK
	deleteCustomer	Se llama al DAO con los parámetros esperados	OK
	updateCustomer	Se llama al DAO con los parámetros esperados	OK
	getCustomer	Se llama al DAO con los parámetros esperados	OK
	findAllCustomers	Se llama al DAO con los parámetros esperados	OK
	findCustomer	Se llama al DAO con los parámetros esperados	OK
ProductService	addNewProduct	Se llama al DAO con los parámetros esperados	OK
	deleteProduct	Se llama al DAO con los parámetros esperados	OK
	updateProduct	Se llama al DAO con los parámetros esperados	OK
	getProduct	Se llama al DAO con los parámetros esperados	OK
	findAllProducts	Se llama al DAO con los parámetros esperados	OK
	findProduct	Se llama al DAO con los parámetros esperados	OK
ProviderService	addNewProvider	Se llama al DAO con los parámetros esperados	OK
	deleteProvider	Se llama al DAO con los parámetros esperados	OK
	updateProvider	Se llama al DAO con los parámetros esperados	OK
	getProvider	Se llama al DAO con los parámetros esperados	OK
	findAllProviders	Se llama al DAO con los parámetros esperados	OK

Clase	Método	Descripción	Resultado
	findProvider	Se llama al DAO con los parámetros esperados	OK
EmployeeService	addNewEmployee	Se llama al DAO con los parámetros esperados	OK
	deleteEmployee	Se llama al DAO con los parámetros esperados	OK
	updateEmployee	Se llama al DAO con los parámetros esperados	OK
	getEmployee	Se llama al DAO con los parámetros esperados	OK
	findAllEmployees	Se llama al DAO con los parámetros esperados	OK
	findEmployee	Se llama al DAO con los parámetros esperados	OK
UserService	addNewUser	Se llama al DAO con los parámetros esperados	OK
	deleteUser	Se llama al DAO con los parámetros esperados	OK
	updateUser	Se llama al DAO con los parámetros esperados	OK
	findAllUsers	Se llama al DAO con los parámetros esperados	OK
CustomerDao	get	Se obtiene el cliente esperado con sus datos correspondientes	OK
	insert	Se inserta el cliente en base de datos con los datos esperados	OK
	delete	Se elimina el cliente de base de datos (flag DELETED)	OK
	update	Se actualizan los datos del cliente en base de datos	OK
	find	Se obtienen los clientes esperados con sus datos correspondientes	OK
	findAll	Se obtienen todos los clientes con sus datos correspondientes	OK

Clase	Método	Descripción	Resultado
ProductDao	get	Se obtiene el producto esperado con sus datos correspondientes	OK
	insert	Se inserta el producto en base de datos con los datos esperados	OK
	delete	Se elimina el producto de base de datos (flag DELETED)	OK
	update	Se actualizan los datos del producto en base de datos	OK
	find	Se obtienen los productos esperados con sus datos correspondientes	OK
	findAll	Se obtienen todos los productos con sus datos correspondientes	OK
ProviderDao	get	Se obtiene el proveedor esperado con sus datos correspondientes	OK
	insert	Se inserta el proveedor en base de datos con los datos esperados	OK
	delete	Se elimina el proveedor de base de datos (flag DELETED)	OK
	update	Se actualizan los datos del proveedor en base de datos	OK
	find	Se obtienen los proveedores esperados con sus datos correspondientes	OK
	findAll	Se obtienen todos los proveedores con sus datos correspondientes	OK
EmployeeDao	get	Se obtiene el empleado esperado con sus datos correspondientes	OK
	insert	Se inserta el empleado en base de datos con los datos esperados	OK
	delete	Se elimina el empleado de base de datos (flag DELETED)	OK
	update	Se actualizan los datos del empleado en base de datos	OK

Clase	Método	Descripción	Resultado
	find	Se obtienen los empleados esperados con sus datos correspondientes	OK
	findAll	Se obtienen todos los empleados con sus datos correspondientes	OK
	get	Se obtiene el usuario esperado con sus datos correspondientes	OK
	insert	Se inserta el usuario en base de datos con los datos esperados	OK
UserDao	delete	Se elimina el usuario de base de datos (flag DELETED)	OK
	update	Se actualizan los datos del usuario en base de datos	OK
	findAll	Se obtienen todos los usuarios con sus datos correspondientes	OK
CustomerValidator	validate	Se validan correctamente los campos del cliente	OK
ProductValidator	validate	Se validan correctamente los campos del producto	OK
ProviderValidator	validate	Se validan correctamente los campos del proveedor	OK
EmployeeValidator	validate	Se validan correctamente los campos del empleado	OK
UserValidator	validate	Se validan correctamente los campos del usuario	OK

## 5.3 Pruebas funcionales

Este tipo de pruebas consisten en probar las distintas funcionalidades de la aplicación, como si de un usuario final se tratara.

De esta forma se consiguen realizar pruebas de integración de las distintas ‘piezas’, pudiendo verificar el correcto funcionamiento, o la presencia de posibles fallos y/o errores.

A continuación se detallan las pruebas funcionales realizadas.

Pruebas funcionales relacionadas con **clientes**:

Descripción	Resultado esperado	Resultado
Crear/editar cliente completando todos los campos correctamente	El cliente se inserta/actualiza en base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Crear/editar cliente con campos obligatorios sin completar	Se muestran los errores de validación de campos obligatorios	OK

Descripción	Resultado esperado	Resultado
Crear/editar cliente con error de formato en alguno de los campos	Se muestran los errores de validación de formato	OK

Descripción	Resultado esperado	Resultado
Crear/editar cliente con un dni existente	Se muestra el error de cliente existente	OK

Descripción	Resultado esperado	Resultado
Eliminar cliente	El cliente se elimina de base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Ver listado de clientes	Se muestra el listado de todos los clientes	OK

Descripción	Resultado esperado	Resultado
Buscar clientes sin completar ningún campo de búsqueda	Se muestra el mensaje para completar campos de búsqueda	OK



Descripción	Resultado esperado	Resultado
Buscar clientes completando uno o varios campos de búsqueda	Se muestra el listado de clientes que cumplen los criterios de búsqueda	OK

Pruebas funcionales relacionadas con **productos**:

Descripción	Resultado esperado	Resultado
Crear/editar producto completando todos los campos correctamente	El producto se inserta/actualiza en base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Crear/editar producto con campos obligatorios sin completar	Se muestran los errores de validación de campos obligatorios	OK

Descripción	Resultado esperado	Resultado
Crear/editar producto con error de formato en alguno de los campos	Se muestran los errores de validación de formato	OK

Descripción	Resultado esperado	Resultado
Crear/editar producto con un código de producto existente	Se muestra el error de código de producto existente	OK

Descripción	Resultado esperado	Resultado
Eliminar producto	El producto se elimina de base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Ver listado de productos	Se muestra el listado de todos los productos	OK

Descripción	Resultado esperado	Resultado
Buscar productos sin completar ningún campo de búsqueda	Se muestra el mensaje para completar campos de búsqueda	OK

Descripción	Resultado esperado	Resultado
Buscar productos completando uno o varios campos de búsqueda	Se muestra el listado de productos que cumplen los criterios de búsqueda	OK

Pruebas funcionales relacionadas con **proveedores**:

Descripción	Resultado esperado	Resultado
Crear/editar proveedor completando todos los campos correctamente	El proveedor se inserta/actualiza en base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Crear/editar proveedor con campos obligatorios sin completar	Se muestran los errores de validación de campos obligatorios	OK

Descripción	Resultado esperado	Resultado
Crear/editar proveedor con error de formato en alguno de los campos	Se muestran los errores de validación de formato	OK

Descripción	Resultado esperado	Resultado
Crear/editar proveedor con un código de proveedor existente	Se muestra el error de código de proveedor existente	OK

Descripción	Resultado esperado	Resultado
Eliminar proveedor	El proveedor se elimina de base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Ver listado de proveedores	Se muestra el listado de todos los proveedores	OK

Descripción	Resultado esperado	Resultado
Buscar proveedores sin completar ningún campo de búsqueda	Se muestra el mensaje para completar campos de búsqueda	OK

Descripción	Resultado esperado	Resultado
Buscar proveedores completando uno o varios campos de búsqueda	Se muestra el listado de proveedores que cumplen los criterios de búsqueda	OK

Pruebas funcionales relacionadas con **empleados**:

Descripción	Resultado esperado	Resultado
Crear/editar empleado con campos obligatorios sin completar	Se muestran los errores de validación de campos obligatorios	OK

Descripción	Resultado esperado	Resultado
Crear/editar empleado con error de formato en alguno de los campos	Se muestran los errores de validación de formato	OK

Descripción	Resultado esperado	Resultado
Crear/editar empleado con un código de empleado existente	Se muestra el error de código de empleado existente	OK

Descripción	Resultado esperado	Resultado
Eliminar empleado	El empleado se elimina de base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Ver listado de empleados	Se muestra el listado de todos los empleados	OK

Descripción	Resultado esperado	Resultado
Buscar empleados sin completar ningún campo de búsqueda	Se muestra el mensaje para completar campos de búsqueda	OK

Descripción	Resultado esperado	Resultado
Buscar empleados completando uno o varios campos de búsqueda	Se muestra el listado de empleados que cumplen los criterios de búsqueda	OK

Pruebas funcionales relacionadas con **usuarios**:

Descripción	Resultado esperado	Resultado
Crear/editar usuario con error de formato en alguno de los campos	Se muestran los errores de validación de formato	OK

Descripción	Resultado esperado	Resultado
Crear/editar usuario con un código de usuario existente	Se muestra el error de código de usuario existente	OK

Descripción	Resultado esperado	Resultado
Eliminar usuario	El usuario se elimina de base de datos y se muestra mensaje de éxito	OK

Descripción	Resultado esperado	Resultado
Ver listado de usuarios	Se muestra el listado de todos los usuarios	OK

Pruebas funcionales relacionadas con **ventas**:

Descripción	Resultado esperado	Resultado
Completar una venta en el módulo TPV	El carrito de compra se vacía y la venta se inserta en la base de datos	OK

Descripción	Resultado esperado	Resultado
Buscar ventas completando uno o varios de los campos de búsqueda	Se muestra el listado de ventas que cumplen los criterios de búsqueda	OK

Pruebas funcionales relacionadas con **acceso a la aplicación**:

Descripción	Resultado esperado	Resultado
Acceder al sistema con datos erróneos	No se accede al sistema y se muestra mensaje de error al usuario	OK

Descripción	Resultado esperado	Resultado
Acceder al sistema con datos correctos	Se accede correctamente al sistema	OK

## **6. Implementación**



## 6. Implementación

En este apartado se muestra el diagrama de despliegue de la aplicación, así como una descripción de los frameworks utilizados para la implementación de la aplicación.

Se aportarán algunas descripciones y ejemplos de código, para mostrar la forma de llevar a cabo las diferentes partes de la aplicación.

### 6.1 Diagrama de despliegue

A continuación se muestra el diagrama de despliegue de la aplicación:

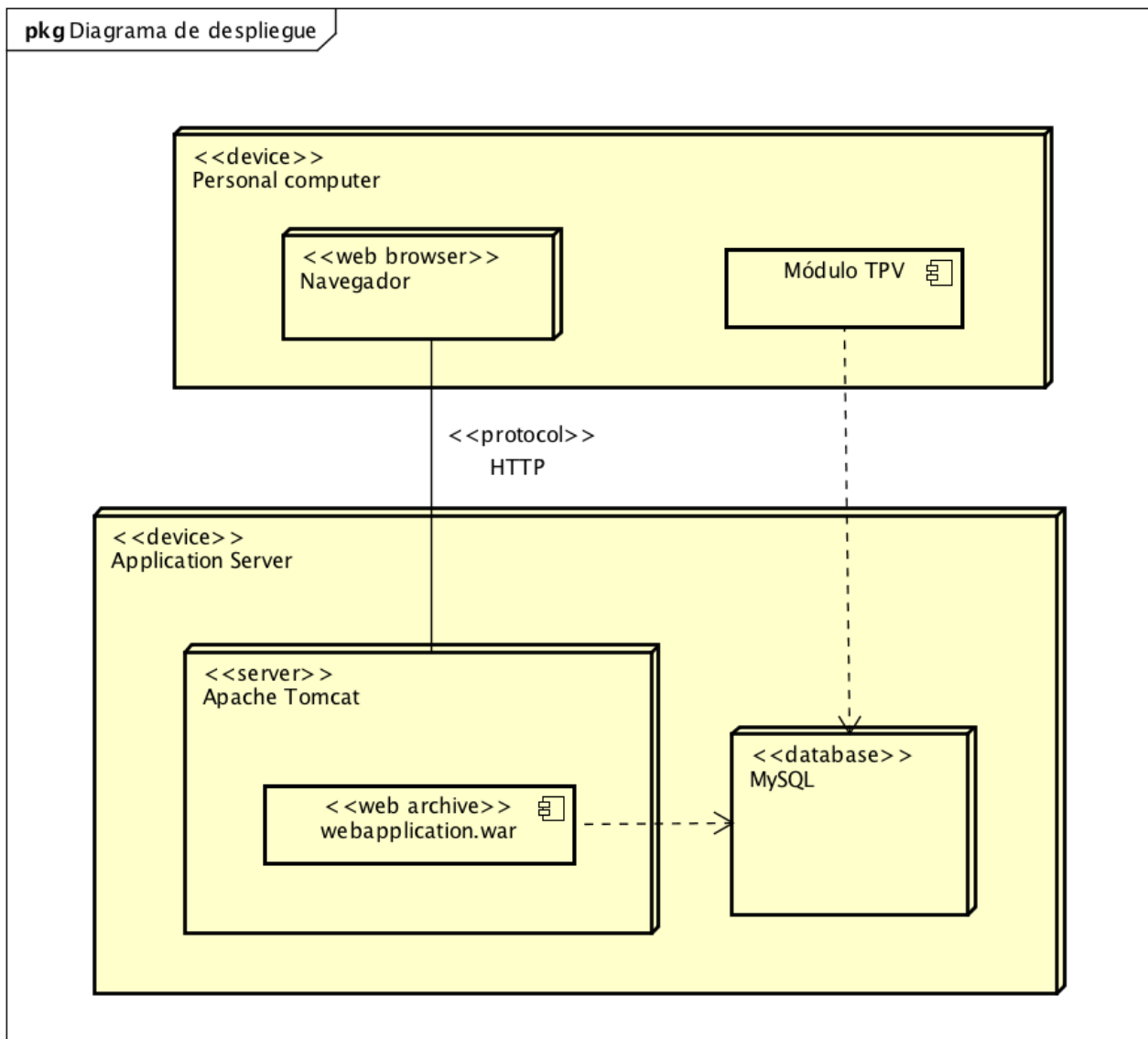


Figura 95 - Diagrama de despliegue

## 6.2 Spring MVC

### 6.2.1 Introducción

El framework de Spring MVC está diseñado en base a un DispatcherServlet, cuya función es 'distribuir' las peticiones entre los distintos controladores de la aplicación.

Los controladores son los responsables de preparar un modelo y elegir el nombre de la vista, pero pueden responder directamente y completar la petición si se desea.

La comunicación básica de estos elementos es la que se muestra en la figura:

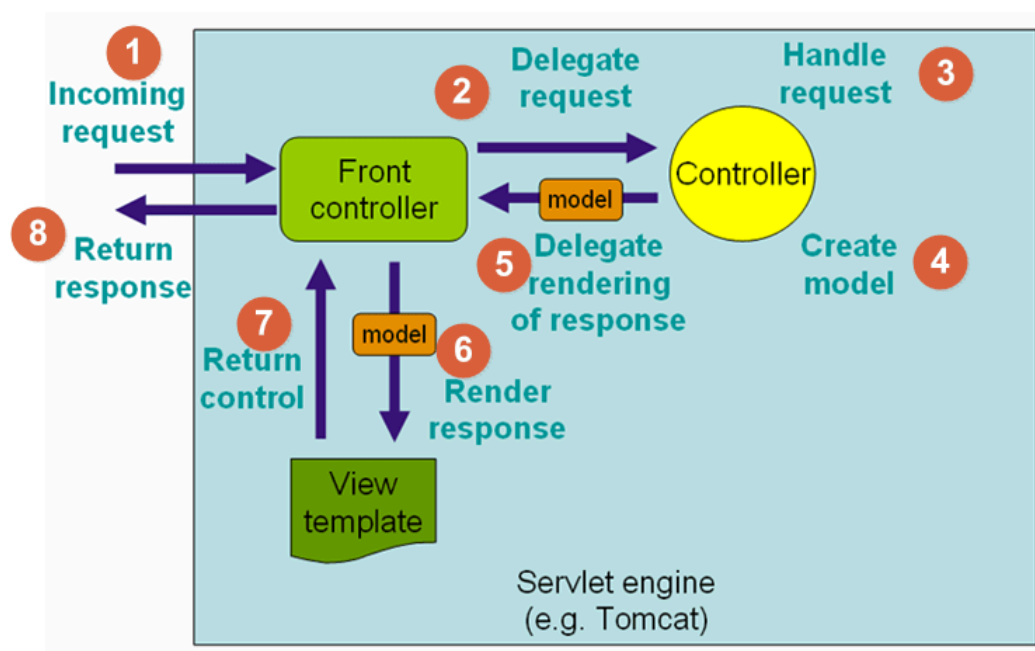


Figura 96 - Diagrama de comunicación entre elementos

Para entender mejor cómo se decide qué controlador va a realizar las acciones y qué vistas se van a utilizar, es necesario entender el concepto de DispatcherServlet.

El DispatcherServlet es el componente que recibe las peticiones HTTP y devuelve las respuestas.

Tras una petición HTTP, el DispatcherServlet realiza las siguientes acciones:

1. Consultar el mapeo de direcciones para llamar al controlador adecuado.
2. El controlador correspondiente recibe la petición mediante el método adecuado. Este método prepara los datos e indica el nombre de la vista al DispatcherServlet.
3. Con ayuda del objeto ViewResolver, el DispatcherServlet escogerá la vista adecuada para la petición.



4. El DispatcherServlet pasa los datos propios de la lógica de negocio a la vista para que se muestre adecuadamente.

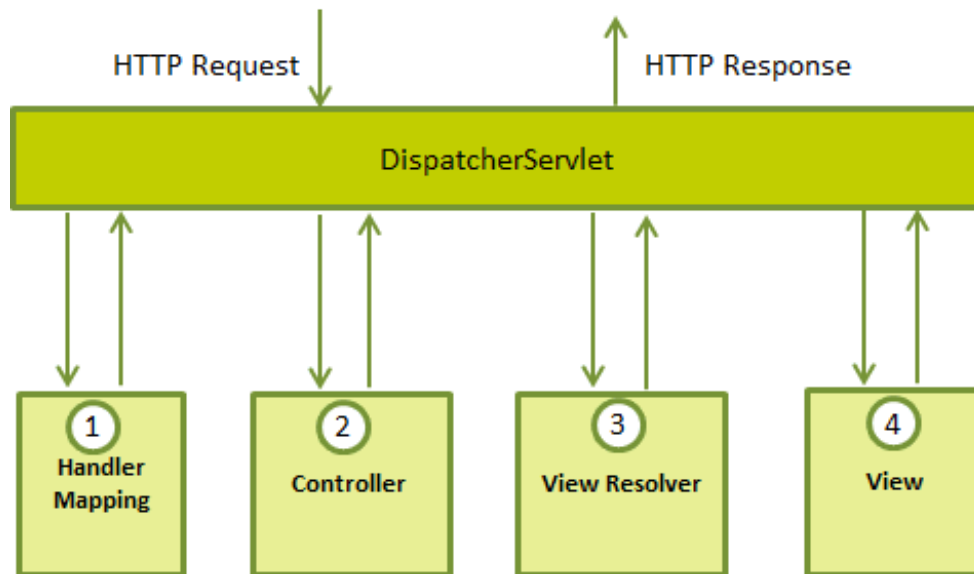


Figura 97 - Diagrama funcionamiento DispatcherServlet

### 6.2.2 Anotaciones

Existen diferentes anotaciones para definir cada uno de los posibles componentes que forman la aplicación.

Entre ellas destacan:

- **@Controller**: Anotación para indicar que el componente se trata de un controlador.
- **@Service**: Indica que se trata de un bean asociado a un servicio, dentro de la capa de lógica de negocio.
- **@Component**: Anotación para beans no específicos, que pueden tener diferentes usos (validadores, clases de configuración, etc).

### 6.2.3 View Resolver

Es el encargado de decidir qué vista debe enviarse dependiendo de un nombre.

Pueden ofrecer características como la resolución de distintas vistas dependiendo de localización u otro tipo de distinciones (internacionalización).

Se considera buena práctica que todas las plantillas se almacenen en el mismo directorio y sean del mismo tipo.

Para simplificar los controladores se puede especificar al ViewResolver el prefijo y el sufijo del nombre del archivo ofrecido.

Este es el motivo por el cual a un ViewResolver normalmente se le da como nombre de la vista tan sólo el nombre del fichero, sin especificar la ruta ni la extensión.

De esta forma, una vez configurados el prefijo y el sufijo que utilizarán los ViewResolver, el controlador de ejemplo sería de esta forma:

```
@RequestMapping(value = "new", method = RequestMethod.GET)
public String newProductGetMethod(Model model, HttpSession session) {

    return "new_product";
}
```

## 6.2.4 JSP

JavaServer Pages (JSP) es una de las muchas tecnologías que se pueden utilizar para renderizar una vista a partir de modelos de Spring. JSP permite generar páginas web dinámicas basadas en documentos estándar de páginas web como HTML o XML.

Al tratarse de una tecnología basada en Java, tiene gran compatibilidad con Spring.

Las páginas JSP son archivos con extensión ‘.jsp’, ejecutadas dentro de contenedores servlet.

Combinan texto plano con código Java insertado entre la apertura ‘<%’ y el cierre ‘%>’.

La salida es generalmente un archivo interpretable por navegadores, combinando el texto plano con el resultado del código Java, encargado de ofrecer el contenido dinámico de la web.

## 6.2.5 JSTL

JSP Standard Tag Library (JSTL) es una agrupación de etiquetas JSP estándares y comúnmente utilizadas en aplicaciones web.

JSTL abarca tareas estructurales como iteraciones y condicionales y tareas de manipulación de información como documentos XML, insertar y extracción de datos a través de SQL.

La colección de etiquetas se puede dividir en una serie de grupos, cada uno de los cuales compone una librería.

Para utilizar cualquiera de estas librerías, se deben añadir mediante la directiva ‘taglib’:

```
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

## 6.2.6 Formularios

Los formularios son una parte básica de una página web, ya que es una de las principales formas de interactuar con el usuario.

Mientras que la manera de presentar estos formularios forma parte de la vista y es una tarea bastante simple, el código necesario para utilizar los datos que el usuario introduce puede ser costoso.

Una vez completada la información por parte del usuario, ésta debe transformarse a objetos Java, ajustándose a algún modelo para posteriormente ser almacenado en nuestra base de datos.

Para evitar posibles inconsistencias o datos erróneos en el modelo, es necesario asegurarse de que los datos introducidos por el usuario son válidos.

Con el fin de conseguir una aplicación de mayor calidad en cuanto a usabilidad, también es necesario implementar el envío de mensajes de error al usuario, indicando los errores cometidos y los campos que lo contienen.

Con Spring es posible simplificar estas tareas, mediante enlaces directos entre las vistas y objetos Java, integración con validadores, y los atributos de redirección para indicar el estado del proceso.

```
@RequestMapping(value = "new", method = RequestMethod.POST)
public String newProductPostMethod(@ModelAttribute("product") Product product,
BindingResult result, RedirectAttributes redirect) {

    // Validación y procesamiento del formulario

}
```

## 6.2.7 Validadores

Spring ofrece una librería de utilidades de validación, a la par que permite a una aplicación definir validadores propios, o usar el paquete de anotaciones estándar de Java.

En nuestro caso todas las validaciones serán realizadas mediante validadores propios.

Cada uno de estos validadores implementará la la interfaz Validator:

```
public interface Validator {
    boolean supports(Class<?> clazz);
    void validate(Object target, Errors errors);
}
```

El método `supports` sirve para indicar qué tipo de clases es capaz de validar, y el método `validate` para implementar la validación en si misma. `Errors` es una interface genérica que define Spring para que las aplicaciones puedan acumular errores.

Un ejemplo de validador propio para un producto tendría la siguiente estructura:

```
@Component
public final class ProductValidator implements Validator {

    @Override
    public boolean supports(Class<?> clazz) {
        return clazz.isAssignableFrom(Product.class);
    }

    @Override
    public void validate(Object target, Errors errors) {

        // Validaciones a aplicar al objeto

        if (validación) {
            errors.rejectValue("campo", "código de error");
        }
    }
}
```

En el método `validate` se realizan las validaciones específicas que se quieren aplicar al objeto.

Como puede apreciarse en el fragmento de código anterior, se hace uso de la interfaz anteriormente mencionada `Errors`, para almacenar los códigos de error, cuyos mensajes de texto asociados se podrán recuperar posteriormente utilizando las capacidades de internacionalización.

Una vez implementado el validador, su utilización desde la capa de vista se realiza así:

```
@RequestMapping(value = "new", method = RequestMethod.POST)
public String newProviderPostMethod(@ModelAttribute(PROVIDER) Provider
    provider, BindingResult result, RedirectAttributes redirect) {

    providerValidator.validate(provider, result);

    if (result.hasErrors()) {
        redirect.addFlashAttribute(PROVIDER_BINDING, result);
        redirect.addFlashAttribute(PROVIDER, provider);
        return REDIRECT_NEW;
    }

    // Acción a realizar si los datos son válidos
}
```

En el método ‘POST’ para enviar los datos del formulario se recibe como parámetro el objeto del modelo de dominio al que se están populando los datos del formulario. También se añaden como parámetros un objeto `BindingResult`, utilizado para obtener los errores de validación, y un objeto `RedirectAttributes`, que sirve para el envío de información en la redirección.

## 6.3 Spring Security

### 6.3.1. Introducción

Spring security es un framework para la securización de aplicaciones basado en los principios fundamentales del framework de Spring (inversión de control e inyección de dependencias).

Es capaz de gestionar seguridad en varios niveles: URLs que se solicitan al servidor, acceso a métodos y clases Java, y acceso a instancias concretas de las clases.

Permite separar la lógica de nuestras aplicaciones del control de la seguridad, utilizando filtros para las peticiones al servidor de aplicaciones o aspectos para la seguridad en clases y métodos.

### 6.3.2. Configuración

Spring Security permite securizar aplicaciones web mediante la utilización de un filtro de seguridad declarado en el fichero web.xml que intercepta las peticiones realizadas.

A continuación se muestra un ejemplo de configuración de dicho filtro:

```
<filter>
  <filter-name>springSecurityFilterChain</filter-name>
  <filter-class>
    org.springframework.web.filter.DelegatingFilterProxy
  </filter-class>
</filter>

<filter-mapping>
  <filter-name>springSecurityFilterChain</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

Es posible securizar recursos Web mediante la declaración de reglas en ficheros XML a través del elemento <http>.

Un ejemplo de securización declarada en el contexto de la aplicación (ApplicationContext.xml) puede verse en el siguiente ejemplo:

```
<http auto-config="true" use-expressions="true">
  <intercept-url pattern="/login" access="permitAll" />
  <intercept-url pattern="/product/**" access="isAuthenticated()" />
  <form-login default-target-url="/home" login-page="/login"
    username-parameter="username" password-parameter="password" />
  <logout logout-success-url="/login" logout-
    url="j_spring_security_logout" />
</http>
```

Otro mecanismo de securización es la utilización de la etiquetas dentro de las páginas jsp.

Para ello es necesario la inclusión de los tags correspondientes, mediante la inclusión de la librería de Spring Security asociada:

```
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>
```

Estas etiquetas permiten mostrar u ocultar elementos en función de, por ejemplo, el rol del usuario conectado, como puede apreciarse en el siguiente fragmento:

```
<sec:authorize access="hasRole('ROLE_ADMIN')">  
    Código a mostrar si se cumple la condición  
</sec:authorize>
```

De esta forma, sólo los usuarios con el rol especificado tendrán acceso al elemento definido dentro de la anotación.

En este caso se estaría limitando el acceso a un único rol, pero también se puede realizar el filtrado especificando un listado de roles, de la siguiente manera:

```
<sec:authorize access="hasAnyRole('ROLE_ADMIN', 'ROLE_EMPLOYEE', 'ROLE_USER')">  
    Código a mostrar si se cumple la condición  
</sec:authorize>
```

## 6.4 MyBatis

### 6.4.1 Introducción

MyBatis es un framework de persistencia, que se encarga de mapear sentencias SQL y procedimientos almacenados con objetos a partir de ficheros XML o anotaciones.

No puede considerarse como un ORM, ya que no pretende realizar un mapeo entre un modelo de objetos y un modelo relacional, sino mapear sentencias específicas con objetos específicos.

Simplifica la programación frente al uso directo de JDBC. Las líneas de código necesarias para ejecutar una sentencia se reducen casi siempre a una.

Esta simplificación ahorra tiempo y evita errores habituales como olvidar cerrar una conexión a base de datos, realizar incorrectamente un mapeo de datos, exceder el tamaño de un result set u obtener varios resultados cuando se esperaba solo uno.

### 6.4.2 Integración con Spring

MyBatis dispone de un módulo de integración con Spring Framework.

Para su utilización es necesario añadir la dependencia en el pom de la aplicación:

```
<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis-spring</artifactId>
  <version>${version}</version>
</dependency>
```

Una aplicación que usa MyBatis debe utilizar una instancia de SqlSessionFactory. Se puede obtener una instancia de SqlSessionFactory mediante un SqlSessionFactoryBuilder.

Un SqlSessionFactoryBuilder puede construir una instancia de SqlSessionFactory a partir de un fichero de configuración XML, de la siguiente manera:

```
<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
  <property name="dataSource" ref="dataSource" />
  <property name="configLocation" value="classpath:mybatis-config.xml" />
  <property name="mapperLocations" value="classpath*:mappers/**/*.xml" />
</bean>
```

Este bean se configura con tres atributos:

- dataSource: bean para obtener la instancia de conexión a la base de datos.
- configLocation: indica el fichero de configuración específico para mybatis
- mapperLocations: indica el fichero o ruta donde se localizan los ficheros de mapeo (mappers).

Cada uno de los ficheros de mapeo contendrá el sql encargado de realizar la manipulación de datos, tal y como puede apreciarse en el siguiente ejemplo:

```
<mapper namespace="tfg.model.Product">

    <resultMap type="Product" id="ProductResultMap">
        <id property="id" column="ID"/>
        <result property="name" column="NAME"/>
        <result property="code" column="CODE"/>
        <result property="description" column="DESCRIPTION"/>
        <result property="priceType" column="PRICE_TYPE"/>
        <result property="price" column="PRICE"/>
        <result property="category.id" column="CATEGORY_ID"/>
        <result property="category.name" column="CATEGORY_NAME"/>
        <result property="active" column="ACTIVE"/>
    </resultMap>

    <insert id="insert" parameterType="Product" useGeneratedKeys="true"
keyProperty="id" keyColumn="ID">
        INSERT INTO PRODUCT (ID, NAME, DESCRIPTION, CODE, ACTIVE, CATEGORY_ID,
            PRICE_TYPE, PRICE, DELETED)
        VALUES (#{id}, #{name}, #{description}, #{code}, #{active},
            #{category.id}, #{priceType}, #{price}, 0)
    </insert>

    <delete id="delete" parameterType="Long">
        UPDATE PRODUCT
        SET DELETED = 1
        WHERE ID = #{productId}
    </delete>

    <select id="get" parameterType="Long" resultMap="ProductResultMap">
        SELECT P.*, C.ID AS CATEGORY_ID, C.NAME AS CATEGORY_NAME
        FROM PRODUCT P
        INNER JOIN CATEGORY C ON C.ID = P.CATEGORY_ID
        WHERE P.ID = #{productId}
        AND P.DELETED = 0
    </select>

</mapper>
```

El fragmento anteriormente expuesto es ya puramente MyBatis, donde se definen las sentencias SQL que deben ser ejecutadas para cada método.

El elemento resultMap es el elemento más importante y potente de MyBatis. Permite realizar el mapeo de datos entre el modelo de datos relacional y el modelo de objetos de la aplicación, especificando en qué atributos del objeto se mapeará el valor de cada una de las columnas obtenidas en la consulta SQL.



## **7. Conclusiones**



## 7. Conclusiones

La realización de este proyecto ha servido para poner en práctica los conocimientos adquiridos durante el desarrollo de la carrera.

Durante el desarrollo del proyecto ha sido necesario realizar cambios en la planificación realizada, para adaptarse a los recursos disponibles y las dificultades que iban surgiendo.

Se ha podido comprobar la dificultad de conlleva la planificación temporal de un proyecto, teniendo que replanificar varias veces las fechas, debido a la estimación temporal errónea o a la falta de recursos en determinados momentos del desarrollo del proyecto.

En cuanto a la planificación establecida durante la etapa de inicio del desarrollo del proyecto, destacar que el retraso ha sido considerable con respecto a la fecha de fin inicialmente estimada.

Esto ha servido para comprender lo complicado que resulta realizar una planificación para un proyecto cuya duración se estima en varios meses.

Esta complejidad se puede asociar a la gran cantidad de factores que hay que tener en cuenta para poder realizar dicha planificación de la manera más realista posible.

Por otra parte, se ha podido comprobar la importancia de las pruebas en la elaboración de un proyecto software, permitiendo al desarrollador asegurar el correcto funcionamiento de cada una de las partes del sistema, de forma que el producto final cumpla con las especificaciones.

### 7.1 Trabajo futuro

Durante el desarrollo de la aplicación se fueron descubriendo posibles funcionalidades adicionales, entre las que destacan las siguientes:

- Acceso a funcionalidades limitadas de la aplicación de gestión para futuros nuevos roles.
- Ampliación de la funcionalidad de gestión, permitiendo añadir un listado de precios de compra para cada producto.
- Ampliación de la funcionalidad de venta TPV, permitiendo seleccionar el cliente al que se realiza la venta.
- Ampliación de la funcionalidad de venta TPV, permitiendo la impresión de un ticket y apertura de un cajón de dinero.



# **Bibliografía**



# Bibliografía

## Referencias bibliográficas:

- Debrauwer, Laurent. “Patrones de diseño en Java: Los 23 modelos de diseño”, Eni, 2013.
- Gamma, Erich., Vlissides, John., Johnson, Ralph., Helm Richard. “Design Patterns: Elements of Reusable Object-Oriented Software”, Addison Wesley, 1995.
- Larman, Craig. “UML y patrones”, Prentice-Hall, 2006.
- Mak, Gary., Rubio, Daniel., Long, Josh. “Spring Recipes - A Problem-Solution Approach”, Apress, 2010
- Martin, Robert C. “Clean Code: A Handbook of Agile Software Craftsmanship”, Prentice-Hall, 2005.
- Sommerville, Ian. “Ingeniería del Software”, Pearson, 2005

## Referencias web:

- Apache Maven Project, disponible en <https://maven.apache.org/> (última vez consultado: Abril de 2018)
- Astah Professional: Software Design Tools for Agile teams with UML, ER Diagram..., disponible en <http://astah.net/> (última vez consultado: Marzo de 2018)
- Balsamiq Mockups, disponible en <https://balsamiq.com/products/> (última vez consultado: Marzo de 2018)
- DBUnit - About DBUnit, disponible en <http://dbunit.sourceforge.net/> (última vez consultado: Marzo de 2018)
- Eclipse IDE, disponible en <https://www.eclipse.org/> (última vez consultado: Enero de 2018)
- Gestión de riesgos de proyectos software, disponible en [https://es.wikiversity.org/wiki/Gesti3n\\_de\\_riesgos\\_de\\_proyectos\\_software](https://es.wikiversity.org/wiki/Gesti3n_de_riesgos_de_proyectos_software), (última vez consultado: Noviembre de 2017)
- Git, disponible en <https://git-scm.com/> (última vez consultado: Marzo de 2018)
- JSTL Quick Reference, disponible en <http://cs.roosevelt.edu/eric/books/JSP/jstl-quick-reference.pdf> (última vez consultado: Abril de 2018)
- JUnit, disponible en <https://junit.org/junit4/> (última vez consultado: Abril de 2018)

- Microsoft Project: Software Project Management | Microsoft Project, disponible en <https://products.office.com/es-es/project/project-and-portfolio-management-software> (última vez consultado: Octubre de 2017)
- MyBatis, disponible en <http://www.mybatis.org/> (última vez consultado: Abril de 2018)
- SonarQube - Continuous code quality, disponible en <https://www.sonarqube.org/> (última vez consultado: Marzo de 2018)
- Spring Framework, disponible en <https://spring.io/>, (última vez consultado: Abril de 2018)
- Spring Security, disponible en <https://docs.spring.io/spring-security/site/docs/current/reference/html/>, (última vez consultado: Abril de 2018)



# **Manual de usuario**

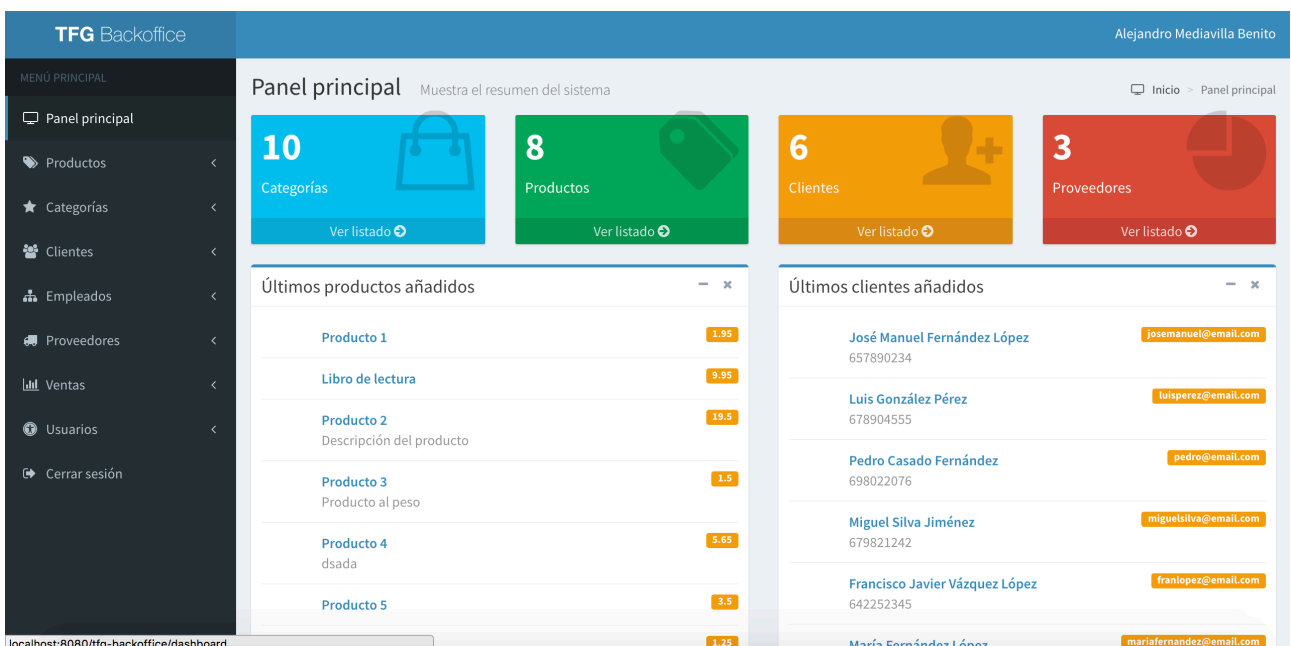
# Anexo I - Manual de usuario

## Acceso a la aplicación:

La aplicación solicitará al usuario su nombre de usuario y contraseña para acceder.



Una vez introducidos los datos, el usuario será redirigido a la página de inicio del sistema, donde podrá ver un breve panel con el resumen de los datos existentes, junto con los últimos productos y clientes añadidos:



## Añadir nuevo producto:

En el menú lateral de la aplicación, seleccionar la opción ‘Productos’.

Se desplegará un submenú con las diferentes opciones disponibles para los productos.

Seleccionar la opción “Añadir nuevo producto”.

Se mostrará al usuario el formulario con los campos para añadir un nuevo producto, tal y como puede apreciarse en la siguiente imagen:

**TFG Backoffice** Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Añadir nuevo producto**
- Listado de productos
- Buscar productos
- Categorías
- Clientes
- Empleados
- Proveedores
- Ventas
- Usuarios

**Nuevo producto** Añade un nuevo producto al sistema

Inicio > Productos > Añadir nuevo producto

Datos del producto

\* Nombre  Descripción

\* Código  \* Categoría

\* Tipo de precio  \* Precio   **Producto activo**

Los campos marcados con (\*) son obligatorios

[Cancelar y volver al listado](#) [Limpiar formulario](#) [Guardar producto](#)

Si existe algún error de validación se mostrará el mensaje correspondiente indicándolo.

Una vez completados correctamente todos los campos, se muestra un mensaje de éxito y se redirige al usuario al listado de productos:

**TFG Backoffice** Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Añadir nuevo producto**
- Listado de productos**
- Buscar productos
- Categorías
- Clientes
- Empleados
- Proveedores
- Ventas
- Usuarios
- Cerrar sesión

**Listado de productos** Muestra el listado de productos del sistema

Productos > Listado de productos

El producto ha sido añadido correctamente

	Nombre	Descripción	Código	Categoría	Tipo de precio	Precio	Activo	Acciones
	Producto 1		CODE_1	Alimentación	PRECIO_UNITARIO	1.95	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Libro de lectura		LIBRO_LECT	Libros y lectura	PRECIO_UNITARIO	9.95	No	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Producto 2	Descripción del producto	CODIGO_PRODUCTO	Alimentación	PRECIO_UNITARIO	19.5	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Producto 3	Producto al peso	CODE_PROD_PESO	Alimentación	PRECIO_PESO	1.5	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Producto 4	dsada	dasdads	Alimentación	PRECIO_UNITARIO	5.65	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Producto 5		CODIGO_PR4	Alimentación	PRECIO_UNITARIO	3.5	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Producto 6		PROD_6	Alimentación	PRECIO_UNITARIO	1.25	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Producto 7		PROD_7	Alimentación	PRECIO_UNITARIO	9.9	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	Producto 7	Descripción	CODIG_7	Alimentación	PRECIO_UNITARIO	1.5	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>

Anterior 1 Siguiente

## Consultar listado de productos:

En el menú lateral de la aplicación, seleccionar la opción ‘Productos’.

Seleccionar la opción “Listado de productos”.

Se mostrará una tabla con todos los productos disponibles, pudiendo ser ordenados por cualquiera de los campos:

Nombre	Descripción	Código	Categoría	Tipo de precio	Precio	Activo	Acciones
Producto 1		CODE_1	Alimentación	PRECIO_UNITARIO	1.95	Sí	Editar Eliminar
Libro de lectura		LIBRO_LECT	Libros y lectura	PRECIO_UNITARIO	9.95	No	Editar Eliminar
Producto 2	Descripción del producto	CODIGO_PRODUCTO	Alimentación	PRECIO_UNITARIO	19.5	Sí	Editar Eliminar
Producto 3	Producto al peso	CODE_PROD_PESO	Alimentación	PRECIO_PESO	1.5	Sí	Editar Eliminar
Producto 4	dsada	dasdas	Alimentación	PRECIO_UNITARIO	5.65	Sí	Editar Eliminar
Producto 5		CODIGO_PR4	Alimentación	PRECIO_UNITARIO	3.5	Sí	Editar Eliminar
Producto 6		PROD_6	Alimentación	PRECIO_UNITARIO	1.25	Sí	Editar Eliminar
Producto 7		PROD_7	Alimentación	PRECIO_UNITARIO	9.9	Sí	Editar Eliminar
Producto 7	Descripción	CODIG_7	Alimentación	PRECIO_UNITARIO	1.5	Sí	Editar Eliminar

## Buscar productos:

En el menú lateral de la aplicación, seleccionar la opción ‘Productos’.

Seleccionar la opción “Buscar productos”.

Se mostrará un formulario con los campos para realizar el filtrado de productos, tal y como puede apreciarse en la siguiente imagen:

Introduzca un criterio para la búsqueda

Una vez completado el formulario, seleccionar la opción “Buscar”.

Se mostrará una tabla con el listado de productos que cumplen los criterios de búsqueda:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Añadir nuevo producto
- Listado de productos
- Buscar productos
- Categorías
- Cientes
- Empleados
- Proveedores
- Ventas
- Usuarios
- Cerrar sesión

Buscar productos Búsqueda de productos por filtros

Productos > Buscar productos

Criterios de búsqueda

\* Nombre: Producto 4 \* Código: \* Categoría: -- Seleccionar --

Q Buscar

Nombre	Descripción	Código	Categoría	Tipo de precio	Precio	Activo	Acciones
Producto 4	dsada	dasdas	Alimentación	PRECIO_UNITARIO	5.65	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>

Anterior 1 Siguiente

### Eliminar producto:

Acceder al listado de productos (o búsqueda de productos), y una vez localizado el producto que se desea borrar, seleccionar la opción “Eliminar”, disponible a la derecha de cada producto.

El sistema mostrará la siguiente ventana de confirmación para eliminar el producto:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Añadir nuevo producto
- Listado de productos
- Buscar productos
- Categorías
- Cientes
- Empleados
- Proveedores
- Ventas
- Usuarios
- Cerrar sesión

Listado de productos

Productos > Listado de productos

Eliminar producto

Se va a eliminar el producto seleccionado. ¿Desea continuar?

Cancelar Eliminar

Nombre	Descripción	Código	Categoría	Tipo de precio	Precio	Activo	Acciones
Producto 4	dsada	dasdas	Alimentación	PRECIO_UNITARIO	5.65	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
Producto 5		CODIGO_PR4	Alimentación	PRECIO_UNITARIO	3.5	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
Producto 6		PROD_6	Alimentación	PRECIO_UNITARIO	1.25	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
Producto 7		PROD_7	Alimentación	PRECIO_UNITARIO	9.9	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
Producto 7	Descripción	CODIG_7	Alimentación	PRECIO_UNITARIO	1.5	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>

Anterior 1 Siguiente

Para confirmar la eliminación del producto pulsar el botón ‘Eliminar’

## Añadir nueva categoría:

En el menú lateral de la aplicación, seleccionar la opción ‘Categorías’.  
Seleccionar la opción “Añadir nueva categoría”.

Se mostrará al usuario el formulario con los campos para añadir una nueva categoría, como puede apreciarse en la siguiente imagen:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Añadir nueva categoría**
- Listado de categorías
- Clientes
- Empleados
- Proveedores
- Ventas
- Usuarios

**Nueva categoría** Añade una nueva categoría de productos al sistema

Inicio > Categorías > Añadir nueva categoría

Datos básicos

\* Nombre Descripción \* Código

Los campos marcados con (\*) son obligatorios

Cancelar y volver al listado Limpiar formulario Guardar categoría

Si existe algún error de validación se mostrará el mensaje correspondiente indicándolo.

Una vez completados correctamente todos los campos, se muestra un mensaje de éxito y se redirige al listado de categorías:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Añadir nueva categoría
- Listado de categorías**
- Clientes
- Empleados
- Proveedores
- Ventas
- Usuarios
- Cerrar sesión

**Listado de categorías** Muestra el listado de categorías de productos

Categorías > Listado de categorías

La categoría ha sido añadida correctamente

Nombre	Descripción	Código	Acciones
★ Alimentación	Artículos de alimentación	CAT_ALIMENTACION	Editar Eliminar
★ Hogar	Artículos para el hogar	CAT_HOGAR	Editar Eliminar
★ Moda	Artículos de moda	CAT_MODA	Editar Eliminar
★ Ofertas	Artículos con las mejores ofertas	CAT_OFERTA	Editar Eliminar
★ Deporte	Artículos para el deporte	CAT_DEPORTE	Editar Eliminar
★ Libros y lectura	Artículos para la lectura	CAT_LECTURA	Editar Eliminar
★ Electrónica	Artículos de electrónica	CAT_ELECTRONICA	Editar Eliminar
★ Consolas y videojuegos	Artículos para el ocio	CAT_CONSOLAS	Editar Eliminar
★ Informática y oficina	Artículos de informática y para la oficina	CAT_INFORMATICA	Editar Eliminar
★ Nombre de la categoría	Descripción de la categoría	CODIGO_CATEGORIA	Editar Eliminar

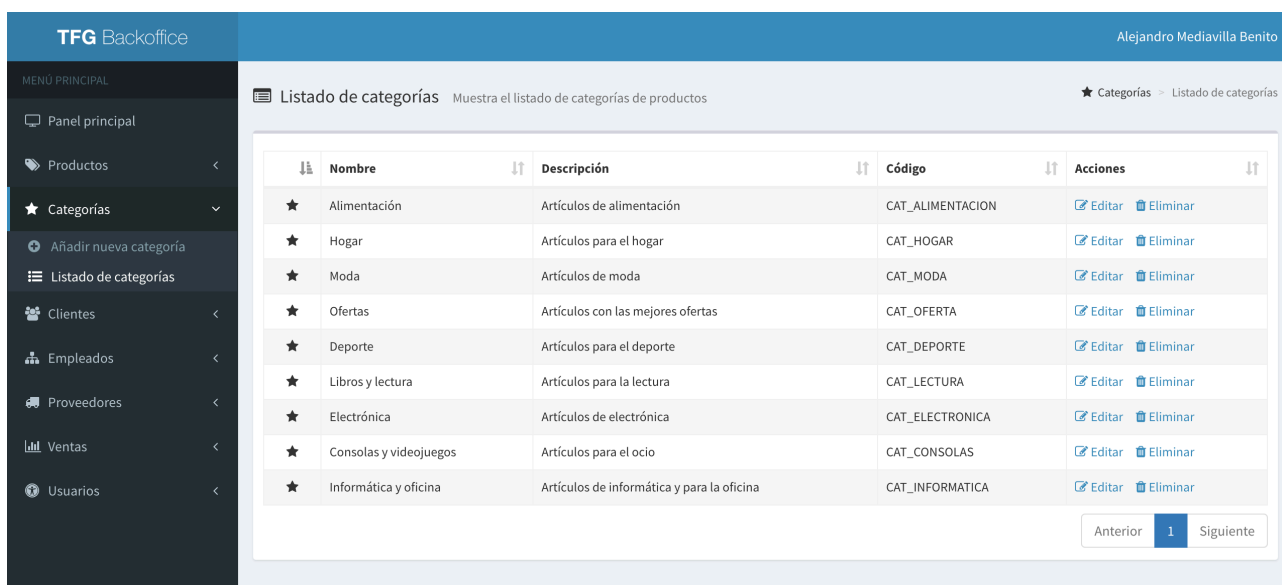
Anterior 1 Siguiente

## Consultar listado de categorías:

En el menú lateral de la aplicación, seleccionar la opción ‘Categorías’.

Seleccionar la opción “Listado de categorías”.

Se mostrará una tabla con todas las categorías disponibles, pudiendo ser ordenadas por cualquiera de los campos (nombre, descripción, código, etc):

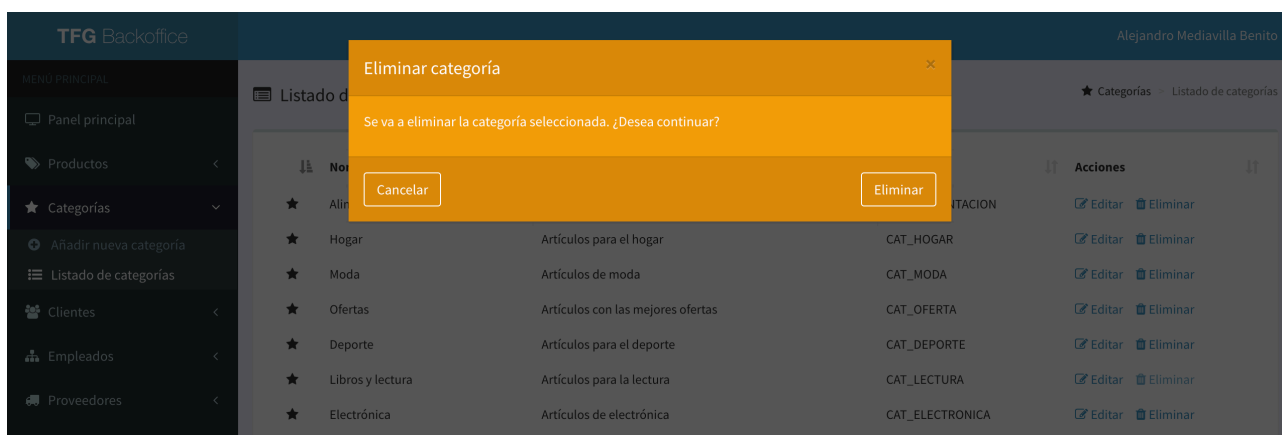


Nombre	Descripción	Código	Acciones
★ Alimentación	Artículos de alimentación	CAT_ALIMENTACION	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Hogar	Artículos para el hogar	CAT_HOGAR	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Moda	Artículos de moda	CAT_MODA	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Ofertas	Artículos con las mejores ofertas	CAT_OFERTA	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Deporte	Artículos para el deporte	CAT_DEPORTE	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Libros y lectura	Artículos para la lectura	CAT_LECTURA	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Electrónica	Artículos de electrónica	CAT_ELECTRONICA	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Consolas y videojuegos	Artículos para el ocio	CAT_CONSOLAS	<a href="#">Editar</a> <a href="#">Eliminar</a>
★ Informática y oficina	Artículos de informática y para la oficina	CAT_INFORMATICA	<a href="#">Editar</a> <a href="#">Eliminar</a>

## Eliminar categoría:

Acceder al listado de categorías (o búsqueda de categorías), y una vez localizada la categoría que se desea borrar, seleccionar la opción “Eliminar”, disponible a la derecha de cada categoría.

El sistema mostrará una ventana de confirmación para eliminar la categoría.



Una vez pulsado el botón eliminar, será eliminada la categoría seleccionada y todos los productos asociados a ella.

## Añadir nuevo cliente:

En el menú lateral de la aplicación, seleccionar la opción ‘Clientes’.

Seleccionar la opción “Añadir nuevo cliente”.

Se mostrará al usuario el formulario con los campos para añadir un nuevo cliente, como puede apreciarse en la siguiente imagen:

**TFG Backoffice** Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Cientes**
  - ➕ Añadir nuevo cliente**
  - Listado de clientes
  - Buscar clientes
- Empleados
- Proveedores
- Ventas
- Usuarios

**➕ Nuevo cliente** Añade un nuevo cliente al sistema Cientes > Añadir nuevo cliente

**Datos personales**

\* DNI/NIF \* Nombre \* Primer apellido \* Segundo apellido

Dirección Número Piso Letra

Localidad Código postal Provincia

**Datos de contacto**

\* Email Teléfono \* Móvil

Los campos marcados con (\*) son obligatorios

Cancelar y volver al listado Limpiar formulario Guardar cliente

Si existe algún error de validación se mostrará el mensaje correspondiente indicándolo.

Una vez completados correctamente todos los campos, se muestra un mensaje de éxito y se redirige al listado de clientes:

**TFG Backoffice** Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Cientes**
  - Añadir nuevo cliente
  - Listado de clientes**
  - Buscar clientes
- Empleados
- Proveedores
- Ventas
- Usuarios
- Cerrar sesión

**Listado de clientes** Muestra el listado de clientes del sistema Cientes > Listado de clientes

El cliente ha sido añadido correctamente

🔽	DNI/NIF	🔽	Nombre	🔽	Apellidos	🔽	Email	🔽	Teléfono	🔽	Teléfono	🔽	Acciones
👤	00874189M		José Manuel		Fernández López		josemanuel@email.com		983456742		657890234		<a href="#">🔗 Editar</a> <a href="#">🗑️ Eliminar</a>
👤	83889593Z		Luis		González Pérez		luisperez@email.com		946195489		678904555		<a href="#">🔗 Editar</a> <a href="#">🗑️ Eliminar</a>
👤	26903603C		Pedro		Casado Fernández		pedro@email.com		935467889		698022076		<a href="#">🔗 Editar</a> <a href="#">🗑️ Eliminar</a>
👤	78514553M		Miguel		Silva Jiménez		miguelsilva@email.com		983645672		679821242		<a href="#">🔗 Editar</a> <a href="#">🗑️ Eliminar</a>
👤	42240359P		Francisco Javier		Vázquez López		franlopez@email.com		932648009		642252345		<a href="#">🔗 Editar</a> <a href="#">🗑️ Eliminar</a>
👤	12365425V		María		Fernández López		mariafernandez@email.com		622346780		645789098		<a href="#">🔗 Editar</a> <a href="#">🗑️ Eliminar</a>

Anterior 1 Sigüiente



## Consultar listado de clientes:

En el menú lateral de la aplicación, seleccionar la opción ‘Clientes’.  
Seleccionar la opción “Listado de clientes”.

Se mostrará el listado completo de clientes, pudiendo ser ordenados por los diferentes campos:

↓↑	DNI/NIF	Nombre	Apellidos	Email	Teléfono	Teléfono	Acciones
👤	00874189M	José Manuel	Fernández López	josemanuel@email.com	983456742	657890234	<a href="#">Editar</a> <a href="#">Eliminar</a>
👤	83889593Z	Luis	González Pérez	luisperez@email.com	946195489	678904555	<a href="#">Editar</a> <a href="#">Eliminar</a>
👤	26903603C	Pedro	Casado Fernández	pedro@email.com	935467889	698022076	<a href="#">Editar</a> <a href="#">Eliminar</a>
👤	78514553M	Miguel	Silva Jiménez	miguelsilva@email.com	983645672	679821242	<a href="#">Editar</a> <a href="#">Eliminar</a>
👤	42240359P	Francisco Javier	Vázquez López	franlopez@email.com	932648009	642252345	<a href="#">Editar</a> <a href="#">Eliminar</a>

## Buscar clientes:

En el menú lateral de la aplicación, seleccionar la opción ‘Clientes’.  
Seleccionar la opción “Buscar clientes”.

Se mostrará un formulario con los campos para realizar el filtrado de clientes.

Buscar clientes Búsqueda de clientes por filtros

Criterios de búsqueda

\* DNI/NIF

\* Nombre

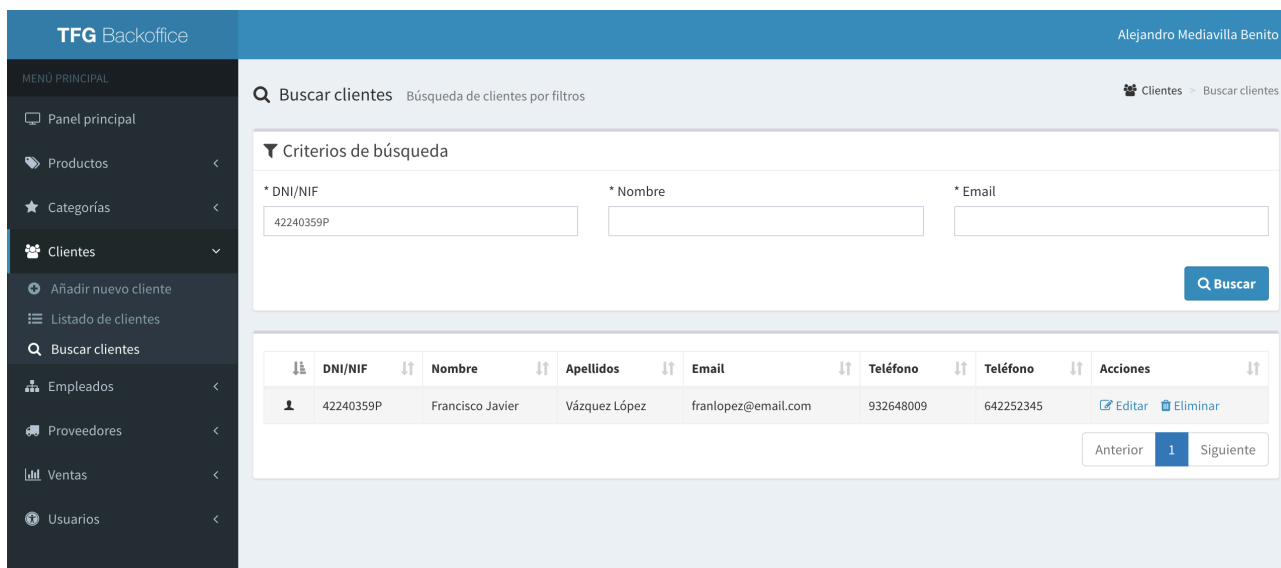
\* Email

[Q Buscar](#)

Introduzca un criterio para la búsqueda

Una vez completado el formulario, seleccionar la opción “Buscar”.

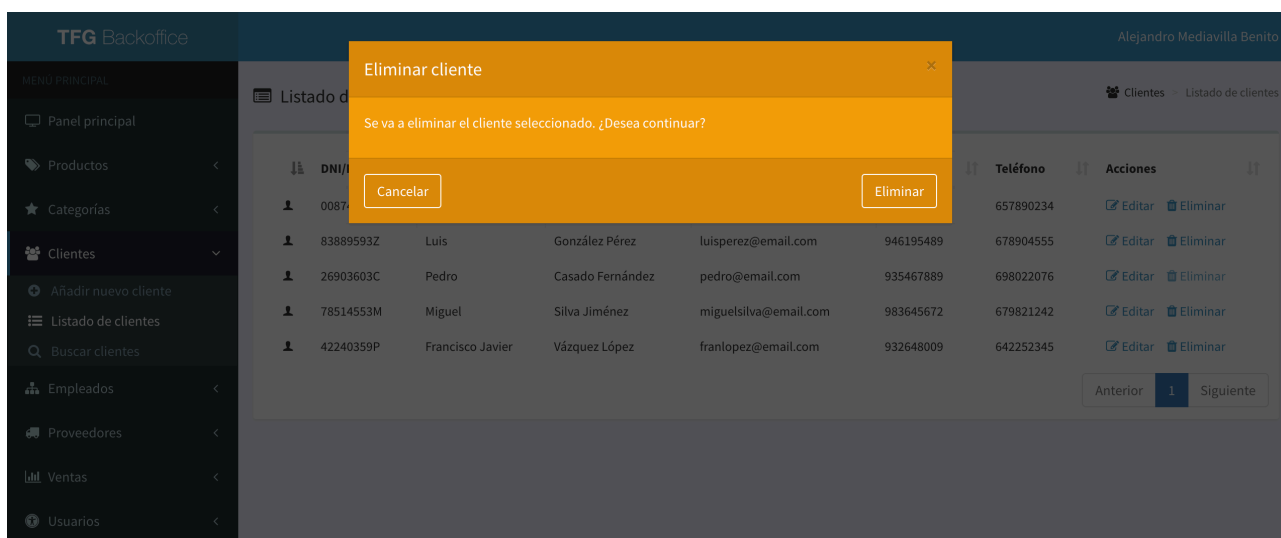
Se mostrará una tabla con el listado de clientes que cumplen los criterios de búsqueda, como se puede apreciar en la siguiente imagen:



### **Eliminar cliente:**

Acceder al listado de clientes (o búsqueda de clientes), y una vez localizado el cliente que se desea borrar, seleccionar la opción “Eliminar”, disponible a la derecha de cada cliente.

El sistema mostrará una ventana de confirmación para eliminar el cliente:



Una vez pulsado el botón eliminar, el cliente será eliminado del sistema.

Para cancelar la eliminación del cliente, pulsar el botón ‘Cancelar’.

## Añadir nuevo proveedor:

En el menú lateral de la aplicación, seleccionar la opción ‘Proveedores’.

Se desplegará un submenú con las diferentes opciones disponibles para los proveedores.

Seleccionar la opción “Añadir nuevo proveedor”.

Se mostrará al usuario el formulario con los campos para añadir un nuevo proveedor, como puede apreciarse en la siguiente imagen:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Cientes
- Empleados
- Proveedores
- Añadir nuevo proveedor
- Listado de proveedores
- Buscar proveedores
- Ventas
- Usuarios

**Nuevo proveedor** Añade un nuevo proveedor al sistema

Proveedores > Añadir nuevo proveedor

Datos personales

\* Nombre Descripción \* Código

Datos de contacto

Email Teléfono

Los campos marcados con (\*) son obligatorios

Cancelar y volver al listado Limpiar formulario Guardar proveedor

Si existe algún error de validación se mostrará el mensaje correspondiente indicándolo.

Una vez completados correctamente todos los campos, se muestra un mensaje de éxito y se redirige al usuario al listado de proveedores:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Cientes
- Empleados
- Proveedores
- Añadir nuevo proveedor
- Listado de proveedores
- Buscar proveedores
- Ventas
- Usuarios
- Cerrar sesión

**Listado de proveedores** Muestra el listado de proveedores del sistema

Proveedores > Listado de proveedores

El proveedor ha sido añadido correctamente

Nombre	Descripción	Código	Email	Teléfono	Acciones
Olid	Productos de Olid	CODE_OLID	olid@email.com	666555444	Editar Eliminar
Grefusa	Productos de marca Grefusa	CODE_GREF	grefusa@email.com	983252525	Editar Eliminar
Amazon	Productos de Amazon	CODIGO_AMAZON	amazon@email.com	655234890	Editar Eliminar

Anterior 1 Siguiente

## Consultar listado de proveedores:

En el menú lateral de la aplicación, seleccionar la opción ‘Proveedores’.  
Seleccionar la opción “Listado de proveedores”.

Se mostrará una tabla con todos los proveedores disponibles, pudiendo ser ordenados por cualquiera de los campos (nombre, descripción, código, email, etc):

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Clientes
- Empleados
- Proveedores
- Añadir nuevo proveedor
- Listado de proveedores
- Buscar proveedores
- Ventas
- Usuarios
- Cerrar sesión

Listado de proveedores Muestra el listado de proveedores del sistema Proveedores > Listado de proveedores

Nombre	Descripción	Código	Email	Teléfono	Acciones
Olid	Productos de Olid	CODE_OLID	olid@email.com	666555444	<a href="#">Editar</a> <a href="#">Eliminar</a>
Grefusa	Productos de marca Grefusa	CODE_GREF	grefusa@email.com	983252525	<a href="#">Editar</a> <a href="#">Eliminar</a>
Amazon	Productos de Amazon	CODIGO_AMAZON	amazon@email.com	655234890	<a href="#">Editar</a> <a href="#">Eliminar</a>

Anterior 1 Siguiente

## Buscar proveedores:

En el menú lateral de la aplicación, seleccionar la opción ‘Proveedores’.  
Seleccionar la opción “Buscar proveedores”.

Se mostrará un formulario con los campos para realizar el filtrado de proveedores.

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Clientes
- Empleados
- Proveedores
- Añadir nuevo proveedor
- Listado de proveedores
- Buscar proveedores
- Ventas
- Usuarios
- Cerrar sesión

Buscar proveedores Búsqueda de proveedores por filtros Proveedores > Buscar proveedores

Criterios de búsqueda

\* Nombre \* Código \* Código

Buscar

Introduzca un criterio para la búsqueda

Una vez completados los filtros para realizar la búsqueda, pulsar el botón ‘Buscar’.  
Se mostrará el listado proveedores que cumplen los criterios de búsqueda establecidos.

The screenshot shows the 'Buscar proveedores' (Search providers) page in the TFG Backoffice. The left sidebar contains a menu with options like 'Panel principal', 'Productos', 'Categorías', 'Clientes', 'Empleados', 'Proveedores', 'Añadir nuevo proveedor', 'Listado de proveedores', 'Buscar proveedores', 'Ventas', 'Usuarios', and 'Cerrar sesión'. The main content area has a search bar with the text 'Buscar proveedores' and a sub-header 'Búsqueda de proveedores por filtros'. Below the search bar, there are three input fields for search criteria: 'Nombre' (containing 'Amazon'), 'Código', and another 'Código'. A 'Buscar' button is located to the right of these fields. Below the search criteria, there is a table listing the search results. The table has columns for 'Nombre', 'Descripción', 'Código', 'Email', 'Teléfono', and 'Acciones'. The first row shows 'Amazon' with description 'Productos de Amazon', code 'CODIGO\_AMAZON', email 'amazon@email.com', and phone '655234890'. The 'Acciones' column for this row contains 'Editar' and 'Eliminar' buttons. At the bottom right of the table, there are pagination controls: 'Anterior', '1', and 'Siguiente'.

### **Eliminar proveedores:**

Acceder al listado de proveedores (o búsqueda de proveedores), y una vez localizado el proveedor que se desea borrar, seleccionar la opción “Eliminar”, de las opciones de la derecha.

El sistema mostrará una ventana de confirmación para eliminar el proveedor.

The screenshot shows the 'Listado de proveedores' (Providers list) page in the TFG Backoffice. A confirmation dialog box is overlaid on top of the page. The dialog has a title 'Eliminar proveedor' and a close button (X). The text inside the dialog says 'Se va a eliminar el proveedor seleccionado. ¿Desea continuar?' (The selected provider is going to be deleted. Do you want to continue?). There are two buttons at the bottom of the dialog: 'Cancelar' (Cancel) and 'Eliminar' (Delete). The background shows the provider list table with columns 'Nombre', 'Descripción', 'Código', 'Email', 'Teléfono', and 'Acciones'. The first row is 'Olid', the second is 'Grefusa', and the third is 'Amazon'. The 'Acciones' column for the 'Amazon' row shows 'Editar' and 'Eliminar' buttons. The pagination controls at the bottom right of the table are 'Anterior', '1', and 'Siguiente'.

Para confirmar la eliminación pulsar el botón ‘Eliminar’.

## Añadir nuevo empleado:

En el menú lateral de la aplicación, seleccionar la opción ‘Empleados’.

Se desplegará un submenú con las diferentes opciones disponibles para los empleados.

Seleccionar la opción “Añadir nuevo empleado”.

Se mostrará al usuario el formulario con los campos para añadir un nuevo empleado, como puede apreciarse en la siguiente imagen:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Clientes
- Empleados
  - Añadir nuevo empleado
  - Listado de empleados
  - Buscar empleados
- Proveedores
- Ventas
- Usuarios

**Nuevo empleado** Añade un nuevo empleado al sistema Empleados > Añadir nuevo empleado

Datos personales

\* Nombre Primer apellido Segundo apellido \* Código

Datos de contacto

Email Teléfono

Los campos marcados con (\*) son obligatorios

Cancelar y volver al listado Limpiar formulario Guardar empleado

Si existe algún error de validación se mostrará el mensaje correspondiente indicándolo.

Una vez completados correctamente todos los campos, se muestra un mensaje de éxito y se redirige al listado de empleados:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Clientes
- Empleados
  - Añadir nuevo empleado
  - Listado de empleados
  - Buscar empleados
- Proveedores
- Ventas
- Usuarios
- Cerrar sesión

**Listado de empleados** Muestra el listado de empleados del sistema Empleados > Listado de empleados

El empleado ha sido añadido correctamente

👤	Código	Nombre	Apellido 1	Apellido 2	Email	Teléfono	Acciones
👤	ALE_MEDIIV	Alejandro	Mediavilla	Benito	alemedi@email.com	616426309	✎ Editar 🗑 Eliminar
👤	CRIS_GONZ	Cristina	González	Suárez	cristinagonz@email.com	610986321	✎ Editar 🗑 Eliminar
👤	Vazquez	empleado_administrador	Javier	López			✎ Editar 🗑 Eliminar

Anterior 1 Siguiente

## Consultar listado de empleado:

En el menú lateral de la aplicación, seleccionar la opción ‘Empleados’.

Seleccionar la opción “Listado de empleados”.

Se mostrará una tabla con todos los empleados disponibles, pudiendo ser ordenados por cualquiera de los campos (código, nombre, apellidos, email, etc):

📄	📄	📄	📄	📄	📄	📄	📄	📄
Código	Nombre	Apellido 1	Apellido 2	Email	Teléfono	Acciones		
ALE_MEDIAV	Alejandro	Mediavilla	Benito	alemedi@email.com	616426309	<a href="#">Editar</a> <a href="#">Eliminar</a>		
LUIS_FERNAN	Luis	Fernández	López	luisfernan@email.com	612341280	<a href="#">Editar</a> <a href="#">Eliminar</a>		
CRIS_GONZ	Cristina	González	Suárez	cristinagonz@email.com	610986321	<a href="#">Editar</a> <a href="#">Eliminar</a>		

## Buscar empleados:

En el menú lateral de la aplicación, seleccionar la opción ‘Empleados’.

Seleccionar la opción “Buscar empleados”.

Se mostrará un formulario con los campos para realizar el filtrado de empleados.

🔍 Buscar empleados Búsqueda de empleados por filtros

Empleados > Buscar empleados

▼ Criterios de búsqueda

\* Nombre

\* Código

\* Email

[🔍 Buscar](#)

Introduzca un criterio para la búsqueda

Una vez completado el formulario, seleccionar la opción “Buscar”.

Se mostrará una tabla con el listado de empleados que cumplen los criterios de búsqueda:

The screenshot shows the 'Buscar empleados' (Search employees) page in the TFG Backoffice. The left sidebar contains a menu with options like 'Panel principal', 'Productos', 'Categorías', 'Clientes', 'Empleados', 'Añadir nuevo empleado', 'Listado de empleados', 'Buscar empleados', 'Proveedores', 'Ventas', 'Usuarios', and 'Cerrar sesión'. The main content area has a search bar with the text 'Buscar empleados' and a subtitle 'Búsqueda de empleados por filtros'. Below the search bar are three input fields for search criteria: '\* Nombre' (containing 'Cristina'), '\* Código', and '\* Email'. A blue 'Buscar' button is located to the right of these fields. Below the search criteria is a table of search results. The table has columns for 'Código', 'Nombre', 'Apellido 1', 'Apellido 2', 'Email', 'Teléfono', and 'Acciones'. One employee is listed: 'CRIS\_GONZ', 'Cristina', 'González', 'Suárez', 'cristinagonz@email.com', and '610986321'. The 'Acciones' column for this employee contains 'Editar' and 'Eliminar' links. At the bottom right of the table are navigation buttons: 'Anterior', '1', and 'Siguiente'.

### Eliminar empleado:

Acceder al listado de empleados (o búsqueda de empleados), y una vez localizado el empleado que se desea borrar, seleccionar la opción “Eliminar”, disponible a la derecha de cada empleado.

El sistema mostrará una ventana de confirmación para eliminar el empleado:

The screenshot shows the 'Listado de empleados' (Employee list) page in the TFG Backoffice. A confirmation dialog box is overlaid on top of the page. The dialog has a title 'Eliminar empleado' and a close button (X). The main text of the dialog says 'Se va a eliminar el empleado seleccionado. ¿Desea continuar?' (The selected employee is going to be deleted. Do you want to continue?). There are two buttons at the bottom of the dialog: 'Cancelar' (Cancel) and 'Eliminar' (Delete). The background shows a table of employees with columns for 'Código', 'Nombre', 'Apellido 1', 'Apellido 2', 'Email', 'Teléfono', and 'Acciones'. The 'Acciones' column contains 'Editar' and 'Eliminar' links for each employee. The table shows three employees: 'ALE...', 'LUIS\_FERNAN', and 'CRIS\_GONZ'. The 'CRIS\_GONZ' employee is highlighted, indicating it is the one being targeted for deletion.



## Añadir nuevo usuario:

En el menú lateral de la aplicación, seleccionar la opción ‘Usuarios’.  
Seleccionar la opción “Añadir nuevo usuario”.

Se mostrará al usuario el formulario con los campos para añadir un nuevo usuario, como puede apreciarse en la siguiente imagen:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Clientes
- Empleados
- Proveedores
- Ventas
- Usuarios**
- Añadir nuevo usuario
- Listado de usuarios

**+ Nuevo usuario** Añade un nuevo usuario al sistema Inicio > Usuarios > Añadir nuevo usuario

Datos de usuario

\* Nombre de usuario \* Contraseña \* Confirmar contraseña  Usuario Activo

Datos de asignación de usuario

\* Rol del usuario \* Empleado

ROLE\_ADMIN Alejandro

Los campos marcados con (\*) son obligatorios

Cancelar y volver al listado Limpiar formulario Guardar producto

En los datos de asignación de usuario se especifica el rol que se quiere dar a ese usuario y el empleado al que le será asignado el usuario creado.

## Consultar listado de usuarios:

En el menú lateral de la aplicación, seleccionar la opción Productos.  
Seleccionar la opción “Listado de productos”.

Se mostrará una tabla con todos los productos disponibles, pudiendo ser ordenados por cualquiera de los campos:

TFG Backoffice Alejandro Mediavilla Benito

MENÚ PRINCIPAL

- Panel principal
- Productos
- Categorías
- Clientes
- Empleados
- Proveedores
- Ventas

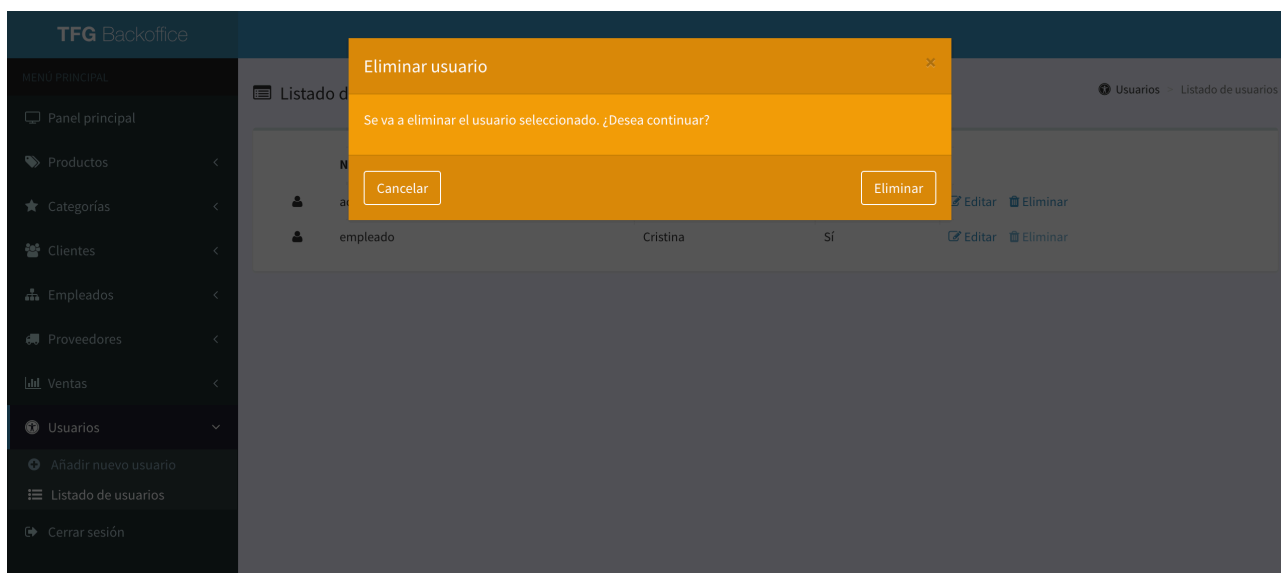
**Listado de usuarios** Muestra el listado de usuarios del sistema Usuarios > Listado de usuarios

	Nombre de usuario	Empleado	Activo	
	administrador	Alejandro	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>
	empleado	Cristina	Sí	<a href="#">Editar</a> <a href="#">Eliminar</a>

## **Eliminar usuario:**

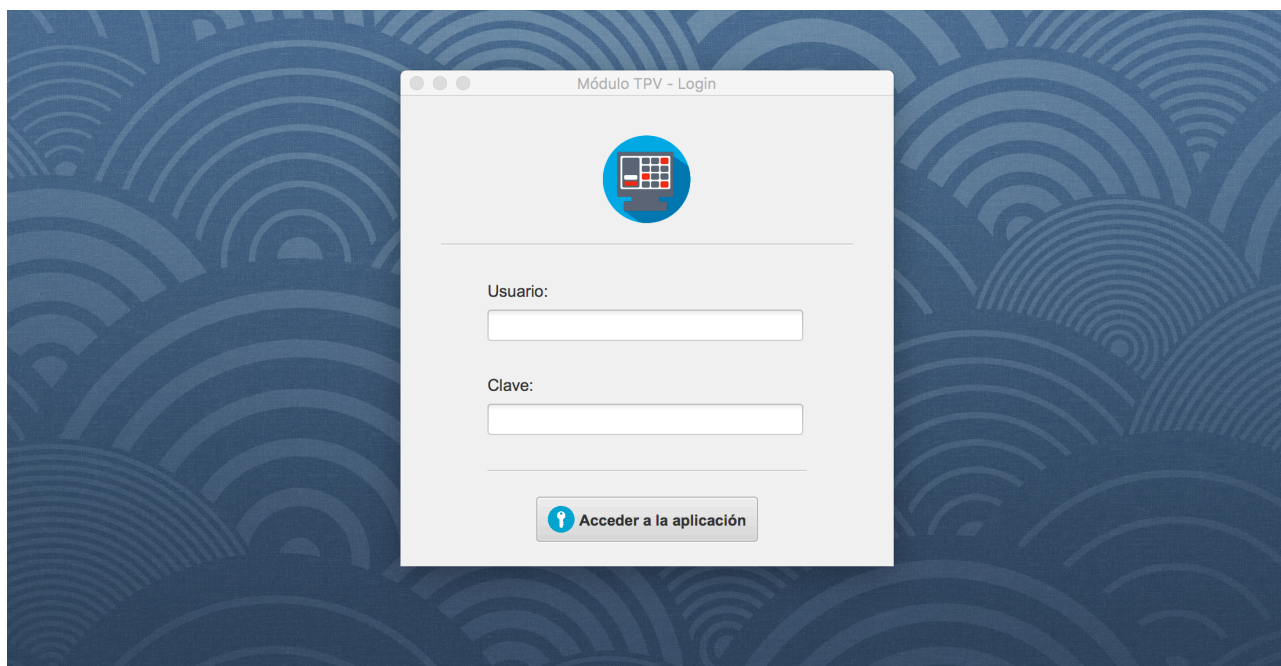
Acceder al listado de usuarios, y una vez localizado el usuario que se desea borrar, seleccionar la opción “Eliminar”, disponible a la derecha del listado.

El sistema mostrará una ventana de confirmación para eliminar el usuario:



## **Acceso a la aplicación de TPV:**

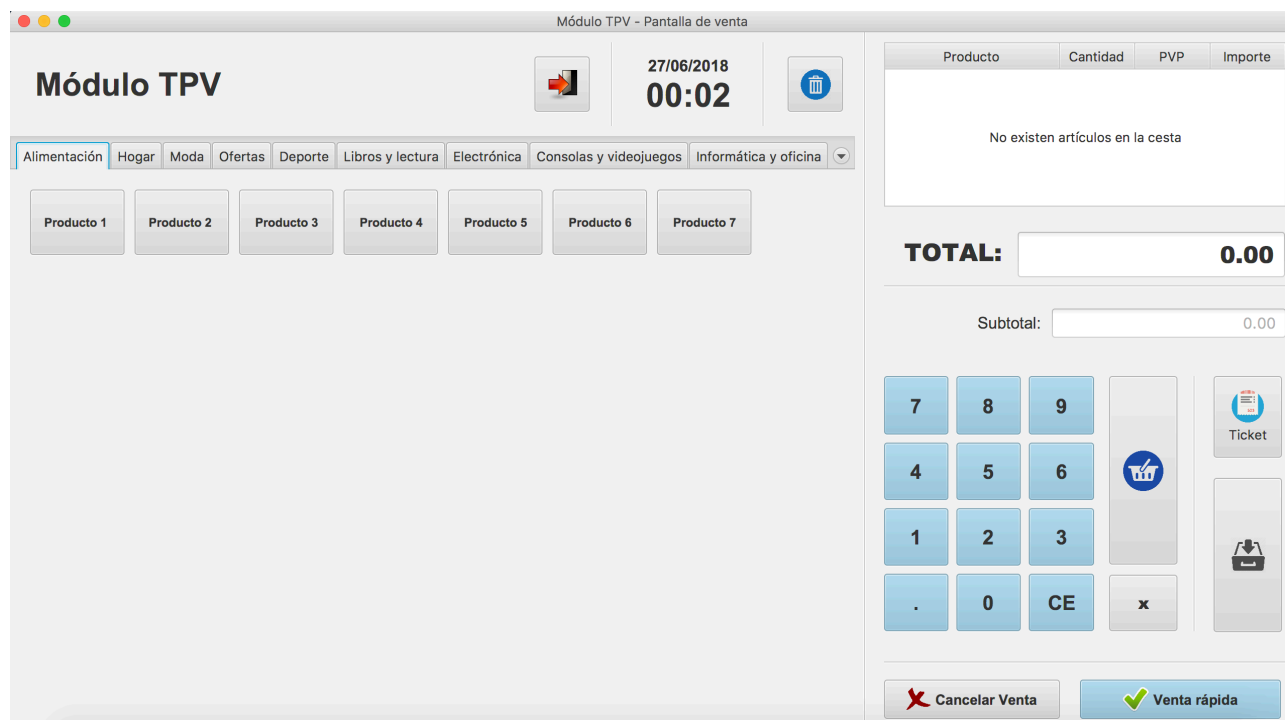
Una vez instalada y ejecutada la aplicación, se solicitará el nombre de usuario y contraseña para poder acceder a la aplicación de venta, tal y como se muestra en la siguiente imagen:



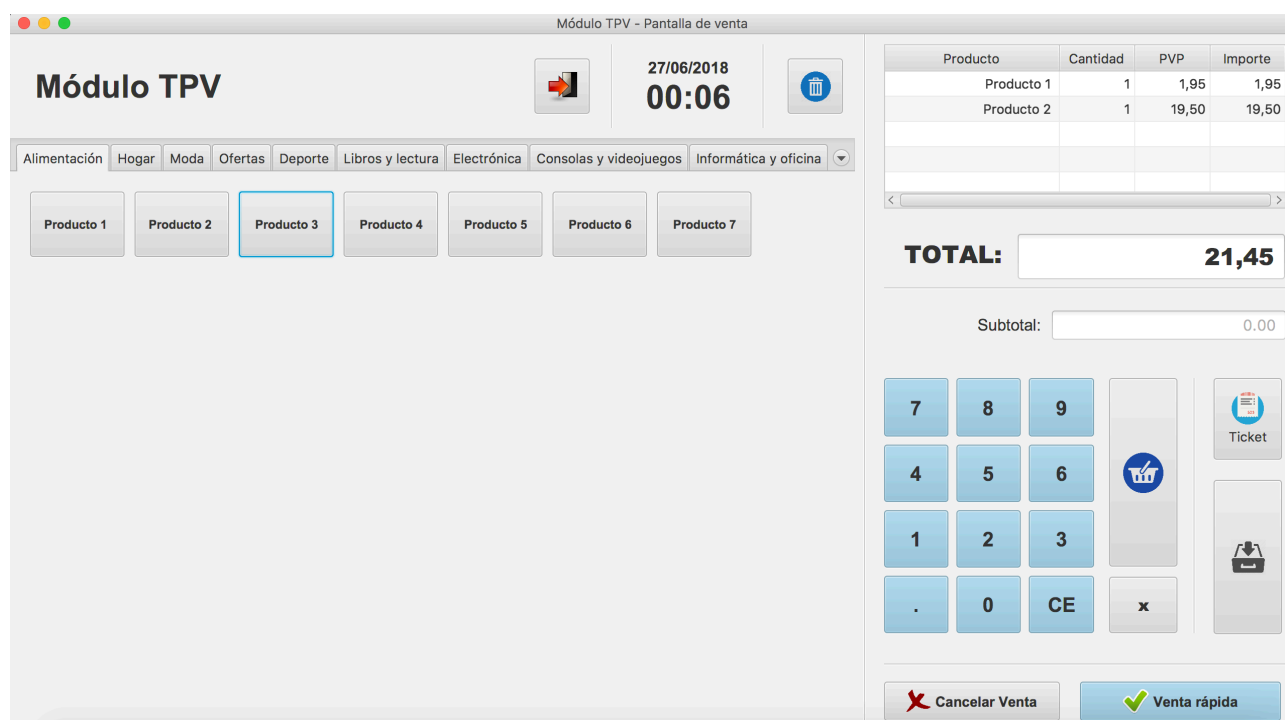
## Realización de ventas:

Una vez realizado el login de usuario, se cargará la pantalla de venta.

Esta pantalla se cargará de forma dinámica en base a las categorías y productos existentes en el sistema. Cada categoría aparecerá en una pestaña, y dentro de cada pestaña estarán los productos asociados a dicha categoría:



Para añadir productos a la venta basta con pulsar el botón del producto, que será incorporado inmediatamente al carrito de la compra, como puede verse en la siguiente imagen:



Si se desea eliminar algún artículo del carrito de la compra basta con seleccionarlo y pulsar el botón situado a la izquierda con el icono de papelera. Este botón eliminará el producto seleccionado de los productos existentes en el carrito.

Una vez añadidos todos los productos que se quieren a la venta, se deberá pulsar el botón ‘Venta rápida’. Una vez hecho esto, la venta será añadida a la base de datos y se volverá a vaciar el carrito de la compra, para estar disponible para la siguiente venta.

También existe la posibilidad de realizar ventas asignando el importe directamente desde la parte de ‘calculadora’. De esta manera se incorporará al carrito de venta el importa marcado desde esta opción, con el valor de ‘Venta sin determinar’. Puede verse en la siguiente imagen:

Producto	Cantidad	PVP	Importe
Producto 1	1	1,95	1,95
Producto 2	1	19,50	19,50
Venta sin determinar	1	1,75	1,75
Venta sin determinar	1	2,25	2,25

**TOTAL:** 25,45

Subtotal: 0.00

7 8 9  
4 5 6  
1 2 3  
. 0 CE x

Cancelar Venta **Venta rápida**

NOTA: Las opciones de abrir cajón e imprimir ticket no están disponibles en esta versión, ya que son funcionalidades que serán implementadas en futuras versiones de la aplicación.

## **Contenido del CD**



## **Anexo II - Contenido del CD**

La distribución de contenidos del soporte digital que acompaña a la memoria entregada es la siguiente:

- **memoria.pdf**: archivo digital en formato pdf que contiene todo lo expuesto en esta memoria.
- **CódigoFuente/**: directorio que contiene el código fuente de todo el software desarrollado.
- **Manual/**: directorio que contiene el archivo digital en formato pdf explicando el funcionamiento del sistema.