

ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

DESARROLLO DE UN E-COMMERCE PARA LA VENTA DE PRODUCTOS ARTESANALES PERSONALIZADOS BASADO EN IA

DESARROLLO DE UN BACKEND

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

ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA

DIRECTOR: BYRON GUSTAVO LOARTE CAJAMARCA

DMQ, julio 2025

CERTIFICACIONES

Yo, **ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA** 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.

ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA

estefania.sanchez01@epn.edu.ec

estefi2000ms2@gmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por **ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA**, 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.

ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA

DEDICATORIA

Dedico este trabajo con todo mi cariño a mis padres, por ser mi mayor ejemplo de esfuerzo, amor y perseverancia. A mis hermanos, por su apoyo incondicional y por estar siempre presentes en cada etapa de mi vida. A mis amigos más cercanos, por sus palabras de aliento, su compañía y por hacer de este camino una experiencia más llevadera y significativa.

Y, con especial cariño, a esa persona que ha estado a mi lado en los momentos más importantes, brindándome su apoyo, comprensión y sabiduría. Gracias por creer en mí incluso cuando yo dudaba, y por impulsarme siempre a dar lo mejor de mí.

ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA

AGRADECIMIENTO

Este trabajo representa el esfuerzo y sacrificio de mis padres, quienes han sido un pilar fundamental en mi vida. Sin su amor incondicional, dedicación y apoyo constante, no habría sido posible alcanzar esta meta. Agradezco profundamente a mis hermanos, cuyas palabras de aliento y compañía me motivaron a dar lo mejor de mí cada día.

También deseo expresar mi sincero agradecimiento a mi persona favorita. Su amor, paciencia y conocimiento fueron claves en este proceso. Gracias por brindarme su tiempo, por estar a mi lado y ayudarme a comprender mejor cada etapa del camino.

Agradezco a Dios por darme la fuerza, la salud, la sabiduría y las ganas de seguir adelante, incluso en los momentos más difíciles. Cuando sentía que ya no podía continuar, siempre encontraba en él el impulso para no rendirme.

Finalmente, agradezco de todo corazón a mi tutor de tesis el Ing. Byron Loarte ya que su guía, compromiso y confianza en mi trabajo fueron esenciales para llevar a cabo este proyecto. Además, ha sido un verdadero mentor, cuya motivación diaria me impulsó a crecer y superarme constantemente.

ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA

ÍNDICE DE CONTENIDO

CERTIFICACIONES.....	I
DECLARACIÓN DE AUTORÍA.....	II
DEDICATORIA.....	III
AGRADECIMIENTO.....	IV
ÍNDICE DE CONTENIDO.....	V
RESUMEN	VII
ABSTRACT	VIII
1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO	1
1.1 Objetivo general.....	2
1.2 Objetivos específicos	2
1.3 Alcance	3
1.4 Marco Teórico	4
2 METODOLOGÍA	7
2.1 Metodología de Desarrollo	7
Roles	8
Artefactos	9
2.2 Diseño de la arquitectura	12
Patrón Arquitectónico MVC	13
2.3 Herramientas de desarrollo	13
3 RESULTADOS.....	16
Sprint 0. Configuración del ambiente de desarrollo	16
Sprint 1. Diseño y codificación de rutas para el usuario Administrador.	20
Sprint 2. Diseño e implementación de rutas para el Cliente.....	25
Sprint 3. Creación de modelo de IA.....	30
Sprint 4. Pruebas de Backend.....	34
Sprint 5. Despliegue del Backend	38
4 CONCLUSIONES	40
5 RECOMENDACIONES	41
6 REFERENCIAS BIBLIOGRÁFICAS.....	42
7 ANEXOS	46
ANEXO I.....	47
ANEXO II.....	48

ANEXO III.....	77
ANEXO IV	78

RESUMEN

Flor&Cera es un emprendimiento dedicado a la elaboración de productos artesanales como jabones, aromatizantes y velas, utilizando ingredientes naturales y ecológicos con el propósito de promover el cuidado del medio ambiente. Actualmente, el emprendimiento busca brindar una interacción más satisfactoria al momento de adquirir sus productos mediante estrategias que resalten el valor de sus productos naturales y personalizados. Sin embargo, enfrenta la problemática de no contar con una plataforma tecnológica para comercializar sus productos, ya que solo dispone de un punto de venta físico. Esta limitación reduce su alcance y visibilidad en el mercado, lo cual representa una oportunidad para proponer una solución tecnológica que optimice su presencia digital y mejore su competitividad.

Con el objetivo de fortalecer el emprendimiento, el presente proyecto de Integración Curricular ha desarrollado un backend para gestionar la venta de productos artesanales. Además, este componente incluye un modelo de Inteligencia Artificial (IA) que permite a los clientes personalizar sus productos según sus preferencias, generando así experiencias únicas. De esta manera, la personalización no solo añade valor al usuario, sino que también posiciona al emprendimiento como un negocio innovador dentro del sector.

Este documento se encuentra organizado en cinco capítulos principales. El primer capítulo, aborda la problemática del componente, establece los objetivos generales y específicos, define el alcance del proyecto y presenta el marco teórico necesario para su comprensión. El segundo capítulo, describe la metodología Scrum que se ha utilizado para la gestión del proyecto, así como el diseño arquitectónico, las herramientas y librerías que se han empleado durante el desarrollo. El tercer capítulo, se enfoca en los resultados de cada Sprint, desde la configuración del entorno de desarrollo hasta el despliegue del backend. El cuarto capítulo, presenta las conclusiones que se han alcanzado al finalizar el proyecto, y finalmente, el quinto capítulo incluye una serie de recomendaciones orientadas a la mejora continua del componente y su implementación futura.

PALABRAS CLAVE: Backend, Inteligencia Artificial, Hugging Face, Express.JS, Productos artesanales.

ABSTRACT

Flor&Cera is an entrepreneurial venture dedicated to the production of handcrafted products such as soaps, air fresheners, and candles, using natural and eco-friendly ingredients with the purpose of promoting environmental care. Currently, the business seeks to provide a more satisfying interaction when purchasing its products through strategies that highlight the value of its natural and personalized offerings. However, it faces the challenge of lacking a technological platform to commercialize its products, as it only has a physical point of sale. This limitation reduces its market reach and visibility, representing an opportunity to propose a technological solution that enhances its digital presence and improves its competitiveness.

With the aim of strengthening the business, this Curricular Integration Project has developed a backend system to manage the sale of handcrafted products. Additionally, this component includes an Artificial Intelligence (AI) model that allows customers to personalize their products according to their preferences, thereby creating unique experiences. In this way, personalization not only adds value for the user but also positions the venture as an innovative business within the sector.

This document is organized into five main chapters. The first chapter, addresses the component's problem statement, establishes general and specific objectives, defines the project's scope, and presents the theoretical framework necessary for its understanding. The second chapter, describes the Scrum methodology used for project management, as well as the architectural design, tools, and libraries employed during development. The third chapter, focuses on the results of each Sprint, from the development environment setup to backend deployment. The fourth chapter, presents the conclusions reached at the end of the project, and finally, the fifth chapter, includes a series of recommendations aimed at the continuous improvement of the component and its future implementation.

KEYWORDS: Backend, Artificial Intelligence, Hugging Face, Express.JS, Handcrafted Products.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

Flor&Cera es un emprendimiento dedicado a la elaboración de productos artesanales como jabones, aromatizantes y velas, utilizando ingredientes naturales y ecológicos con el propósito de promover el cuidado del medio ambiente. Su misión es ofrecer productos personalizados que brinden una experiencia única a cada cliente, permitiéndoles elegir fragancias, colores, formas y empaques según sus gustos y necesidades. Además, cada producto es elaborado de manera artesanal, lo que asegura un acabado cuidadoso y personalizado en cada pieza garantizando de esta manera, una calidad excepcional hacia su clientela [1].

Actualmente, el emprendimiento busca brindar una interacción más satisfactoria al momento de adquirir sus productos mediante estrategias que resalten el valor de sus productos naturales y personalizados. Sin embargo, enfrenta la problemática de no contar con una plataforma tecnológica para comercializar sus productos, ya que solo dispone de un punto de venta físico. Esta limitación reduce su alcance y visibilidad en el mercado, lo cual representa una oportunidad para proponer una solución tecnológica que optimice su presencia digital y mejore su competitividad.

En un contexto donde las compras en línea están en constante crecimiento, la mayoría de plataformas web limitan la experiencia del cliente únicamente a una simple visualización de productos, sin ofrecer elecciones significativas de personalización. Además, esta falta de opciones impide que los consumidores encuentren productos que se ajusten a sus preferencias o necesidades, lo que produce disgusto al momento de la entrega del producto disminuyendo de esta manera el interés y el compromiso en futuras compras. De esta manera, la ausencia de interactividad y funcionalidades personalizadas representan una barrera para el crecimiento de plataformas electrónicas que se basan en este tipo de negocios artesanales como lo es Flor&Cera, generando una necesidad de buscar respuestas innovadoras y así obtener una experiencia más significativa entre el comprador y el negocio [1].

Hoy en día, la Inteligencia Artificial, es considerada como una de las tecnologías emergentes, debido a que ofrece respuestas innovadoras a cada una de las problemáticas que se le presentan, en este caso, permite adaptar los productos que ofrece el e-commerce a las preferencias y necesidades que presentan los clientes. Con esta tecnología, es posible personificar aspectos como aromas, colores, formas, esencias e inclusive generar recomendaciones en base al historial de compras y de navegación del cliente [2]. A su vez, la integración de esta funcional, ayuda a que los clientes obtengan una experiencia

innovadora, que combina lo artesanal con lo tecnológico, facilitando de esta forma, la toma de decisiones y aumentando el nivel de satisfacción.

Con el propósito de ayudar a fortalecer el emprendimiento de Flor&Cera, se ha desarrollado un backend en este Trabajo de Integración Curricular, como parte de una plataforma de comercio electrónico, dado que no solo actúa como un punto de venta digital, sino que también dispone de herramientas avanzadas como lo es la personalización, donde los clientes pueden crear su producto mediante sus preferencias usando IA o creando de una manera interactiva como lo es a través de un juego. Para ello, el componente se encarga de gestionar todas las funcionalidades esenciales para la operación del e-commerce, entre ellos el catálogo de productos, perfiles de usuarios y proceso de compra. Aparte, la lógica de recomendaciones se ha diseñado mediante el uso de un modelo de Inteligencia Artificial. A partir de este componente backend, se pretende no solo mejorar la experiencia del cliente, sino que el emprendimiento tenga una visibilidad mayor en el mercado y así fortalecer su competitividad por medio del uso de la tecnología actual.

1.1 Objetivo general

Desarrollar el backend del e-commerce para la venta de productos artesanales personalizados basado en IA.

1.2 Objetivos específicos

1. Definir los requisitos para la implementación del backend, teniendo en cuenta la personalización de productos artesanales.
2. Diseñar un modelo de datos eficiente basado en los requisitos que se han determinado anteriormente.
3. Codificar cada una de las rutas públicas y privadas con el objetivo de garantizar una correcta funcionalidad y el respectivo consumo.
4. Integrar un modelo de IA que permita generar recomendaciones personalizadas de productos según las preferencias del cliente.
5. Implementar pruebas que verifiquen el funcionamiento adecuado de los endpoints y garanticen la seguridad de los datos.
6. Publicar el backend a producción para permitir la integración con los otros componentes y asegurar su disponibilidad por parte de los clientes.

1.3 Alcance

Este Trabajo de Integración Curricular tiene como propósito principal desarrollar un backend que no solo integre funciones de e-commerce, sino que integre funcionalidades basadas en IA para generar recomendaciones de productos artesanales, considerando las preferencias del cliente en base a su historial de compras y otros aspectos. Además, el backend utiliza una serie de herramientas y librerías modernas que garanticen la escalabilidad y eficiencia. Asimismo, en cada etapa se realizan una serie de pruebas para detectar y solucionar posibles inconvenientes, así como para la gestión de los datos, el backend dispone de varios mecanismos para la protección de la información tales como autenticación, autorización, cifrado contraseñas y roles de acceso con el objetivo de proporcionar un procesamiento ágil y confiable de la información y del consumo por parte los demás componentes.

Mediante el uso de la metodología ágil Scrum, se ha llevado a cabo el componente backend a un entorno de producción, es decir que el proceso ha comenzado con la identificación y organización de los requisitos mediante módulos. De esta forma, esta metodología permite planificar el desarrollo en ciclos, lo que facilita el uso de recursos y tiempo. No obstante, el desarrollo del componente se lleva a cabo empleando Node.js con la estructura Modelo, Vista y Controlador facilitando de esta manera escalabilidad y organización del código. Asimismo, se ha integrado un modelo de Inteligencia Artificial para la generación de recomendaciones personalizadas, empleando una base de datos no relacional (NoSQL) que sirve para el almacenamiento de la información, lo que permite un acceso eficiente a los datos. Por otro lado, en cuanto a la etapa de codificación, se ha realizado pruebas continuas para comprobar el correcto funcionamiento de los módulos que se han creado, asegurando así, estabilidad como seguridad. Por último, el backend se despliega a producción con el propósito de que otros componentes puedan hacer el consumo respectivo.

Este componente gestiona las siguientes acciones por cada rol de usuario, los cuales son:

Roles que se han establecido

- Administrador
- Cliente

Endpoints que comparten ambos roles

- Inicio/cierre de sesión.
- Recuperar contraseña.

Endpoints para el rol administrador

- Gestionar usuarios.
- Gestionar categorías y productos.
- Gestionar ventas.
- Gestionar reportes de ventas.

Endpoints para el rol cliente

- Registrarse.
- Modificar perfil.
- Visualizar categorías y productos.
- Gestionar personalización de productos.
- Gestionar carrito de compras.
- Visualizar reporte de facturas.
- Visualizar reporte de ventas.

1.4 Marco Teórico

MONGODB

Es una base de almacenamiento de datos no relacional, la cual se ha caracterizado por su alta escalabilidad y flexibilidad. Además, permite que los desarrolladores puedan utilizarla rápidamente en diferentes tipos de proyectos, ya que una de sus principales características es el almacenamiento de datos en documentos flexibles, similares a JSON, lo que permite que los campos varíen entre documentos y que la estructura de los datos pueda modificarse con el tiempo. En otras palabras, este modelo de documentos se adapta de manera adecuada a los objetos en el código de la aplicación, lo que facilita una gestión adecuada de los datos [3].

JAVASCRIPT (JS)

Es un lenguaje de programación, empleado para crear páginas web de manera dinámica e interactivas. Es decir, se caracteriza por ser un lenguaje interpretado, en otras palabras, se traduce directamente a código máquina a través de un motor del navegador lo que facilita su aprendizaje y uso. En consecuencia, este lenguaje se integra fácilmente a cualquier página web siendo compatible con diversos marcos y lenguajes de desarrollo web [4].

API RESTful

Una API interviene como un tipo de puente, lo que quiere decir que facilita la comunicación entre distintos sistemas, de manera que, permite el intercambio de datos de una forma más protegida y estructurada. Ahora bien, estas APIs pueden ser utilizadas o desarrolladas para que otros sistemas o aplicaciones se comuniquen entre si mediante un lenguaje de programación. No obstante, la parte REST se refiere a un conjunto de reglas que facilita la comunicación confiable y de alto rendimiento, lo que permite su modificación e implementación. En ese sentido, las APIs pueden desarrollarse utilizando diversas estructuras y aquellas que se enfocan en el estilo REST se conocen como API RESTful, las cuales son alojadas en un web services para su consumo [5].

NODE.JS

Permite que JavaScript se pueda ejecutar en el lado del servidor gracias a su entorno de ejecución de código abierto. Además, se ejecuta en un único proceso, es decir, sin realizar un nuevo hilo para cada solicitud, ofreciendo de esta manera una ventaja distintiva para los desarrolladores, ya que pueden emplear JavaScript tanto en la parte del servidor como en la del cliente sin tener que aprender un lenguaje completamente nuevo [6].

EXPRESS.JS

Es uno de los múltiples Frameworks que acompaña a Node.js dentro del ecosistema de JavaScript, el cual está orientado a desarrollar aplicaciones web ya sea de una sola vista e híbridas. Además, este Framework contiene herramientas de gran valor, que ayudan a crear aplicaciones backend escalables. Por otro lado, proporciona funcionalidades que facilitan el manejo de solicitudes y respuestas HTTP, así como middlewares de enrutamiento y funcionalidades avanzadas para que el despliegue de aplicaciones sea más eficiente [7].

JSON (JavaScript Object Notation)

Es un formato abierto que es usado para el almacenamiento y transmisión de información entre un servidor y cliente. A su vez, se caracteriza por ofrecer una forma ligera y sencilla de presentar datos en comparación a otros estándares abiertos como lo es XML [8].

JWT

Json Web Token, es un formato JSON de estándar abierto que permite el intercambio seguro de datos entre distintas partes como cliente y servidor. No obstante, al ser un estándar abierto estos tokens son firmados digitalmente para la seguridad de la

autenticidad de los datos, además de que se pueden cifrar para la protección de confidencialidad. Además, la estructura de un JWT contiene tres componentes principales un encabezado que define el algoritmo de firma y tipo de token, una carga útil que incluye declaraciones y una firma que certifica la integridad del token. Es por ello, que este tipo de estándar es ampliamente usado en procesos de autenticación y autorización debido a su diseño seguro y de fácil almacenamiento [9].

NODEMAILER

Es una biblioteca que facilita el envío de notificaciones por email. Además, es compatible con diversos servicios de correo, lo cual permite enviar mensajes por varios protocolos siendo SMTP (Simple Mail Transfer Protocol) uno de los más conocidos. De este modo, se convierte en una herramienta útil para incluir en las funcionalidades de la aplicación de lado del servidor gracias a su flexibilidad y fácil integración [10].

POSTMAN

Es una herramienta con una interfaz gráfica para documentar y probar APIs, lo que facilita el envío de solicitudes HTTP, la configuración de encabezados, parámetros, cuerpos de solicitudes, así como el análisis de respuestas. Cabe considerar que su uso es fundamental para el desarrollo y mantenimiento de servicios web, ya que mejora los procesos de prueba y permite identificar errores de forma anticipada durante las distintas fases de desarrollo [11].

RENDER

Esta herramienta es un servicio en la nube que proporciona el proceso de despliegue de aplicaciones web, base de datos y servicios. Además, proporciona características como escalado automático, certificados SSL gratuitos y despliegues continuos desde repositorios de código como GitHub. Al facilitar la implementación y operación de aplicaciones en la nube, Render permite a los desarrolladores concentrarse en desarrollar funcionalidades sin tener que preocuparse por la infraestructura que sostiene la aplicación [12].

2 METODOLOGÍA

Una de las técnicas de investigación es sin duda el estudio de casos que consiste en el análisis profundo y detallado de un tema en particular y el cual es ampliamente utilizado en áreas como la investigación social, educativa, clínica, empresarial y en la actualidad en áreas de tecnología. Además, su diseño suele basarse en métodos cualitativos, aunque en algunos casos también incorpora enfoques cuantitativos para lograr de esta manera describir, comparar, evaluar y comprender distintos aspectos que conforman un problema de investigación [13].

Considerando los aspectos previamente expuestos, el presente proyecto adopta un enfoque metodológico centrado en el estudio detallado de casos específicos, permitiendo de esta manera realizar un análisis en profundidad de aquellos desafíos que enfrenta el propietario del emprendimiento, especialmente en lo concerniente a la gestión y personalización de productos artesanales. A su vez, se detalla como la Inteligencia Artificial es considerada como una tecnología emergente que puede mejorar la experiencia del cliente al permitir recomendarle productos en base a sus preferencias. A su vez, toda la información que se ha recolectado se ha obtenido de varias fuentes tanto primarias como secundarias, haciendo que se fortalezca el análisis del problema y que de esta manera se tengan fundamentos sólidos para el planteamiento de una solución y que la obtención de dichos resultados sea de ayuda para futuras iniciativas similares.

2.1 Metodología de Desarrollo

Es un conjunto estructurado de diferentes pasos que permiten organizar y optimizar el proceso de creación de un producto software con altos estándares de calidad. Además, cabe resaltar que este proceso abarca un conjunto de etapas que inicia desde la planificación, seguida del diseño, codificación, pruebas y, por último, producción. A su vez, dichas metodologías de desarrollo establecen un marco de trabajo que indican cada paso en el desarrollo del componente, incluyendo la asignación de roles y responsabilidades, lo que garantiza que el desarrollo de dicho componente se lleve a cabo de forma organizada y en concordancia con los propósitos que se han definido previamente en el proyecto [14].

Actualmente, hay varias metodologías para el desarrollo y despliegue a producción de un producto software con altos estándares de calidad. Entre ellas se destacan las metodologías ágiles, ya que se basan en un conjunto de principios y enfoques que dan preferencia a la colaboración, la flexibilidad ante cambios y sobre todo a la entrega de resultados útiles de manera constante. De este modo, en lugar de seguir un plan riguroso

y lineal, las metodologías ágiles se organizan en ciclos cortos llamado iteraciones o Sprints cuyo propósito es entregar avances funcionales del producto, obtener retroalimentación de los actores involucrados y ajustar el desarrollo en función de las necesidades emergentes en cada fase [15].

Scrum es una de las metodologías ágiles que más destaca, ya que, al ser un marco de gestión ágil y flexible, éste permite que el equipo de desarrollo pueda organizarse de manera autónoma y puedan colaborar de una forma más eficiente para alcanzar los objetivos establecidos. Además, la flexibilidad en su estructura hace que Scrum sea una opción preferida para los equipos que buscan una metodología que facilite tanto la organización interna como el trabajo conjunto [16]. De esta manera, la metodología Scrum, se ha llevado a cabo en el presente Trabajo de Integración Curricular, tanto para la planificación e implementación de las funcionalidades del backend como también para la gestión de roles, eventos y artefactos, favoreciendo de esta manera un desarrollo adaptable y alineado a estándares de calidad. Por consiguiente, las secciones detalladas a continuación, describen como se ha ejecutado cada una de las fases de esta metodología.

Roles

Con el propósito de implementar Scrum de una manera adecuada, es ideal conformar un equipo de trabajo multifuncional, cuyos miembros cuenten con habilidades necesarias para alcanzar el objetivo del producto final. A su vez, este equipo debe ser responsable de tomar decisiones sobre las tareas a realizar, cuando y como se ejecutarán, así mismo, tareas relacionadas con la colaboración del dueño del producto [17]. En este contexto, el equipo está conformado por un Product Owner, quien es el responsable del producto, por otra parte, el Scrum Master, quien es responsable de facilitar el cumplimiento de los principios ágiles y por último el Developer Team, conformado por los desarrolladores que implementan las funcionalidades del producto de manera significativa. Por lo que a continuación, se presentan las personas asignadas a cada uno de estos roles, y así mismo, se presentan las tareas definidas para el componente backend.

Product Owner

Es el responsable de guiar al equipo hacia los mejores resultados posibles en el proyecto, dado que, tiene un conocimiento profundo sobre las características del producto final. A su vez, es quien toma las decisiones más importantes, además de encargarse de administrar de manera eficiente el Product Backlog [18]. En la **Tabla 2.1** se puede observar al responsable que se ha designado.

Scrum Master

Su función principal consiste en asegurar que todos los miembros comprendan y apliquen los principios y prácticas ágiles de manera coherente. Además, se encarga de eliminar los obstáculos que puedan interferir en el flujo de trabajo [19]. En la **Tabla 2.1** se muestra al responsable que se ha designado.

Development Team

Se trata de un grupo de personas que colaboran activamente en la creación de un software, producto o servicio. Además, participa en todas las fases del proceso Scrum, desde la generación de la idea hasta su entrega final [20]. En la **Tabla 2.1** se muestra al responsable que se ha designado, quien trabaja de manera colaborativa con los otros integrantes encargados de los demás componentes del e-commerce.

Tabla 2.1 Distribución de responsabilidades.

ROL	RESPONSABLE
Product Owner	Sr. Henry Pazto.
Scrum Master	Ing. Byron Loarte, MSc.
Development Team	Srta. Estefanía Sánchez.

Artefactos

En Scrum, los artefactos ofrecen una visión compartida del trabajo a ejecutar, permitiendo que todos los miembros del equipo comprendan con claridad que debe hacerse y quién es el responsable de cada tarea. Es así que, los artefactos ofrecen transparencia, colaboración efectiva y contribuyen al cumplimiento de los objetivos establecidos [21]. A continuación, se definen los artefactos que se han seleccionado para el registro de la información del componente backend.

Recopilación de Requerimientos

La recopilación de requisitos constituye un proceso fundamental para identificar y documentar las necesidades de los interesados, asegurando así que se cumplan con los objetivos del proyecto. Además, estos requisitos comprenden tanto necesidades como expectativas, las cuales son recogidas mediante técnicas como reuniones o entrevistas,

con el propósito de definirlos de manera clara, precisa y verificable [22]. La **Tabla 2.2** presenta un extracto ilustrativo del artefacto, mientras que, en el **ANEXO II** se presenta el contenido completo.

Tabla 2.2 Documentación de un requisito.

ID	DESCRIPCIÓN DEL REQUERIMIENTO
RR - 001	<p>Para los roles cliente y administrador se requieren desarrollar endpoints que permitan:</p> <ul style="list-style-type: none"> Gestionar su cuenta personal.

Historias de Usuario

Es por lo general una tabla con una descripción detallada sobre de una funcionalidad del producto software vista desde el enfoque del dueño del negocio. Además, este recurso es ampliamente habitual en enfoques ágiles ya que permiten a los equipos de desarrollo trabajar en ciclos iterativos, enfocándose en entregar rápidamente pequeñas y valiosas funcionalidades para los usuarios finales o partes interesadas [23]. En el caso del backend, la **Tabla 2.3** muestra un ejemplo representativo del artefacto, mientras que, en el **ANEXO II** se presenta el contenido completo.

Tabla 2.3 HU – Gestionar cuenta (administrador y cliente).

HISTORIA DE USUARIO	
Identificador: HU-001	Usuario: Administrador y Cliente
Nombre historia: Gestionar cuenta	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Iteración asignada: 1	
Responsable (s): Estefanía Sánchez	
<p>Descripción: El componente backend entrega varios endpoints que son utilizados por el administrador y clientes para:</p> <ul style="list-style-type: none"> Realizar el registro (cliente). Iniciar y cerrar sesión. 	

- Modificar perfil.
- Recuperar contraseña.

Observación: Todas las rutas descritas son accesibles por medio de endpoints privados y públicos. No obstante, para el administrador no hay registro ya que el equipo de desarrollo otorga las credenciales de acceso y para el módulo modificar perfil es gestionado únicamente por el perfil cliente.

Product Backlog

Constituye un listado estructurado y priorizado de tareas destinadas al equipo encargado de desarrollo, el cual se construye tomando como base tanto la planificación general del proyecto y de los requisitos que se han definido previamente. Además, las tareas con mayor relevancia se ubican en la parte superior, permitiendo que el equipo tenga claridad sobre cuáles entregables deben entregarse primero [24]. La **Tabla 2.4** presenta un fragmento representativo de este artefacto, mientras que, en el **ANEXO II** se presenta el contenido completo.

Tabla 2.4 Fragmento del artefacto.

HU	DESCRIPCIÓN DE LA HISTORIA	ITERACIÓN ASIGNADA	ESTADO	PRIORIDAD
HU-001	Gestionar cuenta	2	Finalizada	Alta

Sprint Backlog

Se trata de un listado dinámico y organizado por iteraciones que se descompone en unidades de trabajo más manejables con el objetivo de tener una mayor comprensión de cada una y asignarlas a cada integrante del equipo para su ejecución [25]. La **Tabla 2.5** presenta un ejemplo representativo de este artefacto, mientras que, en el **ANEXO II** se presenta el contenido completo.

Tabla 2.5 Fragmento del artefacto.

SPRINT BACKLOG						
SB	NOMBRE	COMPONENTE	HU	HISTORIA DE USUARIO	TAREAS	PLAZO PREVISTO
SB-001	Diseño y codificación de endpoints para el usuario administrador y cliente.	Módulo de Registro	HU-001	Gestionar cuenta	<ul style="list-style-type: none"> • Creación de endpoints para gestionar el inicio/cierre de sesión, así como la recuperación de contraseñas para el administrador. 	20H
		Módulo de Iniciar y cerrar sesión			<ul style="list-style-type: none"> • Creación de rutas que permitan registrarse, iniciar/cerrar sesión, modificar perfil y restablecer contraseña para el cliente en caso de olvido. 	
		Módulo de Modificar Perfil			<ul style="list-style-type: none"> • Validación de datos ingresados. 	
		Módulo de Recuperar Contraseña			<ul style="list-style-type: none"> • Registro en base de datos. • Verificación de integridad de registros. 	

2.2 Diseño de la arquitectura

El diseño arquitectónico de software, surgida en la década de 1960, se basa en modelos, patrones y abstracciones teóricas para planificar el desarrollo de sistemas complejos antes de su implementación. Además, esta planificación proporciona una guía detallada que facilita la comprensión de cómo se integran los distintos componentes internos y externos del producto software. En este contexto, un patrón de arquitectura se define como una solución reutilizable y general para problemas recurrentes en ingeniería de software, similar a los patrones de programación, pero enfocados en la estructura del sistema a un nivel

más alto y abstracto [26]. A continuación, se expone el patrón que se ha elegido para llevar a cabo la construcción del presente componente denominado backend.

Patrón Arquitectónico MVC

Este patrón es ampliamente manejado en el desarrollo de software, dada a su capacidad de organizar el código en tres partes importantes el Modelo, el cual se encarga de la manipulación de datos que son fundamentales para la aplicación, la Vista, la cual se encarga de mostrar los datos del usuario a partir de diversas formas de visualización, y, por último, el Controlador, que actúa como intermediario procesando las acciones del usuario y ejecutando la lógica. Dicha división de responsabilidades beneficia el desacoplamiento entre componentes, mejora la mantenibilidad y facilita la reutilización eficiente del código [26]. En la **Figura 2.1**, se ilustra el patrón arquitectónico para un mejor entendimiento.

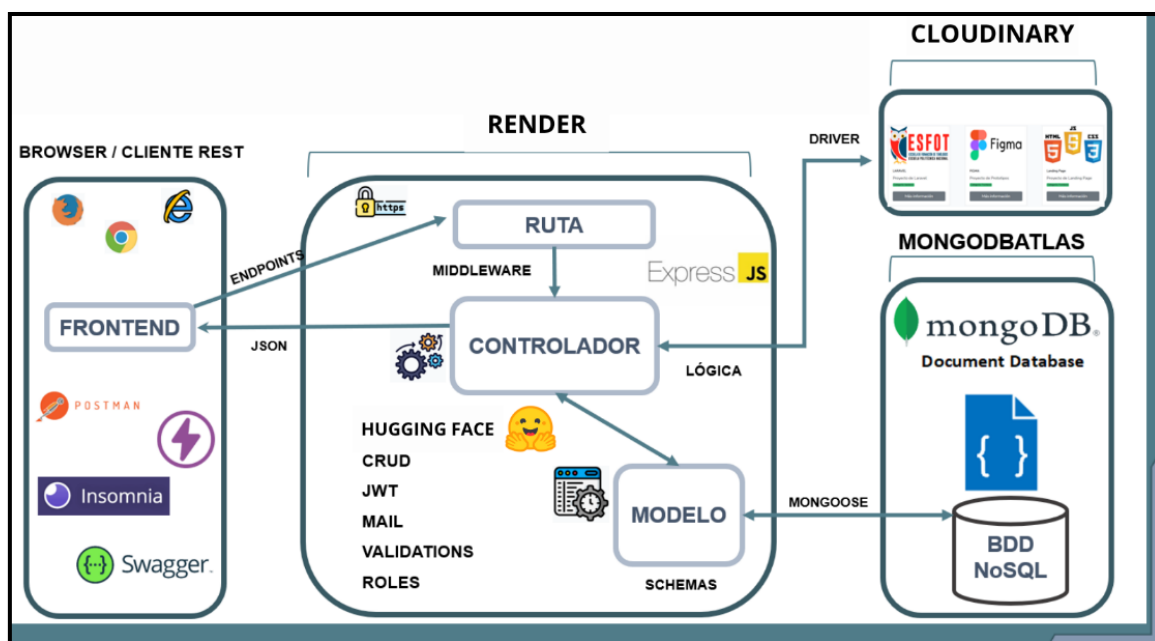


Figura 2.1 Estructura de backend basado en MVC.

2.3 Herramientas de desarrollo

Estas herramientas son tecnologías diseñadas para facilitar y mejorar el proceso de codificación de un producto de software. A su vez, funcionan como un intermediario entre una realidad física y operaciones computacionales, lo que permite una interacción mas eficaz con sistemas informáticos. Es por ello que, entre dichas herramientas se incluyen lenguajes de programación, Frameworks y plataformas que disuelven distintos niveles de complejidad, lo que ayuda a resolver problemas avanzados de una manera más accesible

[27]. En ese sentido, en la **Tabla 2.6** se describen las herramientas claves que se han utilizado para el proceso de codificación del componente.

Tabla 2.6 Selección de herramientas y justificación.

HERRAMIENTA	JUSTIFICACIÓN
GitHub	Aloja el código backend en la nube con la posibilidad de compartirlo con otros desarrolladores para su visualización y modificación [28].
Visual Studio Code	Permite gestionar el código backend de manera eficiente, fácil y rápida gracias a una serie de extensiones [29].
Postman	Permite testear de manera simple las APIs públicas y privadas del backend, agrupándolas en colecciones o catálogos [30].
Express	Gracias a este Framework, el desarrollo del backend se lo ha realizado de una manera ágil con varias funcionalidades escalables [31].
Render	Ayuda a compilar y desplegar el backend a un entorno de producción, así como gestionar las peticiones de manera independiente [32].
MongoDB	Gestiona la base de datos del backend de forma escalable y no relacional [33].

Librerías

Las librerías en programación son recursos que mejoran el desarrollo de software al ofrecer bloques de código reutilizables, lo que permite a los desarrolladores implementar funcionalidades de manera sencilla y rápida [34]. En por ello que, en la **Tabla 2.7** se indican las principales bibliotecas que se han utilizado para el backend, comúnmente denominadas también como librerías.

Tabla 2.7 Selección de librerías y descripción.

LIBRERÍA	DESCRIPCIÓN
Bcryptjs	Ayuda a proteger contraseñas de usuarios [35].

Cors	Facilita la interacción de un dominio con los recursos de otro dominio distinto [36].
Dotenv	Permite ocultar datos sensibles y luego asignar variables de entorno para poder usar [37].
Nodemailer	Permite el envío de correos electrónicos de forma sencilla desde varios servicios SMTP [38].
Mongoose	Permite escribir consultas en la base de datos de MongoDB [39].
JSON Web Token	Permite transferir la identidad de un usuario entre dos entidades de manera segura [40].

3 RESULTADOS

En esta sección del documento, se muestran los resultados que se han logrado conseguir durante la fase de desarrollo, donde se resalta la verificación funcional de cada endpoint conforme al tipo de usuario. No obstante, cada etapa del proyecto se caracteriza por la ejecución de tareas específicas que han sido determinantes para asegurar el correcto rendimiento del componente backend.

A continuación, se listan las actividades y resultados de cada Sprint, los cuales están estrechamente vinculados con los requerimientos que previamente se han establecido.

Sprint 0. Configuración del ambiente de desarrollo

Las actividades fundamentales para desarrollar esta iteración son las siguientes:

- Definición de los alcances y límites del backend.
- Estructura de la arquitectura base del backend.
- Estructuración del esquema de datos NoSQL.
- Asignación y especificación de roles que interactúan en el backend.

Definición de los alcances y límites del backend

Endpoints para gestionar cuenta de usuarios

El componente backend facilita varios endpoints que admiten realizar acciones como el registro, autenticación del correo electrónico, actualización del perfil y recuperación de contraseña. Así mismo, cabe resaltar que el acceso a estas funcionalidades depende de que el usuario haya confirmado previamente su correo electrónico, y que estos endpoints están preparados exclusivamente para los roles de administrador y cliente.

Endpoints para gestionar usuarios clientes

El componente backend dispone de varios endpoints destinados a la administración de usuarios con rol cliente. Es decir, que estos endpoints admiten la consulta de un listado completo de clientes, así también como obtener información detallada de un cliente en específico mediante su ID. Además, es considerable destacar que estos endpoints están preparados exclusivamente para uso por parte del rol administrador.

Endpoints para gestionar categorías, productos y ventas

El componente backend facilita varios endpoints que admite realizar varias acciones para gestionar categorías, productos y ventas. A su vez, estos endpoints permiten visualizar el conjunto completo de registros y obtener datos específicos de un elemento mediante su ID. Por otro lado, se debe considerar que el módulo de ventas es exclusivamente del rol administrador, mientras que los módulos de productos y categorías pueden ser consultados tanto clientes como por el administrador, aunque para los clientes no contiene permisos para realizar modificaciones, ya que estos están reservados únicamente para el rol administrador.

Endpoints para reportes de ventas

La creación de los reportes de ventas se realiza a través de los endpoints ya desarrollados del módulo de ventas. De igual manera, estos endpoints permiten adquirir información detallada al aplicar filtros como el rango de fechas, estados de transacción o montos. Es por ello que, el componente no requiere de rutas nuevas para realizar estas acciones, sino que reutiliza la lógica del módulo de ventas con el fin de mejorar la comprensión y facilitar la obtención de datos.

Endpoints para visualizar categorías, productos y reportes de ventas

El componente backend facilita endpoints que permiten la visualización de categorías y productos disponibles, así como la consulta de reportes de ventas. Asimismo, estas rutas están desarrolladas especialmente para el consumo del cliente en lo que se refiere a productos y categorías, a diferencia de los reportes de ventas son de uso exclusivamente para el rol administrador que solo son con fines de análisis del negocio.

Endpoints para gestionar personalización de productos

Este conjunto de endpoints permite a los clientes personalizar sus productos según sus preferencias. Además, se incluye un endpoint impulsado por Inteligencia Artificial que recomienda productos personalizados. Por medio de un juego interactivo, el usuario puede optar por características para crear su producto, tales como color, forma, aroma y esencias. Mientras que, por el lado de la IA, ésta analiza el historial de compras, las interacciones o vistas que ha tenido el cliente con productos que han llamado su atención, con ello, se genera una recomendación personalizada.

Endpoints para gestionar carrito de compras

Estas rutas están diseñadas para que los clientes puedan, añadir, visualizar, modificar o eliminar productos dentro de su carrito de compras. A su vez, el objetivo es simplificar la gestión de los productos que han sido elegidos antes de realizar el pago, con ello, se brinda una experiencia de compra fluida y controlada. Por último, cada acción que se realiza está asociada al usuario autenticado, asegurando así un manejo seguro y personalizado del carrito de compras.

Endpoints para visualizar reporte de facturas

Estas rutas están diseñadas para que los clientes puedan acceder al historial detallado de sus facturas a partir de las ventas que hayan hecho. Asimismo, esta gestión proviene de los endpoints de ventas para generar reportes más precisos en las facturas, además, el usuario obtiene transparencia sobre sus transacciones y a su vez facilita el rastreo de sus compras.

Estructura de la arquitectura base del backend

Todo el proceso de construcción del backend se ha realizado empleando como lenguaje principal de programación JavaScript en conjunto con el entorno de trabajo Visual Studio Code [41]. Esta combinación ha permitido que la implementación de cada una de las funciones se realice de manera progresiva y que a futuro pueda ser consumido por distintas aplicaciones externas a través de peticiones. En la **Figura 3.1** se presenta la organización general de directorios y archivos que conforman el backend en conjunto con el patrón MVC.

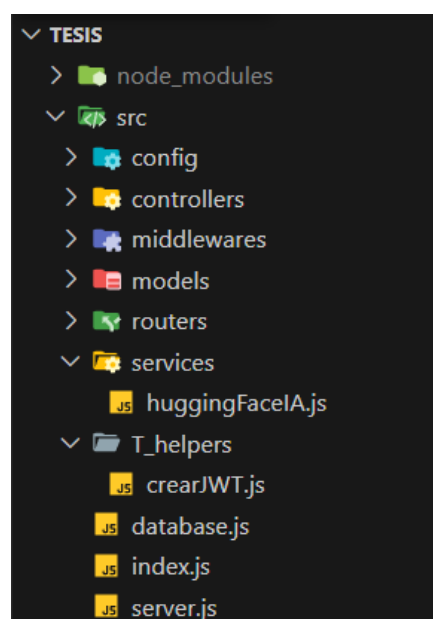


Figura 3.1 Estructura de la arquitectura backend.

Estructuración del esquema de datos NoSQL

Acerca del manejo de la información, se ha optado por utilizar MongoDB, debido a su naturaleza NoSQL que trabaja con colecciones y documentos en formato clave-valor [42]. Además, esta elección facilita una gestión eficiente de los datos, así como una mayor flexibilidad y escalabilidad del backend. En ese sentido, la **Figura 3.2** ilustra el diseño general de las colecciones del modelo de datos, en cuanto a la estructura de cada documento se la puede obtener de una forma más detallada en la sección correspondiente del **ANEXO II**.



Figura 3.2 Colecciones del modelo de datos (NoSQL).

Asignación y especificación de roles que interactúan en el backend

Para un manejo más ordenado y eficiente de la información en este proyecto se ha otorgado dos tipos de roles con el objetivo de designar tareas específicas a cada uno. De este modo, esta división de roles procura que la distribución de responsabilidades sea la adecuada con el objetivo de que los usuarios con ciertos permisos puedan acceder a ciertas funciones. Con ello, la **Figura 3.3** muestra la estructura de funciones a los usuarios.

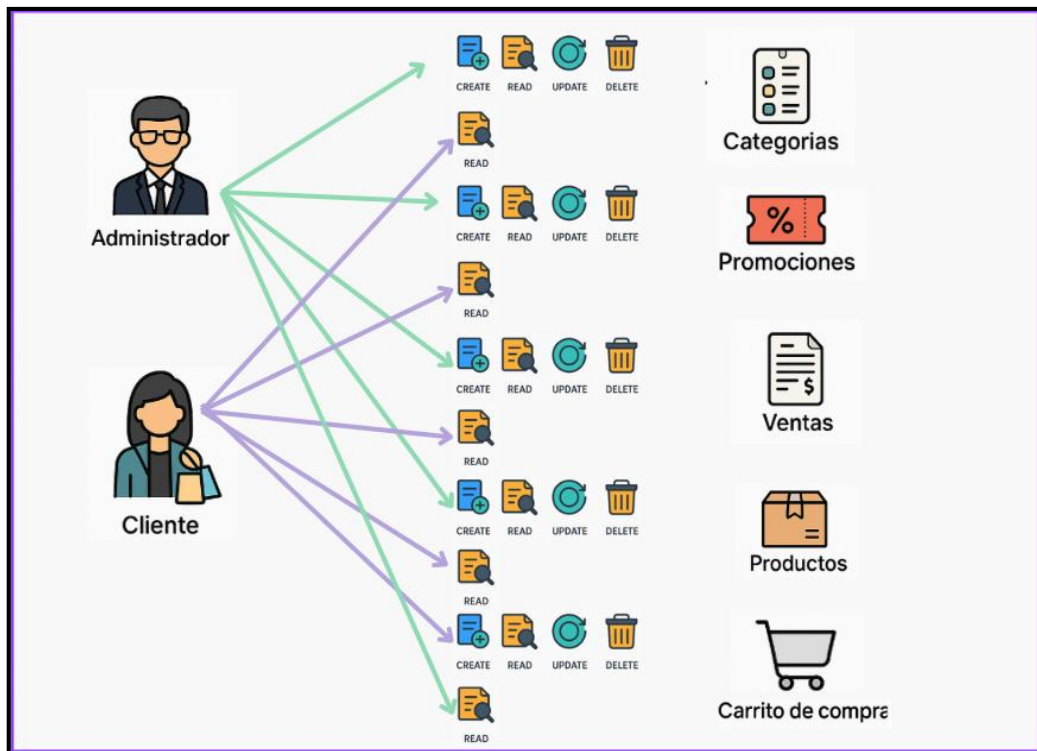


Figura 3.3 Asignación de funciones a los usuarios.

Sprint 1. Diseño y codificación de rutas para el usuario Administrador.

Este Sprint tiene como finalidad especificar las características relacionadas al usuario administrador y para ello, se han llevado a cabo las tareas descritas a continuación:

- Creación de rutas de acceso al backend por parte del administrador y cliente.
- Creación de rutas para la recuperación de contraseña.
- Creación de rutas para el control de usuarios.
- Creación de rutas para el manejo de categorías y productos.
- Creación de rutas para el manejo de ventas y reportes de ventas.

Creación de rutas de acceso al backend por parte del administrador y cliente

La construcción de endpoints específicos para el acceso de administradores y clientes ha permitido una autenticación diferenciada según el tipo de usuario. Además, para validar el acceso, se requiere el ingreso de credenciales y una vez autenticado correctamente, se muestra al usuario la información correspondiente a su perfil, tal y como se evidencia en la **Figura 3.4**, mientras que, el resultado de la prueba unitaria se detalla en la **Figura 3.5** y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

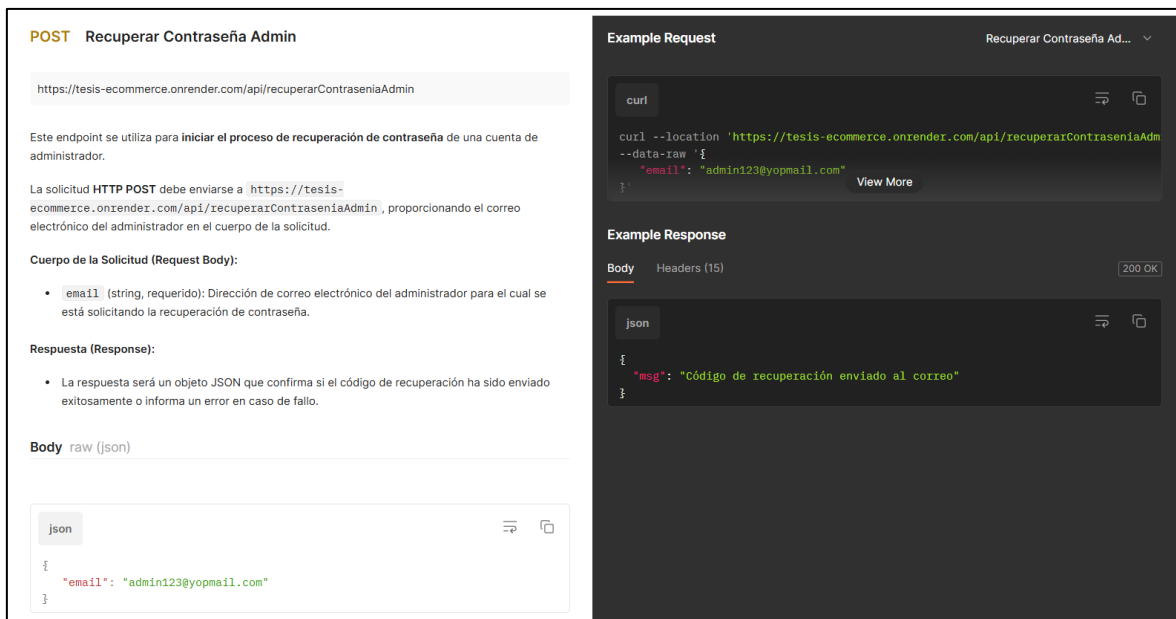


Figura 3.6 Proceso – Recuperar contraseña.

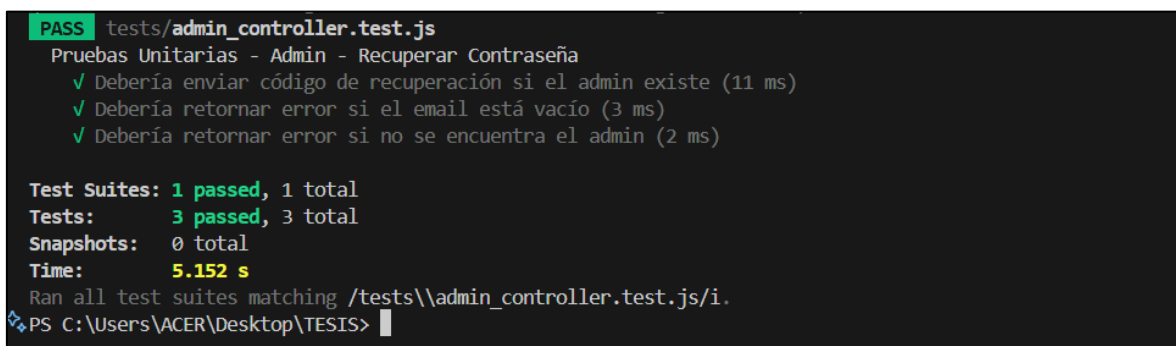


Figura 3.7 Prueba unitaria – Recuperar contraseña.

Creación de rutas para el control de usuarios

La construcción de rutas para la gestión de usuarios está diseñada exclusivamente para el rol administrador con el objetivo de visualizar los registros de clientes. Además, este proceso incluye la posibilidad de listar todos los usuarios o acceder a los detalles de un cliente específico mediante su ID. Por otra parte, el administrador no cuenta con permisos para crear, modificar o eliminar usuarios, ya que estas acciones son gestionadas por el propio cliente a través del registro y la edición de su perfil. En la **Figura 3.8** se expone la visualización de clientes, mientras que, la prueba unitaria se muestra en la **Figura 3.9** y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

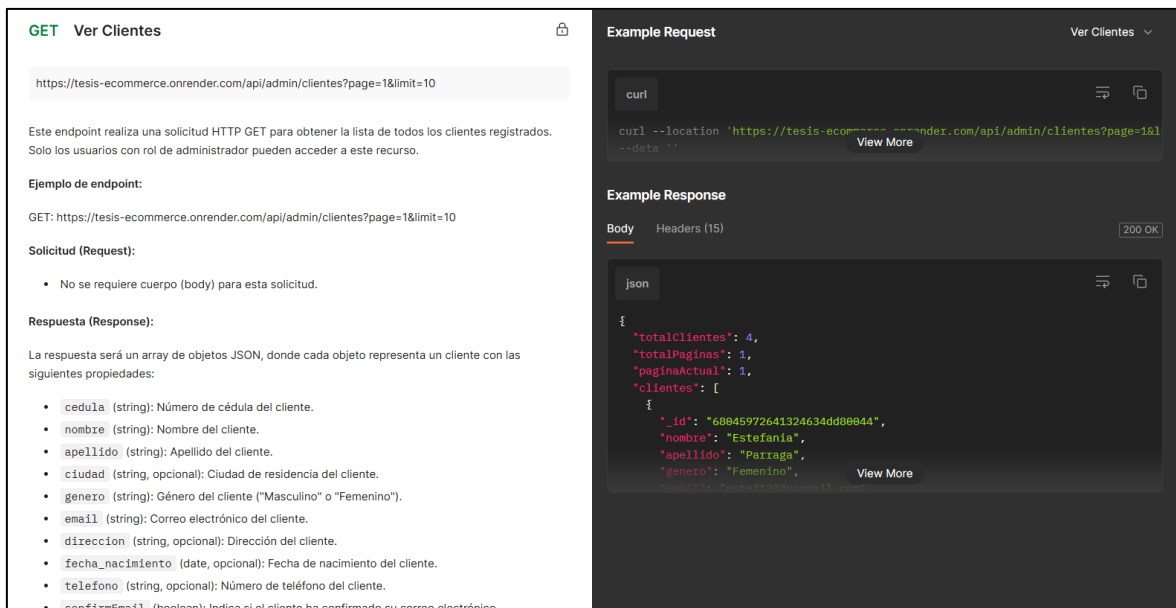


Figura 3.8 Solicitud de visualización de clientes.

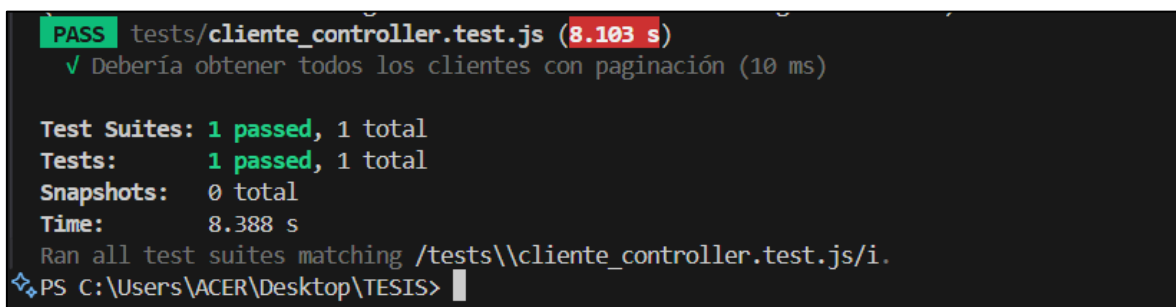


Figura 3.9 Prueba unitaria de visualización de clientes.

Creación de rutas para el manejo de categorías y productos

La creación de rutas para la gestión de categorías (jabones y velas) y productos permite que el rol administrador realice operaciones completas de creación, visualización, actualización y eliminación (CRUD) en ambos módulos. En cambio, los clientes solo tienen acceso a la visualización de categorías y productos disponibles, sin la posibilidad de realizar modificaciones. En la **Figura 3.10** se ilustran las rutas implementadas para esta funcionalidad, mientras que, la prueba unitaria se muestra en la **Figura 3.11** y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

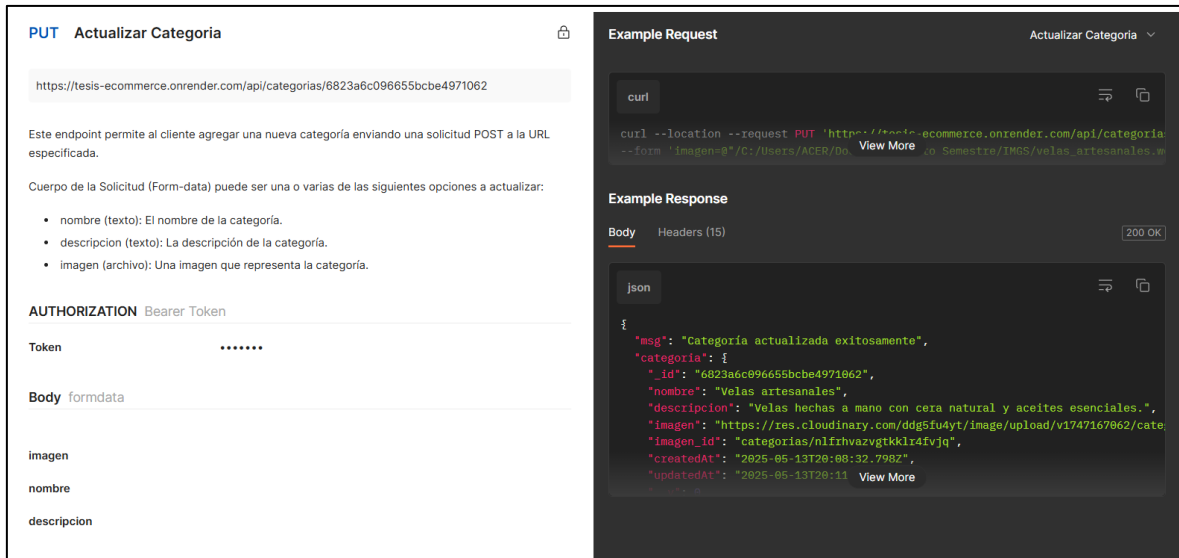


Figura 3.10 Solicitud para actualizar categorías.

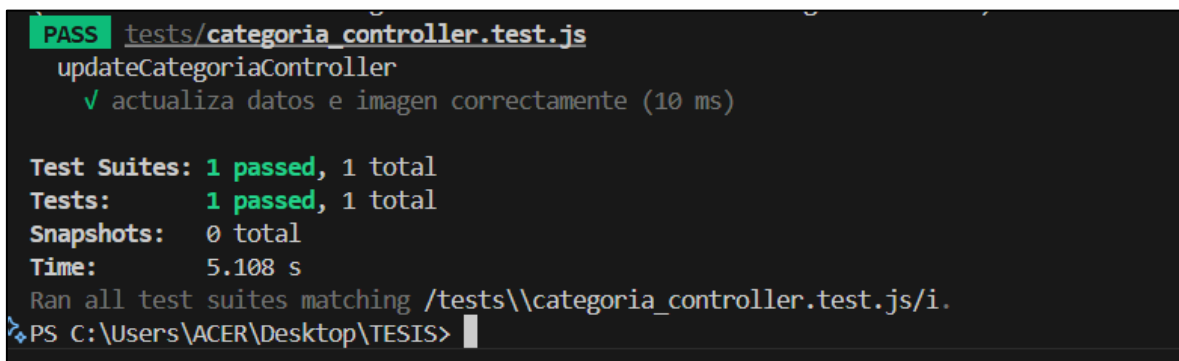


Figura 3.11 Prueba unitaria de actualización de categorías.

Creación de rutas para el manejo de ventas y reportes de ventas

La implementación de endpoints para la gestión de ventas permite al rol administrador realizar el registro, consulta, actualización y eliminación de las transacciones que han sido realizadas por los clientes. Además, estas rutas aseguran un control eficiente del historial de ventas y el seguimiento de las operaciones comerciales. En cuanto a los reportes de ventas, no se han definido endpoints exclusivos, ya que la información se extrae directamente a través de las rutas existentes del módulo de ventas, permitiendo obtener los datos necesarios para generar informes detallados. La **Figura 3.12** muestra una vista general de la ruta de ventas, en cuanto a la **Figura 3.13** muestra la prueba unitaria de dicho endpoint y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

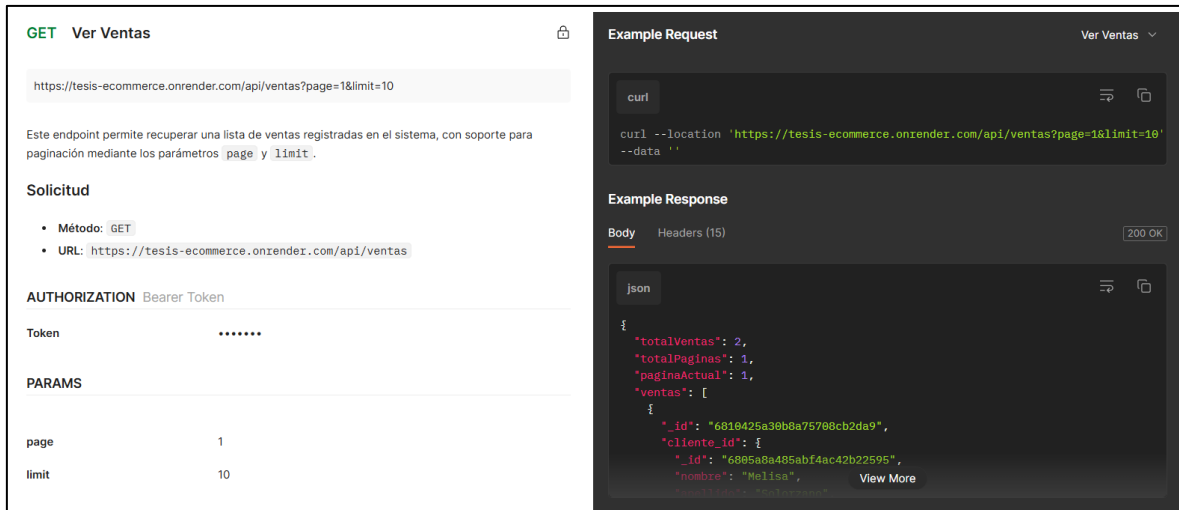


Figura 3.12 Solicitud para visualizar ventas.

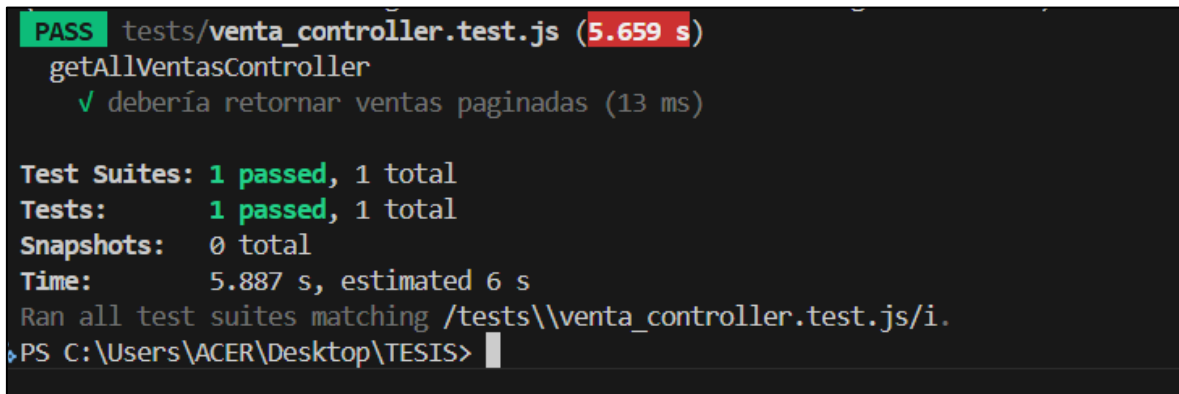


Figura 3.13 Prueba unitaria de visualización de ventas.

Sprint 2. Diseño e implementación de rutas para el Cliente.

Este Sprint tiene como finalidad especificar las características relacionadas al usuario cliente y para ello, se han llevado a cabo las tareas descritas a continuación:

- Creación de rutas para registro de clientes.
- Creación de rutas para la actualización de perfil cliente.
- Creación de rutas para la gestión de carrito de compras.
- Creación de rutas para la visualización de categorías y productos.
- Creación de rutas para la visualización de reportes de ventas y facturas.

Creación de rutas para registro de clientes

Durante esta fase se han desarrollado endpoints destinados al proceso de registro de nuevos clientes a partir del ingreso de datos personales básicos como nombre, apellido, correo electrónico, género y contraseña. Posteriormente, el backend envía automáticamente un

correo de verificación al cliente y solo tras confirmar el correo, el cliente puede acceder a las funcionalidades. Además, este proceso garantiza un registro seguro y controlado, tal y como se evidencia en la **Figura 3.14**, mientras que, en la **Figura 3.15** se detalla la prueba unitaria y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

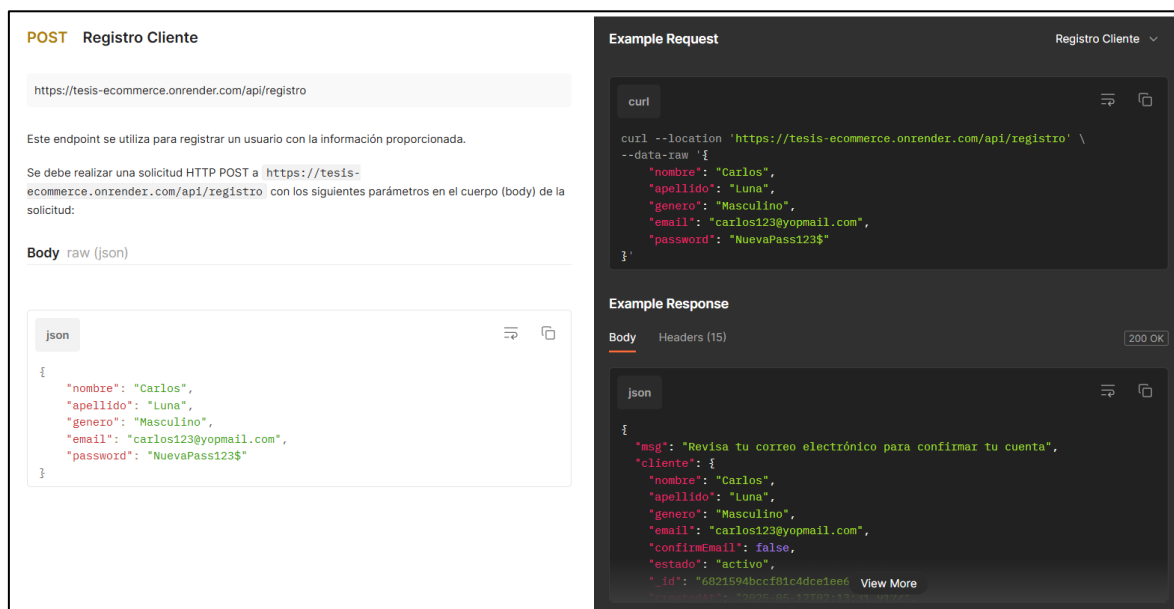


Figura 3.14 Solicitud de registro del cliente.

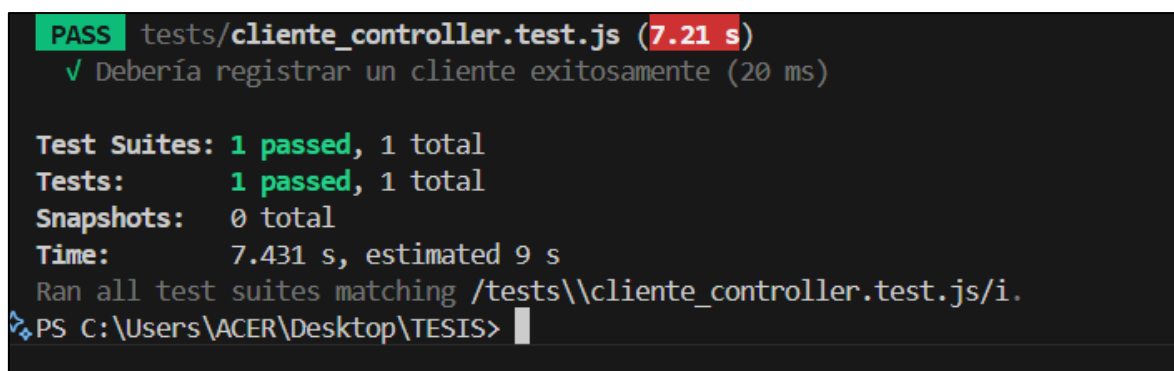


Figura 3.15 Prueba unitaria de registro del cliente.

Creación de rutas para la actualización de perfil cliente

En esta etapa se han desarrollado endpoints que permiten a los usuarios con rol cliente modificar su perfil personal. Además, esta funcionalidad permite actualizar datos como cédula, nombres, apellidos, email, dirección, teléfono e incluso su fecha de nacimiento. Adicional a ello, se han implementado validaciones para asegurar que los campos del formulario sean válidos y no estén duplicados en otros registros. En la **Figura 3.16** se

muestra un ejemplo de la respuesta al actualizar el perfil, en la **Figura 3.17** se detalla la prueba unitaria de este endpoint y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

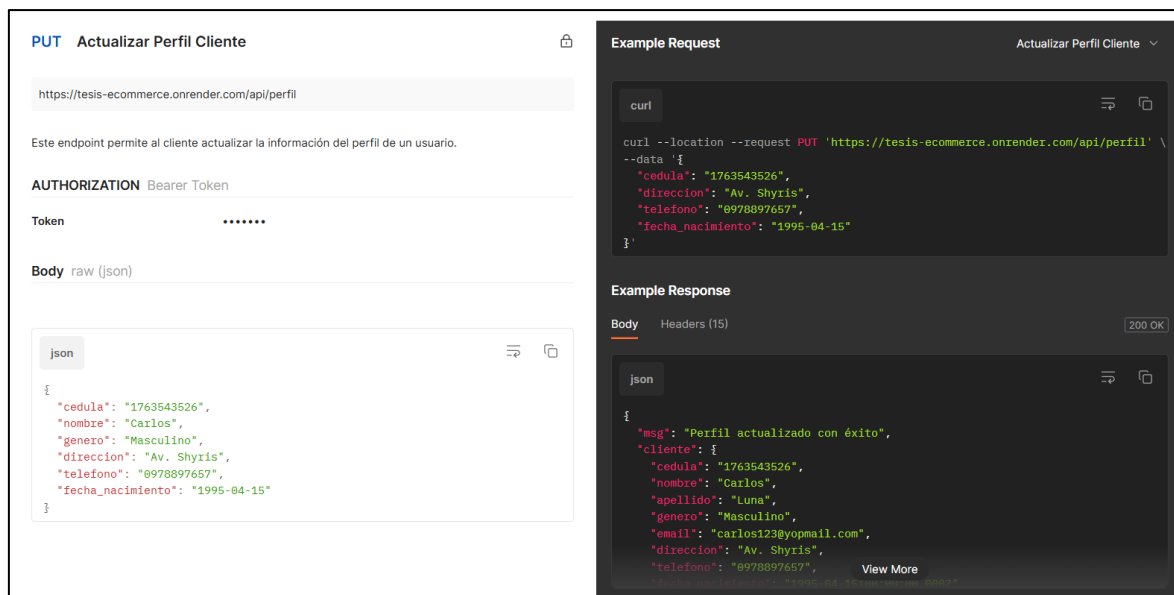


Figura 3.16 Solicitud de modificación de perfil de usuario.

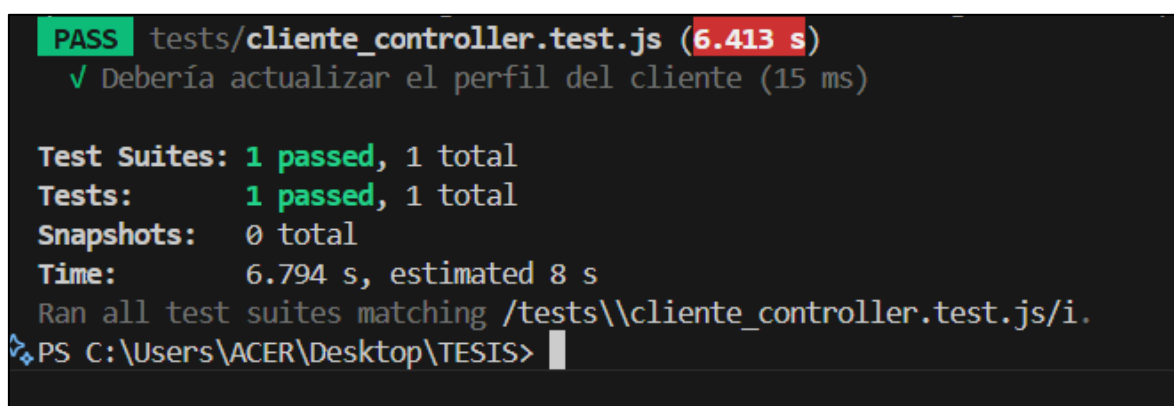


Figura 3.17 Prueba unitaria de modificación de perfil de usuario.

Creación de rutas para la gestión de carrito de compras

Durante este apartado se han desarrollado endpoints para permitir que los clientes gestionen su carrito de compras de forma dinámica. Estas rutas permiten realizar operaciones fundamentales como agregar productos, visualizar el contenido actual del carrito, actualizar la cantidad de productos o eliminarlos. Además, es importante destacar que este módulo es exclusivo para los clientes registrados, quienes son los únicos autorizados para interactuar con el carrito. En la **Figura 3.18** se muestra un ejemplo de petición, en tanto

que, la **Figura 3.19** se detalla la prueba unitaria de este endpoint y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

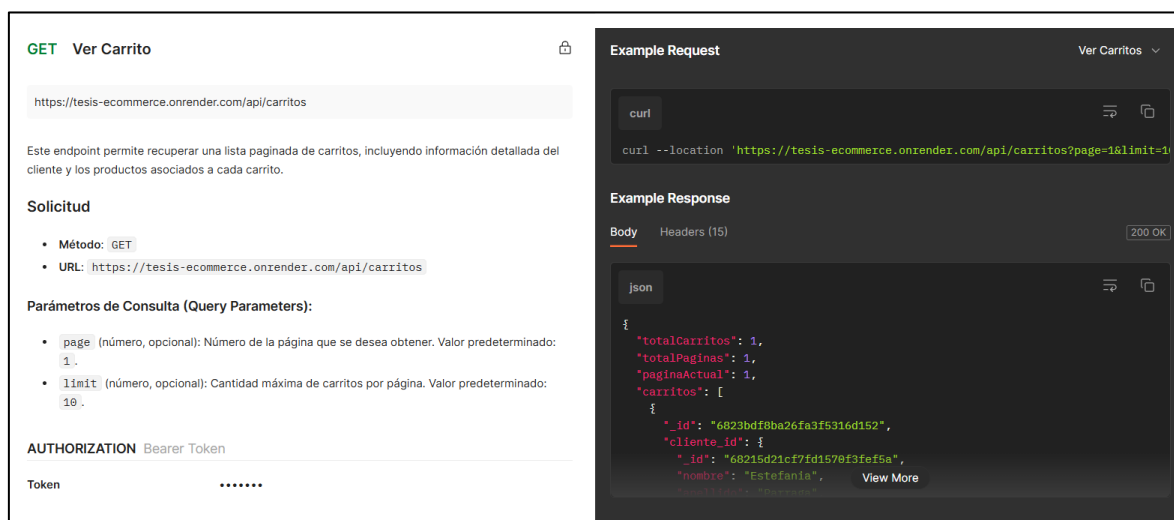


Figura 3.18 Solicitud de visualización de carrito de compras.

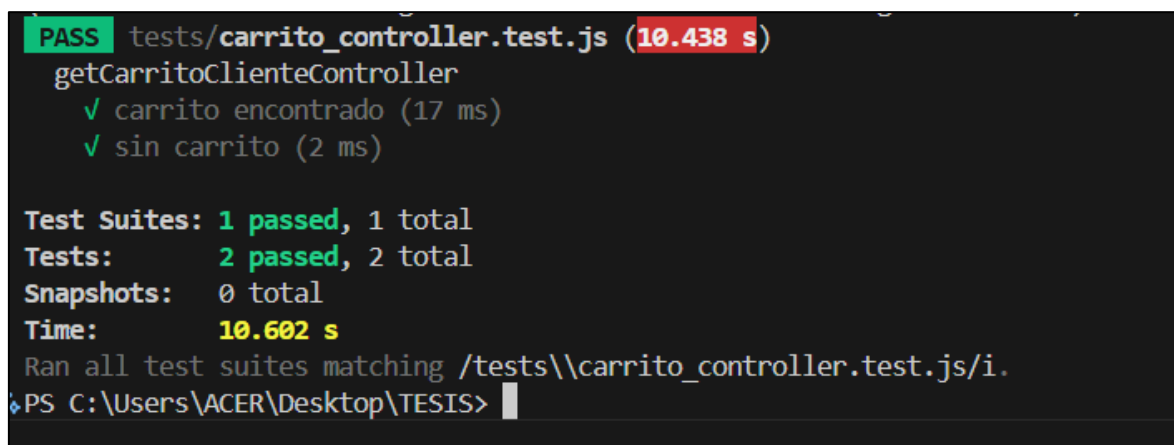


Figura 3.19 Prueba unitaria de visualización de carrito de compras.

Creación de rutas para la visualización de categorías y productos

En esta fase del desarrollo se han implementado endpoints que permiten a los usuarios clientes, consultar información disponible relacionada con categorías y productos. Además, estas rutas están orientadas únicamente a la lectura de datos, por lo que los clientes pueden visualizar el catálogo sin tener permisos para realizar modificaciones. La **Figura 3.20** presenta un ejemplo de respuesta a una solicitud, mientras que, la **Figura 3.21** se detalla la prueba unitaria de este endpoint y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

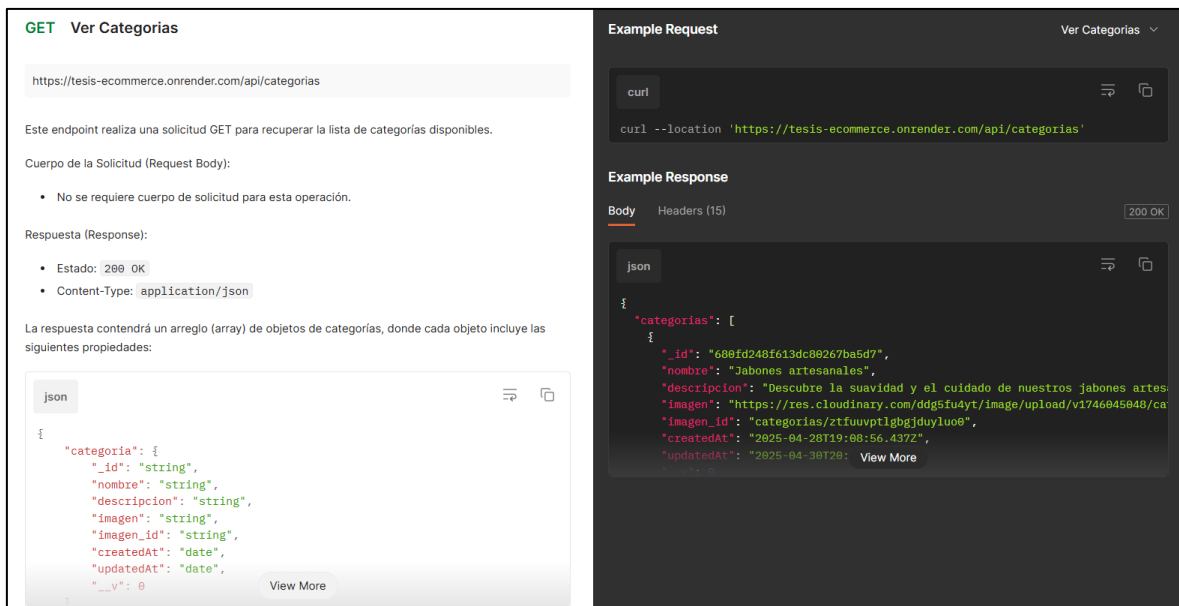


Figura 3.20 Solicitud de visualizar categorías.

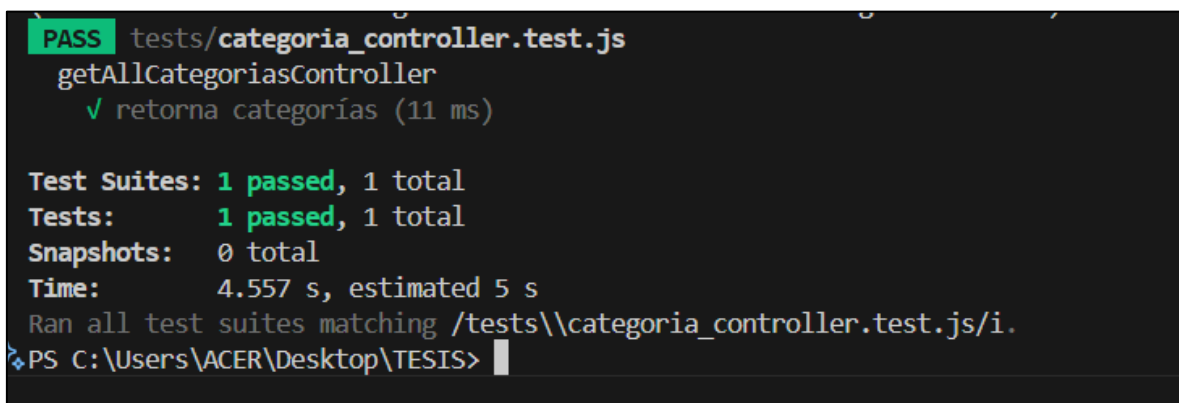


Figura 3.21 Prueba unitaria de visualizar categorías.

Creación de rutas para la visualización de reportes de ventas y facturas

Durante esta fase del desarrollo se han establecido endpoints correspondientes a la gestión y consulta de ventas, los cuales también cumplen la función de generar reportes. A diferencia de otros módulos, no se han desarrollado rutas independientes para los reportes, ya que se reutilizan las rutas existentes del módulo de ventas para extraer la información necesaria tanto para el administrador como para el cliente. De esta manera, el administrador tiene acceso a la totalidad de datos relacionados con las ventas, lo que le permite generar reportes detallados que pueden ser utilizados para el análisis de rendimiento comercial. Por su parte, el cliente puede acceder únicamente a la información vinculada con sus propias transacciones, visualizando los detalles de sus compras en forma de facturas. En la **Figura 3.22** se puede visualizar un ejemplo de cómo el cliente

puede ver sus facturas, mientras que, la **Figura 3.23** se detalla la prueba unitaria de este endpoint y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

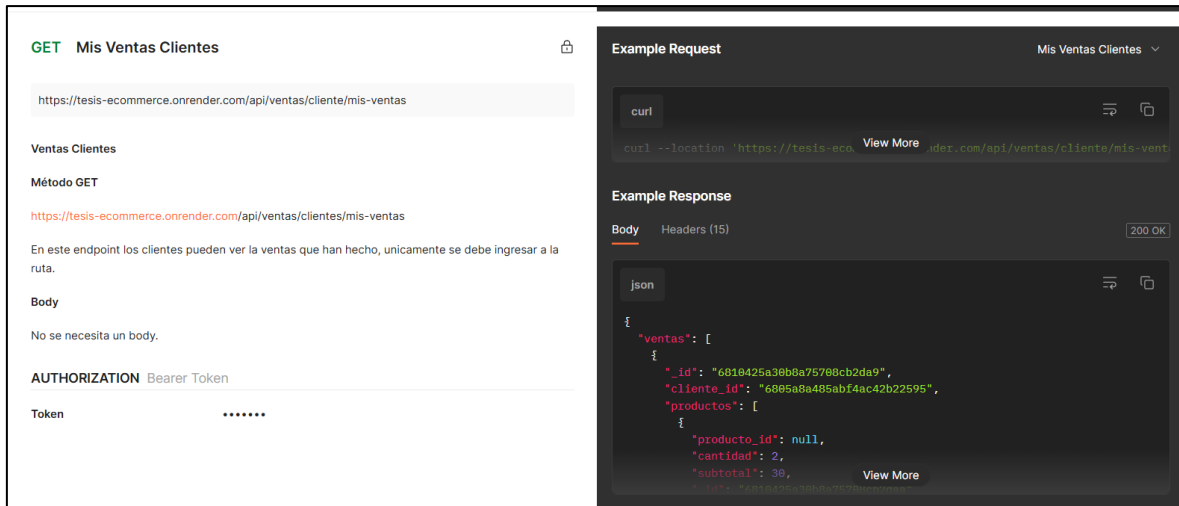


Figura 3.22 Solicitud de visualizar reporte de ventas del cliente.

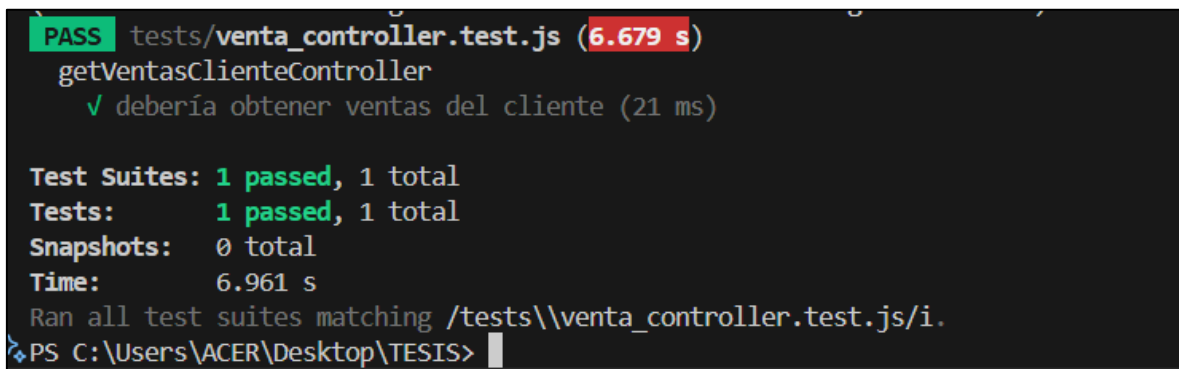


Figura 3.23 Prueba unitaria para visualizar reporte de ventas del cliente.

Sprint 3. Creación de modelo de IA.

El propósito de este Sprint es desarrollar e integrar un modelo de IA para personalizar productos artesanales (jabones y velas) según las elecciones del cliente. Por lo tanto, las actividades para esta etapa son las siguientes:

- Configuración del entorno de trabajo.
- Diseño de la arquitectura.
- Codificación e integración del modelo.
- Pruebas que permitan evaluar la precisión de las recomendaciones.
- Integración en los módulos del backend para su consumo.

Configuración del entorno de trabajo

Para el desarrollo del sistema de recomendación de productos personalizados, se ha configurado un entorno basado en Node.js, utilizando la API de Hugging Face con el modelo meta-llama/Llama-3.1-8B-Instruct, el cual es un modelo que permite crear prompts con la finalidad de generar texto [43]. En la **Figura 3.24** se ilustra la configuración de herramientas para esta integración.

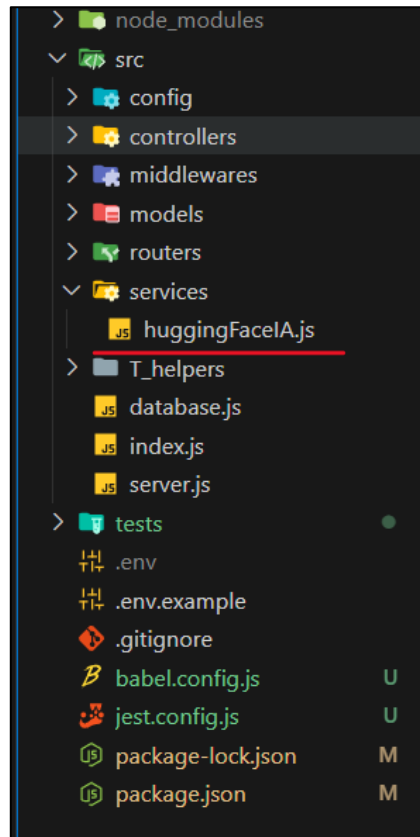


Figura 3.24 Configuración para recomendaciones personalizadas con IA.

Diseño de la arquitectura

En esta etapa se ha establecido una arquitectura modular que permite desacoplar la lógica de recomendación de IA del resto del backend. Además, esta estructura garantiza la escalabilidad y facilita su mantenimiento. Asimismo, toda la lógica del modelo de IA se encapsula en un controlador independiente, el cual puede ser invocado desde distintas rutas protegidas por autenticación, tal y como se evidencia en la **Figura 3.25** y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

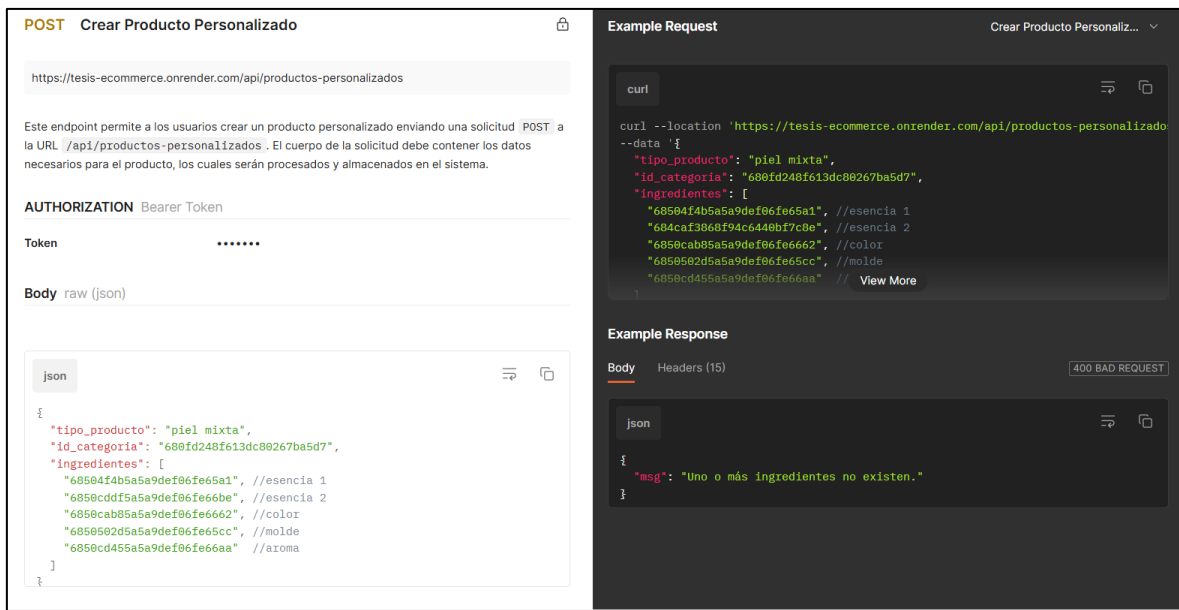


Figura 3.25 Arquitectura de IA dentro del backend para personalización de productos.

Codificación e integración del modelo

Con el objetivo de facilitar la experiencia de personalización de productos, se ha diseñado e implementado un módulo de IA se encarga de construir dinámicamente un prompt en lenguaje natural que se envía al modelo Llama 3.1 de Hugging Face. Este prompt está basado en los ingredientes disponibles en la base de datos, el historial de productos adquiridos por el cliente y las visitas que ha hecho a algunos productos. Además, la respuesta generada por la IA es parseada y validada para asegurar que los ingredientes sugeridos existan previamente y que se pueda calcular el precio final. En la **Figura 3.26** se visualiza el flujo de construcción del prompt y procesamiento de la respuesta y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

```

import Ventas from '../models/ventas.js';
import Ingrediente from '../models/ingredientes.js';
import Categoria from '../models/categorias.js';
import VistaProducto from '../models/vistaProducto.js';
import { HfInference } from "@huggingface/inference";

const hf = new HfInference(process.env.TOKEN_HUGGINGFACE);

async function recomendarProductoConHF(clienteId, tipo, id_categoria) {
  const categoria = await Categoria.findById(id_categoria);
  if (!categoria) throw new Error("La categoría indicada no existe.");

  const ingredientesDisponibles = await Ingrediente.find({ id_categoria: id_categoria });
  if (ingredientesDisponibles.length === 0) throw new Error("No hay ingredientes disponibles para esta categoría.");

  const ingredientesAgrupados = ingredientesDisponibles.reduce((acc, ing) => {
    const tipo = ing.tipo.toLowerCase();
    if (!acc[tipo]) acc[tipo] = [];
    acc[tipo].push(ing.nombre);
    return acc;
  }, {});

  const ventas = await Ventas.find({ cliente_id: clienteId, estado: "finalizado" })
    .populate({ path: "productos.producto_id", populate: { path: "id_categoria ingredientes" } });

  const productosComprados = ventas.flatMap(v => v.productos.map(p => p.producto_id))
    .filter(p => p?.id_categoria && p.id_categoria_id.equals(id_categoria));

  let productosBase = productosComprados;

```

Figura 3.26 Programación del modelo de recomendación con IA.

Pruebas que permitan evaluar la precisión de las recomendaciones

Durante esta fase, se han realizado múltiples pruebas con diferentes usuarios y categorías de productos para validar que las recomendaciones generadas por la IA correspondan a ingredientes existentes, y que no se propongan combinaciones inválidas. La **Figura 3.27** muestra un ejemplo de la vista previa que se ha generado dinámicamente según las opciones que se han seleccionado y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

POST Recomendacion IA

https://tesis-ecommerce.onrender.com/api/productos/recomendacion

Recomendación de Productos Personalizados

Este endpoint permite obtener recomendaciones de productos personalizados (como jabones o velas) según el tipo de piel del cliente y una categoría específica.

Parámetros del cuerpo de la solicitud (body)

Tipo e Id_categoria

AUTHORIZATION Bearer Token

Token:

Body raw (json)

```

{
  "tipo": "decorativa",
  "id_categoria": "6823a6c096655bcb4971062"
}

```

Example Request

```

curl --location 'https://tesis-ecommerce.onrender.com/api/productos/recomendacion' \
--data '{
  "tipo": "decorativa",
  "id_categoria": "6823a6c096655bcb4971062"
}'

```

Example Response

Body Headers (15) 200 OK

```

{
  "recomendacion": {
    "msg": "Producto personalizado creado exitosamente",
    "producto_personalizado": {
      "categoria": "velas artesanales",
      "tipo": "decorativa",
      "aroma": {
        "_id": "6850cca45a5a9def96fe669f",
        "nombre": "aroma lavanda",
        "imagen": "https://res.clo

```

Figura 3.27 Pruebas realizadas al modelo de recomendación con IA.

Integración en los módulos del backend para su consumo

Finalmente, durante esta fase, se ha integrado el modelo de recomendación de productos basado en IA dentro del backend, específicamente en el módulo de productos personalizados. Además, esta integración permite a los clientes autenticados crear un producto de dos formas una manualmente y la otra que es asistido por IA. La **Figura 3.28** muestra esta integración dentro del flujo de trabajo y en el respectivo **ANEXO III** se puede visualizar el funcionamiento completo de las rutas restantes.

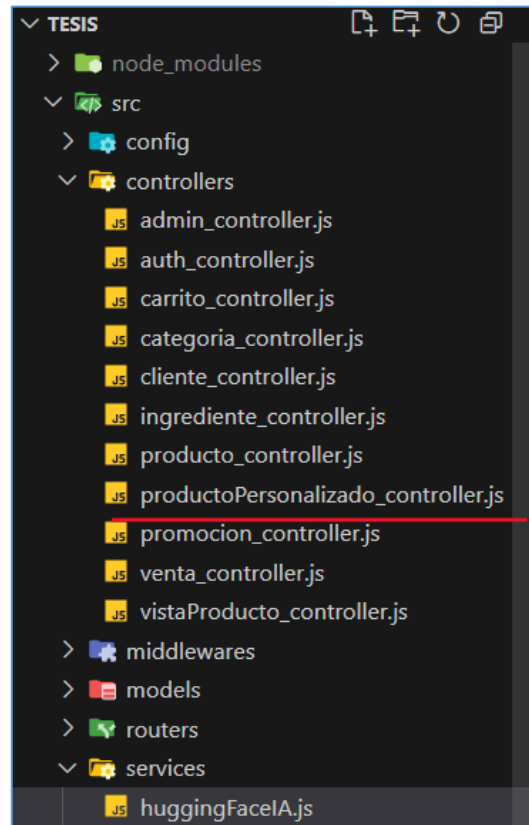


Figura 3.28 Integración del modelo de IA en los módulos backend de productos personalizados.

Sprint 4. Pruebas de Backend.

Esta etapa del Sprint tiene como propósito llevar a cabo pruebas que permitan verificar el funcionamiento correcto de los endpoints y para ello, se han llevado a cabo las tareas descritas a continuación:

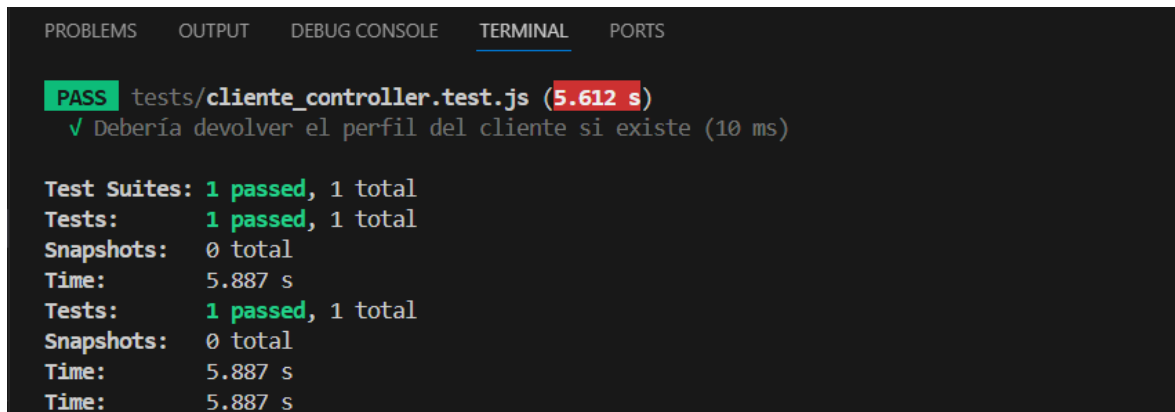
- Ejecución y resultados de pruebas unitarias.
- Ejecución y resultados de pruebas de rendimiento.
- Ejecución y resultados de pruebas de aceptación.

Ejecución y resultados de pruebas unitarias

Consisten en evaluar la porción más pequeña y funcional del código de forma individual. Además, estas pruebas contribuyen a asegurar la calidad del software y representan un componente esencial dentro del proceso de desarrollo [44]. En ese sentido, la **Figura 3.29** se presenta un ejemplo de ejecución de la prueba de ver perfil usando el Framework Jest, a su vez, en la **Figura 3.30** se ilustra la respuesta de dicha prueba y en el respectivo **ANEXO II** se encuentran los demás resultados.

```
184 ✓ test("Debería devolver el perfil del cliente si existe", async () => {
185   ✓ Clientes.findById.mockResolvedValue({
186     _id: "cliente-id",
187     cedula: "1234567890",
188     nombre: "Mario",
189     apellido: "Sánchez",
190     genero: "masculino",
191     email: "mario@correo.com",
192     direccion: "Quito",
193     telefono: "0999999999",
194     fecha_nacimiento: "1990-01-01",
195     imagen: "url-imagen"
196   });
197
198   const req = { clienteBDD: { _id: "cliente-id" } };
199   ✓ const res = {
200     status: jest.fn().mockReturnThis(),
201     json: jest.fn().mockReturnThis()
202   };
203
204   await getClienteProfile(req, res);
205
206   expect(Clientes.findById).toHaveBeenCalledWith("cliente-id");
207   expect(res.status).toHaveBeenCalledWith(200);
208   ✓ expect(res.json).toHaveBeenCalledWith({
209     msg: "Perfil obtenido correctamente",
210     ✓ cliente: expect.objectContaining({
211       nombre: "Mario",
212       apellido: "Sánchez",
213       genero: "masculino",
214     })
215   });
216   });
217
```

Figura 3.29 Comprobación de prueba unitaria.



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PASS tests/cliente_controller.test.js (5.612 s)
  ✓ Debería devolver el perfil del cliente si existe (10 ms)

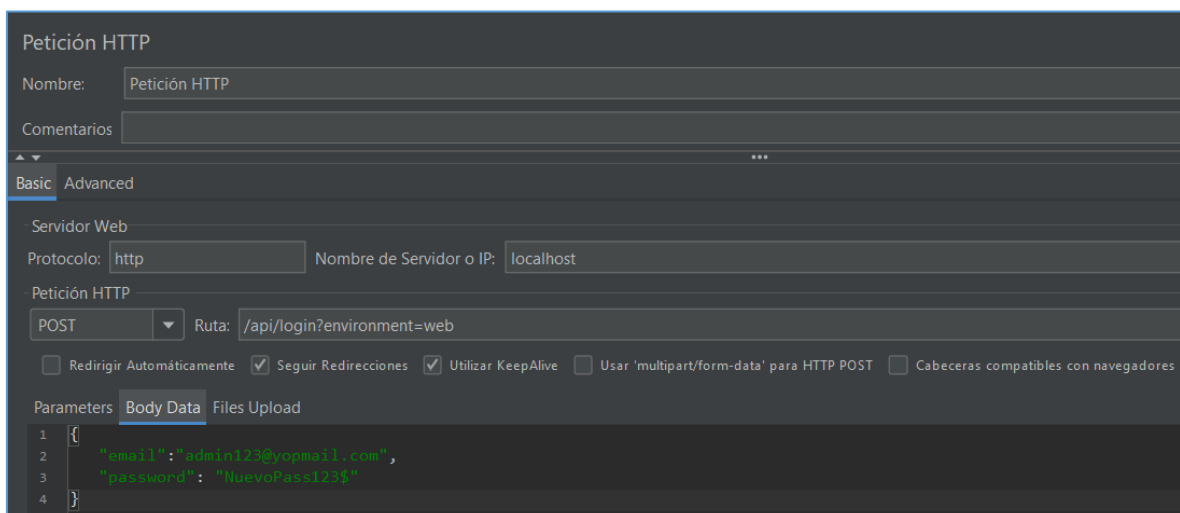
Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        5.887 s
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        5.887 s
Time:        5.887 s
```

Figura 3.30 Respuesta de prueba unitaria.

La ejecución de pruebas unitarias en cada uno de los endpoints que se han desarrollado ha permitido comprobar que cada una de las funcionalidades del backend funcione de manera independiente y correcta. Por otra parte, ha brindado ayuda temprana en cuanto a la detección de errores, lo que genera una mayor confianza en la estabilidad del componente, como también en la capacidad para ser modificado sin arriesgar su funcionamiento general.

Ejecución y resultados de pruebas de rendimiento

Corresponden a un tipo de evaluación centrada en medir la eficiencia del sistema, con el fin de dar una experiencia de usuario más eficiente y elevar la calidad general del producto. Además, en este tipo de validación se analiza variables clave como la velocidad de respuesta, la capacidad de mantener un funcionamiento estable y la respuesta del sistema frente a distintas demandas de procesamiento [45]. En ese sentido, la **Figura 3.31** presenta la solicitud a un endpoint usando la herramienta JMeter, mientras que, la **Figura 3.32** expone la respuesta y en el respectivo **ANEXO II** se encuentran los demás resultados.



Petición HTTP

Nombre: Petición HTTP

Comentarios

Basic Advanced

Servidor Web

Protocolo: http Nombre de Servidor o IP: localhost

Petición HTTP

POST Ruta: /api/login?environment=web

☐ Redirigir Automáticamente ☒ Seguir Redirecciones ☒ Utilizar KeepAlive ☐ Usar 'multipart/form-data' para HTTP POST ☐ Cabeceras compatibles con navegadores

Parameters Body Data Files Upload

```
1 {
2   "email": "admin123@yopmail.com",
3   "password": "NuevoPass123!"
4 }
```

Figura 3.31 Comprobación de prueba de rendimiento.

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Login-Administ...	100	2019	580	4159	839,78	0,00%	20,1/sec	9,89	5,27	503,0
Total	100	2019	580	4159	839,78	0,00%	20,1/sec	9,89	5,27	503,0

Figura 3.32 Respuesta de la prueba de rendimiento.

Estas pruebas han permitido observar el comportamiento del backend ante múltiples solicitudes concurrentes y como resultado final se ha demostrado que cada petición que se ha ejecutado ha respondido de forma satisfactoria, garantizando así, un rendimiento adecuado y una experiencia fluida.

Ejecución y resultados de pruebas de aceptación

Durante las pruebas de aceptación, la conformidad y aceptación del usuario constituye un elemento fundamental para evaluar el éxito de una aplicación software. Es así como, este tipo de pruebas se lleva a cabo en la fase final del ciclo de desarrollo [46]. Por esta razón, esta prueba es de gran importancia, dado que el cliente evalúa el backend lo cual permite establecer si el usuario está conforme con el producto que se ha elaborado. En la **Tabla 3.1** se visualiza un ejemplo de prueba de aceptación sobre un módulo determinado y en el respectivo **ANEXO II** se encuentran los demás resultados.

Tabla 3.1 Comprobación de prueba de aceptación.

PRUEBA DE ACEPTACIÓN DE FLOR&CERA	
Designación (ID): PA-001	Identificador de historia de Usuario: HU001
Nombre: Gestionar cuenta	
Descripción: El componente backend entrega varios endpoints que son utilizados por el administrador y clientes para: <ul style="list-style-type: none"> • Registrarse. • Iniciar y cerrar sesión. • Modificar perfil. • Recuperar Contraseña 	
Instrucciones de funcionamiento: Para registrarse, iniciar/cerrar sesión, modificar el perfil o recuperar la contraseña: <ul style="list-style-type: none"> • Acceder a la URL del componente backend en un navegador o cliente API 	

<p>(Postman).</p> <ul style="list-style-type: none"> • Dirigirse al endpoint correspondiente (Registro, Login, Modificar perfil, Recuperar contraseña). • Enviar las solicitudes HTTP con los parámetros requeridos: <ul style="list-style-type: none"> - Registro: datos personales (nombre, correo, contraseña, etc.). - Login: correo y contraseña. - Modificar perfil: parámetros específicos del usuario a actualizar. - Recuperar contraseña: correo del usuario registrado. • Seguir las respuestas proporcionadas por el backend para confirmar las acciones realizadas.
<p>Resultado deseado:</p> <p>El backend permite crear cuentas nuevas, iniciar sesión, modificar datos personales del usuario y recuperar contraseña.</p> <p>Cabe resaltar que los endpoints de registro y modificación de perfil, son estrictamente para uso del rol cliente, el usuario administrador no cuenta con estos endpoints para registrar o modificar perfil.</p>
<p>Evaluación:</p> <p>El cliente confirma que todas las funcionalidades operan correctamente y cumplen las expectativas.</p>

Estas pruebas han permitido validar que todas las funcionalidades que se han definido previamente hayan sido implementadas correctamente desde la perspectiva del usuario final. Además, una vez que se ha logrado tener la aprobación final del dueño del negocio se puede proceder a la siguiente y última iteración.

Sprint 5. Despliegue del Backend

Para culminar el proyecto, el Sprint final está orientado al despliegue de los endpoints que se han desarrollado en la plataforma Render. Además, este proceso abarca la transición completa del ambiente de desarrollo a uno de producción, garantizando su adecuada organización y disponibilidad para los destinatarios finales [47]. En este caso, se ha elegido Render por ser una plataforma accesible y confiable que permite implementar y publicar la API RESTful del backend de forma eficiente y la cual puede ser accesible mediante una URL pública.

<https://tesis-ecommerce.onrender.com>

Asimismo, es posible acceder a la documentación del backend mediante el siguiente enlace.

<https://documenter.getpostman.com/view/42480684/2sB2j1jDif>

Finalmente, en el **ANEXO II** se puede evidenciar que el administrador del negocio “Flor&Cera” ha emitido un certificado en el que confirma el cumplimiento del 100% de los requisitos y funcionalidades del componente backend.

4 CONCLUSIONES

Una vez que se ha finalizado el proyecto backend mediante la planificación y ejecución de Sprints con la ayuda de la metodología Scrum, se presentan a continuación las principales conclusiones.

- La definición y documentación de los requisitos ha permitido estructurar un backend orientado a satisfacer las necesidades del negocio y sobre todo a la personalización de productos mediante IA, con el objetivo ofrecer una experiencia más adaptada a las preferencias del cliente.
- La adopción de una base de datos NoSQL con un modelo de datos flexible ha permitido gestionar de manera eficiente información clave como productos, clientes, ingredientes, carrito de compras, etc. Además, este modelo de datos en conjunto con el patrón de arquitectura que se ha utilizado ha demostrado ser altamente adaptable, facilitando posibles ampliaciones futuras sin comprometer el rendimiento del backend.
- El desarrollo de rutas públicas y privadas en los distintos módulos del backend, ha asegurado una correcta separación de responsabilidades y ha restringido el acceso según el rol del usuario autenticado. Además, este enfoque ha incrementado la seguridad y ha permitido una integración coherente entre funcionalidades.
- El integrar un modelo de Inteligencia Artificial mediante el servicio Hugging Face, ha permitido generar recomendaciones personalizadas a los usuarios en función de sus preferencias y gustos. Además, esta funcionalidad permite que el negocio ofrezca un valor agregado relevante que otras plataformas no lo ofrecen.
- La ejecución de pruebas unitarias, de carga y de aceptación han sido fundamentales para asegurar el correcto funcionamiento de los endpoints bajo diversas condiciones. Además, este enfoque ha permitido obtener respuestas coherentes y seguras, además de facilitar la identificación y solución oportuna de posibles errores.
- El backend ha sido implementado con éxito en un entorno de producción, lo que ha posibilitado el acceso desde varios medios tales como clientes Rest, plataformas web y móviles. Además, esto garantiza su disponibilidad, escalabilidad y una integración eficiente con otros módulos del ecosistema de comercio electrónico.

5 RECOMENDACIONES

Una vez que se ha finalizado el proyecto backend mediante la planificación y ejecución de Sprints con la ayuda de la metodología Scrum, se presentan a continuación las principales recomendaciones fundamentadas en la experiencia que se ha obtenido durante el proyecto.

- Se recomienda mantener actualizadas las dependencias de Node.js, Express y otras librerías externas como Mongoose, Multer, JWT o el SDK de Hugging Face, con el fin de aprovechar mejoras en el rendimiento, nuevas funcionalidades y actualizaciones de seguridad.
- Considerando la integración de un modelo de IA desde Hugging Face, es aconsejable llevar un registro del historial de versiones, documentar adecuadamente el prompt y realizar evaluaciones periódicas en caso de modificaciones o actualizaciones.
- Para aumentar la precisión de las recomendaciones generadas por el modelo de Hugging Face, se aconseja adaptar el prompt o los datos de entrada en función del comportamiento más reciente de los clientes, incorporando información sobre nuevos productos, ingredientes y patrones de compra que hayan sido detectados.
- Con el fin de preservar la calidad del backend a largo plazo, es recomendable seguir implementando pruebas unitarias en cada uno de los endpoints, especialmente aquellos relacionados con la personalización de productos, recomendaciones basadas en IA y gestión del carrito de compras ya que esto permite identificar errores de manera temprana y agilizar futuras mejoras o ajustes.
- Es aconsejable tener precaución al utilizar modelos de Hugging Face, ya que algunas versiones pueden no estar operativas debido a que sus desarrolladores no las han desplegado o han sido descontinuadas. Además, se recomienda optar por modelos en inglés, ya que suelen estar mejor mantenidos y presentan un funcionamiento más estable.

6 REFERENCIAS BIBLIOGRÁFICAS

- [1] J. Pazto, Interviewee, *Encuesta de productos artesanales*. [Entrevista]. 10 Marzo 2025.
- [2] Seidor, «Modelos de IA: cómo entrenar, validar, ajustar e implementar,» 03 Julio 2024. [En línea]. Available: <https://www.seidor.com/es-es/blog/modelos-ia-como-entrenar-validar-ajustar-implementar>. [Último acceso: 11 Marzo 2025].
- [3] MongoDB, «¿Qué es MongoDB?,» [En línea]. Available: <https://www.mongodb.com/es/company/what-is-mongodb>. [Último acceso: 01 Abril 2025].
- [4] A. W. S. (AWS), «¿Qué es JavaScript (JS)?,» [En línea]. Available: <https://aws.amazon.com/es/what-is/javascript/#:~:text=JavaScript%20es%20un%20lenguaje%20de,usuario%20de%20un%20sitio%20web..> [Último acceso: 11 Marzo 2025].
- [5] A. W. S. (AWS), «¿Qué es una API RESTful?,» [En línea]. Available: <https://aws.amazon.com/es/what-is/restful-api/>. [Último acceso: 11 Marzo 2025].
- [6] nodejs, «Introduction to Node.js,» [En línea]. Available: <https://nodejs.org/es/learn/getting-started/introduction-to-nodejs>. [Último acceso: 11 Marzo 2025].
- [7] Kinsta, «¿Qué es Express.js? Todo lo que Debes Saber,» 05 Marzo 2025. [En línea]. Available: <https://kinsta.com/es/base-de-conocimiento/que-es-express/>. [Último acceso: 11 Marzo 2025].
- [8] hostinger, «¿Qué es JSON?,» 10 Enero 2023. [En línea]. Available: <https://www.hostinger.com/es/tutoriales/que-es-json>. [Último acceso: 11 Marzo 2025].
- [9] L. M. Lopez, «Qué es Json Web Token y cómo funciona,» OpenWebinars, 17 Enero 2020. [En línea]. Available: <https://openwebinars.net/blog/que-es-json-web-token-y-como-funciona/> . [Último acceso: 11 Marzo 2025].
- [10] A. Reinman, «Nodemailer,» nodemailer, [En línea]. Available: <https://nodemailer.com/> . [Último acceso: 11 Marzo 2025].
- [11] formadoresit, «¿Qué es Postman? ¿Cuáles son sus principales ventajas?,» 24 Agosto 2023. [En línea]. Available: <https://formadoresit.es/que-es-postman-cuales-son-sus-principales-ventajas/>. [Último acceso: 11 Marzo 2025].
- [12] faztweb, «Despliegue de Nodejs en Render.com,» 26 Junio 2023. [En línea]. Available: <https://fazitweb.com/contenido/nodejs-deploy-en-render-com>. [Último acceso: 02 Mayo 2025].
- [13] QuestionPro, «¿Qué es un estudio de caso y cómo realizarlo?,» [En línea]. Available: <https://www.questionpro.com/blog/es/que-es-un-estudio-de-caso/>. [Último acceso: 01 Abril 2025].

- [14] Valtx, «Metodologías de desarrollo de software: ¿Qué son y para qué sirven?,» [En línea]. Available: <https://www.valtx.pe/blog/metodologias-para-el-desarrollo-de-software-que-son-y-para-que-sirven>. [Último acceso: 01 Abril 2025].
- [15] FundacionExit, «Metodología Agile en ONGs,» 16 Abril 2024. [En línea]. Available: https://fundacionexit.org/metodologia-agile-en-ongs/?gad_source=1&gclid=Cj0KCQjwna6_BhCbARIsALId2Z31rGAZktlzEWsqqmiHm3ArdHS7MIlkaEF0zUHBbQeaS6ZL6ontU-EaAhzwEALw_wcB. [Último acceso: 01 Abril 2025].
- [16] A. W. S. (AWS), «¿En qué consiste Scrum?,» [En línea]. Available: <https://aws.amazon.com/es/what-is/scrum/>. [Último acceso: 01 Abril 2025].
- [17] gcfglobal, «Roles en Scrum,» [En línea]. Available: <https://edu.gcfglobal.org/es/scrum/roles-en-scrum/1/>. [Último acceso: 01 Abril 2025].
- [18] ServiceTonic, «Qué es y cuáles son las funciones del Product Owner,» 2022. [En línea]. Available: <https://donetonic.com/es/funciones-de-un-product-owner/>. [Último acceso: 01 Abril 2025].
- [19] A. Raeburn, «¿Qué es un Scrum Master y cuál es su función?,» asana, 02 Febrero 2025. [En línea]. Available: <https://asana.com/es/resources/scrum-master>. [Último acceso: 01 Marzo 2025].
- [20] Coursera, «What Is a Development Team?,» 14 Marzo 2025. [En línea]. Available: <https://www.coursera.org/articles/what-is-development-team>. [Último acceso: 01 Abril 2025].
- [21] miro, «Artefactos Scrum,» [En línea]. Available: <https://miro.com/es/agile/que-son-artefactos-scrum/>. [Último acceso: 01 Abril 2025].
- [22] O. Garcia, «Recopilación de requisitos,» proyectum, 01 Mayo 2013. [En línea]. Available: <https://proyectum.com/sistema/blog/recopilacion-de-requisitos/>. [Último acceso: 01 Abril 2025].
- [23] miro, «Cómo escribir buenas historias de usuario en el método Agile,» [En línea]. Available: <https://miro.com/es/agile/que-es-historia-usuario/>. [Último acceso: 01 Abril 2025].
- [24] D. Radigan, «Backlog del producto: consejos para crear y priorizar,» Atlassian, [En línea]. Available: <https://www.atlassian.com/es/agile/scrum/backlogs>. [Último acceso: 01 Abril 2025].
- [25] ServiceTonic, «Product Backlog y Sprint Backlog,» [En línea]. Available: https://donetonic.com/es/product-backlog-y-sprint-backlog/#Definicion_de_Product_Backlog. [Último acceso: 01 Abril 2025].
- [26] P. Huet, «Arquitectura de software: Qué es y qué tipos existen,» OpenWebinars, 24 Agosto 2022. [En línea]. Available: <https://openwebinars.net/blog/arquitectura-de-software-que-es-y-que-tipos-existen/>. [Último acceso: 01 Abril 2025].

- [27] A. W. S. (AWS), «¿Qué son las herramientas para desarrolladores?,» [En línea]. Available: <https://aws.amazon.com/es/what-is/developer-tools/>. [Último acceso: 01 Abril 2025].
- [28] D. d. GitHub, «Acerca de GitHub y Git,» [En línea]. Available: <https://docs.github.com/es/get-started/start-your-journey/about-github-and-git>. [Último acceso: 01 Abril 2025].
- [29] G. Cimas, «Visual Studio Code: Editor de código para desarrolladores,» OpenWebinars, 22 Julio 2022. [En línea]. Available: <https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/>. [Último acceso: 01 Abril 2025].
- [30] radas, «Qué es Postman y primeros pasos,» OpenWebinars, 03 Junio 2019. [En línea]. Available: <https://openwebinars.net/blog/que-es-postman/>. [Último acceso: 01 Abril 2025].
- [31] IfGeekThen, «Qué es Express.JS y primeros pasos,» 09 Marzo 2021. [En línea]. Available: <https://ifgeekthen.nttdata.com/s/post/que-es-expressjs-y-primeros-pasos-MCCIDTDOZFGQBNXGDI5WENXXNNY4?language=es>. [Último acceso: 01 Abril 2025].
- [32] Render, «Web Services,» [En línea]. Available: <https://render.com/docs/web-services>. [Último acceso: 01 Abril 2025].
- [33] A. Robledano, «Qué es MongoDB,» OpenWebinars, 28 Octubre 2019. [En línea]. Available: <https://openwebinars.net/blog/que-es-mongodb/>. [Último acceso: 01 Abril 2025].
- [34] U. -. U. I. d. L. Rioja, «¿Qué son las librerías en programación y para qué sirven?,» [En línea]. Available: <https://www.unir.net/revista/ingenieria/librerias-programacion/>. [Último acceso: 01 Abril 2025].
- [35] K. Inc, «Bcrypt,» [En línea]. Available: https://www.vpnunlimited.com/es/help/cybersecurity/bcrypt?srsId=AfmBOooBnu9LNelm3QY45UnRdlzBIHtyusLYrlGSzzJnjetrvUjj2x_b. [Último acceso: 01 Abril 2025].
- [36] A. W. S. (AWS), «¿Qué es el CORS?,» [En línea]. Available: <https://aws.amazon.com/es/what-is/cross-origin-resource-sharing/>. [Último acceso: 01 Abril 2025].
- [37] E. Arias, «Dotenv: variables de entorno Node js,» 01 Febrero 2023. [En línea]. Available: <https://eduardo-arias.com/dotenv-variables-de-entorno-node-js/>. [Último acceso: 01 Abril 2025].
- [38] A. Casero, «¿Cómo funciona el Nodemailer de Node.js?,» 30 Septiembre 2024. [En línea]. Available: <https://keepcoding.io/blog/como-funciona-el-nodemailer-de-node-js/>. [Último acceso: 01 Abril 2025].
- [39] codigofacilito, «Qué es mongoose,» [En línea]. Available: <https://codigofacilito.com/articulos/que-es-mongoose>. [Último acceso: 01 Abril 2025].

- [40] I. Corporation, «JSON Web Token (JWT),» [En línea]. Available: <https://www.ibm.com/docs/es/cics-ts/6.1.0?topic=cics-json-web-token-jwt>. [Último acceso: 01 Abril 2025].
- [41] F. García, «¿Qué es Visual Studio Code y cuáles son sus ventajas?,» arsys, 30 Octubre 2024. [En línea]. Available: <https://www.arsys.es/blog/que-es-visual-studio-code-y-cuales-son-sus-ventajas>. [Último acceso: 27 Mayo 2025].
- [42] IBM, «¿Qué es MongoDB?,» [En línea]. Available: <https://www.ibm.com/mx-es/topics/mongodb>. [Último acceso: 27 Mayo 2025].
- [43] S. Navarro, «¿Qué es Hugging Face?,» KeepCoding, 11 Abril 2025. [En línea]. Available: https://keepcoding.io/blog/que-es-hugging-face/#%C2%BFQue_es_Hugging_Face. [Último acceso: 27 Mayo 2025].
- [44] A. W. S. (AWS), «¿Qué son las pruebas unitarias?,» [En línea]. Available: <https://aws.amazon.com/es/what-is/unit-testing/>. [Último acceso: 02 Mayo 2025].
- [45] T. It, «Pruebas de Performance Testing: qué son, tipos y su importancia,» 28 Mayo 2024. [En línea]. Available: <https://www.testingit.com.mx/blog/pruebas-de-performance-testing>. [Último acceso: 02 Mayo 2025].
- [46] T. It, «Pruebas de aceptación de software, ¿Cuándo y por qué son necesarias?,» 23 Agosto 2022. [En línea]. Available: <https://www.testingit.com.mx/blog/pruebas-acceptacion-software>. [Último acceso: 02 Mayo 2025].
- [47] M. Wrobel, «¿Qué es el despliegue de software? Definición, alcance y buenas prácticas,» Invgate, 19 Junio 2023. [En línea]. Available: <https://blog.invgate.com/es/despliegue-de-software>. [Último acceso: 02 Mayo 2025].

7 ANEXOS

A continuación, se presenta cada uno de los Anexos que se ha utilizado para el desarrollo del backend, los cuales se encuentran detallados de la siguiente manera:

ANEXO I. Resultado del programa anti-plagio Turnitin.

ANEXO II. Información extra del componente backend.

ANEXO III. Manual de usuario (video).

ANEXO IV. Credenciales de acceso y despliegue.

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 antiplagio Turnitin.

F_AA_236

CERTIFICADO DE ORIGINALIDAD TRABAJO DE INTEGRACIÓN CURRICULAR

Quito, D.M. 19 de julio de 2025

De mi consideración:

Yo, Loarte Cajamarca Byron Gustavo, en calidad de Director del Trabajo de Integración Curricular titulado DESARROLLO DE UN BACKEND asociado al DESARROLLO DE UN E-COMMERCE PARA LA VENTA DE PRODUCTOS ARTESANALES PERSONALIZADOS BASADO EN IA elaborado por la estudiante ESTEFANÍA MELISA SÁNCHEZ PÁRRAGA de la carrera en DESARROLLO DE SOFTWARE, certifico que he empleado la herramienta antiplagio "TURNITIN" para la revisión de originalidad del documento escrito producto del Trabajo de Integración Curricular indicado.

- El documento escrito tiene un índice de similitud del 07%.

Respecto al uso de herramientas de Inteligencia Artificial en el desarrollo del Trabajo de Integración Curricular, se certifica:

- El documento escrito tiene un porcentaje de uso de herramientas de Inteligencia Artificial menor al 20%.

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

Todos los requisitos que se han establecido previamente al desarrollo del proyecto se encuentran registrados en la **Tabla 1** .

Tabla 1 Levantamiento de requerimientos.

RECOPIACIÓN DE REQUERIMIENTOS		
TIPO DE SISTEMA	ID - RR	ENUNCIADO DEL ITEM
BACKEND	RR-002	Para el usuario administrador se requiere generar endpoints para: <ul style="list-style-type: none">• Gestionar usuarios.• Gestionar categorías y productos.• Gestionar ventas.
	RR-003	Para el usuario administrador se requiere generar endpoints para: <ul style="list-style-type: none">• Visualizar reporte de ventas.
	RR-004	Para el usuario cliente se requiere generar endpoints para: <ul style="list-style-type: none">• Visualizar categorías y productos.• Gestionar personalización de productos.• Gestionar carrito de compras.
	RR-005	Para el usuario cliente se requiere generar endpoints para: <ul style="list-style-type: none">• Visualizar reporte de ventas.• Visualizar reporte de facturas.

Historias de Usuario

Las Historias de Usuario se encuentran detalladas a continuación, las mismas que han sido creadas una vez que se ha finalizado con la fase de levantamiento de requerimientos. De esta manera, se han realizado 6 Historias de Usuario, las cuales se detallan a continuación, desde la **Tabla 2** hasta la **Tabla 7** .

Tabla 2 Historia de usuario para gestionar usuarios, categorías, productos y ventas.

HISTORIA DE USUARIO	
Identificador: HU-002	Usuario: Administrador
Nombre historia: Gestionar usuarios, categorías, productos y ventas	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Iteración asignada: 2	
Responsable (s): Estefanía Sánchez	
Descripción: El componente backend entrega varios endpoints que son utilizados por el administrador para: <ul style="list-style-type: none"> • Gestionar usuarios. • Gestionar categorías y productos. • Gestionar ventas. 	
Observación: Todos los endpoints mencionados en la descripción son accesibles para el rol administrador. Además, se garantiza que únicamente el administrador pueda gestionar la información respecto a usuarios, categorías, productos y ventas.	

Tabla 3 Historia de usuario para visualizar reporte de ventas.

HISTORIA DE USUARIO	
Identificador: HU-003	Usuario: Administrador
Nombre historia: Visualizar reporte de ventas	
Prioridad en negocio: Alta	Riesgo en desarrollo: Media
Iteración asignada:	
Responsable (s): Estefanía Sánchez	
Descripción: El componente backend entrega varios endpoints que son utilizados por el administrador para: <ul style="list-style-type: none"> • Visualizar reporte de ventas. 	
Observación: Todos los endpoints mencionados en la descripción son accesibles para el rol administrador. Además, se garantiza que únicamente el administrador pueda gestionar los reportes de ventas.	

Tabla 4 Historia de usuario para visualizar categorías y productos.

HISTORIA DE USUARIO	
Identificador: HU-004	Usuario: Cliente
Nombre historia: Visualizar categorías y productos	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Iteración asignada: 3	
Responsable (s): Estefanía Sánchez	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> • Visualizar categorías y productos. 	
Observación: El endpoint mencionado en la descripción es accesible únicamente para los usuarios con rol cliente.	

Tabla 5 Historia de usuario para gestionar personalización de productos.

HISTORIA DE USUARIO	
Identificador: HU-005	Usuario: Cliente
Nombre historia: Gestionar personalización de productos	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Iteración asignada: 3	
Responsable (s): Estefanía Sánchez	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> • Gestionar personalización de productos. • Personalización de productos mediante IA. 	
Observación: El endpoint mencionado en la descripción es accesible únicamente para los usuarios con rol cliente.	

Tabla 6 Historia de usuario para gestionar carrito de compras.

HISTORIA DE USUARIO	
Identificador: HU-006	Usuario: Cliente
Nombre historia: Gestionar carrito de compras	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Iteración asignada: 3	
Responsable (s): Estefanía Sánchez	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> • Gestionar carrito de compras. 	
Observación: El endpoint mencionado en la descripción es accesible únicamente para los usuarios con rol cliente.	

Tabla 7 Historia de usuario para visualizar reportes.

HISTORIA DE USUARIO	
Identificador: HU-007	Usuario: Cliente
Nombre historia: Visualizar reportes	
Prioridad en negocio: Alta	Riesgo en desarrollo: Media
Iteración asignada: 3	
Responsable (s): Estefanía Sánchez	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> • Visualizar reporte de ventas. • Visualizar reporte de facturas. 	
Observación: Todos los endpoints mencionados en la descripción son accesibles únicamente para los usuarios con rol cliente.	

Product Backlog

Cada módulo tiene una prioridad, dependiendo del impacto del mismo dentro del componente, de esta manera en la **Tabla 8** se encuentra el ciclo según la prioridad.

Tabla 8 Product Backlog.

PRODUCT BACKLOG				
ID-HU	DESCRIPCIÓN DE LA HISTORIA	ITERACION ASIGNADA	ESTADO	PRIORIDAD
HU-002	Gestionar usuarios, categorías, productos y ventas	2	Finalizada	Media
HU-003	Visualizar reporte de ventas	2	Finalizada	Baja
HU-004	Visualizar categorías y productos	3	Finalizada	Alta
HU-005	Gestionar personalización de productos	3	Finalizada	Media
HU-006	Gestionar carrito de compras	3	Finalizada	Alta
HU-007	Visualizar reportes	3	Finalizada	Media

Sprint Backlog

Para el desarrollo completo del proyecto se han realizado en 6 Sprints, los cuales se encuentran delimitados en la **Tabla 9**.

Tabla 9 Sprint Backlog.

ELABORACIÓN DEL SPRINT BACKLOG						
ID-SB	NOMBRE	MÓDULO	HU	HISTORIAS DE USURIO	TAREAS	TIEMPO ESTIMADO
SB-000	Configuración del ambiente de desarrollo	N/A	N/A	N/A	<ul style="list-style-type: none"> Recopilación y definición de requerimientos Diseño y creación de la Base de datos Diseño de arquitectura RESTful para el Backend en base a los requerimientos. 	80H
SB-002	Diseño y codificación de endpoints para el usuario administrador	Módulo Gestión Usuarios	HU002	Gestionar usuarios, categorías, productos y ventas	<ul style="list-style-type: none"> Modelamiento e implementación de endpoints para la gestión de usuarios, categorías y productos. Implementación de la gestión de ventas. 	20H
		Módulo Categorías				
		Módulo Productos				

		Módulo Ventas			<ul style="list-style-type: none"> Validación de datos ingresados por el administrador. Registro y consulta de información en la base de datos. 	
		Módulo Reporte de Ventas	HU003	Visualizar reporte de ventas	<ul style="list-style-type: none"> Diseño e implementación de endpoints para la consulta de reportes de ventas. Validación de acceso y restricciones de seguridad. Generación de reportes con datos relevantes. 	
SB-003	Diseño e implementación de endpoints para el cliente	Módulo Catálogo	HU004	Visualizar categorías y productos	<ul style="list-style-type: none"> Diseño e implementación de endpoints para la visualización de categorías y productos. Consulta de productos y categorías en la base de datos. 	20H

					<ul style="list-style-type: none"> • Implementación de filtros y búsquedas para facilitar la selección de productos. • Verificación de la integridad de los datos presentados al usuario. 	
		Módulo Personalización de Productos	HU005	Gestionar personalización de productos	<ul style="list-style-type: none"> • Implementación de endpoints para la personalización de productos mediante IA. • Integración del modelo de IA con el sistema de personalización. • Validación de los datos de personalización enviados por el usuario. • Almacenamiento de configuraciones personalizadas en la base de datos. 	
		Módulo Carrito de Compras	HU006	Gestionar carrito de compras	<ul style="list-style-type: none"> • Implementación de endpoints para agregar, modificar y eliminar productos del carrito. 	

					<ul style="list-style-type: none"> Validación de disponibilidad y precios de los productos en el carrito. Registro y actualización del carrito en la base de datos. 	
		Módulo Reportes	HU007	Visualizar reportes	<ul style="list-style-type: none"> Implementación de endpoints para la generación y consulta de reportes de compras. Consulta y visualización de facturas y pedidos anteriores. Generación de reportes detallados sobre el historial de compras del usuario. 	
SB-004	Implementación de Inteligencia Artificial	<ul style="list-style-type: none"> Configuración de las herramientas necesarias para desarrollar la IA. Desarrollar la arquitectura para la integración de la IA. Programación del modelo de IA. Pruebas del modelo para identificar posibles conflictos. Integración en los módulos respectivos. 				20H
SB-005	Pruebas del Backend	<ul style="list-style-type: none"> Pruebas unitarias. 				20H

		<ul style="list-style-type: none"> • Pruebas de rendimiento. • Pruebas de aceptación. 	
SB-006	Despliegue del Backend	<ul style="list-style-type: none"> • Despliegue del backend en Render. 	20H
	Documentación	<ul style="list-style-type: none"> • Trabajo de Integración Curricular. • Anexos. 	40H
TOTAL			240H

Elaboración de documentos por cada colección

A continuación, se presentan los documentos por cada colección los cuales van desde la **Figura 1** hasta la **Figura 7**.

```
_id: ObjectId('6843c51fbbc5566b3f2b1e41')
email: "admin123@yopmail.com"
password: "$2b$10$D4uP/10NjWtLKuwINFHBouxYYgAFxFm71fwMuAn2ZmGIikQt5maLm"
token: null
confirmEmail: true
codigoRecuperacion: null
codigoRecuperacionExpires: null
createdAt: 2025-06-07T04:50:39.843+00:00
updatedAt: 2025-06-17T20:59:37.322+00:00
__v: 0
```

Figura 1 Documento para la colección Administrador.

```
_id: ObjectId('6837848b13ee6fd51f765fe5')
nombre: "Estefania"
apellido: "Parraga"
genero: "Femenino"
email: "estef123@yopmail.com"
codigoRecuperacion: "324309"
codigoRecuperacionExpires: 2025-06-17T21:43:42.145+00:00
password: "$2b$10$TdUqpoYISlRHgJuBQ4YNc.M04.VWe0Jtoq.akyb0jGbQ3eRn6Zera"
token: null
confirmEmail: true
estado: "activo"
createdAt: 2025-05-28T21:47:55.246+00:00
updatedAt: 2025-06-17T21:39:42.145+00:00
__v: 0
fecha_nacimiento: 1995-04-15T00:00:00.000+00:00
cedula: "1763543526"
direccion: "America Av, Pichincha, Quito"
telefono: "0978897657"
```

Figura 2 Documento para la colección Cliente.

```
_id: ObjectId('680fd248f613dc80267ba5d7')
nombre: "Jabones artesanales"
descripcion: "Descubre la suavidad y el cuidado de nuestros jabones artesanales elab..."
imagen: "https://res.cloudinary.com/ddg5fu4yt/image/upload/v1746045048/categori..."
imagen_id: "categorias/ztfuuvptlgbgjduyluo0"
createdAt: 2025-04-28T19:08:56.437+00:00
updatedAt: 2025-04-30T20:30:49.867+00:00
__v: 0
```

Figura 3 Documento para la colección Categorías.

```

_id: ObjectId('684cb8441ef9e158fa483124')
nombre: "Vela Relax Zen"
descripcion: "Una vela pensada para inducir una profunda sensación de calma y descan..."
▼ beneficios: Array (3)
  0: "Reduce el estrés y la ansiedad "
  1: "Mejora el estado de ánimo "
  2: "Promueve un ambiente de tranquilidad "
▼ ingredientes: Array (3)
  0: ObjectId('684caf3868f94c6440bf7c8e')
  1: ObjectId('684cb05b68f94c6440bf7ca2')
  2: ObjectId('684cb03968f94c6440bf7c9e')
aroma: "floral"
tipo: "decorativa"
precio: 10
imagen: "https://res.cloudinary.com/ddg5fu4yt/image/upload/v1749858368/producto..."
imagen_id: "productos/xlshk1yx5cqwy1m1ff8n"
stock: 14
descuento: 0
id_categoria: ObjectId('6823a6c096655bcbe4971062')
activo: true
createdAt: 2025-06-13T23:46:12.164+00:00
updatedAt: 2025-06-14T02:19:19.051+00:00
__v: 0

```

Figura 4 Documento para la colección Productos.

```

_id: ObjectId('6837850513ee6fd51f765ff2')
cliente_id: ObjectId('6837850513ee6fd51f765ff0')
▶ productos: Array (empty)
total: 0
estado: "pendiente"
fecha_creacion: 2025-05-28T21:49:57.363+00:00
createdAt: 2025-05-28T21:49:57.364+00:00
updatedAt: 2025-05-28T21:49:57.364+00:00
__v: 0

```

Figura 5 Documento para la colección Carritos.

```

_id: ObjectId('684cdc27d482126de8a8e5cd')
cliente_id: ObjectId('6837848b13ee6fd51f765fe5')
▼ productos: Array (1)
  ▶ 0: Object
total: 10
estado: "finalizado"
fecha_venta: 2025-06-14T02:19:19.186+00:00
createdAt: 2025-06-14T02:19:19.186+00:00
updatedAt: 2025-06-14T02:19:19.186+00:00
__v: 0

```

Figura 6 Documento para la colección Ventas.

```

    _id: ObjectId('684cdf085a5a9def06fe575a')
    cliente_id: ObjectId('6837848b13ee6fd51f765fe5')
    ▼ ingredientes: Array (5)
      0: ObjectId('6850502d5a5a9def06fe65cc')
      1: ObjectId('6850cab85a5a9def06fe6662')
      2: ObjectId('68504f4b5a5a9def06fe65a1')
      3: ObjectId('6850cddf5a5a9def06fe66be')
      4: ObjectId('6850cd8c5a5a9def06fe66b4')
    id_categoria: ObjectId('680fd248f613dc80267ba5d7')
    precio: 12
    aroma: "aroma de romero"
    tipo_producto: "piel mixta"
    createdAt: 2025-06-14T02:31:36.173+00:00
    updatedAt: 2025-06-17T22:12:54.940+00:00
    __v: 2
    imagen: "https://res.cloudinary.com/ddg5fu4yt/image/upload/v1750196947/producto..."
    imagen_id: "productos-personalizados/z81lurv7yx4lmetec0si"

```

Figura 7 Documento para la colección Productos personalizados.

Pruebas

Al concluir la etapa de desarrollo, se procede a realizar pruebas pertinentes para asegurar la calidad del backend.

Pruebas unitarias

A continuación, se presentan las pruebas unitarias restantes las cuales van desde la **Figura 8** hasta la **Figura 27**.

Cambio de contraseña Administrador

```

158 describe("Pruebas Unitarias - Admin - Cambiar Contraseña", () => {
167   beforeEach(() => {
178     res = {
179       status: jest.fn().mockReturnThis(),
180       json: jest.fn().mockReturnThis(),
181     };
182
183     Admin.findOne.mockResolvedValue(adminMock);
184     bcrypt.genSalt.mockResolvedValue("mockSalt");
185     bcrypt.hash.mockResolvedValue("hashedPassword");
186   });
187
188   afterEach(() => {
189     jest.clearAllMocks();
190   });
191
192   test("Debería cambiar la contraseña con éxito", async () => {
193     await cambiarContraseñaController(req, res);
194
195     expect(Admin.findOne).toHaveBeenCalledWith({ email: "admin123@yopmail.com" });
196     expect(bcrypt.hash).toHaveBeenCalledWith("NuevoPass123$", "mockSalt");
197     expect(adminMock.save).toHaveBeenCalled();
198     expect(res.status).toHaveBeenCalledWith(200);
199     expect(res.json).toHaveBeenCalledWith({ msg: "Contraseña cambiada con éxito" });
200   });
201 }

```

Figura 8 Petición de endpoint para cambio de contraseña administrador.

```
PASS tests/admin_controller.test.js (7.914 s)
  Pruebas Unitarias - Admin - Cambiar Contraseña
    ✓ Debería cambiar la contraseña con éxito (28 ms)
    ✓ Debería devolver error si algún campo falta (3 ms)
  Pruebas Unitarias - Admin - Cambiar Contraseña
    ✓ Debería cambiar la contraseña con éxito (28 ms)
    ✓ Debería devolver error si algún campo falta (3 ms)
    ✓ Debería cambiar la contraseña con éxito (28 ms)
    ✓ Debería devolver error si algún campo falta (3 ms)
    ✓ Debería devolver error si código es inválido (4 ms)
    ✓ Debería devolver error si el admin no existe (4 ms)

Test Suites: 1 passed, 1 total
Tests:       4 passed, 4 total
```

Figura 9 Prueba unitaria de endpoint para cambio de contraseña administrador.

Visualizar categoría

```
41 describe("getCategoriaByIDController", () => {
42   test("ID válido y categoría encontrada", async () => {
43     req.params.id = new mongoose.Types.ObjectId().toString();
44     Categoria.findById.mockResolvedValue({ nombre: "Velas" });
45
46     await getCategoriaByIDController(req, res);
47
48     expect(res.status).toHaveBeenCalledWith(200);
49     expect(res.json).toHaveBeenCalledWith({ categoria: { nombre: "Velas" } });
50   });
51 });
```

Figura 10 Petición de endpoint para visualizar categoría.

```
PASS tests/categoria_controller.test.js (7.291 s)
  getCategoriaByIDController
    ✓ ID válido y categoría encontrada (23 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        7.527 s
Ran all test suites matching /tests\\categoria_controller.test.js/i.
PS C:\Users\ACER\Desktop\TESIS>
```

Figura 11 Prueba unitaria de endpoint para visualizar categoría.

Crear producto

```
it("201 si todo OK", async () => {
  Ingrediente.find.mockResolvedValue([{ _id: "ing1" }, { _id: "ing2" }]);
  Categoria.findById.mockResolvedValue({ nombre: "Jabones artesanales" });
  Producto.findOne.mockResolvedValue(null);
  Producto.prototype.save = jest.fn().mockResolvedValue({});
  Producto.prototype.populate = jest.fn().mockResolvedValue({});

  const res = mockRes();
  await createProductoController({ ...baseReq }, res);

  expect(res.status).toBe(201);
});
```

Figura 12 Petición de endpoint para crear producto.

```
PASS tests/producto_controller.test.js (9.989 s)
  createProductoController
    ✓ 201 si todo OK (20 ms)
    ✓ 400 si faltan campos (3 ms)
    ✓ 400 si nombre duplicado y elimina imagen (3 ms)

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        10.294 s
```

Figura 13 Prueba unitaria de endpoint para crear producto.

Ver productos

```
describe("getAllProductosController", () => {
  afterEach(clearMocks);

  it("debería devolver lista paginada (200)", async () => {
    Producto.find.mockResolvedValue([{ nombre: "jabón A" }]);
    Producto.countDocuments.mockResolvedValue(1);

    const req = { query: { page: "1", limit: "10" } };
    const res = mockRes();

    await getAllProductosController(req, res);

    expect(Producto.find).toHaveBeenCalled();
    expect(res.status).toBe(200);
    expect(res.json).toHaveBeenCalledWith(
      expect.objectContaining({ totalProductos: 1, productos: [{ nombre: "jabón A" }] })
    );
  });
});
```

Figura 14 Petición de endpoint para ver productos.

```
PASS tests/producto_controller.test.js (8.316 s)
  getAllProductosController
    ✓ debería devolver lista paginada (200) (18 ms)
    ✓ debería responder 404 si no hay productos (3 ms)
    ✓ maneja errores internos (500) (191 ms)

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        8.554 s, estimated 9 s
Ran all test suites matching /tests\\producto_controller.test.js/i.
```

Figura 15 Prueba unitaria de endpoint para ver productos.

Desactivar producto

```
describe("deleteProductoController", () => {
  afterEach(clearMocks);

  it("200 y desactiva", async () => {
    Producto.findById.mockResolvedValue({ _id: "p1", activo: true });
    Producto.findByIdAndUpdate.mockResolvedValue({ activo: false });

    const req = { params: { id: "p1" } };
    const res = mockRes();

    await deleteProductoController(req, res);
    expect(res.status).toHaveBeenCalledWith(200);
  });
});
```

Figura 16 Petición de endpoint para desactivar producto.

```
PASS tests/producto_controller.test.js (8.277 s)
  deleteProductoController
    ✓ 200 y desactiva (14 ms)
    ✓ 400 si ya está inactivo (2 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        8.571 s, estimated 9 s
Ran all test suites matching /tests\\producto_controller.test.js/i.
```

Figura 17 Prueba unitaria de endpoint para desactivar producto.

Activar producto

```
230 describe("reactivarProductoController", () => {
231   afterEach(clearMocks);
232
233   it("200 al reactivar", async () => {
234     Producto.findById.mockResolvedValue({ _id: "p1", activo: false, save: jest.fn() });
235
236     const req = { params: { id: "p1" } };
237     const res = mockRes();
238
239     await reactivarProductoController(req, res);
240     expect(res.status).toHaveBeenCalledWith(200);
241   });
242 }
```

Figura 18 Petición de endpoint para activar producto.

```
PASS tests/producto_controller.test.js (7.521 s)
  reactivarProductoController
    ✓ 200 al reactivar (18 ms)
    ✓ 400 si ya está activo (3 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        7.78 s, estimated 9 s
Ran all test suites matching /tests\\producto_controller.test.js/i.
```

Figura 19 Prueba unitaria de endpoint para activar producto.

Ver venta

```
34 describe("getAllVentasController", () => {
35   test("debería retornar ventas paginadas", async () => {
36     Ventas.find.mockReturnValue({
37       populate: jest.fn().mockReturnValue({
38         skip: jest.fn().mockReturnValue({
39           limit: jest.fn().mockResolvedValue([{}])
40         })
41       })
42     });
43     Ventas.countDocuments.mockResolvedValue(1);
44
45     await getAllVentasController(req, res);
46
47     expect(res.status).toHaveBeenCalledWith(200);
48     expect(res.json).toHaveBeenCalledWith(
49       expect.objectContaining({ totalVentas: 1 })
50     );
51   });
52 });
```

Figura 20 Petición de endpoint para ver venta.

```

PASS tests/venta_controller.test.js (5.426 s)
  getAllVentasController
    ✓ debería retornar ventas paginadas (20 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        5.68 s, estimated 7 s
Ran all test suites matching /tests\\venta_controller.test.js/i.

```

Figura 21 Prueba unitaria de endpoint para ver venta.

Ver factura cliente

```

describe("getFacturaClienteById", () => {
  afterEach(() => jest.clearAllMocks());

  it("debería retornar la factura de una venta", async () => {
    const req = {
      params: { id: "venta123" },
      clienteBDD: { _id: "cli123" }
    };

    const res = {
      status: jest.fn().mockReturnThis(),
      json: jest.fn()
    };

    const ventaMock = {
      _id: "venta123",
      fecha_venta: "2024-01-01",
      total: 100,
      estado: "finalizado",
      cliente_id: {
        _id: "cli123",

```

Figura 22 Petición de endpoint para ver factura cliente.

```

PASS tests/venta_controller.test.js (5.949 s)
  getFacturaClienteById
    ✓ debería retornar la factura de una venta (20 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        6.211 s, estimated 7 s
Ran all test suites matching /tests\\venta_controller.test.js/i.

```

Figura 23 Prueba unitaria de endpoint para ver factura cliente.

Añadir producto al carrito

```
describe("addCarritoController", () => {
  test("agrega producto correctamente", async () => {
    req.body = { producto_id: new mongoose.Types.ObjectId().toString(), cantidad: 2 };

    Clientes.findById.mockResolvedValue({ _id: req.clienteBDD._id });
    Producto.findById.mockResolvedValue({ _id: req.body.producto_id, nombre: "Jabón", stock: 5, precio: 3, activo: true });
    Carrito.findOne.mockResolvedValue({
      _id: "carritoId",
      cliente_id: req.clienteBDD._id,
      estado: "pendiente",
      productos: [],
      save: jest.fn(),
      total: 0
    });
    Carrito.findById.mockResolvedValue({
      _id: "carritoId",
      productos: [
        {
          producto_id: req.body.producto_id,
          cantidad: 2,
          precio_unitario: 3,
          subtotal: 6
        }
      ]
    });
  });
});
```

Figura 24 Petición de endpoint para añadir producto.

```
PASS tests/carrito_controller.test.js (7.667 s)
  addCarritoController
    ✓ agrega producto correctamente (23 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        7.935 s, estimated 11 s
Ran all test suites matching /tests\carrito_controller.test.js/i.
```

Figura 25 Prueba unitaria de endpoint para añadir producto.

Vaciar carrito

```
describe("emptyCarritoController", () => {
  test("vacía carrito", async () => {
    Carrito.findOne.mockResolvedValue({ productos: [{}, {}], total: 10, estado: "pendiente", save: jest.fn() });

    await emptyCarritoController(req, res);
    expect(res.status).toHaveBeenCalledWith(200);
    expect(res.json).toHaveBeenCalledWith({ msg: "El carrito fue vaciado exitosamente." });
  });
});
```

Figura 26 Petición de endpoint para vaciar carrito.

```

PASS tests/carrito_controller.test.js (7.918 s)
  emptyCarritoController
    ✓ vacía carrito (22 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        8.135 s
Ran all test suites matching /tests\\carrito_controller.test.js/i.

```

Figura 27 Prueba unitaria de endpoint para vaciar carrito.

Pruebas de rendimiento

A continuación, se presentan las pruebas de rendimiento restantes las cuales van desde la **Figura 28** hasta la **Figura 39**.

Visualizar clientes

The screenshot shows the 'Petición HTTP' (HTTP Request) editor. The 'Nombre' (Name) field is 'Visualizar clientes'. The 'Comentarios' (Comments) field is empty. The 'Basic' tab is selected. Under 'Servidor Web' (Web Server), the 'Protocolo' (Protocol) is 'http', 'Nombre de Servidor o IP' (Server or IP Name) is 'localhost', and 'Puerto' (Port) is '3000'. The 'Petición HTTP' (HTTP Request) section shows a 'GET' method and a 'Ruta' (Path) of '/api/admin/clientes?page=1&limit=10'. There are checkboxes for 'Redirigir Automáticamente' (Automatically Redirect), 'Seguir Redirecciones' (Follow Redirects), 'Utilizar KeepAlive' (Use KeepAlive), 'Usar \'multipart/form-data\' para HTTP POST' (Use \'multipart/form-data\' for HTTP POST), and 'Cabeceras compatibles con navegadores' (Browser-compatible headers). The 'Parameters' tab is selected, showing a table with columns: Nombre, Valor, ¿Codificar?, Content-Type, and ¿Incluir Equals?.

Figura 28 Petición de endpoint para visualizar clientes.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Visualizar client...	100	1754	476	2698	543,12	0,00%	28,1/sec	39,56	10,89	1442,0
Total	100	1754	476	2698	543,12	0,00%	28,1/sec	39,56	10,89	1442,0

Figura 29 Prueba de rendimiento para visualizar clientes.

Visualizar categorías

Petición HTTP

Nombre: Visualizar categorías

Comentarios:

Basic Advanced

Servidor Web

Protocolo: http Nombre de Servidor o IP: localhost Puerto: 3000

Petición HTTP

GET Ruta: /api/categorias Codificación del contenido:

☐ Redirigir Automáticamente ☒ Seguir Redirecciones ☒ Utilizar KeepAlive ☐ Usar 'multipart/form-data' para HTTP POST ☐ Cabeceras compatibles con navegadores

Parameters Body Data Files Upload

Enviar Parámetros Con la Petición:

Nombre:	Valor	¿Codificar?	Content-Type	¿Incluir Equals?
---------	-------	-------------	--------------	------------------

Figura 30 Petición de endpoint para visualizar categorías.

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Visualizar categ...	100	638	114	1401	292,45	0,00%	43,5/sec	35,44	6,96	835,0
Total	100	638	114	1401	292,45	0,00%	43,5/sec	35,44	6,96	835,0

Figura 31 Prueba de rendimiento para visualizar categorías.

Actualizar producto

Petición HTTP

Nombre: Actualizar producto

Comentarios:

Basic Advanced

Servidor Web

Protocolo: http Nombre de Servidor o IP: localhost Puerto: 3000

Petición HTTP

GET Ruta: /api/productos/684cb8441ef9e158fa483124 Codificación del contenido:

☐ Redirigir Automáticamente ☒ Seguir Redirecciones ☒ Utilizar KeepAlive ☐ Usar 'multipart/form-data' para HTTP POST ☐ Cabeceras compatibles con navegadores

Parameters Body Data Files Upload

```
1 {
2   "nombre": "Nuevo Rollos 2en1",
3   "precio": "1200000"
4 }
5
```

Figura 32 Petición de endpoint para actualizar producto.

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Actualizar prod...	100	4376	2821	5534	464,32	0,00%	16,5/sec	18,35	7,86	1142,0
Total	100	4376	2821	5534	464,32	0,00%	16,5/sec	18,35	7,86	1142,0

Figura 33 Prueba de rendimiento para actualizar producto.

Visualizar venta

Petición HTTP

Nombre: Visualizar venta

Comentarios

Basic Advanced

Servidor Web

Protocolo: http Nombre de Servidor o IP: localhost Puerto: 3000

Petición HTTP

GET Ruta: /api/ventas/684cdc27d482126de8a8e5cd Codificación del contenido:

☐ Redirigir Automáticamente ☒ Seguir Redirecciones ☒ Utilizar KeepAlive ☐ Usar 'multipart/form-data' para HTTP POST ☐ Cabeceras compatibles con navegadores

Parameters Body Data Files Upload

Enviar Parámetros Con la Petición:

Nombre:	Valor	¿Codificar?	Content-Type	¿Incluir Equals?
---------	-------	-------------	--------------	------------------

Figura 34 Petición de endpoint para visualizar venta.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Visualizar venta	100	5027	1462	6602	1255,83	0,00%	13,4/sec	11,23	4,80	857,0
Total	100	5027	1462	6602	1255,83	0,00%	13,4/sec	11,23	4,80	857,0

Figura 35 Prueba de rendimiento para visualizar venta.

Visualizar perfil cliente

Petición HTTP

Nombre: Visualizar Perfil Cliente

Comentarios

Basic Advanced

Servidor Web

Protocolo: http Nombre de Servidor o IP: localhost Puerto: 3000

Petición HTTP

GET Ruta: /api/perfil Codificación del contenido:

☐ Redirigir Automáticamente ☒ Seguir Redirecciones ☒ Utilizar KeepAlive ☐ Usar 'multipart/form-data' para HTTP POST ☐ Cabeceras compatibles con navegadores

Parameters Body Data Files Upload

Enviar Parámetros Con la Petición:

Nombre:	Valor	¿Codificar?	Content-Type	¿Incluir Equals?
---------	-------	-------------	--------------	------------------

Figura 36 Petición de endpoint para visualizar perfil cliente.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Visualizar Perfil ...	100	4410	638	8999	2708,59	0,00%	10,2/sec	7,30	3,63	734,0
Total	100	4410	638	8999	2708,59	0,00%	10,2/sec	7,30	3,63	734,0

Figura 37 Prueba de rendimiento para visualizar perfil cliente.

Ver Carrito

Petición HTTP

Nombre: Visualizar carrito de compras

Comentarios:

Basic Advanced

Servidor Web

Protocolo: http Nombre de Servidor o IP: localhost Puerto: 3000

Petición HTTP

GET Ruta: /api/carritos Codificación del contenido:

☐ Redirigir Automáticamente ☒ Seguir Redirecciones ☒ Utilizar KeepAlive ☐ Usar 'multipart/form-data' para HTTP POST ☐ Cabeceras compatibles con navegadores

Parameters Body Data Files Upload

Enviar Parámetros Con la Petición:

Nombre:	Valor	¿Codificar?	Content-Type	¿Incluir Equals?
---------	-------	-------------	--------------	------------------

Figura 38 Petición de endpoint para visualizar carrito.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Visualizar carr...	100	7638	3367	10408	2258,21	0,00%	9,4/sec	6,63	3,39	719,0
Total	100	7638	3367	10408	2258,21	0,00%	9,4/sec	6,63	3,39	719,0

Figura 39 Prueba de rendimiento para visualizar carrito.

Pruebas de aceptación

A continuación, se presentan las pruebas de aceptación restantes las cuales van desde la **Tabla 10** hasta la **Tabla 15**.

Tabla 10 Prueba de aceptación para la gestión de usuarios, categorías, productos y ventas.

PRUEBA DE ACEPTACIÓN DE FLOR&CERA	
Designación (ID): PA-002	Identificador de historia de Usuario: HU002
Nombre: Gestionar usuarios, categorías, productos y ventas.	
Descripción: El componente backend entrega varios endpoints que son utilizados por el administrador para: <ul style="list-style-type: none"> Gestionar usuarios. Gestionar categorías y productos. Gestionar ventas. 	
Instrucciones de funcionamiento: Para gestionar usuarios, categorías, productos y ventas: <ul style="list-style-type: none"> Acceder a la URL correspondiente del componente backend en un navegador. Iniciar sesión en el endpoint de Login Administrador. 	

<ul style="list-style-type: none"> • Dirigirse al endpoint correspondiente para: <ul style="list-style-type: none"> - Visualizar, desactivar o activar un usuario (cliente) - Crear, visualizar, editar y eliminar una categoría - Crear, visualizar, editar y eliminar un producto - Visualizar, editar una venta • Seguir las instrucciones proporcionadas en cada endpoint. Los endpoints están restringidos al rol de usuario “Administrador”, por lo que solo los usuarios con ese rol podrán acceder a las funcionalidades de gestión. • Cabe resaltar que a pesar de que hay una opción para crear categorías hasta el momento existen únicamente 2 que son las especificadas (velas y jabones artesanales).
<p>Resultado deseado:</p> <p>El backend permite gestionar usuarios, categorías, productos y ventas, de acuerdo con los permisos de acceso y las funcionalidades habilitadas para su rol.</p>
<p>Evaluación:</p> <p>El cliente confirma que todas las funcionalidades operan correctamente y cumplen las expectativas.</p>

Tabla 11 Prueba de aceptación para la visualización de reporte de ventas.

PRUEBA DE ACEPTACIÓN DE FLOR&CERA	
Designación (ID): PA-003	Identificador de historia de Usuario: HU003
Nombre: Visualizar reporte de ventas.	
<p>Descripción: El componente backend entrega varios endpoints que son utilizados por el administrador para:</p> <ul style="list-style-type: none"> • Visualizar reporte de ventas. 	
<p>Instrucciones de funcionamiento:</p> <p>Para visualizar los reportes de ventas:</p> <ul style="list-style-type: none"> • Acceder a la URL del componente backend en un navegador o cliente API (Postman). • Dirigirse al endpoint correspondiente (Reporte de ventas por parte de administrador). • Enviar las solicitudes HTTP con los parámetros requeridos según la acción 	

deseada.
Resultado deseado: El backend permite ver los reportes de ventas para el rol administrador, es decir, ver información de los clientes que han comprado los productos.
Evaluación: El cliente confirma que todas las funcionalidades operan correctamente y cumplen las expectativas.

Tabla 12 Prueba de aceptación para la visualización de categorías y productos.

PRUEBA DE ACEPTACIÓN DE FLOR&CERA	
Designación (ID): PA-004	Identificador de historia de Usuario: HU004
Nombre: Visualizar categorías y productos.	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> Visualizar categorías y productos. 	
Instrucciones de funcionamiento: Para visualizar las categorías o productos: <ul style="list-style-type: none"> Acceder a la URL del componente backend en un navegador o cliente API (Postman). Dirigirse al endpoint correspondiente (Visualizar categorías o visualizar productos). Enviar las solicitudes HTTP con los parámetros requeridos según la acción deseada. Seguir las respuestas proporcionadas por el backend para confirmar las acciones realizadas. 	
Resultado deseado: El backend permite tanto al administrador como a los clientes visualizar las categorías y productos disponibles.	
Evaluación: El cliente confirma que todas las funcionalidades operan correctamente y cumplen las expectativas.	

Tabla 13 Prueba de aceptación para la gestión de personalización de productos.

PRUEBA DE ACEPTACIÓN DE FLOR&CERA	
Designación (ID): PA-005	Identificador de historia de Usuario: HU005
Nombre: Gestionar personalización de productos.	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> • Gestionar personalización de productos. • Personalización de productos mediante IA 	
Instrucciones de funcionamiento: Para gestionar la personalización de productos: <ul style="list-style-type: none"> • Acceder a la URL del componente backend en un navegador o cliente API (Postman). • Dirigirse al endpoint correspondiente (Crear, visualizar, editar, eliminar o recomendación por IA). • Enviar las solicitudes HTTP con los parámetros requeridos según la acción deseada. • Seguir las respuestas proporcionadas por el backend para confirmar las acciones realizadas. 	
Resultado deseado: El backend permite crear, visualizar, editar, eliminar o recomendar un producto por IA de manera funcional y segura. Cabe resaltar que la personalización de productos los hace el cliente a sus preferencias o mediante la recomendación de la IA.	
Evaluación: El cliente confirma que todas las funcionalidades operan correctamente y cumplen las expectativas.	

Tabla 14 Prueba de aceptación para la gestión de carrito de compras.

PRUEBA DE ACEPTACIÓN DE FLOR&CERA	
Designación (ID): PA-006	Identificador de historia de Usuario: HU006
Nombre: Gestionar carrito de compras.	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> • Gestionar carrito de compras. 	
Instrucciones de funcionamiento: Para la gestión del carrito de compras se siguen las siguientes instrucciones: <ul style="list-style-type: none"> • Acceder a la URL del componente backend en un navegador o cliente API (Postman). • Dirigirse al endpoint correspondiente (ver, añadir, modificar cantidad, sacar, vaciar o pagar los productos del carrito de compras). • Enviar las solicitudes HTTP con los parámetros requeridos según la acción deseada • Seguir las respuestas proporcionadas por el backend para confirmar las acciones realizadas. 	
Resultado deseado: El backend permite al cliente añadir, ver, sacar, modificar la cantidad, vaciar o pagar el carrito de compras de manera funcional y segura.	
Evaluación: El cliente confirma que todas las funcionalidades operan correctamente y cumplen las expectativas.	

Tabla 15 Prueba de aceptación para la visualización de reportes.

PRUEBA DE ACEPTACIÓN DE FLOR&CERA	
Designación (ID): PA-007	Identificador de historia de Usuario: HU007
Nombre: Visualizar reportes	
Descripción: El componente backend entrega varios endpoints que son utilizados por el cliente para: <ul style="list-style-type: none"> • Visualizar reporte de ventas. • Visualizar reporte de facturas. 	

Instrucciones de funcionamiento:

Para visualizar los reportes:

- Acceder a la URL del componente backend en un navegador o cliente API (Postman).
- Dirigirse al endpoint correspondiente (ver reportes de ventas o facturas).
- Enviar las solicitudes HTTP con los parámetros requeridos según la acción deseada.
- Seguir las respuestas proporcionadas por el backend para confirmar las acciones realizadas.

Resultado deseado:

El backend permite al cliente ver las ventas que ha realizado o también ver reportes de facturas de las ventas hechas de una manera funcional y segura.

Evaluación:

El cliente confirma que todas las funcionalidades operan correctamente y cumplen las expectativas.

Certificado de cumplimiento de requisitos por parte de Flor&Cera



Flor & Cera

Flor&Cera

Quito, 14 de julio de 2025

CERTIFICADO

Yo, Henry Pazto Corregidor, con CI 1726685934, como representante legal de Flor&Cera ubicada en Quito-Ecuador.

Por medio de la presente certifico:

Que la Srta. **Estefanía Melisa Sánchez Párraga** con cédula de ciudadanía **1316288412**, estudiante de la carrera de **Desarrollo de Software**, realizó su Trabajo de Integración Curricular de **“DESARROLLO DE UN E-COMMERCE PARA LA VENTA DE PRODUCTOS ARTESANALES PERSONALIZADOS BASADO EN IA”**, el cual cumple al 100% con todos los requerimientos y funcionalidades que se han definido 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:

Henry Alexander Pazto Corregidor

Representante de Flor&Cera

0990721714

ANEXO III

A continuación, para visualizar el Manual de Usuario del backend se debe digitar la siguiente URL.

<https://www.youtube.com/watch?v=30zP7An7qwg>

En donde se explica de forma clara y sencilla las diversas funcionalidades del backend, así como cada uno de los perfiles que forman parte de este componente.

ANEXO IV

A continuación, se detallan las credenciales para el acceso al backend, junto con el vínculo al repositorio de GitHub que alberga el código íntegro y las instrucciones de instalación en la sección README.

Credenciales para el ingreso al backend

Para ingresar al backend en producción, se puede hacer a través de la URL siguiente:

<https://tesis-ecommerce.onrender.com>

Para ingresar a la documentación del backend en producción, se puede hacer a través de la URL siguiente:

<https://documenter.getpostman.com/view/42480684/2sB2j1jDif>

Credenciales para el perfil administrador:

- Email: admin123@yopmail.com
- Contraseña: NuevaPass123\$

Credenciales para el perfil cliente:

- Email: cliente123@yopmail.com
- Contraseña: NuevaPass123\$

Repositorio del backend

El proyecto se encuentra en un repositorio de GitHub, al cual se puede acceder a través del enlace siguiente:

<https://github.com/estefaniamp/Tesis-Ecommerce.git>