

DUOC UC - Escuela de informática y telecomunicaciones

# Propuesta de Proyecto y Especificación de Requisitos de Software

---

*Proyecto: Perfulandia SPA*

Revisión:  
11/04/2025

## Contenido

<b>FICHA DEL DOCUMENTO</b>	<b>3</b>
<b>1. INTRODUCCIÓN</b>	<b>4</b>
1.1. PROPÓSITO	4
1.2. ÁMBITO DEL SISTEMA	4
1.3. DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS	4
1.4. REFERENCIAS	4
1.5. VISIÓN GENERAL DEL DOCUMENTO	4
<b>2. DESCRIPCIÓN GENERAL</b>	<b>5</b>
2.1. PERSPECTIVA DEL PRODUCTO	5
2.2. FUNCIONES DEL PRODUCTO	5
2.3. CARACTERÍSTICAS DE LOS USUARIOS	5
2.4. RESTRICCIONES	5
2.5. SUPOSICIONES Y DEPENDENCIAS	6
2.6. REQUISITOS FUTUROS	6
<b>3. REQUISITOS ESPECÍFICOS</b>	<b>7</b>
3.1 REQUISITOS COMUNES DE LAS INTERFACES	8
3.1.1 <i>Interfaces de usuario</i>	8
3.1.2 <i>Interfaces de hardware</i>	8
3.1.3 <i>Interfaces de software</i>	8
3.1.4 <i>Interfaces de comunicación</i>	8
3.2 REQUISITOS FUNCIONALES	8
3.3 REQUISITOS NO FUNCIONALES	9
3.3.1 <i>Requisitos de rendimiento</i>	9
3.3.2 <i>Seguridad</i>	9
3.3.3 <i>Fiabilidad</i>	10
3.3.4 <i>Disponibilidad</i>	10
3.3.5 <i>Mantenibilidad</i>	10
3.3.6 <i>Portabilidad</i>	10
3.4 OTROS REQUISITOS	10
<b>4. PROPUESTA DE PLANIFICACIÓN</b>	<b>11</b>
4.1 DESCRIPCIÓN GENERAL ACERCA DE LA PLANIFICACIÓN	11
4.1.2 <i>Definición del Equipo de Trabajo</i>	11
4.1.3 <i>Definición de Actividades principales del Proyecto</i>	11
4.1.4 <i>Diagrama EDT</i>	11
4.1.5 <i>Carta Gantt</i>	11
4.1.6 <i>Resumen Costos del Desarrollo del Proyecto</i>	11

### Integrantes:

Nombre Integrante del Equipo	Rol Definido
<i>Valeria Gallardo</i>	<i>Líder de proyecto y desarrolladora Front-End</i>
<i>Sebastian Fredes</i>	<i>Analista y desarrollador Back-End</i>

## 1. Introducción

En este informe se tratará el caso de una empresa emergente, la cual está usando un sistema monolítico y no escalable. El caso en el que se nos pone es uno donde la empresa ha comenzado a tener un aumento exponencial de clientes y se ha comenzado a expandir bastante el último tiempo, lo cual provocó que su sistema, el cual es el mismo que usan tanto sus clientes como sus trabajadores, comience a presentar problemas y fallos al recibir una carga inesperada y masiva a comparación de cómo había empezado y para la cantidad de personas que inicialmente el programa estaba preparado. Claramente el sistema de Perfulandia SPA no era lo suficientemente escalable, porque no fue ideado pensando en una posible masificación futura de este mismo, y eso significa también que no se tenía una preparación para el momento en que la empresa creciera. El sistema monolítico que se está usando era suficiente para un inicio. Quizá era más barato y sencillo de desarrollar por las pocas necesidades, pero en este momento se requiere un cambio, y eso es lo que tratará este informe, cómo un sistema monolítico puede ser perjudicial para una empresa que busca escalabilidad y expandirse, así como también una solución para la necesidad de migrar un sistema como este a los microservicios, los cuales resuelven las necesidades actuales de Perfulandia SPA.

### 1.1. Propósito

El propósito de este informe es especificar los requerimientos del sistema y mostrar cómo se puede resolver esta problemática. Se busca describir con precisión y detalle métodos con estructuras definiciones técnicas el desarrollo de este proyecto. El informe se dirige a las personas que buscan desarrollar este sistema o comprender cómo se puede hacer una implementación de micro servicios a partir de un sistema monolítico.

### 1.2. Ámbito del Sistema

En esta subsección:

- El sistema se llama Perfulandia SPA.
- El sistema está capacitado para soportar grandes cargas de trabajo. Está diseñado para tener una gran escalabilidad, mantenerse estable bajo un gran flujo de información y proveer de los servicios necesarios a sus usuarios para que puedan crear sus cuentas, realizar sus pagos, añadir direcciones, comunicarse con servicio al cliente, etc. El sistema no está diseñado para empresas o personal con intenciones de compras industriales. Está dirigido a un público minorista y que generalmente realizará compras de no más de 10 productos. Se pueden realizar más, pero no se tiene una facilidad para las personas que desean comprar cientos de unidades.
- Los beneficios que se consiguen con este nuevo sistema es una solución inmediata a los problemas de conectividad que se pueden presentar y las fallas que un sistema monolítico suele presentar al tener un flujo de información mayor al esperado. El objetivo es que el programa sea escalable a futuro, permitiendo una mayor facilidad y flexibilidad para la modificación de este mismo por diferentes personas.

### 1.3. Definiciones, Acrónimos y Abreviaturas

- Backup: Significa “Respaldo” en inglés, y se utiliza cuando se menciona un sistema que respalde información.
- Front-end: Es la parte visible de una aplicación o sitio web, es todo lo que el usuario común llega a ver y con lo que interactúa en un sistema.
- Back-end: Es todo lo que el usuario no puede ver ni tampoco interactúa directamente con el back-end. Todas las funciones del sistema que ocurren como las solicitudes o las bases de datos son back-end.
- API: Es la forma en que se comunican el front-end y el back-end permite que estos trabajen en conjunto.
- Framework: Es un marco de trabajo de programación. Ayuda y facilita el desarrollo de programas y sistemas.
- JSON: Es un formato de guardado de archivos. Contiene datos en su interior.

### 1.4. Referencias

Se hace referencia a otro documento dentro de este mismo, en la sección 5.2 se presenta un anexo para responder y expandirse sobre las pruebas ágiles realizadas para este proyecto.

### 1.5. Visión General del Documento

Se verán requisitos funcionales, no funcionales, organización por tiempos guiados por carta Gantt, análisis del sistema anterior y sus puntos débiles, análisis del nuevo sistema y sus puntos fuertes, se mostrarán los usuarios a los que va dirigido el sistema, los métodos utilizados para realizar una migración de este nivel y los programas que ayudan a esto.

El documento busca especificar la forma en que se lleva a cabo una tarea de este nivel y explicar detalladamente la manera en que el grupo decidió llevar la organización para resolver este caso.

## 2. Descripción General

El sistema puede ser afectado por una carga de datos superior a la que está diseñada para recibir, entregando resultados desfavorables para la empresa como una respuesta lenta al momento de que el cliente intente realizar una compra o que directamente el programa deje de funcionar completamente. Si ocurre un solo error por minúsculo que sea, no se tiene un sistema de backup para que el programa siga funcionando porque no existen servicios capaces de hacer algo así dentro de una estructura monolítica.

### 2.1. Perspectiva del Producto

El producto es dependiente de otros y funciona en conjunto con diferentes procesos y programas. Con la ayuda de Oracle se pueden traspasar las bases de datos previamente creadas e integrarlas en un nuevo sistema, ayudándose de la nube para almacenar los datos.

## 2.2. Funciones del Producto

El sistema cuenta con una interfaz para el manejo de diferentes tipos de usuarios.

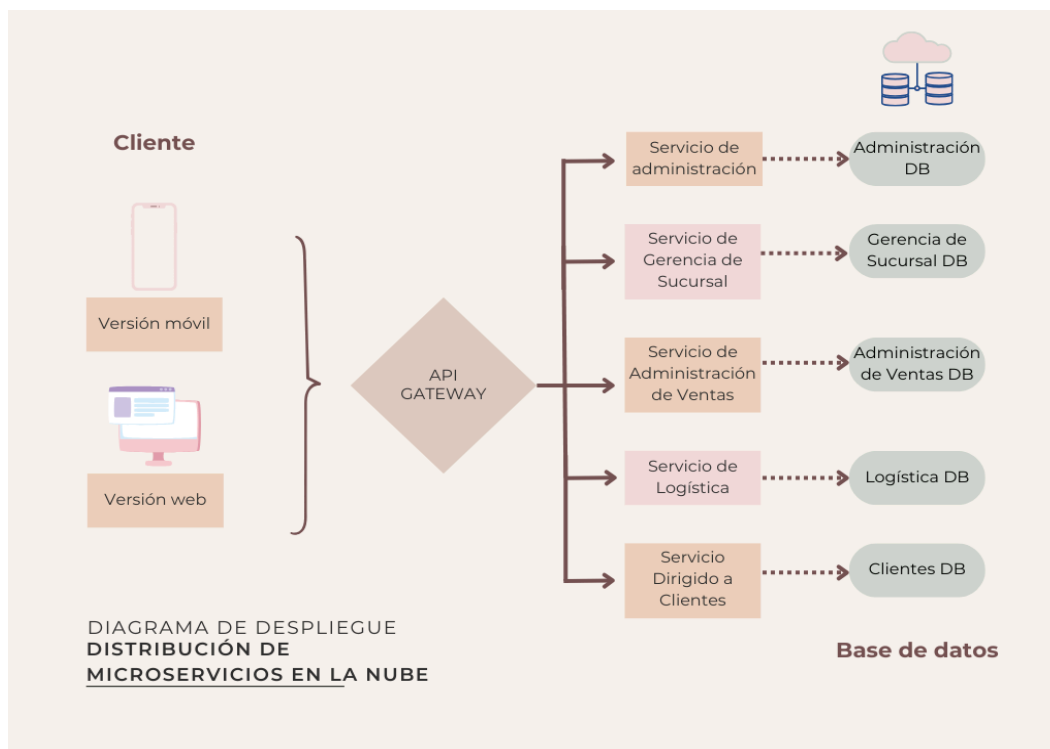
**Sistema de administración:** Lo utilizará solo personal autorizado y desde aquí se podrá acceder a la información que el administrador desee. Se conciben todos los permisos para buscar información y modificar datos existentes en el sistema.

**Servicio de gerencia:** Lo utiliza el personal designado a cada sucursal específicamente. Desde aquí se pueden realizar solicitudes y manejo de compras o envíos de la sucursal, así como también consultar datos previos de la base de datos.

**Servicio de administración de ventas:** Se pueden modificar compras de clientes que desean un reembolso, se pueden aplicar descuentos por temporada, administrar ventas por sucursal, acceder a los datos de ventas generales y revisar la base de datos del historial de ventas a cada usuario.

**Servicio de logística:** Es el servicio que los encargados de logística utilizan para comunicar el estado de los envíos, recibir nuevos pedidos, solicitar la reposición de productos agotados o no disponibles en ciertas sucursales y desde aquí la base de datos se actualiza automáticamente al confirmar el recibimiento de los pedidos.

**Servicio de clientes web:** Es el servicio más extenso, ya que provee a los clientes del sitio web de todas las posibles herramientas necesarias para realizar sus pedidos y tener su cuenta creada. Desde crear y modificar sus usuarios, hasta el menú de accesibilidad que cada persona puede activar, hay un gran número de opciones que los usuarios pueden usar para tener una experiencia fluida y sencilla para realizar sus compras.



### 2.3. Características de los Usuarios

**Administrador del Sistema:** Su nivel educacional es profesional. Está completamente capacitado para dirigir el sistema y usar la información cuando sea necesario, así como también administrar las sucursales usando las diferentes funcionalidades específicas del sistema. Tiene conocimientos en programación y entendimiento de los sistemas usados para saber el uso que le debe dar a su sistema.

Perfil: Puede acceder a las bases de datos y modificarlas, puede crear nuevos perfiles de gerentes o trabajadores, puede acceder a la información específica de cada sucursal de forma individual, puede solicitar reposiciones de productos y revisar el inventario actual, puede gestionar los envíos y puede hacer modificaciones en casi cualquier ámbito.

**Gerente de Sucursal:** Su nivel educacional es alto y está capacitado para estar a cargo de una sucursal y tener la responsabilidad de usar el sistema que le otorga privilegios que los otros empleados no tienen. Tiene experiencia en estar a cargo de sucursales pequeñas previamente.

Perfil: Puede actualizar, modificar o revisar el historial tanto de trabajadores como los de las ganancias por día de la sucursal, así como también generar facturas y realizar devoluciones de manera extraprofesional a los clientes.

**Empleado de Ventas:** Su nivel de educación puede ser medio, sin necesidad de tener un título profesional. Debe estar capacitado para comunicarse correctamente con los clientes y resolver sus solicitudes. Debe saber cómo manejar el sistema para realizar un pedido, actualizar el sistema reportando una nueva venta y generar una boleta para cada cliente, así como también saber cómo hacer válidos sus pagos.

Perfil: Puede actualizar las ventas de la sucursal, modificar una compra en caso de que un cliente solicite una devolución, y también puede imprimir boletas de cada venta, así como revisar también el historial previo de ventas si un cliente regresa al poco tiempo después por cualquier motivo.

**Encargado de Logística:** Su nivel de educación puede ser medio, ya que ejerce un trabajo que requiere principalmente de experiencia en el área y saber cómo afrontar situaciones como que hayan calles cerradas o que no quede stock de un producto.

Perfil: Puede recibir envíos en su sistema y él debe recogerlos. Puede actualizar el estado del pedido constantemente. Una vez que indica que inició el recorrido, el GPS del vehículo mantiene una conexión con el sistema y muestra en todo momento su ubicación al usuario.

**Cliente Vía Web:** Puede no tener ningún estudio. Es un usuario común que se registra en el sistema y crea su cuenta para comenzar a hacer compras y pedidos.

Perfil: Puede crear, modificar o eliminar su perfil al momento de ingresar una cuenta al sistema. Puede realizar compras vía online y pedir que se envíen a su domicilio. Puede seleccionar el método de pago que más le convenga. Puede revisar su historial de pedidos y puede solicitar hablar con personal dedicado.

## 2.4. Restricciones

Una de las principales restricciones y algo que complica un poco el trabajo es mantener la integridad y encriptación total de las bases de datos incluso para los programadores que trabajen en armar este nuevo sistema.

Trae algunas limitaciones el tener que ser tan cuidadosos con esto, ya que se deben buscar otros métodos aparte de los convencionales para no tener los datos de los clientes en el poder de los desarrolladores.

Otro de los inconvenientes que puede presentar este cambio es el hecho de estar tan restringido a solo 1 lenguaje de programación. Se deben tener conocimientos en Java, ya que los Frameworks están hechos solo para este lenguaje, y este es el mismo que usaba el sistema monolítico anterior, por lo que se facilita llevar a cabo el cambio del sistema al trabajar con el mismo lenguaje previo.

## 2.5. Suposiciones y Dependencias

Los requisitos corresponden a una situación ideal y también flexible donde se le otorga el poder de las bases de datos y los privilegios especiales a personas de confianza, que garantizarán la protección de el sistema y sus componentes. El sistema es completamente dependiente tanto del Administrador como de los servidores donde se albergan las bases de datos y la información en general. Si cualquiera de estos factores tiene problemas, el sistema dejará de funcionar también y no se podrá garantizar tampoco la seguridad de la información.

## 2.6. Requisitos Futuros

Algunas posibles mejoras podrían ser nuevas modalidades para compras. Tener menús específicos para cada sucursal en lugar de que el cliente busque un producto en línea y los resultados sean de todas las sucursales. El cliente podría encontrar su coincidencia y pensar que es la sucursal que le queda más cerca, pero al ver los detalles de disponibilidad, verá que en realidad solo está en la sucursal más lejana. Un buen cambio sería tener un sitio para visitar la disponibilidad específica de cada sucursal de la empresa.

Otro posible requisito a futuro es la opción de que el cliente escoja qué servicio de transporte desea que le despache el pedido al momento de realizar su compra, así puede cotizar qué servicio llega antes, qué servicio tiene disponibilidad de envío, qué servicio es más barato y qué servicio le da más confianza.



### 3. Requisitos Específicos

#### 3.1 Requisitos comunes de las interfaces

##### 3.1.1 Interfaces de usuario

El sistema contará con una interfaz web responsiva. Se diseñará priorizando la experiencia de usuario y la accesibilidad.

**Estilo visual:** Colores suaves, letras legibles y contraste adecuado para facilitar la lectura.

**Compatibilidad:** Adaptable a distintos dispositivos como PC y también celulares Android e iOS.

**Elementos comunes:** Menú específico para empleados y para clientes barra de navegación superior que está todo el tiempo disponible.

**Pantallas previstas:**

- Inicio de sesión
- Registro de usuario
- Catálogo de productos
- Gestión de inventario
- Panel administrativo
- Seguimiento de pedidos
- Gestión de ventas y reportes
- Página de soporte y contacto

**Accesibilidad:** Modo accesibilidad con opciones de tener una fuente de letra más grande con alto contraste y también una opción de navegación por voz.

##### 3.1.2 Interfaces de hardware

El sistema requerirá conectividad con ciertos dispositivos para el funcionamiento en las sucursales físicas:

- Impresoras térmicas para la emisión de boletas y facturas. Estas pueden estar conectadas por USB o WiFi en caso de tener la posibilidad.
- Dispositivos de escaneo de códigos de barra con entrada USB o inalámbricos con pilas o con carga.
- Computadores y celulares para el uso por parte del personal de ventas, gerentes y administradores.

### 3.1.3 Interfaces de software

Software	Función	Formato
Oracle DataBase	Almacenamiento de datos	Conexión JDBC
Spring Boot	Framework para el desarrollo de la parte de Back-End	APIs REST con formato JSON
Google Maps	Planificación de rutas con GPS	HTTP
Spring Security	Seguridad y autenticación de usuarios	Tokens JWT
Transbank	Realización y procesamiento de pagos	API REST

### 3.1.4 Interfaces de comunicación

El sistema se comunicará entre sus componentes mediante APIs REST, asegurando una estructura escalable.

#### Características

- Protocolo: HTTP
- Formato de datos: JSON
- Seguridad: Autenticación mediante Tokens JWT con Spring Security
- Microservicios comunicados entre sí a través de Gateways API
- Los servicios externos como Google Maps y Transbank se integrarán por medio de sus propias APIs que son públicas.

## 3.2 Requisitos funcionales

### 1. Administrador del sistema:

#### Usuarios

- RF1.1: Crear
- RF1.2: Eliminar
- RF1.3: Actualizar.
- RF1.4: Desactivar usuario
- RF1.5: Eliminar usuario

#### Permisos

- RF1.6: Asignar permisos de acceso a las funciones del sistema
- RF1.7: Modificar permisos de acceso a las funciones del sistema

#### Monitorización

- RF1.8: Opciones para la visualización
- RF1.9: Sistema de notificación de errores
- RF1.10: Monitorización constante del sistema
- RF1.11: Monitorización activa del sistema para reportar errores
- RF1.12: Monitorización activa para prevenir posibles fallos

#### Datos

- RF1.13: Realizar copias de seguridad manuales
- RF1.14: Realizar copias de seguridad automáticas de manera periódica

### 2. Gerente de sucursal:

#### Gestión de Inventario

- RF2.1: Agregar productos del inventario
- RF2.2: Actualizar productos del inventario
- RF2.3: Eliminar productos del inventario
- RF2.4: Opción para reajustar el stock de productos por nombre
- RF2.5: Opción para reajustar el stock de productos por código

**Gestión de reportes**

- RF2.6: Creación de reportes de ventas
- RF2.7: Creación de reportes de inventario
- RF2.8: Creación de reportes de rendimiento de la sucursal

**Gestión de sucursales**

- RF2.9: Configurar nombre de la sucursal
- RF2.10: Configurar ubicación de la sucursal
- RF2.11: Configurar políticas de la sucursal
- RF2.12: Configurar empleados de la sucursal
- RF2.13: Configurar horarios de la sucursal
- RF2.14: Configurar perfiles de empleados

**Gestión de pedidos**

- RF2.15: Supervisar pedidos de productos por nombre
- RF2.16: Supervisar pedidos de productos por código
- RF2.17: Autorizar pedidos de productos por nombre
- RF2.18: Autorizar pedidos de productos por código

**3. Empleado de ventas****Registro de ventas**

- RF3.1: Realizar transacciones en el sistema
- RF3.2: Agregar un descuento a productos habilitados

**Devoluciones**

- RF3.3: Ingresar un reclamo al sistema
- RF3.4: Almacenar el reclamo con un código de reclamo
- RF3.5: El código de reclamo se puede registrar como devolución en caso de cumplir los requisitos

**Consulta de inventario**

- RF3.6: Acceso a la base de datos de productos
- RF3.7: Consultar si el producto está próximo a ser abastecido

**Facturas**

- RF3.8: Generar facturas electrónicas de cada venta individual
- RF3.9: Enviar facturas al correo electrónico de los clientes

**4. Logística****Gestión de envíos**

- RF4.1: Crear envíos
- RF4.2: Actualizar envíos
- RF4.3: Seguir los envíos en tiempo real

**Optimización de rutas de entrega**

- RF4.4: Revisar rutas
- RF4.5: Poder seleccionar una ruta sugerida generada automáticamente

**Estados de pedidos**

- RF4.6: Actualizar manualmente los pedidos
- RF4.7: Recibir la información cuando el transportista reporta exitosamente la entrega

**Gestión de proveedores**

- RF4.8: Guardar los proveedores en el sistema
- RF4.9: Actualizar proveedores
- RF4.10: Solicitar abastecimiento de productos sin stock

## 5. Cientes vía web

### Creación de cuenta

- RF5.1: Crear cuenta usando un correo electrónico y una contraseña.
- RF5.2: Agregar nombres a la cuenta
- RF5.3: Agregar apellidos a la cuenta
- RF5.4: Agregar dirección de correo electrónico
- RF5.5: Agregar ubicación a enviar sus pedidos
- RF5.6: Agregar métodos de pago Inicio de sesión
- RF5.7: Ingresar con una cuenta ya creada

### Recuperación de contraseña

- RF5.8: Enviar un código de recuperación al correo electrónico

### Navegación de productos

- RF5.9: Búsqueda de productos por nombre
- RF5.10: Búsqueda de productos por códigos
- RF5.11: Pantalla de inicio con el catálogo de productos general

### Realización de pedidos

- RF5.12: Enviar al cliente a otra página al decidir comprar
- RF5.13: Poder ingresar la cantidad que desea comprar
- RF5.14: Ingresar el método de pago
- RF5.15: Confirmar si la dirección de envío asignada a la cuenta es correcta
- RF5.16: Tener la opción de usar una nueva dirección de envío

### Consulta de historial

- RF5.17: Revisar el historial de productos comprados
- RF5.18: Revisar el historial de productos vistos sin comprar
- RF5.19: Revisar el estado en que están sus compras
- RF5.20: Las compras pueden estar en estado “por pagar”
- RF5.21: Las compras pueden estar en estado “pagado”
- RF5.22: Las compras pueden estar en estado “en camino”
- RF5.23: Las compras pueden estar en estado “recibido”

### Modificación de perfil

- RF5.24: Cambiar dirección predeterminada
- RF5.25: Añadir una nueva dirección
- RF5.26: Modificar detalles de pago Sistema de soporte
- RF5.27: Opción para solución de dudas frecuentes
- RF5.28: Tener una opción para contactar al personal dedicado

### Sistema de calificaciones

- RF5.29: Poder calificar los productos comprados con una puntuación de 1 a 5 estrellas
- RF5.30: Añadir fotos a la calificación
- RF5.31: Calificar individualmente el producto
- RF5.32: Calificar individualmente el envío
- RF5.33: Calificar la tienda en general
- RF5.34: Calificar el sitio web Descuentos
- RF5.36: Añadir cupones de descuento al momento de realizar una compra
- RF5.37: Añadir promociones activas al momento de realizar una compra

### Modo de accesibilidad

- RF5.38: Tener acceso en todo momento a una modalidad de accesibilidad
- RF5.39: La letra es más grande
- RF5.40: La letra tiene tonos más destacados
- RF5.41: Las imágenes son más grandes
- RF5.42: Acceder a un buscador por voz

### Filtros de búsqueda

- RF5.43: Habilitar la búsqueda de productos agotados
- RF5.44: Ordenar los productos por nombre de la A a la Z
- RF5.45: Ordenar los productos por nombre de la Z a la A
- RF5.46: Ordenar los productos por fechas de salida seleccionadas por el usuario

### 3.3 Requisitos no funcionales

#### Interfaz

- RNF1: Contar con una interfaz agradable a la vista
- RNF2: Interfaz con colores que no sean saturados
- RNF3: Interfaz con letra lo suficientemente grande para un usuario promedio

#### Errores

- RNF4: Poseer una pantalla de errores en caso de que suceda algo que provoque la caída del sitio web
- RNF5: Notificar al usuario si la razón del error es por el sistema

#### Búsqueda

- RNF6: Al hacer una búsqueda aparecen coincidencias similares

#### Envíos

- RNF7: El seguimiento de un envío muestra la ubicación del vehículo donde va el pedido
- RNF8: El operador que realiza el envío puede acceder a rutas en el sistema automáticamente generadas

#### 3.3.1 Requisitos de rendimiento

En principio, el sistema monolítico fue diseñado para que un 90% de las transacciones sean realizadas en 1 segundo, esto considerando 100 transacciones simultáneas en cada sucursal tanto físicamente como por vía web. Con el paso del tiempo estos porcentajes dejaron de cumplirse correctamente y se empezaron a presentar diversos retrasos al momento de hacer las solicitudes de compra.

El nuevo sistema garantiza que se puedan realizar un 95% de las transacciones en 1 segundo considerando 1000 usuarios simultáneos. Esto es un incremento del 1000% a comparación del sistema anterior, es decir, 10 veces más.

Gracias a la estructura basada en micro servicios, el nuevo sistema puede ser mucho más eficiente al momento de procesar estos pagos, y si en algún punto es necesario, se puede hacer una mejora al sistema actual para mantener siempre este 95% de garantía incluso con más usuarios, aunque los costos de un sistema como este con este nivel de movimiento incrementarían exponencialmente y mantener servidores y bases de datos tan extensas protegidas al 100% se comienza a volver un desafío.



### 3.3.2 Seguridad

Para mantener un sistema seguro y garantizar que no hayan fugas de datos o que alguien intente tomar información con intenciones maliciosas, se usa el sistema de Spring Boot llamado Spring Security.

El propio sistema hace un constante chequeo de información. Se mantienen estructuras muy concretas que no se deben modificar sin autorización. En caso de que esto ocurra, el sistema lo reconoce y bloquea sus datos hasta que el administrador resuelva el asunto. Él decide de qué forma se debe tratar una circunstancia donde se intentó ingresar a la información confidencial del sistema.

Los datos están guardados en archivos con formato JSON, por lo que este Framework permite el uso de Tokens JWT (JSON Web Token), los cuales son tokens volátiles designados a cada archivo. Esto funciona por identificaciones exclusivas y que no perduran en el tiempo, por lo que se requiere primero una identificación y con el tiempo esta misma cambia, ya que alguien podría tomar esta información tan importante y guardarla para hacer un ataque cuando sea un momento apropiado. El sistema JWT impide que esto suceda.

Además, el sistema cuenta con una pantalla de errores y, en caso de detectar cualquier inconveniente de seguridad, se le notifica inmediatamente al administrador y a los gerentes de sucursal.

### 3.3.3 Fiabilidad

El sistema garantiza que haya un funcionamiento correcto un 95% del tiempo. Cada vez que ocurre un error, se tiene un backup para prevenir que esto perjudique la experiencia de los usuarios y la página se caiga. Sin sistema, no hay actualización de ventas o siquiera ventas y eso no se puede permitir. Por eso cada vez que se deba activar un backup del sistema, se alertará al administrador que ocurrió un error y se le notificará el tipo de error.

### 3.3.4 Disponibilidad

El software presenta un 95% de disponibilidad, con un pequeño margen de error para las situaciones donde se deba recurrir a un backup temporal por cualquier razón, así como también se cuenta los momentos donde se está integrando una nueva modificación o se está implementando un cambio al sistema y por eso debe estar en mantenimiento.

### 3.3.5 Mantenibilidad

Periódicamente el sistema debe ser revisado por el administrador y se le debe hacer un mantenimiento en caso de presentar un problema recurrente o errores aleatorios sin una razón aparente.

Para esto es que el administrador puede contar con un desarrollador externo, aunque esto no debería ser del todo necesario si el administrador cuenta con los conocimientos. Lo que podría ser un impedimento para este mismo es la falta de tiempo por tener que encargarse de monitorizar constantemente el sistema y al personal designado para cada rol dentro de la empresa. Idealmente realizar una revisión de mantenimiento al finalizar un mes de registro de ventas y movimientos como devoluciones para verificar la integridad de los archivos y los micro servicios.

### 3.3.6 Portabilidad

El software es altamente escalable debido a su nueva naturaleza basada en micro servicios.

Lo que impedirá una gran flexibilidad al momento de portar el sistema a otra plataforma es el lenguaje de programación con el que funciona, ya que los Frameworks utilizados aquí están estrictamente ligados al lenguaje de programación Java, así como también sus APIs.

## 3.4 Otros Requisitos

Requisitos ligados a logística:

- El sistema advierte constantemente de los peligros o desvíos usando Google Maps
- El rastreo de ubicación se actualiza en tiempo real automáticamente vía GPS

Requisitos ligados a errores:

- Notificar al usuario si la razón del error es por un problema con el dispositivo del mismo usuario

Requisitos ligados a portabilidad:

- El sistema debe ser flexible y escalable, no puede depender estrictamente de ciertos servidores o fuentes externas, ya que su objetivo principal es ser modificable y escalable con el paso del tiempo.

## 4. Propuesta de Planificación

### 4.1 Descripción general acerca de la Planificación

La planificación del proyecto es de una duración total de 4 semanas, distribuidas entre análisis, diseño, desarrollo, pruebas y despliegue del nuevo sistema basado en microservicios. El equipo en este momento se divide entre 2 personas.

Se utilizaron herramientas como Trello o Jira para la organización de tareas y Git para colaborar con el código. Una forma de garantizar el correcto funcionamiento es hacer pruebas y análisis semanales de los avances.

Las condiciones necesarias para el proyecto incluyen:

- Acceso al sistema actual para analizar sus funcionalidades.
- Asignación de responsabilidades entre los integrantes.
- Espacios de prueba para la migración sin afectar al sistema anterior.
- Usar herramientas adecuadas para facilitar el trabajo y acortar los tiempos.

#### 4.1.2 Definición del Equipo de Trabajo

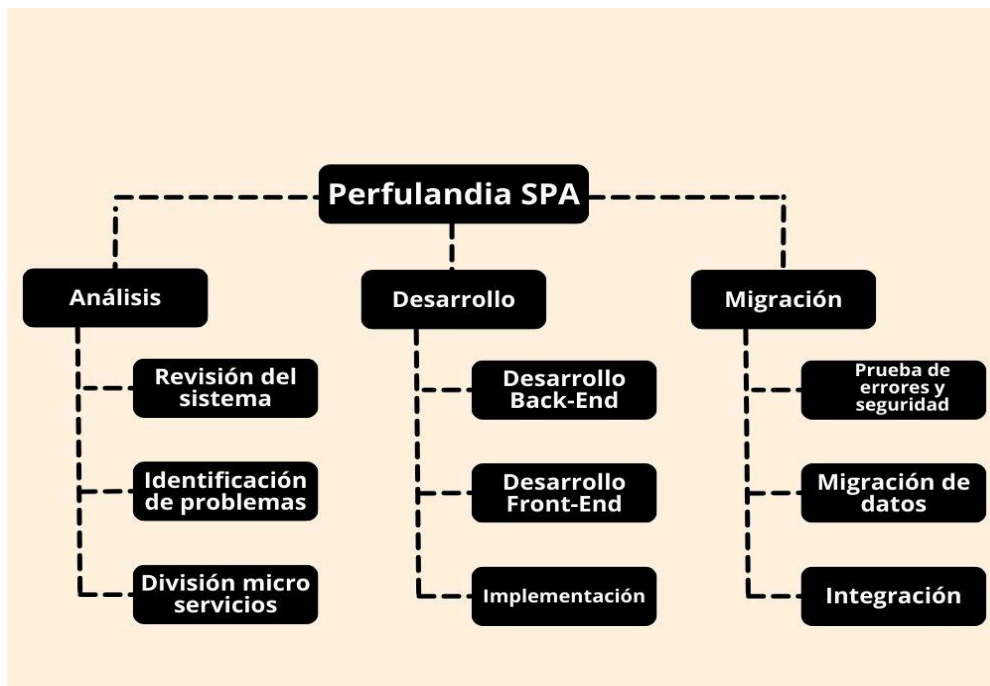
Nombres	Rol asignado	Funciones
Valeria Gallardo	Front-End	Analizar requisitos, diseñar interfaces y desarrollar el Front-End
Sebastian Fredes	Back-End	Realizar desarrollo del Back-End, de estructura de datos y microservicios

### 4.1.3 Definición de Actividades principales del Proyecto

Las principales actividades del proyecto están organizadas por fases siguiendo las prácticas del PMI y principios de Ingeniería de Software:

1. **Inicio del Proyecto:** Definición de objetivos, planificación, asignación de roles.
2. **Análisis del Sistema Actual:** Identificación de servicios y reconocimiento de problemas.
3. **Diseño de Arquitectura:** Definición de micro servicios y estructuras de datos.
4. **Desarrollo:** Construcción del nuevo sistema con el uso de Frameworks.
5. **Pruebas:** Verificación de rendimiento, funcionalidad y seguridad.
6. **Implementación:** Migración completa al nuevo sistema.

### 4.1.4 Diagrama EDT



#### 4.1.5 Carta Gantt

Perfulandia SPA - Plan de 1 mes				
Tareas / Tiempo	Semana 1	Semana 2	Semana 3	Semana 4
Análisis del sistema actual	✓			
Dividir en microservicios	✓			
Implementación de microservicios		✓	✓	
Migración de base de datos		✓	✓	
Pruebas y solución de errores			✓	
Optimización del sistema y revisión final				✓
Reemplazo del sistema monolítico				✓

En la primera semana el equipo debe comenzar con el análisis del sistema monolítico actual y reconocer sus servicios principales, reconocer los problemas del sistema y luego separar todos los servicios encontrados en microservicios, los cuales servirán como guía para desarrollar un nuevo sistema basado en microservicios. La primera semana es exclusivamente teórica y no se desarrolla ningún programa aún.

La segunda semana se debe seguir la base de microservicios identificados y el nuevo sistema propuesto en base a esto. Se comienza con el desarrollo del nuevo sistema, lo cual es normalmente tardado, por lo cual el equipo puede tomarse 2 semanas para hacer los cambios necesarios en el sistema anterior y migrar la base de datos también, para no perder la información.

En la semana 3 se terminan de hacer los últimos cambios y se pone a prueba el nuevo sistema para evitar posibles errores y tener un programa que cumpla con sus funciones de manera eficaz.

Tras revisar que todo funciona correctamente, en la semana 4 se realizan las optimizaciones finales para que el programa sea eficiente, y consiga un buen resultado con un mejor código.

Una vez completadas todas las tareas anteriores, lo último que queda por hacer es migrar del sistema anterior al nuevo, teniendo que poner la página en mantenimiento por un tiempo indefinido, hasta que el nuevo sistema se incorpore por completo y esté listo para su uso a un nivel más masivo que el sistema anterior.

#### 4.1.6 Resumen Costos del Desarrollo del Proyecto

Fase	Total de horas	Costo por hora	Total
Inicio del proyecto	8h	\$10.000	\$80.000
Análisis del sistema	16h	\$10.000	\$160.000
Diseño del sistema	20h	\$10.000	\$200.000
Desarrollo	60h	\$10.000	\$600.000
Pruebas	24h	\$10.000	\$240.000
Implementación	12h	\$10.000	\$120.000
<b>Total</b>	<b>140h</b>	-	<b>\$1.400.000</b>

Rol	Horas aproximadas	Costo por hora	Total aproximado
Encargado Back-End	80h	\$10.000	\$800.000
Encargado Front-End	60h	\$10.000	\$600.000
<b>Total general</b>	<b>140h</b>	-	<b>\$1.400.000</b>

## 5. Prototipo

### 5.1 Prototipo funcional con herramientas para prototipar (Link)

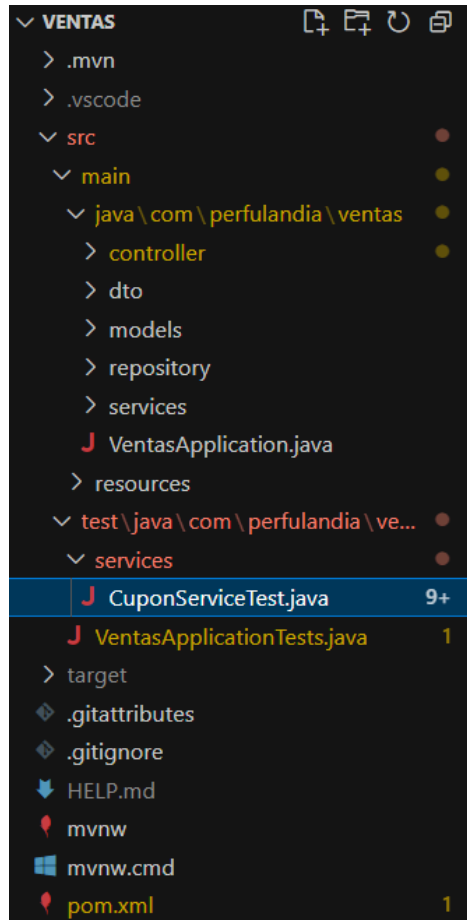
LINK Mockup Diseñado en Figma

<https://www.figma.com/design/mGX21esYSVzsbjDAn6wYKf/Perfulandia?node-id=0-1&t=sWXyZ3NDL3bxW9u6-1>

### 5.2 Plan de pruebas según normativa ISO25000 con planillas

En este apartado se hablará sobre pruebas unitarias dentro del sistema back-end desarrollado. La prueba unitaria JPA que servirá como ejemplificación en este caso será la API Ventas, usando un Cupón que es parte de este microservicio.

En este ejemplo se usará el apartado de Ventas Cupón, el cual tiene varios métodos asignados al mismo. Para comenzar, se crea una nueva carpeta llamada “services” dentro de la carpeta de test “test”. Luego, se crea el Cupón que se pondrá a prueba con sus métodos sin necesidad de conectarse a una Base de Datos activa o realizar pruebas manuales HTTP.





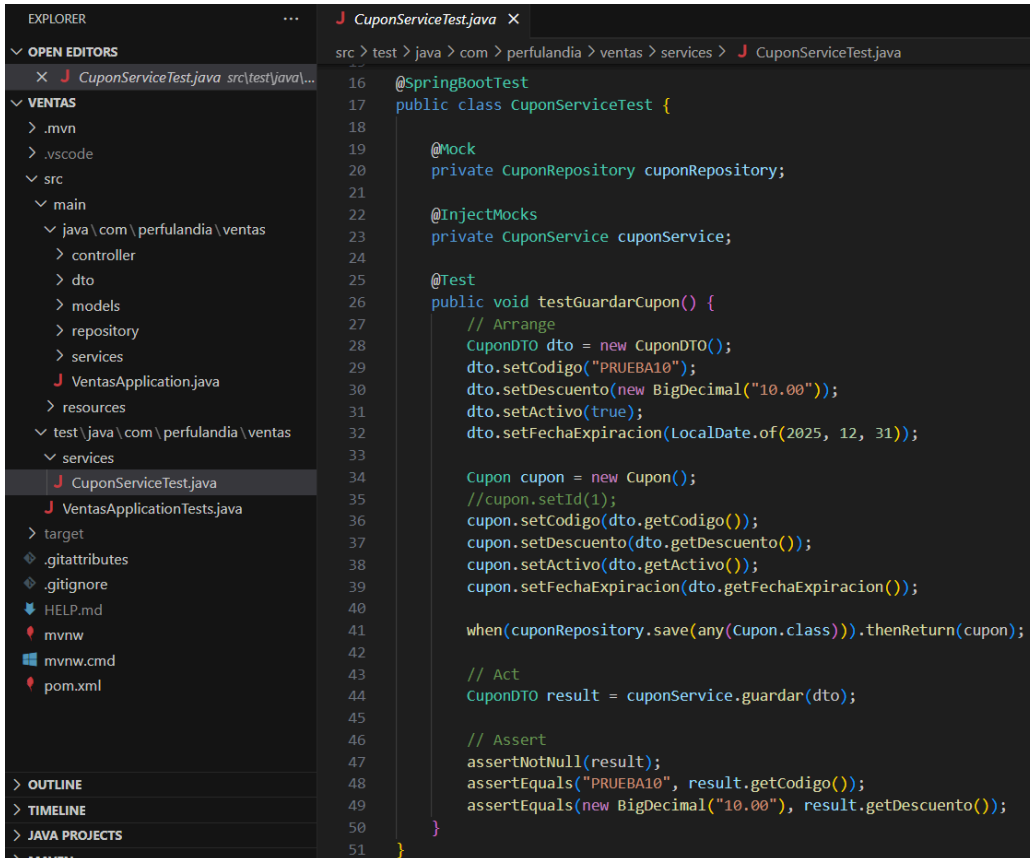
Se importan todas las librerías necesarias y se llaman a los archivos que requiere el sistema. JUnit es el Framework que se utilizará para realizar esta prueba de manera aislada del resto del sistema y Mockito simulará las dependencias necesarias del sistema.

```
1  package com.perfulandia.ventas.services;
2
3  import com.perfulandia.ventas.dto.CuponDTO;
4  import com.perfulandia.ventas.models.Cupon;
5  import com.perfulandia.ventas.repository.CuponRepository;
6  import org.junit.jupiter.api.Test;
7  import org.mockito.InjectMocks;
8  import org.mockito.Mock;
9  import static org.junit.jupiter.api.Assertions.*;
10 import static org.mockito.Mockito.*;
11 import org.springframework.boot.test.context.SpringBootTest;
12
13 import java.math.BigDecimal;
14 import java.time.LocalDate;
15
```

Se añaden también las dependencias de pruebas que se estarán utilizando dentro del archivo pom

```
67      <!-- Test: JUnit, Mockito, Hamcrest, AssertJ, etc -->
68      <dependency>
69          <groupId>org.springframework.boot</groupId>
70          <artifactId>spring-boot-starter-test</artifactId>
71          <scope>test</scope>
72      </dependency>
```

En este caso, el Cupón simulado posee una prueba de descuento del 10% y se pone a prueba el método “guardar()”, que almacena este nuevo cupón creado dentro del sistema y dentro del servicio más específicamente de CuponService.



```
16 @SpringBootTest
17 public class CuponServiceTest {
18
19     @Mock
20     private CuponRepository cuponRepository;
21
22     @InjectMocks
23     private CuponService cuponService;
24
25     @Test
26     public void testGuardarCupon() {
27         // Arrange
28         CuponDTO dto = new CuponDTO();
29         dto.setCodigo("PRUEBA10");
30         dto.setDescuento(new BigDecimal("10.00"));
31         dto.setActivo(true);
32         dto.setFechaExpiracion(LocalDate.of(2025, 12, 31));
33
34         Cupon cupon = new Cupon();
35         //cupon.setId(1);
36         cupon.setCodigo(dto.getCodigo());
37         cupon.setDescuento(dto.getDescuento());
38         cupon.setActivo(dto.getActivo());
39         cupon.setFechaExpiracion(dto.getFechaExpiracion());
40
41         when(cuponRepository.save(any(Cupon.class))).thenReturn(cupon);
42
43         // Act
44         CuponDTO result = cuponService.guardar(dto);
45
46         // Assert
47         assertNotNull(result);
48         assertEquals("PRUEBA10", result.getCodigo());
49         assertEquals(new BigDecimal("10.00"), result.getDescuento());
50     }
51 }
```

## Explicando la lógica dentro de este testeo JUnit:

- 1- El sistema toma un objeto del archivo CuponDTO.
- 2- Este objeto es convertido a una entidad Cupon.
- 3- Es guardado con CuponRepository.
- 4- El objeto que fue guardado es retornado como un CuponDTO.

Si la prueba unitaria resulta exitosa, se va a demostrar que el método “guardar()” funciona correctamente según los requerimientos del sistema, ya que se pone a prueba la capacidad de este test para que el CuponService convierta un DTO a Entidad, llamando y conectándose al Repositorio y luego retornando un DTO con los mismos datos que fueron enviados.

### Ahora hablando específicamente sobre el código de manera más detallada:

“@SpringBootTest” carga el contexto de Spring para realizar la prueba.

“@Mock” crea un objeto simulado del repositorio para que no haya necesidad de acceder a una Base de Datos real.

“@InjectMocks” crea una instancia de CuponService e inyecta el “mock” del repositorio al archivo.

“@Test” registra el método “testGuardarCupon()” como una prueba que debe ser ejecutada.

“when” y “thenReturn” simulan el comportamiento del método “save()” del repositorio real.

Ahora se realiza la prueba unitaria para comprobar el funcionamiento de este método aislado. De esta manera se puede saber también si el método es el problema en caso de arrojar error o si es un problema con la conexión a la Base de Datos o si es un problema de relación con algún otro método, quizá por falta de una correcta conexión o por un problema externo a la API que se está probando, como un error de puerto.

**BUILD SUCCESS**

-----  
Total time: 12.982 s

Finished at: 2025-06-14T22:10:49-04:00  
-----

Y con esto se comprueba finalmente que la prueba JUnit ha sido realizada correctamente y que el método asignado a esta API para probar no tiene problemas al menos de forma individual y aislada del resto de métodos y conexiones externas.

Se realizó este mismo procedimiento para el resto de APIs, lo que se presenta en este informe es solo uno de esos ejemplos y una muestra de que tiene un correcto funcionamiento.

Estas pruebas en vez de ser realizadas en la Terminal, aparecen en la Debug Console, ya que no tienen un real impacto sobre el sistema al ser simulaciones dentro del entorno controlado usado por el Framework asignado, en este caso JUnit, que es sencillo de usar porque ya viene implementado con Spring Boot.

### Planilla con información de Casos de pruebas para este ejemplo:

Test	Método probado	Descripción breve	Resultado esperado	Resultado obtenido
testGuardarCupon()	guardar()	Verifica que el método convierta correctamente un DTO a entidad y lo retorne	DTO con los mismos datos que el guardado	Correcto

**Anexo a plan de pruebas ágiles:**

[https://docs.google.com/document/d/128a1wpL5js7ha1MMsUS\\_pXhUnGSq6IOzNEdUPJGtP78/edit?usp=drive\\_link](https://docs.google.com/document/d/128a1wpL5js7ha1MMsUS_pXhUnGSq6IOzNEdUPJGtP78/edit?usp=drive_link)

**Planilla de Casos de pruebas sencillos:**

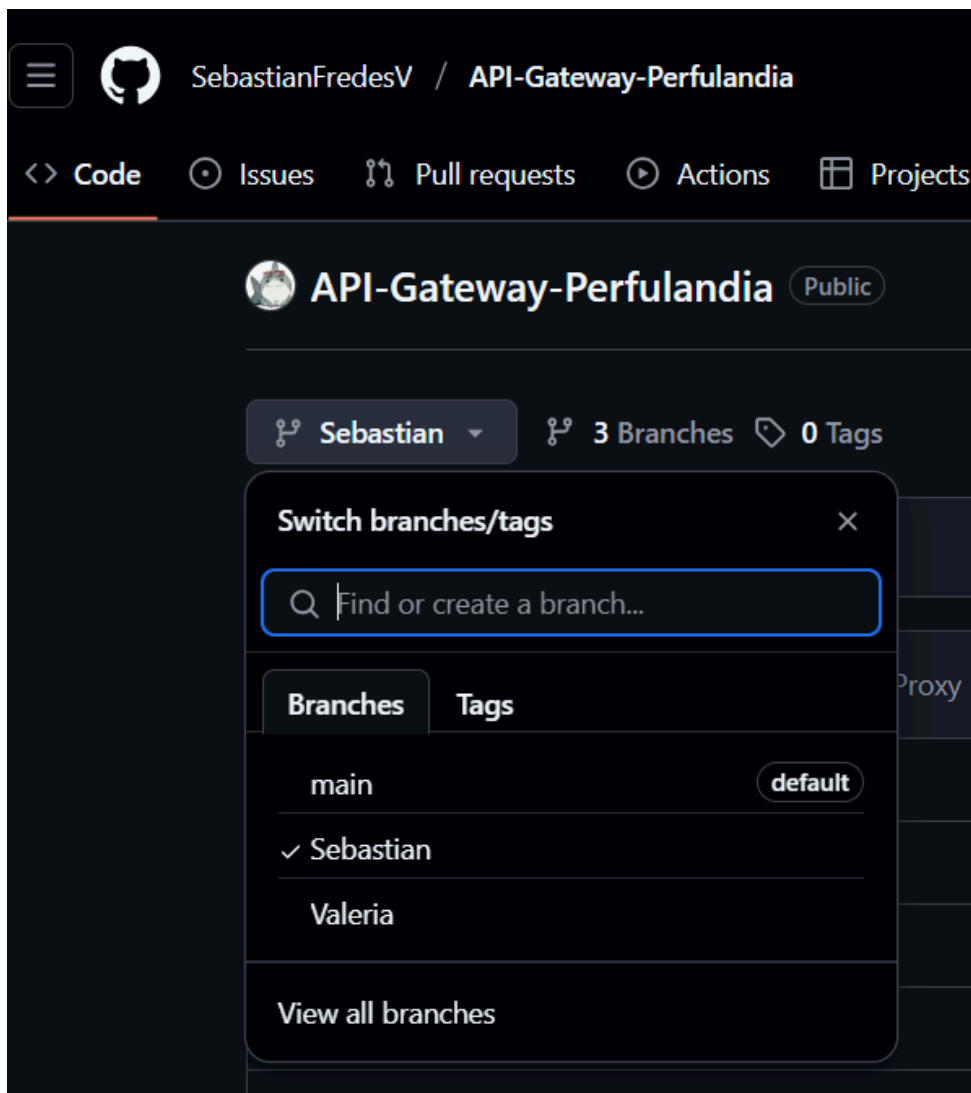
Nro	Nombre/Identificador	Caso de Prueba	Requisito Asociado	Entradas	Resultados Esperados	Prioridad
1	CP_Add1	Crear nuevo usuario	RF1	Nombre, email, rol	Usuario registrado con éxito	Alta
2	CP_Add2	Generar venta con descuento	RF5	Producto, descuento	Venta registrada y descuento aplicado	Media
3	CP_Add3	Ver historial de pedidos como cliente	RF10	Usuario logueado	Lista de pedidos previos	Baja
4	CP_Add4	Acceso restringido a funciones de administrador	RNF3	Usuario sin permisos	Acceso denegado	Alta

**Planilla de Reporte de defectos:**

No/ID	Fecha	Probador que lo identificó	Título	Elemento	Resultado esperado	Resultado real	Gravedad	Prioridad	Estado
1	07-06-2025	Sebastian Fredes	Falta de método delete en API Inventario	API Inventario	Se selecciona un ID de Inventario existente y se utiliza el método DELETE en Postman para acceder a la Base de Datos y eliminar el Inventario seleccionado por completo	Se muestra un mensaje de error 400, ya que el método delete no había sido añadido en la API de Inventario, por lo que el sistema no sabe qué se le está pidiendo	Media	Media	Arreglado
2	10-06-2025	Sebastian Fredes	Ausencia de implementación de Token de seguridad JWT	Sistema general de Perfulandia	Las solicitudes HTTP como POST, PUT o DELETE solicitan un Token de acceso para verificar la identidad y rol del usuario. Solo administradores pueden usare estos métodos	Cualquier usuario puede realizar cambios sin necesidad de verificación y sin necesidad de ingresar Token de seguridad	Muy alta	Muy alta	En proceso
3	08-06-2025	Valeria Gallardo	Puertos de servidor duplicados	API Vendedores	Se conecta a la API Gateway mediante el puerto designado y realiza las solicitudes HTTP a través de este	La API no inicia, ya que comparte puerto con otra API anteriormente creada y su puerto no puede ser utilizado al tener otro programa usando este mismo	Alta	Media	Arreglado
4	04-06-2025	Sebastian Fredes	Falta de dependencias en pom	API Ventas	La API Ventas genera automáticamente los métodos constructores con getters y setters	El sistema presenta errores al no tener las dependencias Lombok añadidas al archivo pom de la API	Alta	Alta	Arreglado
5	09-06-2025	Valeria Gallardo	Inicialización errónea de archivo	API Reportes	La API corre con normalidad y se conecta a la Base de Datos del sistema	La API no inicia debido a que fue inicializada en una carpeta externa que no posee el archivo pom integrado	Alta	Alta	Arreglado

### 5.3 Registro del control de cambios de versiones (con GitHub)

Muestra de trabajo colaborativo con ramas, aplicando estructura y buenas prácticas para el desarrollo del proyecto:



Cada integrante del grupo realizó una API diferente, por lo que ambos perfiles tienen separaciones de API, mostrando quién realizó cada una de manera individual luego de hacer una reunión donde se decidió cómo sería la división de trabajo, utilizando el trabajo colaborativo con registros de historial y ramas principales en la API Gateway.

