BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

APLICACIÓN WEB DISTRIBUIDA PARA LA ADMINISTRACIÓN CONTABLE

TESIS PROFESIONAL

PARA OBTENER EL TÍTULO DE:

LICENCIATURA EN INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:

GILBERTO PÉREZ GARRIDO

DIRECTOR DE LA TESIS:

M. en C. María Antonia Ruíz Díaz

Puebla, México SEPTIEMBRE, 2024

Dedicatoria

A mi familia y a quienes fueron mis verdaderos amigos. A mi amigo de cuatro patas, Minino, por hacer más llevaderas las clases en línea durante la pandemia. También, a quien fue mi cónyuge al inicio de mi educación superior y a las personas amables que conocí durante la carrera universitaria, a quienes, por circunstancias ajenas, ya no volví a ver, sin saber que las últimas veces que nos encontraríamos serían al inicio de la crisis sanitaria del dos mil veinte. Por su apoyo incondicional de todos. Finalmente quiero agradecerme a mí mismo por la paciencia que tuve, incluso cuando no estuve de acuerdo con algunos profesores desde la preparatoria.

Agradecimientos

Agradezco a mi supervisora, M. en C. María Antonia Ruíz Díaz, así como a todas las personas que me brindaron su apoyo durante este proyecto. Mi reconocimiento también va dirigido a algunos profesores que tuve tanto en la facultad como en la educación media superior, y a la **Benemérita Universidad Autónoma de Puebla** por ser parte esencial de mi formación.

Índice

Capítulo 1

Introducción

1.1. Problemática

El desafío fundamental que aborda el proyecto de tesis se centra en la necesidad creciente de Micros y Pequeñas empresas por acceder a información financiera en tiempo real, con actualización automática, minimizando errores y asegurando la seguridad de los datos. Esta urgencia se evidencia al contrastar las limitaciones de almacenamiento local con las capacidades más dinámicas y seguras de la nube, especialmente en términos de bases de datos y servicios como Amazon Web Services. Además, se hace patente la desventaja de los sistemas contables locales y manuales, como Excel, frente a la agilidad requerida en la elaboración de reportes financieros. La focalización en negocios pequeños y la implementación de roles diferenciados en lugar de depender de un único administrador se erigen como elementos cruciales para mejorar la eficiencia y precisión en la gestión de datos.
El objetivo esencial de este proyecto radica en ofrecer soluciones tecnológicas adaptadas a las necesidades específicas de las Micros y Pequeñas empresas, atendiendo a su demanda por información actualizada y segura. La transición de almacenamiento local a opciones en la nube y la evolución de sistemas contables manuales hacia software especializado para estos negocios emergentes prometen ser un cambio significativo. Al priorizar la facilidad de acceso, la seguridad de la información y la optimización de roles para una gestión más especializada, este proyecto busca no solo mejorar la administración financiera, sino también posibilitar un flujo de trabajo más ágil y preciso en la generación de reportes cruciales para estas empresas.

1.2. Objetivos generales

Desarrollar e implementar una Aplicación Web especializada para la gestión contable en el sector de abarrotes, con un enfoque en la distribución de software que facilite operaciones simultáneas desde diversas locaciones. Haciendo uso de la metodología ágil Scrum, Esta solución tiene las siguientes funcionalidades, proporcionando un conjunto integral de herramientas para la gestión contable en el sector de abarrotes. Las funcionalidades de login, como inicio y cierre de sesión, establecen una sólida base para la seguridad del sistema. Las funciones de gestión abarcan áreas críticas como categorías, productos, denominaciones, usuarios y roles, junto con permisos detallados.

La gestión del carrito en ventas no solo facilita la contabilización del total a pagar y el número de artículos, sino que también permite seleccionar denominaciones o efectivo recibido para calcular el cambio a entregar al cliente. Además, la capacidad de búsqueda en todas las gestiones, excepto en la gestión de permisos por rol, agrega una capa adicional de eficiencia. Las funcionalidades de información contabilizada incluyen corte de caja por rango de fechas, proporcionando detalles de ventas para un usuario específico. También se incorpora un reporte de ventas básico con opciones de descarga en PDF o Excel, complementado con un resumen detallado de las ventas directamente en la interfaz. Estas funcionalidades tienen como consecuencia una serie de ventajas para el rubro de abarrotes, ofreciendo una gestión contable ágil.

1.3. Objetivos específicos

Los pasos específicos seguidos para llevar a cabo el desarrollo de software consistieron en aplicar la metodología conocida como el ciclo de vida del desarrollo de software (SDLC, por sus siglas en inglés "Software Development Life Cycle"). El SDLC, una metodología ampliamente adoptada en la industria del software, se estructura en fases que abarcan desde la planificación y análisis hasta la implementación y mantenimiento.

El autor (Balaji y Murugaiyan, 2012) menciona que "Un ciclo de vida de desarrollo de software (SDLC) se adhiere a fases importantes que son esenciales para los desarrolladores, como la planificación, el análisis, el diseño y la implementación" (p. 1).

- **Análisis**: En la etapa de análisis del ciclo de vida del desarrollo de software (SDLC) para el diseño de un sistema contable orientado al rubro de abarrotes, se realiza una investigación detallada centrada en comprender a fondo las particularidades y requerimientos específicos de este sector.

  En una primera fase, se llevará a cabo un estudio exhaustivo de los sistemas contables actualmente utilizados por pequeñas y medianas empresas de abarrotes. Este análisis permitirá identificar áreas de mejora, como la optimización en la gestión de inventario y la necesidad de reportes financieros más adaptados a las operaciones diarias de este tipo de negocios.

  Adicionalmente, se explorarán las características específicas del flujo de trabajo en los abarrotes, destacando la importancia de aspectos como la rápida gestión de ventas de productos y la necesidad de una interfaz fácil de usar para usuarios con diversos niveles de habilidad tecnológica.

  La investigación se centrará en aspectos prácticos y operativos sin recurrir a regulaciones fiscales específicas ni entrevistas directas con propietarios de abarrotes, buscando una comprensión profunda de los procesos cotidianos y los desafíos inherentes al sector.

  Diseño:

  Diseño de alto nivel:

  En la fase de diseño a nivel alto, se establece la arquitectura general del sistema contable para abarrotes. Se definirán las principales características y módulos que responderían

a las necesidades identificadas en la fase de análisis. También se estructurará el diseño de casos de usos para proporcionar una visión general de las funciones principales del sistema.

Se diseñan componentes clave como el módulo de gestión de inventario, el sistema de ventas y punto de venta, así como los módulos contables. Además, se planificaron las interfaces de usuario con un enfoque intuitivo, asegurando una experiencia amigable para los usuarios finales.

Diseño de bajo nivel:

A un nivel más detallado, se abordaron aspectos específicos de cada módulo y componente. Se diseñaron las bases de datos, definiendo tablas, relaciones y esquemas para garantizar un almacenamiento eficiente y seguro de la información.

En el diseño a nivel bajo, se establecen las tecnologías específicas a utilizar, como el framework Laravel para el desarrollo back-end y las tecnologías como HTML, CSS y JavaScript para el desarrollo front-end. Se planifica la integración con servicios en la nube, especialmente con AWS RDS (Servicio de Bases de Datos Relaciones de Amazon) para la gestión de bases de datos distribuidas.

Desarrollo: En la fase de desarrollo, se opta por aplicar los principios ágiles de manera individual, centrándose en la flexibilidad, adaptabilidad y entrega continua de valor. Inspirado en los fundamentos del Manifiesto Ágil, se busca mantener un enfoque colaborativo y eficiente.

La metodología ágil adoptada sigue un enfoque iterativo, dividiendo el desarrollo en incrementos manejables. A través de ciclos de trabajo cortos y enfoque en la entrega constante de funcionalidades, se busca incorporar feedback de manera continua y ajustar el rumbo según las necesidades emergentes que conllevará el estudio de sistemas contables actualmente utilizados por pequeñas y medianas empresas de abarrotes.

Se divide el desarrollo del sistema contable mediante el uso del concepto sprint, el cual implica la planificación, ejecución, revisión y retrospectiva de cada iteración.

Planificación del Sprint: Se define el alcance de cada Sprint en función de los requisitos del sistema contable y las necesidades de los potenciales clientes. Se seleccionan las funcionalidades prioritarias del backlog del producto, como los menús de gestiones, registros de transacciones, generación de informes, etc.

Sprint: Durante el Sprint, se trabaja en la implementación de las funcionalidades seleccionadas. Se desarrollan y prueban las características del sistema contable de acuerdo con las especificaciones definidas en la planificación del Sprint.

Revisión del Sprint: Al final del Sprint, se presenta al cliente o al potencial cliente del producto las funcionalidades completadas del sistema contable.

Retrospectiva del Sprint: Después de la revisión del Sprint, el equipo Scrum realiza una retrospectiva para analizar lo que salió bien, lo que no salió tan bien y cómo pueden

mejorar en el próximo Sprint. En este caso, la retrospectiva se omite ya que se realiza al presentar los entregables al cliente o al potencial cliente.

La aplicación de los principios ágiles, como la entrega temprana de software valioso, la capacidad de respuesta a cambios y el mantenimiento de un enfoque constante en la mejora técnica, fue fundamental. Aunque se trabajó de manera individual en cuanto a los aspectos técnicos, se procuró mantener la transparencia en el progreso del desarrollo y fomentar la adaptabilidad mediante revisiones periódicas de los objetivos y prioridades.

Pruebas:

Pruebas de caja negra: Para llevar a cabo las pruebas de caja negra, se emplearían diversas técnicas. Se aplicaría la partición de equivalencia para los casos exitosos y no exitosos en el módulo de ventas, así como pruebas de valores límite, especialmente enfocadas en el mencionado módulo. Además, se llevaría a cabo pruebas de transición de estados y casos de uso.

Pruebas de caja blanca: Para realizar las pruebas de caja blanca, se emplean pruebas de declaración y pruebas de cobertura de código para evaluar algunas decisiones clave (particularmente en el módulo de ventas).

Implementación: Durante la fase de implementación del sistema contable, se utilizaría Docker para encapsular la aplicación y sus dependencias en contenedores, lo que facilita su despliegue. Además, se aprovechará AWS (Amazon Web Services) para desplegar la aplicación en un EC2 (Elastic Compute Cloud). Para hacer que el sistema sea distribuido, se emplea RDS (Relational Database Service) de AWS para alojar la base de datos, lo que permite gestionar de manera eficiente el almacenamiento y acceso a los datos de manera distribuida. Por último, se utiliza S3 (Simple Storage Service) de AWS para almacenar y gestionar archivos estáticos.

1.4. Contribuciones

Este trabajo de tesis aporta significativamente al ámbito de los sistemas contables al enfocarse en el desarrollo de funcionalidades clave diseñadas específicamente para atender las necesidades de las micro y pequeñas empresas (MYPEs). Entre las contribuciones más destacadas se incluyen:

Interfaz simplificada:

Diseño de una interfaz intuitiva y simplificada, permitiendo a las MYPEs acceder y utilizar fácilmente las funcionalidades contables esenciales sin requerir un conocimiento técnico profundo.

Gestión de transacciones:

Implementación de herramientas especializadas para la gestión eficiente de transacciones diarias, simplificando la entrada de datos contables y facilitando algunos de los procesos críticos para la gestión financiera de las MYPEs.

Accesibilidad en costos: Garantiza que el sistema contable sea accesible para micro y pequeñas

empresas en términos de costos. Se busca minimizar las barreras financieras, permitiendo a una amplia gama de empresas beneficiarse de las ventajas de un sistema contable ágil y eficiente.

Implementación distribuida en la nube con AWS RDS: Una solución ágil para el almacenamiento en el sistema contable: Una de las contribuciones clave de esta tesis es la implementación de un sistema contable distribuido, aprovechando las avanzadas tecnologías en la nube proporcionadas por AWS RDS. Esta elección estratégica permite, con apenas unos clics, distribuir de manera eficiente y segura el almacenamiento de datos, proporcionando una solución ágil y escalable para las Micros y Pequeñas empresas.

Además se contribuye en cómo aplicar la metodología ágil en un sistema contable. Las aportaciones clave en este sentido incluyen:

Integración de prácticas ágiles:

Proporcionar una guía sobre cómo integrar prácticas ágiles en el desarrollo de sistemas contables, permitiendo una adaptabilidad continua a los cambios en los requisitos y mejorando la capacidad de respuesta a las necesidades cambiantes de las empresas.

Ciclos de desarrollo iterativos:

Promueve la adopción de ciclos de desarrollo iterativos y entregas incrementales, permitiendo la revisión continua del software contable en desarrollo y la incorporación temprana del feedback del usuario. Esto se traduce en un proceso de desarrollo más ágil y alineado con las expectativas del cliente.

Colaboración interdisciplinaria:

Fomenta la colaboración continua entre profesionales financieros y desarrolladores de software. Esta integración garantiza que el sistema contable no solo cumpla con algunos estándares financieros, sino que también sea eficiente y usable, reflejando las necesidades reales del usuario.

Flexibilidad para cambios:

Diseña el sistema contable de manera que sea intrínsecamente flexible ante los cambios en los requisitos contables. La capacidad de adaptarse ágilmente a nuevas regulaciones o ajustes en los procesos empresariales es una característica fundamental de las soluciones propuestas.

Capítulo 2

Marco teórico

Para el desarrollo del sistema contable, se llevó a cabo una investigación exhaustiva sobre las tecnologías y herramientas esenciales para asegurar un proceso de desarrollo eficiente y la creación de un sistema robusto y moderno. Entre las herramientas y tecnologías seleccionadas se encuentran:

2.1. Generales

Aplicación web: Según el autor (Luján-Mora, 2002) "una aplicación web se puede definir como una aplicación en la cual el usuario por medio de un navegador realiza peticiones a una aplicación remota accesible a través de Internet (o a través de una Intranet) y que recibe una respuesta que se muestra en el propio navegador" (p. 2).

Servidor web apache: El autor (Muñoz Escoí y cols., 2013) menciona que "Apache es un servidor web que emplea programación concurrente. El objetivo de un servidor web es la gestión de cierto conjunto de páginas web ubicadas en ese servidor. Dichas páginas resultan accesibles utilizando el protocolo HTTP (o Hypertext Transfer Protocol)" y que "un servidor de este tipo interesa tener múltiples hilos de ejecución, de manera que cada uno de ellos pueda atender a un cliente distinto, paralelizando así la gestión de múltiples clientes" (p. 7).

Sistema distribuido. El autor (Muñoz Escoí y cols., 2013) menciona que "Un sistema distribuido es aquel en el que los componentes ubicados en computadoras interconectadas en una red se comunican y coordinan sus acciones simplemente pasando mensajes. Esta definición conduce a las siguientes características especialmente importantes de los sistemas distribuidos: simultaneidad de componentes, falta de un reloj global y fallas de componentes independientes" (p. 2).

Base de datos: El autor (Gómez y cols., 2017) menciona que "Una base de datos es un conjunto ordenado y estructurado de datos que representan una realidad objetiva y que están organizados independientemente de las aplicaciones, significa que puedan ser utilizadas y compartidas por usuarios y aplicaciones diferentes" (p. 11).

Transacciones: El autor (Combaudon, 2018) menciona que "una transacción es un grupo de peticiones que son consideradas por el servidor como una sola unidad lógica: todas las peticiones se

ejecutan con éxito y las modificaciones se actualizan de manera permanente en la base de datos (COMMIT), o bien se produce un error y todas las modificaciones introducidas en la transacción son anuladas (rollback)" (p. 50).

Base de datos distribuida: El autor (Navas y Gallego, s.f.) comenta que es "una colección de múltiples bases de datos interrelacionadas lógicamente y distribuidas por una red de computadores" (p. 3).

Cliente: El autor (Navas y Gallego, s.f.) comenta que "un cliente se podría corresponder con una máquina de usuario que proporciona capacidad de interfaz al usuario y procesamiento local" (p. 9).

Servidor: El autor (Navas y Gallego, s.f.) menciona que "un servidor es una máquina que puede proporcionar a las máquinas cliente servicios, tales como impresión, acceso a ficheros, o acceso a la base de datos" (p. 9).

Escalabilidad: El autor (Muñoz Escoí y cols., 2013) comenta que "Si una determinada aplicación puede diseñarse e implantarse como un conjunto de componentes que interactúen y colaboren entre sí, generando al menos una actividad por cada componente, se facilitará la distribución de la aplicación sobre diferentes ordenadores" (p. 4).

Termial basado en UNIX: El autor (Bourne, 1978) comenta que "El shell UNIX es un lenguaje de programación de comandos que proporciona una interfaz para el sistema operativo UNIX. Contiene varios mecanismos que se encuentran en los lenguajes algorítmicos, como primitivas de flujo de control, variables y paso de parámetros. Construcciones como while, if, for y case están disponibles. Es posible la comunicación bidireccional entre el shell y los comandos. Los parámetros con valores de cadena, normalmente nombres de archivos o indicadores, se pueden pasar a un comando" (s. p.), por otro lado (Love, 2010) comenta que "Hoy en día, Unix es una familia de sistemas operativos que implementan una interfaz de programación de aplicaciones (API) similar y se basan en decisiones de diseño compartidas". (p. 1).

Máquina Virtual: (Laureano, Maziero, y Jamhour, 2004) comenta que "Una máquina virtual es una réplica de software de una máquina real subyacente. Varias máquinas virtuales pueden funcionar en la misma máquina host al mismo tiempo, sin interferir entre sí. Este concepto está adquiriendo valor en los sistemas informáticos de producción, debido a sus beneficios en términos de costos y portabilidad". (s. p).

Linux: (Love, 2010) comenta que "Los conceptos básicos de un sistema Linux son el kernel, la biblioteca C, la cadena de herramientas y las utilidades básicas del sistema, como un proceso de inicio de sesión y un shell". (p. 4).

Compilar: (Network, Sin fecha) comenta que "Compilar es el proceso de transformar un programa informático escrito en un lenguaje en un conjunto de instrucciones en otro formato o lenguaje. Un compilador es un programa de computadora que realiza dicha tarea". (s. p).

Sistema de control de versiones: Según el autor (Leal, Sosa, y Leal, 2012), "Los sistemas de control de versiones son aplicaciones que ayudan al proceso de desarrollo de software, facilitando la gestión del control de versiones de los archivos de código fuente generados por los desarrolladores,

proporcionando herramientas para la fusión y generación de una nueva versión de un proyecto, permitiendo que múltiples desarrolladores trabajen en el mismo proyecto sin ocasionar pérdida de datos o bloqueos de archivos" (s. p.).

Github: Según el autor (Cosentino, Luis, y Cabot, 2016), "GitHub, una de las plataformas de codificación social más populares, es la plataforma de referencia a la hora de minar repositorios Open Source para aprender de experiencias pasadas" (s. n.).

Entorno de desarrollo: Según el autor (Solutions, 2021), "Este es el entorno que está en tu ordenador/laptop. Aquí es donde harás todas tus actualizaciones de código. Es donde viven todos tus commits y ramas, junto con los de tus compañeros de trabajo" (s. n.).

Stage o pruebas: Según el autor (Solutions, 2021), "El entorno de staging es tan similar al entorno de production como sea posible. Esta vez tendrá todo el código en un servidor en lugar de una máquina local. Se conectará a tantos servicios como pueda sin tocar el entorno de producción. Todos los hard core testing ocurren aquí" (s. n.).

Entorno de producción: Según el autor (Solutions, 2021), "El entorno de producción es donde los usuarios acceden al código final después de todas las actualizaciones y pruebas. De todos los entornos este es el más importante" (s. n.).

Entorno de desarrollo

Según Díaz, C. A. (2019), "Aunque es posible realizar nuestros programas con herramientas sencillas, como el Bloc de notas o Notepad++, es necesario considerar que el uso de un entorno de desarrollo o IDE (Integrated Development Environment) nos permitirá trabajar en aplicaciones más completas, gracias a la posibilidad de acceder a servicios y ventajas que los métodos más simples no proporcionan" (p. 31).

- **Visual Studio Code**: Utilizado como el editor de código principal. para escribir y depurar código de forma rápida y eficiente en cualquier plataforma, para cualquier dispositivo, utilizando cualquier lenguaje de programación y en el sistema operativo de su elección. Según (Microsoft, 2024), "Visual Studio Code es un editor de código fuente ligero pero eficaz que se ejecuta en el escritorio y está disponible para Windows, macOS y Linux" (s. p).
- **xdebug 3.1.5**: Empleado para facilitar la depuración y análisis de código, mejorando la eficiencia del desarrollo. Según (Romero, s.f.) es "Acción de diferentes modalidades de revisión para la detección de defectos de defectos con el propósito de efectuar su corrección", ya que el mismo menciona información relevante en cuanto Laravel, PHP y herramientas de depuración de código en el back-end (p. 167)
- **iTerm (fish)**: Según (Mohamad Adib, 2014), "Es un terminal Mac OSX que se centra en el rendimiento, internacionalización y compatibilidad con funciones personalizadas para permitir productividad" (p. 31). Hay varios otros shells entre los que se puede elegir. Según (Morris, McCubbin, y Page, 2019), "Fish también es un excelente shell inicial con muchas funciones para ayudar al usuario a navegar por el shell sin sentirse perdido" (p. 8).

2.2. Back-end

El back-end se refiere a la parte del sistema que gestiona la lógica de negocio, la manipulación de datos y la comunicación con la base de datos. En el desarrollo web, el back-end maneja las solicitudes del usuario, procesa la información y envía las respuestas al cliente. Según Pérez Ibarra, S. G., Quispe, J. R., Mullicundo, F. F., y Lamas, D. A. (2021), "El Backend se encarga de la manipulación de los datos; un BackEnd no sirve de mucho si no existe un FrontEnd de por medio. El desarrollador BackEnd debe conocer de bases de datos, frameworks y aspectos de seguridad" (s. p.).

XAMPP (En su versión de máquina virtual): XAMPP es un paquete de software que proporciona un entorno de desarrollo local para aplicaciones web. En su versión de máquina virtual, XAMPP se presenta como una máquina virtual preconfigurada que incluye servicios esenciales para el desarrollo web, como Apache para el servidor web, MySQL y/o MariaDB para la base de datos, y PHP para el procesamiento del lado del servidor. El autor (Bou, 2019) comenta que "XAMPP nos permite instalar un proyecto en local de nuestro PC, lo que pone a nuestra disposición un magnifico entorno de pruebas.", y que "XAMPP es ideal para aprender lenguajes como PHP o MySQL sin la necesidad de contratar ningún hosting". (p. 3).

- **MariaDB / MySQL**: XAMPP incluye un sistema de gestión de bases de datos que puede ser MySQL o MariaDB. Estos son sistemas de gestión de bases de datos relacionales ampliamente utilizados en el desarrollo web. El autor (Bou, 2019) comenta que "MySQL / MariaDB: con MySQL, XAMPP cuenta con uno de los sistemas relacionales de gestión de bases de datos mas populares del mundo. En combinación con el servidor web Apache y el lenguaje PHP, MySQL sirve para el almacenamiento de datos para servicios web. En las versiones actuales de XAMPP esta base de datos se ha sustituido por MariaDB, que es una ramificación ("Fork") del proyecto MySQL". (p. 4).
- **phpMyAdmin**: XAMPP integra phpMyAdmin, una herramienta de administración para gestionar bases de datos MySQL / MariaDB de manera gráfica a través de una interfaz web. El autor (Bou, 2019) comenta que "phpMyAdmin es una herramienta que está escrita en PHP con la intención de que podamos administrar MySQL / MariaDB a través de localhost". (p. 77).

IDE de Base de Datos: Actuando como un IDE de Base de Datos adicional a phpMyAdmin, Navicat facilita el diseño y la manipulación de bases de datos. El autor (Ozar, 2012) comenta que "Navicat es una herramienta GUI (de interfaz gráfica de usuario) que se utiliza para administrar todos los aspectos de un servidor MySQL, como la administración de herramientas visuales y un editor de código inteligente para codificar manualmente SQL y procedimientos almacenados". (s. p).

PHP: Lenguaje de programación principal para el desarrollo del back-end. El autor (Bagwan y Ghule, 2019) "A lo largo de los años, PHP se ha convertido en un lenguaje potente y popular para aplicaciones web. Sin un marco subyacente, las aplicaciones web basadas en PHP pueden volverse confusas y casi imposibles de ampliar o mantener". (p. 1).

Composer: Utilizado como gestor de bibliotecas en PHP. Según (Bagwan y Ghule, 2019) "La

idea de gestión de componentes se logra mejor utilizando el administrador de dependencias de PHP Composer". (p. 708).

DomPdf: Según Setiawan (Setiawan, 2017), el flujo de trabajo básico de esta biblioteca consiste en "convertir HTML a PDF y enviar el PDF resultante para mostrarlo en navegadores" (p. 19).

Eloquent ORM: El autor (Surguy, 2014) comenta que "El mapeo relacional de objetos (ORM) permite el acceso y la manipulación de datos dentro de una base de datos como un objeto PHP". (p. 9).

Artisan CLI: El autor (Surguy, 2014) comenta que "Artisan Command Line Tool, o Artisan CLI, es una herramienta auxiliar que viene con Laravel y podría usarse a través de la línea de comandos del sistema operativo para acelerar el desarrollo de una aplicación web". (p. 14).

MVC: El autor (Surguy, 2014) comenta que "Dichas aplicaciones utilizan el patrón de arquitectura de software Modelo-Vista-Controlador, que impone la separación entre la información (modelo), la interacción del usuario con la información (controlador) y la representación visual de la información (vista)". (p. 3).

Middleware: El autor (Razzaque, Milojevic-Jevric, Palade, y Clarke, 2015) comenta que "El middleware puede facilitar un proceso de desarrollo al integrar dispositivos informáticos y de comunicaciones heterogéneos y respaldar la interoperabilidad dentro de las diversas aplicaciones y servicios". (s. p.).

Policies: La fuente (Contributors, s.f.) comenta que "Las políticas son clases que le ayudan a organizar la lógica de autorización en torno a un recurso modelo". (p. 15), además que, "Una vez creadas, las políticas deben registrarse para ayudar a Laravel a saber qué políticas utilizar al autorizar acciones en los modelos". (p. 15).

HTTP: El autor (Fielding y cols., 1999) comenta que "El Protocolo de transferencia de hipertexto (HTTP) es un protocolo a nivel de aplicación para sistemas de información hipermedia distribuidos y colaborativos. Es un protocolo genérico, sin estado, que puede usarse para muchas tareas más allá de su uso para hipertexto, como servidores de nombres y sistemas de gestión de objetos distribuidos, mediante la extensión de sus métodos de solicitud, códigos de error y encabezados". (s. p.).

FTP: El autor (Gien, 1978) comenta que "Puede utilizarse para transferir archivos entre sistemas informáticos heterogéneos conectados por una red de comunicaciones". (s. p.).

SSH: El autor (Ylonen y Lonvick, 2006) comenta que "El protocolo Secure Shell (SSH) es un protocolo para el inicio de sesión remoto seguro". (p. 1.).

Docker: Utilizado para contenerizar la aplicación backend, simplificando el despliegue y garantizando consistencia en distintos entornos. Según el autor (Docker, 2020), "Docker fue diseñado para simplificar la creación, implementación y ejecución de aplicaciones utilizando contenedores" (p. 1). "El núcleo de Docker y sus contenedores virtualizados son aplicaciones agrupadas en imágenes, incluidas todas sus dependencias. Las imágenes contienen la aplicación y todo su entorno de ejecución" (p. 1).

Amazon Web Services (SSH): Según el autor (Bai, Jhaney, Wells, y cols., 2019), esta conexión por SSH en un terminal nos es útil "Para conectarnos de forma segura a nuestras instancias EC2, lo que permite el acceso remoto a una terminal a través de la cual se podían ingresar comandos de Linux para procesar datos" (p. 4).

Arquitectura de una aplicación web en Laravel

Laravel: Elegido como el marco de trabajo back-end para estructurar y organizar la aplicación web. Chen, X. (2017), comenta que "Laravel es un framework web PHP gratuito y de código abierto, creado por Taylor Otwell y destinado al desarrollo de aplicaciones web siguiendo el patrón arquitectónico modelo-vista-controlador (MVC). Algunas de las características de Laravel son un sistema de empaquetado modular con un administrador de dependencias dedicado". (p. 2).

- **Modelo**: Moreira, G. (2022), comenta acerca de este framework que "Posee un sistema de mapeo de datos relacional llamado Eloquent ORM que facilita la creación de modelos, mismo que está basado en un patrón active récord". (p. 7).
- **Vista**: Silva, H. (2017), comenta que "La vista se desliza hacia la interfaz del usuario que es desarrollado de acuerdo al modelo de la base de datos". (p. 38).
- **Controlador**: Álvarez, F. (2015), comenta que "El controlador es el encargado de actuar y responder a las solicitudes pedidas por el usuario, aplicando de forma correcta y óptima el proceso para que este muestre al cliente el modelo adecuado y muestre la vista que este solicitando". (p. 22).

2.3. Infraestructura

Martínez, J. (2023) ofrece una explicación completa sobre la infraestructura en el ámbito de desarrollo web y la nube. Se refiere a ella como "Dependiendo de la arquitectura del nuevo sistema este puede requerir uno o más servidores para hacer funcionar una sola instancia, dentro de esto se comprenden servidores de bases de datos, servidores de aplicacióny en ocasiones servidores de tareas especializadas adicionalespara el correcto funcionamiento del sistema". (p. 112).

Load Balancers: Implementados para distribuir el tráfico de aplicaciones entre varios destinos y zonas de disponibilidad. El autor (Arvindhan y Anand, 2019) comenta que "El componente de equilibrio de carga con métodos de escalado automático sería vital en las condiciones de computación en la nube. Con el objetivo final de evitar que el marco se sobrecargue y se bloquee, la aplicación y la actividad del cliente deben realizar un cálculo productivo". (s. p).

VPC: Servicio de computación en la nube comercial que brinda una nube privada virtual. El autor (Soni, 2018) comenta que "Le permite crear instancias en una red virtual aislada lógicamente". (p. 16).

Servicios de Storage (ej. S3): Empleados para el almacenamiento de archivos estáticos. El autor (Murty, 2008) comenta que "Ofrece espacio de almacenamiento seguro en línea para cualquier tipo de datos, proporcionando una alternativa a la creación, el mantenimiento y la copia

de seguridad de sus propios sistemas de almacenamiento. Hace que sus datos sean accesibles para cualquier otra aplicación o persona que usted permita desde cualquier lugar de la Web". (p. 12).

Servicios de Bases de datos relacionales (ej. RDS): (Salazar Cárdenas, 2014) comenta que "Amazon RDS es otro servicio web que apoya a una base de datos en la configuración, funcionamiento y escalado; todo esto en la nube. Se puede decir que es un entorno donde las bases de datos se encuentran asiladas y están en ejecución. Dentro de una instancia, el usuario pude crear una o varias bases de datos, utilizando herramientas o aplicaciones para su mantenimiento y gestión. Para el acceso a una base de datos que se encuentra en una instancia de AWS, se tiene que utilizar una aplicación de SQL estándar, ya que Amazon RDS no admite el acceso directo en su interfaz de host. Amazon RDS soporta diversas bases de datos como Oracle, PostgreSQL, MySQL y SQL Server, entre otros". (s. p).

Servicios de máquinas virtuales (ej. EC2): Acerca de este servicio, el autor (Murty, 2008) comenta que "hace posible ejecutar múltiples servidores Linux virtuales bajo demanda, proporcionando tantas computadoras como necesite para procesar sus datos o ejecutar su aplicación web sin tener que comprar o alquilar máquinas físicas. En EC2, usted tiene control total sobre cada servidor con acceso raíz al sistema operativo (el usuario raíz es el administrador final del sistema en máquinas Linux), un firewall configurable para administrar el acceso a la red y la libertad de instalar cualquier software que desee". (p. 2).

2.4. Front-end

El autor (Pérez Ibarra, Quispe, Mullicundo, y Lamas, 2021) comenta que "El FrontEnd trabaja la interfaz visual, y hace que el usuario pueda interactuar con nuestro sitio o sistema. Está orientado al lenguaje de marcas y al lenguaje de programación web de ejecución en equipos clientes". (s. p).

HTML: Según el autor (Gauchat, 2012) en cuanto a sitios web, "HTML se encarga de definir la estructura del documento"(s. p.).

CSS: Acerca de HTML, el autor (Gauchat, 2012) comenta que "CSS, el cual permite a los desarrolladores preparar el documento que se va a presentar en la pantalla", ya que de este proviene una mejor estructura y contenido que se le puede proporcionar al usuario (s. p.).

JavaScript: Según el autor (Gauchat, 2012), "JavaScript difiere de los demás lenguajes en que puede realizar tareas personalizadas, desde almacenar valores hasta calcular algoritmos complejos, incluida la capacidad de interactuar con los elementos del documento y procesar su contenido" todo esto en cuanto refieren los documentos en HTML y CSS, ya que "Al igual que HTML y CSS, JavaScript se incluye en los navegadores y, por lo tanto, se encuentra disponible en todos nuestros documentos" (p. 8).

Blade templates: Motor de plantillas en Laravel para la construcción de interfaces de usuario dinámicas. El autor (Surguy, 2014) comenta que "Laravel viene con un motor de plantillas simple pero potente llamado "Blade". Las plantillas hechas con Blade se parece mucho al HTML simple pero con algunas declaraciones especiales que hacen posible genere datos PHP, realice bucles sobre ellos, inyecte otras plantillas, use diferentes diseños y más". (p. 27).

Livewire: Según el autor (Pulido Fuentes, Martínez, y cols., 2022), "Livewire es un framework fullstack para el desarrollo de componentes Laravel que pueden comunicarse automáticamente entre la vista y el controlador, de modo que se produzcan comportamientos dinámicos sin usar JavaScript" (p. 19).

Bootstrap 3+: El autor (Spurlock, 2013) comenta que "Bootstrap es un producto de código abierto de Mark Otto y Jacob Thornton quienes, cuando se lanzó inicialmente, ambos eran empleados de Twitter. Era necesario estandarizar los conjuntos de herramientas frontend de los ingenieros de toda la empresa", y que "En los primeros días de Twitter, los ingenieros utilizaban casi cualquier biblioteca con la que estaban familiarizados para cumplir con los requisitos del front-end. Las inconsistencias entre las aplicaciones individuales dificultaron su escalamiento y mantenimiento. Bootstrap comenzó como una respuesta a estos desafíos y se aceleró rápidamente durante la primera Hackweek de Twitter". (p. 1).

npm: El autor (Abdellatif, Zeng, Elshafei, Shihab, y Shang, 2020) comenta que "Sistema de gestión de paquetes por defecto para Node.js, utilizado para gestionar dependencias front-end. El ecosistema npm es un sistema de gestión de paquetes Node.js (JavaScript) que contiene más de 700.000 paquetes Node.js y, para ayudar a los desarrolladores a encontrar paquetes de alta calidad que satisfagan sus necesidades". (s. p).

Google Chrome: El autor (PK y Jevitha, s.f.) comenta que "Navegador web principal para el desarrollo y pruebas de la aplicación. Google Chrome es el navegador web más popular disponible en la actualidad" (p. 1).

Chrome DevTools: Utilizado para inspeccionar el HTML, renderizado (DOM) y actividad de red de las páginas. El autor (Abudureheman, 2017) comenta que "Chrome DevTool se utiliza para recopilar información sobre eventos de carga de páginas. El objetivo principal es comparar la frecuencia de carga de la página y los momentos en que las páginas cambian" (p. 1).

2.5. Pruebas

Pruebas de caja blanca: En cuanto a esta clase de pruebas, el autor (Saucedo y del Castillo Serpa, 2012) comenta que "Las técnicas de caja blanca, examinan la parte interna del programa, siempre se está observando el código, y los casos de prueba están basados en la estructura interna del programa" (p. 23).

Pruebas de caja negra: En cuanto a esta clase de pruebas, el autor (Saucedo y del Castillo Serpa, 2012) comenta que "La técnica de caja negra, se enfoca en probar el sistema sin tomar en cuenta la estructura interna del mismo, su objetivo es validar que las salidas sean las esperadas" (p. 27).

2.6. Metodología ágil

En el desarrollo de la presente tesis, se ha adoptado la metodología ágil Scrum como marco de trabajo. Scrum, ampliamente reconocido en la industria del desarrollo de software, proporciona

un enfoque iterativo e incremental que se alinea perfectamente con los objetivos de este estudio. Según Schwaber y Sutherland (2017), es "un marco dentro del cual las personas puedan abordar problemas adaptativos complejos, al mismo tiempo que entregar de manera productiva y creativa productos del mayor valor posible" (p. 3). Esta elección metodológica se sustenta en la necesidad de gestionar eficazmente el desarrollo de un sistema contable distribuido, aprovechando los principios ágiles para asegurar la entrega de funcionalidades valiosas de manera oportuna y adaptativa. A través de la aplicación de Scrum, se busca no solo cumplir con los objetivos de este proyecto, sino también contribuir al creciente cuerpo de conocimiento en el ámbito de sistemas contables y el desarrollo de software.

Según el Manifiesto Ágil (Beck y cols., 2001), los doce principios ágiles incluyen:

1. **Nuestra máxima prioridad es satisfacer al cliente mediante la entrega temprana y continua de software valioso**. Según Rad y Turley (2019), "Al fin y al cabo, esto es un negocio y necesitamos que los clientes estén felices. Entonces, ¿cómo les satisfacemos? Por el software que creamos, que tiene el potencial de generarle valor (por ejemplo, dinero)" (p. 9).

2. **Bienvenidos los requisitos cambiantes, incluso en las últimas etapas del desarrollo. Los procesos ágiles aprovechan el cambio para lograr la ventaja competitiva del cliente**. Los cambios son bienvenidos en cualquier etapa del proyecto.

3. **Entregar software que funcione con frecuencia, desde un par de semanas hasta un par de meses, dando preferencia al plazo más corto**. Se entrega frecuentemente, en períodos de tiempo cortos.

4. **Los empresarios y los desarrolladores deben trabajar juntos diariamente durante todo el proyecto**. Según Rad y Turley (2019), "Esto va en contra de la idea de separar a los responsables de negocios (clientes u otros) de los técnicos, lo cual sigue siendo un problema en proyectos hoy en día" (p. 10).

5. **Construya proyectos en torno a personas motivadas. Bríndeles el entorno y el apoyo que necesitan y confíe en ellos para hacer el trabajo**. Se debe dar un buen entorno y confianza al equipo de desarrollo.

6. **El método más eficiente y eficaz para transmitir información hacia y dentro de un equipo de desarrollo es la conversación cara a cara**. Es el método más eficiente y efectivo de la comunicación con el equipo.

7. **El software funcional es la principal medida del progreso**. Según Rad y Turley (2019), "La mayoría de los proyectos miden conceptos equivocados. Es un problema de base porque lo que se mide es lo que se obtiene" (p. 11).

8. **Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, desarrolladores y usuarios deberían poder mantener un ritmo constante indefinidamente**. Según Rad y Turley (2019), "Nada de horas extras adicionales antes de las entregas. Se trata de maximizar el valor a largo plazo. No se trata de ganancias a corto plazo que puedan disminuir la productividad y la calidad" (p. 11).

9. **La atención continua a la excelencia técnica y al buen diseño mejora la agilidad**. Como son entregas iterativas se puede ver que se hizo bien y qué se puede mejorara para la próxima iteración.

10. **La simplicidad (el arte de maximizar la cantidad de trabajo no realizado) es esencial**. Según Rad y Turley (2019), "Es bueno simplificar la solución y tener sólo las funcionalidades que son realmente útiles porque ahorra tiempo y dinero (que se pueden utilizar para otros proyectos) y reduce el coste de mantenimiento" (p. 12).

11. **Las mejores arquitecturas, requisitos y diseños surgen de equipos autoorganizados**. Las mejores arquitecturas y diseños emergen de equipos autoorganizados (el equipo decide por sí mismo cómo trabajar, cómo hacer las cosas y quién va hacer las cosas).

12. **A intervalos regulares, el equipo reflexiona sobre cómo ser más eficaz y luego ajusta y ajusta su comportamiento en consecuencia**. El equipo es capaz de ajustar y perfeccionar su comportamiento.

Capítulo 3

Estado del arte

Durante este capítulo del estado del arte, se decidió especificar desde el concepto de un sistema contable, su relación en el ámbito de tecnologías de la información, abordando aspectos clave como la integración de herramientas digitales, la automatización de procesos contables y la optimización del flujo de información financiera.

3.1. Sistema contable

Para comprender el concepto de un sistema contable, (Josar, 2011) comenta que "Un sistema de información contable comprende los métodos, procedimientos y recursos utilizados por una entidad para llevar un control de las actividades financieras y resumirlas en forma útil para la toma de decisiones. La información contable se puede clasificar en dos grandes categorías: la contabilidad financiera o la contabilidad externa y la contabilidad de costos o contabilidad interna. La contabilidad financiera muestra la información que se facilita al público en general, y que no participa en la administración de la empresa, como son los accionistas, los acreedores, los clientes, los proveedores, los analistas financieros, entre otros, aunque esta información también es de mucho interés para los administradores y directivos de la empresa. Esta contabilidad permite obtener información sobre la posición financiera de la empresa, su grado de liquidez y sobre la rentabilidad de la empresa". (p. 27).

(Navarro Silva, López Macas, y Pérez Espinosa, 2017) comenta que "El control interno Contable, comprende el plan de organización y todos los métodos y procedimientos cuya misión es salvaguardar los activos y la fiabilidad de los registros financieros; y el sistema contable de una empresa es el conjunto de registros, procedimientos y equipos que rutinariamente trata con los eventos que afectan su desempeño y posición financieros. El sistema mantiene la contabilidad de los activos, pasivos y patrimonio de la empresa". (p. 47).

(Sánchez, Matamoros, Hidalgo, Gutiérrez, y Lozada, 2019) comenta que "En el vocabulario popular de México, Bolivia, Colombia, República Dominicana y América Central, "abarrotes" es el término con el que se le conoce a las tiendas en las que se comercializan productos comestibles, así como otros insumos de primera necesidad.Este tipo de comercios en México, se han convertido

en negocios de primera necesidad y, por lo tanto, redituables. De tal manera que los analistas aseguran que las tiendas se están yendo a poblaciones donde tradicionalmente no tenían presencia. Las tiendas de abarrotes han adquirido notable importancia en cualquier rincón del país, ya que tienen una importante participación, conformando un fuerte esquema en el comercio mexicano, dentro del esquema detallista, que marcó la pauta para que en Méxicolos tenderos se transformaran en abarroteros, y sus negocios en las tiendas de abarrotes. Sin lugar a dudas, este tipo de negocios marcan una diferencia al otorgar un mejor formato y llevar a cabo ventas adicionales de productos lo que representa mayores ingresos para los dueños de las mismas, pues se centran en la calidad de servicio y un trato más directo con sus clientes, lo que también las hace más rentables". (p. 85).

3.2. Tecnologías de la información

Cuando se consideraron sistemas relacionados con la contabilidad de inventarios para abarrotes, se prestó atención a seguir fielmente los principios de administración de productos. Esto se hizo con la comprensión de que, aunque el enfoque principal no necesariamente sería en abarrotes, seguir estos principios proporcionaría una visión más completa en términos de contabilidad de productos en un almacén. Por ejemplo, se exploró el caso del Centro Automotriz Cuenca Llantas, que se dedica a la venta al por menor y al por mayor de accesorios, partes y piezas de vehículos automotores, el tesista (Guevara, 2017) comenta que a través de una aplicación web se consigue "una información sistematizada para un mejor funcionamiento, contribuyendo a la tecnificación y automatización de procesos que actualmente necesitan atención y que deben incorporarse al globalizado mundo", en su estudio resalta que con esto, "Se identificaron las necesidades de la empresa en el control del inventario a través del análisis de sus procesos, eliminando procesos redundantes que se realizaban de forma manual que retrasaban las operaciones de la empresa, solventándose con la automatización de las entradas y salidas de mercadería" (p. 108).

El estudio realizado por (Guevara, 2017) también destaca que, entre los principales requisitos para implementar un sistema contable eficaz para la gestión de productos en un negocio, se encuentra el acceso, "El sistema debe permitir que el administrador pueda crear, buscar y modificar usuarios, para que puedan tener acceso al sistema web", al almacén, "Dentro de la opción almacén del sistema web el usuario autorizado podrá realizar la creación de artículos, buscar, modificar, activarlos e inactivarlos para su uso. Dentro de la opción almacén del sistema web el usuario autorizado podrá realizar la creación de categorías, buscar, modificar y eliminar la misma", mantenimiento, "Se encontrara la definición de los tipos de usuarios que maneja el sistema para su acceso", Ventas, "Dentro de la opción ventas del sistema web el usuario autorizado podrá realizar la salida de la mercadería, para darle movimiento al inventario por motivos de ventas", Reportes, "El sistema web permitirá generar reportes de catálogo de clientes, proveedores, stock de productos con su respectivo precio de venta actualizado, productos valorizados, diario de compras". (p. 41).

El sistema descrito por (Guevara, 2017) comprende tres roles diferentes: el Administrador, responsable de generar reportes detallados sobre clientes, proveedores y el stock de productos, así como valorizar los productos y mantener un registro diario de compras; el Usuario Supervisor de Bodega, quien inicia sesión para gestionar productos y categorías, crear proveedores, registrar

productos por categorías y generar reportes sobre el stock de productos, el diario de compras y productos valorizados; y finalmente, el Usuario General Vendedor, cuya función principal consiste en visualizar el stock de productos y los precios de venta.

Según (INEGI, 2019), "Cerca del 80 por ciento de los minisúper que sí cuentan con un control de gastos e ingresos, instrumentaron registros contables.", "Un 20 por ciento no lleva un registro contable". (p. 4).

(Josar, 2011) comenta que "El sistema contable de cualquier empresa independientemente del sistema contable que utilice, se deben ejecutar tres pasos básicos utilizando relacionada con las actividades financieras", "1. Registro de la actividad financiera: en un sistema contable se debe llevar un registro sistemático de la actividad comercial diaria en términos económicos.", "2. Clasificación de la información: un registro completo de todas las actividades comerciales implica comúnmente un gran volumen de datos, demasiado grande y diverso para que pueda ser útil para las personas encargadas de tomar decisiones. Por tanto, la información de debe clasificar en grupos o categorías.", 3. Resumen de la información: para que la información contable utilizada por quienes toman decisiones, esta debe ser resumida.". (p. 4).

Capítulo 4

Metodología

Durante este capítulo se describe la metodología, esta se conoce comúnmente como el ciclo de vida del desarrollo de software o SDLC (por sus siglas en inglés "Software Development Life Cycle"). El SDLC es una metodología ampliamente utilizada en la industria del software y se divide en fases que van desde la planificación y el análisis hasta la implementación y el mantenimiento.

4.1. Análisis

En la etapa de análisis del ciclo de vida del desarrollo de software (SDLC) para el diseño de un sistema contable orientado al rubro de abarrotes, se realizó una investigación detallada centrada en comprender a fondo las particularidades y requerimientos específicos de este sector.

En una primera fase, se llevó a cabo un estudio exhaustivo de los sistemas contables actualmente utilizados por pequeñas y medianas empresas de abarrotes. Este análisis permitirá identificar áreas de mejora, como la optimización en la gestión de inventario y la necesidad de reportes financieros más adaptados a las operaciones diarias de este tipo de negocios.

Requerimientos del sistema

Desarrollar una aplicación web para la gestión contable en el sector de abarrotes, tiene como objetivo principal facilitar las operaciones simultáneas desde diversas locaciones. Esta solución debe incluir funcionalidades de login, como inicio y cierre de sesión, para establecer una base de seguridad del sistema. Las funciones de gestión abarcarán áreas críticas como categorías, productos, denominaciones, usuarios, roles y permisos. La gestión del carrito en ventas facilitará la contabilización del total a pagar, el número de artículos y permitirá seleccionar denominaciones o efectivo recibido para calcular el cambio a entregar al cliente de manera un poco más rápida. Además, la capacidad de búsqueda en todas las gestiones, excepto en la gestión de permisos por rol. Las funcionalidades de información contabilizada incluirán un corte de caja por rango de fechas, proporcionando detalles de ventas por usuario/s, y un reporte de ventas básico con opciones de descarga en PDF o Excel, complementado con un resumen mediante el detalle de las ventas, este integrado directamente en la interfaz. El sistema debe permitir el acceso desde diversas locaciones y

soportar operaciones simultáneas, asegurando una gestión remota, estas configuraciones contemplan configurar un sistema de información distribuido. La seguridad del sistema estará garantizada mediante autenticación y autorización, y protección de datos. La interfaz de usuario debe ser intuitiva y accesible, facilitando su adopción por el personal de abarrotes, previniendo así que el usuario no tenga que adoptar un perfil técnico en específico. El sistema debe ser escalable para soportar un incremento en el número de usuarios y transacciones, y fácil de mantener y actualizar. Estos requerimientos resultarán en una mejora significativa en la eficiencia operativa, seguridad, flexibilidad, y experiencia del usuario, proporcionando información precisa y oportuna que apoye la toma de decisiones informadas en el sector de abarrotes.

1. **Ánalisis de requerimientos**: Para llevar a cabo la identificación de requerimientos funcionales (en conjunto con su respectiva historia de usuario) y no funcionales, basándose en la investigación previa, se realizó un análisis de las necesidades específicas de los negocios de abarrotes. Este análisis incluyó la revisión de 7 documentaciones técnicas. El autor (Luján-Mora, 2002) comenta que "Las historias de usuario se usan, en el contexto de la ingeniería de requisitos ágil, como una herramienta de comunicación que combina las fortalezas de ambos medios: escrito y verbal. Describen, en una o dos frases, una funcionalidad de software desde el punto de vista del usuario, con el lenguaje que éste emplearía. El foco está puesto en qué necesidades o problemas soluciona lo que se va a construir. Su origen viene de la metodología eXtremeProgramming (programación extrema, abreviado normalmente como XP), donde las historias de usuario deben ser escritas por los clientes." (p. 4). El autor (Luján-Mora, 2002) comenta que "Las historias de usuario son una herramienta que agiliza la administración de requisitos, reduciendo la cantidad de documentos formales y tiempo necesarios. Forman parte de la fórmula de captura de funcionalidades definida en 2001 por Ron Jeffries de las tres Cs." (p. 5). El autor (Luján-Mora, 2002) comenta que estas son "**Card**, **Conversation** y **Confirmation**" (p. 5), "**Card**" se encargaría de que cada historia de usuario se reduzca hasta hacerla fácil de memorizar y sindetizar en una tarjeta. "**Consersation**" se encargaría de agregar criterios de aceptación por parte del equipo de desarrollo y el propietaro del producto a cada historia poco antes de su implementación, no es necesario profundizar aquí y los cambios son bienvenidos. "**Confirmation**" de que el propiestario del producto o usuario de negocio connfirma que el equipo de desarrollo ha entendido y recogido correctamente sus requisitos revisando los criterios de aceptación señadados.

   Los requerimientos identificados que pueden implementarse en un sistema, con el fin de mejorar la comunicación entre estos roles y agilizar dichos procesos, estos están plasmados en historias de usuario, en Cuadro 4.1 se detallan:

Historia de usuario	Descripción
HU-001	Gestionar categorías para organizar productos.
HU-002	Gestionar productos para mantener inventario.
HU-003	Gestionar denominaciones para manejo de efectivo.
HU-004	Gestionar usuarios para control de acceso.
HU-005	Gestionar roles y permisos de usuarios.
HU-006	Gestionar carrito de ventas para contabilizar total y artículos.
HU-007	Seleccionar denominaciones o efectivo para calcular cambio.
HU-008	Buscar información dentro de módulos.
HU-009	Generar corte de caja por rango de fechas.
HU-010	Generar reporte de ventas con descarga en PDF/Excel.
HU-011	Ver resumen de ventas en la interfaz.
HU-012	Acceder al sistema desde diversas locaciones.

Cuadro 4.1: Historia de usuario.

1.1 **Requerimientos funcionales**: El autor (Sommerville, 2005) comenta que "Son declaraciones de los servicios que proveerá el sistema". (s. p). (Glinz, 2007) comenta que "son aquellos requisitos que especifican las entradas (estímulos) del sistema, las salidas (respuestas) del sistema y las relaciones de comportamiento entre ellos". (p. 1).

En Cuadro 4.2 se puede observar cómo se detallan las historias de usuario en conjunto con estas declaraciones previamente analizadas:

Requerimiento funcional	Historia de Usuario	Módulo
Registro de entrada y salida de productos	Gestión de categorías, Gestión de productos, Buscar información dentro de módulos	Módulo de productos y/o Módulo de categorias
Control de stock	Gestión de productos, Generar reporte de ventas	Módulo de reporte de ventas
Registro de transacciones de ventas	Gestionar carrito de ventas, Gestión de denominaciones, Realizar búsquedas, Seleccionar denominaciones, Ver resumen de ventas en la interfaz, Realizar búsquedas (Productos)	Módulo de ventas, Módulo de denominaciones
Manejo de carritos de compras	Gestionar carrito de ventas	Módulo de ventas

Procesamiento de pagos (efectivo)	Gestionar carrito de ventas, Seleccionar denominaciones	Módulo de ventas, Módulo de denominaciones
Creación de reportes de ventas diarios, semanales y mensuales	Generar corte de caja por rango de fechas, Generar reporte de ventas, Ver resumen de ventas en la interfaz	Módulo de corte de caja y/o Módulo de reporte de ventas
Balance general y estado de resultados	Generar corte de caja por rango de fechas, Generar reporte de ventas	Módulo de corte de caja y/o Módulo de reporte de ventas
Exportación de reportes en formato PDF y Excel	Generar reporte de ventas	Módulo de reporte de ventas
Creación y administración de perfiles de usuario	Gestionar usuarios, Buscar información dentro de módulos	Módulo de usuarios
Asignación de roles y permisos	Gestión de roles y permisos de usuarios, Realizar búsquedas (Roles)	Módulo de permisos y/o Módulo de roles
Autenticación y autorización seguras para escalar en el rubro	Acceder al sistema desde diversas locaciones	Módulo de autenticación
Buscar información dentro de módulos	Realizar búsquedas	Módulo de productos, Módulo de categorias, Módulo de ventas, Módulo de denominaciones, Módulo de usuarios, Módulo de roles, Módulo de permisos.

Cuadro 4.2: Requerimientos funcionales.

1.2 **Requerimientos no funcionales**: El autor (Rojo, 2012) comenta que en su estudio comenzó presentando acerca del estado de arte de los conceptos que definen a los Requerimientos No Funcionales dentro de la literatura existente en la Ingeniería de Requerimientos "quedo expuesto que la carencia de consenso surge a partir de la diversidad de términos utilizados en las definiciones, con significados poco claros, que dan lugar a la ambigüedad sobre su alcance o lo que representan los mismos; las discrepancias conceptuales también se encontraron en

las clasificaciones propuestas para Requerimientos No Funcionales, que incluyen árboles o categorías de conceptos que se muestran en forma separada en otras definiciones" (p. 88), además responde a las preguntas; ¿Se contemplan los Requerimientos No Funcionales en las metodologías de aplicaciones Web?, ¿Utilizan técnicas de la ingeniería de requerimientos para el tratamiento de los Requerimientos No Funcionales? "Si, a pesar de la carencia de consenso en su significado. En los enfoques estudiados se puede ver que no hay técnicas especificas para la elicitación de Requerimientos No Funcionales, no hay consenso sobre como especificarlos y tampoco lineamientos de como validarlos" (p. 89). Sin embargo, en este estudio, se toman en cuenta los Requerimientos No Funcionales, aunque para fines de practicidad, el enfoque principal está en el estudio y desarrollo de la etapa de pruebas. En Cuadro 4.3 se especifican las necesidades a desarrollar no funcionales.

Requerimiento No Funcional	**Descripción**	**Categoría**
Acceso desde diversas locaciones	El sistema debe soportar operaciones simultáneas desde múltiples ubicaciones, facilitando la gestión remota y distribuida.	Distribución y Operaciones Simultáneas
Autenticación y Autorización	El sistema debe asegurar que solo los usuarios autorizados puedan acceder a las funciones y datos según sus roles y permisos asignados.	Seguridad
Interfaz de usuario intuitiva	El sistema debe proporcionar una interfaz de usuario que sea intuitiva y fácil de usar.	Usabilidad y Accesibilidad
Rendimiento	El sistema debe responder rápidamente a las solicitudes de los usuarios y manejar eficientemente grandes volúmenes de datos.	Rendimiento y Escalabilidad
Escalabilidad	Debe ser escalable para soportar un incremento en el número de usuarios y transacciones.	Rendimiento y Escalabilidad
Facilidad de mantenimiento	Debe ser fácil de mantener y actualizar, permitiendo la corrección de errores y la implementación de nuevas funcionalidades.	Mantenimiento y Actualizaciones

Cuadro 4.3: Requerimientos no funcionales.

2. **Flujo de Trabajo y Operaciones Comerciales**:

2.1 **Mapeo de Procesos:** Se tuvo un análisis de los flujos de trabajo actuales y deseados en las tiendas de abarrotes. Esto incluyó la recepción de mercancías, gestión de inventarios,

procesos de venta y cierre de caja. A continuación se presenta un resumen de los principales procesos identificados:

- Recepción de Mercancías: Los productos son recibidos de los proveedores y se registran en el sistema. En esta etapa, se verifica la calidad y cantidad de los productos antes de agregarlos al inventario. Este proceso asegura que los productos recibidos cumplen con los estándares requeridos.

- Gestión de Inventarios: Se actualiza el inventario con cada transacción de entrada o salida de productos. Se generan alertas de inventario bajo para productos con existencias reducidas.

- Procesos de Venta: Los clientes seleccionan los productos y los agregan al carrito de compras. Se procesan los pagos y se generan facturas para las ventas realizadas.

- Cierre de Caja: Al final del día, se realiza el cierre de caja para calcular el total de ventas y el efectivo disponible. Se generan reportes financieros para resumir las transacciones del día.

2.2 **Optimización de Operaciones:**

Durante el análisis, se identificaron varias áreas de mejora en los flujos de trabajo actuales. Estas áreas incluyen:

- Simplificación de procesos para mejorar la eficiencia operativa: La simplificación de los procesos operativos es esencial para mejorar la eficiencia. Esto implica eliminar pasos innecesarios y reestructurar los procedimientos para hacerlos más directos y fáciles de seguir.

- Automatización de tareas repetitivas para reducir errores humanos: Implementar la automatización en tareas repetitivas ayuda a reducir la posibilidad de errores humanos.

- Mejora de la comunicación interna y externa para agilizar los procesos: Fortalecer la comunicación dentro de la empresa y con proveedores y clientes es crucial para agilizar los procesos, fomentando la colaboración entre equipos y estableciendo tareas claras.

Estas mejoras no solo optimizan las operaciones diarias, sino que también contribuyen a un mejor control y manejo del negocio, resultando en una mayor satisfacción del cliente y un aumento en la rentabilidad.

3. **Planificación**:

Scrum menciona la cantidad de Sprints, así como el tiempo de duración de cada uno. Para el presente proyecto se establecieron 11 Sprints, los cuales tuvieron la siguiente duración, esto según Cuadro 4.4:

Sprint / Módulo	Número de HU	Duración
Módulo de productos	HU001, HU008	Dos semanas
Módulo de categorias	HU002, HU008	Dos semanas
Módulo de denominaciones	HU003, HU008	Una semana
Módulo de ventas	HU006, HU007, HU008, HU011	Tres semanas y media
Módulo de usuarios	HU004, HU008	Una semana
Módulo de roles	HU005, HU008	Una semana
Módulo de permisos	HU005	Una semana y media
Módulo de corte de caja	HU009	Una semana y media
Módulo de reporte de ventas	HU010, HU011	Dos semanas y media
Módulo de autenticación	HU012	Una semana o menos
Configuración de ambiente	HU012	Dos semanas

Cuadro 4.4: Tabla de Sprints y Módulos

Se ha logrado definir 12 historias de usuario. La finalidad de esta fase es clasificar los criterios de aceptación, midiendo el esfuerzo necesario que se tomará para el desarrollo de cada historia. Para medir el nivel esfuerzo de Sprint se realiza una tabla de escalas, esto detallado en Cuadro 4.5.

Número	Nivel de esfuerzo	Descripción
1	Obligatorio	Primer nivel
2	Requerido	Segundo nivel
3	Bajo	Tercer nivel

Cuadro 4.5: Niveles de esfuerzo

Sprint I: Contiene las historias de usuario asignada por el Product owner acerca del módulo (Módulo de productos), esto detallado en Cuadro 4.6.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Registro y salida de productos	Requerido	1	Módulo de productos
2	Buscar información dentro de módulos	Bajo	3	Módulo de productos

Cuadro 4.6: Requerimientos de Sprint I

Sprint II: Contiene las historias de usuario asignada por el Product owner acerca del módulo (Módulo de categorías), esto detallado en Cuadro 4.7.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Registro de transacciones de ventas	Obligatorio	1	Módulo de categorías
2	Buscar información dentro de módulos	Requerido	2	Módulo de categorías

Cuadro 4.7: Requerimientos de Sprint II

Sprint III: Contiene las historias de usuario asignada por el Product owner acerca del módulo (Módulo de denominaciones), las búsquedas son para facilitar la gestión, esto detallado en Cuadro 4.8.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Registro de transacciones de ventas	Obligatorio	1	Módulo de denominaciones
2	Buscar información dentro de módulos	Requerido	2	Módulo de denominaciones

Cuadro 4.8: Requerimientos de Sprint III

Sprint IV: Contiene las historias de usuario asignada por el Product owner acerca del módulo (Módulo de ventas), esto detallado en Cuadro 4.9.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Registro de transacciones de ventas	Obligatorio	1	Módulo de ventas
2	Manejo de carritos de compras	Obligatorio	1	Módulo de ventas (Carrito de ventas)
3	Procesamiento de pagos (efectivo)	Obligatorio	1	Módulo de denominaciones (Cálculo de montos)
4	Buscar información dentro de módulos	Obligatorio	1	Módulo de ventas (Carrito de ventas), Módulo de denominaciones (Cálculo de montos)

Cuadro 4.9: Requerimientos de Sprint IV

Sprint V: Este sprint contiene las historias de usuario asignadas por el Product Owner para el quinto módulo (Módulo de usuarios), el cual permite capturar datos relevantes de la entidad misma, esto detallado en Cuadro 4.10.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Creación y administración de perfiles de usuario	Obligatorio	1	Módulo de usuarios
2	Buscar información dentro de módulos	Requerido	2	Módulo de usuarios

Cuadro 4.10: Requerimientos de Sprint V

Sprint VI: Este módulo es sencillo, permite que un usuario apunte a un registro (módulo de roles), esto detallado en Cuadro 4.11.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Asignación de roles y permisos	Obligatorio	1	Módulo de roles
2	Buscar información dentro de módulos	Requerido	2	Módulo de roles

Cuadro 4.11: Requerimientos de Sprint VI

Sprint VII: Las historias de usuario asignadas por el Product Owner incluyen el séptimo módulo (Módulo de permisos). Este módulo, aunque actualmente no cuenta con una función de caja búsqueda, incluirá un sistema de filtrado para especificar qué permisos pertenecen a cada rol y permitirá la sincronización correspondiente, esto detallado en Cuadro 4.12.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Asignación de roles y permisos	Obligatorio	1	Módulo de permisos

Cuadro 4.12: Requerimientos de Sprint VII

Sprint VIII: Contiene las historias de usuario asignada por el Product owner acerca del módulo (Módulo de corte de caja), esto detallado en Cuadro 4.13.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Creación de reportes de ventas diarios, semanales y mensuales	Obligatorio	1	Módulo de corte de caja, Módulo de reporte de ventas
2	Balance general y estado de resultados	Bajo	3	Módulo de corte de caja, Módulo de reporte de ventas

Cuadro 4.13: Requerimientos de Sprint VIII

Sprint IX: Las historias de usuario asignadas por el Product Owner incluyen el noveno módulo (Módulo de reporte de ventas). Lo fundamental aquí es proveer la exportación de reportes en formato PDF y Excel. El objetivo es extraer la información necesaria, filtrarla y analizar los resultados. No se busca explícitamente proporcionar un balance general y estado de resultados, ya que estos

pueden variar según los objetivos de cada cliente o dueño. Sin embargo, se pretende ser el punto de entrada para agregar funcionalidades específicas según las necesidades, esto detallado en Cuadro 4.14.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Exportación de reportes en formato PDF y Excel	Obligatorio	1	Módulo de reporte de ventas
2	Creación de reportes de ventas diarios, semanales y mensuales y estado de resultados	Requerido	2	Módulo de ventas (Carrito de ventas)
3	Balance general y estado de resultados	Bajo	3	Módulo de ventas (Carrito de ventas)

Cuadro 4.14: Requerimientos de Sprint IX

Sprint X: Las historias de usuario asignadas por el Product Owner para este sprint incluyen el último módulo (Módulo de autenticación). Este módulo tiene diversas capas, lo que puede hacer que sea abstracto especificar cada una de las proporcionadas por Laravel y Spatie. Sin embargo, se mencionan algunas de las más importantes, esto detallado en Cuadro 4.15.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Autenticación y autorización seguras para escalar en el rubro	Obligatorio	1	Módulo de autenticación

Cuadro 4.15: Requerimientos de Sprint X

Sprint XI: Las historias de usuario asignadas por el Product Owner para este sprint abordan las configuraciones de despliegue. Se detallan los accesos necesarios en el archivo .env de Laravel, el proceso de contenerización de software mediante Dockerfile y las opciones a seleccionar en la consola de Amazon Web Services (AWS) a grandes rasgos, esto detallado en Cuadro 4.16.

N°	Requerimiento funcional	Criterio de prioridad	Nivel de esfuerzo	Módulo
1	Autenticación y autorizaciones seguras para escalar en el rubro	Obligatorio	1	Configuraciones de despliegue

Cuadro 4.16: Requerimientos de Sprint XI

4.2. Diseño

4.2.1. Diseño de alto nivel

En la etapa de análisis del ciclo de vida del desarrollo de software (SDLC), el diseño de alto nivel se enfoca en establecer la arquitectura general y los componentes principales del sistema. Este diseño proporciona una visión amplia de cómo funcionará el sistema en su conjunto y cómo interactuarán sus diferentes partes.

Módulos del sistema

Después de haber elaborado el análisis de los requerimientos, se puede especificar el diagrama modular, y cómo interactúan estos módulos entre sí según los roles de usuarios ya especificados. En la Figura 4.1 se detalla el diagrama modular.

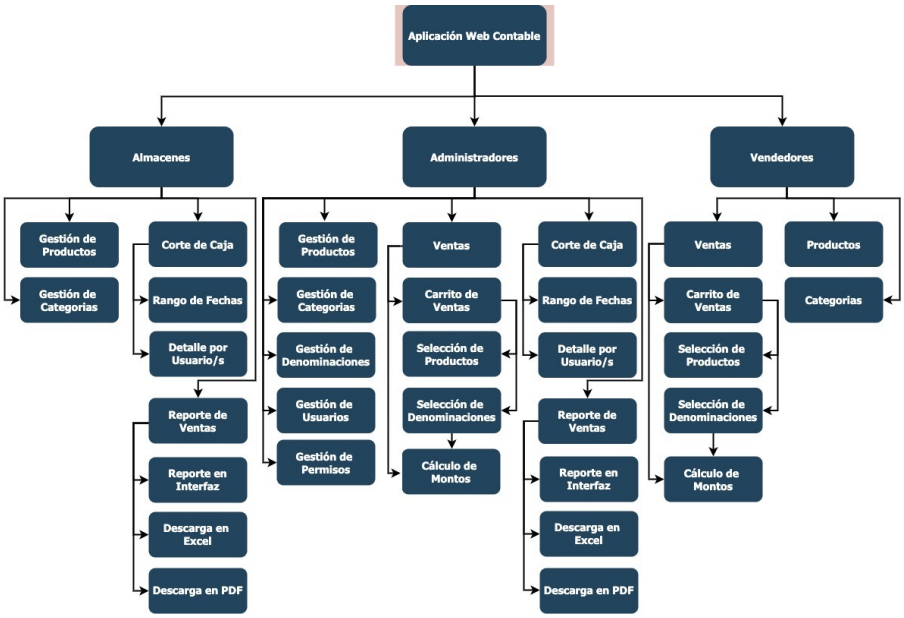

Figura 4.1: Diagrama de modulos.

Es importante mencionar que en el desarrollo del diagrama de módulos mencionado anteriormente, se presupone la integración de un sistema de autenticación adaptado a los roles de usuario previamente definidos. Para esta implementación, se utilizó la biblioteca Spatie de Laravel, la cual cumple con nuestras necesidades específicas, como se detalla en la sección correspondiente de bajo nivel.

4.2.2. Diseño de bajo nivel

En la fase de diseño de bajo nivel, se profundiza en los detalles específicos de implementación de los componentes identificados en el diseño de alto nivel. Se definen los algoritmos, estructuras de datos y métodos que se utilizarán en cada módulo del sistema.

Base de Datos

Durante los dos años de experiencia desarrollando Back-end al momento de redactar, se ha comprendido la importancia de identificar los modelos de un sistema al especificar un diseño desde bajo nivel. Esta práctica se convierte en una necesidad fundamental para garantizar la claridad y coherencia en el desarrollo del proyecto. En la Figura 4.2 se detalla el diagrama de Base de Datos relacional.

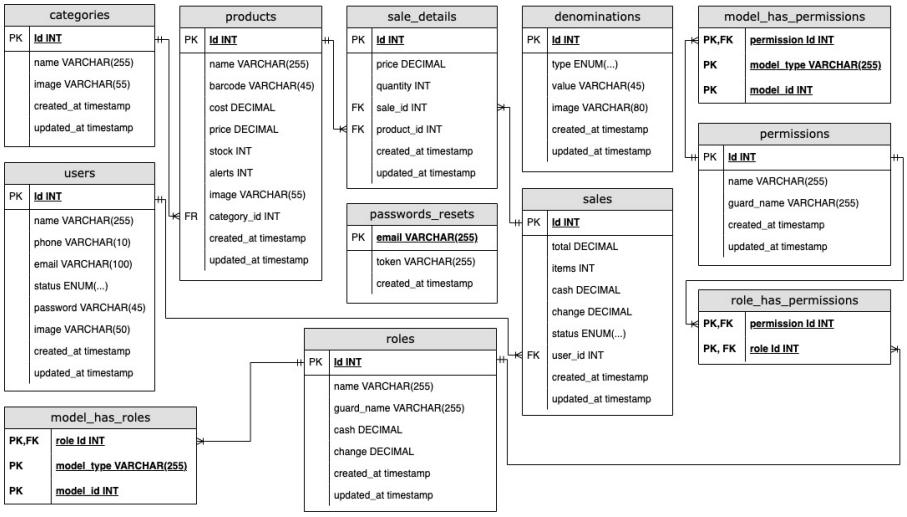

Figura 4.2: Diagrama entidad relación.

1. **Category**
   - **Descripción**: Este modelo representa las categorías de los productos en el sistema. Las categorías permiten organizar los productos en grupos lógicos para facilitar su gestión y búsqueda.
2. **Denomination**
   - **Descripción**: Este modelo se utiliza para gestionar las denominaciones de las monedas o unidades de medida empleadas en el sistema contable. Es crucial para mantener la consistencia en las transacciones financieras.
3. **Product**
   - **Descripción**: Representa los productos disponibles en el sistema. Este modelo incluye información detallada sobre cada producto, como el nombre, precio, descripción y la categoría a la que pertenece.
4. **Sale**
   - **Descripción**: Gestiona las ventas realizadas en el sistema. Incluye detalles como el cliente, la fecha de la venta y el monto total. Este modelo es esencial para registrar y seguir las transacciones comerciales.
5. **SaleDetails**

- **Descripción**: Este modelo contiene los detalles de cada venta, especificando los productos vendidos, las cantidades y los precios individuales. Está estrechamente relacionado con el modelo 'Sale' para proporcionar un desglose detallado de cada transacción.

6. **User**

   - **Descripción**: Representa a los usuarios del sistema, incluyendo tanto a los administradores como a los empleados que interactúan con el sistema contable. Este modelo gestiona la información de autenticación y autorización de los usuarios.

Roles de usuario en el sistema: Para la implementación de los roles de usuario se necesitan algunas tablas ya definidas mediante una especificación denominada Spatie. El autor (Pulido Fuentes y cols., 2022) comenta que "Este paquete nos permite asociar a nuestros usuarios roles y permisos que serán guardados en nuestra base de datos sin tener que crear las migraciones manualmente, sino que ya el paquete nos las trae listas, además nos ofrece un par de modelos para los roles y permisos con una serie de métodos que nos garantizan mucha simplicidad.". (p. 29).

Infraestructura

La Figura 4.3 muestra la infraestructura implementada mediante los servicios en la nube proporcionados por Amazon Web Services, configurada con Docker. Es importante mencionar que para desplegar esta infraestructura se utilizaría la imagen de Docker especificada como php:apache2.

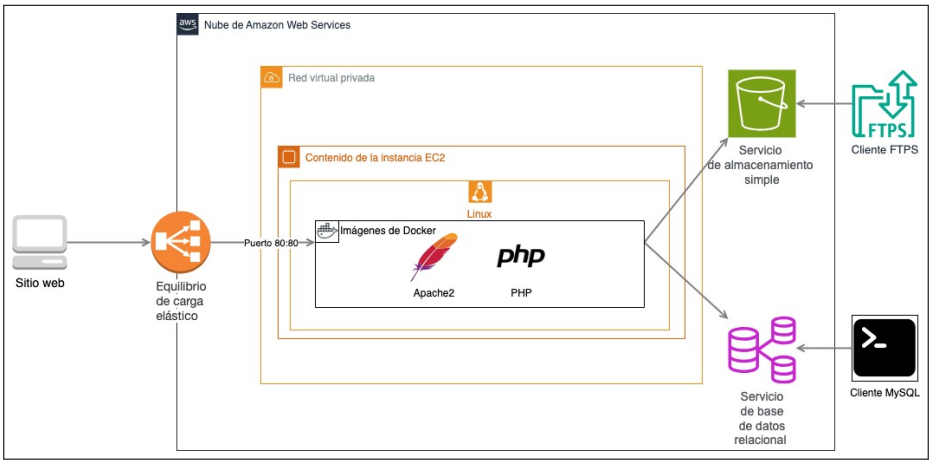

Figura 4.3: Diagrama de infraestructura.

Carga distribuida a través de servicios suministrados

Para lograr carga distribuida mediante el uso de servicios suministrados se hace uso la especificación Multi-AZ de Amazon RDS con una instancia en espera, deacuerdo con (Amazon Web Services, 2024c) se comenta que "En una implementación Multi-AZ de Amazon RDS, Amazon RDS crea de manera automática una instancia de base de datos (DB) principal y replica de manera síncrona los datos en una instancia de una AZ diferente. Cuando detecta un error, Amazon RDS conmuta por error automáticamente a una instancia en espera sin necesidad de intervención manual" (s. p.). En la Figura 4.4 se detalla un ejemplo del funcionamiento inicial de Multi-AZ con una instancia en

espera, junto con la configuración de Réplicas de lectura, lo cual favorece la alta disponibilidad y la escalabilidad.

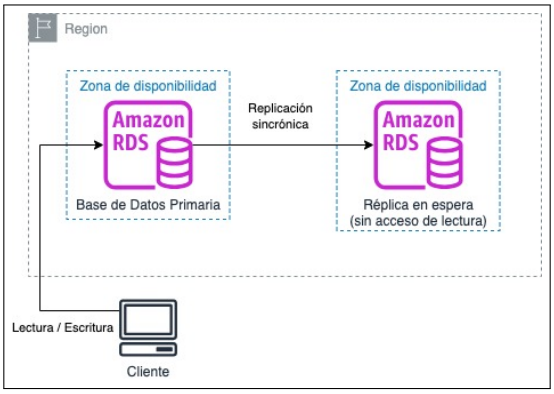

Figura 4.4: Detalle funcionamiento de Multi-AZ de Amazon RDS con una instancia en espera.

Además de la especificación denominada Réplicas de lectura de Amazon RDS, la cual, en palabras de (Amazon Web Services, 2024a), "Las réplicas de lectura de Amazon RDS ofrecen mayor rendimiento y durabilidad para instancias de base de datos (DB) de Amazon RDS. Las réplicas facilitan la capacidad para escalar horizontalmente más allá de las limitaciones de capacidad de una única instancia de base de datos para cargas de trabajo de base de datos con uso intensivo de las lecturas." (s. p.). En la Figura 4.5 se detalla el proceso de recuperación ante fallos utilizando la configuración de Multi-AZ con una instancia en espera y Réplicas de lectura.

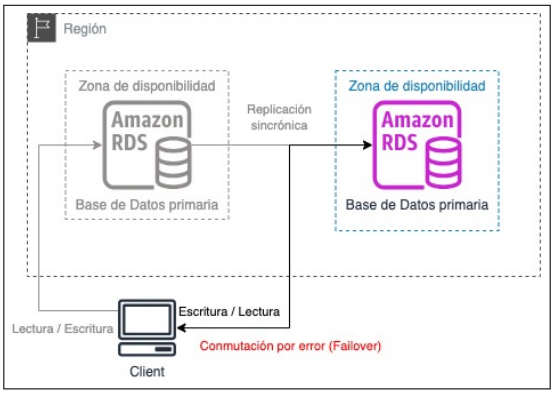

Figura 4.5: Detalle de Multi-AZ de Amazon RDS, una instancia en espera y Failover.

En caso de fallo de la instancia principal, se realiza un failover apuntando a una de las réplicas de lectura previamente configuradas. Esta réplica, al ser promovida, adquiere permisos de escritura y lectura, asumiendo el rol de instancia principal para asegurar la continuidad del servicio y la disponibilidad de datos. Con esta configuración, mientras que Multi-AZ se encarga de la alta disponibilidad sin distribución de carga, la combinación con réplicas de lectura permite una distribución efectiva de las consultas de lectura. En este caso, Multi-AZ garantiza la disponibilidad y recuperación ante desastres, mientras que las réplicas de lectura ayudan a equilibrar la carga de

lectura y mejorar el rendimiento.

Estrategia de Carga: Con esta combinación, las solicitudes de lectura se distribuyen entre la instancia principal y las réplicas de lectura, mientras que las operaciones de escritura se dirigen a la instancia principal. Esto permite una escalabilidad horizontal efectiva para operaciones de lectura sin afectar la disponibilidad ni la consistencia de la base de datos. Es importante destacar que estas especificaciones deben configurarse al momento de crear la base de datos en los servicios de Amazon RDS. Multi-AZ y Réplicas de lectura son opciones que se pueden seleccionar durante el proceso de creación para asegurar alta disponibilidad y mejorar el rendimiento de lectura. Estas configuraciones no pueden modificarse una vez que la base de datos está en funcionamiento, por lo que es crucial planificar y seleccionar las opciones adecuadas desde el principio.

4.3. Desarrollo

Durante el desarrollo de una aplicación web se crearon los modelos especificados para representar eficazmente las entidades clave del sistema contable. Cada modelo fue diseñado cuidadosamente para reflejar la estructura de datos necesaria, asegurando coherencia e integridad.

4.3.1. Módulo de categorias

En la Figura 4.6 se detalla el uso del listado de categorias previamente creadas para que los productos puedan relacionarse a estas. Al tratarse de un módulo de gestión, estos cuentan con el alcance de buscar, filtrar, registrar, editar o eliminar en el listado de categorías. Es importante destacar que todos los módulos de gestión cuentan con el mismo flujo en cuanto a búscar en la lista de resultados, registrar, editar o eliminar, sin embargo, esto se explica más a detalle en el módulo de productos para simplificar la explicación en el resto de módulos de gestión.

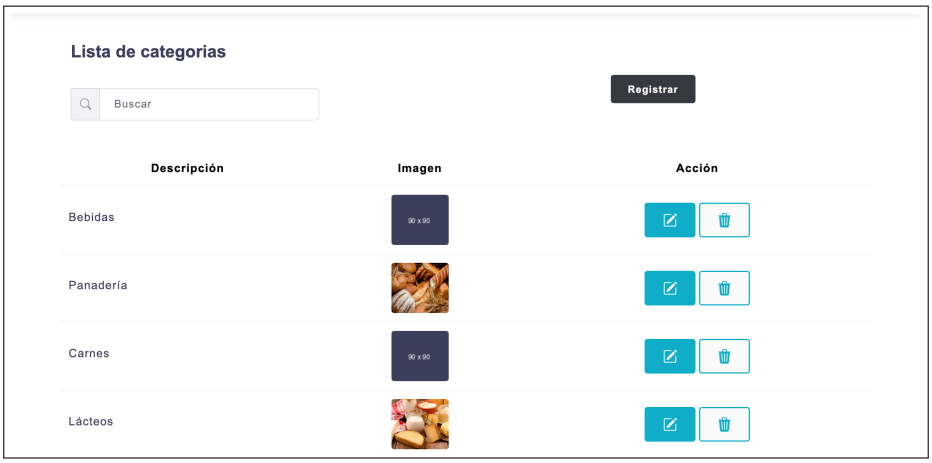

Figura 4.6: Detalle de lista de categorias.

En la Figura 4.7 se detalla el uso del modal para la creación de una categoría.

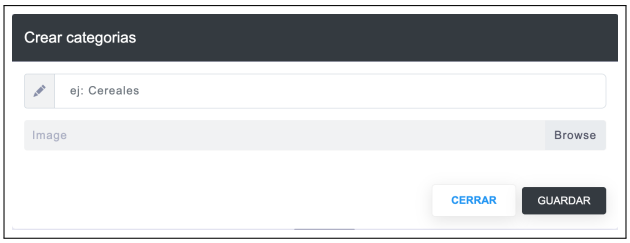

Figura 4.7: Detalle de creación de categorias.

4.3.2. Módulo de productos

En la Figura 4.8 se detalla el uso del listado de productos previamente creados, y ya relacionados a según sea, al listado de categorias anteriormente detallado. Al tratarse de un módulo de gestión, este permite buscar, registrar, editar o eliminar productos en el listado. Sin embargo, al detallar la explicación, la opción de ordenar en orden ascendente o descendente solo se implementa en este módulo mediante el uso de botones (que se encuentran entre la caja de búsqueda y la tabla retornada), permitiendo ordenar por nombre o por stock.

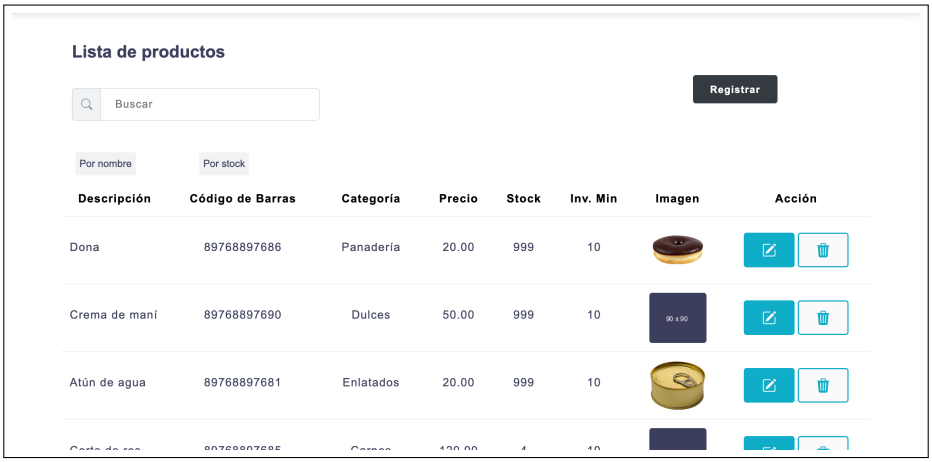

Figura 4.8: Detalle de lista de productos.

En la Figura 4.9 se muestra el formulario para la creación de un registro en la tabla de productos, para la creación de este registro se requiere completar atributos como: el nombre, código, costo, precio, stock, las alertas, su respectiva categoría mediante un dropdown, y una imagen del mismo (esta última opcionalmente).

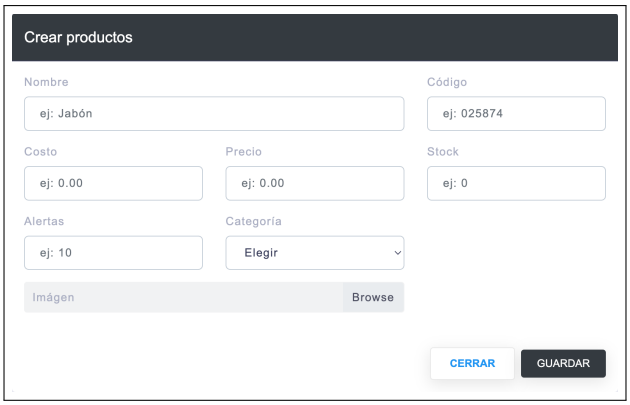

Figura 4.9: Detalle de registro de productos mediante modal.

En la Figura 4.10 se ejemplifica el uso del llenado del formulario para la creación de un registro en la tabla de productos. Es importante destacar que siempre es necesario avisarle al usuario en caso de que este registro se haya ejecutado exitosamente y ocultar la ventana modal o redirigir al listado de productos según sea el caso.

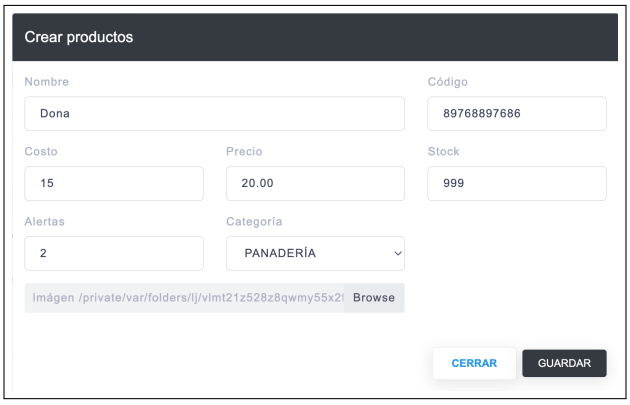

Figura 4.10: Detalle de llenado del formulario para realizar el registro de algún producto mediante modal.

En la Figura 4.11 se ejemplifica el uso de las validaciones y los mensajes (con color de fuente en rojo) con los que se comprometieron a cumplir para poder insertar algún registro en la tabla de productos.

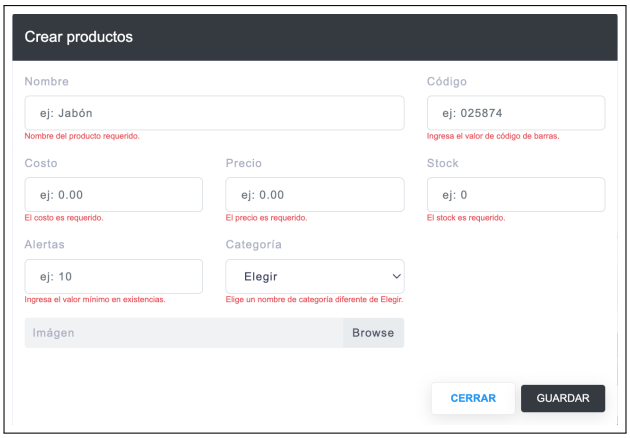

Figura 4.11: Detalle de los mensajes comprometidos en conjunto con sus validaciones.

En la Figura 4.12 se ejemplifica el uso de la caja de búsqueda por nombre en la lista de productos, similar al resto de módulos de gestión que cuentan con esta, sin embargo, en el caso del módulo de productos se puede usar para buscar por nombre o por el código asociado al producto, el cual se encuentra validado para que sea único en la tabla de productos.

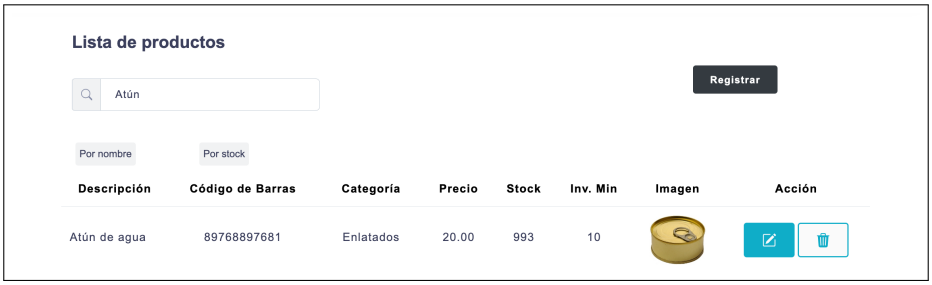

Figura 4.12: Detalle de los mensajes comprometidas en conjunto con sus validaciones.

En la Figura 4.13 se ejemplifica el uso de uno de los modales para confirmar la eliminación de un producto, los cuales son similares a los que se abordan en el resto de módulos de gestión.

Figura 4.13: Detalle del modal para confirmar la eliminación de un producto.

4.3.3. Módulo de denominaciones

La Figura 4.14 muestra el módulo de gestión de denominaciones, donde se pueden ver las filas previamente registradas. Para cada fila, es necesario especificar su valor, tipo de denominación y opcionalmente una imagen. Este listado incluye todas las denominaciones asociadas al sistema.

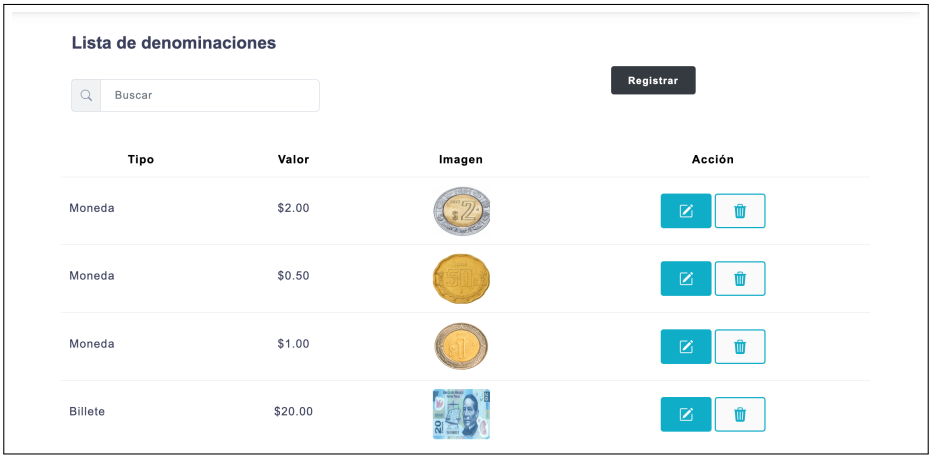

Figura 4.14: Detalle de la lista de denominaciones.

En la Figura 4.15 se detalla el modal para creación de una denominación, este modal cuenta con los atributos anteriormente mencionados.

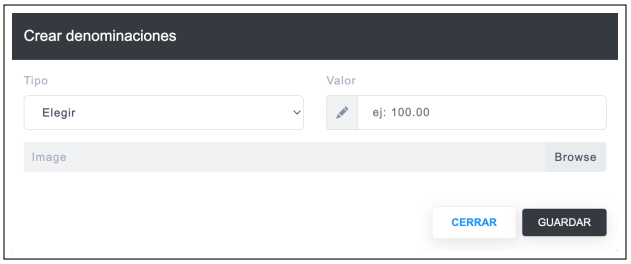

Figura 4.15: Detalle del modal para crear una denominación.

4.3.4. Módulo de ventas

Durante la etapa de desarrollo de este módulo, fue necesario dividirlo en submódulos, como el carrito de ventas. Este carrito se desglosa en dos submódulos: la selección de productos y la selección de denominaciones, siendo este último compartido con otro módulo. Para la implementación de esta funcionalidad, se empleó la biblioteca darryldecode/cart. En la siguiente especificación de código se muestra que, aunque no es visible para el usuario, el carrito de ventas es una sublógica del módulo de ventas. Este paquete es una implementación de carrito de compras para Laravel 5, 6, 7 y 9. Admite múltiples instancias de carrito, almacenamiento basado en sesiones, asociación de modelos e impuestos. También le permite aplicar condiciones a nivel de carrito o artículo, como descuentos, costos de envío o cargos por servicio.

```
<?php
```

```
trait CartTrait {
    public function ScanCode($barcode, $cant = 1)
    {
        $product = Product::where('barcode', $barcode)->first();
        if ($product == null || empty($product)) {
            $this->emit('scan-notfound', 'El producto no esta
                registrado');
        } else {
            if ($this->InCart($product->id)) {
                $this->IncreaseQuatity($product);
                return;
            }
            if ($product->stock < 1) {
                $this->emit('no-stock', 'Stock no suficiente*');
                return;
            }
            Cart::add($product->id, $product->name, $product->
                price, $cant, $product->imagen);
            $this->total = Cart::getTotal();
            $this->itemsQuantity = Cart::getTotalQuantity();
            $this->emit('scan-ok', 'Producto agregado*');
        }
    }

}
```

Código 4.1: Detalle del método ScanCode

En cuanto al Código 4.1 es importante destacar que para mantener el controlador del módulo de ventas limpio y manejable, se optó por utilizar Traits para encapsular la lógica relacionada con el carrito de compras. Esta decisión permitió abstraer y modularizar los métodos responsables de la gestión del carrito, evitando así que la clase del controlador se volviera excesivamente larga y compleja. Este fragmento es perteneciente a el manejo de identificación de productos para agregarlos en el carrito de ventas asociado a la sesión, el cual, a grandes rasgos, introduce el registro `$barcode` al objeto `Cart` mediante el método estático `add`.

Según el autor (Zandstra, 2013) "Trait es una implementación de clase parcial (es decir, constantes, propiedades y métodos) que se puede mezclar en una o más clases PHP existentes. Los rasgos cumplen una doble función: dicen lo que una clase puede hacer (como una interfaz) y proporcionan una implementación modular (como una clase)". (p. 2). En el Código 4.2 se puede observar el uso del método para comprender si se encuentra un producto en el carrito a partir del id del producto `$productId` que se reciba por parámetro para retortar un booleano (verdadero o falso).

```php
<?php
    public function InCart($productId)
    {
        $exist = Cart::get($productId);
        return $exist ? true : false;
    }
```

Código 4.2: Detalle del método InCart

En el Código 4.3 se puede observar el uso del método para incrementar la cantidad de productos en el carrito de ventas. Por defecto debe recibir un identificador para el producto, opcionalmente se puede recibir la cantidad o reutilizar el método para incrementar la cantidad de productos en el carrito por 1 (en automático).

Es importante destacar que el método 'IncreaseQuantity', asociado al botón con el signo (+), mostrado en la Figura 4.45, permite incrementar la cantidad de un producto en el carrito. Si el producto ya existe en el carrito asociado a la sesión, la cantidad especificada como parámetro se suma a la cantidad actual del producto en el carrito, siempre y cuando el valor en el atributo 'stock' del producto sea mayor.

```php
<?php
public function IncreaseQuatity($product, $cant = 1)
{
    $title = '';
    $exist = Cart::get($product->id);
    $title = $exist ? 'Cantidad actualizada' : 'Producto agregado
        ';

    if ($exist) {
        if ($product->stock < ($cant + $exist->quantity)) {
            $this->emit('no-stock', 'Stock insuficiente');
            return;
        }
    }
}
```

Código 4.3: Detalle del método IncreaseQuatity

En el Código 4.4 se puede observar el uso del método para remover un producto en particular del carrito de ventas asociado a la sesión. Basicamente utilizamos la clase carrito para ubicarlo con `$productId` y actualizar algunos atributos públicos correspondientes al módulo de ventas.

```php
<?php
```

```
public function removeItem($productId)
{
    Cart::remove($productId);

    $this->total = Cart::getTotal();
    $this->itemsQuantity = Cart::getTotalQuantity();
    $this->emit('scan-ok', 'Producto eliminado');
}
```

Código 4.4: Detalle del método removeItem

En la Figura 4.16 se puede observar la interfaz de usuario del módulo de ventas, incluyendo el detalle del carrito, cálculo de monto y la selección de monedas.

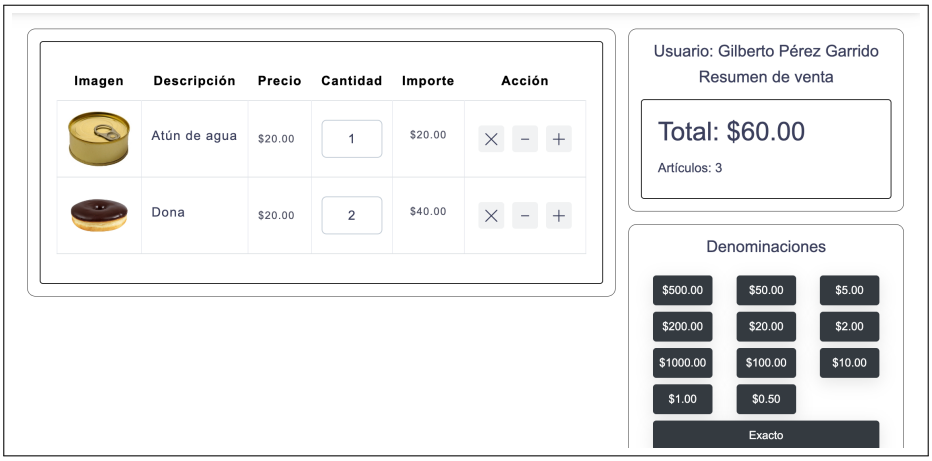

Figura 4.16: Detalle del módulo de ventas.

4.3.5. Módulo de usuarios

En la Figura 4.17 se detalla el listado de usuarios registrados previamente en el sistema con algunos de los atributos más importantes que representa esta entidad, además de ser un módulo de gestión más, en este se puede observar que es aquí cuando se comienza a hacer uso de los registros en la lista de roles del módulo de roles al momento de registrar a algún usuario en su respectivo modal.

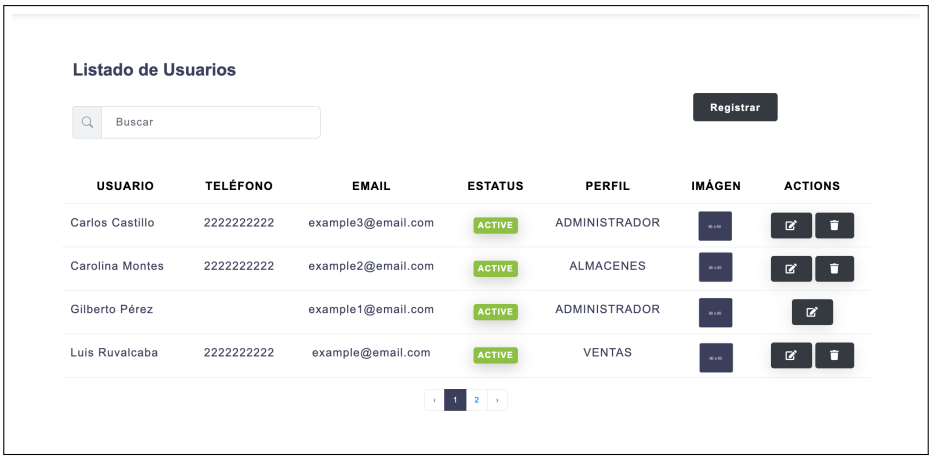

Figura 4.17: Detalle de la lista de usuarios registrados.

En la Figura 4.18 se detalla el formulario de registro para un usuario en el sistema. Cabe destacar que los roles a desplegar en el menú desplegable son los abordados en la Figura 4.19.

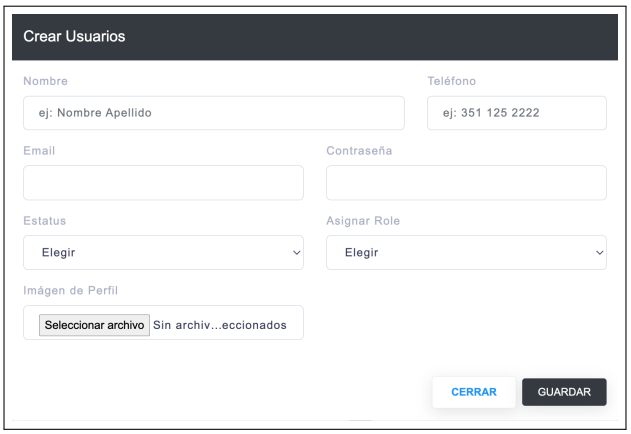

Figura 4.18: Detalle del modal para crear un usuario en el sistema.

4.3.6. Módulo de roles

En la Figura 4.19 se detalla el listado de roles que se desea registrar en la lógica del sistema, esto para especificar sus permisos asociados posteriormente. Es importante destacar que, al tratarse de un módulo de gestión, este cuenta con los casos de uso soportados anteriormente, así como se explicó en los módulos anteriores.

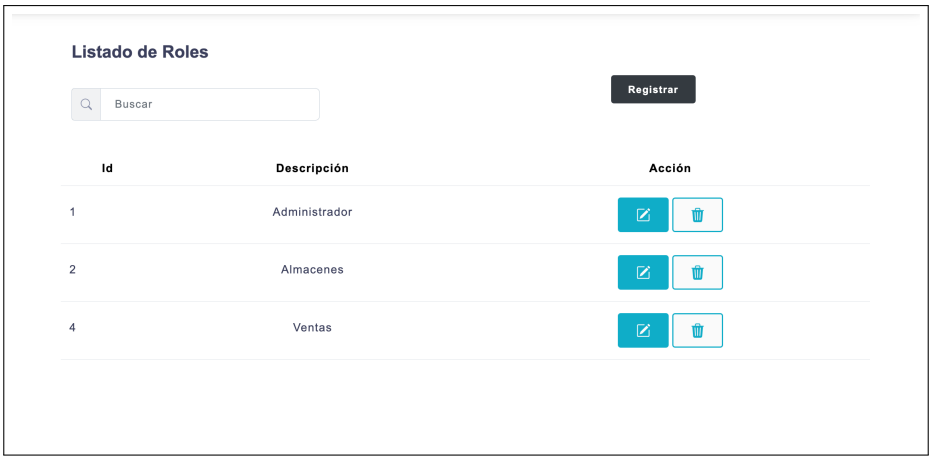

Figura 4.19: Detalle del los roles diponibles.

En la Figura 4.20 se detalla el formulario para la creación de un rol.

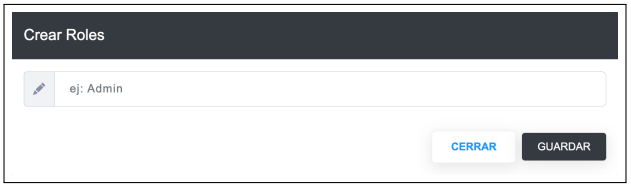

Figura 4.20: Detalle del modal para la creación de un rol en el sistema.

4.3.7. Módulo de permisos y asignación de permisos

En Figura 4.21 se detalla el listado de permisos previamente adjuntados en el sistema contable, para posteriormente poder asignarlos a los roles requeridos.

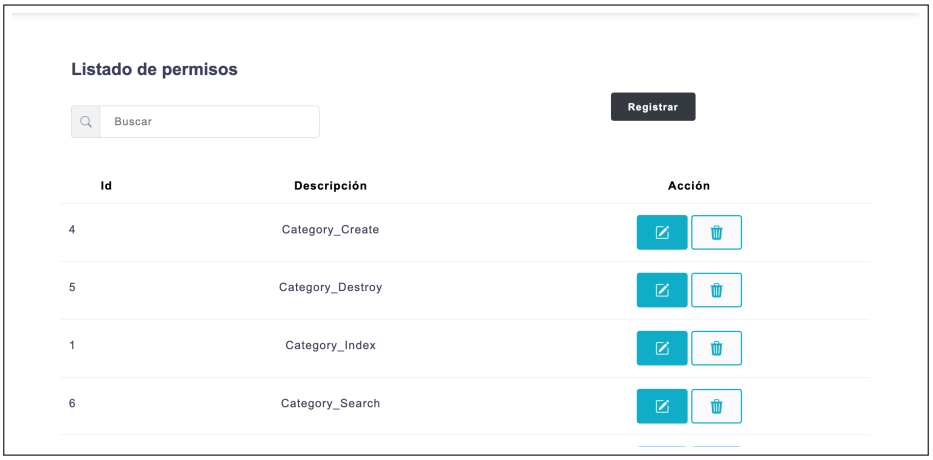

Figura 4.21: Detalle de la lista de permisos diponibles.

En la Figura 4.22 se detalla el creado de permisos para posteriormente adjuntarlos en el sistema

contable.

Figura 4.22: Detalle del modal para la creación de un permiso en el sistema.

En Figura 4.23 se detalla la asignación de permisos, según el rol previamente seleccionado, en la imagen se puede observar que, en este caso, el rol de administrador cuenta con todos los permisos registrados en el sistema.

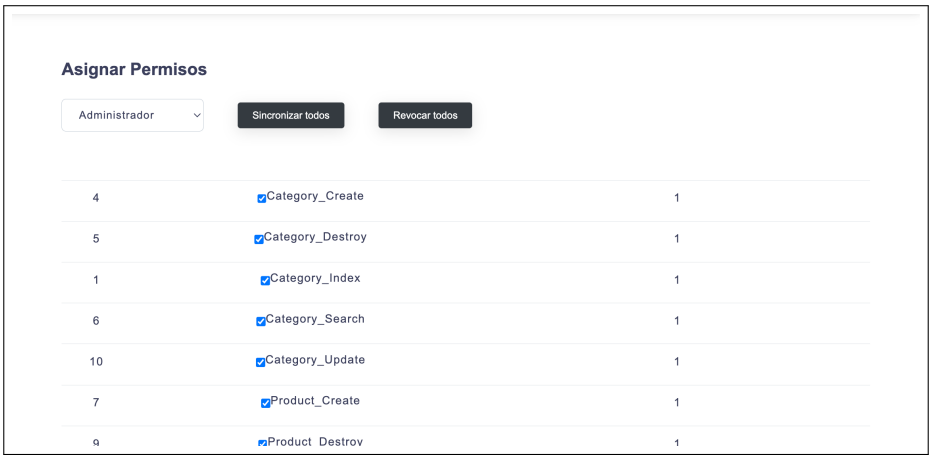

Figura 4.23: Detalle de los asignación de permisos por rol.

4.3.8. Módulo de corte de caja

En la Figura 4.24 se detalla el uso del módulo de corte de caja, este módulo puede resultar muy similar al módulo de reporte de ventas, sin embargo, en este se detalla a primera vista aspectos importantes como las ventas totales y el total de artículos.

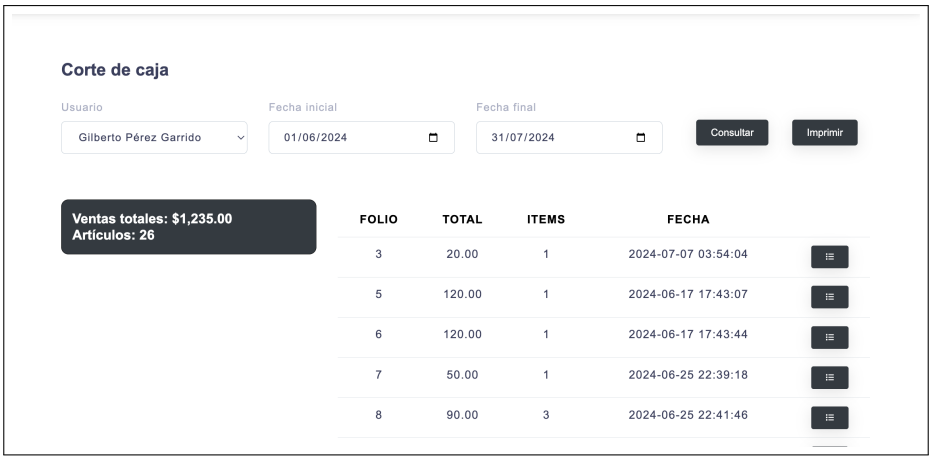

Figura 4.24: Detalle de corte de caja.

En la Figura 4.25 se puede observar el detalle de un registro de ventas.

Figura 4.25: Detalle del modal para observar el detalle de una venta.

4.3.9. Módulo de reportes de ventas

En la Figura 4.26 se detalla el uso del módulo de reporte de ventas, en el cual se puede escoger algún usuario registrado previamente, y que haya tenido accesos al módulo de ventas para ejecutar alguna misma, para posteriormente poder observalo en este listado, en este se puede seleccionar un corte dependiendo de la fecha de inicio y la fecha de fin seleccionada por el usuario que consulta, dónde automáticamente se realiza una petición para listar los registros que cumplan con los filtros especificados. Es importante destacar que el análisis de detalles a través de la interfaz de usuario se realiza reutilizando el modal que se muestra en la figura 4.25. Además, al seleccionar las especificaciones, las opciones que cumplen con los criterios se muestran automáticamente en el listado.

Se utiliza la fachada proporcionada por el paquete Laravel Excel para la exportación de los archivos en Excel, que es una biblioteca para trabajar con archivos de Excel en aplicaciones Laravel. La generación del PDF se realiza mediante el uso del motor de plantillas Blade (DomPdf). Este proceso implica la utilización de directivas de Blade, HTML, y CSS, junto con la lógica interpretada por las directivas de Blade, que se extiende desde el Back-end.

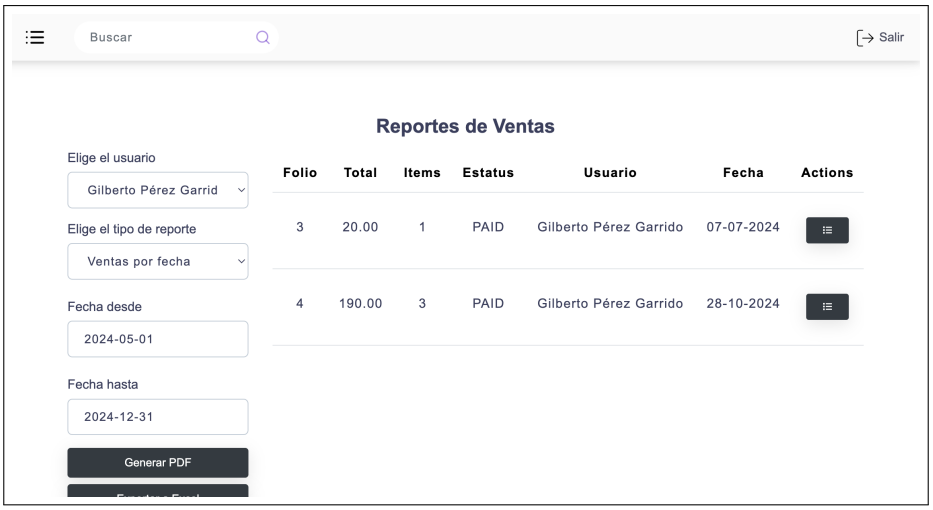

Figura 4.26: Detalle de reporte de ventas en interfaz.

En Figura 4.27 se puede observar de ejemplo el detalle, del cual se seleccionó un usuario en especifico mostrado en la anterior figura:

FOLIO	IMPORTE	ITEMS	ESTATUS	USUARIO	FECHA
7	50.00	1	PAID	Gilberto Pérez Garrido	2024-06-25 22:39:18
8	90.00	3	PAID	Gilberto Pérez Garrido	2024-06-25 22:41:46
9	525.00	5	PAID	Gilberto Pérez Garrido	2024-06-25 22:43:43
10	50.00	1	PAID	Gilberto Pérez Garrido	2024-06-25 22:44:27
11	100.00	5	PAID	Gilberto Pérez Garrido	2024-06-25 22:46:11
12	20.00	1	PAID	Gilberto Pérez Garrido	2024-06-26 16:54:29
TOTALES	**$835.00**	16			

Figura 4.27: Detalle del archivo de reporte en formato PDF.

En la Figura 4.28 se puede observar la información del detalle anterior en formato de hoja de cálculo, ya que regulamente sólo se requiere las filas y columnas para estos.

FOLIO	IMPORTE	ITEMS	ESTATUS	USUARIO	FECHA
7	50	1	PAID	Gilberto Pérez Garrido	2024-06-25T22:39:18.000000Z
8	90	3	PAID	Gilberto Pérez Garrido	2024-06-25T22:41:46.000000Z
9	525	5	PAID	Gilberto Pérez Garrido	2024-06-25T22:43:43.000000Z
10	50	1	PAID	Gilberto Pérez Garrido	2024-06-25T22:44:27.000000Z
11	100	5	PAID	Gilberto Pérez Garrido	2024-06-25T22:46:11.000000Z
12	20	1	PAID	Gilberto Pérez Garrido	2024-06-26T16:54:29.000000Z

Figura 4.28: Detalle del archivo de reporte en formato hoja de cálculo (xlsx).

4.3.10. Módulo de autenticación

Es importante destacar que el alcance actual, es que el usuario, por ejemplo, el usuario con rol de administrador sea el encargado de crear las cuentas de usuarios y asignarle algún rol validado en el sistema. En Figura 4.29 se puede mostrar el ejemplo de la autenticación mediante el uso de correo electrónico y contraseña.

Figura 4.29: Detalle de autenticación para inicio de sesión.

4.3.11. Configuración del ambiente

Para la configuración del ambiente se intenta realizar lo más a fin a la infraestructura detallada en la Figura 4.3.

Docker

El Dockerfile mostrado en la Figura 4.30 configura un entorno de servidor web con PHP y Apache. Se define el directorio de trabajo, se instalan librerías necesarias, y se habilita el módulo `rewrite` de Apache. También se configuran permisos, se copian archivos de configuración y se instala Composer para gestionar dependencias de PHP. El contenedor expone el puerto 80 y utiliza

`apache2-foreground` para iniciar el servidor web. Finalmente, se procede a construir la imagen con el comando `docker build -t prueba .` (donde `build` indica la construcción de la imagen, `-t prueba` asigna la etiqueta `prueba` a la imagen, y (`.`) especifica el directorio actual como el contexto de construcción), y posteriormente, a crear el contenedor. En cuanto al uso del puerto 80, se requiere destacar que, el autor (Schneider, 1996) comenta que "En Internet, para la comunicación en HTTP, generalmente se requiere colocar a través de conexiones TCP/IP. El puerto predeterminado es TCP 80". (p. 3.). Esto significa que cada vez que se envía una solicitud HTTP, como cuando se accede a una página web, se utiliza el protocolo TCP/IP para establecer una conexión. El autor (Corona y cols., 2004) comenta que "En otras palabras, el protocolo TCP/IP es el que utilizan todas las computadoras conectadas a Internet". (p. 3.).

```
# Utiliza la imagen base de PHP 7.3 con Apache
FROM php:7.3-apache
# Configura el directorio de trabajo en el contenedor
WORKDIR /var/www/html
# Actualiza los paquetes del sistema e instala librerías necesarias
# Configura e instala la extensión de PHP para manejo de archivos zip
RUN apt-get update && apt-get install -y libpng-dev libjpeg-dev
libfreetype6-dev \ zip unzip nano libzip-dev zip \ &&
docker-php-ext-configure zip \ && docker-php-ext-install zip
# Instala las extensiones de PHP: gd (para manejo de imágenes),
# pdo y pdo_mysql (para manejo de bases de datos)
RUN docker-php-ext-install gd pdo pdo_mysql
# Habilita el módulo rewrite de Apache para usar .htaccess
RUN a2enmod rewrite
# Cambia los permisos del directorio /var/www
# para permitir lectura, escritura y ejecución
RUN chmod 777 -R -c /var/www
# Copia el archivo de configuración de Apache personalizado al contenedor
COPY docker/000-default.conf /etc/apache2/sites-available/000-default.conf
# Copia el archivo de entorno al directorio de trabajo en el contenedor
COPY docker/.env-pro /var/www/html/.env
# Copia el archivo de configuración principal de Apache personalizado al contenedor
COPY docker/apache2.conf /etc/apache2/
# Instala Composer
COPY --from=composer /usr/bin/composer /usr/bin/composer
# Configura Composer para permitir el uso como superusuario
ENV COMPOSER_ALLOW_SUPERUSER 1
# Copia los archivos de configuración de Composer al directorio de trabajo
COPY composer.json composer.lock ./
# Copia todo el contenido del proyecto al directorio de trabajo en el contenedor
COPY . /var/www/html/
# Cambia el propietario del directorio de trabajo a www-data y ejecuta Composer para instalar las dependencias y generar el autoload
RUN chown -R www-data:www-data /var/www/html \
&& composer install && compo
# Expone el puerto 80 para acceder al servidor web
EXPOSE 80
# Comando para iniciar el servidor web de Apache
CMD ["apache2-foreground"]

```

Figura 4.30: Detalle del contenedor con Apache2 y PHP.

En la Figura 4.31 se muestra el levantamiento del contenedor a partir de la imagen denominada `prueba`. Para el levantamiento se usó el comando `docker run -d -p 80:80 --name accounting-system prueba`, donde `run` inicia un nuevo contenedor desde una imagen especificada, `-d` ejecuta el contenedor en modo desatendido (detached mode), permitiendo que el terminal quede libre, `-p 80:80` mapea el primer 80 es el puerto de la máquina anfitriona (host) al segundo 80 (es el puerto del contenedor), permitiendo el acceso al servicio web en el puerto 80 de la máquina anfitriona, y `--name accounting-system` asigna un nombre específico al contenedor para facilitar su identificación. Además se pueden observar datos de interés, como la versión de apache2 con `apache2 -v`, la versión de php con `php -v` y mostrar información sobre el sistema que estamos

usando en el que se ejecuta esta versión de php con apache2 con `uname -a`. Cabe destacar que después de comprobar que el contenedor se está ejecutando, es posible acceder desde la configuración `http://0.0.0.0:80`, que se detalló anteriormente en la especificación de la imagen mediante el uso de Dockerfile.

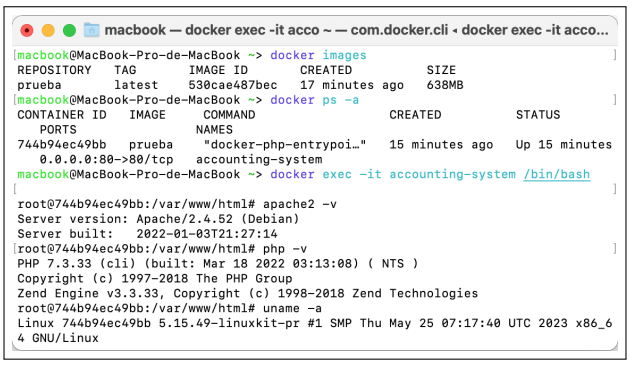

Figura 4.31: Detalle del contenedor con Apache2 y PHP.

Amazon S3

En cuanto al uso del protocolo FTP (Protocolo de transferencia de archivos), se trabaja mediante aplicación Cyberduck, esto para verificar la conexión de almacenamiento con Amazon S3. Cabe mencionar que cada módulo que implementa el manejo de imágenes guarda estos archivos en directorios con el mismo nombre del módulo, facilitando así su identificación, así como se detalla en Figura 4.44. En la Figura 4.32 se muestra el levantamiento de los servicios con Amazon Simple Cloud Storage (S3) con el nombre del Bucket (contenedor de objetos (archivos)), sistema-de-administracion-contable.

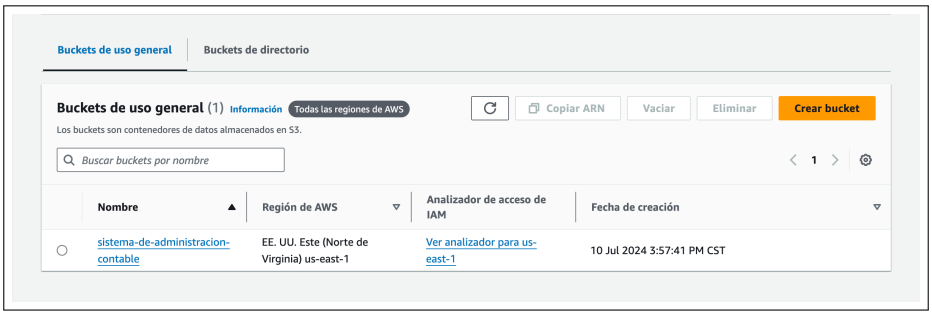

Figura 4.32: Detalle de la conexión con el servicio de almacentamiento en Amazon Simple Cloud Storage (S3).

Amazon RDS

En la Figura 4.33 se muestra la sincronización desde el ambiente local al ambiente de pruebas mediante el uso de una instancia de base de datos. Esta funcionalidad es provista por herramientas como el ID de Base de Datos Navicat, aunque otras herramientas como phpMyAdmin provee la funcionalidad similar.

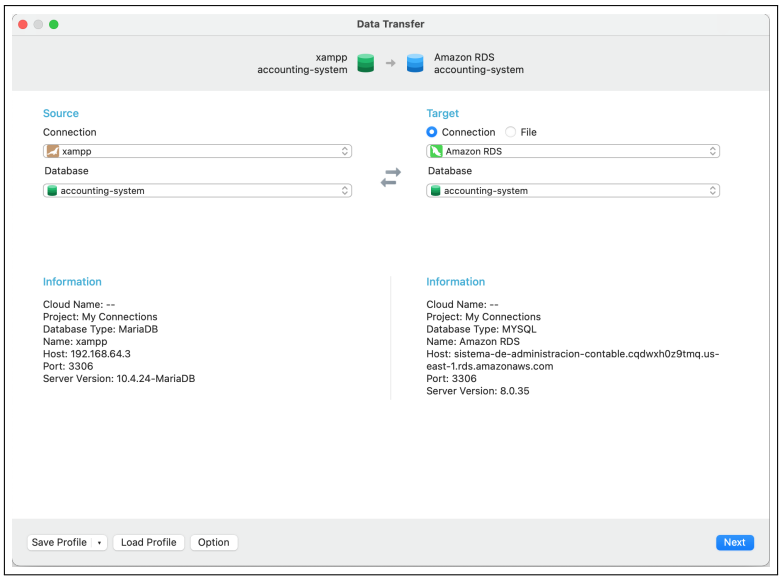

Figura 4.33: Detalle de sincronización de las tablas desde ambiente local a pruebas.

Para escalar una Base de Datos en memoria, es importante seleccionar las siguientes opciones según la Figura 4.34, en la cual, la consola de Amazon Web Services, Según la documentación (Amazon Web Services, 2024d) “Si su carga de trabajo es impredecible, puede habilitar el escalado automático de almacenamiento para una instancia de base de datos de Amazon RDS. Para ello, puede usar la consola de Amazon RDS, la API de Amazon RDS o la AWS CLI” (s. p.). Además, en la documentación (Amazon Web Services, 2024d) se menciona que: Çon la opción de escalado automático de almacenamiento habilitada, si Amazon RDS detecta que se está quedando sin espacio en la base de datos, aumenta automáticamente el almacenamiento. Amazon RDS inicia una modificación en el almacenamiento para una instancia de base de datos habilitada con la opción de escalado automático cuando se aplican los siguientes factores”(s. p.). Esto garantiza que el sistema pueda adaptarse automáticamente a las necesidades de almacenamiento.

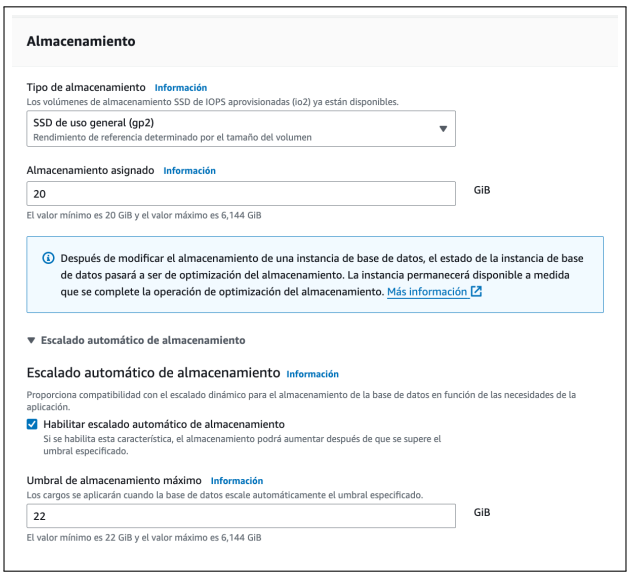

Figura 4.34: Detalle de configuración para autoescalado en almacenamiento.

En la Figura 4.35 se muestra la implementación de los servicios de Instancia de base de datos Multi-AZ.

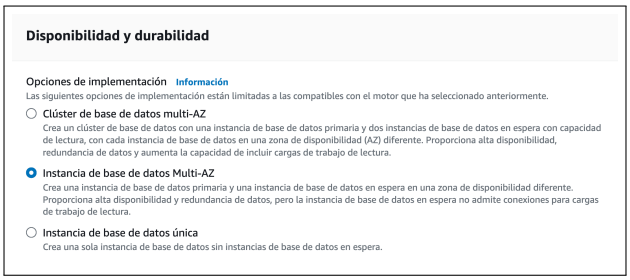

Figura 4.35: Detalle de configuración para Multi-AZ.

En la Figura 4.36 se muestra la implementación de los servicios de replicas de lectura.

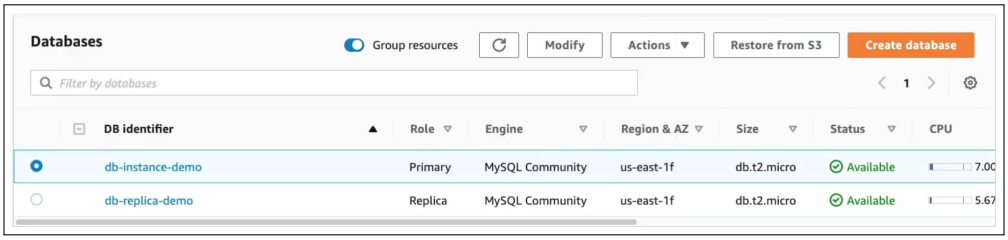

Figura 4.36: Detalle de configuración con replicas de lectura.

Amazon EC2

Para utilizar una instancia de Amazon EC2, se emplea el protocolo SSH para conectarse de forma remota al cliente de la instancia, el cual está basado en un sistema operativo Linux. Este proceso se realiza utilizando una instancia respaldada por Amazon. Según la documentación (Amazon Web Services, 2024b) “Amazon Linux es una distribución respaldada y actualizada por Amazon Web Services, y disponible para usar mediante las instancias de Elastic Compute Cloud (EC2)” (s. p.). En Figura 4.37 se puede observar el detalle mediante el uso de la consola para verificar algunos datos relevantes, como el nombre de la instancia y la dirección IPv4 pública.

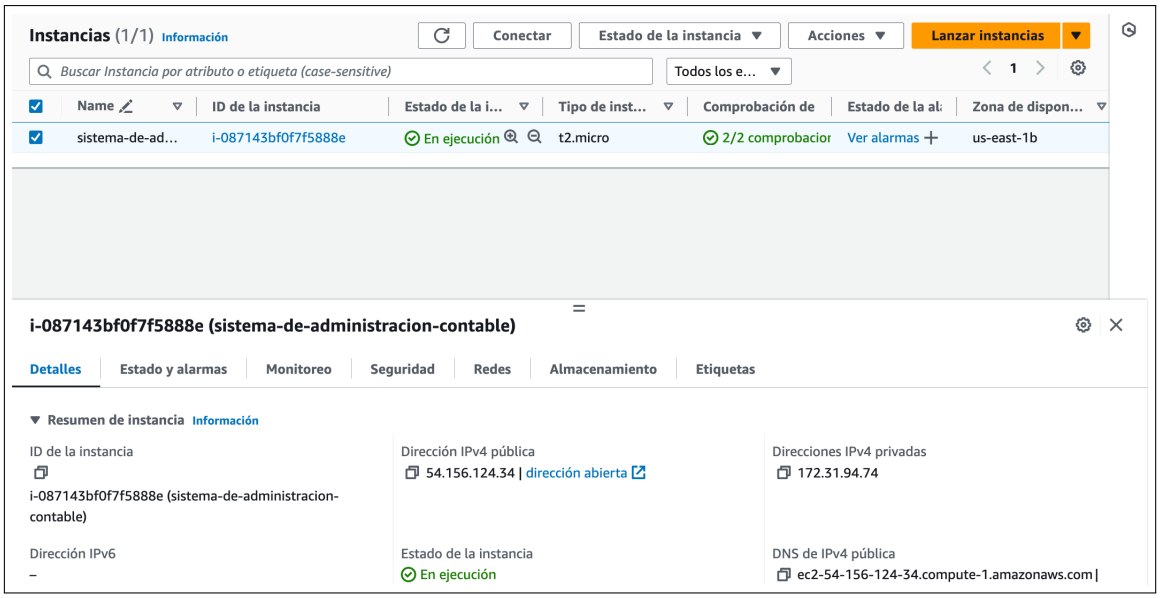

Figura 4.37: Detalle de acceso a EC2 mediate consola.

En la Figura 4.38 se muestran las reglas de acceso de entrada para las instancias, incluyendo los puertos TCP 22 (protocolo SSH) y 80 (protocolo HTTP), permitidos desde todas las direcciones IPv4 e IPv6

Figura 4.38: Detalle de reglas de entrada en EC2.

4.4. Pruebas

En esta etapa se abordan las pruebas de caja blanca y caja negra mencionadas. El autor (Saucedo y del Castillo Serpa, 2012) comenta que "Las pruebas de software son importantes porque aseguran el correcto cumplimiento de la funcionalidad del producto, ayudan a ganar confianza, confirman la fiabilidad del uso y previenen defectos en producción, lo cual tiene un impacto económico positivo en la empresa en cuestión" (p. 3).

4.4.1. Pruebas de caja blanca

Un método común para realizar pruebas de caja blanca es la depuración, que permite observar en tiempo real lo que sucede en cada línea de código, ya sea de manera implícita o explícita. En la Figura 4.39 se muestra el uso del módulo de denominaciones y el cálculo de montos dependiendo del caso de venta en el módulo de ventas. Se utilizan parámetros de ejemplo, como los que se detallan a continuación, para evaluar el flujo de código en particular que se ejecuta al ejecutar una venta.

Figura 4.39: Detalle del cálculo en ventas.

En la Figura 4.40 se muestra el uso del sistema de depuración Xdebug, una herramienta muy poderosa, ya que permite realizar diversas operaciones fundamentalmente en el lenguaje de programación PHP, para esto previamente se configuraron los archivos necesarios para que desde Visual Studio Code pueda interpretar las interrupciones de código necesarias. Del lado izquierdo del detalle (INSPECCIÓN), se pueden observar algunos valores que se reciben del lado del cliente para el carrito de ventas. Entre estos valores se incluyen: el total a cobrar con un valor de 90, la cantidad total de productos con un valor de 3, el efectivo seleccionado por el usuario mediante la interfaz con un valor de 500, y el cambio calculado a partir del efectivo menos el total a cobrar, que da un valor de 410. Además, el identificador del usuario que está ejecutando la venta con valor en 4, el tamaño de la lista de `count($items)` extraídos directamente del carrito de ventas con valor en 1 y los datos del único `count($item)` que se tiene en esta lista: precio en 30, cantidad en 3, ID

del producto en 7, y el ID del registro que se hizo en el modelo (`Sale`) al estarse ejecutando una venta. En cuanto a las lineas de código desde (`DB::beginTransaction()`) hasta (`DB::commit()`), nos referimos a una transacción de base de datos, en la cual se procede realizar un registro en el modelo (`Sale`), con los valores mostrados en (INSPECCIÓN), total a pagar, cantidad de productos, efectivo seleccionado, cambio calculado y el identificador del usuario que está ejecutando la venta, dependiendo de que se pueda escribir dentro de este modelo (`Sale`). Si fue existoso se procede a registrar dentro de (`SaleDetails`), se itera sobre el tamaño de la cantidad `count($items)` que indica la variedad de productos en el carrito de ventas, el precio del único `$item` (producto) que iteró la lista `items`, su precio, cantidad e ID, y finalmente el ID que se obtiene al inicializar un registro de venta en el modelo de ventas (`Sales`). Todo esto se guarda en el modelo de detalle de ventas (`SaleDetails`).

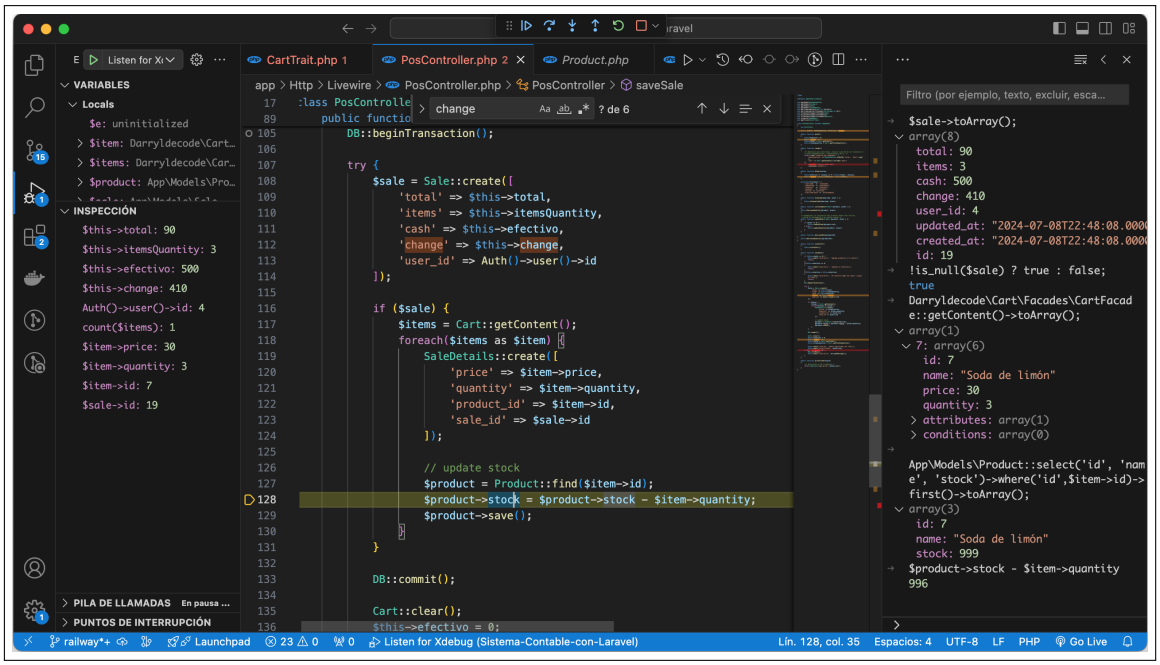

Figura 4.40: Detalle de la depuración con Xdebug para registrar la ejecución del módulo de ventas.

En cuanto a la consola de depuración de la Figura 4.40, la cual se encuentra al lado derecho, se puede mostrar que se evaluaron los registros realizados durante la ejecución del módulo de ventas. La variable `$sale` se convierte en un arreglo en la consola para facilitar su lectura. Se evalúa si `!is_null($sale)`, lo que indica si el registro se ha logrado dentro del modelo (`Sale`). También se evalúa el contenido que había en el carrito durante la ejecución, esto convirtiéndolo también a un arreglo para facilitar su lectura (`Darryldecode\Cart\Facades\CartFacade::getContent()`). También se evalúa el contenido del único producto, en el cual se iteró, seleccionando sólo el identificador, el nombre y el stock que contenía antes de llegar a la línea que lo modifica debido a su venta ejecutada mediante (`$product->save()`), esto convirtiéndolo también a un arreglo para facilitar su lectura (`App\Models\Product::select('id', 'nombre', 'stock')->where('id', $item->id)->first()`).

En la Figura 4.41 se muestra el uso del sistema de depuración Tinker, una herramienta provista

por Artisan CLI, en la cual se consultan los registros realizados por parte de el módulo de ventas en cuanto a las lineas de código de la Figura 4.40.

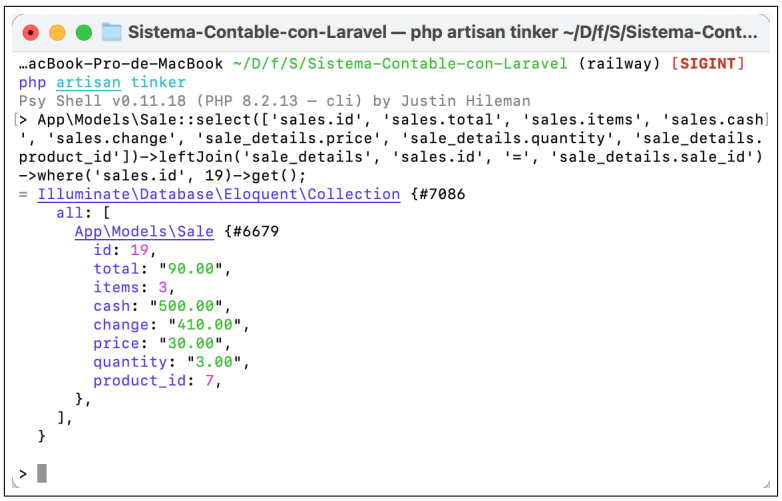

Figura 4.41: Detalle de la depuración con Tinker.

En la Figura 4.42 se muestra el uso del modal para la creación de un producto, el cual se depurará.

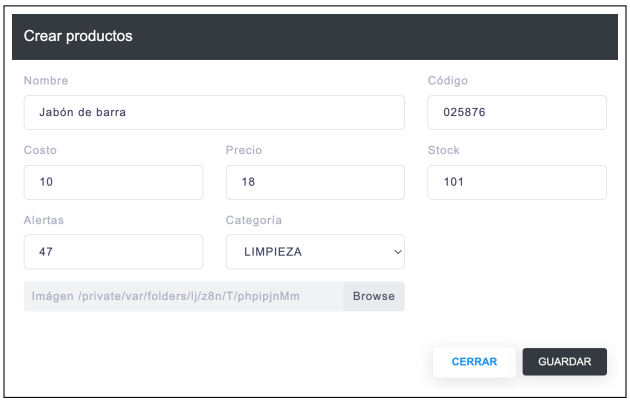

Figura 4.42: Detalle de creación de un producto mediante modal para depurar.

En la Figura 4.43 se muestra el uso del sistema de depuración Xdebug para la creación de un producto. Del lado izquierdo del detalle (INSPECCIÓN), se pueden observar algunos valores que se reciben del lado del cliente, los cuales son el nombre del producto, costo, precio, código de barras, stock, alertas y la categoría asociada. En cuanto a las lineas de código se puede observar que se guardan los valores antes mencionados para guardar la consulta en la variable (`$product`), esto para efectuarla posteriormente al realizar la manipulación de la imagen (`$this->image`), ya que se evalúa si esta variable no sea nula, para proceder a inicializar las variables de conexión con el servicio de Amazon S3 mediante este adaptador (biblioteca), se procede a guardar en el directorio productos del mismo mediante la linea (`Storage::disk('s3')->putFileAs('productos', $this->image, $customFileName, 'public')`).

En cuanto a la consola de depuración se puede observar que se intenta evalúar el directorio relativo

(`$filePath`) para observar su contenido, para después evalúar si existe en el directorio relativo (`Illuminate\Support\Facades\Storage::disk('s3')->exists($filePath)`) dicho archivo, lo cual retorna verdadero (true).

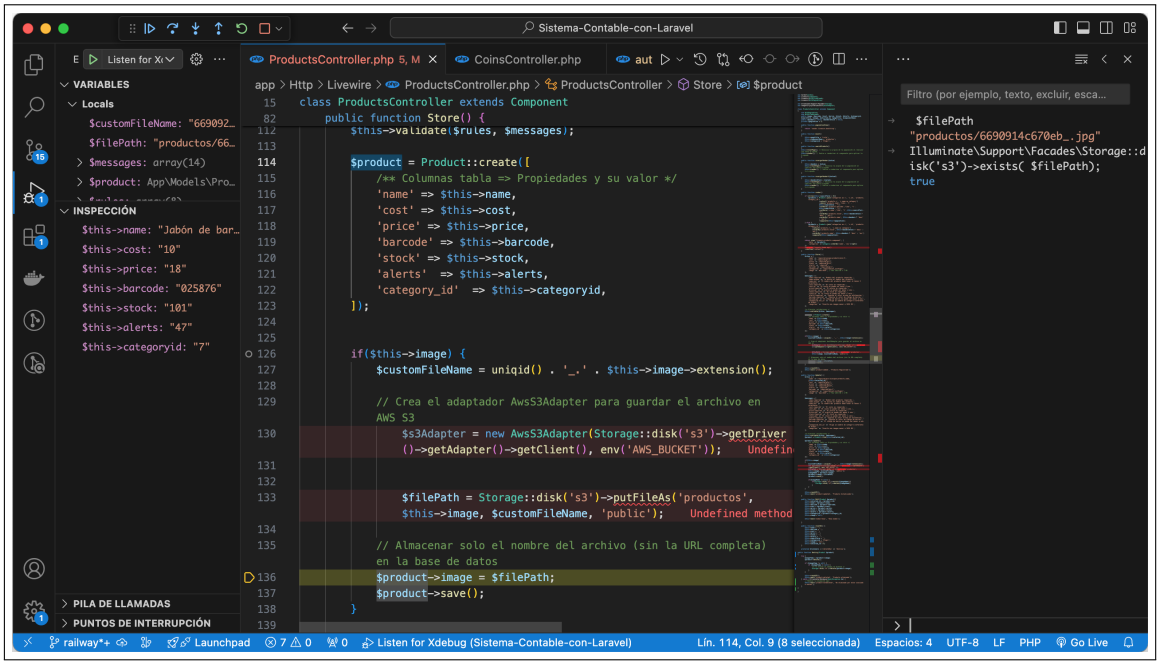

Figura 4.43: Detalle de la depuración con Xdebug para la creación de un producto.

En la Figura 4.44 se detalla y comprueba que el objeto (imagen) que se intentó manipular se haya ejecutado correctamente su inserción.

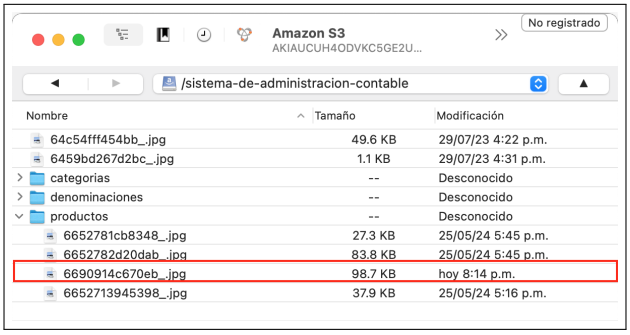

Figura 4.44: Detalle de revisión en Cyberduck para comprobar la subida de archivos.

4.4.2. Pruebas de caja negra

- **Pruebas de transición de estados**: Se analizaron las transacciones en distintos estados del módulo de ventas, desde la selección del producto y la gestión del stock hasta el registro de la venta y la generación del detalle correspondiente. El objetivo fue verificar la coherencia en cada una de estas etapas.

  Casos de Uso: Se realizaron pruebas basadas en escenarios de uso típicos del módulo de ventas. Por ejemplo:

- **Ejecutar compra y aumenta el stock**: Un usuario selecciona un producto válido, aumenta el stock através de las tres formas posibles y completa la compra. En la Figura 4.45 se puede observar el mensaje con el que se interactúa cuando se actualiza la cantidad de productos en el carrito de venta.

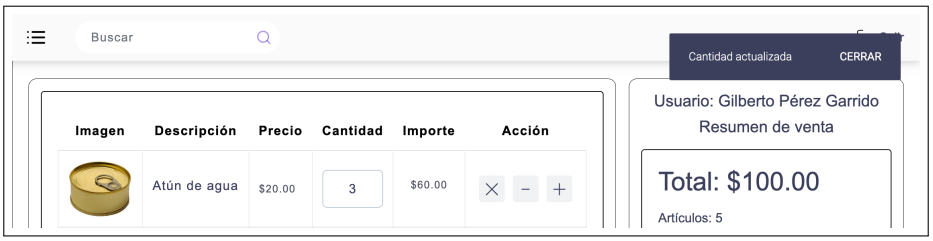

Figura 4.45: Detalle de mensajes en la interfaz para actualizar stock en el carrito de venta.

- **Ejecutar compra sin aumentar el stock**: Un usuario selecciona un producto y completa la compra sin aplicar ninguna intervención adicional en el stock para la compra de este producto. En la Figura 4.46 se detalla cómo se encuentra ejecutando el registro de una venta.

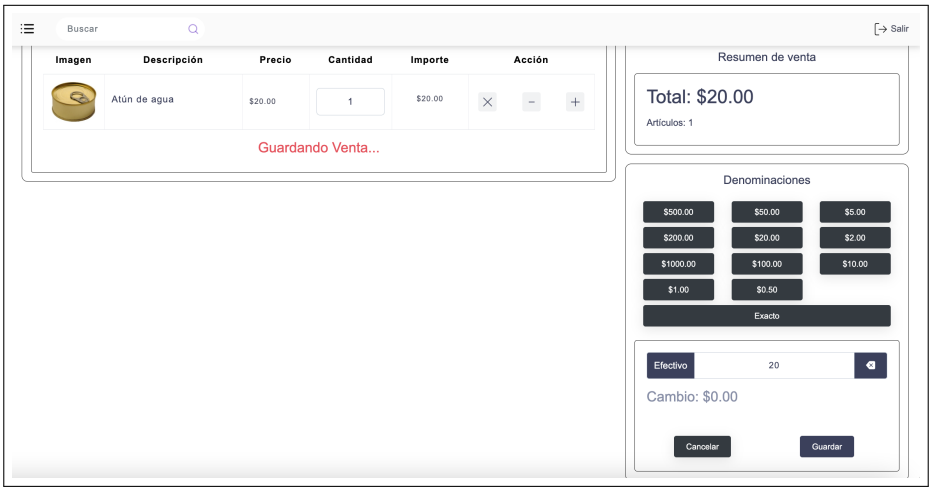

Figura 4.46: Detalle de mensajes para el guardado de ventas.

- **Compra con stock excedente**: Un usuario selecciona productos cuyo stock total supera el límite permitido de acuerdo a la disponibilidad.

Control de stock en módulo de ventas, para este ejemplo se mostrará cómo se interactua con el atributo stock de la tabla de productos (products). En la Figura 4.47 se puede observar la manipulación del siguiente producto, el cual se encuentra con stock en cero, y se intentará adjuntar en el carrito de ventas de la sesión activa:

Figura 4.47: Detalle del módulo de ventas con producto sin disponibilidad.

En la Figura 4.48 se puede observar el mensaje con el que se interactúa cuando se intenta actualizar la cantidad de productos en el carrito de venta, esto sin contar con la disponibilidad de la misma.

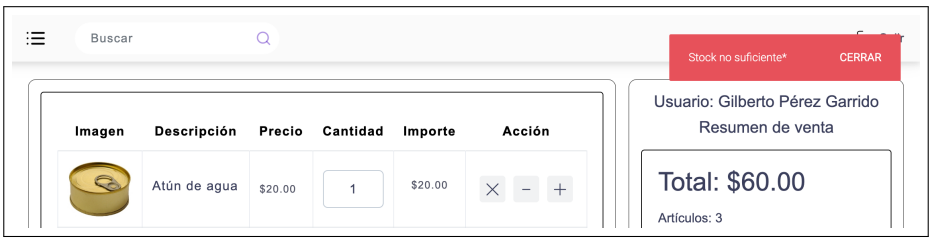

Figura 4.48: Detalle de mensajes en la interfaz para stock insuficiente.

Descartar producto en el carrito de ventas: Un usuario selecciona la eliminación de un producto en el carrito de ventas asociado a la sesión. En la Figura 4.49 se detalla el modal para confirmar la eliminación de un producto en el carrito de ventas asociado a la sesión para prevenir incidentes en caso de que el usuario haya hecho alguna selección de eliminación accidentalmente.

Figura 4.49: Detalle de modal en la interfaz para confirmar actualizar stock en el carrito de venta para eliminar.

En la Figura 4.50 se puede observar el mensaje de resultado al ejecutar la acción detallada en Figura 4.49

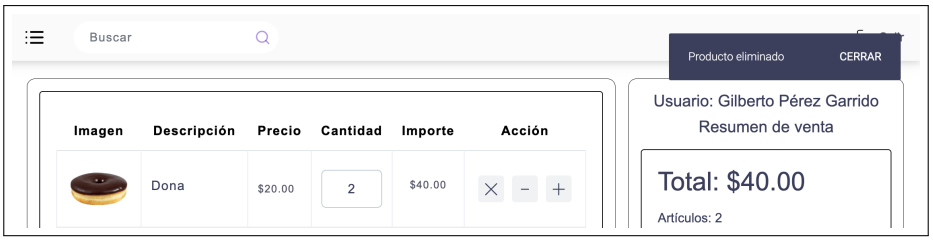

Figura 4.50: Detalle de mensajes en la interfaz para actualizar stock en el carrito de venta descartando de la lista.

Partición de equivalencia: Se identificaron conjuntos de entradas válidas y no válidas para el módulo de autenticación, se probaron los siguientes casos:

- **Casos exitosos**: Credendiales de usuario correctas y ser redirigido para contar con los accesos necesarios.

- **Casos no exitosos**: Credendiales de usuario incorrectas y ser notificado sin otorgar accesos de sesión.

Pruebas de valores límite: Se evaluaron los límites de las entradas para asegurar que el sistema maneje correctamente los valores en los bordes de los rangos permitidos para la creación de productos y ser usados en el módulo de ventas. Por ejemplo:

- **Monto base**: Se probaron inventario base de 1, como el mínimo permitido, esto para verificar la correcta aplicación de creación de productos y su uso en el módulo de ventas. Un ejemplo se puede observar en la Figura 4.11, durante la subsección "Desarrollo" para el caso del atributo stock de la tabla productos.

4.5. Implementación

En la Figura 4.51 se muestra el proceso de acceso a la instancia previamente configurada mediante el uso de llaves PEM en AWS. Estas llaves permiten autenticar de manera segura el acceso a las instancias EC2. También se incluye el proyecto clonado desde GitHub, utilizando el sistema de control de versiones Git (`Sistema-Contable-con-Laravel`).

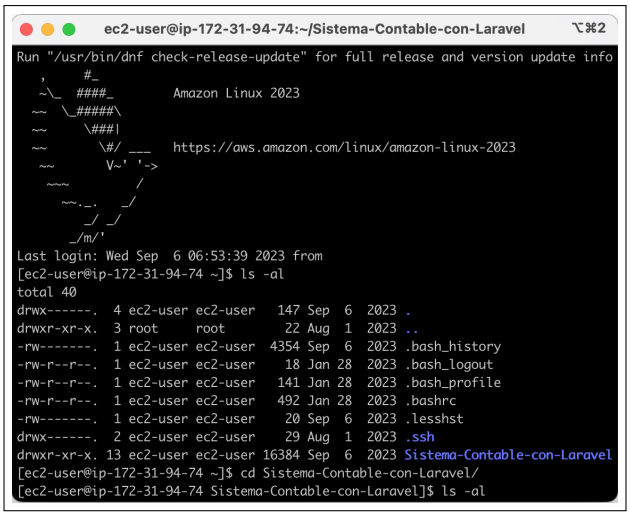

Figura 4.51: Detalle de acceso a EC2 mediate protocolo SSH.

En la Figura 4.52 se muestra la ejecución del contenedor Docker, configurado para exponer el puerto 80 (HTTP). La configuración inicial del contenedor se realizó desde una computadora personal con macOS. Posteriormente, se estableció una conexión SSH a una instancia EC2 que ejecuta Linux. Desde esta instancia EC2, se accedió al contenedor mediante Bash. En el contenedor, se ajustaron los accesos necesarios en el archivo (`.env`) para integrar Amazon RDS y Amazon S3. La instancia EC2, que aloja el contenedor, ejecuta un servidor Apache2 junto con PHP 7.3, permitiendo así servir los recursos de manera concurrente.

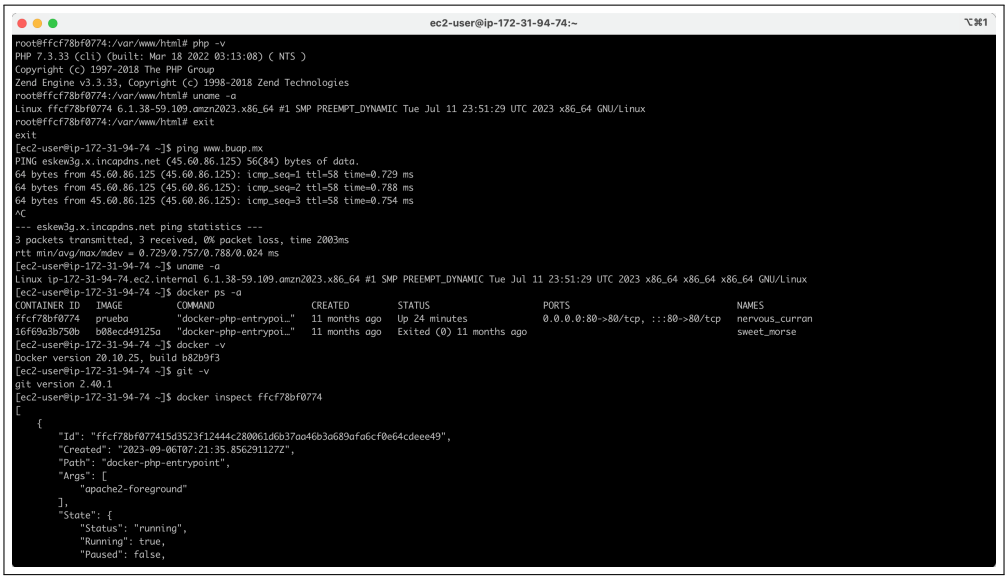

Figura 4.52: Detalle de acceso a EC2 mediate protocolo SSH y la ejecución del contenedor.

Capítulo 5

Resultados

Durante este capítulo se detalla la implementación del sistema contable tras la finalización del ciclo de vida del software. Se describe cómo el sistema, alojado en un contenedor, opera en una instancia de Amazon EC2 y utiliza los servicios de S3 y RDS con las configuraciones previamente especificadas. Además de la revisión de los Sprints realizados, en la cual se lleva a cabo un análisis detallado de los aspectos importantes relacionados con la descripción de cada historia de usuario y sus condiciones de ejecución.

5.1. Revisión de sprints

Verificando que el producto cumple con todos los requerimientos establecidos brindados por el usuario, los entregables son brindados al cliente. Según el autor (Pardo, Suescún, Jojoa, Zambrano, y Ordoñez, 2020) "El equipo de desarrollo se encarga de hacer realidad los deseos del cliente. Las personas que desempeñan este rol deben realizar los entregables funcionales planificados en cada Sprint" (p. 20).
Especificaciones de los Requerimientos, Según el autor (Pardo y cols., 2020) "Sabiendo que en la metodología ágil se debe llevar un monitoreo completo de las ediciones del sistema en base a los requerimientos, se plasma las especificaciones de cada requerimiento divididas por historias" (p. 20). En las siguientes tablas se hace hincapié en lo abordado previamente.

En Cuadro 5.1, Cuadro 5.2, Cuadro 5.3, Cuadro 5.4, Cuadro 5.5, Cuadro 5.6, Cuadro 5.7, Cuadro 5.8, Cuadro 5.9, Cuadro 5.10 y Cuadro 5.11 se detallan las historias de usuario mencionadas en el Cuadro 4.2. Cada cuadro corresponde respectivamente a una historia de usuario específica dentro del Cuadro 4.2.

Cuadro 5.1: Especificación de la historia Gestionar categorías para organizar productos

N° de especificación: 001	**Historia de Usuario**: Gestionar categorías para organizar productos
Descripción:	Se lleva a cabo la vista para el listado de categorías, en la cual permitirá dar acceso al registro de productos, edición y eliminación según sea el caso de los permisos por rol de usuarios en este módulo. Los usuarios que tendrán acceso son el rol de administrador, almacenes y ventas (sólo para leer).
Registro de categorías	
Condiciones de ejecución	En primer lugar, el campo 'nombre' es obligatorio y debe ser proporcionado por el usuario, ya que sin él no se puede identificar la categoría de manera adecuada. Además, el nombre de la categoría debe ser único dentro del sistema, lo que evita duplicidades y posibles confusiones. Por último, el nombre debe tener un mínimo de tres caracteres para garantizar que sea suficientemente descriptivo y útil.

Cuadro 5.2: Especificación de la historia Gestionar productos para mantener inventario

N° de especificación: 002	**Historia de Usuario**: Gestionar productos para mantener inventario
Descripción:	Se lleva a cabo la vista para el listado de productos, en la cual permitirá dar acceso al registro de productos, edición y eliminación según sea el caso de los permisos por rol. Los usuarios que tendrán acceso son el rol de administrador, almacenes y ventas (sólo para leer).
Registro de productos	

Condiciones de ejecución	En primer lugar, el campo 'nombre' es obligatorio y debe ser proporcionado por el usuario, ya que sin él no se puede identificar el producto de manera adecuada. Además, el nombre del producto debe ser único dentro del sistema, lo que evita duplicidades y posibles confusiones. Además, el nombre debe tener un mínimo de tres caracteres para garantizar que sea suficientemente descriptivo y útil. El campo 'costo' también es requerido y debe ser mayor a uno, asegurando que se registre un valor válido. De igual manera, el 'precio' es un campo obligatorio y debe ser superior a uno, para evitar precios no válidos. El 'stock' del producto es requerido y no puede ser menor a cero, garantizando que siempre se registre una cantidad válida de productos en inventario. Adicionalmente, se debe ingresar un valor mínimo en existencias para el campo 'alertas', y un valor para el 'código de barras', el cual no puede ser menor a uno. El 'id de categoría' es un campo obligatorio y debe ser diferente de la opción predeterminada 'Elegir'. Por último, si se adjunta una imagen del producto, esta debe ser menor a 1024 KB para asegurar que el archivo no sea demasiado grande.

Cuadro 5.3: Especificación de la historia Gestionar denominaciones para manejo de efectivo

N° de especificación: 003	**Historia de Usuario**: Gestionar denominaciones para manejo de efectivo
Descripción:	Se lleva a cabo la vista para el listado de denominaciones, en la cual permitirá dar acceso al registro de alguna denominación, edición y eliminación según sea el caso de los permisos por rol. Los usuarios que tendrán acceso son el rol de administrador.
Registro de denominaciones	

Condiciones de ejecución	En primer lugar, el campo 'tipo' es obligatorio y debe ser proporcionado por el usuario, ya que sin él no se puede clasificar la denominación de manera adecuada. Además, el tipo seleccionado debe ser diferente de la opción predeterminada 'Elegir', para asegurar que se elija un valor válido y significativo. El campo 'valor' también es requerido y debe ser único dentro del sistema, lo que evita duplicidades y posibles confusiones. Además, el valor debe ser mayor o igual a diez centavos, garantizando que se registre un monto significativo. Si alguna de estas condiciones no se cumple, se mostrará un mensaje de error específico al usuario para guiarlo en la corrección de la información ingresada.

Cuadro 5.4: Especificación de la historia Gestionar usuarios para control de acceso

N° de especificación: 004	**Historia de Usuario**: Gestionar usuarios para control de acceso
Descripción:	Se lleva a cabo la vista para el listado de usuarios, en la cual permitirá dar acceso al registro de estos, edición y eliminación según sea el caso de los permisos por rol. Los usuarios que tendrán acceso son el rol de administrador.
Registro de usuarios	
Condiciones de ejecución	En primer lugar, el campo 'nombre' es obligatorio y debe tener al menos tres caracteres para garantizar que el nombre del usuario sea suficientemente descriptivo. El campo 'correo' también es obligatorio y debe ser único dentro del sistema, además de tener un formato válido de correo electrónico para evitar duplicidades y errores en la comunicación. El campo 'estatus' es requerido y debe ser seleccionado, asegurando que no se use la opción predeterminada 'Elegir'. De igual manera, el campo 'perfil' es necesario y debe ser diferente de 'Elegir', garantizando que se seleccione un perfil o rol válido para el usuario. Por último, el campo 'contraseña' es obligatorio y debe tener un mínimo de tres caracteres para asegurar la seguridad y robustez del acceso al sistema. Si alguna de estas condiciones no se cumple, se mostrará un mensaje de error específico al usuario para guiarlo en la corrección de la información ingresada.

Cuadro 5.5: Especificación de la historia Gestionar roles y permisos de usuarios

N° de especificación: 005	**Historia de Usuario**: Gestionar roles y permisos de usuarios
Descripción:	Se implementa la vista para el listado de roles, que permite el acceso a gestionar productos, categorías, denominaciones, productos en el carrito de ventas, usuarios, roles y permisos. Además del acceso, asignación de permisos, corte de caja y reporte de ventas, esto según los permisos asignados a cada rol, haciendo uso del Middleware proporcionado por Spatie. Los usuarios que tendrán acceso son el rol de administrador.
Registro de roles	
Condiciones de ejecución	En primer lugar, el campo 'nombre del rol' es obligatorio y debe tener al menos dos caracteres para asegurar que el nombre del rol sea suficientemente descriptivo. Además, el nombre del rol debe ser único dentro del sistema, evitando duplicidades y posibles confusiones. Si alguna de estas condiciones no se cumple, se mostrará un mensaje de error específico al usuario para guiarlo en la corrección de la información ingresada.
Registro de permisos	
Condiciones de ejecución	En primer lugar, el campo 'nombre del permiso' es obligatorio y debe tener al menos dos caracteres para asegurar que el nombre del permiso sea suficientemente descriptivo. Además, el nombre del permiso debe ser único dentro del sistema, evitando duplicidades y posibles confusiones. Si alguna de estas condiciones no se cumple, se mostrará un mensaje de error específico al usuario para guiarlo en la corrección de la información ingresada.

Cuadro 5.6: Especificación de la historia Gestionar carrito de ventas para contabilizar total y artículos

N° de especificación: 006	**Historia de Usuario**: Gestionar carrito de ventas para contabilizar total y artículos
Descripción:	Se lleva a cabo la vista para la ejecución de ventas, en la cual permitirá dar acceso al registro de estas según sea el caso de los permisos por rol. Los usuarios que tendrán acceso son el rol de administrador y ventas.
Registro de ventas	

Condiciones de ejecución	Para proceder, es necesario agregar productos a la venta, asegurándose de que al menos haya un producto seleccionado en el carrito de ventas. Además, la cantidad de 'efectivo' utilizado para el pago debe ser superior a cero y el valor en el campo 'efectivo' debe ser mayor que la suma total de los precios de los productos en el carrito de ventas.
Ver resumen ventas en la interfaz	
Condiciones de ejecución	Para proceder, es necesario agregar productos a la venta, asegurándose de que al menos haya un producto seleccionado en el carrito de ventas. Además, la cantidad de 'efectivo' utilizado para el pago debe ser superior a cero y el valor en el campo 'efectivo' debe ser mayor que la suma total de los precios de los productos en el carrito de ventas. En automático aparece el nombre de usuario de la sesión que está a punto de ejecutar la venta y el resto de información como el cálculo del monto y la lista de productos en la interfaz.

Cuadro 5.7: Especificación de la historia Seleccionar denominaciones o efectivo para calcular cambio

N° de especificación: 007	**Historia de Usuario**: Seleccionar denominaciones o efectivo para calcular cambio
Descripción:	Se implementa la vista para seleccionar denominaciones, que a su vez representa el efectivo para realizar cálculos en cuanto al cambio. Los usuarios que tendrán acceso son el rol de administrador y el rol de ventas.
Registro de permisos	
Condiciones de ejecución	Primero, es necesario agregar productos y cumplir con las condiciones requeridas para registrar una denominación en el sistema. Esto permite obtener la suma de precios de los productos y la denominación deseada para proceder con el cálculo. Además, para guardar el registro de cálculos, se debe seleccionar una denominación mayor, ya sea mediante la selección de denominaciones predefinidas o ingresando manualmente la cantidad recibida del cliente para el pago.

Cuadro 5.8: Especificación de la historia Realizar búsquedas

N° de especificación: 008	**Historia de Usuario**: Realizar búsquedas
Descripción:	Se ha implementado una caja de texto que se reutiliza en todo el proyecto para realizar la búsqueda de productos, sin importar si se está en el módulo de ventas o no, siempre y cuando se haya iniciado sesión. También se incluye la posibilidad de buscar dentro de los listados del módulo de categorías, módulo de productos, módulo de denominaciones, módulo de usuarios, módulo de roles y módulo de permisos. Este proceso está asociado según los accesos por rol y/o permisos. Se estima que el rol de almacenes pueda buscar en el módulo de categorías y módulo de productos. Se estima que el rol de ventas pueda buscar en el módulo de categorías, módulo de productos, y utilizar la búsqueda del banner en el módulo de ventas. Se estima que el rol de administrador pueda buscar en el módulo de categorías, módulo de productos, módulo de denominaciones, módulo de usuarios, módulo de roles, módulo de permisos, y utilizar la búsqueda del banner en el módulo de ventas.
Realizar búsquedas de productos para agregar al carrito de ventas	
Condiciones de ejecución	En primer lugar, debe haber al menos un producto registrado previamente para que pueda ser encontrado mediante el buscador correspondiente, que es el banner. Si el código de barras no se encuentra, se notificará al usuario. Si el código de barras es encontrado, se notificará al usuario y se agregará el producto al listado del carrito de ventas asociado a la sesión.
Realizar búsquedas dentro de módulos de gestión	

Condiciones de ejecución	En primer lugar, deben existir registros en los módulos correspondientes para poder realizar búsquedas. Para realizar una búsqueda en los listados, es necesario contar con acceso a uno o más módulos que permitan búsquedas, como el Módulo de Productos, Módulo de Categorías, Módulo de Ventas, Módulo de Denominaciones, Módulo de Usuarios, Módulo de Roles o Módulo de Permisos. Una vez dentro del módulo deseado, se debe ingresar el texto relacionado con la búsqueda en la caja te texto correspondiente, luego presionar la tecla Enter o simplemente dejar de seleccionar el campo de entrada que se encuentra entre el banner superior y las listas de resultados devueltos por el sistema en cada uno de estos módulos.

Cuadro 5.9: Especificación de la historia Generar corte de caja por rango de fechas

N° de especificación: 009	**Historia de Usuario**: Generar corte de caja por rango de fechas
Descripción:	Se lleva a cabo la vista para la selección de denominaciones en el módulo de ventas, en la cual permitirá dar acceso al registro de productos en sesión (de manera temporal), edición (cambiar la cantidad por producto en el carrito de ventas) y eliminación (de los registros temporales) según sea el caso de los permisos por rol. Los usuarios que tendrán acceso son el rol de administrador.
Registro de productos en el carrito de ventas	
Condiciones de ejecución	Suma el valor de la denominación seleccionada al total de efectivo acumulado. Después de cada transacción de efectivo, se calcula el cambio como la diferencia entre el efectivo acumulado y el total a pagar.

Cuadro 5.10: Especificación de la historia Ver resumen de ventas en la interfaz

N° de especificación: 010	**Historia de Usuario**: Ver resumen de ventas en la interfaz en PDF/Excel
Descripción:	Se lleva a cabo la vista y los apartados para poder mostrar un resumen de ventas en la interfaz de usuario de ventas realizadas o ventas por realizar.
Ver resumen de ventas	

Condiciones de ejecución	Para realizar una consulta, es necesario especificar si se desea consultar entre todos los usuarios registrados o un usuario específico que haya tenido acceso al módulo de ventas y haya registrado ventas exitosamente. Luego, se debe seleccionar el rango de fechas mediante la elección de una fecha de inicio y una fecha de fin. Si algún registro cumple con los requisitos especificados, se mostrarán automáticamente en una lista con la opción de abrir el detalle. Alternativamente, los resultados pueden descargarse en un documento Excel o PDF según los mismos criterios especificados.

Cuadro 5.11: Especificación de la historia Acceder al sistema desde diversas locaciones

N° de especificación: 011	**Historia de Usuario**: Acceder al sistema desde diversas locaciones
Descripción:	Se ha implementado la vista para el inicio de sesión. Este proceso es público y no está asociado a un rol específico, pero es necesario proporcionar las credenciales para evaluar los roles y permisos del usuario.
Manejo de sesiones	
Condiciones de ejecución	Para iniciar sesión, es necesario proporcionar las credenciales suministradas previamente por un administrador. Para cerrar sesión, se debe seleccionar el apartado "Salir"que se encuentra en el banner después de iniciar sesión. Si las credenciales no coinciden, se notificará al usuario. Si las credenciales son correctas, el usuario será redirigido al sistema.

5.2. Simulación de resultados

La Figura 5.1 muestra cómo acceder al sistema desde diferentes dispositivos simultáneamente.

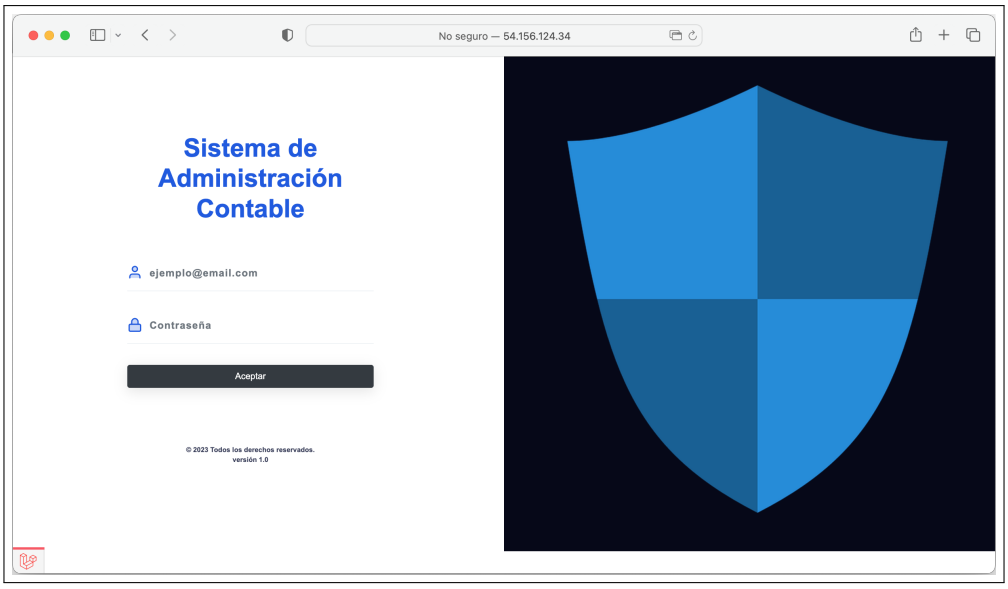

Figura 5.1: Detalle de autenticación para inicio de sesión (Infraestructura).

En la Figura 5.2 se muestra el uso del menú lateral izquierdo, de fondo negro y desplazable verticalmente, para navegar entre: Categorías, Productos, Ventas, Roles, Permisos, Asignar Permisos, Usuarios, Monedas, Corte de Caja y Reportes. En el listado de productos, encontrarás columnas que incluyen la descripción del producto, el código de barras único, la categoría, el precio de venta, el stock disponible y el inventario mínimo. Este último se utiliza con la intención de activar una acción asincrónica que alerta al administrador cuando es necesario actualizar el stock disponible.

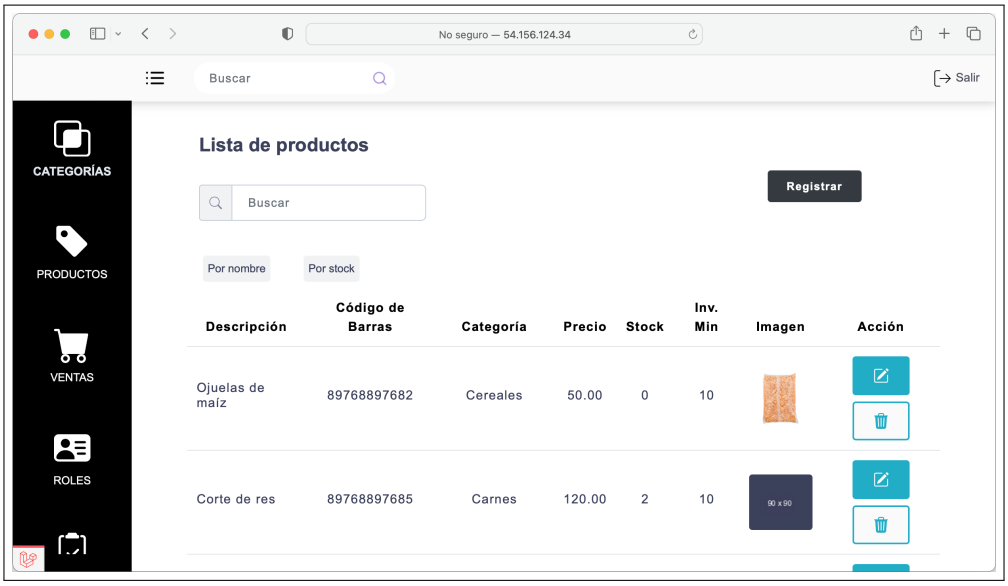

Figura 5.2: Detalle de listado de productos desde la instancia (Infraestructura).

La Figura 5.3 muestra la interfaz completa del módulo de ventas. Se puede ver el usuario que inició sesión intentando agregar el producto "Jabón de barra"(código de barras 025876) al carrito de ventas. El producto se busca en la tabla de productos, y el sistema notificará mediante eventos del DOM si el producto se encuentra disponible.

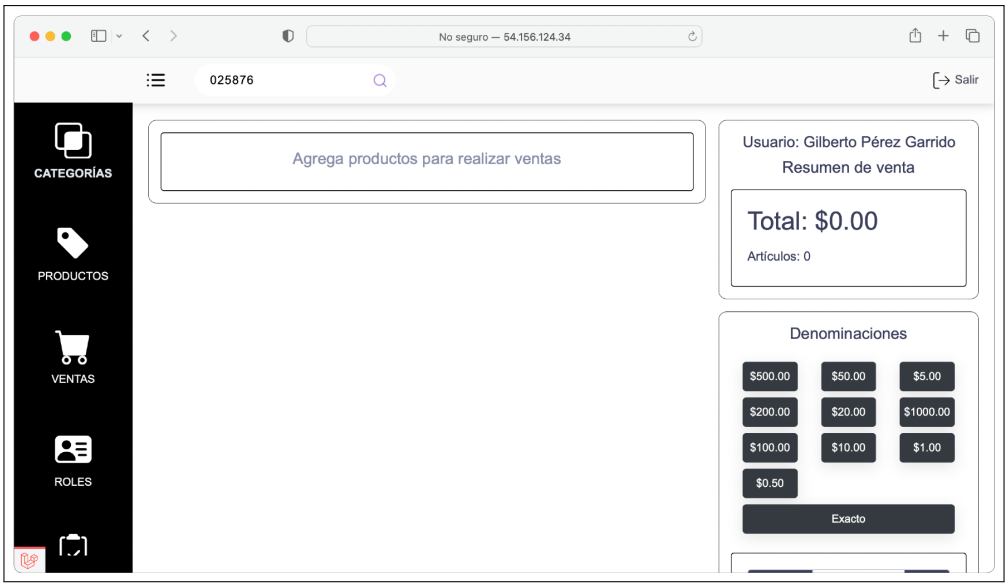

Figura 5.3: Detalle de productos sesión (Infraestructura).

En la Figura 5.4 se ilustra que el producto con código de barras 025876 ha sido agregado exitosamente al carrito de ventas. Se muestra el usuario asociado a la sesión, el total a pagar según los productos registrados en el carrito, con su respectiva cantidad y precio, calculado y mostrado en el detalle “Total $”. También se indica el número de artículos, que en este caso es uno.

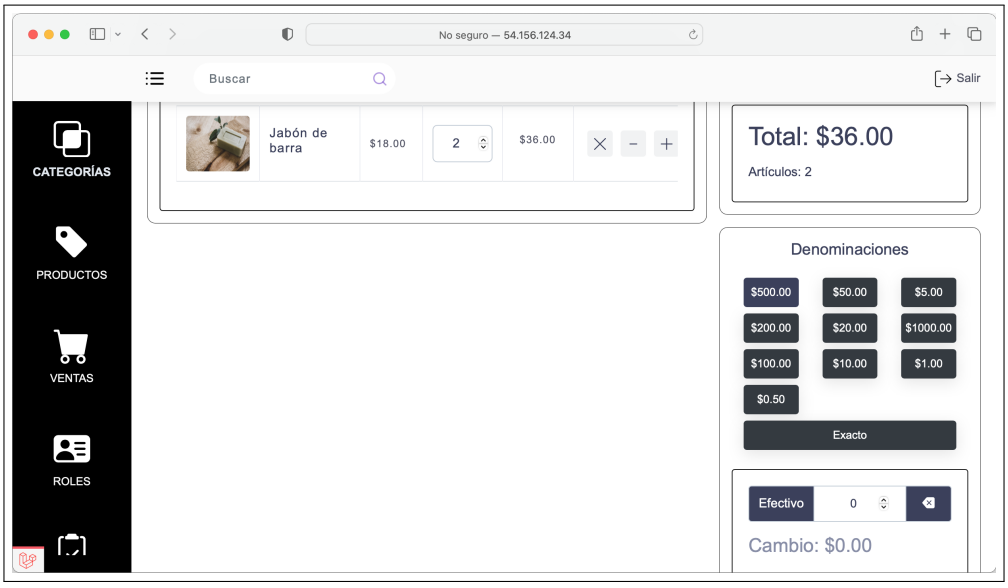

Figura 5.4: Detalle de módulo de ventas (Infraestructura).

En la Figura 5.5 se muestra cómo localizar un producto especificando la fecha por día para todos los usuarios, sin necesidad de definir un rango de fechas. En el ejemplo proporcionado, se puede encontrar la venta efectuada y optar por descargarla en formato CSV o PDF, o ver el detalle a través de la interfaz de usuario.

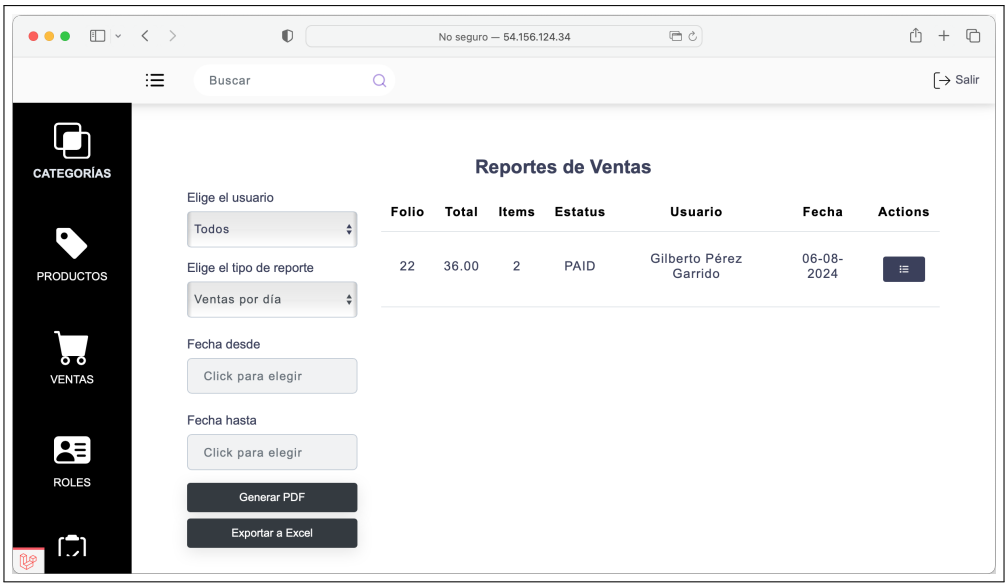

Figura 5.5: Detalle de reportes (Infraestructura).

En la Figura 5.6 se muestra que se puede comprobar el detalle de venta realizado en la figura 5.5, dónde folio es el identificador utilizado en la tabla de ventas

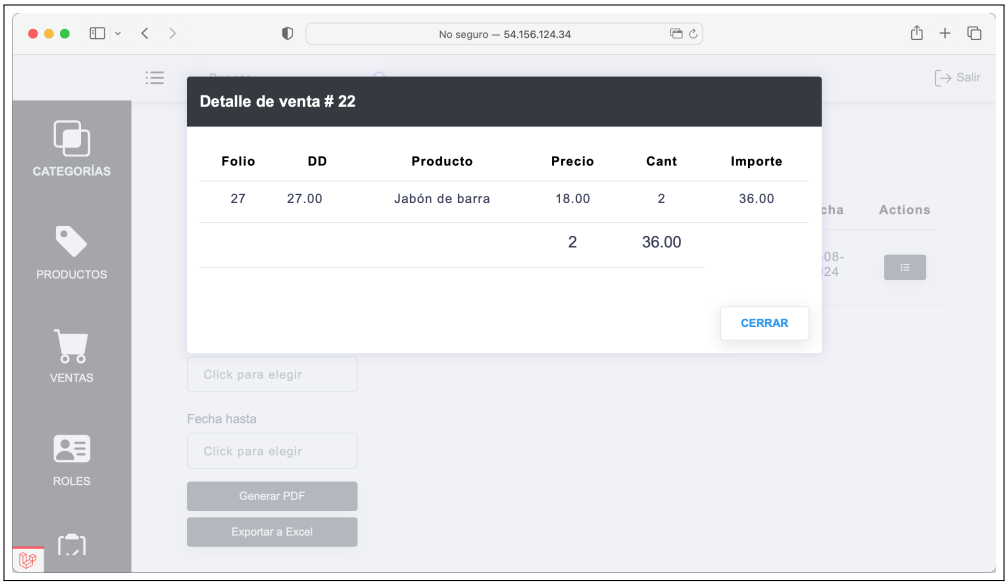

Figura 5.6: Detalle de productos (Infraestructura).

Capítulo 6

Conclusiones y trabajo a futuro

1. Se logró realizar el diagnóstico del estado actual del proceso contable en la investigación realizada, concluyendo que era necesaria la creación de una aplicación web para la administración contable con el fin de **optimizar las operaciones** habituales en los negocios de este rubro. Esto incluye la simplificación de procesos mediante el uso de una estructura determinada en el diseño de la Base de Datos para la manipulación de conglomeraciones de información a través de búsquedas en los módulos, la selección de rangos de fechas o filtros por usuarios, y la **automatización de tareas repetitivas** como el cálculo automático de montos, además accediendo al registro de ventas mediante reportes en la interfaz o documentos PDF o Excel para un análisis específico. Se **mejora la comunicación** tanto interna como externa mediante la opción inicial de establecer tareas claras en cuanto al acceso a los módulos o políticas establecidas, utilizando roles y permisos.

2. Se diseñó el sistema contable siguiendo el **ciclo de vida del desarrollo de software**. En la etapa de desarrollo, se implementó la metodología Scrum, concluyendo con la agrupación de sprints y sus respectivas historias de usuario para un mejor control de los procesos, se validaron las pruebas de caja blanca y caja negra dentro de este ciclo, enfocándose principalmente en el módulo de ventas.

3. Se elaboró el sistema contable con la alimentación de datos de prueba (Seeders), concluyendo como resultado final la disminución de tiempos en cuanto a las tareas habituales de las mismas.

4. Se **concluye** que el sistema puede ser utilizado en diferentes tipos de negocios, no solo en aquellos dedicados a la venta de abarrotes. Es aplicable a cualquier giro comercial que requiera una gestión eficiente de inventarios, ventas y operaciones, incluyendo sectores como la distribución, retail, servicios y manufactura.

5. Se **concluye** que la contenedorización con Docker es altamente escalable en Laravel y puede ser especialmente útil para proyectos grandes o que necesitan manejar una alta demanda. Docker permite empaquetar la aplicación y sus dependencias en un contenedor que se puede replicar fácilmente, facilitando el despliegue en diferentes entornos, ya sea local, de prueba o producción. Esto hace que la escalabilidad sea más manejable porque puedes desplegar

múltiples contenedores para ejecutar varias instancias de la aplicación Laravel, permitiendo el balanceo de carga y mejorando la capacidad de respuesta.

6. Para el **trabajo a futuro**, se está considerando la posibilidad de descartar el uso de Livewire. Esto se debe a que, durante la depuración con herramientas como DevTools desde el frontend, Livewire no resulta lo suficientemente explícito para mi preferencia personal. Sin embargo, esto no implica que Livewire no sea una buena opción para el progreso en la configuración de código. Por razones personales, me parece más interesante trabajar esa parte mediante las configuraciones que provee la especificación del ecosistema de Laravel denominada Laravel Mix. Laravel Mix facilita la gestión de assets y la compilación de archivos, proporcionando una forma más clara y explícita de manejar el frontend, incluyendo el empaquetado de JavaScript, CSS y otros recursos, o explorando ciertas características mediante servicios web y frontend con React para aplicaciones de una sola página (SPA). Además, sería necesario detallar los permisos por rol de usuario según el análisis de alto nivel especificado. No obstante, ya se han definido permisos iniciales, como la capacidad de modificar productos y categorías, asignados exclusivamente a los roles de administrador y almacén, respectivamente. La opción por Middleware ya cuenta con la definición inicial por rol, aunque cada negocio puede tener preferencias específicas en cuanto a estos accesos o permisos dentro de los accesos.

7. Además, en **trabajo a futuro**, se planea estandarizar algunos títulos que, debido a decisiones iniciales en el desarrollo, fueron maquetados en inglés. Esto es evidente en casos como `Actions` en lugar de `Acciones`, `Items` en lugar de `Cantidad`, y `PAID` en lugar de `Pagado`, algunas abreviaturas, `DD` en lugar de `Precio unitario`, `Cant` en lugar de `Cantidad`. Sin embargo, es importante destacar que ciertos aspectos de bajo nivel, como los nombres de las tablas en la base de datos, deben mantenerse en inglés, ya que asegura que el framework pueda aplicar automáticamente sus convenciones y generar código relacionado, como migraciones y controladores, de manera correcta. Otros aspectos que siguen esta estandarización incluyen las políticas de autorización y las configuraciones predeterminadas en Laravel, que también están en inglés. Esto se debe a varias razones. Primero, la documentación oficial de Laravel, que está en inglés, proporciona ejemplos y explicaciones que facilitan la comprensión y la implementación de estas políticas y configuraciones cuando se adhieren a los nombres y convenciones en inglés. Además, utilizar inglés asegura que el código sea uniforme y comprensible para otros desarrolladores que están familiarizados con el framework, sin importar su idioma nativo.

Capítulo 7

Anexos

Configuraciones en apache2 para el archivo de configuración Código 7.1 y 7.2.

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/public
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Código 7.1: Detalle de 000-default.conf.

```
    ServerName server
    ServerRoot "/etc/apache2"
    <Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
    </Directory>
    <Directory /usr/share>
        AllowOverride None
        Require all granted
    </Directory>
    <Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride None
        Require all granted
    </Directory>
    AccessFileName .htaccess
```

Código 7.2: Detalle de fragmento importante en apache2.conf (no completo).

Referencias

Abdellatif, A., Zeng, Y., Elshafei, M., Shihab, E., y Shang, W. (2020). Simplifying the search of npm packages. *Information and Software Technology*, *126*, 106365.

Abudureheman, A. (2017). Creating a cost saving web application.

Álvarez Flores, F. V. (2015). Plataforma interactiva multiusuario de gestión y análisis del proyecto ciempiess-unam.

Amazon Web Services. (2024a). *Réplicas de lectura de amazon rds.* Descargado de `https://aws.amazon.com/es/rds/features/read-replicas/` (Accedido: 2024-07-25)

Amazon Web Services. (2024b). *Soluciones de aws marketplace para amazon linux.* Descargado de `https://aws.amazon.com/es/mp/solutions/amazonlinux/` (Accedido: 2024-08-07)

Amazon Web Services. (2024c). *Zonas de disponibilidad múltiples (multi-az) de amazon rds.* Descargado de `https://aws.amazon.com/es/rds/features/multi-az/` (Accedido: 2024-07-25)

Amazon Web Services. (2024d). *Zonas de disponibilidad múltiples (multi-az) de amazon rds.* Descargado de `https://aws.amazon.com/es/rds/features/multi-az/` (Accedido: 2024-07-25)

Arvindhan, M., y Anand, A. (2019). Scheming an proficient auto scaling technique for minimizing response time in load balancing on amazon aws cloud. En *International conference on advances in engineering science management & technology (icaesmt)-2019, uttaranchal university, dehradun, india.*

Bagwan, M. K., y Ghule, P. S. (2019). A modern review on laravel-php framework. *Ire Journals*, *2*(12), 1–3.

Bai, J., Jhaney, I., Wells, J., y cols. (2019). Developing a reproducible microbiome data analysis pipeline using the amazon web services cloud for a cancer research group: proof-of-concept study. *JMIR medical informatics*, *7*(4), e14667.

Balaji, S., y Murugaiyan, M. S. (2012). Waterfall vs. v-model vs. agile: A comparative study on sdlc. *International Journal of Information Technology and Business Management*, *2*(1), 26–30.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... others (2001). *The agile manifesto.* The Agile Alliance, www. agilemanifesto. org.

Bou, R. C. (2019). *Usando xampp con bootstrap y wordpress.* Mercedes Gómez Alcalá.

Bourne, S. R. (1978). Unix time-sharing system: The unix shell. *The Bell System Technical Journal*, *57*(6), 1971–1990.

Chen, X., Ji, Z., Fan, Y., y Zhan, Y. (2017, 10). Restful api architecture based on laravel framework. *Journal of Physics: Conference Series*, *910*, 012016. doi: 10.1088/1742-6596/910/1/012016

Combaudon, S. (2018). *Mysql 5.7: administración y optimización.* Ediciones Eni.

Contributors, S. O. (s.f.). *Learning laravel ebook.* Free, Unaffiliated eBook created from Stack Overflow Contributors. Descargado de `https://riptutorial.com/Download/laravel.pdf`

Corona, A. E., y cols. (2004). Protocolos tcp/ip de internet.

Cosentino, V., Luis, J., y Cabot, J. (2016). Findings from github: methods, datasets and limitations. En *Proceedings of the 13th international conference on mining software repositories* (pp. 137–141).

Díaz, C. A. (2019). *Programacion en java i: El entorno de programación–sintaxis–elementos–estructuras de control* (Vol. 1). RedUsers.

Docker, I. (2020). Docker. *línea].[Junio de 2017]. Disponible en: https://www. docker. com/what-docker*.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., y Berners-Lee, T. (1999). *Rfc2616: Hypertext transfer protocol–http/1.1.* RFC Editor.

Gauchat, J. D. (2012). *El gran libro de html5, css3 y javascript.* Marcombo.

Gien, M. (1978). A file transfer protocol (ftp). *Computer Networks (1976)*, *2*(4-5), 312–319.

Glinz, M. (2007). On non-functional requirements. En *15th ieee international requirements engineering conference (re 2007)* (pp. 21–26).

Gómez, Á. P., Jalca, J. J. R., García, J. G., Sánchez, O. Q., Parrales, K. M., y Merino, J. M. (2017). *Fundamentos sobre la gestión de base de datos* (Vol. 23). 3Ciencias.

Guevara, C. (2017). Desarrollo de un sistema en entorno web para el control de la gestión del inventario de la empresa cuenca llantas, utilizando como framework de desarrollo laravel. *Universidad de Guayaquil, Guayaquil, Ecuador*.

Guilindro Moreira, Y. W. (2022). *Análisis comparativo de tecnologías django js vs laravel, en el modelo de procesos para el desarrollo de aplicaciones web.* (B.S. thesis). Babahoyo: UTB-FAFI. 2022.

INEGI. (2019). *Las tiendas de conveniencia o minisúper en méxico.*

Josar, C. (2011). La contabilidad y el sistema contable. *Gestiopolis. Recuperado de https://www. gestiopolis. com/la-contabilidad-y-el-sistema-contable*.

Laureano, M., Maziero, C., y Jamhour, E. (2004). Intrusion detection in virtual machine environments. En *Proceedings. 30th euromicro conference, 2004.* (pp. 520–525).

Leal, E. T., Sosa, C. M., y Leal, D. A. T. (2012). Revisión de los sistemas de control de versiones utilizados en el desarrollo de software. *Ingenierías USBMed*, *3*(1), 74–81.

Love, R. (2010). *Linux kernel development.* Pearson Education.

Luján-Mora, S. (2002). *Programación de aplicaciones web: historia, principios básicos y clientes web.* Editorial Club Universitario.

Microsoft. (2024). *Visual studio code.* `https://visualstudio.microsoft.com/es/`. (Consultado el 3 de septiembre de 2024)

Mohamad Adib, M. F. (2014). Search result synonymy indexing for social network using latent semantic analysis.

Morris, J., McCubbin, C., y Page, R. (2019). *Hands-on data science with the command line: Automate everyday data science tasks using command-line tools.* Packt Publishing Ltd.

Muñoz Escoí, F. D., Argente Villaplana, E., Espinosa Minguet, A. R., Galdamez Saiz, P., García Fornes, A. M., Juan Marín, R. d., y Sendra Roig, J. S. (2013). *Concurrencia y sistemas distribuidos.* Editorial Universitat Politècnica de València.

Murty, J. (2008). *Programming amazon web services: S3, ec2, sqs, fps, and simpledb.* OReilly Media, Inc.

Navarro Silva, O., López Macas, M. E., y Pérez Espinosa, M. J. (2017). Normas de control contable: operación imprescindible en la gestión empresarial: un caso ecuatoriano. *Revista Universidad y Sociedad*, *9*(3), 46–51.

Navas, F. C., y Gallego, A. D. (s.f.). Base de datos distribuidas.

Network, M. D. (Sin fecha). *Compilar.* Descargado de `https://developer.mozilla.org/es/docs/Glossary/Compile`

Ozar, G. (2012). *Mysql management and administration with navicat.* Packt Publishing Ltd.

Pardo, C., Suescún, E., Jojoa, H., Zambrano, R., y Ordoñez, W. A. O. (2020). Modelo de referencia para la adopción e implementación de scrum en la industria de software. *Investigación e Innovación en Ingenierías*, *8*(3), 14–28.

Pérez Ibarra, S. G., Quispe, J. R., Mullicundo, F. F., y Lamas, D. A. (2021). Herramientas y tecnologías para el desarrollo web desde el frontend al backend. En *Xxiii workshop de investigadores en ciencias de la computación (wicc 2021, chilecito, la rioja).*

PK, A. D., y Jevitha, K. (s.f.). Stride based analysis of the chrome browser extensions api.

Pulido Fuentes, K. N., Martínez, M. J., y cols. (2022). *Desarrollo de una aplicación web para la inscripción de cursos libres del departamento de computación de la unan-león, mediante la utilización del framework laravel livewire de php* (Tesis Doctoral no publicada).

Rad, N. K., y Turley, F. (2019). *Los fundamentos de agile scrum.* Van Haren.

Razzaque, M. A., Milojevic-Jevric, M., Palade, A., y Clarke, S. (2015). Middleware for internet of things: a survey. *IEEE Internet of things journal*, *3*(1), 70–95.

Rojo, S. d. V. (2012). *Requerimientos no funcionales para aplicaciones web* (Tesis Doctoral no publicada). Universidad Nacional de La Plata.

Romero, L. M. R. (s.f.). Estrategias y herramientas para depuración de código en el back-end. *Cuadernos Técnicos Universitarios de la DGTIC*, *1*(1).

Salazar Cárdenas, J. E. (2014). Análisis comparativo de dos bases de datos sql y dos bases de datos no sql.

Sánchez, B. R., Matamoros, K. L. V., Hidalgo, K. G., Gutiérrez, J. L. C., y Lozada, M. Á. R. (2019). Situación económica de las mipymes de abarrotes en xaloztoc, tlaxcala méxico y su capitalización. *Revista de investigación interdisciplinaria en métodos experimentales*, *1*(8), 77–98.

Saucedo, Á. E. P., y del Castillo Serpa, A. (2012). Aplicación de la tabla ortogonal en el diseño de los casos de prueba de software. *Avanzada Científica*, *15*(2), 1–12.

Schneider, P. (1996). Tcp/ip traffic classification based on port numbers. *Division Of Applied Sciences, Cambridge, MA*, *2138*(5), 1–6.

Schwaber, K., y Sutherland, J. (2017). *The scrum guide.* Descargado de `https://scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf`

Setiawan, T. B. (2017). *Sistem informasi pengolahan data nilai siswa sekolah menengah kejuruan (smk) pgri 2 salatiga berbasis web menggunakan framework codeigniter* (Tesis Doctoral no publicada). Program Studi Teknik Informatika FTI-UKSW.

Silva, H. (2017). *Desarrollo e implementación de un sistema web con mvc para el control del mantenimiento preventivo y correctivo de los bienes del cuerpo de bomberos del gobierno autónomo descentralizado municipal de santo domingo; periodo 2016-2017* (B.S. thesis). Pontificia Universidad Católica del Ecuador sede Santo Domingo, Escuela de Sistemas, Santo Domingo, Ecuador.

Solutions, M. (2021). *Diferencias entre development, stage y production.* Descar-

gado de https://www.linkedin.com/pulse/diferencias-entre-development-stage-y-production-meraki-s/?originalSubdomain=es

Sommerville, I. (2005). Requerimientos del software. *Ingeniería del software, 7a ed., PEARSON EDUCACIÓN, Madrid, SPA*, 109–110.

Soni, M. (2018). *Practical aws networking: Build and manage complex networks using services such as amazon vpc, elastic load balancing, direct connect, and amazon route 53*. Packt Publishing Ltd.

Spurlock, J. (2013). *Bootstrap: responsive web development*. OReilly Media, Inc.

Surguy, M. (2014). Laravel: my first framework. *Brimingham: Leanpub*, 1–165.

Ylonen, T., y Lonvick, C. (2006). *The secure shell (ssh) protocol architecture* (Inf. Téc.).

Zandstra, M. (2013). *Php objects, patterns, and practice*. Apress.