ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

DESARROLLO DE SISTEMA PARA LA GESTIÓN DEL INVENTARIO EN FARMECC

DESARROLLO DE UN SISTEMA DE ESCRITORIO

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE

BRYAN ARMANDO QUISAGUANO CASA

DIRECTOR: LOARTE CAJAMARCA BYRON GUSTAVO

DMQ, septiembre 2022

CERTIFICACIONES

Yo, Bryan Armando Quisaguano Casa declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

BRYAN ARMANDO QUISAGUANO CASA

bryan.quisaguano@epn.edu.ec

bryanquisaguano@gmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por Bryan Armando Quisaguano Casa, bajo mi supervisión.

Ing. Byron Loarte, MSc.

DIRECTOR

byron.loarteb@epn.edu.ec

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Bryan Armando Quisaguano Casa

DEDICATORIA

Con mucho entusiasmo, dedico este proyecto a mis padres, quienes me han brindado su apoyo incondicional durante toda mi vida estudiantil y es gracias a ellos que esto ha sido posible, cada uno de mis logros obtenidos y los que llegue a obtener en el futuro serán gracias y para ellos. Además, a mis hermanos Alex y Liseth que han sido fuente de inspiración para poder seguir adelante, queriendo convertirme en un ejemplo para ellos.

Bryan Armando Quisaguano Casa

AGRADECIMIENTO

Doy gracias principalmente a Dios por las bendiciones dadas tanto a mi familia como a mí, gracias a mis padres, hermanos y amigos que han estado a mi lado en este largo camino, que si bien estuvo lleno de tropiezos todos han estado apoyándome en todo momento, cada momento que he pasado junto a ellos me ayudó a crecer como persona. Por lo cual, les estaré eternamente agradecido.

Bryan Armando Quisaguano Casa

ÍNDICE DE CONTENIDO

RESUMEN

El crecimiento que tiene una empresa de comercialización de productos médicos conlleva al aumento de responsabilidades por parte de los propietarios y si a futuro existe la posibilidad de expansión ya sea de sucursales, clientes y ventas, en este sentido es importante tener una correcta gestión de todos sus recursos. Por lo cual, a raíz del crecimiento que ha tenido la cadena de farmacias “FARMECC” y la apertura de una nueva sucursal ocasiona una serie de dificultades en lo que respecta a la administración de los recursos existentes en cada una de sus sucursales. En ese sentido, y con el fin de apoyar a esta empresa y a toda su comunidad, en el presente proyecto de integración curricular se ha desarrollado un sistema de escritorio la cual está dividido en dos estratos, la primera es que les permita a los administradores realizar una fácil gestión y control sobre su inventario, ventas, personal, médicos a disposición, sucursales, generación de reportes, etc. Y, por otra parte, permite generar varios *endpoints* posibilitando la interacción mediante una aplicación del lado del cliente ya sea web o móvil, para que de esta manera la presente empresa dedicada a la salud tenga una presencia digital en Internet gracias al uso de la tecnología.

El presente Trabajo de Integración Curricular está estructurado de la siguiente manera: en la primera sección, se detallan los antecedentes, objetivos, alcance del proyecto junto con el marco teórico. En la segunda sección, se puntualiza la implementación de la metodología *Scrum*, prototipos y herramientas utilizadas para el desarrollo del sistema de escritorio y *endpoints*. En la tercera sección, se presenta las actividades por iteración y resultados de cada *Sprint*. Finalmente, en la cuarta sección se especifica las conclusiones y recomendaciones que se han obtenido.

PALABRAS CLAVE: inventario, sistema de escritorio, *endpoints,* gestión, *Scrum*.

ABSTRACT

The growth of a medical products marketing company leads to increased responsibilities on the part of the owners and if in the future there is the possibility of expansion of branches, customers and sales, in this sense it is important to have a correct management of all their resources. Therefore, as a result of the growth that the "FARMECC" pharmacy chain has had and the opening of a new branch, it causes a series of difficulties with regard to the administration of the existing resources in each of its branches. In this sense, and in order to support this company and its entire community, in this curricular integration project, a desktop system has been developed which is divided into two layers, the first is that it allows administrators to carry out easy management and control over your inventory, sales, personnel, available doctors, branches, generation of reports, etc. And, on the other hand, it allows the generation of several endpoints, enabling interaction through a client-side application, whether web or mobile, so that this company dedicated to health has a digital presence on the Internet thanks to the use of technology.

This Curriculum Integration Work is structured as follows: in the first section, the background, objectives, scope of the project are detailed along with the theoretical framework. In the second section, the implementation of the Scrum methodology, prototypes and tools used for the development of the desktop system and endpoints are specified. In the third section, the activities per iteration and results of each Sprint are presented. Finally, the fourth section specifies the conclusions and recommendations that have been obtained.

KEYWORDS: inventory, desktop system, endpoints, management, Scrum.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

Una parte importante de los negocios de pequeñas empresas es la Administración, si bien se puede decir que los negocios y empresas siempre llevan un control y una administración de negocios. No obstante, muchos de estos negocios o empresas pequeñas emergentes carecen de un registro de movimientos tanto económicos como materiales, no tienen un sistema automatizado de ingresos, gastos, *stock* y en muchos casos estos controles se hacen de manera manual, lo cual ocasiona inconvenientes durante el proceso [1].

La falta de un control de inventario por parte de empresas de comercialización de medicamentos puede incurrir en problemas tales como faltantes en productos médicos los cuales pueden llegar a ser vitales para la salud de sus clientes. Por otra parte, si se manejan existencias excesivas esto limita la capacidad de inversión de la empresa, además de ocasionar pérdidas económicas al no ser vendidas antes de su fecha de caducidad [2].

En este sentido, es indispensable que todo negocio de cualquier tipo y tamaño pueda disponer de una buena gestión de inventarios. Hoy en día, gracias a los avances tecnológicos estas herramientas pueden acompañar a la administración de estos negocios [3]. Un sistema de gestión apropiado para el negocio brinda la facilidad de obtener indicadores, estadísticas, automatización de procesos y el manejo integro de la información [4]. Además, las empresas buscan tener soluciones rápidas y eficientes basadas en tecnología, ya que el desarrollo de estos sistemas es hecho a la medida y de acuerdo con los requerimientos de cada empresa solventando sus necesidades. Para el manejo del inventario estos sistemas permiten supervisar la evolución de un producto, el ciclo de vida que este tenga y la disponibilidad del mismo. Con esto se puede realizar una mejora en los procesos internos de la empresa en cuanto a eficiencia y tener un control optimizado en las operaciones realizadas [5].

En base a lo mencionado anteriormente, este proyecto de integración curricular tiene como propósito el desarrollo de un sistema de escritorio codificado en base a *Java* como lenguaje de programación para la cadena de farmacias “FARMECC” con sede principal en Quito, con la finalidad de brindar una herramienta que solucione problemas, automatice ciertos procesos administrativos, gestión dentro de la cadena, además de la creación de varios *endpoints* para el consumo de un sistema *web* destinado al control de movimientos económicos y materiales de cada sucursal de la cadena de farmacias.

1.1 Objetivo general

Desarrollar un sistema para la gestión del inventario en FARMECC.

1.2 Objetivos específicos

1. Identificar los requerimientos necesarios para el desarrollo del sistema de escritorio.
2. Desarrollar la arquitectura de datos para el sistema de escritorio de acuerdo a los requerimientos.
3. Codificar cada uno de los módulos de acuerdo a los requerimientos.
4. Generar *endpoints* de acuerdo a los requerimientos.
5. Probar el sistema de escritorio y *endpoints* verificando su correcta funcionalidad.

1.3 Alcance

Hoy en día las empresas deben contar con un buen control y gestión de cada uno de los movimientos económicos y materiales que se tengan dentro de una organización para obtener mayor eficiencia en su funcionamiento, para esto una pieza fundamental es el uso de tecnologías de la información o *software* de control, los cuales ayudan a mantener un mejor control de entrada y salida de productos, obteniendo un registro sencillo de cada movimiento que se ha realizado, además de encargarse de tareas administrativas o de cualquier operación realizada dentro de la empresa [6].

Con la finalidad de implementar una solución tecnológica la cual otorgue grandes ventajas, en el presente trabajo se ha desarrollado un sistema de escritorio el cual brinda las funcionalidades para poder gestionar el inventario de forma adecuada, un correcto control del *stock*, proveedores, sucursales, personal, médicos, cotizaciones y generando resúmenes de contabilidad orientado a los dueños de la red de farmacias. Por otra parte, para garantizar un manejo adecuado de la información y aplicación por parte de los usuarios finales, se ha implementado diferentes perfiles de usuario, definidos a continuación:

El perfil empleado permite:

- Iniciar y cerrar sesión.
- Visualizar, agregar y editar inventario.
- Realizar ventas.

- Generar reporte de cierre de caja.

El perfil administrador permite:

- Iniciar y cerrar sesión.
- Gestionar información general.
- Gestionar productos.
- Gestionar sucursales.
- Gestionar personal.
- Gestionar ventas.
- Gestionar médicos.
- Generar reporte de cierre de caja.
- Generar reportes de contabilidad en un período de días dados.
- Generar reportes destinados al Ministerio de Salud Pública.

Por último, se ha implementado un apartado para generación de varios *endpoints RESTful* para su consumo desde un sistema *web*, los cuales se describen a continuación:

Información general

- Creación de varios *endpoints* para visualizar página informativa.

Inicio de sesión

- Creación de varios endpoints para iniciar sesión y cerrar sesión.

Sucursales

- Creación de varios *endpoints* para gestionar sucursales.

Catálogo

- Creación de varios *endpoints* para la visualización de productos.

Productos

- Creación de varios *endpoints* para realizar cotizaciones de productos.
- Creación de varios *endpoints* para visualizar disponibilidad de *stock* de productos.

Ventas

- Creación de varios *endpoints* para la visualización de ventas por sucursales.

Personal

- Creación de varios *endpoints* para visualizar el personal.

Reportes

- Creación de varios endpoints para visualizar reportes de cada sucursal.

Pedidos

- Creación de varios *endpoints* para gestionar pedidos con proveedores.

1.4 Marco teórico

Una disciplina derivada de la Ingeniería es la Ingeniería de *software* que tiene como finalidad el desarrollo de sistemas informáticos de *software*, en general estos sistemas son intangibles y abstractos, este tipo de sistemas no se encuentran limitados por materiales o procesos de manufacturación, lo cual hace que el potencial del *software* no tenga limitaciones físicas y esta libertad del *software* hace que los desarrollos lleguen a ser muy complejos y difíciles de entender para personas que no tengan experiencia en esa rama [7].

La calidad de un *software* tiene diferentes perspectivas y su definición varía dependiendo del autor, para Pressman calidad en un *software* es “la concordancia con los requisitos funcionales y de rendimientos explícitamente establecidos, estándares de desarrollo explícitamente documentados y características implícitas que se espera de todo *software* desarrollado profesionalmente” [8]. Para el caso de un desarrollo particular se puede tomar como calidad a que cumpla con los requisitos que se han establecido, además que sea resistente a fallos cumpliendo así con todas las necesidades del cliente.

Una forma de asegurar la calidad en un *software* independientemente del ambiente en el cual se desarrolla es la implementación de una metodología, la cual ayuda al equipo de desarrollo a seguir parámetros y tener un control sobre tiempos de entrega [9]. Existen diferentes metodologías de desarrollo enfocados en sistemas de escritorio como: *Scrum, XP, KANBAN*, entre otras, las cuales tienen diferentes características y dependiendo del proyecto se pueden implementar.

Un sistema de escritorio es un tipo de *software* que funciona en un ordenador, sin tener conexión a Internet como necesidad, este tipo de *software* se almacena en la memoria del ordenador y se ejecuta a través del sistema operativo. La seguridad en estos tipos de *software* se basa en generar respaldos y copias de seguridad además de modelos lógicos ejecutados para evitar un mal uso de esta [10].

La implementación de un sistema que aloje todos los datos recopilados por el sistema es fundamental, por lo cual la implementación de una base de datos relacional (SQL) resulta ser critica, este tipo de base de datos como su nombre lo indica contiene datos que se relacionan entre sí, es una forma de representar datos en tablas, en donde cada tabla contiene un identificador único el cual facilita realizar relaciones entre 2 puntos de datos [11].

AlwaysData, es una plataforma de alojamiento de bases de datos SQL y NoSQL en la nube, el cual otorga un Sistema Gestor de Bases de Datos Relacional (SGBD) para la gestión y almacenamiento de todos los datos de manera segura todo ello gracias al uso de un panel administrativo [12].

Uno de los lenguajes de programación más utilizados a nivel de *software* empresarial es *Java,* el cual a su vez brinda grandes capacidades para realizar aplicaciones de escritorio, entornos *web* e incluso aplicaciones móviles, siendo así uno de los lenguajes más fiables y rápidos. Por último, brinda una gran facilidad para los desarrolladores para que escriban su código una sola vez y lo ejecuten en cualquier dispositivo independientemente del *hardware* y *software*, haciéndolo altamente portable [13].

Spring, es un *Framework* popular de *Java* de código abierto, el cual es empleado para crear todo tipo de aplicaciones basadas en *Java,* permite desarrollar código liviano, reutilizable y de alto rendimiento. Este *Framework* brinda soporte a nivel de aplicación junto con un modelo de configuración realmente completo, lo cual hace que los desarrolladores únicamente se enfoquen en la lógica que requiere la aplicación [14].

Spring Boot, es un proyecto de *Spring* que proporciona varias herramientas que facilitan la construcción y configuración de una aplicación del lado del servidor específicamente un *backend* y que el despliegue de la aplicación sea por medio de un servidor embebido *Tomcat,* permitiendo al desarrollador librarse del proceso de realizar configuraciones tediosas y enfocarse únicamente en el desarrollo de la lógica de su sistema [15].

Hibernate ORM, es una herramienta de mapeo objeto-relacional o por sus siglas (ORM) la cual provee una solución eficiente y completa al problema de persistencia de datos

ocurridos en *Java*, esta herramienta facilita el mapeo de una base de datos y los objetos de la aplicación [16].

Aplicación de Interfaz de Programación por sus siglas API, son funciones y procedimientos que integran sistemas como capa de abstracción de datos, es empleada para realizar intercambio y transferencia de datos entre sistemas mediante el protocolo HTTP, estableciendo comunicaciones mediante solicitudes y respuestas entre el cliente y servidor [17]. Cabe recalcar que actualmente existen 2 tipos de API's:

- **Privadas**: tiene acceso restringido a todos sus datos.
- **Publicas**: se puede acceder de forma libre a todos sus datos.

El *Backend* es el medio de acceso a datos la cual se encuentra y se ejecuta del lado del servidor, se encarga de ejecutar procesos y funcionalidades como: comunicación con la base de datos, comunicación con el *hosting* de alojamiento, integración con otros sistemas, entre otros. Además, contiene toda la lógica del negocio la cual no es visible para el usuario final [18].

2 METODOLOGÍA

El estudio de casos es un método de investigación el cual se basa en recopilar información y realizar análisis de la misma, por lo cual se lo considera como el estudio particular de un caso específico con la finalidad de entender su actividad en una circunstancia determinada. Estos estudios se presentan de forma disciplinada y cualitativa para un caso particular de investigación, destacando así la creación de hipótesis y teorías sustentadas, todo basado en una investigación profunda [19].

Por esta razón, el presente proyecto de integración curricular implementa un estudio de casos, ya que inicia con la investigación sobre los problemas presentados en la gestión manual de inventarios de empresas, negocios pequeños y la falta de automatización de procesos internos. Con lo cual, se ha llevado a cabo el desarrollo de un sistema de escritorio para la cadena de farmacias en el cual puedan gestionar inventarios y procesos internos relacionados con persona, reportes, sucursales, entre otros. Optimizando de esta manera los procesos mediante la implementación de un *software* y con el apoyo de la tecnología.

2.1 Metodología de Desarrollo

Las metodologías empleadas para la codificación de un producto *software* son estándares de trabajo que ayudan a ordenar, planificar y guiar todo el proceso que conlleva el desarrollo de un *software.* Estas a su vez permiten la división de tareas y responsabilidades con enfoque en la productividad y eficiencia [20].

Las metodologías ágiles dentro del desarrollo de un *software* dan la facilidad de adaptar el marco de trabajo de acuerdo a las condiciones específicas del proyecto, obteniendo beneficios tales como: flexibilidad, adaptabilidad, inmediatez de respuesta y aprendizaje rápido. Además, se obtiene una participación activa del cliente, al realizar entregables constantes y obtener retroalimentación por su parte; mejorando así el proceso y el producto final esperado [21]. En este sentido, se ha desarrollado el proyecto mediante la metodología *Scrum,* así mismo se describe a continuación la implementación de cada una de las fases de esta metodología en el sistema de escritorio y en los *endpoints*.

Roles

Los roles pertinentes del presente proyecto de titulación son desempeñados por los miembros del equipo, garantizando así, una óptima comunicación con el cliente y un correcto desarrollo del *software* [22]. Cada uno de los miembros se enfoca en desarrollar una parte específica del proyecto teniendo en cuenta el objetivo general y la relación que

debe tener cada una de sus componentes [23]. En ese sentido, cada uno de los roles que se han implementado en el proyecto se describen a continuación:

Product Owner

Es el propietario del *software* o la persona con mayor autoridad dentro del grupo de trabajo, se encarga de supervisar y proporcionar datos sobre el manejo del proyecto [23]. En este sentido, en la **TABLA I** se evidencia a la persona que desempeña este rol.

Scrum Master

La persona asignada para desempeñar este rol se encarga de que el equipo de desarrollo siga cada una de las prácticas y valores que se han descrito dentro de la metodología; además, de guiar y orientar al equipo de trabajo [23]. La persona encargada de este rol se puede observar en la **TABLA I.**

Development Team

Es el equipo de profesionales autoorganizados y multifuncionales encargados de presentar el proyecto "terminado" y crear un incremento del producto *software*. Además, los miembros de este equipo realizan las tareas proporcionadas por el *Product Owner* y el *Development Team* no puede tener sub-etiquetas ni se puede subdividir en grupos [24]. En tal sentido, en la **TABLA I** se evidencia la manera en la cual se ha distribuido cada uno de los roles.

TABLA I: Designación de roles para el proyecto.

Rol	Integrantes
Product Owner	Lcdo. Efraín Caza.
Scrum Master	Ing. Byron Loarte, MSc.
Development Team	Bryan Quisaguano

Artefactos

La metodología *Scrum* proporciona elementos denominados artefactos, los cuales se encuentran diseñados para garantizar sustento y fiabilidad dentro del equipo de desarrollo, para así obtener un correcto manejo de información, es decir, se evita una mala comunicación [24]. Estos elementos que se han implementado en el desarrollo del proyecto se describen a continuación:

Recopilación de requerimientos

En base al marco definido por *Scrum*, la Recopilación de requerimientos es una pieza fundamental para realizar un *software* de calidad. Cada uno de los requerimientos se generan de acuerdo a las funcionalidades y características que el *software* debe poseer, además de las necesidades del cliente [25]. Por tal motivo, la **TABLA II** presenta la plantilla que se ha establecido en la Recopilación de requerimientos, mientras que la tabla completa con cada uno de los requerimientos que se han obtenido se presenta en el **ANEXO II** del presente documento.

TABLA II: Plantilla para la obtención de la Recopilación de requerimientos.

Recopilación de requerimientos		
Tipo de sistema	**ID-RR**	**Enunciado del Ítem**
Sistema de escritorio	**RR001**	Como usuario administrador y empleado necesitan realizar lo siguiente: • Iniciar sesión. • Cerrar sesión.
	RR002	Como usuario administrador, necesita realizar lo siguiente: • Gestionar personal.

Historias de Usuario

Una vez que se han establecido cada uno de los requerimientos necesarios para el sistema, se fijan las funciones; estas son las Historias de Usuario, las cuales son los requisitos detallados de manera general, de acuerdo a las necesidades y perspectivas del cliente y en donde se asigna una prioridad. Su propósito es formular cómo se desarrollan los elementos y su valor para el cliente [26]. Esto ayuda al equipo a saber qué se está creando y su peso en el proyecto. Por lo cual, en la **TABLA III** muestra la implementación de una Historia de usuario para el desarrollo del presente proyecto, mientras que las demás tablas completas se encuentran en el **ANEXO II** del presente documento.

TABLA III: Historia de usuario 1 – Iniciar sesión y cerrar sesión.

	HISTORIA DE USUARIO
Identificador (ID): HU001	**Usuario**: Administrador, empleado.

Nombre Historia: Iniciar sesión y cerrar sesión.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Iteración Asignada: 1	
Responsable (es): Bryan Quisaguano	
Descripción: El usuario administrador y empleado tiene la posibilidad de iniciar sesión para poder interactuar con los módulos correspondientes de acuerdo a su rol. Por otra parte, los usuarios tienen la posibilidad de dejar de interactuar con el sistema mediante el cierre de sesión.	
Observación: En caso de que los usuarios ingresen su usuario y contraseña de manera incorrecta se presenta de un mensaje de error.	

Product Backlog

Es una tabla donde se encuentran todos los requisitos que se han obtenido; pues, abarca una descripción de las actividades y sus tareas a realizar para la ejecución de cada requisito. Su finalidad es la identificación de las necesidades del producto; de esta forma, se mejora la usabilidad [27]. Por consiguiente, la **TABLA IV** presenta la plantilla que se ha implementado para listar el *Product Backlog*, mientras que el listado completo se presenta en el **ANEXO II** del presente documento.

TABLA IV: Plantilla para listar el *Product Backlog*.

ID – HU	HISTORIA DE USUARIO	ITERACIÓN	ESTADO	PRIORIDAD
HU001	Iniciar sesión y cerrar sesión.	1	Finalizado	Alta
HU002	Gestionar personal.	1	Finalizado	Media

Sprint Backlog

Una vez que se ha desarrollado el *Product Backlog*, se listan cada elemento y se los ordena de acuerdo a las iteraciones establecidas, con el propósito de completar cada objetivo en un tiempo establecido. Por otra parte, al completar una iteración la metodología pide generar un entregable al cliente [28]. En ese sentido, la **TABLA V** se observa el formato que se ha desarrollado con el fin de implementar una lista de 5 *Sprints* que son: Configuración del ambiente de desarrollo, diseño e implementación de la interfaz destinadas al usuario administrador y empleado con sus respectivos módulos, diseño e

implementación de *endpoints* para el consumo desde un sistema *web,* pruebas en el sistema de escritorio y despliegue del sistema a producción.

TABLA V: Formato de *Sprint Backlog*.

ELABORACIÓN DE *SPRINT BACKLOG*						
ID – SB	**NOMBRE**	**MÓDULO**	**ID-HU**	**HISTORIA DE USUARIO**	**TAREA**	**TIEMPO ESTIMADO**
SB000	Configuración del ambiente de desarrollo.	----------	---------	----------	• Recopilación y definición de requerimientos. • Estructura del proyecto.	20H

2.2 Diseño de interfaces

Una vez que se ha establecido los requerimientos, se inicia con la construcción del diseño de las interfaces para el sistema de escritorio.

Herramienta utilizada para el diseño

En el caso de la visualización del sistema de escritorio a través del lenguaje *Java* se ha utilizado *SceneBuilder*, la cual es una herramienta proporcionada para programación visual que da la facilidad de crear interfaces de usuario para este lenguaje de programación sin necesidad de codificar [29].

Para el prototipo del sistema de escritorio se ha realizado prototipos y módulos correspondientes, teniendo así una base para codificar la funcionalidad y lógica del negocio. Por consiguiente, en la **Fig. 1** se muestra un ejemplar de un prototipo que se ha diseñado e implementado, mientras que el listado completo se presenta en el **ANEXO II** del presente documento.

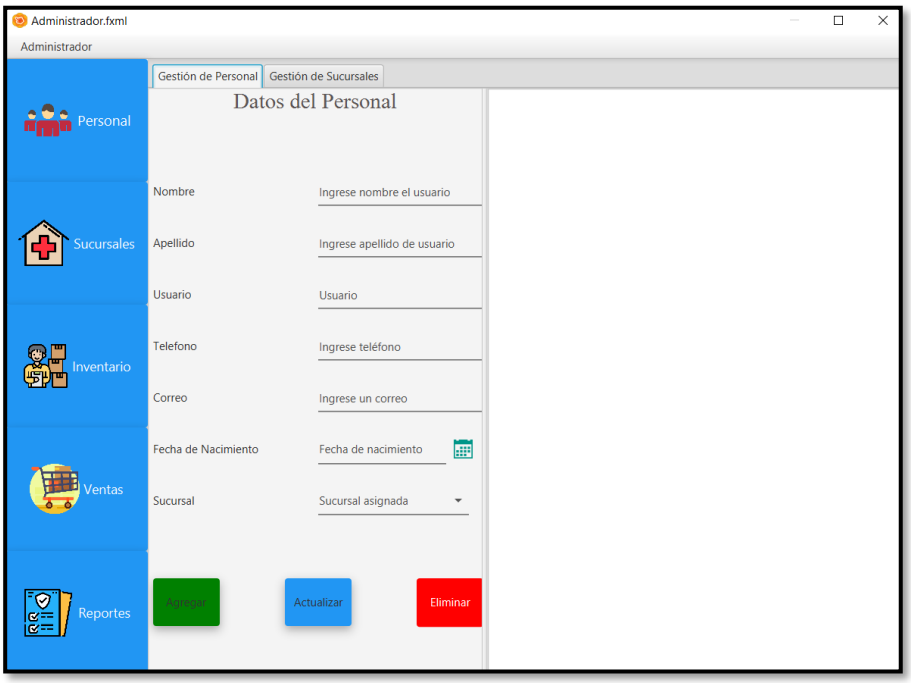

Fig. 1: Prototipo – Módulos perfil administrador.

2.3 Diseño de arquitectura

La selección del patrón arquitectónico sirve para dar una solución óptima al problema planteado. A continuación, se detalla el modelo arquitectónico que se ha seleccionado para el desarrollo del sistema de escritorio.

Patrón arquitectónico Modelo Vista Controlador (MVC)

Es una arquitectura de *software* que se basa en la división de código, formando capas que realizan una tarea en específico, de esta manera se separa la interfaz de usuario, la lógica del proyecto y los datos en tres componentes [22]. Estas capas se describen a continuación:

- **Modelo**: Es la capa donde se encuentra la lógica del proyecto, además se trabaja directamente con los datos para su correcta manipulación.
- **Vista**: Es la interfaz del usuario, la cual interactúa directamente con el usuario; atiende solicitudes y muestra resultados. Se trabaja realizando llamados a los datos sin tener un acceso directo a los mismos.
- **Controlador:** Es la capa que funciona como enlace entre la capa de vista y modelo, la cual implementa mecanismos de respuesta a solicitudes que el usuario tenga dentro de un sistema *software*.

Por consiguiente, la **Fig. 2** ilustra el diseño del patrón de arquitectura que ha sido implementado en conjunto con las herramientas para su desarrollo e implementación a producción.

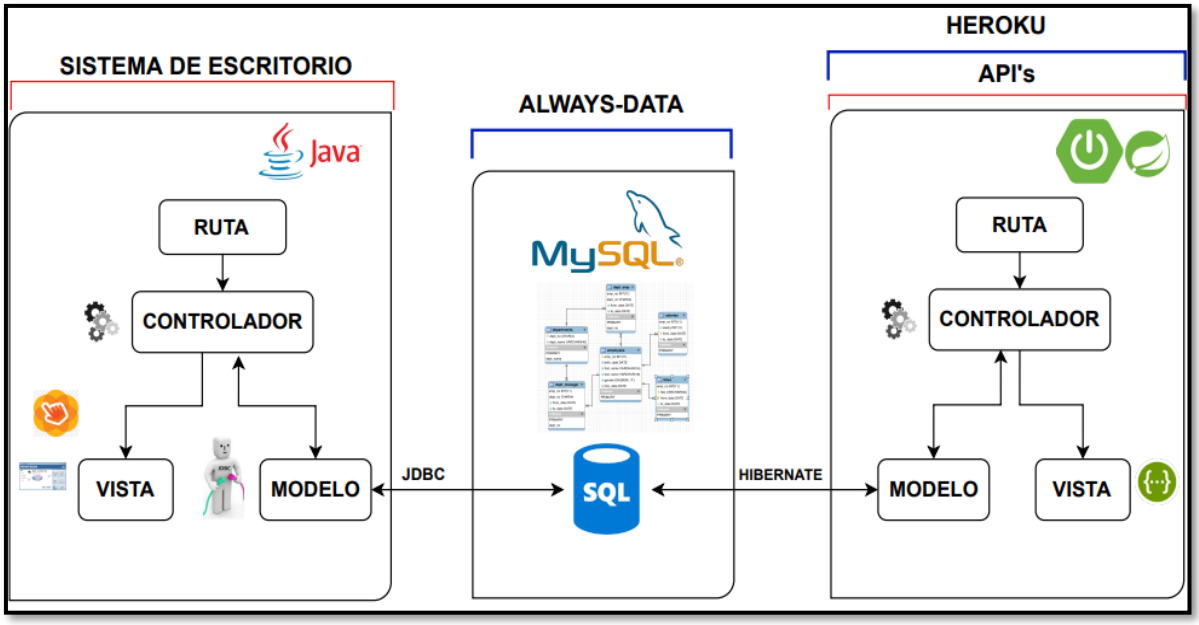

Fig. 2: Patrón Arquitectónico – Sistema de escritorio.

2.4 Herramientas de desarrollo

Las herramientas que se han seleccionado para el desarrollo del sistema de escritorio y creación de los *endpoints* son en base a los requerimientos que se han obtenido y los conocimientos de los integrantes que conforman el equipo de desarrollo, estas herramientas se emplean para documentar el proceso de desarrollo de *software*, por otra parte, optimiza cada uno de los procesos haciendo que el proyecto sea más productivo [30]. Por este motivo en la **TABLA VI** se presentan las herramientas que se han seleccionado para el desarrollo con una breve justificación.

TABLA VI: Herramientas para el desarrollo del sistema de escritorio y *endpoints*.

HERRAMIENTA	JUSTIFICACIÓN
SpringBoot	Proyecto de *Spring* que facilita la configuración y despliegue de aplicaciones basados en servicios web [15].
MySQL	Es un Sistema Gestor de Bases de Datos Relacionales de código abierto [31].

Heroku	Plataforma en la nube que permite desplegar proyectos de distintas tecnologías de manera gratuita o bajo pago [32].
Scene Builder	Emplea programación visual que da la facilidad de crear interfaces de usuario para este lenguaje de programación sin necesidad de codificar [29].

Librerías

La **TABLA VII** muestra a cada una de las librerías que se han empleado para el desarrollo y codificación del sistema de escritorio y generación de los *endpoints*.

TABLA VII: Librerías para el desarrollo del sistema de escritorio y *endpoints*.

LIBRERÍA	DESCRIPCIÓN
Spring-Data-JPA	Maneja la persistencia de los datos con *Java Persistence API* (JPA) junto con *Hibernate* y *Spring Data* [33].
Lambok	Es una librería de Java que mediante una etiqueta permite generar automáticamente *getter*, *setter* y constructores sin escribir código [34].
Spring Security	Marco de trabajo estándar que se enfoca en brindar a aplicaciones *Java* realizadas en *Spring* autenticación y autorización personalizable [35].
Spring Web	Esta dependencia proporciona *Apache Tomcat* como contenedor embebido para generar aplicaciones web y servicios *RESTful* [33].
MySQLDriver	Dependencia que permite la conexión a base de datos SQL, se implementa junto con JDBC [33].
Itext	Librería de código abierto que permite la creación y manipulación de archivos en varios formatos [36].

3 RESULTADOS

A partir de esta sección, se describen todos los resultados que se han conseguido durante la implementación de cada uno de los módulos y componentes visuales del sistema de escritorio, además de los diferentes *endpoints* destinados para su consumo desde un sistema *web*, de la misma manera el resultado de las pruebas y el procedimiento que se ha realizado para su despliegue a producción. No obstante, cada resultado se presenta en forma de *Sprints* los cuales se presentan en el **ANEXO II** de este documento.

3.1 *Sprint* 0. Configuración del ambiente de desarrollo

Para el presente *Sprint* se comprende las siguientes tareas:

- Recopilación y definición de requerimientos.
- Diseño e implementación de la Base de datos Relacional.
- Estructura del proyecto.
- Roles de usuario para el sistema de escritorio y *endpoints*.

Recopilación y definición de requerimientos

Iniciar sesión y cerrar sesión

El sistema de escritorio por medio de la implementación de métodos da al usuario con perfil administrador y empleado la capacidad de poder iniciar sesión y cerrar sesión. Las credenciales de acceso iniciales para el usuario administrador y empleado (nombre de usuario y contraseña) son creadas y designadas por el equipo de desarrollo. En adición, a través de un módulo se generan *endpoints* con las mismas cualidades inicio de sesión y cierre de sesión, las mismas que son consumidas por un sistema *web*.

Gestionar personal

El sistema de escritorio a través de métodos tiene a su disponibilidad un módulo en el cual el usuario con perfil administrador puede gestionar el personal, con lo cual tiene la posibilidad de crear, visualizar, editar, y eliminar información respecto al personal como: nombre, usuario, teléfono, correo electrónico, fecha de nacimiento y la sucursal asignada.

Gestionar sucursales

El sistema de escritorio a través de métodos tiene a su disponibilidad un módulo en el cual el perfil administrador puede gestionar sucursales, con lo cual tiene la posibilidad de crear,

visualizar, editar y eliminar información respecto a las sucursales como: nombre, ciudad, dirección y teléfono. En el caso de pretender dar de baja una de las sucursales, se debe toma en cuenta que a la misma no se encuentre asociada a ningún producto, personal, etc.

Gestionar productos

El sistema de escritorio a través de métodos tiene a su disponibilidad un módulo en el cual el perfil administrador puede gestionar productos, con lo cual tiene la posibilidad de crear, visualizar, editar y eliminar información respecto a los productos como: nombre comercial, nombre genérico, descripción, precio compra, precio venta, cantidad, registro sanitario, fecha de ingreso, fecha de caducidad, ubicación y tipo de producto. En adición, este módulo también se encuentra disponible para el usuario con perfil empleado, con la particularidad de que este usuario no tiene la posibilidad de eliminar productos.

Gestionar ventas

El sistema de escritorio a través de métodos tiene a su disponibilidad un módulo en el cual el perfil administrador puede gestionar ventas, con lo cual tiene la posibilidad de crear, visualizar, editar y eliminar información respecto a las ventas como: nombre, cantidad, precio unitario, precio total y fecha de venta. En adición, este módulo también se encuentra disponible para el usuario con perfil empleado, con la particularidad de que este usuario no tiene la posibilidad de editar y eliminar ventas.

Generar reportes

El sistema de escritorio a través de métodos tiene a su disponibilidad un módulo en el cual el perfil administrador puede generar reportes de ventas realizadas en cada sucursal, reportes de ventas globales y reportes de ventas destinados al Ministerio de Salud Pública del país, estos reportes son generados y tienen la posibilidad de ser descargados. En adición, este módulo también se encuentra disponible para el usuario con perfil empleado, con la particularidad de que este usuario únicamente puede generar el reporte de ventas diario.

Gestionar médicos

El sistema de escritorio a través de métodos tiene a su disponibilidad un módulo en el cual el perfil administrador puede gestionar médicos, con lo cual tiene la posibilidad de crear, visualizar, editar y eliminar información respecto a los médicos como: nombre, apellido, descripción y una imagen.

Gestionar información general

El sistema de escritorio a través de métodos tiene a su disponibilidad un módulo en el cual el perfil administrador puede gestionar la información general, con lo cual tiene la posibilidad de visualizar y editar información respecto a su información general como: misión, visón y nosotros. Finalmente, en la **Fig. 3** se muestra los usuarios y acciones que pueden realizar dentro del sistema de escritorio.

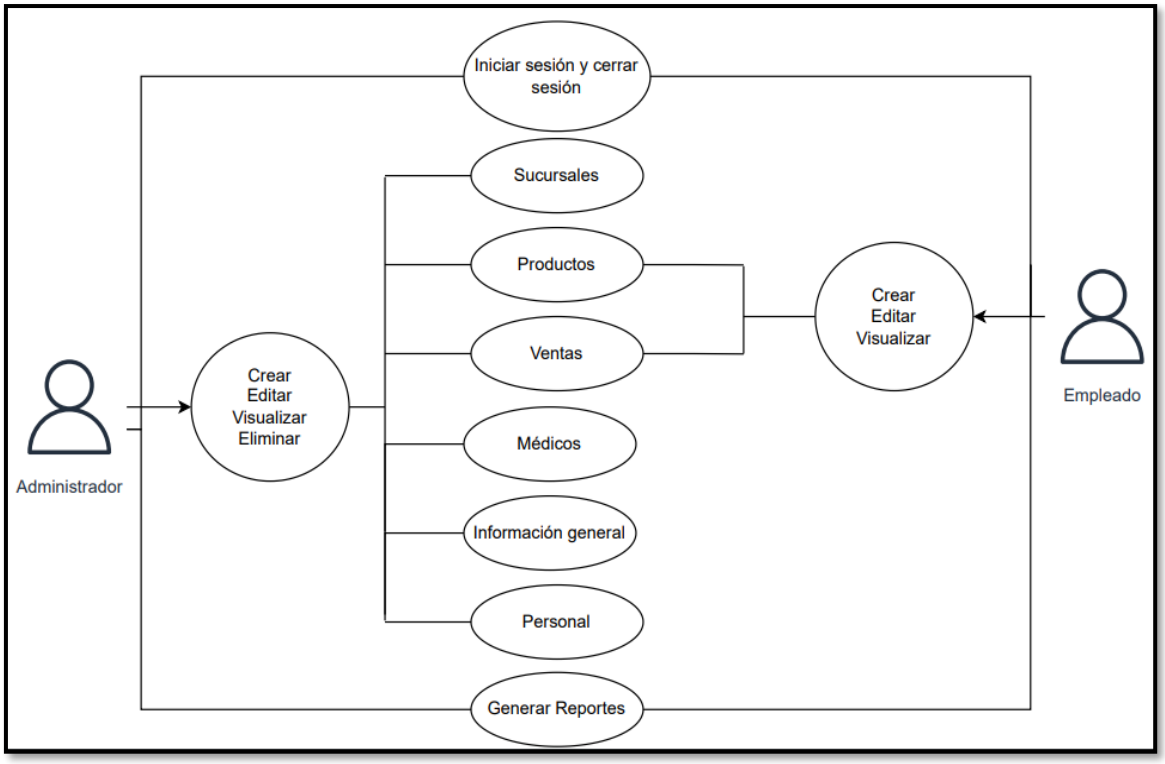

Fig. 3: Usuarios y funcionalidades dentro del sistema de escritorio

Generar *endpoints* para visualización de información general de la farmacia

El sistema crea varios métodos y rutas las cuales permiten obtener toda la información necesaria para visualizar una página informativa como lo es: misión, visión e información general de médicos disponibles. Esto *endpoints* se encuentran disponibles para cualquier usuario.

Generar *endpoints* para gestionar sucursales

El sistema crea varios métodos y rutas privadas las cuales permiten gestionar toda la información de las sucursales, estos *endpoints* generados se encuentran disponibles únicamente para el usuario con perfil administrador. En adición, se ha generado un método

y una ruta pública la cual permite obtener toda la información relacionada a las sucursales disponibles para cualquier usuario.

Generar *endpoint* para la visualización productos

El sistema crea un método y una ruta pública la cual permite obtener toda la información relacionada a los productos disponibles, este *endpoint* se encuentra disponible para usuarios con perfil administrador.

Generar *endpoints* para visualizar reportes de sucursales

El sistema crea varios métodos y rutas privadas las cuales permiten generar reportes de sucursales, es decir, recibe parámetros de fechas con día, mes y año con lo cual genera un archivo en formato .pdf con información de ventas realizadas en la fecha ingresada para ser descargado, estos *endpoints* se encuentran disponibles para usuarios con perfil administrador.

Generar *endpoint* para visualizar disponibilidad de *stock*

El sistema crea un método y una ruta la cual permite obtener información necesaria para visualizar disponibilidad de productos, es decir, la cantidad de existencias disponibles de cada producto, este *endpoint* se encuentra disponible para usuarios con perfil administrador.

Generar *endpoints* para visualizar las ventas de sucursales

El sistema crea varios métodos y varias rutas las cuales permiten obtener información necesaria para visualizar las ventas realizadas de cada sucursal, estos *endpoints* se encuentran disponibles para usuarios con perfil administrador.

Generar *endpoints* para gestionar pedidos con proveedores

El sistema crea varios métodos y varias rutas las cuales permiten gestionar toda la información relacionada a los pedidos con proveedores, estos *endpoints* se encuentran disponibles para usuarios con perfil administrador.

Generar *endpoint* para visualizar el personal

El sistema crea un método y una ruta la cual permite visualizar toda la información relacionada al personal, este *endpoint* se encuentra disponible para usuarios con perfil administrador.

Generar *endpoint* para realizar cotizaciones de productos

El sistema crea un método y una ruta la cual permite visualizar información necesaria de productos, mediante esta información se puede realizar cotizaciones de cada producto disponible, este *endpoint* se encuentra disponible para todos los usuarios. Finalmente, en la **Fig. 4** se muestra los usuarios y los métodos que pueden realizar en los *endpoints*.

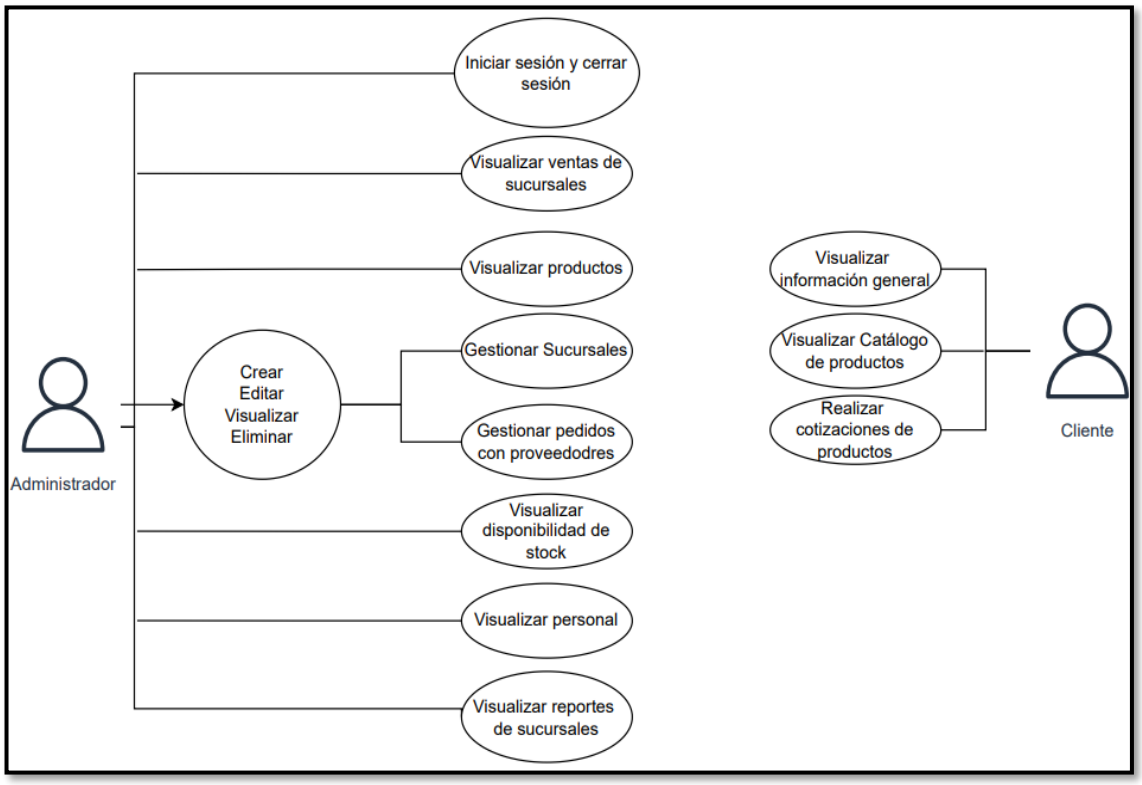

Fig. 4: Usuarios y funcionalidades para los *endpoints.*

Diseño e implementación de la Base de datos Relacional

Para el almacenamiento de datos para este proyecto se ha optado por la utilización de un sistema gestor de bases de datos relacionales SQL llamado MySQL a través de *phpMyAdmin*, el cual permite almacenar datos en forma de tablas y relacionarlas entre sí, además de su sincronización. En la **Fig. 5** se muestra el listado de las tablas relacionadas que se ha implementado para la comunicación y transmisión de datos.

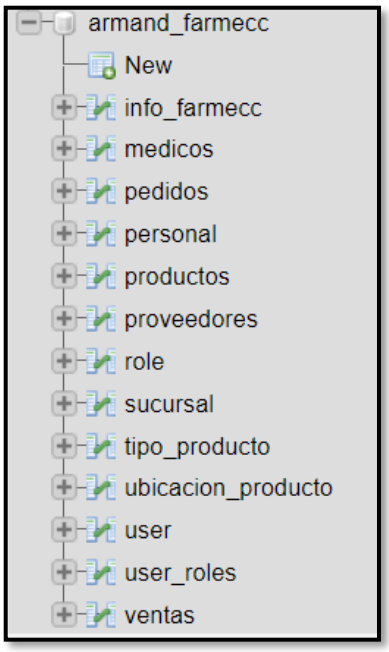

Fig. 5: Base de datos relacional

Estructura del proyecto

Para el desarrollo del sistema de escritorio y generación de *endpoints* se ha empleado *IntelliJ IDEA* como entorno de desarrollo, mediante el panel de inicio se ha establecido la estructura de cada directorio basándose en el patrón arquitectónico anteriormente seleccionado, es así que la **Fig. 6** y **Fig. 7** muestran la estructura actual del proyecto.

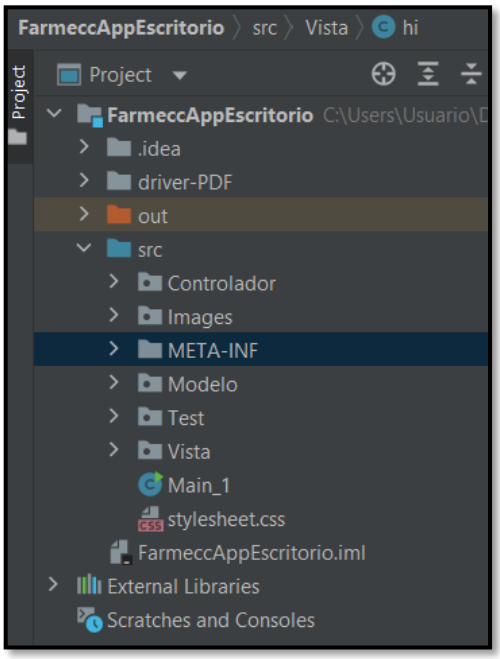

Fig. 6: Estructura del proyecto – sistema de escritorio.

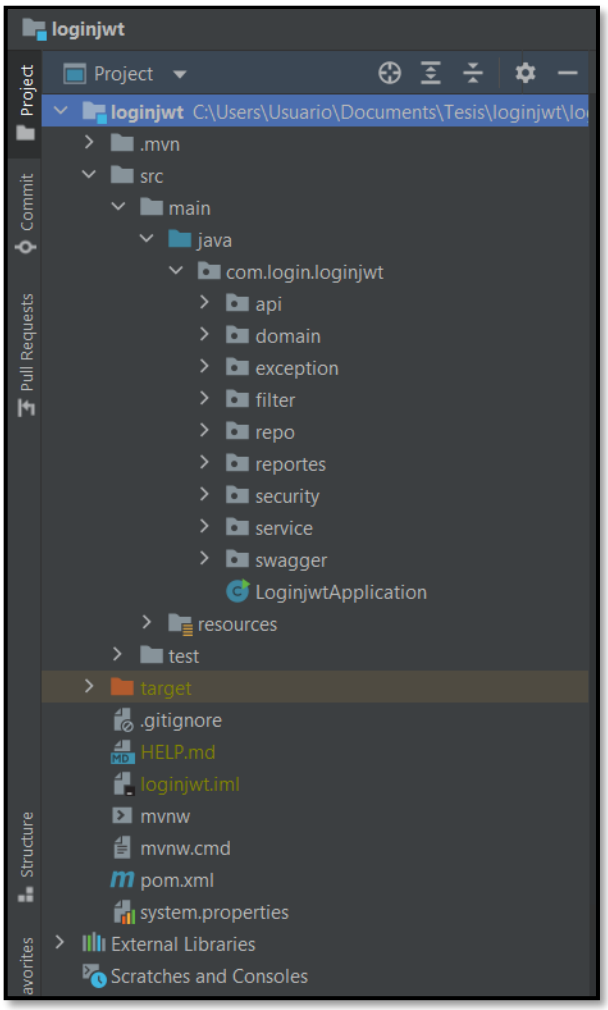

Fig. 7: Estructura del proyecto – *endpoints*.

Roles de usuario para el sistema de escritorio y *endpoints*

A continuación, en la **Fig. 8** se presenta todos los módulos a los cuales pueden acceder los usuarios: administrador y empleado, iniciando sesión de manera obligatoria. Por otra parte, para los *endpoints* en la **Fig. 9** se presenta todos los módulos a los cuales tienen acceso cada uno de los usuarios: administrador y cliente, iniciando sesión en caso de ser necesario.

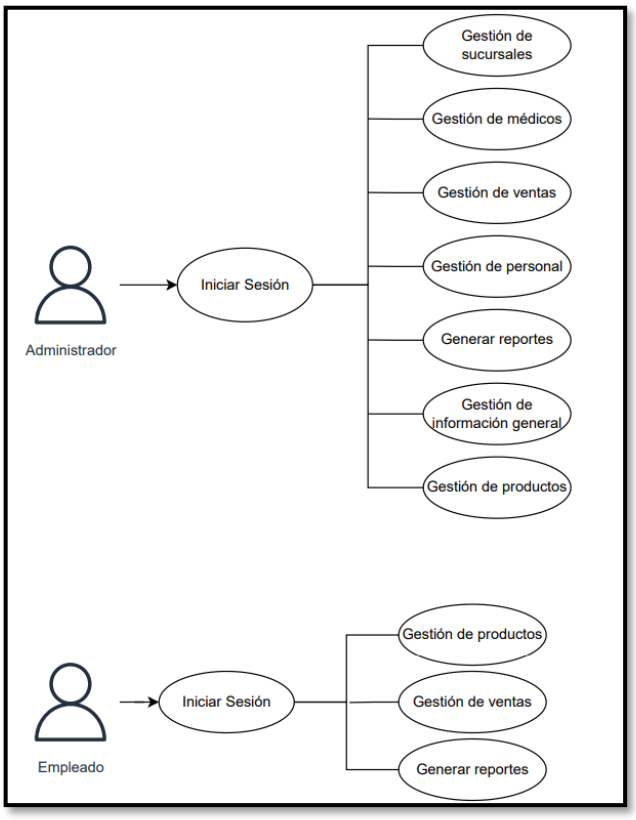

Fig. 8: Usuarios y roles para el sistema de escritorio.

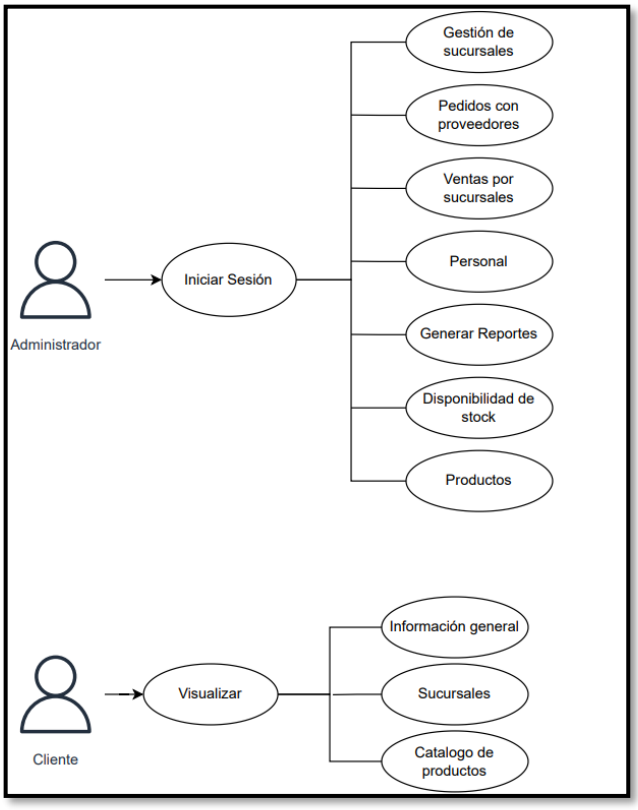

Fig. 9: Usuarios y roles para uso de *endpoints.*

3.2 *Sprint* 1. Resultado de la implementación de la interfaz destinada al usuario administrador y empleado.

Este *Sprint* comprende las siguientes tareas:

- Iniciar sesión y cerrar sesión.
- Gestionar personal.
- Gestionar sucursales.
- Gestionar productos.
- Gestionar ventas.
- Generar reportes.
- Gestionar médicos.
- Gestionar información general.

Iniciar sesión y cerrar sesión

El sistema de escritorio implementa un módulo el cual es destinado para inicio de sesión por parte del usuario administrador y usuario empleado. Se debe colocar las credenciales necesarias (nombre de usuario y contraseña) de cada rol proporcionadas por el equipo de desarrollo. Además, el usuario tiene posibilidad de cerrar sesión como se presenta en la **Fig. 10**. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

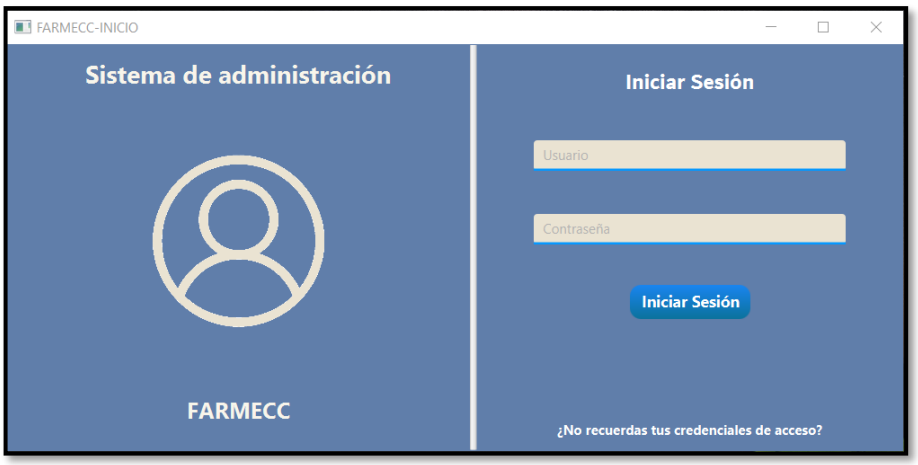

Fig. 10: Módulo inicio de sesión.

Gestionar personal

El sistema de escritorio implementa un módulo el cual es destinado a la gestión de personal, es decir, el usuario con perfil administrador puede visualizar, crear, editar y eliminar información relacionada al personal. Además, se ha implementado una tabla para poder visualizar la información relacionada a este módulo, mientras que, para agregar y editar se lo realiza a través de un formulario con diferentes campos tales como: nombre, usuario, teléfono, correo electrónico, fecha de nacimiento y un campo de selección en el cual se puede asignar a este usuario a una sucursal, por otra parte, para el caso de eliminar el personal se lo realiza mediante la selección del mismo como se presenta en la **Fig. 11**. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

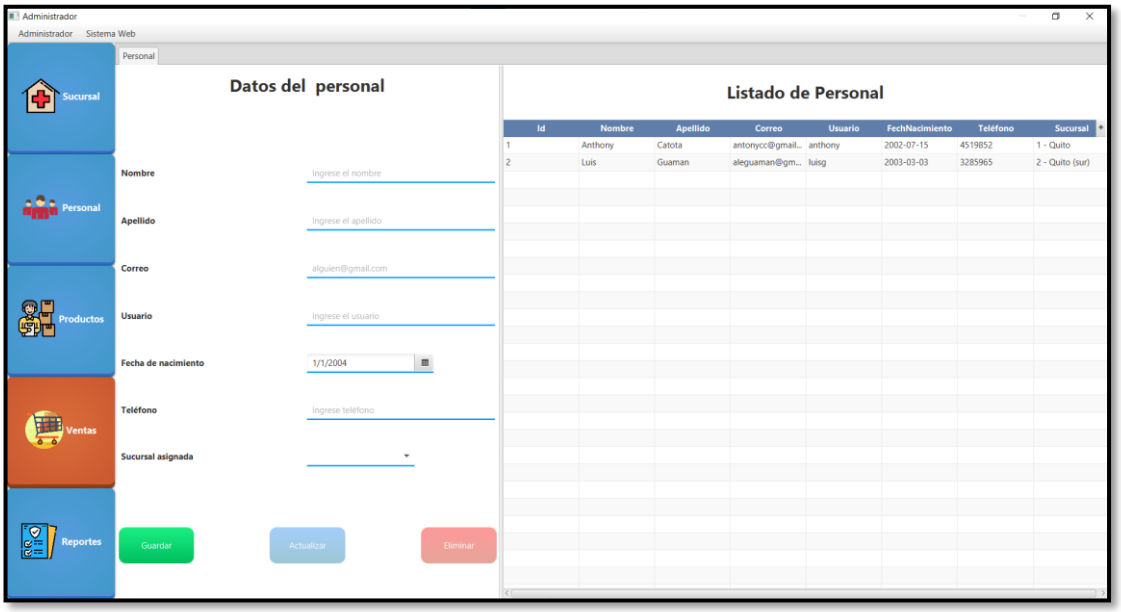

Fig. 11: Módulo de gestión de personal del sistema de escritorio.

Gestionar sucursales

El sistema de escritorio implementa un módulo el cual es destinado a la gestión de sucursales, es decir, el usuario con perfil administrador puede visualizar, crear, editar y eliminar información relacionada a sucursales. Además, se ha implementado una tabla para poder visualizar la información de cada sucursal, mientras que, para agregar y editar información se lo realiza a través de un formulario con diferentes campos tales como: nombre, ciudad, dirección y teléfono. Para el caso de eliminar una sucursal únicamente el usuario administrador tiene esta posibilidad cuando la sucursal no se encuentre ligada con un empleado o productos como se presenta en la **Fig. 12**. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

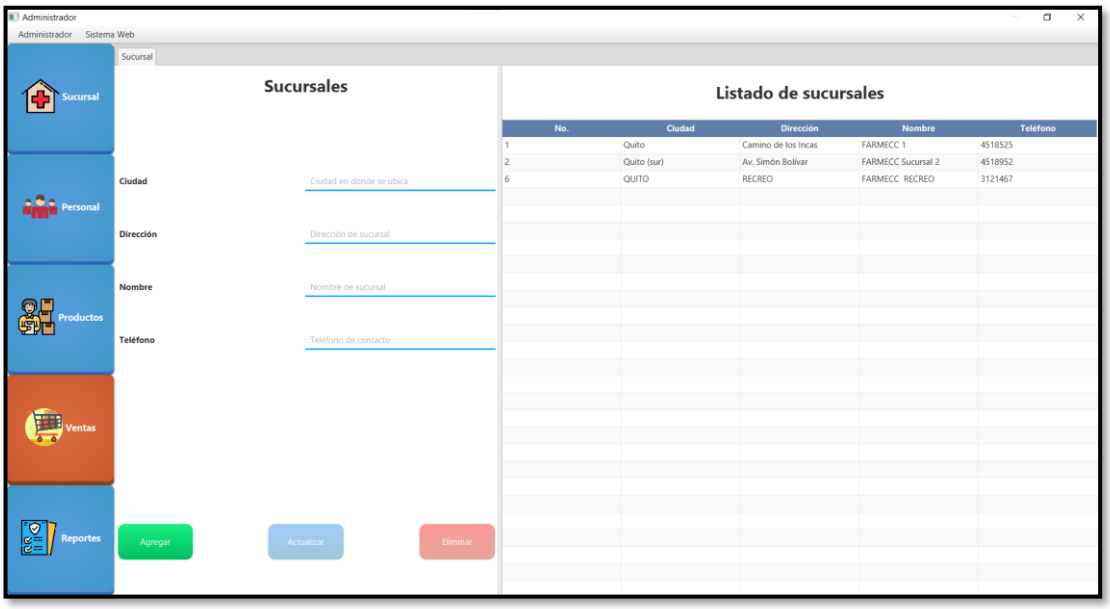

Fig. 12: Módulo de gestión de sucursales del sistema de escritorio.

Gestionar productos

El sistema de escritorio implementa un módulo el cual es destinado a la gestión de productos, es decir, el usuario con perfil administrador puede visualizar, crear, editar y eliminar información relacionada a productos. Además, se ha implementado una tabla para poder visualizar la información de los productos de todas las sucursales, mientras que, para agregar y editar información se lo realiza a través de un formulario con diferentes campos tales como: nombre comercial, nombre genérico, descripción, precio compra, precio venta, cantidad, registro sanitario, fecha de ingreso, fecha de caducidad, ubicación, tipo de producto e imagen del producto. Para el caso de eliminar el producto se lo realiza mediante la selección del mismo como se presenta en la **Fig. 13**. En adición, este módulo también se encuentra disponible para el usuario con perfil empleado, con la particularidad de que este usuario no tiene la posibilidad de eliminar productos. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

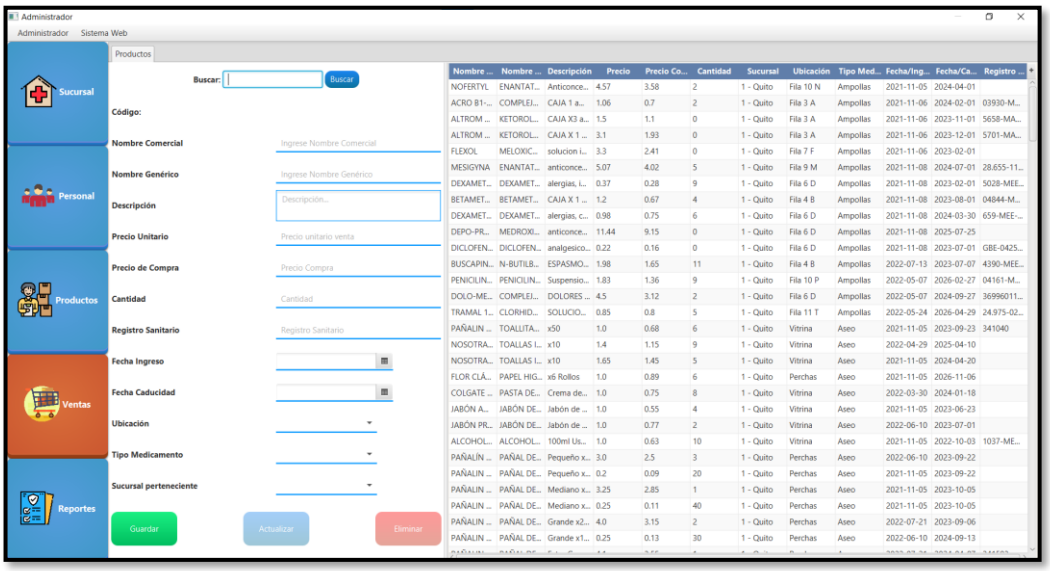

Fig. 13: Módulo de gestión de productos del sistema de escritorio.

Gestionar ventas

El sistema de escritorio implementa un módulo el cual es destinado a la gestión de ventas, es decir, el usuario con perfil administrador puede, visualiza, crear y eliminar información relacionada a las ventas, esto se ha realizado a través de un formulario con diferentes campos tales como: cantidad y descuento, además de la selección del producto del cual se realiza la venta como se presenta en la **Fig. 14**. En adición, este módulo también se encuentra disponible para el usuario con perfil empleado, con la particularidad de que este usuario no tiene la posibilidad de eliminar ventas. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

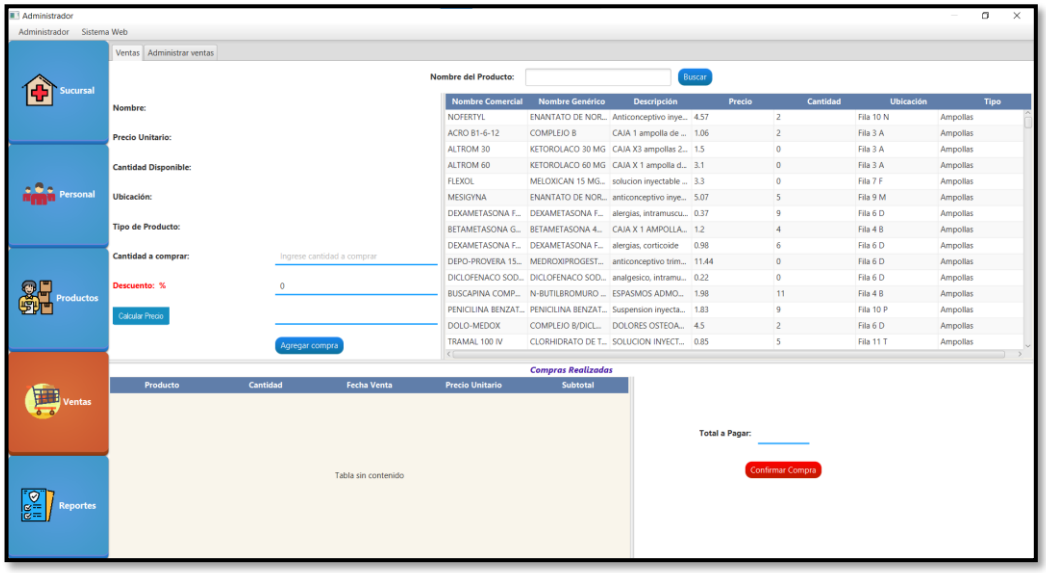

Fig. 14: Módulo de gestión de ventas del sistema de escritorio.

Generar reportes

El sistema de escritorio implementa un módulo el cual es destinado a generar reportes, es decir, el perfil administrador puede generar reportes de ventas realizadas en cada sucursal, reportes de ventas globales y reportes de ventas destinadas al Ministerio de Salud Pública del país, los reportes se presentan en una tabla y tienen la posibilidad de ser descargados en archivos de formato *PDF* con una estructura preestablecida como se presenta en la **Fig. 15**. Por otra parte, el perfil empleado únicamente puede generar un reporte de venta diario de cierre de caja. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

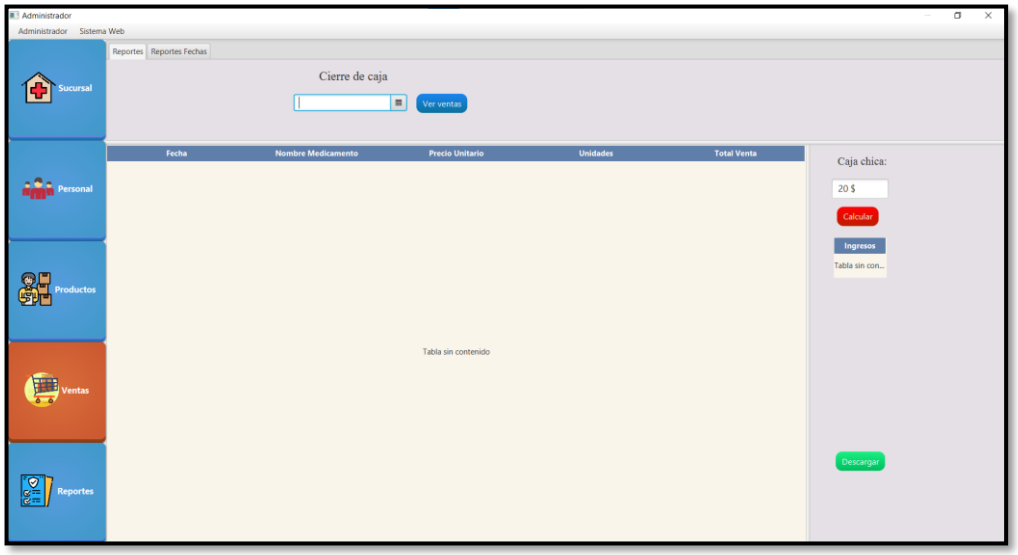

Fig. 15: Módulo de gestión de reportes del sistema de escritorio.

Gestionar médicos

El sistema de escritorio implementa un módulo el cual es destinado a la gestión de médicos, es decir, el usuario con perfil administrador puede visualizar, crear, editar y eliminar información relacionada a los médicos. Además, se ha implementado una tabla para poder visualizar la información relacionada a los médicos, mientras que, para agregar y editar se lo realiza a través de un formulario con diferentes campos tales como: nombre, apellido, descripción y una imagen ya sea cargada o simplemente colocar el enlace dirigido a la imagen, por otra parte, para el caso de eliminar el médico se lo realiza mediante la selección del mismo como se presenta en la **Fig. 16**. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

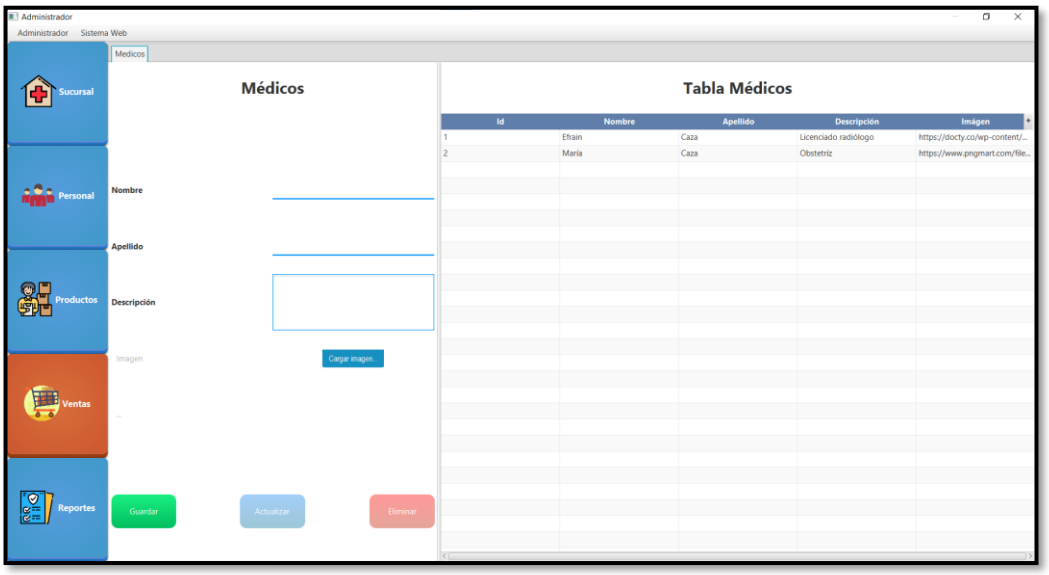

Fig. 16: Módulo de gestión de médicos del sistema de escritorio.

Gestionar información general

El sistema de escritorio implementa un módulo el cual es destinado a la gestión de información general, es decir, el usuario con perfil administrador puede visualizar y editar la información general de la cadena de farmacias. Además, se ha implementado recuadros para poder visualizar la información general, mientras que, para agregar y editar se lo realiza a través de un formulario con diferentes campos tales como: misión, visión y nosotros como se presenta en la **Fig. 17**. En este sentido, el proceso que detalla el uso de este módulo se encuentra en el **ANEXO III** del presente documento.

Fig. 17: Módulo información general del sistema de escritorio.

3.3 *Sprint* 2. Resultado de la implementación de los diferentes *endpoints*.

Este *Sprint* comprende las siguientes tareas:

- Generar *endpoint* para visualizar información general de la farmacia.
- Generar *endpoints* para el inicio y cierre de sesión.
- Generar *endpoints* para gestionar sucursales.
- Generar *endpoints* para visualizar productos.
- Generar *endpoints* para visualizar reportes.
- Generar *endpoints* para visualizar disponibilidad de *stock*.
- Generar *endpoint* para realizar cotizaciones.
- Generar *endpoints* para visualizar ventas.
- Generar *endpoint* para visualizar personal.
- Generar *endpoints* para gestionar pedidos con proveedores.

Generar *endpoint* para visualizar información general de la farmacia

Toda la información relacionada con la cadena de farmacias “FARMECC” se encuentra almacenada en varias tablas de una base de datos SQL, las mismas contienen información como: médicos, sucursales, misión, visión, entre otros. Además, se ha implementado un método tipo *GET* con ruta pública la misma que a través de su consumo mediante un sistema *web* permite obtener toda la información previamente mencionada como se presenta en la **Fig. 18**. En este sentido, el proceso que detalla el consumo de la información se encuentra en el **ANEXO III** del presente documento.

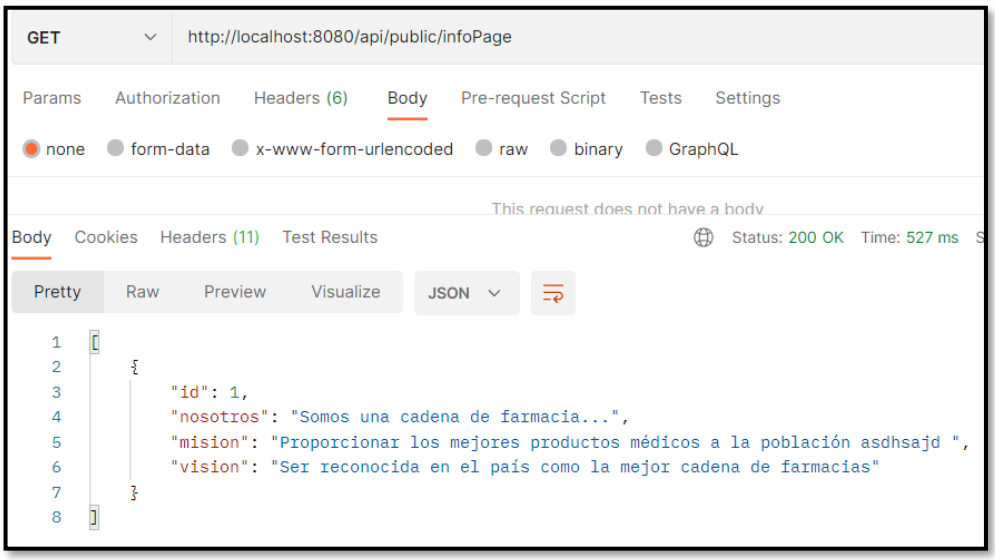

Fig. 18: Método *GET* para la visualización de información general de la farmacia.

Generar *endpoints* para inicio y cierre de sesión

Para el inicio y cierre de sesión se han implementado métodos y rutas públicas las mismas que permiten al usuario consumir estos *endpoints* sin restricciones. Además, se ha implementado un método de tipo *POST* con ruta pública para el envío de credenciales para su inicio de sesión la misma que a través de su consumo mediante un sistema *web* permite obtener un *token* de acceso para los *endpoints* privados como se presenta en la **Fig. 19**. En este sentido, el proceso que detalla el consumo de estos *endpoints* se encuentra en el **ANEXO III** presente documento.

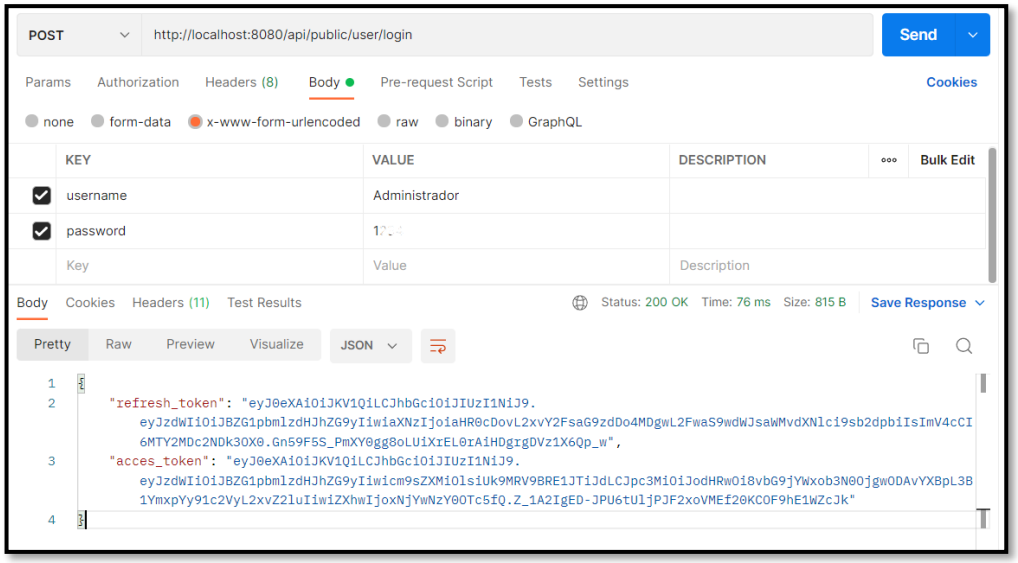

Fig. 19: Método *POST* para inicio se sesión.

Generar *endpoints* para gestionar sucursales

Para la gestión de la información sobre las sucursales se han creado varios métodos, rutas públicas y privadas las cuales permiten al administrador gestionar toda la información relacionada con las sucursales. Además, se ha implementado un método de tipo *GET* con una ruta pública para la obtención de la información de las sucursales, métodos de tipo *POST* y *PUT* con rutas privadas para el ingreso y actualización de la información respectivamente; esto se lo realiza a través de un formulario, por último, un método *DELETE* con una ruta privada para la eliminación de una sucursal en caso de ser necesario como se presenta en la **Fig. 20**. En este sentido, el proceso que detalla tanto el consumo de la información y las validaciones correspondientes se encuentran en el **ANEXO III** del presente documento.

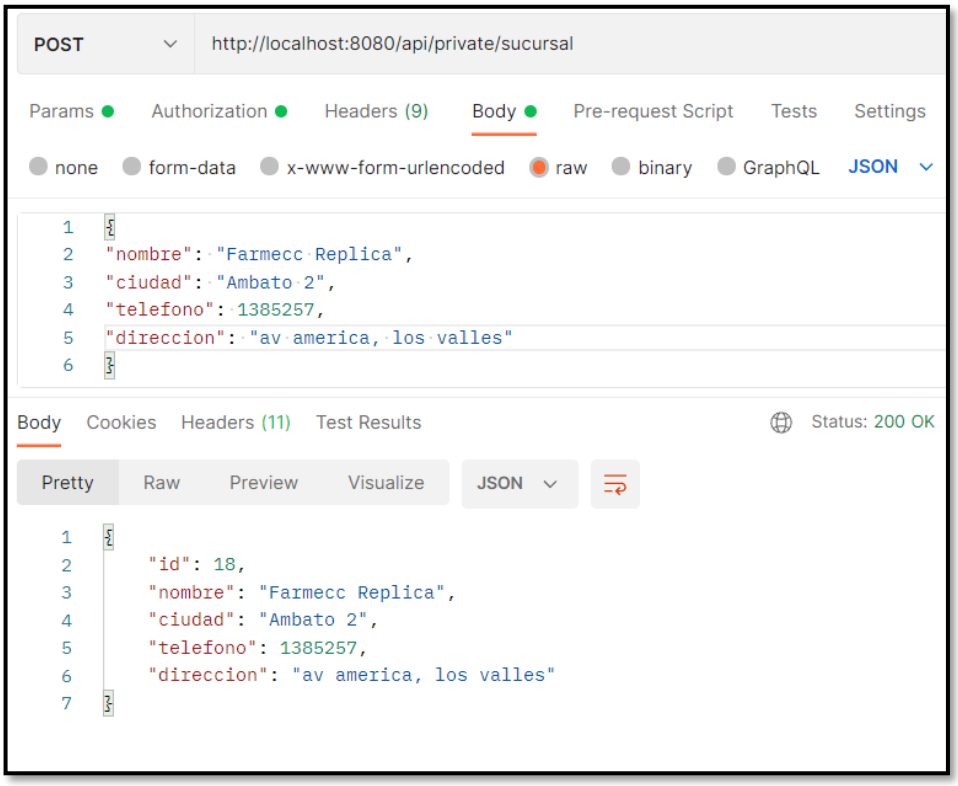

Fig. 20: Método *POST* para añadir una sucursal.

Generar *endpoints* para visualizar productos

Toda la información correspondiente a los productos que se han registrado se encuentra almacenado en una tabla de una base de datos SQL. Además, se han creado métodos *GET* con rutas públicas las misma que a través de su consumo mediante un sistema *web* permiten obtener toda la información de los productos registrados en su totalidad o por

sucursales como se presenta en la **Fig. 21**. En este sentido, el proceso que detalla el consumo de la información se encuentra en el **ANEXO III** del presente documento.

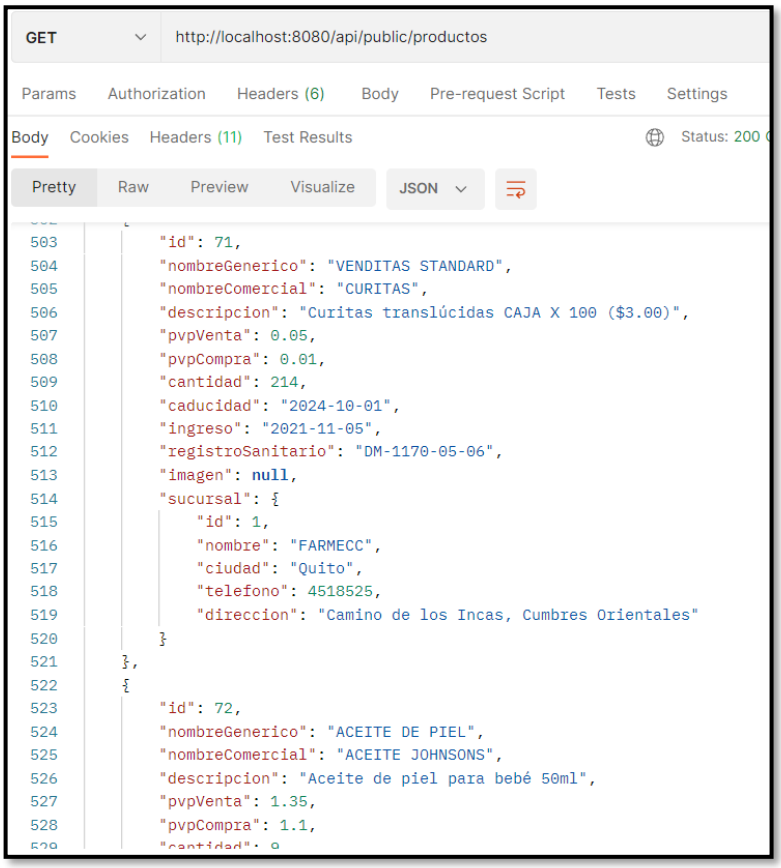

Fig. 21: Método *GET* para la visualización de productos.

Generar *endpoints* para visualizar reportes

Para la obtención de reportes se han creado varios métodos de tipo *GET* con rutas privadas las cuales permiten que el administrador pueda obtener los reportes que son generados mediante consultas, los reportes se envían al sistema *web* en donde estos pueden ser descargados como se presenta en la **Fig. 22**. En este sentido, el proceso que detalla tanto el consumo de la información y las validaciones correspondientes se encuentran en el **ANEXO III** del presente documento.

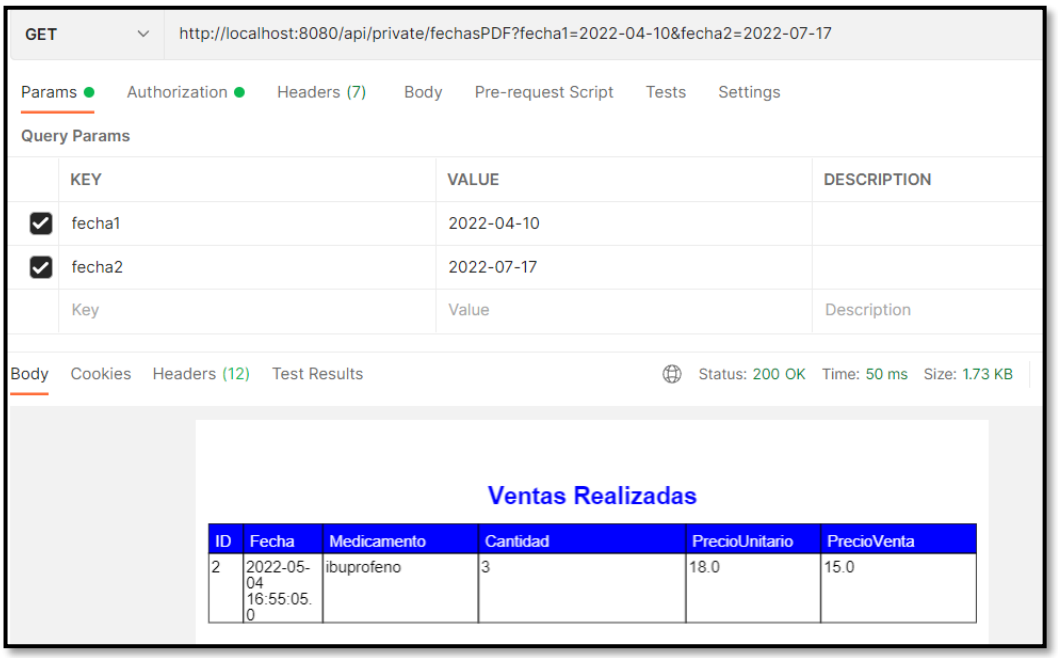

Fig. 22: Método *GET* para generar reportes.

Generar *endpoint* para visualizar disponibilidad de *stock*

Toda la información correspondiente a los productos que se han registrado se encuentra almacenado en una tabla de una base de datos SQL. Además, se ha creado un método *GET* con ruta pública la misma que a través de su consumo mediante un sistema *web* permiten obtener la información sobre la disponibilidad de cada producto como se presenta en la **Fig. 23**. En este sentido, el proceso que detalla el consumo de la información se encuentra en el **ANEXO III** del presente documento.

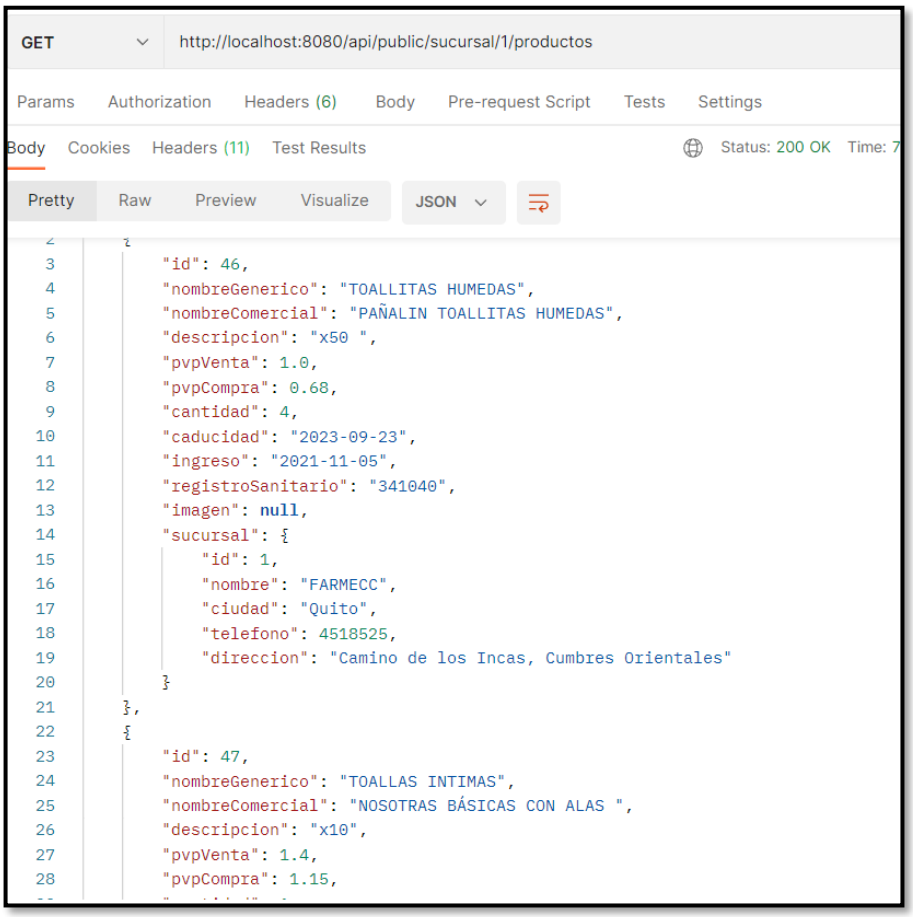

Fig. 23: Método *GET* para visualizar disponibilidad de *stock*.

Generar *endpoints* para realizar cotizaciones

Toda la información correspondiente a los productos que se han registrado se encuentra almacenados en una tabla de una base de datos SQL. Además, se han creado métodos *GET* con rutas públicas las mismas que a través de su consumo mediante un sistema *web* permiten obtener la información necesaria para que cualquier usuario del sistema *web* pueda realizar cotizaciones de los productos disponibles y de los que ellos requieran como se presenta en la **Fig. 24**. En este sentido, el proceso que detalla el consumo de la información se encuentra en el **ANEXO III** del presente documento.

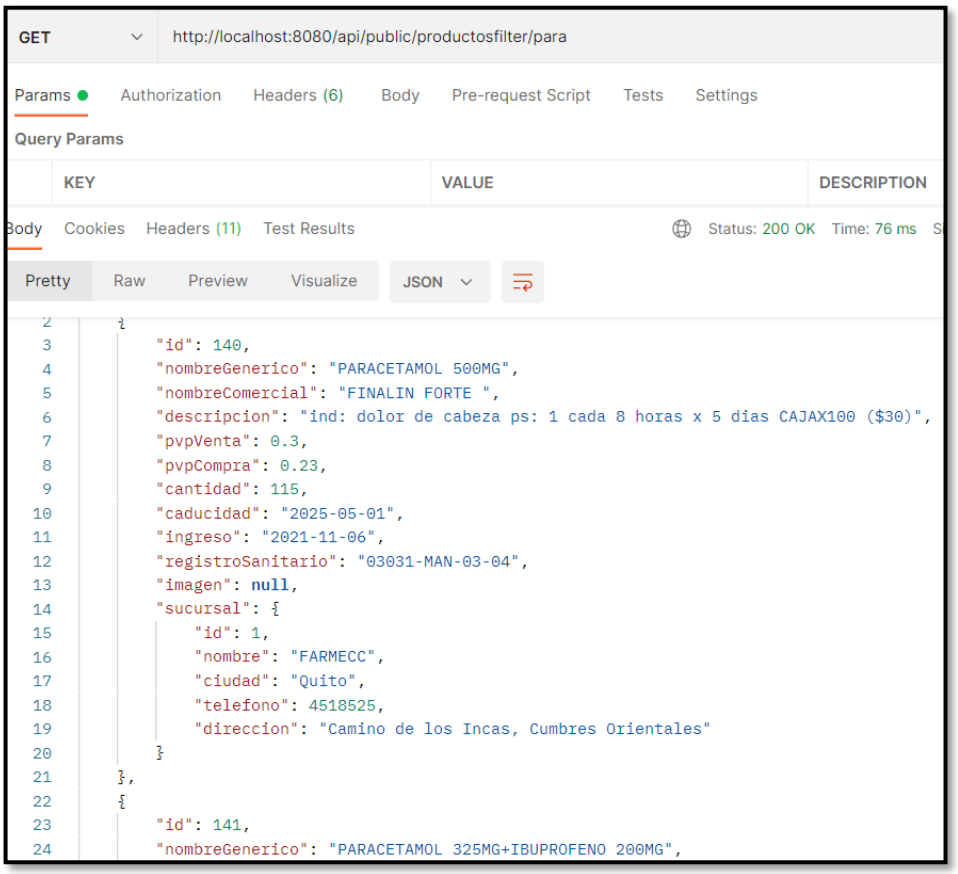

Fig. 24: Método *GET* para realizar cotizaciones.

Generar *endpoints* para visualizar ventas

Toda la información correspondiente a las ventas realizadas se encuentra almacenada en una tabla de una base de datos SQL. Además, se han creado métodos *GET* con rutas privadas las cuales permiten al administrador visualizar las ventas que se han realizado en su totalidad o por sucursales como se presenta en la **Fig. 25**. En este sentido, el proceso que detalla tanto el consumo de la información y las validaciones correspondientes se encuentran en el **ANEXO III** del presente documento.

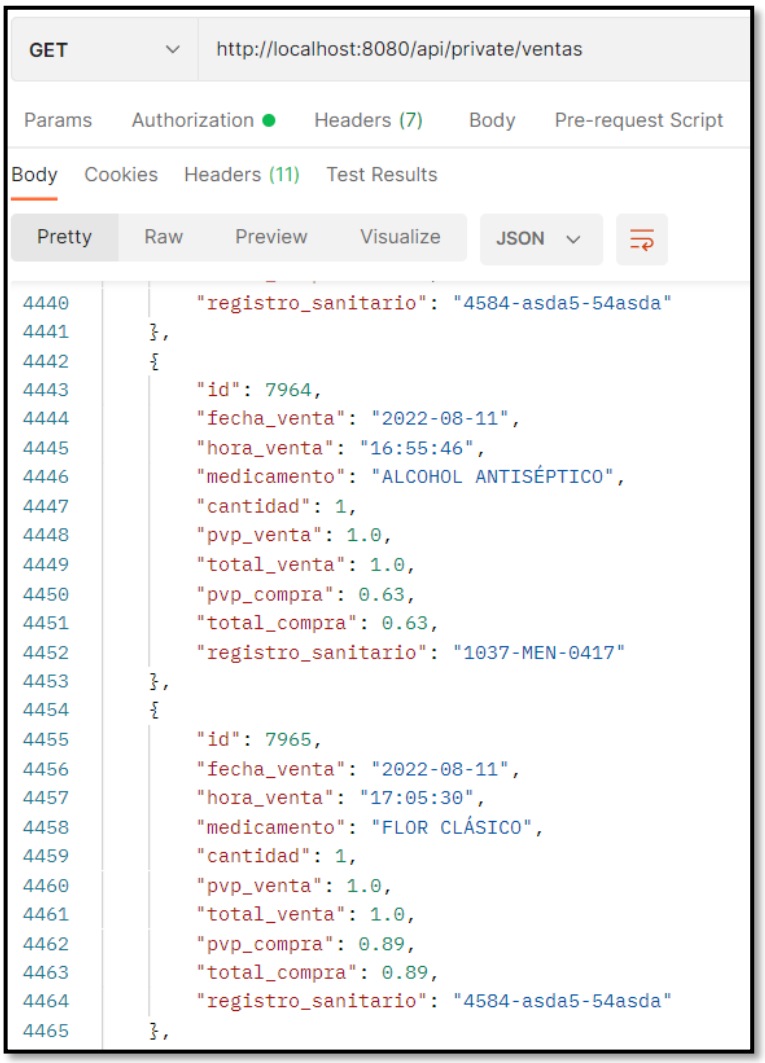

Fig. 25: Método *GET* para visualizar ventas.

Generar *endpoint* para visualizar personal

Toda la información correspondiente al personal registrado se encuentra almacenado en una tabla de una base de datos SQL. Además, se ha creado un método tipo *GET* con ruta privada la cual permite al administrador visualizar el personal disponible y la asignación a cada sucursal como se presenta en la **Fig. 26**. En este sentido, el proceso que detalla tanto el consumo de la información y las validaciones correspondientes se encuentran en el **ANEXO III** del presente documento.

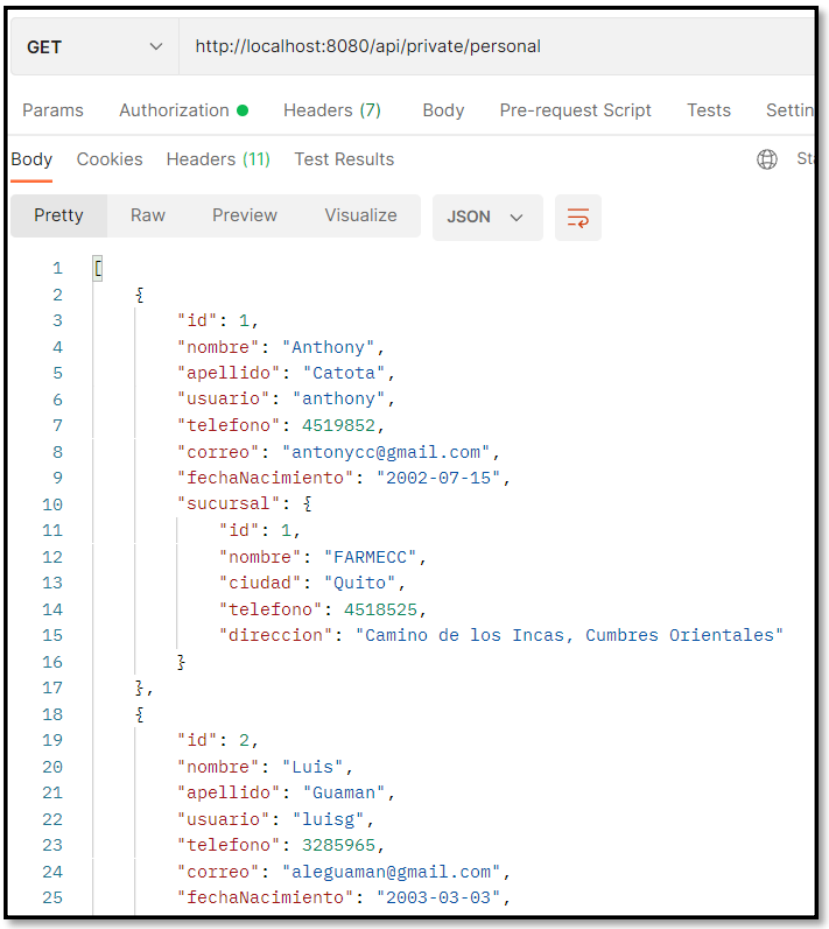

Fig. 26: Método *GET* para visualizar personal.

Generar *endpoints* para gestionar pedidos con proveedores

Para la gestión de la información sobre los proveedores se han creado varios métodos con rutas públicas y privadas las cuales permiten al administrador gestionar toda la información acerca de proveedores. En ese sentido, se han implementado métodos de tipo *GET* con rutas privadas para la obtención de la información de los proveedores y productos con *stock* reducido, métodos de tipo *POST* y *PUT* con rutas privadas para el ingreso y actualización de información respectivamente; esto se lo realiza a través de un formulario, por último, un método *DELETE* con una ruta privada para la eliminación algún proveedor en caso de ser necesario como se presenta en la **Fig. 27**. En este sentido, el proceso que detalla tanto el consumo de la información y las validaciones correspondientes se encuentran en el **ANEXO III** del presente documento.

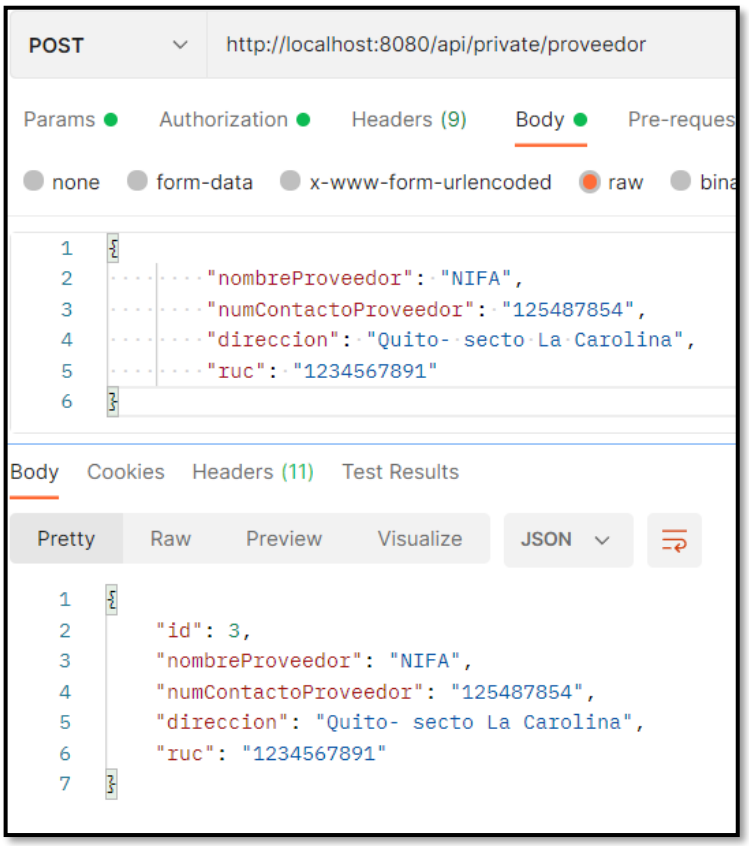

Fig. 27: Método *POST* para agregar proveedores.

3.4 *Sprint* 3. Pruebas en el sistema de escritorio.

Finalizada la fase de codificación de módulos del sistema de escritorio y *endpoints* en base a lo planificado en el *Sprint Backlog* el presente *Sprint* comprende las siguientes tareas:

- Resultados de la ejecución de pruebas unitarias.
- Resultados de la ejecución de pruebas de compatibilidad.
- Resultad de la ejecución de pruebas de aceptación.

Resultados de la ejecución de pruebas unitarias

Las pruebas unitarias o *unit testing* son pruebas de *software*, estas se consideran parte del control de calidad en el desarrollo de *software* siendo uno de los procesos más importantes, en este sentido las pruebas unitarias comprueban el correcto funcionamiento de los componentes individuales de una clase, incluido el funcionamiento de un método especifico [37]. En base a esto, *Java* permite la realización de pruebas unitarias mediante librerías externas como: *Junit* y *Mockito* las cuales permiten realizar pruebas simulando llamados para no alterar el funcionamiento del *software*.

La **Fig. 28** presenta una parte del código empleado para agregar productos el cual se ha implementado para una de las pruebas del sistema de escritorio, en adición, en la **Fig. 29** se puede identificar el resultado posterior a la ejecución de la prueba previamente mencionada.

```
    @Test
    public void guardarProducto(){
        conexion.establecerConexion();
        System.out.println("Guardar");
        int resutado=producto.guardarRegistro(conexion.getConnection());
        conexion.cerrarConexion();
        if (resutado==0){
            System.out.println("Prueba exitosa");
        }else System.out.println("Prueba Fallida");
    }
}
```

Fig. 28: Fragmento de código para agregar productos.

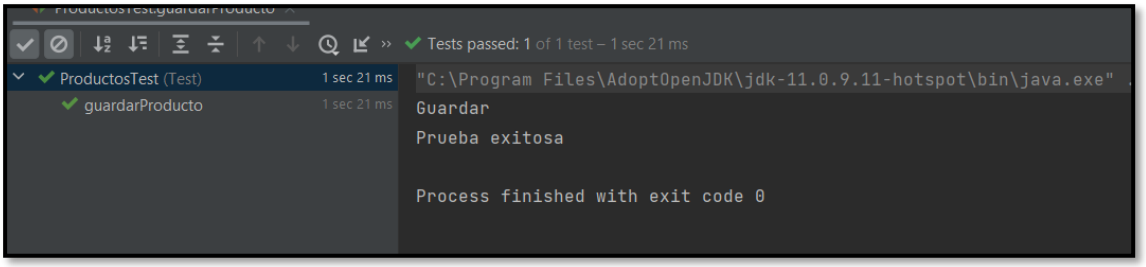

Fig. 29: Resultado de la prueba.

Por otra parte, en la **Fig. 30** se presenta una parte del código empleado para visualizar los médicos el cual se ha implementado para realizar una de las pruebas de los *endpoints*, en adición, en la **Fig. 31** se puede identificar el resultado posterior a la ejecución de la prueba previamente mencionada. La especificación completa de estas ejecuciones, así como los resultados del resto de las pruebas realizadas de encuentra en el **ANEXO II** del presente documento.

```java
@SpringBootTest
public class MedicosRepoTest {
    @Mock
    private MedicosRepo testMedicos;
    @InjectMocks
    private MedicosController medicosController;
    private Medicos medicos;
    @BeforeEach
    void setUp(){
        MockitoAnnotations.openMocks( testClass: this);
        medicos = new Medicos();
        medicos.setId(1);
        medicos.setNombre("Efrain");
        medicos.setApellido("Caza");
        medicos.setDescripcion("Medico");
        medicos.setUrlImagen("algo.jpg");
    }
    @Test
    void verMedicos(){
        when(testMedicos.findAll()).thenReturn(Arrays.asList(medicos));
        assertNotNull(testMedicos.findAll());
    }
```

Fig. 30: Fragmento de código para visualizar médicos.

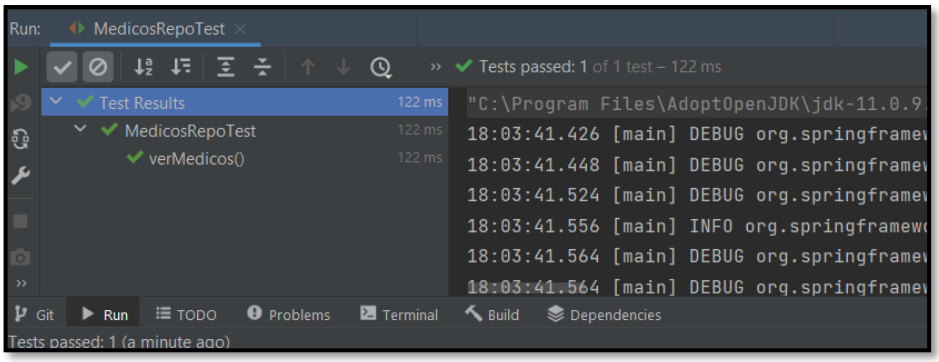

Fig. 31: Resultado de la prueba.

Culminada la etapa de pruebas unitarias y en base a los resultados que se han obtenido, se puede establecer que los módulos del sistema de escritorio y los métodos de los *endpoints* funcionan correctamente, sin problemas a nivel de código y validaciones respectivamente.

Resultados de la ejecución de pruebas de compatibilidad

Las pruebas de compatibilidad, permite verificar si un sistema tiene la capacidad de ejecutarse correctamente en diferentes entornos, ya sea de hardware, sistema operativo o redes. Por lo cual, en la **TABLA VIII** se presenta las versiones del sistema operativo que se han empleado para comprobar si el sistema de escritorio es ejecutable en estos entornos. Por otra parte, la **TABLA IX** presenta los clientes HTTP en donde se han realizado estas pruebas de compatibilidad. La descripción tanto de las ejecuciones como

los resultados del resto de pruebas de compatibilidad se encuentran en el **ANEXO II** del presente documento.

TABLA VIII: Sistemas operativos utilizados para pruebas de compatibilidad.

NOMBRE	VERSIÓN	OBSERVACIÓN
Windows 10 *Home*	10.0.19044	Completamente funcional
Windows 11 Pro	10.0.22000	Completamente funcional

TABLA IX: Clientes HTTP empleados para pruebas de compatibilidad.

NOMBRE	VERSIÓN
Postman	10.0.19044
Talend API Tester	25.10.0

Culminada la etapa de pruebas de compatibilidad y en base a los resultados obtenidos en las ejecuciones de las respectivas pruebas en los sistemas operativos, se presencia que el sistema de escritorio es compatible con cada uno de estos, tanto en funcionamiento como en visibilidad del contenido sin presentar algún tipo de error. Mientras que los *endpoints* al pasar por los clientes HTTP no presenta ningún fallo en su presentación de datos o en tiempo de ejecución, aceptando así todas las solicitudes enviadas.

Resultados de la ejecución de pruebas de aceptación

Las pruebas de aceptación en el desarrollo de un *software* hecho a la medida del cliente buscan satisfacer las necesidades de este, permitiendo que el cliente verifique todos los requisitos especificados, además, se garantiza que el sistema desarrollado sea lo suficientemente intuitivo para que el cliente pueda usarlo sin inconvenientes [38]. En base a los requerimientos inicialmente solicitados por los directivos de la cadena de farmacias FARMECC, en la **TABLA X** se muestra el detalle de la primera prueba de aceptación junto con sus resultados, los detalles de las demás pruebas de aceptación se encuentran en el **ANEXO II** del presente documento.

TABLA X: Prueba de aceptación N°2 – Gestionar y habilitar personal.

<table>
<tr><th colspan="2">PRUEBA DE ACEPTACIÓN</th></tr>
<tr><td>Identificador (ID): PA002</td><td>Identificador historia de Usuario: HU002</td></tr>
<tr><td colspan="2">Nombre: Gestionar y habilitar personal.</td></tr>
<tr><td colspan="2">Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a:
Personal.
El usuario administrador debe llenar los campos del siguiente formulario:
Nombre.
Usuario.
Teléfono.
Correo electrónico.
Fecha de nacimiento.
para el registro de un nuevo personal</td></tr>
<tr><td colspan="2">Pasos de ejecución:
Abrir el ejecutable del sistema de escritorio.
Colocar credenciales del sistema.
Seleccionar “Personal” en el panel lateral.
Verificar los métodos de visualizar, editar, agregar y eliminar.</td></tr>
<tr><td colspan="2">Resultado deseado:
El sistema de escritorio permite realizar los métodos CRUD (crear, visualizar, actualizar y eliminar) en el módulo personal.</td></tr>
<tr><td colspan="2">Evaluación de la prueba:
El cliente aprueba al 100% con la verificación de los resultados esperados.</td></tr>
</table>

Culminada la etapa de pruebas de aceptación y en base a los resultados que se han obtenido, se contempla una aceptación del 100% por parte del cliente, tanto en funcionalidad como en interacción con los módulos presentados en el sistema de escritorio y *endpoints*. En este sentido se verifica el cumplimiento de los requerimientos iniciales, dando paso a la siguiente etapa.

3.5 *Sprint* 4. Despliegue del sistema de escritorio.

Tras la finalización de las etapas de codificación y pruebas de los diferentes módulos del sistema de escritorio, *endpoints* y en base a lo planificado en el *Sprint Backlog* el presente *Sprint* comprende las siguientes tareas:

- Despliegue del sistema de escritorio.
- Despliegue de *endpoints* a *Heroku.*

Despliegue del sistema de escritorio

Este apartado describe el proceso que se ha realizado para el despliegue a producción del sistema de escritorio a un ejecutable ".jar". El detalle de esta sección de encuentra en el **ANEXO IV** del presente documento. La etapa inicial de este proceso de identifica en la **Fig. 32** la cual corresponde a la adición de los archivos necesarios en el *IDE* de desarrollo para generar un ejecutable. Por otra parte, la **Fig. 33** presenta el despliegue del sistema de escritorio como un archivo ejecutable.

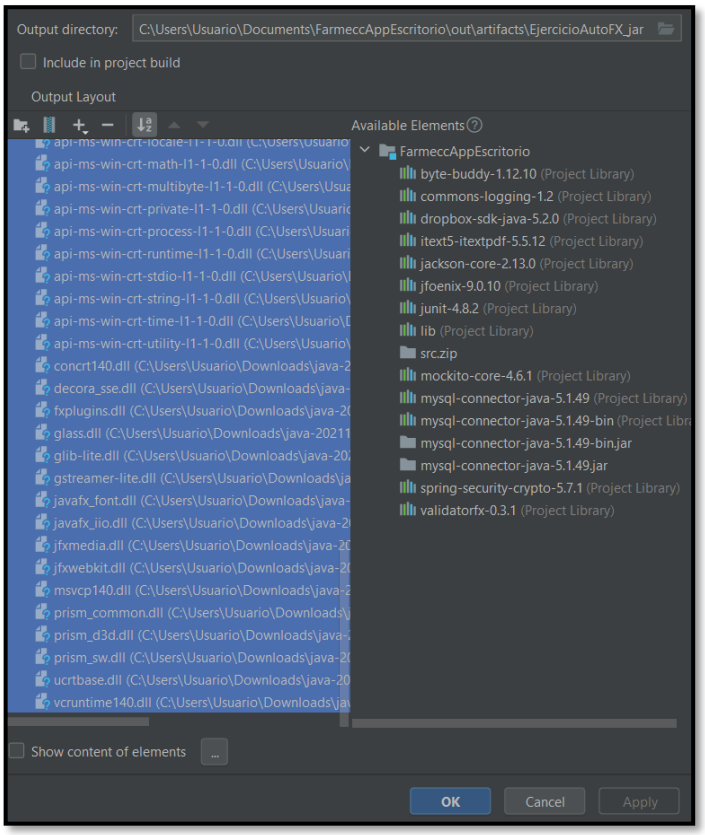

Fig. 32: Adición de archivos para generar ejecutable.

Fig. 33: Archivo ejecutable del sistema de escritorio.

Despliegue de *endpoints* a *Heroku*

Este apartado describe el proceso que se ha realizado para el despliegue a producción de los *endpoints* desarrollados a la plataforma de *Heroku*. El detalle de esta sección se puede encontrar en el **ANEXO IV**. Para la etapa inicial como se identifica en la **Fig. 34** se realiza la creación del proyecto y despliegue, mientras que en la **Fig. 35** se presenta el despliegue apropiado con la documentación y vista mediante *swagger.*

https://farmecc.herokuapp.com/doc/swagger-ui/index.html

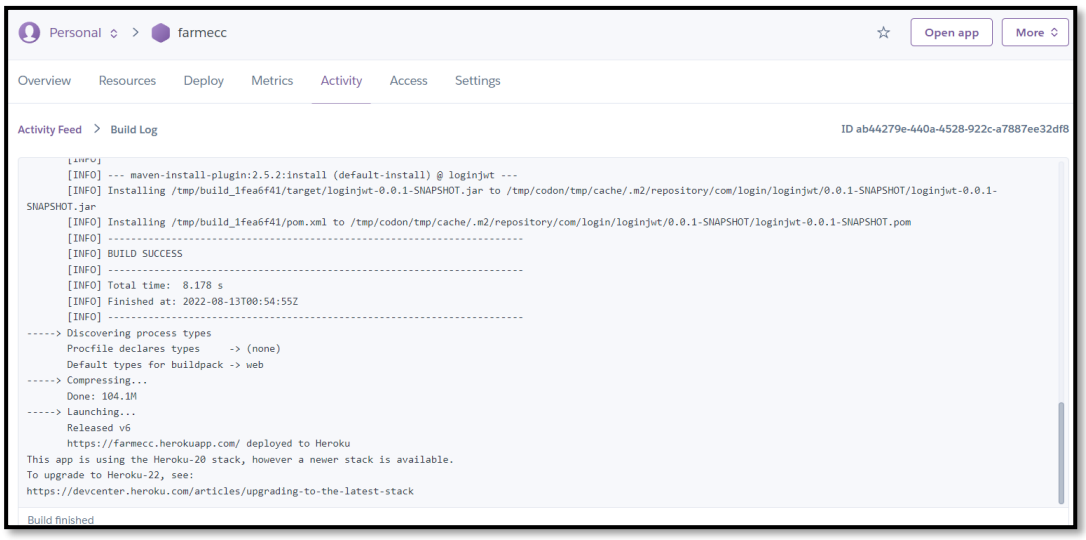

Fig. 34: Despliegue de *endpoints* a *Heroku.*

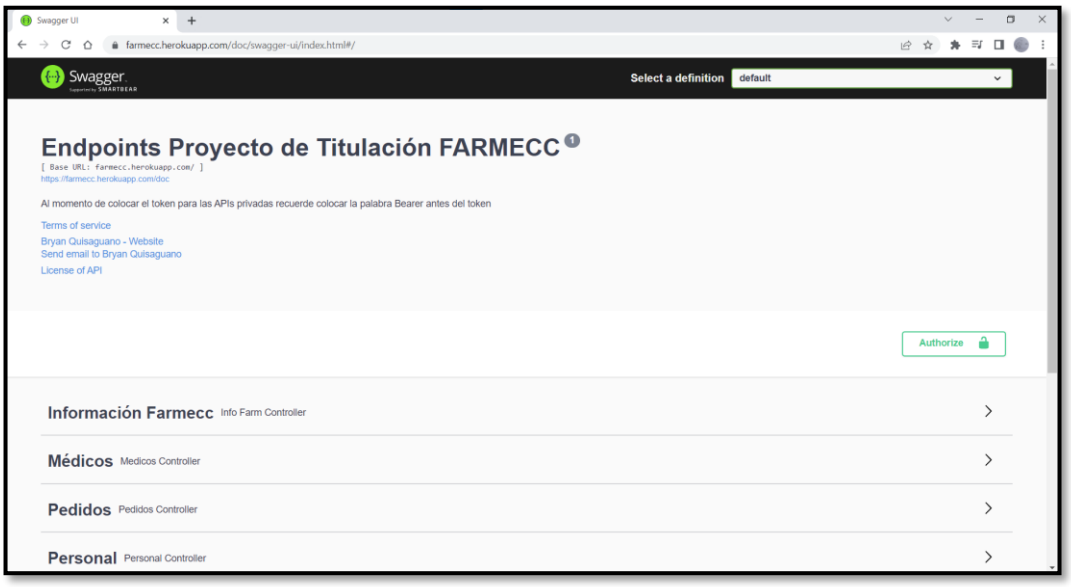

Fig. 35: *Endpoints* desplegados vista con *swagger.*

Finalmente, los directivos de la cadena de farmacias FARMECC ha generado un certificado confirmando el acatamiento de todos los requerimientos solicitados inicialmente, así como de las funcionalidades del sistema de escritorio y *endpoints*. El certificado se adjunta en el **ANEXO II**.

4 CONCLUSIONES

A continuación, se presenta las conclusiones que se han obtenido a lo largo del desarrollo del actual trabajo de integración curricular.

- El sistema de escritorio y los *endpoints* generados cumplen en su totalidad con los alcances, objetivos y requerimientos del proyecto previamente establecidos, permitiendo a la cadena de farmacias FARMECC contar con un sistema de escritorio para gestionar inventarios, permitiendo automatizar procesos con mayor eficiencia y mantener un control sobre las operaciones realizadas dentro de la cadena.

- El emplear la metodología ágil *Scrum* dentro del proceso de desarrollo del presente trabajo de integración curricular, ha dado la posibilidad de tener avances entregables en la finalización de cada *Sprint*, permitiendo de esta manera tener un sistema completo validando todas sus funcionalidades en los periodos de tiempo establecidos.

- El emplear la arquitectura Modelo-Vista-Controlador en el desarrollo tanto del sistema de escritorio como en la generación de *endpoints,* ha permitido estructurar el proyecto bajo un estándar el cual es entendible para cualquier desarrollador, en caso de integrar un nuevo miembro al equipo de desarrollo.

- El uso de una base de datos SQL, ha dado la posibilidad de gestionar toda la información garantizando la integridad de los datos, además de permitir una gestión de datos adecuada mediante las relaciones que se han creado.

- Durante el proceso de codificación se han empleado herramientas y librerías tanto para el sistema de escritorio como *endpoints,* mediante los cuales se han obtenido resultados favorables por su alta compatibilidad y un proceso de desarrollo fácil.

- La fase de pruebas que se han aplicado para el sistema de escritorio y *endpoints* han dado la posibilidad de probar todas las funcionalidades, además de comprobar su compatibilidad en diferentes sistemas operativos, así como la aceptación por parte del usuario final.

5 RECOMENDACIONES

A continuación, se presentan las recomendaciones obtenidas a lo largo del desarrollo del actual trabajo de integración curricular.

- Se recomienda que para el empleo del sistema de escritorio bajo cualquier sistema operativo se verifique la instalación de *Java* en su versión 11, ya que puede afectar de manera considerable a la ejecución del sistema de escritorio.
- Con el fin de salvaguardar la información almacenada, se recomienda realizar *backups* de la Base de Datos al menos una vez por semana, para así de garantizar la existencia de un respaldo en caso de ser necesario.
- En caso de que se requiera añadir un nuevo módulo al sistema de escritorio, se recomienda contactar al equipo de desarrollo para e informar sobre sus nuevas necesidades, para así mantener el sistema actualizado.

6 REFERENCIAS BIBLIOGRÁFICAS.

[1] Academia simple, «Academia simple,» 20 Agosto 2021. [En línea]. Available: https://academiasimple.com/como-administrar-un-negocio-pequeno-2020-facil/. [Último acceso: 21 Febrero 2022].

[2] M. Bolaño, H. L. Pérez y W. Eusebio, «Universidad de Cartagena,» 2013. [En línea]. Available: https://repositorio.unicartagena.edu.co/flip/index.jsp?pdf=/bitstream/handle/11227/734/421-%20TTG%20-%20AN%c3%81LISIS%20Y%20DISE%c3%91O%20DE%20UN%20SG%20DE%20INVENTARIO%20PARA%20LA%20FARMACIA%20DE%20LA%20FUNDACI%c3%93N%20MADRE%20HERLINDA%20MOISES%2c%20BAS. [Último acceso: 05 11 2022].

[3] M. I. Gavilánes, M. E. Espín y M. A. Palacios, «eumed,» 18 Julio 2018. [En línea]. Available: https://www.eumed.net/rev/oel/2018/07/gestion-administrativa-pymes.html. [Último acceso: 21 Febrero 2022].

[4] S. Serejski, «buenos negocios,» 2012. [En línea]. Available: http://www.buenosnegocios.com/notas/227-la-importancia-contar-herramientas-gestion. [Último acceso: 21 Febrero 2022].

[5] D. L. Quintana, «Propuesta de un sistema de gestión de inventarios para una empresa comercializadora de productos de plástico,» Universidad Peruana de Ciencias Aplicadas, Lima, 2010.

[6] S. Delgado, L. Cruz y E. Lince Olguin, «El uso de software libre en el control de inventarios: caso de estudio,» Instituto Tecnológico Superior de Tantoyuca, 2019.

[7] I. Sommerville, de *Ingeniería del Software*, Madrid, Pearson Educación. S.A., 2005, pp. 4-8.

[8] R. Pressman, Ingeniería de Software. Un enfoque práctico, México: MCGRAW-HILL, 2005.

[9] M. A. Mascheroni, C. Greiner, R. Petris, G. Dapozo y M. Estayno, «SEDICI Repositorio Institucional de la UNPL,» 06 Agosto 2012. [En línea]. Available: http://sedici.unlp.edu.ar/handle/10915/19202. [Último acceso: 28 Mayo 2022].

[10] Internet Ya, «Internet Ya,» 13 Julio 2020. [En línea]. Available: https://www.internetya.co/aplicaciones-web-vs-escritorio-2/. [Último acceso: 28 Mayo 2022].

[11] Oracle, «Oracle,» 2021. [En línea]. Available: https://www.oracle.com/ar/database/what-is-a-relational-database/. [Último acceso: 12 Mayo 2022].

[12] AlwaysData, «AlwaysData,» 2022. [En línea]. Available: https://www.alwaysdata.com/en/. [Último acceso: Mayo 2022].

[13] Wikipedia, «Wikipedia,» 25 Abril 2022. [En línea]. Available: https://es.wikipedia.org/wiki/Java_(lenguaje_de_programación). [Último acceso: 12 Mayo 2022].

[14] Y. Muradas, «Open Webinars,» 05 Junio 2018. [En línea]. Available: https://openwebinars.net/blog/conoce-que-es-spring-framework-y-por-que-usarlo/. [Último acceso: 14 Mayo 2022].

[15] E. Haro, T. Guarda, A. O. Zambrano y G. Ninahualpa, «ProQuest,» Junio 2019. [En línea]. Available: https://www.proquest.com/openview/a78cfaa62708fd24f38ac8d1025050eb/1?pq-origsite=gscholar&cbl=1006393. [Último acceso: Mayo 2022].

[16] G. Grefory, C. Bauer y G. King, Java Persistence with Hibernate, Manning, 2015.

[17] V. R. Anshu Soni, «API Features Individualizing of Web Services: REST and SOAP,» *International Journal of Innovative Technology and Exploring Engineering (IJITEE),* vol. 8, nº 9S, pp. 664-671, 2019.

[18] Epitech España, «Epitech-it,» 3 Junio 2021. [En línea]. Available: https://www.epitech-it.es/backend/. [Último acceso: 15 Mayo 2022].

[19] R. E. Stake, «Estudio intensivo de los métodos de investigación con estudio de casos,» de *Investigación con estudio de casos*, Madrid, Morata, 2007, pp. 11-12.

[20] E. G. Maida y J. Pacienzia, «Repositorio Institucional UCA,» 2015. [En línea]. Available: https://repositorio.uca.edu.ar/handle/123456789/522. [Último acceso: 28 Mayo 2022].

[21] Aritmetrics, «Arimetrics.com,» 15 Julio 2021. [En línea]. Available: https://www.arimetrics.com/glosario-digital/scrum. [Último acceso: 21 Marzo 2022].

[22] M. A. Alvarez, «Desarrolloweb.com,» 28 Julio 2020. [En línea]. Available: https://desarrolloweb.com/articulos/que-es-mvc.html. [Último acceso: 10 julio 2022].

[23] E. Abellán, «We are marketing,» Mayo 2020. [En línea]. Available: https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html. [Último acceso: 10 Junio 2022].

[24] J. M. de Agar Tirado, «¿Qué es Scrum?,» 2020. [En línea]. Available: https://mamaqueesscrum.com/2020/04/29/que-es-un-development-team-os-proponemos-una-dinamica/. [Último acceso: 15 07 2021].

[25] C. A. Guerra, «Sg.com,» 28 Diciembre 2007. [En línea]. Available: https://sg.com.mx/revista/17/obtencion-requerimientos-tecnicas-y-estrategia. [Último acceso: 15 Junio 2022].

[26] M. Rehkopf, «Atlassian Agile coach,» s.f. [En línea]. Available: https://www.atlassian.com/es/agile/project-management/user-stories. [Último acceso: 15 07 2021].

[27] EALDE, «EALDE,» 07 Agosto 2019. [En línea]. Available: https://www.ealde.es/product-backlog-sprint-backlog/. [Último acceso: 30 Junio 2022].

[28] M. Garcia, «ITtude,» 17 Julio 2020. [En línea]. Available: https://ittude.com.ar/b/scrum/que-es-el-sprint-backlog/. [Último acceso: 30 Julio 2022].

[29] Oracle, «Oracle,» 2022. [En línea]. Available: https://www.oracle.com/java/technologies/javase/javafxscenebuilder-info.html. [Último acceso: 19 Mayo 2022].

[30] D. Martin, «velneo,» 7 Marzo 2019. [En línea]. Available: https://velneo.es/herramientas-software-2019/. [Último acceso: 19 Mayo 202].

[31] Postgresql, «PostgreSQL,» 2022. [En línea]. Available: https://www.postgresql.org/about/. [Último acceso: 19 Mayo 2022].

[32] Heroku, «Heroku,» 2022. [En línea]. Available: https://www.heroku.com/what. [Último acceso: 19 Mayo 2022].

[33] Spring Initializr, «Spring Initializr,» VMware, Inc, 2013. [En línea]. Available: https://start.spring.io/. [Último acceso: 19 Mayo 2022].

[34] Project Lombok Authors, «Project Lombok,» 2022. [En línea]. Available: https://projectlombok.org/. [Último acceso: 19 Mayo 2022].

[35] Spring, «Spring,» 2022. [En línea]. Available: https://spring.io/projects/spring-security. [Último acceso: 19 Mayo 2022].

[36] iTextGroup, «itextpdf,» 2022. [En línea]. Available: https://itextpdf.com/en. [Último acceso: 19 Mayo 2022].

[37] S. Viteri Arias, T. Mayorga Soria, P. Navas Moya y P. Molina Palma, «Control de calidad del software mediante pruebas automatizadas de integración y pruebas unitarias,» *Ciencia Digital,* vol. 3, nº TICs en la Educación, pp. 101-115, 01 Julio 2018.

[38] G. J. Michael, H. A. Violena, A. M. Dialexis, C. D. Rosalía, C. A. Lucy y F. P. Yamilis, «Pruebas de aceptación para un software con la presencia de una entidad certificadora de la calidad,» *Revista Cubana de Ciencias Informáticas,* vol. 1, pp. 84-93, 2007.

7 ANEXOS

A continuación, se muestra la división de los Anexos que se han utilizado para el desarrollo del sistema de escritorio y *endpoints*.

- **ANEXO I.** Resultado del programa antiplagio *Turnitin* y certificado emitido por los directivos de la cadena de farmacias FARMECC.
- **ANEXO II.** Manual Técnico.
- **ANEXO III.** Manual de Usuario.
- **ANEXO IV.** Manual de Instalación.

ANEXO I

A continuación, se presenta el certificado que el Director de Tesis ha emitido y en donde se evidencia el resultado que se ha obtenido en la herramienta anti-plagio Turnitin.

ESCUELA POLITÉCNICA NACIONAL
ESCUELA DE FORMACIÓN DE TECNÓLOGOS
CAMPUS POLITÉCNICO "ING. JOSÉ RUBÉN ORELLANA"

CERTIFICADO DE ORIGINALIDAD

Quito, D.M. 22 de agosto de 2022

De mi consideración:

Yo, Loarte Cajamarca Byron Gustavo, en calidad de Director del Trabajo de Integración Curricular titulado Desarrollo de un sistema de escritorio asociado al DESARROLLO DE SISTEMA PARA LA GESTIÓN DEL INVENTARIO EN FARMECC elaborado por el estudiante Bryan Armando Quisaguano Casade la carrera en Tecnología Superior en Desarrollo de Software, certifico que he empleado la herramienta Turnitin para la revisión de originalidad del documento escrito secciones: Descripción del componente desarrollado, Metodología, Resultados, Conclusiones y Recomendaciones, producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 10%.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el informe generado por la herramienta Turnitin.

Atentamente,

Loarte Cajamarca Byron Gustavo
Profesor Ocasional a Tiempo Completo
Escuela de Formación de Tecnólogos

ANEXO II

Recopilación de requerimientos

A continuación, la **TABLA XI** se presentan los requerimientos que ha sido obtenidos a partir de lo solicitado por parte del *Product Owner*.

TABLA XI: Recopilación de requerimientos.

Recopilación de requerimientos		
Tipo de sistema	**ID-RR**	**Enunciado del Ítem**
Sistema de escritorio	**RR003**	Como usuario administrador, necesita realizar lo siguiente: • Gestión de las sucursales.
	RR004	Como usuario administrador, necesita realizar lo siguiente: • Gestión de productos.
	RR005	Como usuario administrador, necesita realizar lo siguiente: • Gestión de ventas.
	RR006	Como usuario administrador, necesita realizar lo siguiente: • Generar reportes.
	RR007	Como usuario administrador, necesita realizar lo siguiente: • Gestión de médicos.
	RR008	Como usuario administrador, necesita realizar lo siguiente: • Gestión de información general
	RR009	Como usuario empleado, necesita realizar lo siguiente: • Gestión de productos.
	RR010	Como usuario empleado, necesita realizar lo siguiente: • Gestión de ventas.
	RR011	Como usuario empleado, necesita realizar lo siguiente: • Generar reportes
	RR012	Como usuario administrador y empleado necesita generar varios *endpoints* para realizar lo siguiente: • Visualizar página informativa.
	RR013	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Iniciar sesión. • Cerrar sesión.

	RR014	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Gestión de sucursales.
	RR015	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Visualizar productos
	RR016	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Visualizar reportes.
	RR017	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Disponibilidad de stock de productos.
	RR018	Como usuario cliente necesita generar varios *endpoints* para realizar lo siguiente: • Realizar cotización.
	RR019	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Visualizar ventas por sucursales.
	RR020	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Visualizar personal.
	RR021	Como usuario administrador necesita generar varios *endpoints* para realizar lo siguiente: • Gestionar pedidos con proveedores.

Historias de Usuario

Culminada la etapa de Recopilación de requerimientos, se procede a desarrollar cada una de las Historias de Usuario para el sistema de escritorio. En ese sentido, se presentan las 18 Historias de Usuario escritas en base a los requerimientos del *Product Owner*.

TABLA XII: Historia de usuario 2 – Gestionar personal.

<table>
<tr><th colspan="2">HISTORIA DE USUARIO</th></tr>
<tr><td>Identificador (ID): HU002</td><td>Usuario: Administrador.</td></tr>
<tr><td colspan="2">Nombre Historia: Gestionar personal.</td></tr>
<tr><td>Prioridad en negocio: Alta</td><td>Riesgo en desarrollo: Medio</td></tr>
<tr><td colspan="2">Iteración Asignada: 1</td></tr>
<tr><td colspan="2">Responsable (es): Bryan Quisaguano</td></tr>
<tr><td colspan="2">Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a:
Personal.
El usuario administrador debe llenar los campos del siguiente formulario:
Nombre.
Apellido.
Usuario.
Teléfono.
Correo electrónico.
Fecha de nacimiento.
Sucursal asignada
para el registro de un nuevo personal.</td></tr>
<tr><td colspan="2">Observación:
El usuario administrador debe iniciar sesión de manera obligatoria para poder gestionar toda la información anteriormente mencionada. Además, debe existir al menos una sucursal previamente registrada.</td></tr>
</table>

TABLA XIII: Historia de usuario 3 – Gestionar sucursales.

HISTORIA DE USUARIO	
Identificador (ID): HU003	**Usuario**: Administrador.
Nombre Historia: Gestionar y habilitar sucursales.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Iteración Asignada: 1	
Responsable (es): Bryan Quisaguano	
Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a: • Sucursales. Para el registro y modificación de una sucursal se lo realiza mediante un formulario donde se ingresa la siguiente información: • Nombre. • Ciudad. • Dirección. • Teléfono.	
Observación: El usuario administrado debe iniciar sesión para poder gestionar toda la información respecto a sucursales.	

TABLA XIV: Historia de usuario 4 - Gestionar productos.

HISTORIA DE USUARIO	
Identificador (ID): HU004	**Usuario**: Administrador, Empleado.
Nombre Historia: Gestionar productos.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Iteración Asignada: 1	
Responsable (es): Bryan Quisaguano	
Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a: • Productos Por otra parte, el usuario empleado tiene la posibilidad de crear, visualizar y modificar esta misma información, pero no puede eliminar.	

Para el registro y modificación de un producto se lo realiza mediante un formulario donde se ingresa la siguiente información: • Nombre comercial. • Nombre genérico. • Descripción del producto. • Precio de compra. • Precio unitario de venta al público. • Cantidad. • Registro sanitario. • Fecha de ingreso. • Fecha de caducidad. • Ubicación. • Tipo de producto. • Sucursal perteneciente.
Observación: El formulario valida cada uno de los campos requeridos. Además, para el caso de ubicación y tipo de producto se debe desplegar un menú con opciones predefinidas.

TABLA XV: Historia de usuario 5 – Gestionar ventas.

HISTORIA DE USUARIO	
Identificador (ID): HU005	**Usuario**: Administrador, Empleado
Nombre Historia: Gestionar ventas.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Iteración Asignada: 1	
Responsable (es): Bryan Quisaguano	
Descripción: El usuario administrador tiene la posibilidad de crear, visualizar y eliminar información respecto a: • Ventas Por otra parte, el usuario empleado tiene la posibilidad de crear y visualizar esta misma información, pero no puede eliminar. Para el registro de ventas se lo realiza mediante la búsqueda y selección de un producto colocando la cantidad requerida y además un descuento en caso de ser necesario. Para esto el usuario que inicie sesión debe llenar un formulario	

con la siguiente información: • Cantidad requerida. • Descuento. Una vez realizada la venta esta es almacenada y se realiza el descuento automático del *stock* del producto.
Observación: El formulario verifica que la cantidad requerida por el cliente se encuentre disponible en el *stock* del producto, caso contrario la venta no se puede procesar.

TABLA XVI Historia de usuario 6 - Generar reportes.

HISTORIA DE USUARIO	
Identificador (ID): HU006	**Usuario**: Administrador, Empleado.
Nombre Historia: Generar reportes.	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Iteración Asignada: 1	
Responsable (es): Bryan Quisaguano	
Descripción: El usuario administrador puede generar reportes de las ventas realizadas en cada sucursal, reportes de ventas globales y reportes de ventas destinados al Ministerio de Salud Pública del país. Por otra parte, el usuario empleado únicamente puede generar reportes de ventas diario.	
Observación: Para el usuario administrador estos reportes pueden ser generados por un día específico o en un intervalo fechas.	

TABLA XVII Historia de usuario 7 – Gestionar médicos.

HISTORIA DE USUARIO	
Identificador (ID): HU007	**Usuario**: Administrador.
Nombre Historia: Gestionar médicos.	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Iteración Asignada: 1	

Responsable (es): Bryan Quisaguano
Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a: • Médicos. Para el registro y modificación de un médico se lo realiza mediante un formulario donde se ingresa la siguiente información: • Nombre. • Apellido. • Descripción. • Imagen.
Observación: El usuario administrado debe iniciar sesión para poder gestionar toda la información respecto a médicos.

TABLA XVIII Historia de usuario 8 – Gestionar información general.

HISTORIA DE USUARIO	
Identificador (ID): HU008	**Usuario**: Administrador.
Nombre Historia: Gestionar información general.	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Iteración Asignada: 1	
Responsable (es): Bryan Quisaguano	
Descripción: El usuario administrador tiene la posibilidad de visualizar y modificar información respecto a: • Información general. Para el registro y modificación de un médico se lo realiza mediante un formulario donde se ingresa la siguiente información: • Misión. • Visión. • Nosotros.	
Observación: El usuario administrado debe iniciar sesión para poder gestionar toda la información respecto a información general.	

TABLA XIX Historia de usuario 9 - Generar *endpoints* para visualizar página informativa.

HISTORIA DE USUARIO	
Identificador (ID): HU009	**Usuario**: Administrador, Cliente.
Nombre Historia: Generar *endpoints* para visualizar página informativa.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite visualizar toda la información necesaria para presentar una página informativa, esta información concierne a: • Misión, visión y nosotros. • Médicos. • Sucursales.	
Observación: Estos *endpoints* mencionados anteriormente están disponibles para cualquier aplicación del lado del cliente.	

TABLA XX Historia de usuario 10 - Generar *endpoints* para inicio y cierre de sesión.

HISTORIA DE USUARIO	
Identificador (ID): HU010	**Usuario**: Administrador.
Nombre Historia: Generar *endpoints* para inicio y cierre de sesión.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite: • Iniciar Sesión. • Cerrar Sesión.	
Observación: El usuario administrador es el único que puede acceder a los *endpoints* listados anteriormente. Además, se debe generar un token de acceso disponible para acceder a los *endpoints* privados.	

TABLA XXI Historia de usuario 11 - Generar *endpoints* para gestionar sucursales.

HISTORIA DE USUARIO	
Identificador (ID): HU011	**Usuario**: Administrador.
Nombre Historia: Generar *endpoints* para gestionar sucursales.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que da la posibilidad de gestionar las sucursales, esto permite: • Crear sucursales. • Modificar sucursales. • Visualizar sucursales. • Eliminar sucursales.	
Observación: El usuario administrador es el único que puede acceder a los *endpoints* listados anteriormente.	

TABLA XXII Historia de usuario 12 - Generar *endpoint* para visualizar productos.

HISTORIA DE USUARIO	
Identificador (ID): HU012	**Usuario**: Administrador, Cliente.
Nombre Historia: Generar *endpoint* para visualizar productos.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador y cliente permite generar un *endpoint* que permite visualizar toda la información de los productos disponibles en cada sucursal.	
Observación: Este *endpoint* mencionado anteriormente está disponible para cualquier aplicación del lado del cliente.	

TABLA XXIII Historia de usuario 13 - Generar *endpoints* para visualizar reportes.

HISTORIA DE USUARIO	
Identificador (ID): HU013	**Usuario**: Administrador.
Nombre Historia: Generar *endpoints* para visualizar reportes.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador puede generar varios *endpoints* que permiten visualizar reportes como: • Reportes generales de cada sucursal. • Reportes globales de ventas. • Reportes entre fechas específicas.	
Observación: El usuario administrador es el único que puede acceder a estos *endpoints*.	

TABLA XXIV Historia de usuario 14 - Generar *endpoint* para visualizar disponibilidad de *stock*.

HISTORIA DE USUARIO	
Identificador (ID): HU014	**Usuario**: Administrador.
Nombre Historia: Generar *endpoint* para visualizar disponibilidad de *stock*.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano.	
Descripción: El sistema a través del usuario administrador puede genera un *endpoint* que permiten visualizar la disponibilidad de *stock* de cada producto.	
Observación: El usuario administrador es el único que puede acceder a este *endpoint*.	

TABLA XXV Historia de usuario 15 - Generar *endpoints* para realizar cotizaciones

HISTORIA DE USUARIO	
Identificador (ID): HU015	**Usuario**: Cliente.

Nombre Historia: Generar *endpoints* para realizar cotizaciones.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario cliente puede generar varios *endpoints* que permiten realizar cotizaciones, con la información general de los productos.	
Observación: Estos *endpoints* mencionados anteriormente están disponibles para cualquier aplicación del lado del cliente.	

TABLA XXVI Historia de usuario 16 - Generar *endpoints* para visualizar ventas por sucursales.

HISTORIA DE USUARIO	
Identificador (ID): HU016	**Usuario**: Administrador.
Nombre Historia: Generar varios *endpoints* para visualizar ventas por sucursales.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite visualizar las ventas realizadas por sucursales.	
Observación: El usuario administrador es el único que puede acceder a estos *endpoints*.	

TABLA XXVII Historia de usuario 17 - Generar *endpoint* para visualizar personal.

HISTORIA DE USUARIO	
Identificador (ID): HU017	**Usuario**: Administrador.
Nombre Historia: Generar *endpoint* para visualizar personal.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador permite generar un	

endpoint que permite visualizar toda la información acerca del personal.
Observación: El usuario administrador es el único que puede acceder a este *endpoint*.

TABLA XXVIII Historia de usuario 18 - Generar varios *endpoints* para gestionar pedidos con proveedores.

HISTORIA DE USUARIO	
Identificador (ID): HU018	**Usuario**: Administrador.
Nombre Historia: Generar varios *endpoint* para gestionar pedidos con proveedores.	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alto
Iteración Asignada: 2	
Responsable (es): Bryan Quisaguano	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite la gestión de proveedores, esto permite: • Crear proveedor. • Modificar proveedor. • Visualizar proveedor. • Eliminar proveedor. De la misma manera permite la gestión de pedidos con los mismos.	
Observación: El usuario administrador es el único que puede acceder a estos *endpoints*.	

Product Backlog

La **TABLA XXIX** se ordena la prioridad de los requerimientos para el sistema de escritorio. Estos requerimientos se clasifican en base a las necesidades del *Product Owner* y el grado de complejidad en el desarrollo de cada ítem.

TABLA XXIX: *Product Backlog.*

ELABORACIÓN DEL *PRODUCT BACKLOG*				
ID – HU	**HISTORIA DE USUARIO**	**ITERACIÓN**	**ESTADO**	**PRIORIDAD**
HU003	Gestionar sucursales.	1	Finalizado	Media
HU004	Gestionar productos.	1	Finalizado	Alta
HU005	Gestionar ventas.	1	Finalizado	Alta
HU006	Generar reportes.	1	Finalizado	Media
HU007	Gestión médicos	1	Finalizado	Media
HU008	Gestión información general	1	Finalizado	Media
HU009	Generar *endpoints* para visualizar página informativa.	2	Finalizado	Media
HU010	Generar *endpoints* para inicio y cierre de sesión.	2	Finalizado	Media
HU011	Generar *endpoints* para gestionar sucursales.	2	Finalizado	Alta
HU012	Generar *endpoint* para visualizar productos.	2	Finalizado	Alta
HU013	Generar *endpoints* para visualizar reportes.	2	Finalizado	Media
HU014	Generar *endpoint* para visualizar disponibilidad de *stock*.	2	Finalizado	Media
HU015	Generar *endpoints* para realizar cotizaciones.	2	Finalizado	Media
HU016	Generar *endpoints* para visualizar ventas por sucursales.	2	Finalizado	Media
HU017	Generar *endpoint* para visualizar	2	Finalizado	Baja

	personal			
HU018	Generar *endpoints* para gestionar pedidos con proveedores.	2	Finalizado	Baja

Sprint Backlog

En la **TABLA XXX** se presentan los cinco *Sprints* desarrollados para el sistema de escritorio, se listan cada una de las actividades y el tiempo establecido para cumplir con los entregables establecidos con el *Product Owner*.

TABLA XXX: *Sprint Backlog*.

ELABORACIÓN DEL *SPRINT BACKLOG*						
ID – SB	**NOMBRE**	**MÓDULO**	**ID-HU**	**HISTORIA DE USUARIO**	**TAREAS**	**TIEMPO ESTIMADO**
SB001	Diseño e implementación de la interfaz destinada al usuario administrador y empleado con sus	Modulo inicio de sesión	HU001	Inicio y cierre de sesión.	• Diseño e implementación de interfaz para inicio y cierre de sesión de cada usuario. • Validación de credenciales de usuarios. • Carga de los respectivos módulos de cada usuario	70H

	respectivos módulos.	Modulo personal	HU002	Gestionar personal.	• Diseño e implementación de interfaz para gestión de personal. • Diseño e implementación de formulario para registrar, visualizar, editar y eliminar información del personal. • Guardar formulario. • Establecer reglas de validación para cada campo del formulario. • Pruebas de registro, visualización, edición y eliminación de personal.	
		Modulo sucursales	HU003	Gestionar sucursales.	• Diseño e implementación deinterfaz para gestionar sucursales. • Diseño e implementación de formulario para registrar, visualizar, modificar y eliminar información de sucursales. • Guardar formulario. • Establecer reglas de validación para cada campo del formulario. • Prueba para registrar, visualizar, modificar y eliminar información de	

					sucursales.	
		Módulo productos	HU004	Gestionar productos.	• Diseño e implementación de interfaz para gestionar productos. • Diseño e implementación de formulario para registrar, visualizar, modificar y eliminar productos. • Establecer reglas de validación para cada campo del formulario. • Guardar formulario. • Prueba para registrar, visualizar, modificar y eliminar información de productos.	
		Módulo ventas	HU005	Gestionar ventas.	• Diseño e implementación de interfaz para gestionar ventas. • Diseño e implementación de formulario para registrar, visualizar y eliminar ventas. • Establecer reglas de validación para cada campo del formulario. • Guardar formulario. • Prueba de registrar, visualizar y eliminar información de ventas.	

		Módulo reportes	HU006	Generar reportes.	• Diseño e implementación de interfaz para generación de reportes. • Establecer parámetros para la generación de reportes. • Establecer formato de visualización de reportes.	
		Módulos médicos	HU007	Gestionar médicos.	• Diseño e implementación de interfaz para gestionar médicos. • Diseño e implementación de formulario para registrar, visualizar, modificar y eliminar médicos. • Establecer reglas de validación para cada campo del formulario. • Guardar formulario. • Prueba para registrar, visualizar, modificar y eliminar información de médicos.	

		Módulo información general	HU008	Gestionar información general.	• Diseño e implementación de interfaz para gestionar información general. • Diseño e implementación de formulario para visualizar ymodificar información general. • Establecer reglas de validación para campo del formulario. • Guardar formulario. • Prueba para visualizar y modificar formación general.	
SB002	Diseño e implementación de *endpoints* para el consumo desde un sistema *web*.	Módulo información general	HU009	Generar *endpoints* para visualizar página informativa.	• Diseño e implementación de *endpoints* para la visualización de página informativa.	70H
		Módulo de inicio y cierre de sesión	HU010	Generar *endpoints* para inicio y cierre de sesión.	• Diseño e implementación de endpoints para el inicio y cierre de sesión.	

		Módulos sucursales	HU011	Generar *endpoints* para gestionar sucursales.	• Diseño e implementación de *endpoints* para registrar, visualizar, editar y eliminar datos de sucursales. • Realizar consultas a la base de datos.	
		Módulo productos	HU012	Generar *endpoint* para visualizar productos.	• Diseño e implementación de *endpoint* para visualizar los productos registrados.	
		Módulo reportes	HU013	Generar *endpoints* para visualizar reportes.	• Diseño e implementación de *endpoints* para la visualización de reportes.	
		Módulo *stock*	HU014	Generar *endpoint* para visualizar disponibilidad de *stock*.	• Diseño e implementación de *endpoint* para la visualización de la disponibilidad de *stock*.	
		Módulo cotizaciones	HU015	Generar *endpoints* para realizar cotizaciones.	• Diseño e implementación de *endpoints* para realizar cotizaciones.	

		Módulo ventas	HU016	Generar *endpoints* visualizar ventas por sucursales.	• Diseño e implementación de *endpoints* para visualizar ventas por sucursales.	
		Modulo personal	HU017	Generar *endpoint* para visualizar personal.	• Diseño e implementación de *endpoint* para visualizar personal.	
		Modulo proveedor	HU018	Generar *endpoints* para gestionar pedidos con proveedores.	• Diseño e implementación de *endpoints* para registrar, visualizar, editar y eliminar datos de proveedores. • Realizar consultas a la base de datos.	
SB003	Pruebas en el sistema de escritorio y *endpoints.*	• Pruebas unitarias. • Pruebas de compatibilidad. • Pruebas de aceptación.				20H
SB004	Despliegue del sistema de escritorio y *endpoints.*	• Despliegue del sistema de escritorio. • Despliegue de *endpoints* a *Heroku*.				10H

Documentación	• Informe Técnico. • Anexos.	50H
TOTAL		**240 H**

Diseño de interfaces

A continuación, se presentan los prototipos de cada uno de los módulos del sistema de escritorio, en los cuales se identifica el diseño de las interfaces para cada rol de usuario: administrador y empleado. Desde la **Fig. 36** a la **Fig. 49** se presentan las interfaces del sistema de escritorio.

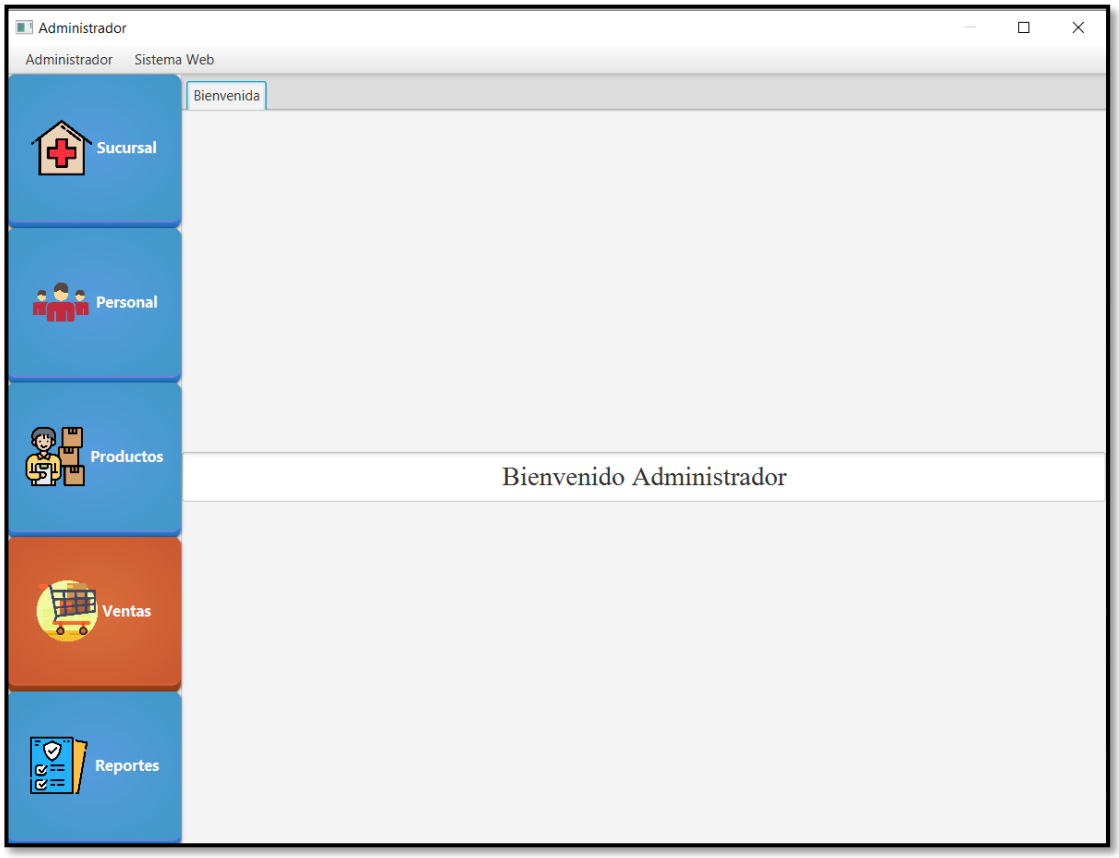

Fig. 36: Diseño de interfaz - Bienvenida panel usuario administrador.

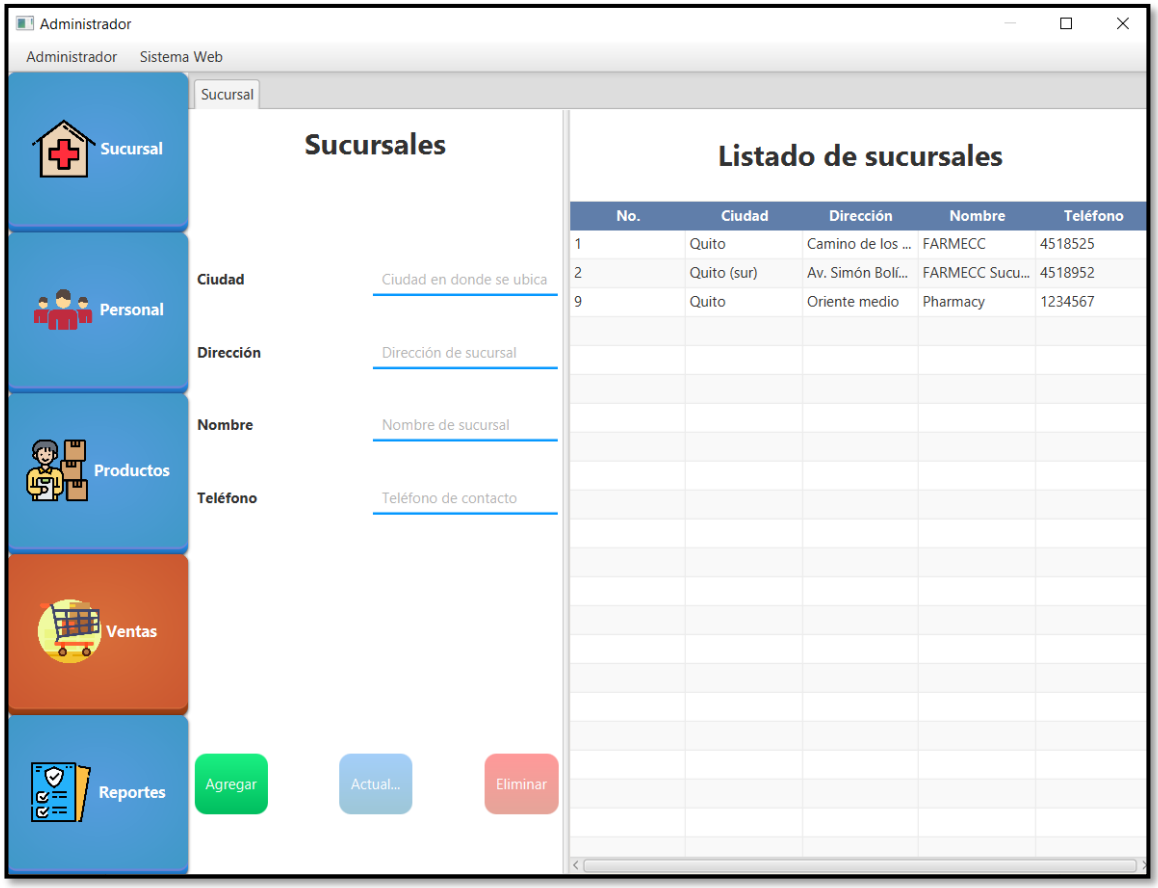

Fig. 37: Diseño de interfaz – Módulo gestionar sucursales.

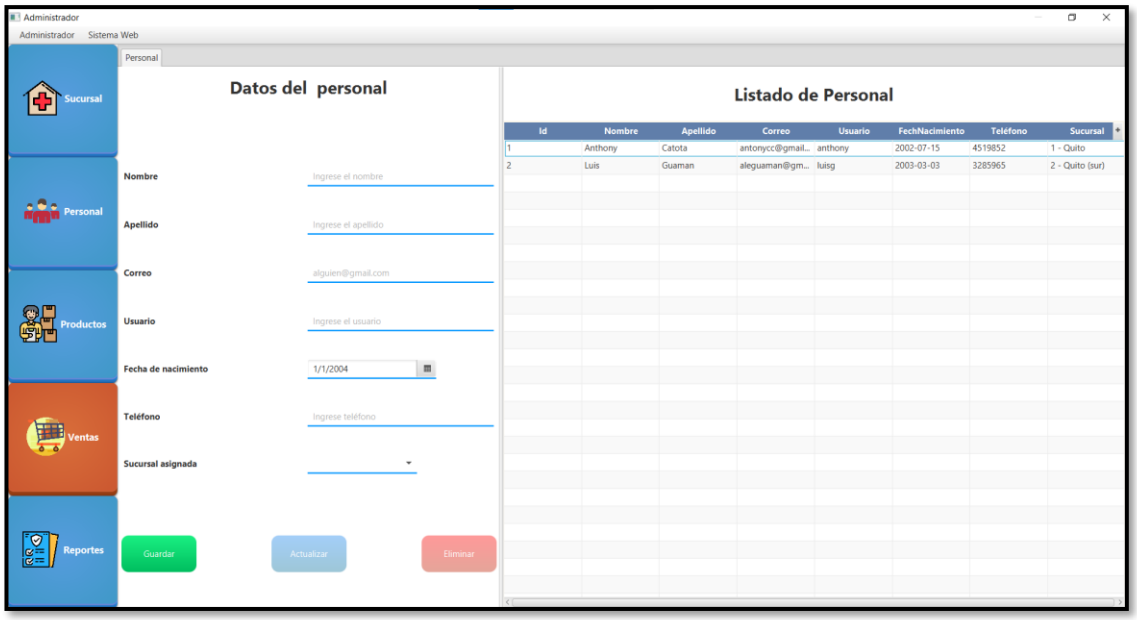

Fig. 38: Diseño de interfaz – Módulo gestionar personal.

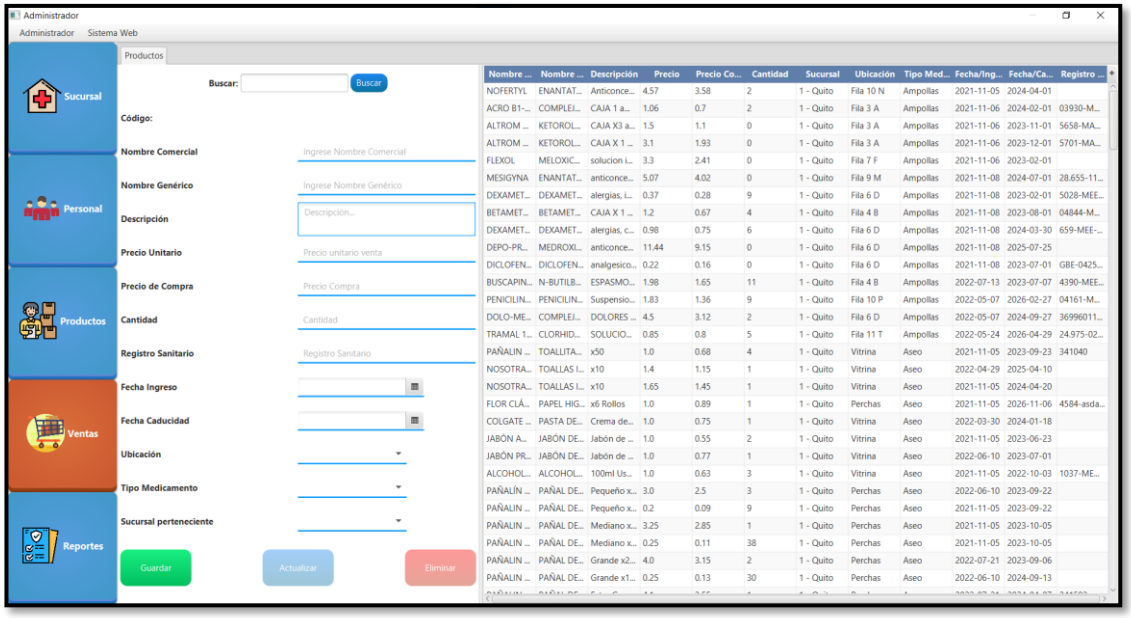

Fig. 39: Diseño de interfaz – Módulo gestionar productos.

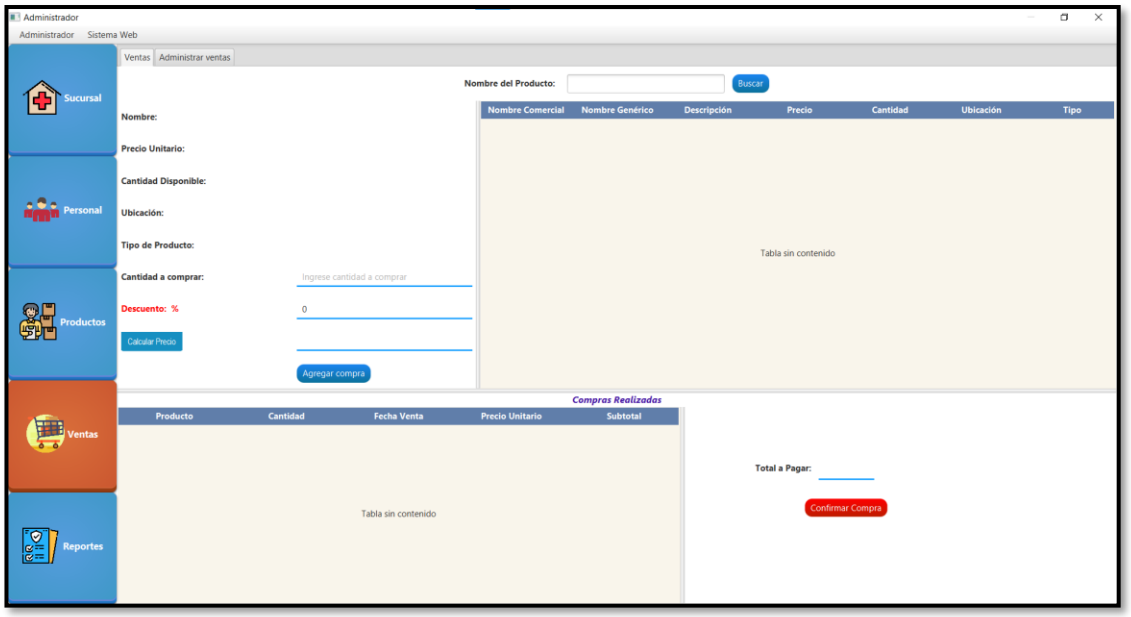

Fig. 40: Diseño de interfaz – Módulo ventas.

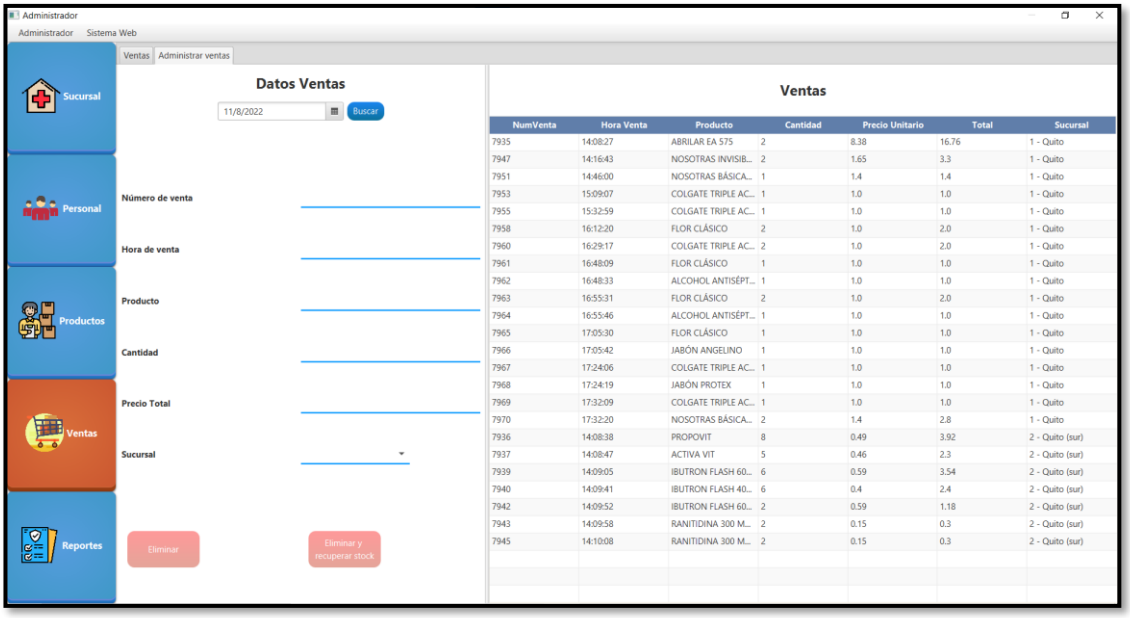

Fig. 41: Diseño de interfaz – Módulo administrar ventas.

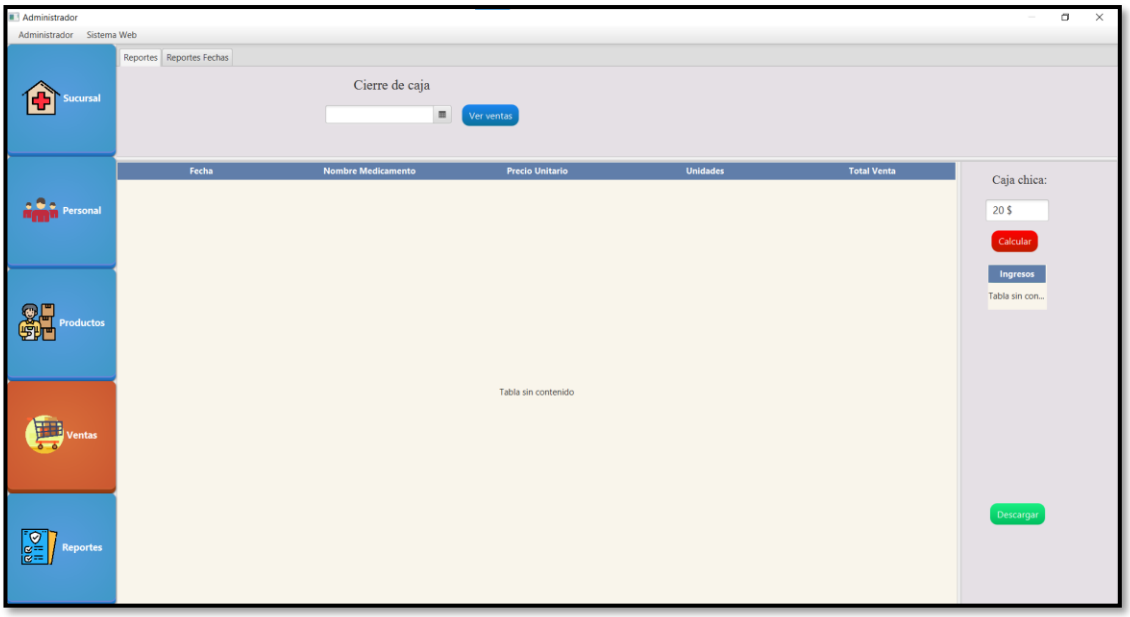

Fig. 42: Diseño de interfaz – Módulo reporte cierre de caja.

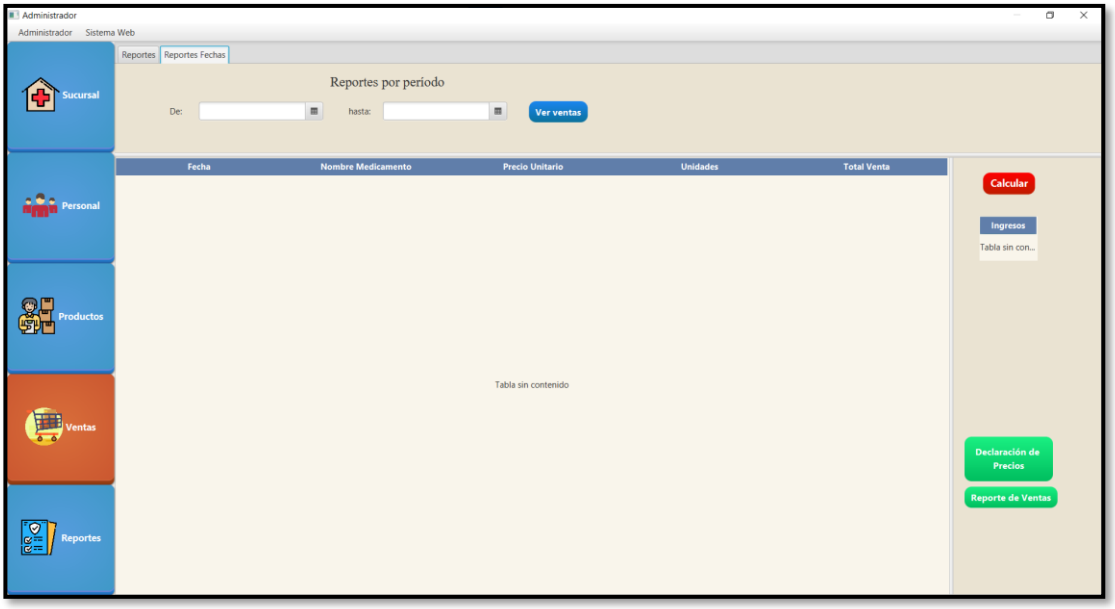

Fig. 43: Diseño de interfaz – Módulo reportes por período.

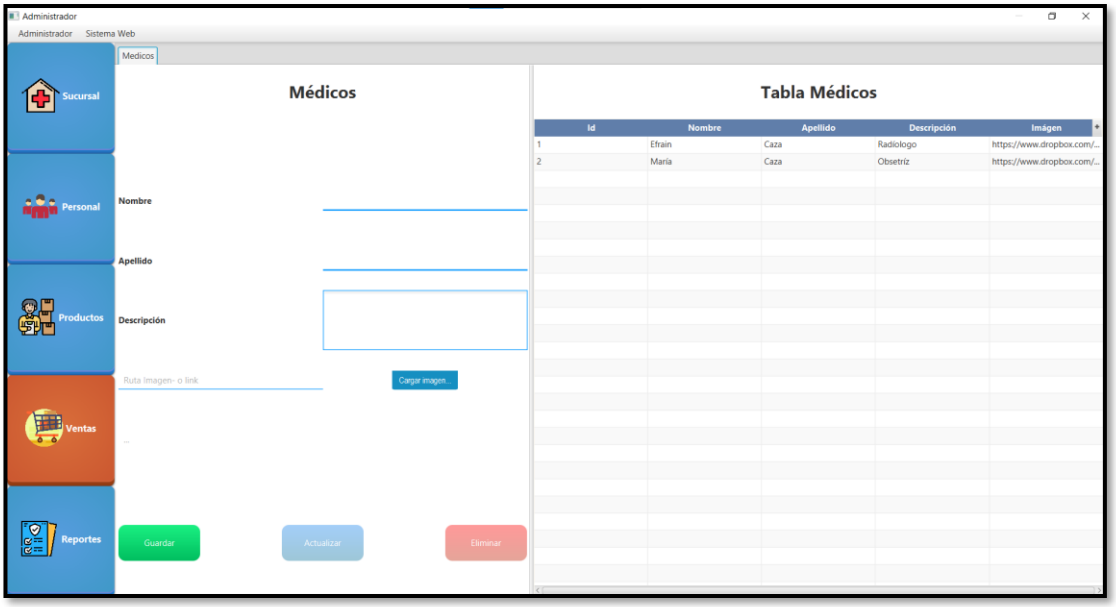

Fig. 44: Diseño de interfaz – Módulo gestionar médicos.

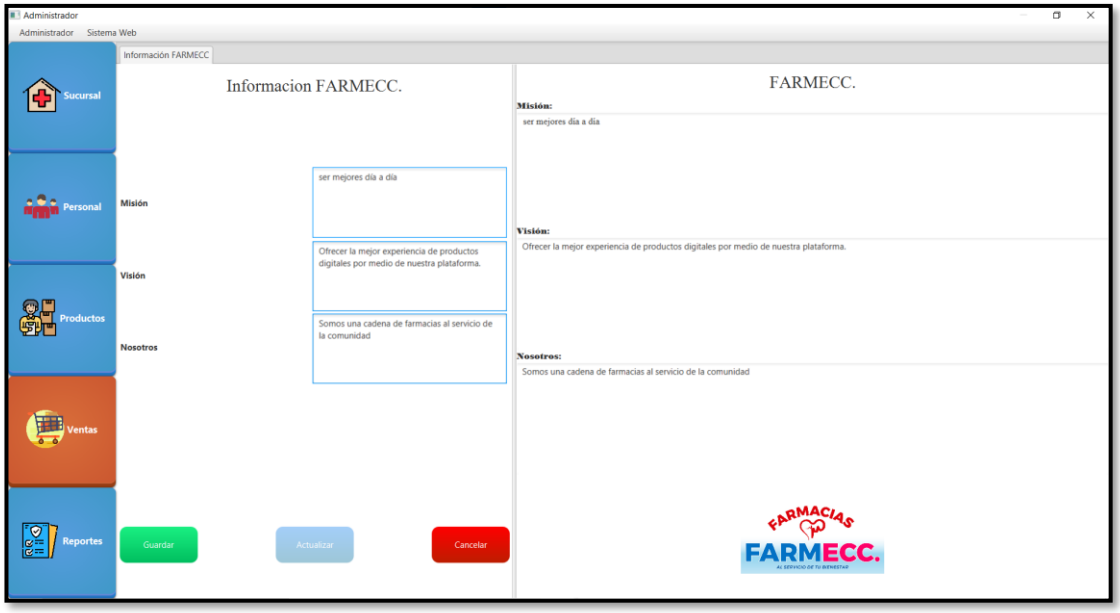

Fig. 45: Diseño de interfaz – Módulo gestionar información general.

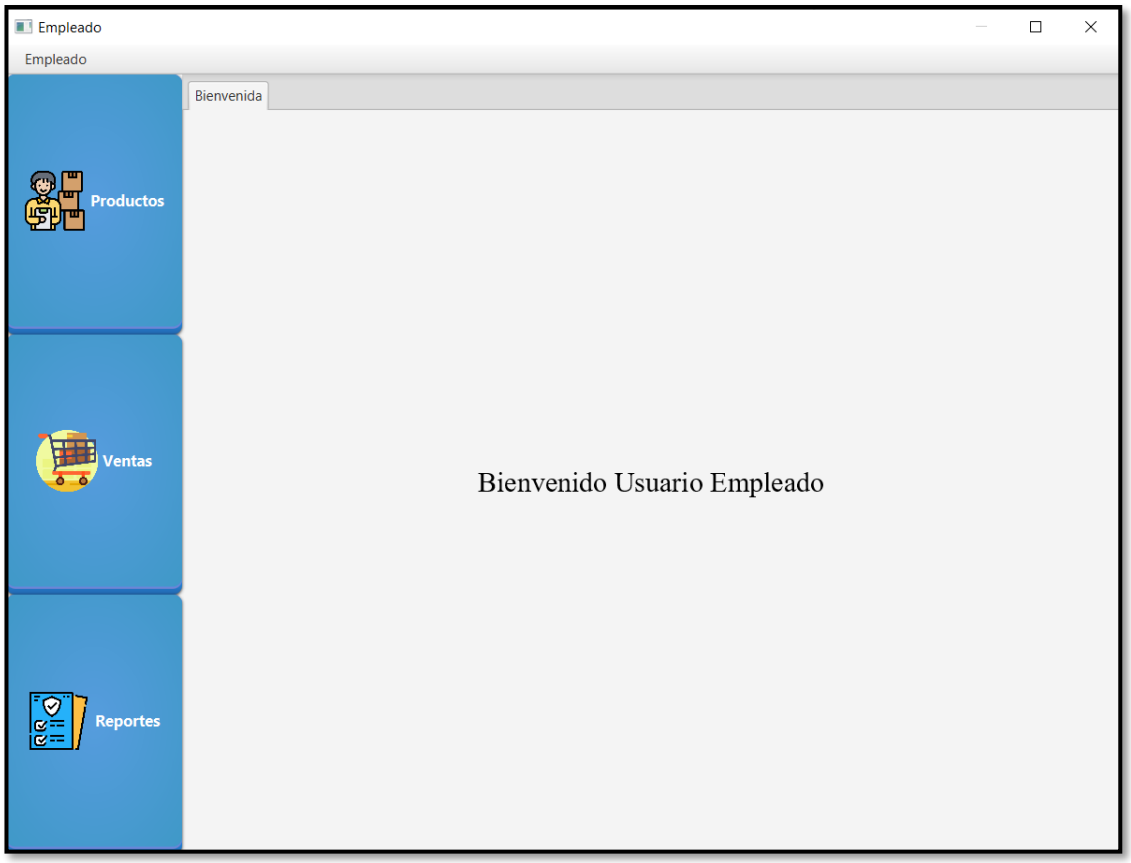

Fig. 46: Diseño de interfaz – Bienvenida panel usuario empleado.

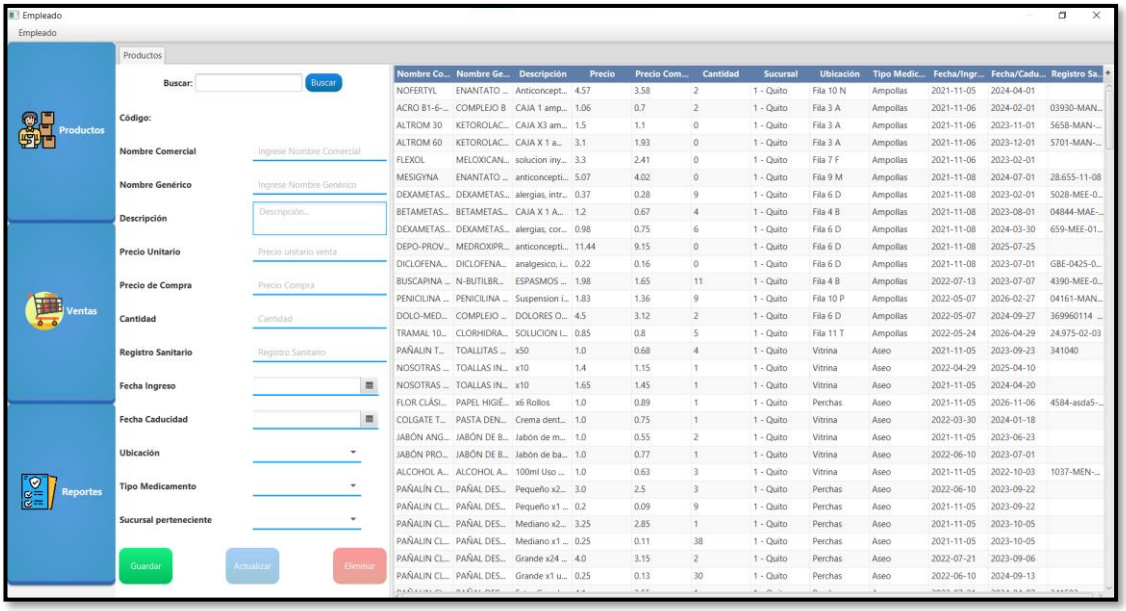

Fig. 47: Diseño de interfaz – Módulo gestionar productos usuario empleado.

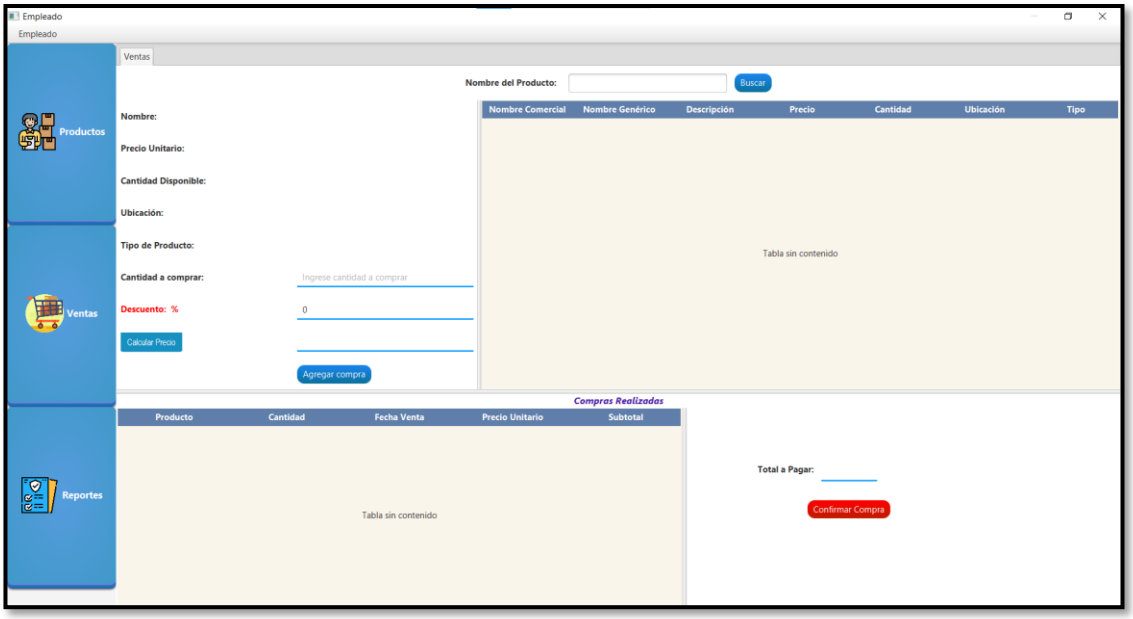

Fig. 48: Diseño de interfaz – Módulo realizar ventas usuario empleado.

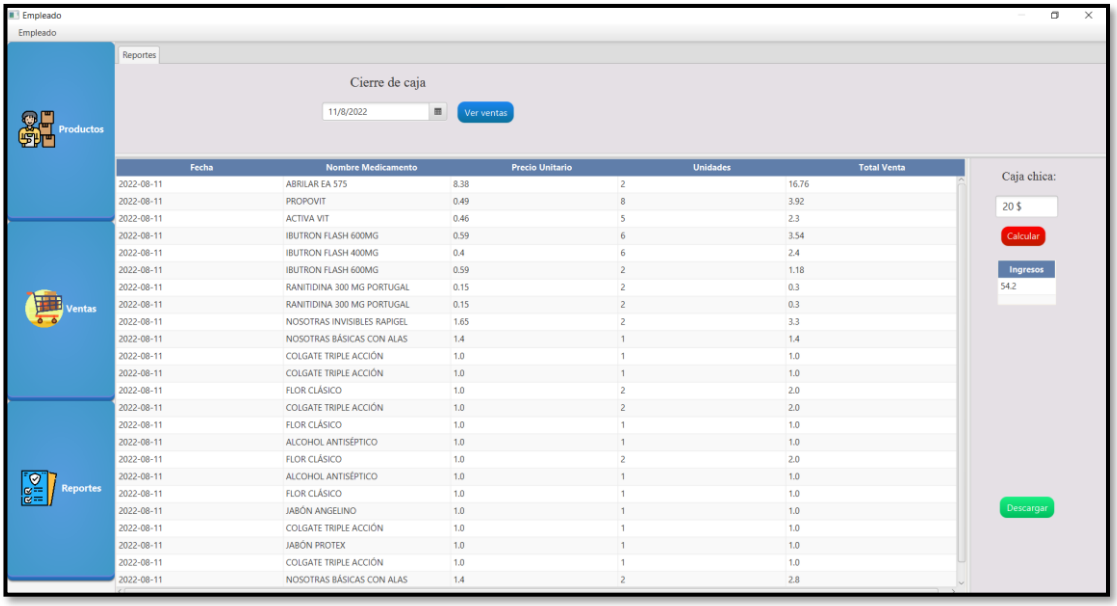

Fig. 49: Diseño de interfaz – Módulo reporte cierre de caja usuario empleado.

Diseño de la Base de datos SQL

En la **Fig. 50**, se presentan las tablas que han sido necesarias para el desarrollo de *endpoints* y el consumo de las mismas por parte del sistema de escritorio. Con esto se mantiene la toda la información organizada en tablas relacionadas para su consumo eficiente.

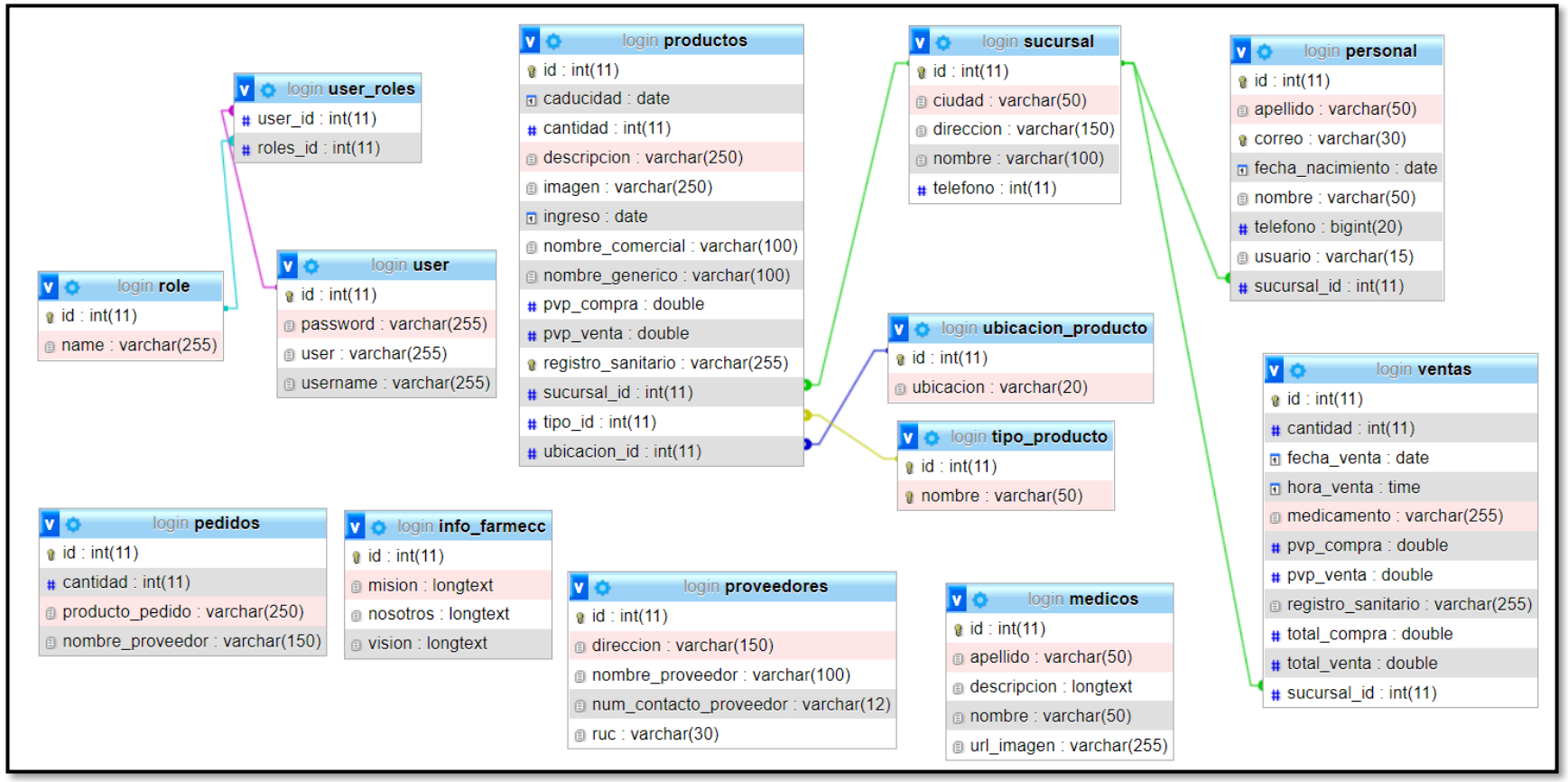

Fig. 50: Diseño de base de datos relacional SQL.

Pruebas

Una vez finalizada la etapa de codificación se han implementado pruebas unitarias, compatibilidad y de aceptación para corroborar la correcta ejecución del código ya sea en el sistema de escritorio como en los *endpoints* que se han desarrollado. Estas pruebas se han ejecutado para los módulos disponibles.

Pruebas unitarias

La figura muestra la ejecución de las pruebas realizadas al sistema de escritorio en los respectivos módulos, las pruebas se han ejecutado en los métodos CRUD para precautelar la integridad de cada dato y funcionalidades en estos procesos. Por otra parte, en los *endpoints* que se han desarrollado se emplean pruebas a las consultas realizadas en las interfaces de repositorio de cada las entidades establecidas. De **Fig. 51** a la **Fig. 70** se presentan las pruebas realizadas tanto en el sistema de escritorio como en los *endpoints* generados.

```java
@RunWith(MockitoJUnitRunner.class)
public class PersonalTest {

    @InjectMocks
    private ConexionBDD conexionBDD;
    @InjectMocks
    private Personal personal = new Personal( id: 1, nombre: "Abeas", apellido: "Corpus",
            correo: "algo@gamil.com", usuario: null, fechaNacimiento: null, telefono: 1234567, sucursales: null);

    @Test(expected = NullPointerException.class)
    public void NullPointerExceptionTest(){
        conexionBDD.inicarConexionFarm();
        System.out.println("Guardar objeto vacío");
        int resultado = personal.guardarPersona(conexionBDD.getConnection());
        conexionBDD.cerrarConexionFarm();
    }
}
```

Fig. 51: Prueba unitaria #1 Módulo personal sistema de escritorio.

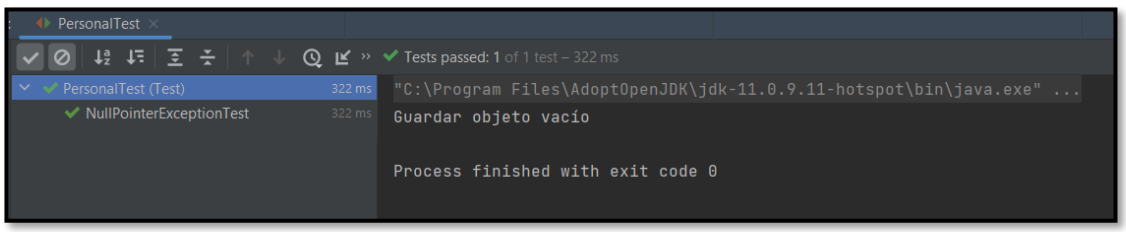

Fig. 52: Resultado de la prueba unitaria #1.

```java
@RunWith(MockitoJUnitRunner.class)
public class SucursalesTest {
    @InjectMocks
    private Sucursales sucursales = new Sucursales( id_sucursal: 16, ciudad: "Quito",
            direccion: "Oriente medio", nombre: "Pharmacy", telefono: 1234567);
    @InjectMocks
    private ConexionBDD conexionBDD;
    @Test
    public void guardarSucursal(){
        conexionBDD.inicarConexionFarm();
        System.out.println(sucursales.getId_sucursal());
        System.out.println("guardar");
        int resutado=sucursales.guardarSucursal(conexionBDD.getConnection());
        conexionBDD.cerrarConexionFarm();
        if (resutado==1){
            System.out.println("Prueba exitosa");
        }else System.out.println("Prueba Fallida");
    }
    @Test
    public void eliminarSucursal(){
        conexionBDD.inicarConexionFarm();
        System.out.println(sucursales.getId_sucursal());
        int resultado = sucursales.eliminarSucursal(conexionBDD.getConnection());
        if (resultado==1){
            System.out.println("Eliminada id " + sucursales.getId_sucursal());
        }else System.out.println("no se pudo eliminar o no existe id "+sucursales.getId_sucursal());
    }
```

Fig. 53: Prueba unitaria #2 Módulo sucursales sistema de escritorio.

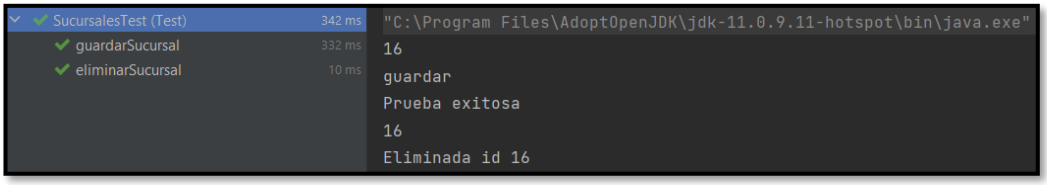

Fig. 54: Resultado prueba unitaria #2.

```java
@RunWith(MockitoJUnitRunner.class)
public class MedicosTest {

    @InjectMocks
    private ConexionBDD conexionBDD = new ConexionBDD();

    @InjectMocks
    private Medicos medicos = new Medicos( idMedicos: 1000, nombreMedicos: "Mar",
            apellidoMedicos: "Mari", descripcionMedicos: "Obse", urlImagen: "dropboc.com");
    @Test()
    public void eliminarMedico(){
        conexionBDD.inicarConexionFarm();
        System.out.println("eliminar");
        int resutado=medicos.eliminarMedico(conexionBDD.getConnection());
        conexionBDD.cerrarConexionFarm();
    }
}
```

Fig. 55: Prueba unitaria #3 Módulo médicos sistema de escritorio.

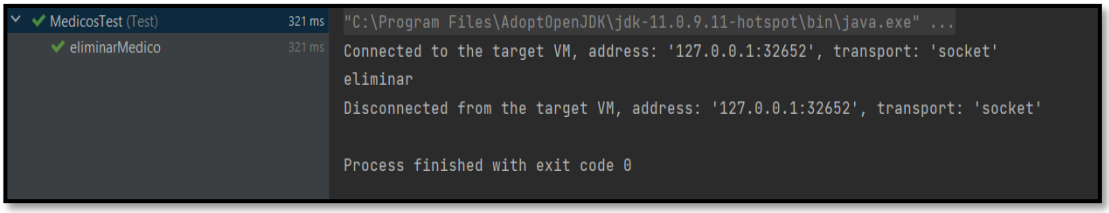

Fig. 56: Resultado prueba unitaria #3.

```java
@RunWith(MockitoJUnitRunner.class)
public class ProductosTest {
    @InjectMocks
    private ConexionBDD conexion = new ConexionBDD();

    @Mock
    private Medicamentos producto;
    @Test
    public void eliminarProductos(){
        conexion.inicarConexionFarm();
        System.out.println("eliminar");
        int resutado=producto.eliminarRegistro(conexion.getConnection());
        conexion.cerrarConexionFarm();
    }

    @Test
    public void actualizarnumProductos() {
        conexion.inicarConexionFarm();
        System.out.println("actualizar");
        int resultado = producto.actualzarNumProductos(conexion.getConnection());
        conexion.cerrarConexionFarm();
    }
```

Fig. 57: Prueba unitaria #4 Módulo productos sistema de escritorio.

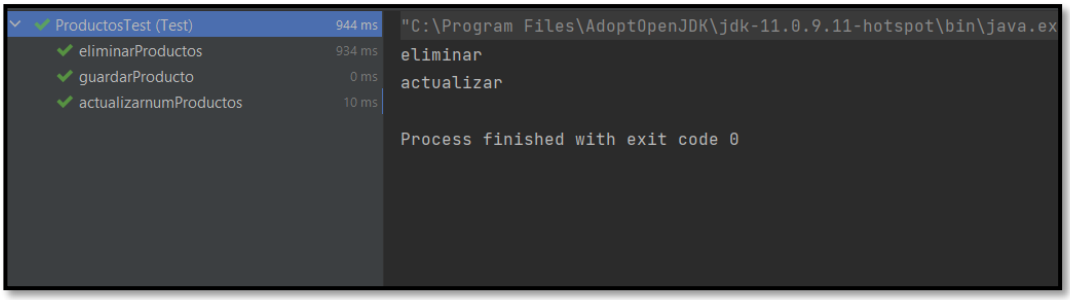

Fig. 58: Resultado prueba unitaria #4.

```java
@SpringBootTest
public class MedicosRepoTest {
    @Mock
    private MedicosRepo testMedicos;
    @InjectMocks
    private MedicosController medicosController;
    private Medicos medicos;
    @BeforeEach
    void setUp(){
        MockitoAnnotations.openMocks( testClass: this);
        medicos = new Medicos();
        medicos.setId(1);
        medicos.setNombre("Efrain");
        medicos.setApellido("Caza");
        medicos.setDescripcion("Medico");
        medicos.setUrlImagen("algo.jpg");
    }
    @Test
    void verMedicos(){
        when(testMedicos.findAll()).thenReturn(Arrays.asList(medicos));
        assertNotNull(testMedicos.findAll());
    }
```

Fig. 59: Prueba unitaria #5 Repositorio médicos *endpoints*.

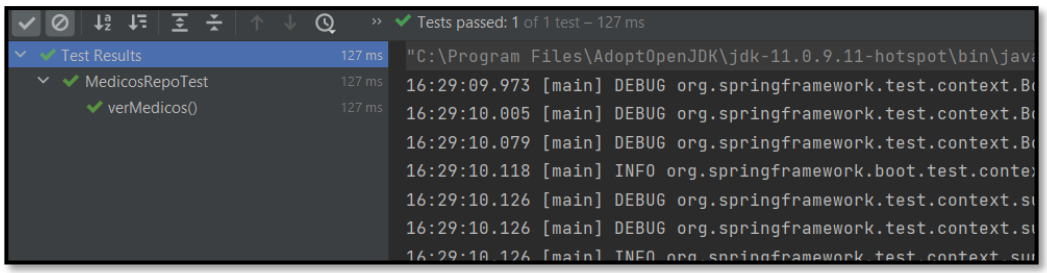

Fig. 60: Resultado prueba unitaria #5

```java
@SpringBootTest
public class PersonalRepoTest {
    @Mock
    private PersonalRepo testPersonal;
    @InjectMocks
    private PersonalController personalController;
    private Personal personal;
    @Test
    void verPersonal(){
        when(testPersonal.findAll()).thenReturn(Arrays.asList(personal));
        assertNotNull(testPersonal.findAll());
    }
}
```

Fig. 61: Prueba unitaria #6 Repositorio personal *endpoints*.

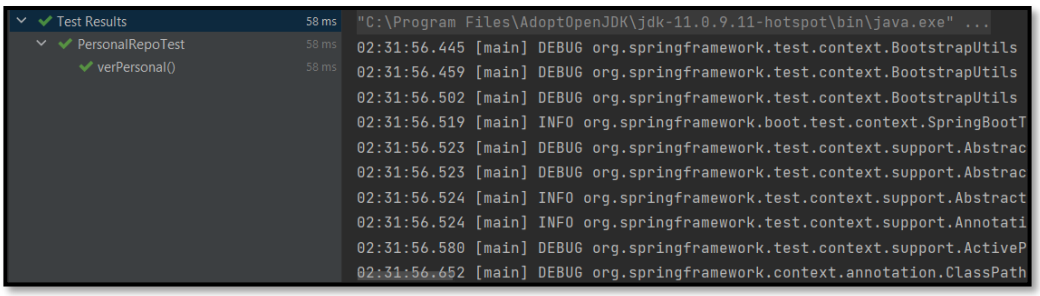

Fig. 62: Resultado prueba unitaria #6

```java
@ExtendWith(SpringExtension.class)
@SpringBootTest
@AutoConfigureMockMvc
public class ProveedoresRepoTest {
    @Autowired
    private MockMvc mockMvc;

    @Autowired
    private ObjectMapper objectMapper;

    @Test
    void registrationWorksThroughAllLayers() throws Exception {
        Proveedores proveedor = new Proveedores( id: 1, nombreProveedor: "Zaphod", numContactoProveedor: "1234",
                direccion: "algo", ruc: "1245632157");
        mockMvc.perform(post( urlTemplate: "/api/private/proveedor")
                        .contentType("application/json")
                        .header( name: "Authorization", ...values: "Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.
                        .content(objectMapper.writeValueAsString(proveedor)))
                .andExpect(status().isOk());
    }
}
```

Fig. 63: Prueba unitaria #7 Repositorio proveedores *endpoints*.

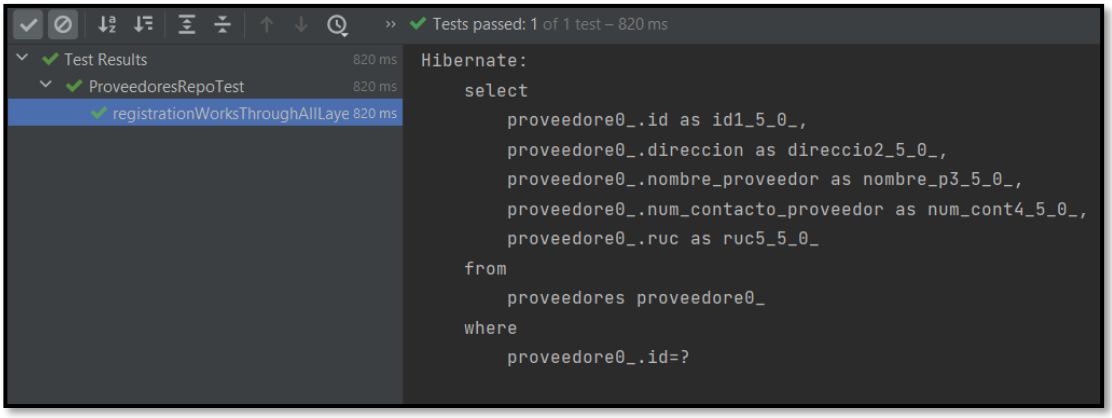

Fig. 64: Resultado prueba unitaria #7

```java
@SpringBootTest
public class SucursalRepoTest {
    @Mock
    private SucursalRepo testSucursal;
    @InjectMocks
    private Sucursales sucursal;
    @Test
    void verSucursalPorId(){
        when(testSucursal.findById(1)).thenReturn(java.util.Optional.of(sucursal));
        assertNotNull(testSucursal.findById(1));
    }
    @Test
    void verSucursalAll(){
        when(testSucursal.findAll()).thenReturn(Arrays.asList(sucursal));
        assertNotNull(testSucursal.findAll());
    }
}
```

Fig. 65: Prueba unitaria #8 Repositorio sucursal *endpoints*.

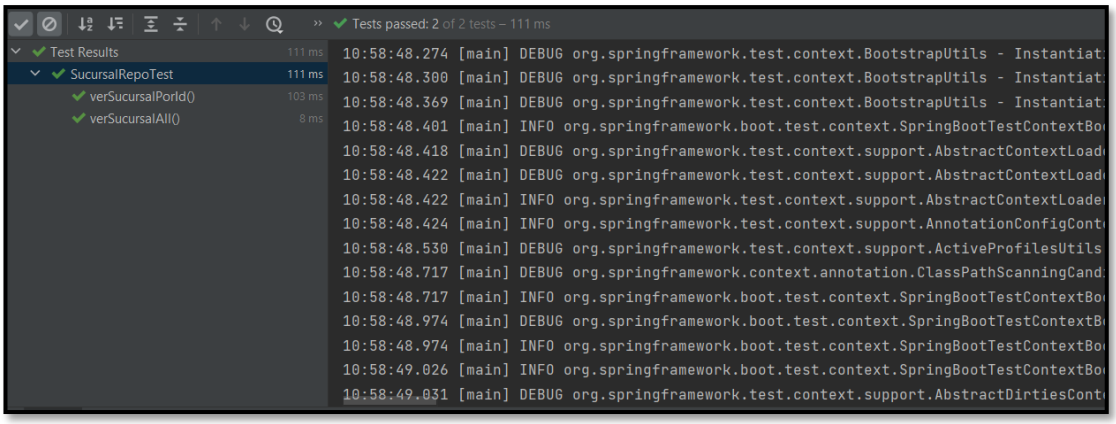

Fig. 66: Resultado prueba unitaria #8

```java
@SpringBootTest
public class ProductosRepoTest {
    @Mock
    private ProductoRepo testProductos;
    @InjectMocks
    private Productos productos;
    @Test
    void verProductoPorId(){
        when(testProductos.findById(1)).thenReturn(java.util.Optional.of(productos));
        assertNotNull(testProductos.findById(1));
    }
    @Test
    void verProductoAll(){
        when(testProductos.findAll()).thenReturn(Arrays.asList(productos));
        assertNotNull(testProductos.findAll());
    }

}
```

Fig. 67: Prueba unitaria #9 Repositorio productos *endpoints*.

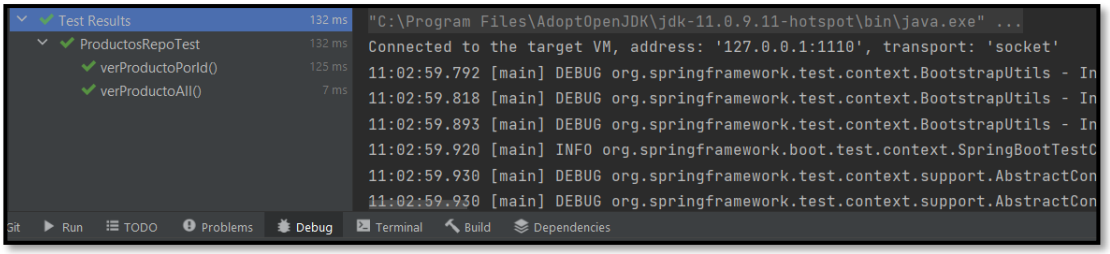

Fig. 68: Resultado prueba unitaria #9

```java
@SpringBootTest
public class VentasRepoTest {

    @Mock
    private VentasRepo testVentas;
    @InjectMocks
    private Ventas ventas;
    @Test
    void verVentasPorSucursalId(){
        when(testVentas.findBySucursalId(1)).thenReturn(Arrays.asList(ventas));
        assertNotNull(testVentas.findBySucursalId(1));
    }
    @Test
    void verVentasPorFiltro(){
        when(testVentas.getPVentaFilter("2022-08-11","2022-08-22")).thenReturn(Arrays.asList(ventas));
        assertNotNull(testVentas.getPVentaFilter("2022-08-11","2022-08-22"));
    }
}
```

Fig. 69: Prueba unitaria #10 Repositorio ventas *endpoints*.

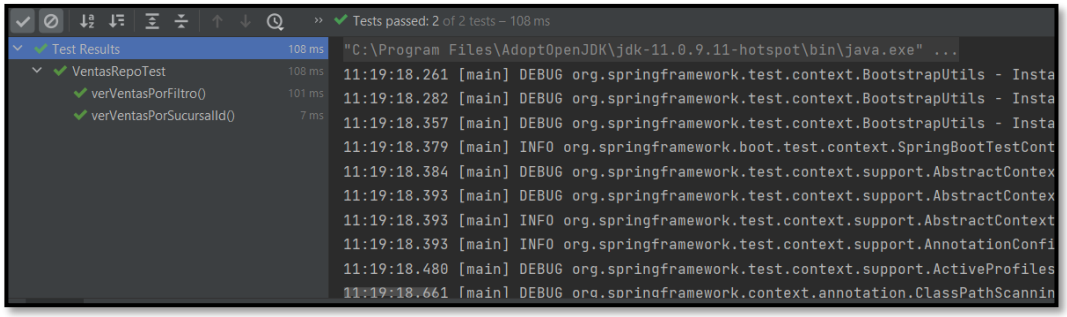

Fig. 70: Resultado prueba unitaria #10

Pruebas de compatibilidad

Para la presente sección se verifica la correcta funcionalidad del sistema de escritorio en las distintas versiones del sistema operativo con el fin de detectar posibles fallas y si existen cambios en la presentación del sistema de escritorio. Además, se presenta la funcionalidad del consumo de los *endpoints* desarrollados mediante un cliente HTTP como lo es *Talent API Tester,* verificando su correcto funcionamiento. A continuación, desde la **Fig. 71** hasta la **Fig. 81** se presentan los resultados obtenidos tanto del sistema de escritorio como de los *endpoints* en sus diferentes entornos de ejecución.

Fig. 71: Módulo iniciar sesión vista desde *Windows* 11.

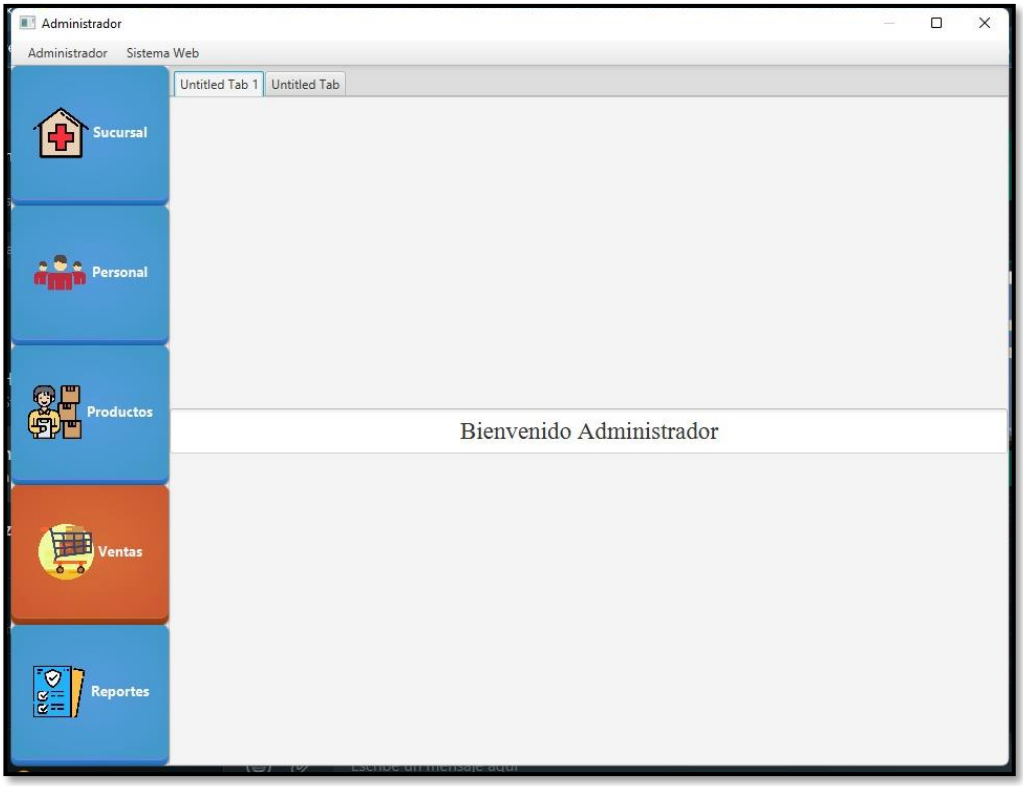

Fig. 72: Bienvenida usuario administrador vista desde *Windows* 11.

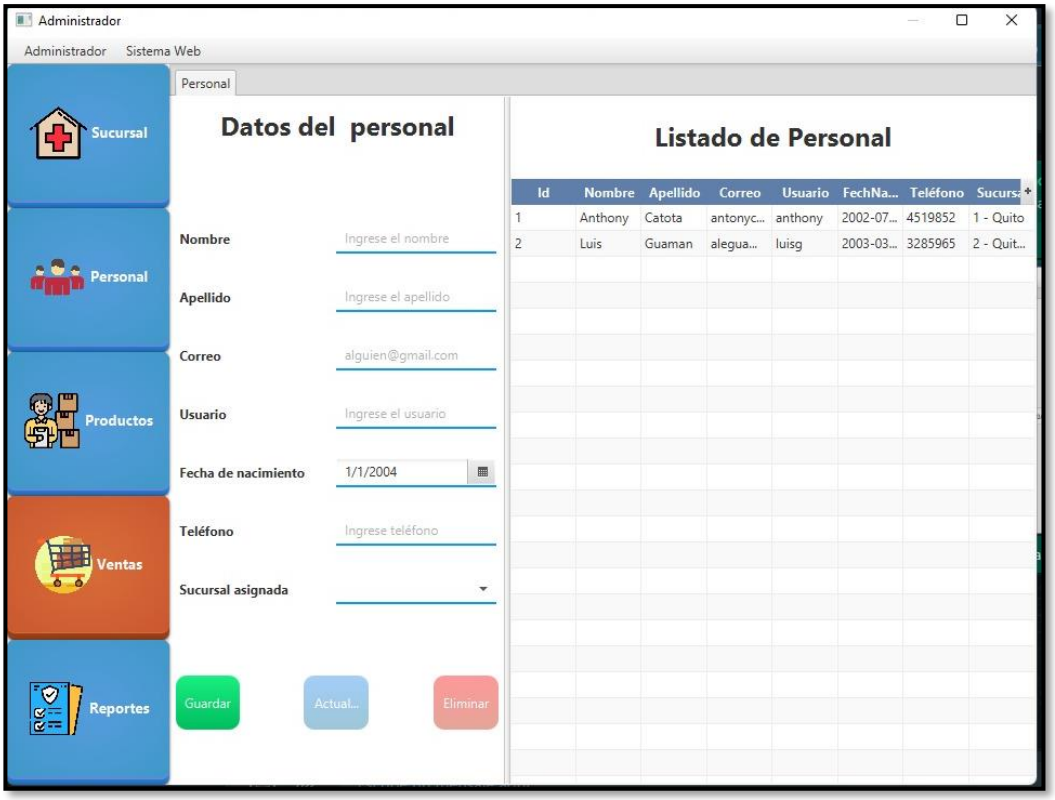

Fig. 73: Módulo personal vista desde *Windows* 11.

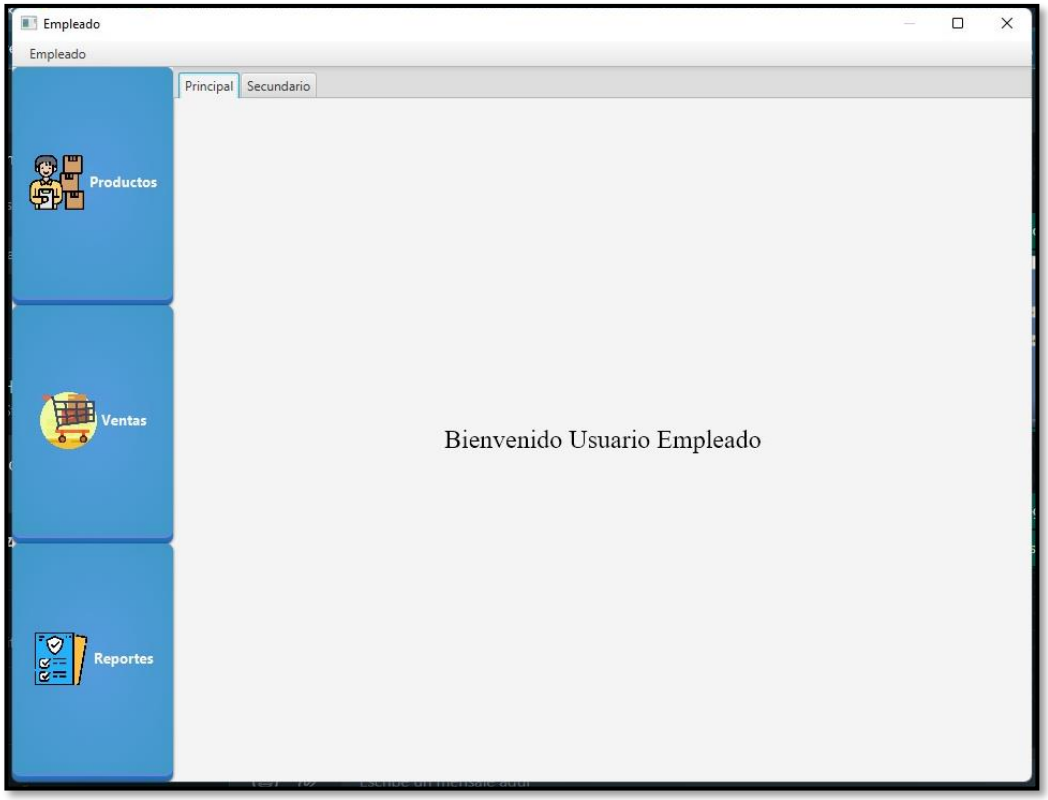

Fig. 74: Bienvenida usuario empleado vista desde *Windows* 11.

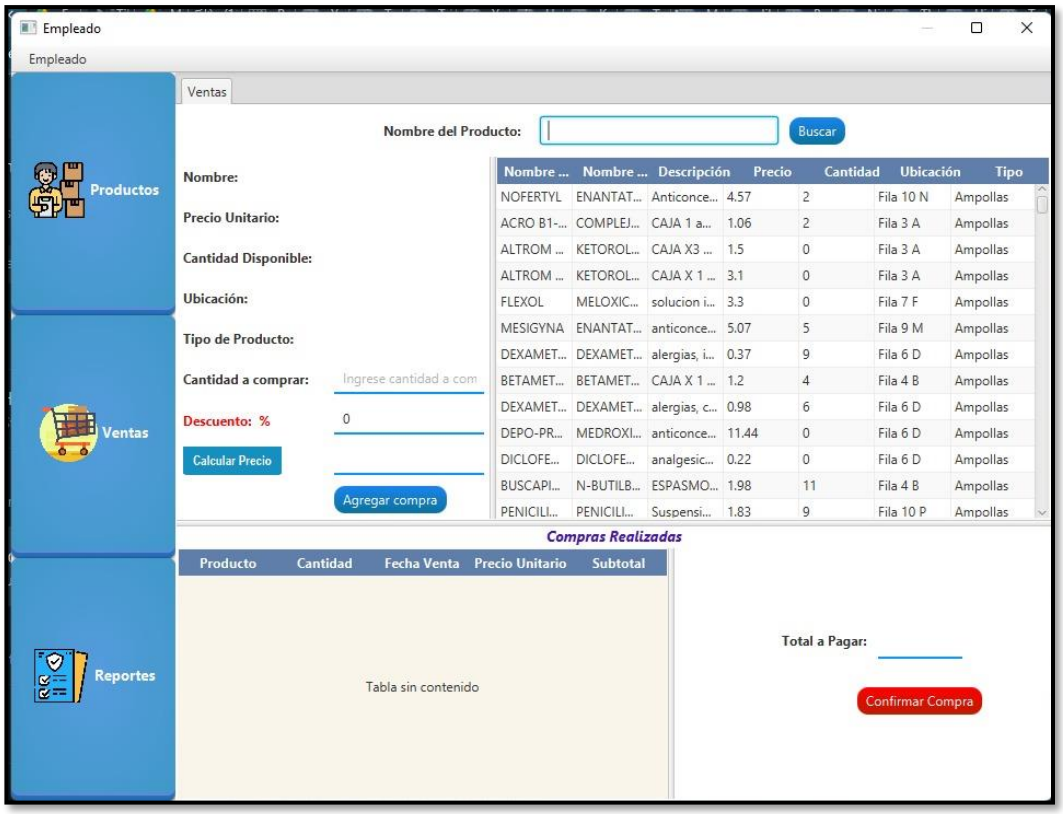

Fig. 75: Módulo ventas empleado vista desde *Windows* 11.

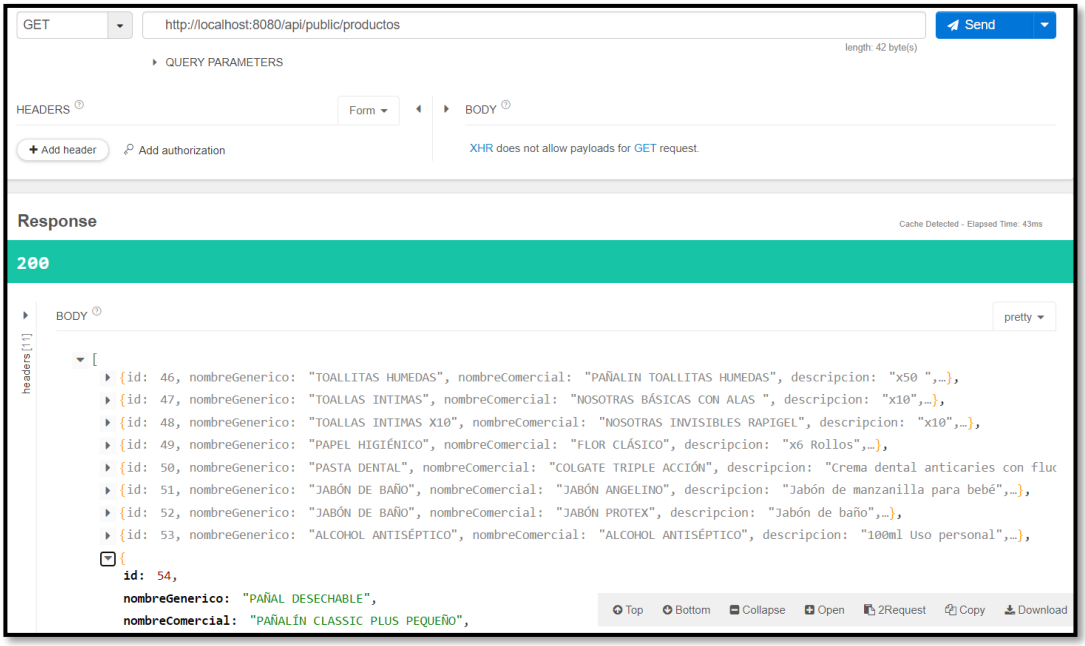

Fig. 76: Método *GET* productos en *Talend API Tester.*

Fig. 77: Método *GET* información general en *Talend API Tester*.

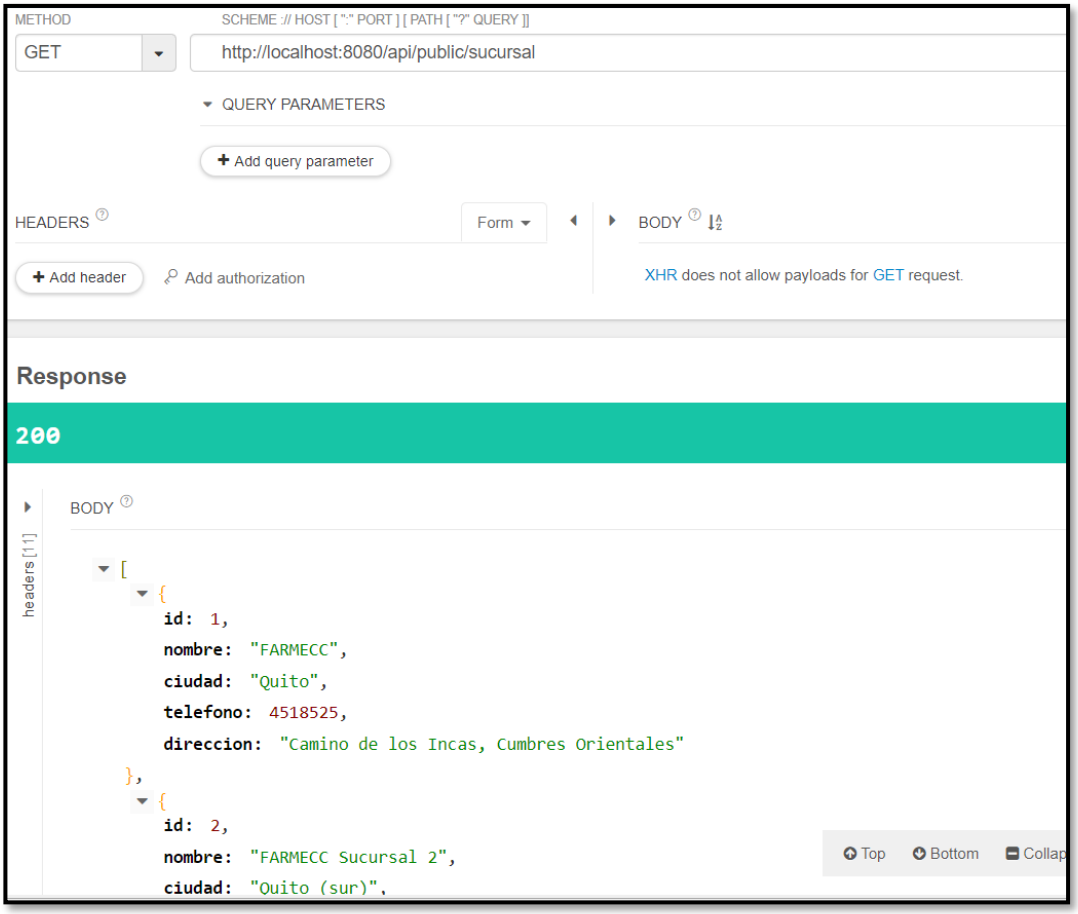

Fig. 78: Método *GET* sucursales en *Talend API Tester*.

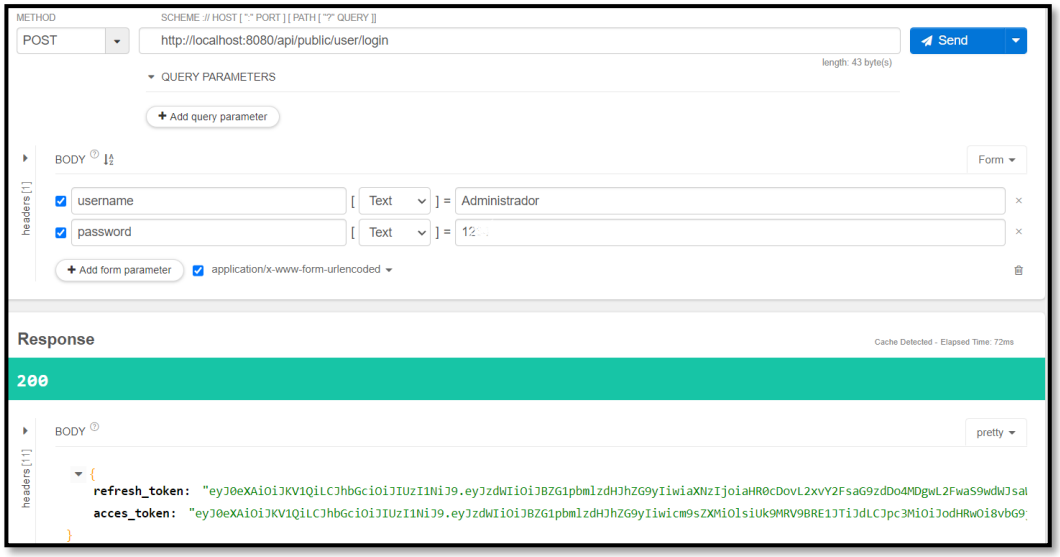

Fig. 79: Método *POST* inicio de sesión en *Talend API Tester*.

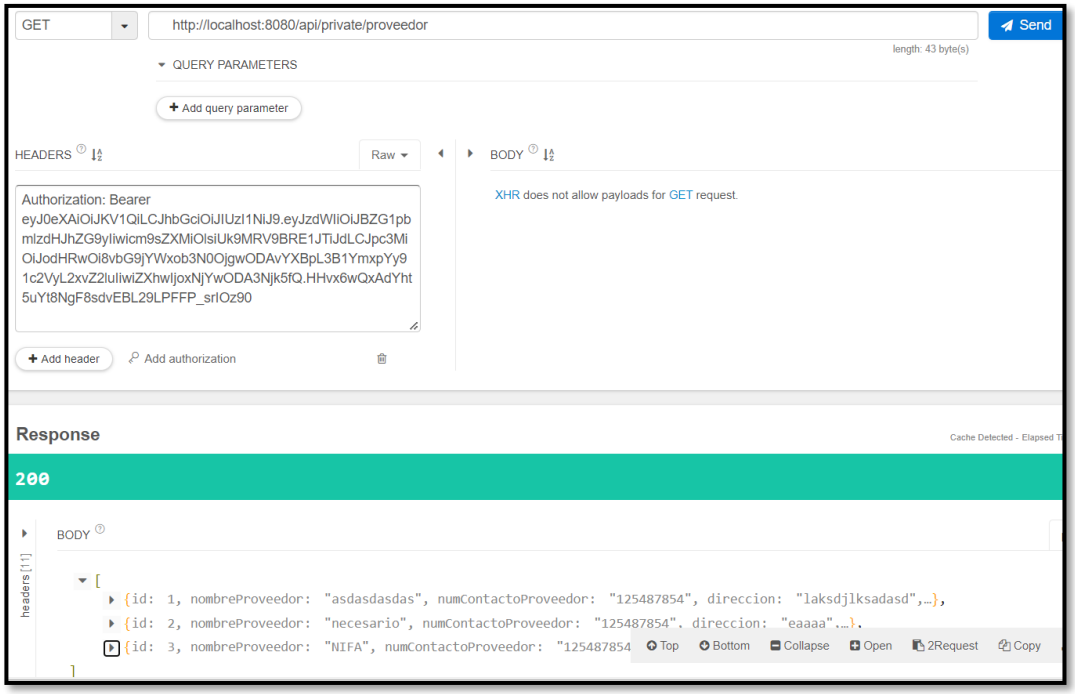

Fig. 80: Método *GET* proveedores en *Talend API Tester*.

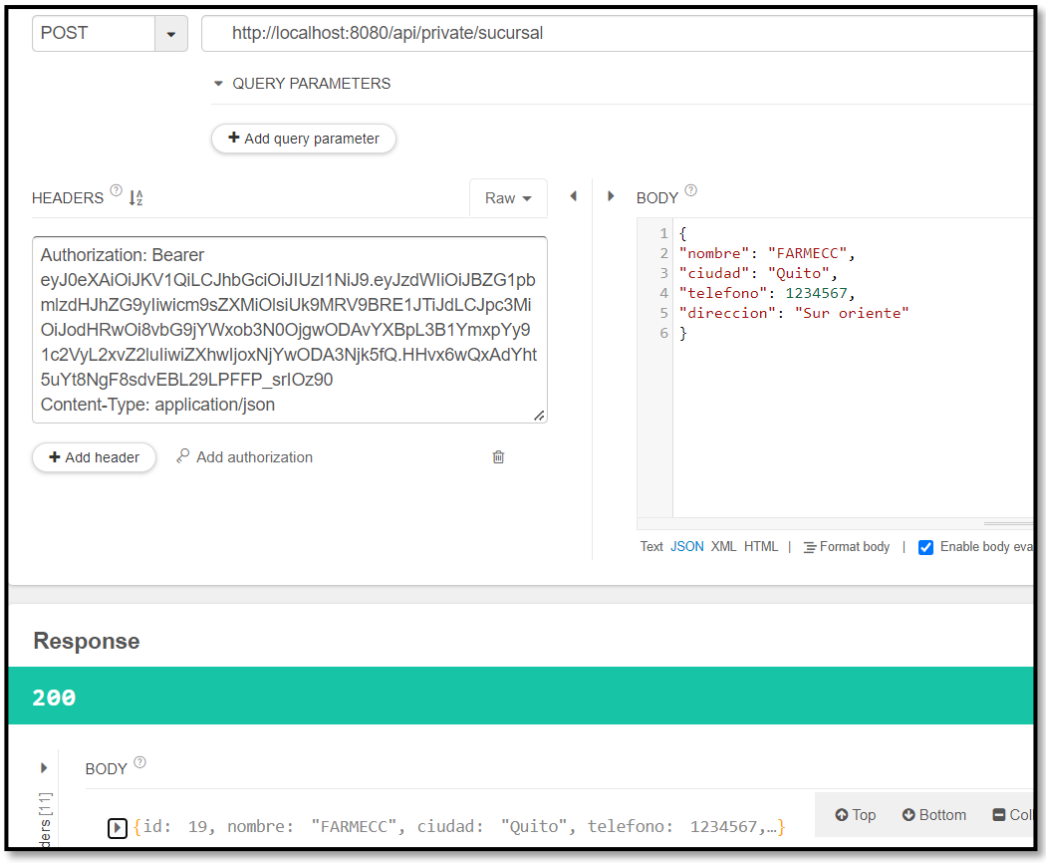

Fig. 81: Método *POST* sucursales en *Talend API Tester*.

Prueba de aceptación

Se procede a mostrar las 18 pruebas de aceptación las cuales van desde la **TABLA XXXI** hasta la **TABLA XLVII**. Cabe mencionar que cada una de las pruebas detallan el proceso correspondiente a las tareas designadas para cada tipo de usuario, teniendo así una correcta ejecución y aprobación de la misma.

TABLA XXXI: Prueba de aceptación N°1 – Inicio y cierre sesión.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA001	**Identificador historia de Usuario:** HU001
Nombre: Inicio y cierre de sesión	
Descripción: El usuario administrador y empleado tiene la posibilidad de iniciar sesión para poder interactuar con los módulos correspondientes de acuerdo a su rol. Por otra parte, los usuarios tienen la posibilidad de dejar de interactuar con el sistema mediante el cierre de sesión.	
Pasos de ejecución: • Abrir el ejecutable del sistema de escritorio. • Colocar credenciales de usuario (nombre de usuario y contraseña) • Seleccionar el botón “Iniciar Sesión”. • Verificar la redirección a los módulos correspondientes a su tipo de usuario.	
Resultado deseado: El sistema de escritorio acepta las credenciales de usuario y permite la redirección a sus respectivos módulos.	
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.	

TABLA XXXII: Prueba de aceptación N°3 – Gestionar sucursales.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA003	**Identificador historia de Usuario:** HU003
Nombre: Gestionar sucursales.	

Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a: • Sucursales. Para el registro y modificación de una sucursal se lo realiza mediante un formulario donde se ingresa la siguiente información: • Nombre. • Ciudad. • Dirección. • Teléfono.
Pasos de ejecución: • Abrir el ejecutable del sistema de escritorio. • Colocar credenciales del sistema de usuario administrador. • Seleccionar “Personal” en el panel lateral. • Verificar los métodos de visualizar, modificar, agregar y eliminar.
Resultado deseado: El sistema de escritorio permite realizar los métodos CRUD (crear, visualizar, actualizar y eliminar) en el módulo sucursales.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XXXIII: Prueba de aceptación N°4 – Gestionar productos.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA004	**Identificador historia de Usuario:** HU004
Nombre: Gestionar productos.	
Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a: • Productos Por otra parte, el usuario empleado tiene la posibilidad de crear, visualizar y modificar esta misma información, pero no puede eliminar. Para el registro y modificación de un producto se lo realiza mediante un formulario donde se ingresa la siguiente información: • Nombre comercial.	

• Nombre genérico. • Descripción del producto. • Precio de compra. • Precio unitario de venta al público. • Cantidad. • Registro sanitario. • Fecha de ingreso. • Fecha de caducidad. • Ubicación. • Tipo de producto. • Sucursal perteneciente
Pasos de ejecución: • Abrir el ejecutable del sistema de escritorio. • Colocar credenciales del sistema ya sea de usuario administrador o empleado. • Seleccionar “Productos” en el panel lateral. • Verificar los métodos de visualizar, modificar, agregar y eliminar para el usuario administrador. • Verificar que el método eliminar en el usuario empleado se encuentre deshabilitado.
Resultado deseado: El sistema de escritorio permite realizar los métodos CRUD (crear, visualizar, actualizar y eliminar) en el módulo productos para el usuario administrador, mientras que para el usuario empleado el método eliminar se encuentre deshabilitado.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XXXIV: Prueba de aceptación N°5 – Gestionar ventas.

	PRUEBA DE ACEPTACIÓN
Identificador (ID): PA005	**Identificador historia de Usuario:** HU005
Nombre: Gestionar ventas.	

Descripción: El usuario administrador tiene la posibilidad de crear, visualizar y eliminar información respecto a:

- Ventas

Por otra parte, el usuario empleado tiene la posibilidad de crear y visualizar esta misma información, pero no puede eliminar.

Para el registro de ventas se lo realiza mediante la búsqueda y selección de un producto colocando la cantidad requerida y además un descuento en caso de ser necesario. Para esto el usuario que inicie sesión debe llenar un formulario con la siguiente información:

- Cantidad requerida.
- Descuento.

Una vez realizada la venta esta es almacenada y se realiza el descuento automático del *stock* del producto.

Pasos de ejecución:

- Abrir el ejecutable del sistema de escritorio.
- Colocar credenciales del sistema ya sea de usuario administrador o empleado.
- Seleccionar “Ventas” en el panel lateral.
- Verificar los métodos de visualizar, agregar y eliminar para el usuario administrador.
- Verificar que el método eliminar en el usuario empleado se encuentre deshabilitado.

Resultado deseado:

El sistema de escritorio permite realizar los métodos crear, visualizar y eliminar en el módulo ventas para el usuario administrador, mientras que para el usuario empleado el método eliminar se encuentre deshabilitado.

Evaluación de la prueba:

El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XXXV: Prueba de aceptación N°6 – Generar reportes.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA006	**Identificador historia de Usuario:** HU006
Nombre: Generar reportes.	
Descripción: El usuario administrador puede generar reportes de las ventas realizadas en cada sucursal, reportes de ventas globales y reportes de ventas destinados al Ministerio de Salud Pública del país. Por otra parte, el usuario empleado únicamente puede generar reportes de ventas diario.	
Pasos de ejecución: • Abrir el ejecutable del sistema de escritorio. • Colocar credenciales del sistema ya sea de usuario administrador o empleado. • Seleccionar “Reportes” en el panel lateral. • Seleccionar fecha o intervalo de fechas y verificar la correcta obtención de información de las tablas presionando el botón “ver ventas” de cada panel. • Presionar los botones habilitados para generar reportes y verificar su correcta descarga dentro del escritorio. • Ejecutar el archivo y verificar la escritura de información y los formatos correspondientes.	
Resultado deseado: El sistema de escritorio permite la obtención de información correspondiente a las ventas y permite generar archivos tipo PDF con información de ventas realizadas como: cierre de caja, reporte entre fechas y reporte destinado al Ministerio de Salud Pública del país.	
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.	

TABLA XXXVI: Prueba de aceptación N°7 – Gestión de médicos.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA007	**Identificador historia de Usuario:** HU007
Nombre: Gestión de médicos.	

Descripción: El usuario administrador tiene la posibilidad de crear, visualizar, modificar y eliminar información respecto a: • Médicos. Para el registro y modificación de un médico se lo realiza mediante un formulario donde se ingresa la siguiente información: • Nombre. • Apellido. • Descripción. • Imagen.
Pasos de ejecución: • Abrir el ejecutable del sistema de escritorio. • Colocar credenciales del sistema como usuario administrador. • Seleccionar “Sistema Web” en el menú superior y posteriormente seleccionar la opción “Médicos”. • Verificar los métodos visualizar, modificar, agregar y eliminar.
Resultado deseado: El sistema de escritorio permite realizar los métodos CRUD (crear, visualizar, actualizar y eliminar) en el módulo médicos.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XXXVII: Prueba de aceptación N°8 – Gestión de información general.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA008	**Identificador historia de Usuario:** HU008
Nombre: Gestión de información general.	
Descripción: El usuario administrador tiene la posibilidad de visualizar y modificar información respecto a: • Información general. Para el registro y modificación de un médico se lo realiza mediante un formulario donde se ingresa la siguiente información: • Misión. • Visión.	

• Nosotros.
Pasos de ejecución: • Abrir el ejecutable del sistema de escritorio. • Colocar credenciales del sistema como usuario administrador. • Seleccionar “Sistema Web” en el menú superior y posteriormente seleccionar la opción “Información FARMECC”. • Verificar los métodos visualizar y modificar.
Resultado deseado: El sistema de escritorio permite realizar los métodos visualizar y modificar en el módulo información general.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XXXVIII: Prueba de aceptación N°9 – Generar *endpoints* para visualizar página informativa.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA009	**Identificador historia de Usuario:** HU009
Nombre: Gestión de información general.	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite visualizar toda la información necesaria para presentar una página informativa, esta información concierne a: • Misión, visión y nosotros. • Médicos. • Sucursales.	
Pasos de ejecución: • Acceder a las rutas generadas por el sistema, las cuales son públicas. • Llamar a los métodos *GET* de información general, médicos y sucursales. • El sistema genera respuestas tipo *JSON* con información obtenida de la Base de datos.	

Resultado deseado: El sistema permite al usuario generar *endpoints* con información para la visualización de una página informativa de la cadena de farmacias FARMECC.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XXXIX: Prueba de aceptación N°10 – Generar *endpoints* para inicio y cierre de sesión.

	PRUEBA DE ACEPTACIÓN
Identificador (ID): PA010	**Identificador historia de Usuario:** HU010
Nombre: Generar *endpoints* para inicio y cierre de sesión.	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite: • Iniciar Sesión. • Cerrar Sesión	
Pasos de ejecución: • Acceder a las rutas generadas por el sistema, las cuales son públicas. • Llamar al método *POST* de inicio de sesión y enviar las credenciales de acceso. • El sistema genera una respuesta tipo *JSON* con los *tokens* de acceso para *endpoints* privados del sistema.	
Resultado deseado: El sistema permite al usuario generar *endpoints* para inicio y cierre de sesión, además, proporciona un *token* de acceso.	
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.	

TABLA XL: Prueba de aceptación N°11 – Generar *endpoints* para gestionar sucursales.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA011	**Identificador historia de Usuario:** HU011
Nombre: Generar *endpoints* para gestionar sucursales.	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que da la posibilidad de gestionar las sucursales, esto permite: • Crear sucursales. • Modificar sucursales. • Visualizar sucursales. • Eliminar sucursales.	
Pasos de ejecución: • Acceder a las rutas generadas por el sistema, las cuales son privadas. • Llamar al método de inicio de sesión de tipo *POST* para autenticarse y obtener un *token* de acceso. • Llamar a los métodos *GET, PUT, POST y DELETE* para la gestión de información de sucursales. • El sistema genera respuestas tipo *JSON* con información obtenida de la Base de datos.	
Resultado deseado: El sistema permite al usuario administrador generar *endpoints* para la gestión de información relacionada a sucursales, las cuales permiten el uso de métodos CRUD (crear, visualizar, actualizar y eliminar).	
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.	

TABLA XLI: Prueba de aceptación N°12 – Generar *endpoints* para visualizar productos.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA012	**Identificador historia de Usuario:** HU012
Nombre: Generar *endpoint* para visualizar productos.	
Descripción: El sistema a través del usuario asignado permite generar un *endpoint* que permite visualizar toda la información de los productos disponibles en cada sucursal.	
Pasos de ejecución: • Acceder a la ruta generada por el sistema, la cual es pública. • Llamar al método *GET* para obtener toda la información relacionada a los productos. • El sistema genera una respuesta de tipo *JSON* con información obtenida de la Base de datos.	
Resultado deseado: El sistema permite al usuario generar un *endpoint* para la visualización de productos.	
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.	

TABLA XLII: Prueba de aceptación N°13 – Generar *endpoints* para visualizar reportes.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA013	**Identificador historia de Usuario:** HU013
Nombre: Generar *endpoints* para visualizar reportes.	
Descripción: El sistema a través del usuario administrador puede generar varios *endpoints* que permiten visualizar reportes como: • Reportes generales de cada sucursal. • Reportes globales de ventas. • Reportes entre fechas específicas.	

Pasos de ejecución: • Acceder a las rutas generadas por el sistema, las cuales son privadas. • Llamar al método de inicio de sesión de tipo *POST* para autenticarse y obtener un *token* de acceso. • Llamar a los métodos *GET* para obtener reportes de ventas. • El sistema genera respuestas en documentos con formato *PDF* con información sobre ventas obtenida de la Base de datos.
Resultado deseado: El sistema permite al usuario administrador generar varios *endpoints* para obtener reportes de ventas.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XLIII: Prueba de aceptación N°14 – Generar *endpoint* para visualizar disponibilidad de *stock*.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA014	**Identificador historia de Usuario:** HU014
Nombre: Generar *endpoint* para visualizar disponibilidad de *stock*.	
Descripción: El sistema a través del usuario administrador puede genera un *endpoint* que permiten visualizar la disponibilidad de *stock* de cada producto.	
Pasos de ejecución: • Acceder a la ruta generada por el sistema, la cual es privada. • Llamar al método de inicio de sesión de tipo *POST* para autenticarse y obtener un *token* de acceso. • Llamar al método *GET* para obtener información sobre el *stock* de productos. • El sistema genera una respuesta de tipo *JSON* con información obtenida de la Base de datos.	
Resultado deseado: El sistema permite al usuario administrador genera un *endpoint* para visualizar información respecto al *stock* de productos.	

Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XLIV: Prueba de aceptación N°15 – Generar *endpoints* para realizar cotizaciones.

	PRUEBA DE ACEPTACIÓN
Identificador (ID): PA015	**Identificador historia de Usuario:** HU015
Nombre: Generar *endpoints* para realizar cotizaciones.	
Descripción: El sistema a través del usuario cliente puede generar varios *endpoints* que permiten realizar cotizaciones, con la información general de los productos.	
Pasos de ejecución: • Acceder a las rutas generadas por el sistema, las cuales son públicas. • Llamar al método *GET* para obtener información para cotización de productos. • El sistema genera respuestas de tipo *JSON* con información obtenida de la Base de datos.	
Resultado deseado: El sistema permite al usuario cliente generar *endpoints* para visualizar información respecto al cotizaciones.	
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.	

TABLA XLV: Prueba de aceptación N°16 – Generar *endpoints* para visualizar ventas por sucursales.

	PRUEBA DE ACEPTACIÓN
Identificador (ID): PA016	**Identificador historia de Usuario:** HU016
Nombre: Generar *endpoints* para visualizar ventas por sucursales.	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite visualizar las ventas realizadas por sucursales.	

Pasos de ejecución: • Acceder a las rutas generadas por el sistema, las cuales son privadas. • Llamar al método de inicio de sesión de tipo *POST* para autenticarse y obtener un *token* de acceso. • Llamar a los métodos *GET* para obtener información sobre las ventas realizadas en cada sucursal o ventas globales. • El sistema genera respuestas de tipo *JSON* con información obtenida de la Base de datos.
Resultado deseado: El sistema permite al usuario administrador generar varios *endpoints* para visualizar información respecto a ventas por sucursales.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XLVI: Prueba de aceptación N°17 – Generar *endpoint* para visualizar personal.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA017	**Identificador historia de Usuario:** HU017
Nombre: Generar *endpoint* para visualizar personal.	
Descripción: El sistema a través del usuario administrador permite generar un *endpoint* que permite visualizar toda la información acerca del personal.	
Pasos de ejecución: • Acceder a la ruta generada por el sistema, la cual es privada. • Llamar al método de inicio de sesión de tipo *POST* para autenticarse y obtener un *token* de acceso. • Llamar al método *GET* para obtener información sobre el personal. • El sistema genera una respuesta de tipo *JSON* con información obtenida de la Base de datos.	

Resultado deseado: El sistema permite al usuario administrador generar un *endpoint* para visualizar información respecto al personal.
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.

TABLA XLVII: Prueba de aceptación N°18 – Generar *endpoints* para gestionar pedidos con proveedores.

PRUEBA DE ACEPTACIÓN	
Identificador (ID): PA018	**Identificador historia de Usuario:** HU018
Nombre: Generar *endpoints* para gestionar pedidos con proveedores.	
Descripción: El sistema a través del usuario administrador permite generar varios *endpoints* que permite la gestión de proveedores, esto permite: • Crear proveedor. • Modificar proveedor. • Visualizar proveedor. • Eliminar proveedor. De la misma manera permite la gestión de pedidos con los mismos.	
Pasos de ejecución: • Acceder a las rutas generadas por el sistema, las cuales son privadas. • Llamar al método de inicio de sesión de tipo *POST* para autenticarse y obtener un *token* de acceso. • Llamar a los métodos *GET, PUT, POST y DELETE* para la gestión de proveedores. • El sistema genera respuestas tipo *JSON* con información obtenida de la Base de datos.	
Resultado deseado: El sistema permite al usuario administrador generar varios *endpoints* para gestionar información relacionada a proveedores.	
Evaluación de la prueba: El cliente aprueba al 100% con la verificación de los resultados esperados.	

Certificado de la cadena de farmacias FARMECC

FARMACIAS FARMECC.

Quito, 20 de agosto del 2022

CERTIFICADO

Yo, Efrain Caza Chiluiza, con CI. 1722656194, como gerente general de FARMACIAS FARMECC ubicada en Quito – Ecuador.

Por medio de la presente certifico:

Que el Sr. **Bryan Armando Quisaguano Casa** con cédula de ciudadanía **1725175648**, estudiante de la carrera de Desarrollo de Software, realizó su trabajo de integración curricular de desarrollo de un sistema para la gestión del inventario en FARMECC (sistema de escritorio), mismo que cumple con los requerimientos y funcionalidades definidas en las reuniones mantenidas.

Es todo en cuanto puedo mencionar en honor a la verdad pudiendo el interesado hacer uso de este documento como estime conveniente.

Atentamente:

EFRAIN CAZA CHILUIZA
LICENCIADO RADIÓLOGO
0967676843

ANEXO III

El siguiente enlace, permite observar el Manual de Usuario, donde se detalla la información con respecto a las funcionalidades del sistema de escritorio, así como la participación de los perfiles en la interacción.

https://youtu.be/C7QawKLBAb4

ANEXO IV

A continuación, se especifica las credenciales de acceso para los *endpoints* privados, así como el enlace al repositorio en *GitHub* en donde se encuentra el código fuente y en el apartado de README los pasos para realizar la instalación de forma local.

Credenciales de acceso para sistema de escritorio y *endpoints*

Para acceder a los *endpoints* desplegados en producción, ingresar a la siguiente URL:

https://farmecc.herokuapp.com/doc/swagger-ui/index.html

Repositorio del código fuente del sistema de escritorio y *endpoints*

El código fuente de todo el proyecto, se encuentra alojado en el repositorio *GitHub* separado cada uno en ramas, las cuales se pueden acceder a través de la siguiente URL:

https://github.com/BryanArmando/SisEscritorio_FARMECC.git