UNIVERSIDAD TÉCNICA DEL NORTE

Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Ingeniería en Sistemas Computacionales

"DISEÑO DE UN ECOSISTEMA DE SOFTWARE, PARA LA INTEROPERABILIDAD ENTRE SISTEMAS DE E-COMMERCE Y COURIER MEDIANTE APIS RESTFUL EFICIENTES Y SEGURAS"

Trabajo de grado previo a la obtención del título de ingeniero en Sistemas Computacionales

Autor:

Jorge Arturo Castro Querembás

Director:

Ing. Diego Javier Trejo España, MSC.

Ibarra – Ecuador

2020

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN
A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1004201792		
APELLIDOS Y NOMBRES:	Castro Querembás Jorge Arturo		
DIRECCIÓN:	Ecuador, Imbabura, Antonio Ante, Atuntaqui		
EMAIL:	azarturowe@hotmail.com		
TELÉFONO FIJO:	(62)535-544	**TELÉFONO MÓVIL:**	0996236541

DATOS DE LA OBRA	
TÍTULO:	"Diseño de un ecosistema de software, para la interoperabilidad entre sistemas de E-commerce y Courier mediante APIs Restful eficientes y seguras"
AUTOR:	Jorge Arturo Castro Querembás
FECHA:	2021-01-04
PROGRAMA:	Pregrado
TÍTULO POR EL QUE OPTA:	Ingeniero en Sistemas Computacionales
ASESOR /DIRECTOR:	MSc. Diego Trejo

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de esta y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 4 días del mes de enero de 2021

EL AUTOR:

.....

Jorge Arturo Castro Querembás

UNIVERSIDAD TÉCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS
CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

Ibarra, 30 de noviembre 2020

CERTIFICACIÓN DEL DIRECTOR

Por medio del presente, yo MSc. Diego Trejo, certifico que el Sr. Jorge Arturo Castro Querembás, portador de la cédula de identidad Nro. 1004201792. Ha trabajado en el desarrollo del proyecto de grado **"DISEÑO DE UN ECOSISTEMA DE SOFTWARE, PARA LA INTEROPERABILIDAD ENTRE SISTEMAS DE E-COMMERCE Y COURIER MEDIANTE APIS RESTFUL EFICIENTES Y SEGURAS"**, previo a la obtención del título de Ingeniero en Sistemas Computacionales, lo cual ha realizado en su totalidad con responsabilidad.

Es todo cuanto puedo certificar en honor a la verdad.

Atentamente,

DIEGO JAVIER TREJO ESPANA

Firmado digitalmente por DIEGO JAVIER TREJO ESPANA
Fecha: 2020.12.01 08:45:44 -05'00'

MSc. Diego Trejo

DIRECTOR DE TESIS

UNIVERSIDAD TECNICA DEL NORTE
FACULTAD DE INGENIERIA EN CIENCIAS APLICADAS
CARRERA DE INGENIERIA EN SISTEMAS COMPUTACIONALES

CLUB ETHICAL HACKING UTN
18 de Agosto 2020

Certifica que al trabajo de titulación **DISEÑO DE UN ECOSISTEMA DE SOFTWARE, PARA LA INTEROPERABILIDAD ENTRE SISTEMAS DE ECOMMERCE Y COURIER, MEDIANTE APIS RESTFUL EFICIENTES Y SEGURAS** perteneciente al Sr. **Castro Querembas Jorge Arturo** con cédula de ciudadanía **1004201792**, se le realizó una evaluación de seguridad enfocado en el sistema de APIs, debido a que este componente fue desarrollado utilizando buenas prácticas de seguridad basados en **Open Web Application Security Project (OWASP)** , no se encontró ninguna vulnerabilidad crítica y logró superar con éxito las pruebas realizadas.

Es todo cuanto puedo mencionar en honor a la verdad, el propietario puede hacer libre uso de este documento.

Atentamente

Sr. Alexander Castro
Presidente CEH-UTN

Sr. Nelson Cacoango
Responsable de pruebas CEH-UTN

Tabla de Contenido

Índice de Figuras

Índice de Cuadros

Resumen

El E-Commerce y Courier en Ecuador necesitan interoperar para llevar a cabo el comercio electrónico en el país, por ello se propone un diseño de ecosistema de software que en base a buenas prácticas permita la interoperabilidad a sistemas de E-Commerce y Courier.

Para el desarrollo y diseño de este ecosistema de software se considera buenas prácticas las cuales fueron aplicadas para desarrollar un sistema distribuido de arquitectura orientada a servicios, es por ello por lo que el giro de negocio y el acceso a su data se desarrolla en base en APIs Restful las cuales deben cumplir todas las restricciones Rest, por otra parte este ecosistema debe tener la característica de poder cohesionar con otros módulos de un macroecosistema de software, para ello se aplica la ISO 8583 en puntos de acceso de interés que engloban transacciones financieras con mensajes originados en una tarjeta, además para comprobar su eficiencia se realiza varias pruebas de carga hasta llegar al estrés del API, obteniendo un resultado cuantitativo del índice APDEX con el que se determina la eficiencia y satisfacción, por otro lado para asegurar los sistemas Restful se utilizó las recomendaciones emitidas por OWASP, se evaluó el sistema con pruebas de vulnerabilidades realizado por un ente interno y externo.

Se obtuvo resultados en la comprobación de eficiencia y seguridad, se construyó un cliente móvil de las APIs desarrolladas. Todo el prototipo de ecosistema de software fue puesto en producción fuera de localhost, en sistemas de hosting compartido y nube.

Palabras Clave: E-Commerce, Courier, Interoperabilidad, ecosistema de software ISO 8583, Restful, APDEX, OWASP

Abstract

E-Commerce and Courier in Ecuador need to interoperate to carry out electronic commerce in the country, for this reason a software ecosystem design is proposed that based on good practices allows interoperability to E-Commerce and Courier systems.

For the development and design of this software ecosystem, good practices are considered which were applied to develop a distributed service-oriented architecture system, which is why the business line and access to its data is developed based on Restful APIs which must comply with all Rest restrictions, on the other hand, this ecosystem must have the characteristic of being able to unite with other modules of a software macroecosystem, for this, ISO 8583 is applied in access points of interest that include financial transactions with messages originated in a card, in addition to checking its efficiency, several load tests are carried out until reaching the API stress, obtaining a quantitative result of the APDEX index with which efficiency and satisfaction are determined, on the other hand, to ensure Restful systems are used the recommendations issued by OWASP, the system was evaluated with vulnerability tests performed or by an internal and external entity.

Results were obtained in the verification of efficiency and security, a mobile client of the developed APIs was built. The entire software ecosystem prototype was put into production off localhost, on shared and cloud hosting systems.

Keywords: E-Commerce, Courier, Interoperability, software ecosystem, Restful, ISO 8583, APDEX, OWASP

INTRODUCCIÓN

Antecedentes

Los modelos de negocio, acorde al auge tecnológico han comenzado a apoyar sus actividades para la oferta de productos y servicios en propuestas tecnológicas que agilicen el proceso compra entrega, alrededor del mundo Estados Unidos ha sido ejemplo de lo eficaz que pueden resultar los sistemas de comercio electrónico. En América Latina los sistemas “E-Commerce” han crecido en un plazo bianual de alrededor del 98.8% donde Brasil lidera el ranking con el 60% del comercio electrónico, mientras que Ecuador ha movido compras en línea por alrededor de 75.000 USD millones, que representa alrededor del 2% del comercio electrónico de América Latina, en Ecuador existe una alta demanda de sistemas de E-Commerce (Basantes, Gallegos, & Cathy, 2016).

Ecuador desarrolló la "Ley de Comercio Electrónico, Firmas Electrónicas y Mensajes de Datos", esta ley fue publicada en el registro oficial 557 el 17 de abril del 2002, a pesar de ello una de las problemáticas es que no existe una norma específica y especializada en materia de tributos operabilidad y seguridad del comercio electrónico (Laudon, Kearney, & Pueyrredon, 2014).

En la Universidad Técnica del Norte se ha realizado una investigación con título “Diseño de plataforma tecnológica de medio de pago local para el comercio electrónico en Ecuador, mediante la integración de infraestructuras de servicio tecnológico financiero (FINTEC)”(Trejo & Ortega, 2018), que propone varios aspectos fundamentales para integrar infraestructuras eficientes y seguras que persiguen apoyar a los modelos de E-Commerce en Ecuador.

A pesar de que no existe una normativa adecuada que motive buenas prácticas de eficiencia e interoperabilidad, el comercio electrónico sigue creciendo en el Ecuador y se constituye en alternativa para apalancar los giros de negocios de las pequeñas, medianas y grandes industrias del país.

Situación Actual

En Ecuador existe la necesidad de apoyar la interoperabilidad de cada subproceso del comercio electrónico, actualmente existen alternativas que carecen de una verdadera interoperabilidad y seguridad en el desarrollo del E-Commerce, sin embargo el comercio electrónico sigue creciendo, aunque muchas veces los clientes de los sistemas E-Commerce únicamente suelen disponer de informes finales y no son capaces para verificar el proceso en cada etapa, por esta razón se genera un alto nivel de incertidumbre, además,

en los consumidores existe una desconfianza muy alta, pues se sienten expuestos a estafas. Los consumidores en muchas ocasiones no confían en la veracidad de las empresas que venden sus productos en línea(GAIBOR, 2015).

Para apoyar a la mejora de sistemas seguros, eficientes e interoperables es necesario diseñar ecosistemas de software que implementen buenas prácticas basadas en estándares, el realizar prototipos contribuirá a mejorar la interoperabilidad del comercio electrónico, específicamente diseñar prototipos para mejorar la seguridad y eficiencia de la interoperabilidad de sistemas E-Commerce con sistemas Courier, posibilita a la posteridad el diseño y construcción de todo un ecosistema que implemente buenas prácticas teniendo como guía normas y estándares que permitirán el crecimiento adecuado del comercio electrónico, satisfacción y garantía para las partes del ecosistema diseñado.

Prospectiva

Los ecosistemas de software son entornos de trabajo en los que varias herramientas de software conviven y se comportan de acuerdo con buenas prácticas permitiendo a los usuarios y dispositivos interactuar de manera sobresaliente(Pozo, 2018). El diseñar un ecosistema de software para apoyar la interoperabilidad de los sistemas de E-Commerce y Courier posibilitará la adopción de buenas prácticas con base en estándares, mejorando la eficiencia y seguridad del canal en el cual interoperan, esto permitirá mejorar la confianza de estos sistemas, posibilitando la realización de una transacción segura, monitoreable la cual generará confianza en los clientes y usuarios de los sistemas de E-Commerce y Courier, esta confianza beneficiará el crecimiento de modelos de negocio y emprendimientos alrededor del comercio electrónico, permitiendo a países como Ecuador crecer dinámicamente y apoyar su cultura tecnológica, la seguridad de las transacciones financieras y crecimiento económico.

Planteamiento del Problema

Baja interoperabilidad y seguridad entre los sistemas Courier y sistemas E-Commerce en Ecuador, es decir es necesario diseñar un ecosistema de software para que puedan converger de manera óptima y segura aplicando buenas prácticas estandarizadas.

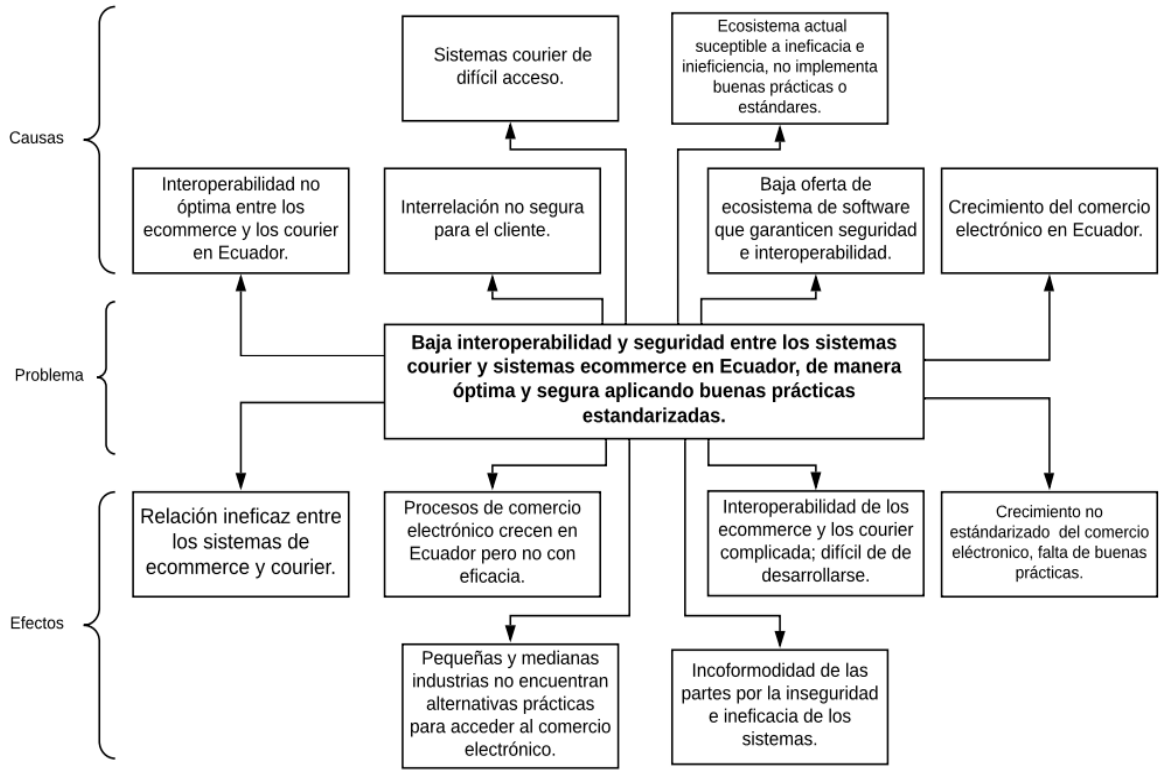


Fig. 1. Árbol del Problema
Fuente: Propia

Objetivos

Objetivo General

Diseñar un ecosistema de software que permita y optimice de manera segura la interacción entre los E-Commerce y sistemas Courier, desarrollando un prototipo de arquitectura de software que implemente condiciones óptimas aplicando la ISO 8583.

Objetivos Específicos

1. Definir un marco teórico acerca de los sistemas E-Commerce y Courier en el Ecuador, para comprender y diseñar un ecosistema de software que permita su interoperabilidad.
2. Desarrollar un prototipo para la exposición de APIs Restful, implementado del lado del E-Commerce y Courier para el consumo de APIs por medio de módulos móviles.
3. Implementar en el desarrollo y funcionamiento del prototipo buenas prácticas basadas en el estándar ISO 8583.
4. Evaluar parámetros de eficiencia y seguridad en el ecosistema de software propuesto, mediante consumidores de APIs Restful.

Alcance

El presente proyecto tiene como finalidad realizar el diseño de un ecosistema de software para la interoperabilidad eficiente y seguras de sistemas de E-Commerce y sistemas Courier, implementado para esta interoperabilidad buenas prácticas que tomen como base la norma ISO 8583, este estándar persigue el objetivo de estandarizar un contexto seguro para las transacciones comerciales y financiares, a través de canales y mensajes tecnológicos.

El diseño de un prototipo que implemente estas características requiere de un módulo web que permitirá exponer los servicios APIs Restful del E-Commerce hacia el Courier, estos servicios web deberán exponer información en formato JSON, es necesario asegurar el acceso a este formato, utilizando el estándar abierto JWT “JSON Web Tokens”. Los sistemas E-Commerce y Courier en el prototipo diseñado se constituirán en sistemas móviles, cada sistema dispondrá de la administración y gestión de su información en bases de datos, una vez realizado el diseño del prototipo y su construcción es necesario validar que el canal que implementa buenas prácticas propende a la operabilidad eficiente, para ello se evaluará mediante pruebas de integración de tipo funcional para servicios Restful, implementando un ambiente de desarrollo o cliente API que permita realizar estas pruebas.

A continuación, se muestra en formato de tabla las posibles tecnologías que se podrían implementar en la construcción y diseño que propone el presente proyecto.

TABLA 1
DISEÑO DE LA ARQUITECTURA DEL PROTOTIPO.

E-Commerce	**Courier**
Aplicación Móvil (Xamarin) Base de datos (SQL Server) JWT (JSON Web Token)	Aplicación Móvil (Xamarin) Base de datos (SQL Server) JWT (JSON Web Token)
Exposición de Servicios	
Sistema Web Front-end (Swagger UI) Back-end (.Net Core)	
Ambiente de desarrollo API	
Test funcionales (Postman)	

Fuente: Propia

En la figura 2 se puede apreciar un esquema gráfico que representa el ecosistema interoperabilidad entre un sistema de E-Commerce y un sistema Courier.

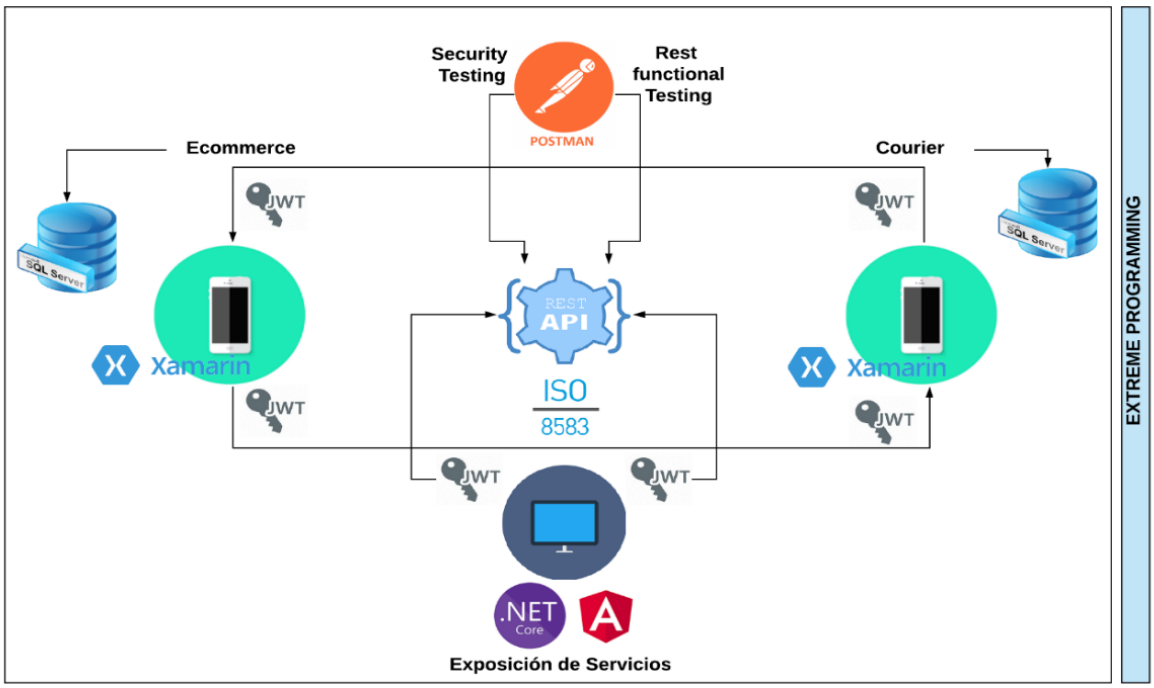


Fig. 2. Gráfico del ecosistema de software.
Fuente: Propia

El presente proyecto para su construcción implementará un marco de desarrollo ágil similar a “Extreme Programming”, esta metodología deberá adaptarse a las condiciones y los roles disponibles que para la construcción principalmente lo constituyen el programador y el coach, esta metodología XP adaptada permitirá planificar la construcción del prototipo buscando la eficiencia en tiempo del desarrollo y la cohesión del desarrollador y su coach.

“XP es un marco de desarrollo de software ágil que tiene como objetivo producir software de mayor calidad, el equipo de desarrollo. XP enmarca prácticas de ingeniería adecuadas para el desarrollo de software”(Appleton, 2017).

En la ilustración 3 se representa el posible flujo de trabajo que tendría la metodología adoptada “Extreme Programming”.

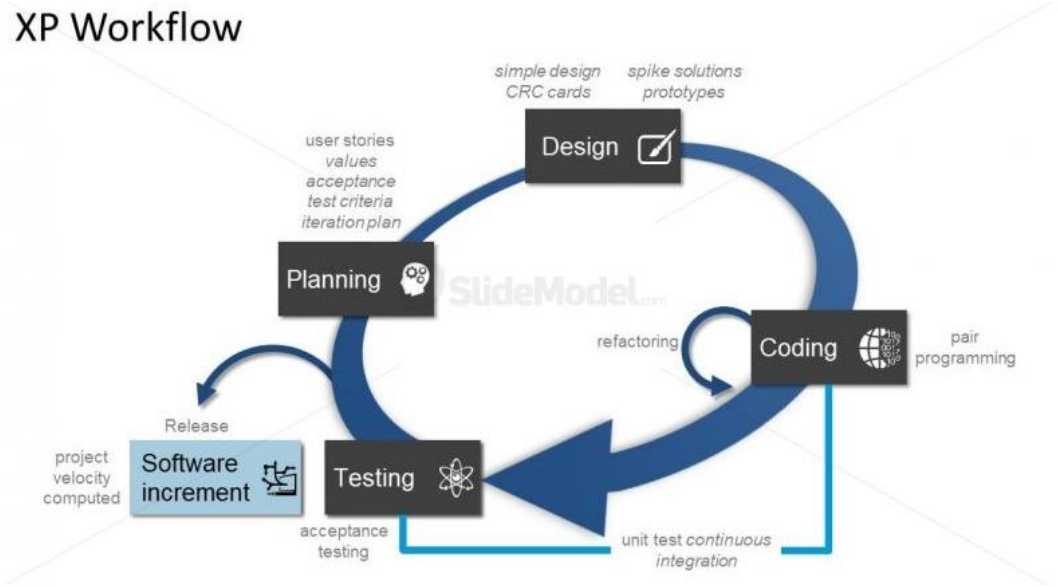


Fig. 3. Gráfico del flujo de trabajo de Extreme Programming.
Fuente: (Narula, 2017)

Justificación

Este proyecto tiene enfoque a los siguientes objetivos de desarrollo sostenible que provienen de los ODM y han sido planteados por la Organización de Naciones Unidas (ONU).

Este proyecto apoya al objetivo ocho “Promover el crecimiento económico sostenido, inclusivo y sostenible, el empleo pleno y productivo y el trabajo decente para todos”(ONU-CEPAL, 2016)

Se apoya a la siguiente meta del objetivo:

Lograr niveles más elevados de productividad económica mediante la diversificación, la modernización tecnológica y la innovación, entre otras cosas centrándose en los sectores de gran valor añadido.

Justificación Teórica. – El presente proyecto hace referencia a la investigación ”Diseño de plataforma tecnológica de medio de pago local para el comercio electrónico en Ecuador, mediante la integración de infraestructuras de servicio tecnológico financiero (FINTEC)” que propone lograr una verdadera integración eficiente y segura de los sistemas FINTEC en el Ecuador(Trejo & Ortega, 2018).

Justificación Tecnológica. – El diseño de un ecosistema de software que facilite la interoperabilidad entre los sistemas de E-Commerce y Courier, requiere del diseño de buenas prácticas en las arquitecturas tecnológicas que hacen posible la interoperabilidad

de los sistemas de transacciones financieras electrónicas y el comercio electrónico en Ecuador.

Justificación Económica. – La interoperabilidad de sistemas de E-Commerce y Courier requiere varios intermediarios que de manera monetaria se benefician indirectamente de este proceso de interoperabilidad, por lo que diseñar un ecosistema que facilite el comercio electrónico beneficiaría no solo a los involucrados directos E-Commerce y Courier, además clientes e industrias tendrían un canal seguro y eficiente para promover sus giros de negocio.

Justificación Metodológica. – El diseño y desarrollo del prototipo para la interoperabilidad entre los sistemas de E-Commerce y sistemas Courier, se desarrollará en un marco de investigación exploratorio pues la seguridad y eficiencia de la interoperabilidad es un tema del cual existe información pero no sé ha profundizado en el contexto ecuatoriano, además el desarrollo de este proyecto buscará información en libros y artículos por lo que se define que según la fuente de datos la metodología será documental, mientras que para la validación y resultados del presente trabajo será necesario realizar pruebas de integración del prototipo diseñado.

CAPÍTULO 1
Marco Teórico

1.1 Interoperabilidad de E-Commerce y Courier

1.1.1 Definición de Interoperabilidad.

La interoperabilidad está definida por la facultad que posee un sistema o producto para adaptarse y cohesionar con otros similares homologando criterios con la finalidad de comunicar distintas dependencias, en el caso específico de la interoperabilidad de sistemas informáticos se define como la "característica esencial para que arquitecturas de información puedan enlazarse para trabajar parámetros embebidos en un entorno heterogéneo"(Yzquierdo & González, 2016), la interoperabilidad permite a varios sistemas informáticos con arquitecturas y construcción de software distintos converger, en el caso específico de los sistemas y tecnologías E-Commerce y Courier esta interoperabilidad suele ser solventada con el desarrollo de Interfaces de programación de aplicaciones(Papazoglou, Jeusfeld, Weigand, & Jarke, 2016) que hacen referencia a servicios web de diferentes tipos.

Para conceptualizar interoperabilidad es necesario definir algunos aspectos que se desarrollan con base en el concepto y llegan a ser relevantes, estos aspectos son:

- Interoperabilidad Semántica

Asegura el significado exacto de la información a intercambiar, es decir información sin ambigüedad y entendible para todas las transacciones de las distintas aplicaciones que intervienen en una transacción de información.

- Interoperabilidad Organizacional

Las estructuras organizacionales que desean intercambiar información se apalancan en la interoperabilidad organizacional para definir sus objetivos de negocio modelando procesos y de esa manera facilitan la colaboración al intercambiar información, por lo general esta interoperabilidad también se encamina a cumplir los requerimientos del usuario.

- Interoperabilidad Técnica

Solventa todos los requerimientos y aspectos técnicos en lo referente al software, hardware y telecomunicaciones que son indispensables para transaccionar e interconectar sistemas informáticos y computacionales.

- Interoperabilidad de Gobierno Electrónico

Establece estándares de interoperabilidad, son acuerdos los cuales son realizados y desarrollados por los actores que participan en el proceso de interoperabilidad.

1.1.2 Arquitectura de la Interoperabilidad.

La arquitectura de la interoperabilidad permite definir la manera en que las aplicaciones serán construidas en lo referente a sus componentes y servicios, además una arquitectura de interoperabilidad proporciona una vista integral de los sistemas informáticos de cómo serán construidos tecnológicamente tomando en cuenta por menores como flexibilidad, escalabilidad y seguridad de las aplicaciones.

En la figura 4, se muestra un ejemplo de arquitectura para el desarrollo de la interoperabilidad apoyado en aspectos fundamentales.

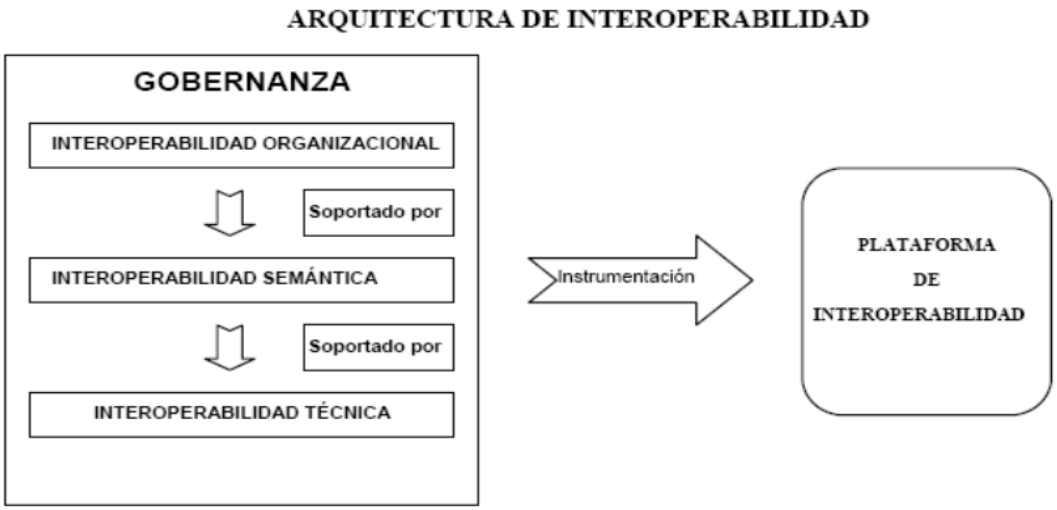


Fig. 4. Ejemplo de arquitectura de interoperabilidad.
Fuente: (Yzquierdo & González, 2016)

En definitiva, la interoperabilidad posibilita a varias organizaciones con distintos giros de negocio interconectarse entre sí, para ello es necesario que estas organizaciones coordinen y llegue a acuerdos en base a la interoperabilidad organizacional enfocada en estándares que posibiliten una adecuada semántica para posibilitar y asegurar la interoperabilidad técnica.

1.2 Sistemas E-Commerce

1.2.1 Definición de E-Commerce

El comercio electrónico denominado E-Commerce se conceptualiza como la manera de realizar una transacción comercial de servicios y bienes en las cuales el proveedor

interacciona de manera electrónica con el consumidor o comprador, los medios de interacción se basan en datos como texto, imágenes y video.

En lo que refiere a las actividades que comprende las transacciones e interacción por compra y venta de bienes y servicios se define, “El E-Commerce comprende actividades muy diversas, como comercio electrónico de bienes y servicios, suministros en línea de contenidos digitales, transferencia electrónica de fondos“(Basantes et al., 2016), es decir el E-Commerce comprende varios campos y entornos en los que se desenvuelve en el campo empresarial, comercial, medios públicos y transacciones financieras que automatizan la oferta de bienes y servicios, mediante sistemas que automatizan y agilizan el E-Commerce.

1.2.2 Tipos de E-Commerce

El E-Commerce se puede categorizar en diferentes tipos según los actores que participan en la relación de comercio mediante medios electrónicos.

- E-Commerce B2B

La categoría de E-Commerce B2B (Business to Business) indica que se establece una relación comercial entre empresas(Romero & Mauricio, 2015) que tienen los medios electrónicos para realizarlo, por lo general estas operaciones se dirigen a vendedores, compradores e intermediarios como minoristas y sistemas de entrega.

- E-Commerce B2C

Esta categoría se denomina B2C (Business to Costumer), es decir entre una empresa y un consumidor, se suele llevar a cabo mediante sitios virtuales, tiendas en línea de rápido acceso y que suelen estar disponibles de manera permanente.

- E-Commerce C2C

El E-Commerce C2C (Costumer to Costumer), se da cuando una persona busca vender un objeto a otra, por lo general sucede mediante medios digitales como redes sociales, también existen plataformas que permiten realizar este tipo de transacciones.

- E-Commerce B2E

E-Commerce B2E (Business to Employe) es el tipo de E-Commerce que sucede cuando existe una relación comercial entre una empresa y sus colaboradores o empleados, por lo general este tipo de E-Commerce existe para incentivar a los trabajadores de una compañía.

- E-Commerce G2C

E-Commerce G2C (Goverment to Costumer), este tipo de comercio se caracteriza por las facilidades que un gobierno permite o realiza para que sus ciudadanos puedan realizar transacciones de trámites y pagos de manera electrónica o digital.

1.2.3 Situación del E-Commerce en Ecuador

En Ecuador el E-Commerce es “considerado entre las empresas como una herramienta estratégica que permite llegar a más personas a un menor costo”(Daniela & Navarro, 2018), se estima además que el país en todo lo relacionado a transacciones digitales y comerciales, a través, de internet ha experimentado un crecimiento importante sobre todo en ciudades como Quito, Guayaquil y Cuenca donde se concentran el 51% de las compras por medios digitales de este estimando se considera que del total de compras virtuales un 31% corresponde a bienes y servicios, 33% corresponde a prendas de vestir, 15% a recreación y cultura y un 20% a otro tipo de rubros adquiridos(CECE, 2018).

En definitiva, la situación del E-Commerce en Ecuador tiende a crecer constantemente gracias al acceso a la internet y dispositivos electrónicos, las empresas ecuatorianas de bienes y servicios ven en el comercio electrónico una oportunidad clara para difundir sus giros de negocio, por lo que se puede inferir que este proceso de crecimiento seguirá con una tendencia positiva.

1.2.4 Legislación de Comercio Electrónico en Ecuador

En Ecuador existe la Ley del Comercio Electrónico, Firmas Digitales y Mensajes de Datos la cual se encuentra vigente desde el año 2002, esta ley intenta proteger a las partes involucradas en el proceso de comercio electrónico, sin embargo, la inseguridad jurídica es un factor limitante para el comercio electrónico debido a la falta de mecanismos para controlar que la adquisición de bienes y servicios por medios digitales no resulten en estafa (Eduardo, 2019).

La legislación de Comercio Electrónico en Ecuador existe, sin embargo, debido al crecimiento constante y exponencial de la tecnología, esta ley puede o se está quedando rezagada por lo que para proteger al consumidor y al proveedor de recursos o servicios en Ecuador sería importante definir reglamentos y protocolos propios en cada organización, independientemente de la legislación Ecuatoriana es necesario que el consumidor se sienta respaldado y protegido al realizar una transacción electrónica eso dará lugar a un crecimiento sostenible del comercio electrónico en el país.

1.2.5 Tecnologías aplicadas en el E-Commerce

La tecnologías aplicadas en el comercio electrónico varían desde algunas que podrían catalogarse como tradicionales como sitios web, difusión por correo electrónico y llamadas de voz, estas tecnologías suelen ser eficientes y bastante efectivas, pero además de ellas existen nuevas tecnologías que probablemente revolucionen el comercio electrónico a escala global, estas nuevas tecnologías principalmente están relacionadas al reconocimiento de comandos de voz implementando inteligencia artificial y el uso de realidad aumentada para mejorar la experiencia previa de compradores en línea(Umbarila Méndez & Romero Rodríguez, 2015) ,permitiéndoles tener una perspectiva más amplia del producto o servicio que están adquiriendo.

1.2.6 Ventajas y Desventajas del E-Commerce

El E-Commerce enfrenta varios desafíos para su desarrollo y control una vez se ha puesto en marcha, al igual que cualquier medio convencional presenta ventajas y desventajas, a continuación(Nana, 2018), a breves rasgos se presenta varias de las ventajas y desventajas más significativas.

Los sistemas E-Commerce presentan varias ventajas, se considera que el alcance es una de la más grandes ventajas que el comercio electrónico puede ofrecer ya que permite a una empresa mostrar su producto o servicio a escala global, desde el punto de vista del consumidor existen varias ventajas que significativamente pueden ser aprovechadas y que en un comercio físico tradicional no están disponibles, entre ellas podemos mencionar que los sistemas E-Commerce se encuentran abiertos y disponibles las 24 horas del día, suelen ofrecer ofertas y descuentos para clientes, permiten una optimización del tiempo en la adquisición de un bien o servicio y ofertan mayor variedad de productos y servicios.

El comercio electrónico a pesar de sus bondades presenta desventajas entre las cuales se distingue el riesgo de adquirir un producto o un servicio que no es exactamente lo que se ha acordado, está desventaja se refleja en un riesgo de autenticidad y seguridad, esta desventaja genera desconfianza en los usuarios y consumidores, otra de las desventajas es el tiempo que se requiere para entregar un producto desde el punto de vista de un vendedor y el tiempo requerido para recibir un producto desde el punto de vista de un comprador, es decir se requiere un sistema de logística y un tiempo significativo en que se complete con satisfacción una transacción.

Es importante mencionar también que en lo relacionado a productos perecibles como alimentos los consumidores suelen preferir establecimientos físicos para asegurarse de la calidad de aquellos productos.

1.3 Sistemas Courier

1.3.1 Definición de Courier

El término Courier se usa para referirse a una persona o empresa que realiza el servicio de llevar correspondencia de paquetes, documentos o cualquier otro tipo de objetos, por lo general la empresas denominadas Courier cuentan con casilleros u oficinas repartidas a lo largo de sus recorridos geográficos(Rocío, 2017), estas oficinas sueles estar habilitadas y tener la capacidad para almacenar mercancías de diferentes tipos.

Los sistemas Courier deben ofrecer garantías a sus usuarios y consumidores dependiendo de la legislación del país en donde realicen sus operaciones, además existen Courier especializados en determinados tipos de mercancías, por ejemplo se pueden identificar sistemas Courier que únicamente realizan el transporte de medicamentos y productos de la rama farmacéutica, así mismo existen Courier que son capaces o están hábiles para realizar el envío de ciertos productos que necesitan ser transportados conservando una cadena fría.

En el auge actual los Courier suelen estar ligados con sistemas de compra y venta de productos en línea, por esa razón estos sistemas buscan la manera de conectarse con medios de comercio electrónico para así realizar una complementación exitosa, en la coyuntura y crecimiento empresarial los Courier con los sitios de Marketplace suelen estar relacionados como complemento uno del otro, es así que los sistemas de Courier buscan estrategias para integrarse a los sitios de Marketplace, permitiéndoles realizar la generación de guías de transporte para el envío de un producto de un punto geográfico hacía otro, la cotización del envío dependiendo del tipo de mercancía, peso volumétrico y distancia que se deberá realizar hasta que el paquete o encomienda llegue a su destino, el seguimiento geográfico con distinción de procedimientos y fechas de las guías de transporte, además los sistemas Courier acostumbran a mantener actualizados y disponibles la ubicación de cada una de sus oficinas y puntos físicos en una región geográfica.

1.3.2 Servicios de Courier en Ecuador

Los servicios que ofrecen los sistemas Courier en Ecuador se pueden distinguir en dos tipos, envíos o recepción de paquetes en el territorio nacional y envíos y recepción de paquetes fuera del territorio nacional, en el caso de que la logística de los servicios Courier sucedan fuera del país se deben considerar para su desarrollo una serie de normas y reglas nacionales e internacionales que deben cumplirse entre ellas tenemos: la ley orgánica de aduanas, el convenio postal universal, el reglamento de la corporación aduanera

ecuatoriana, resoluciones del servicio de rentas internas con todo lo concerniente a rubros e impuestos, en el caso de la logística para sistemas Courier dentro de la nación es importante tomar en cuenta la normativa legal y financiera respecto a los aportes al servicio de rentas internas, mientras que la normativa de aduanas no sería indispensable, ya que no se realiza el paso de mercancías por la frontera.

- Crecimiento de la cantidad de envíos en Ecuador.

En el reporte de comercio que realiza el Banco Central del Ecuador(BCE), publicado en febrero del 2017 se revela que el envío y recepción de productos, a través de empresas Courier aumentó casi en un 4%(Carolina, 2017), lo que supone un crecimiento bastante aceptable, sin embargo, se deberían adoptar estrategias para aumentar ese porcentaje.

El crecimiento de los envíos, a través, de sistemas Courier es aceptable, aunque no revela un gran desarrollo o desenvolvimiento, se estima que a medida que el comercio electrónico crezca en Ecuador existirá mayor demanda de los servicios de paquetería Courier.

- Asociación Ecuatoriana de Empresas de Courier

En Ecuador las empresas Courier han formado una asociación denominada “Asociación Ecuatoriana de Empresas de Mensajería Expresa y Courier” (ASEMEC), esta asociación tiene su sede principal en la ciudad de Quito la cual es la capital ecuatoriana, la visión de esta asociación la de consolidar la unidad de las empresas que se dedican al transporte expreso, para así consolidar la industria en el país(Armando, 2018).

Entre los socios destacados de la ASEMEC se encuentran las siguientes empresas ecuatorianas de Courier:

  - DHL express.
  - Servientrega.
  - In Express.
  - Laarcourier.
  - UPS
  - Grupo Entregas.
  - Siati Group.
  - Transsky.
  - World Courier.

El conocimiento de esta organización para el desarrollo del presente trabajo es fundamental ya que uno de los objetivos persigue la consecución de un marco conceptual y teórico de los sistemas Courier en el Ecuador para así, proponer una alternativa factible y

eficiente para el desarrollo de la interoperabilidad de estos sistemas con similares E-Commerce.

- Servicios de envíos preferidos a nivel nacional.

En Ecuador la Universidad Espíritu Santo UEES su centro de investigaciones con apoyo de la Cámara Ecuatoriana de Comercio Electrónico ha realizado en año 2018 un informe acerca de la industria y el comercio electrónico en Ecuador, en este informe se señalan a las empresas Courier preferidas en el mercado ecuatoriano.

En la figura 5 se muestra los resultados obtenidos en el informe ya mencionado.

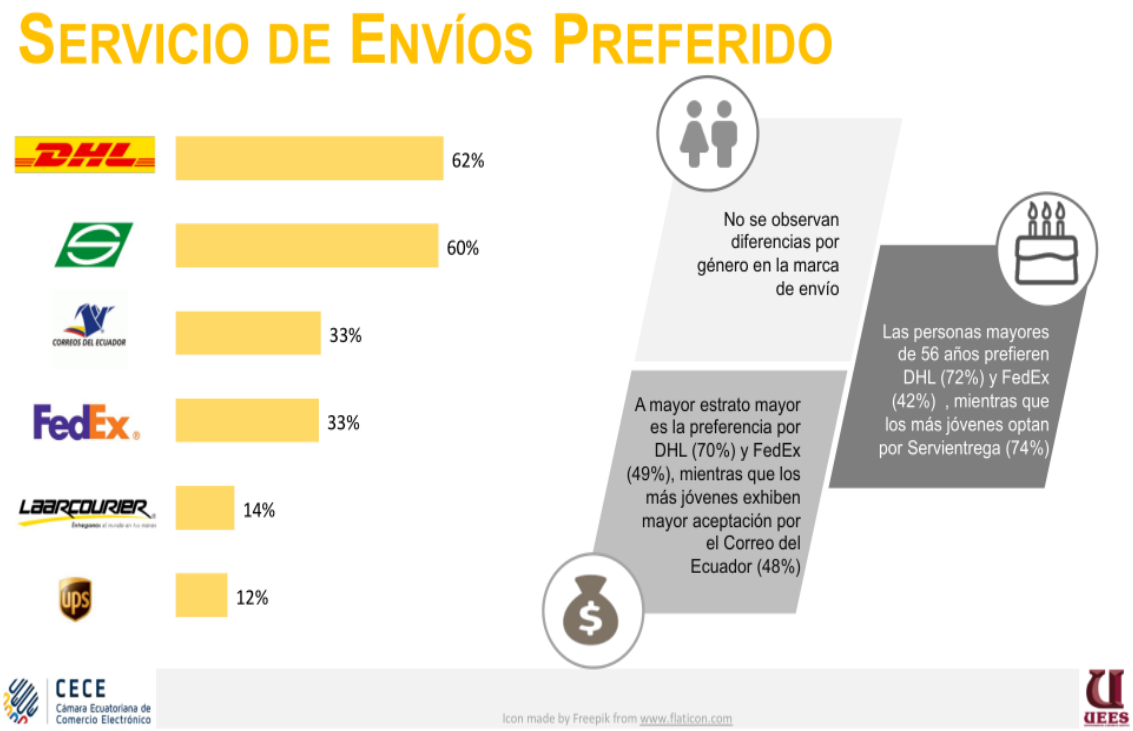


Fig. 5. Servicios de Courier en Ecuador.
Fuente: (CECE, 2018)

Las empresas de Courier preferidas en Ecuador son DHL, Servientrega, Correos del Ecuador, FedEx y Laarcourier, de este grupo de empresas Courier nacionales ecuatorianos se encuentran correos del Ecuador y Laarcourier.

1.3.3 Proceso de operación inbound de un Courier

El proceso inbound hace referencia a los productos de procedencia nacional o internacional que serán entregados dentro del país, la encomienda hasta llegar a su destinatario final pasa por un proceso antes durante y hasta la llegada de la encomienda o paquetería a su destino.

En la figura 6 se muestra un diagrama representativo del proceso conocido como inbound refleja las posibles etapas que suceden cuando un Courier se encarga de la entrega de este tipo de envíos.

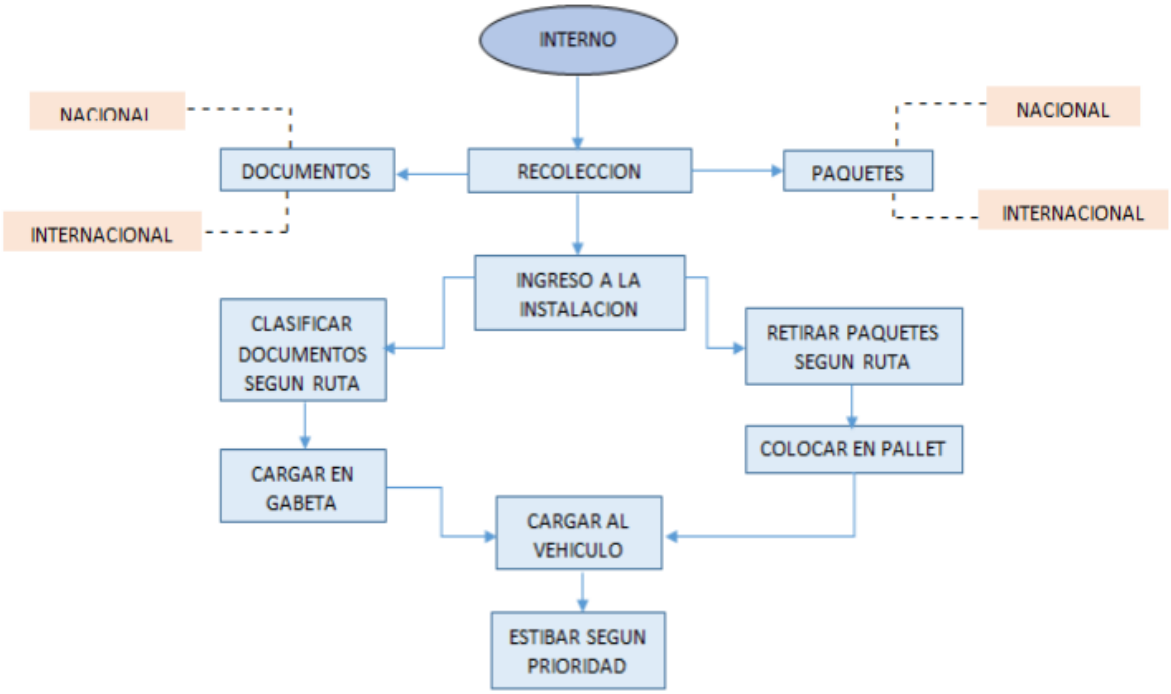


Fig. 6. Proceso Inbound
Fuente: (Hidalgo, 2016)

1.3.4 Ventajas y Desventajas del Courier

Elegir y utilizar el servicio de empresas de Courier para el envío o recepción de paquetería suele traer consigo una serie de ventajas y desventajas, a continuación, se enumeran varias ventajas y desventajas del transporte de paquetería mediante empresas Courier.

- Ventajas:

  - Es posible asegurar la encomienda.
  - Existe leyes y normas que protegen al usuario.
  - Seguimiento de la encomienda, en muchos casos en tiempo real.
  - El costo suele ser en varias ocasiones una ventaja.

- Desventajas:

  - Existe la posibilidad que la encomienda sufra daños en el transcurso del proceso.
  - Existe la posibilidad que la encomienda se pierda.
  - Los tiempos estimados de la entrega suelen variar por factores como el clima entre otras.

1.4 Interoperabilidad E-Logistics

1.4.1 Definición de E-Logistics

E-Logistics se define como la logística del comercio electrónico, toma en cuenta desde la transacción de medios digitales hasta la cadena de suministro que únicamente es posible gracias a un sistema compuesto por múltiples compañías que deben llegar a converger y cohesionar(Francesc Robusta, 2015) para cumplir con los objetivos del comercio electrónico de manera eficiente y efectiva.

El término E-Logistics es fundamental ya que toma en cuenta desde el aprovisionamiento de materias primas, la fabricación de un producto la venta por medio de sistemas E-Commerce de un tipo probablemente retail o minorista hasta la logística y coordinación con los sistemas de entrega para que un producto llegue a las manos de un consumidor o usuario.

E-Logistics toma en cuenta la creación de fases intermedias que se requieren para la integración de logística el uso de pagos electrónicos y las ventas por internet, en ese contexto toma referencia este proceso como una cadena de suministro, en la que cooperan varias empresas que por lo general trabajan entorno a alianzas estratégicas entre proveedores, fabricantes, distribuidores, vendedores mayoristas y minoristas y principalmente estableciendo como actor clave al consumidor. El E-Logistics marca la pauta de la necesidad que tienen las compañías participantes en una cadena de suministro para integrarse de una manera estrecha en los ámbitos de relaciones comerciales sobre todo técnicos y tecnológicos.

1.4.2 Logística para el E-Commerce

La logística en el E-Commerce resulta ser la piedra angular del comercio electrónico, ya que permite la movilización desde un almacén u origen hasta un lugar de destino un producto, es por esa razón que cualquier empresa exitosa en el campo del E-Commerce debe definir su logística de manera pragmática y efectiva(Valenzuela, 2018), esta logística debe considera dese la recepción hasta la logística inversa.

A continuación, se detalla los seis pasos fundamentales que son cumplidos el proceso logístico antes mencionado.

- Recepción del Producto

Consiste en la recepción directa del producto desde el proveedor, se realiza una verificación de la calidad y detalles técnicos del tipo de paquetería a ser enviada.

- Almacenamiento del Producto

En las ocasiones que un producto requiera almacenamiento es necesario realizar el ingreso de este a un inventario de artículos, en este punto se comprueba que no exista algún tipo de pormenor o defecto con la encomienda, producto o mercadería a ser enviada.

- Gestión de pedidos

Se gestionan los pedidos al recibir la solicitud del cliente el cual específica las características del cliente y recibe una retroalimentación por parte de la empresa de Courier, además en esta etapa se determina el método de pago y un tiempo de entrega.

- Envío del Producto

La empresa de courier determina la modalidad de envío más adecuada dependiendo del producto, además el cliente puede asegurar su pedido si lo considera necesario, en este proceso es necesario e indispensable tener en cuenta factores como pago de impuestos y legislación.

- Pago contra entrega

En este procedimiento el cliente cancela el rubro por la entrega en efectivo, además sería posible receptar el pago con otros medios como tarjeta de crédito al momento de realizar la entrega.

- Logística Inversa

Si por alguna razón el cliente decide devolver el producto, realizando un proceso de logística inversa el Courier retorna el producto al proveedor.

En la figura 7 se muestra un resumen sencillo y expedito de las fases anteriormente mencionadas.

Fig. 7. Proceso de Logística para el E-Commerce
Fuente: (Martín, 2015)

1.5 Análisis de accesibilidad a varios Courier del Ecuador.

En el contexto de la realización del presente proyecto se buscó la posibilidad de establecer conexión con un sistema de Courier, es decir se realizó las consultas a varios sistemas de Courier de la ASEMEC para acceder a su tecnología tal y como lo haría un sistema de E-Commerce interesado en acceder a su logística y hacer la generación de guías de transporte, la cotización de envío de paquetes.

Se encontraron varios requerimientos, por ello como aporte al marco teórico se analiza y describe de manera cualitativa cada una de las opciones de Courier y sus requerimientos, además de las opciones que ofrecen para conectarse con sus tecnologías.

TABLA 2
ACCESIBILIDAD A VARIOS SISTEMAS COURIER DEL ECUADOR

Courier	Requerimientos
UPS	• Facilita del proceso de conexión a su tecnología. • El proceso es entendible y claro. • Tecnología de conexión Rest. • Costo por la accesibilidad.
DHL	• Facilita del proceso de conexión a su tecnología. • El proceso es entendible y claro. • Tecnología de conexión Rest. • No existe Costo por la accesibilidad. • Permite tener acceso como desarrollador. • Aparentemente Seguro.
Servientrega	• No existe información clara del proceso de conexión. • Se requiere contacto directo para establecer acceso. • Tecnología de conexión SOAP. • Se desconoce costo por la accesibilidad. • Parámetros de seguridad desconocidos. • Contacto con baja respuesta.
Tramaco Express	• No existe información clara del proceso de conexión. • Se requiere contacto directo para establecer acceso. • Tecnología de conexión desconocida. • Se desconoce costo por la accesibilidad. • Parámetros de seguridad desconocidos. • Contacto con alta respuesta. • Facturación mensual requerida.
Laar Courier	• No existe información clara del proceso de conexión. • Se requiere contacto directo para establecer acceso. • Tecnología de conexión Rest. • Se desconoce costo por la accesibilidad. • Parámetros de seguridad bajos. • Contacto con regular respuesta. • Facturación mensual requerida.

Fuente:(MDN, 2019) Propia

Los datos recogidos en la tabla anteriormente expuesta, fueron resultado de una recolección empírica que demostró los requerimientos al intentar y pedir información para conectarse con varios sistemas de Courier del Ecuador, se menciona con especial énfasis a Laar Courier, que es el cuarto Courier preferido en el país como se evidencia en la figura 5, este sistema dispone de APIs Rest, pide se asegure una facturación mensual y tiene una respuesta regular al intentar contactarse con ellos para realizar el proceso de conexión, sin embargo, es una de las opciones que tendría mayor interés en conectarse, pues es uno de los servicios preferidos en el Ecuador.

1.6 Ecosistemas de Software

1.6.1 Definición de Ecosistema de Software

Conceptualmente “Un ecosistema de software es un espacio de trabajo en el que conviven una serie de herramientas que acompañadas de la implementación de buenas prácticas, darán cabida a aun equipo de desarrollo realizar una metodología de trabajo”(Bravo, 2017)

En la figura 8 se muestra de manera gráfica los componentes que podría tener un ecosistema de software, se puede observar además que en la figura se toma en cuenta factores como la integración continua y la calidad del código.

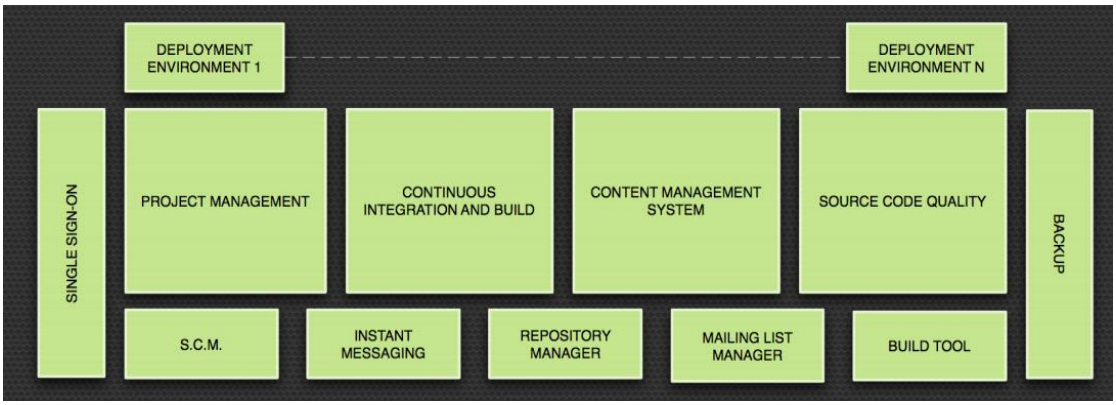


Fig. 8. Representación gráfica Ecosistema de Software
Fuente:(Bravo, 2017)

La conceptualización de ecosistema de software básicamente predica la coexistencia de componente de software enmarcados en buenas prácticas, es importante señalar la característica de integración continua, ya que en base a este tipo de integración se podría definir una metodología de trabajo para un grupo de programadores, además de puntos angulares como el mantenimiento del ecosistema de software y la escalabilidad.

1.6.2 Web Service

Los web service se definen como un canal o vía de interoperabilidad e intercomunicación entre varios puntos de red o máquinas, en la actualidad han tomado relevancia conjuntamente con las arquitecturas orientadas a servicios SOA(Baquero & Blanch, 2017), por lo general este tipo de intercomunicación guarda relación con la arquitectura cliente-servidor, esta comunicación suele estar soportado por el protocolo http, las respuestas de información suelen definirse en formato XML o JSON.

Existen varias tecnologías web service, se realiza una breve explicación de cada uno de ellos en los siguientes puntos:

- SOAP

Este protocolo usa XML como lenguaje para intercambiar data, la estructura de este protocolo suele ser compleja, sin embargo, fue altamente adoptada y en la actualidad todavía existen servicios de comunicación, a través de este servicio.

- REST

Este tipo de web service usa el protocolo HTTP para la comunicación entre puntos de red, REST es caracterizado por no tener estado, utiliza el lenguaje JSON y tiende a ser más sencillo de implementar, además la implementación de autenticación suele ser práctica.

- GraphQL

GraphQL es fuertemente tipado, se caracteriza por proporcionar una sintaxis flexible, además cuenta con varias bondades como reducir la sobrecarga de la transparencia de datos, es decir reduce el número de consultas separadas(Sayago & Flores, 2019) reduce la cantidad de datos innecesarios transferidas.

1.6.3 Métodos de Petición HTTP

Los métodos de petición http indican una sintaxis específica dependiendo de la acción que se realizará, a continuación, se muestra una tabla con las especificaciones de varios de estos métodos.

TABLA 3
MÉTODOS DE PETICIÓN HTTP

Método de Petición	Acción que realiza
GET	Solicita una representación de un recurso.
HEAD	Es idéntica a GET, pero sin el cuerpo de la respuesta.
POST	Envía una entidad a un recurso en específico.
PUT	Reemplaza las representaciones actuales de un recurso.

DELETE	Elimina un recurso específico.
CONNECT	Define un túnel hacia el servidor identificado por el recurso.
OPTIONS	Describe las opciones de comunicación para el recurso de destino.
TRACE	Realiza una prueba de bucle de retorno de mensaje.
PATCH	Aplica modificaciones parciales a un recurso.

Fuente:(MDN, 2019)

1.6.4 Arquitectura REST

REST o Representational State Transfer, se define como una arquitectura de software la cual cumple varios principios, en base a esos principios se diseñan servicios web(Álvarez, 2016), que en base al protocolo HTTP se comparte información entre distintos componentes o aplicaciones de software.

La arquitectura REST está constituida por 6 constraints los cuales se deben cumplir al realizar un API RESTful real, a continuación, se enumeran y se hace una breve explicación.

- Interfaz uniforme

Un recurso individual debe tener el tamaño adecuado, además deben contener enlaces HATEOAS si se considera relevante obtener información relacionada.

- Servidor de cliente

El cliente y el servidor deben evolucionar de manera independiente, el punto clave es que el recurso mediante el cual se comunican se mantenga estable.

- Sin estado

Cada solicitud se trata como una nueva, es decir el servidor no almacenará ningún contexto para tratar al cliente.

- Caché

El almacenar datos en caché hace que las respuestas se agilicen, esta data en caché puede almacenarse del lado del cliente o del servidor, mejora la escalabilidad y el rendimiento.

- Sistema de capas

La arquitectura REST al implementar un sistema de capas, indiferentemente de donde se encuentre el servidor cliente, servidor y la autenticación en diferentes dependencias conservará su eficiencia.

- Código bajo demanda

Este constraint es especial, en algunas ocasiones se puede retornar código ejecutable, que podrá ser admitido por una parte de la aplicación.

1.6.5 Definición de API

Se define como API a una interfaz de programación de aplicaciones, estas interfaces son un conjunto de reglas y varias especificaciones que las aplicaciones son capaces de establecer para comunicarse entre ellas(Marcos, 2016), además se puede ver llegar a ver a las API como la subcontratación de funciones debido que, a través de una API se puede obtener respuesta de una aplicación sin conocer realmente sus entrañas.

En la figura 9 se puede observar de manera gráfica la comunicación que puede existir entre diferentes aplicaciones mediante APIs.

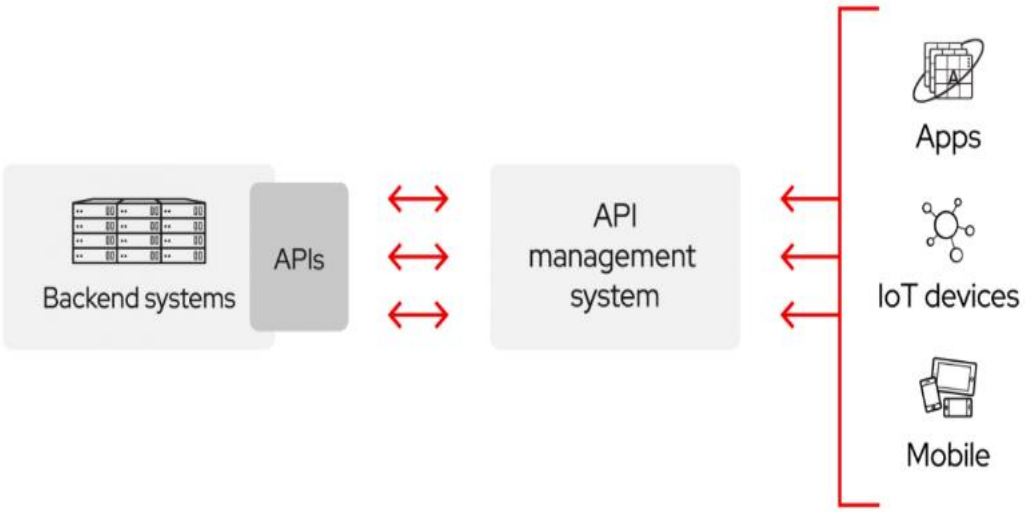


Fig. 9. Representación Interoperabilidad con APIs
Fuente:(Red Hat, 2017)

1.6.6 Tests y performance APIs RESTful

El realizar pruebas o tests para verificar y entender el performance o puesta en marcha un APIs RESTful probar una colección de peticiones a la API con el objetivo de verificar si se cumple con los objetivos de confiabilidad, funcionalidad, rendimiento y seguridad(Kotecha, 2018), además de asegurar que se obtiene la respuesta esperada de las API.

Para realizar los tests a un API se usan softwares capaces de realizar peticiones http a un servidor, estos softwares facilitan el control de calidad de un API, ya que, realizándolo directamente con una aplicación construida, suele existir un retraso entre el back-end y el front-end, el software se encarga de alinear esos factores y obtener respuestas fiables, resales y objetivas.

1.6.7 JSON Web Tokens JWT

JWT es un bloque JSON el cual está firmado, es decir gracias a esa firma se puede verificar la autenticidad, JWT es usada para realizar autenticación y proveer seguridad a los servicios web que son ofrecidos como APIs(Olivera, 2017).

El uso de JWT para la autorización y autenticación suele seguir un flujo esencial, se describe este flujo a continuación.

1. Un usuario solicita uno de los recursos del servidor, se presenta la necesidad autenticarse.
2. El usuario solicita un token o firma de autenticación al servidor.
3. El usuario es asignado un token para realizar login.
4. El usuario usa las credenciales proveídas por el servidor.
5. El servidor verifica las credenciales del usuario.
6. El usuario obtiene el recurso solicitado.

Se realiza la representación de este flujo en la figura 10, mostrada a continuación.

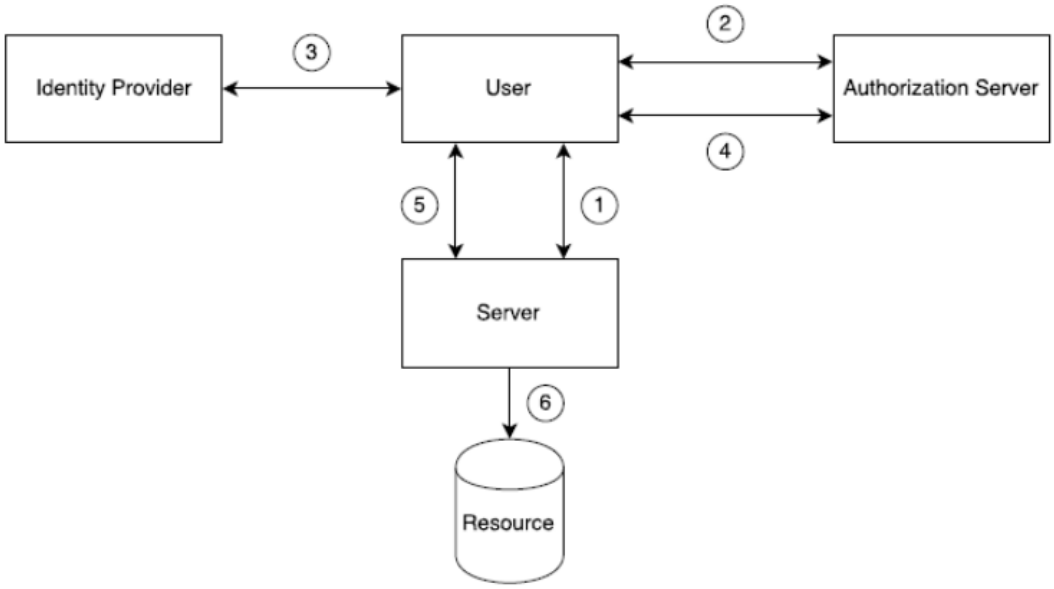


Fig. 10. Flujo JWT para la autenticación
Fuente:(Olivera, 2017)

1.6.8 Swagger documentación servicios RESTful

Swagger es un framework o marco de trabajo para consumir, documentar y visualizar servicios REST, uno de los objetivos primordiales de Swagger es que la documentación del API se vaya actualizando constantemente a la par de los cambios(Crespo, 2017).

Este marco de trabajo ofrece una GUI o interfaz visual para realizar peticiones http al API, además se puede verificar su documentación de manera práctica, en definitiva, facilita la exposición del contrato del API.

1.6.9 .Net Core

.Net Core es un framework para el desarrollo de software, este marco de trabajo se de código abierto además provee al desarrollo alto rendimiento(Microsoft, 2019), se enumeran algunas de las características principales de este framework.

- Multiplataforma: Se ejecuta en prácticamente todos los sistemas operativos.
- Coherente entre arquitecturas: El código es capaz de ejecutarse con igual comportamiento en varias arquitecturas.
- Posee línea de comandos: Incluye herramientas de línea de comandos, fundamental para varios escenarios ye integración continua.
- Implementaciones flexibles: Se puede realizar su implementación de forma paralela a cualquier sistema, además es posible usarlo en contenedores Docker.

.Net Core es un framework altamente eficiente y efectivo, tiene una alta aceptación entre los desarrolladores como lo demuestran las encuestas realizadas por Stack Overflow, en la figura 11 se muestran resultados de una encuesta realizado respecto a la preferencia de marcos de trabajo según los desarrolladores.

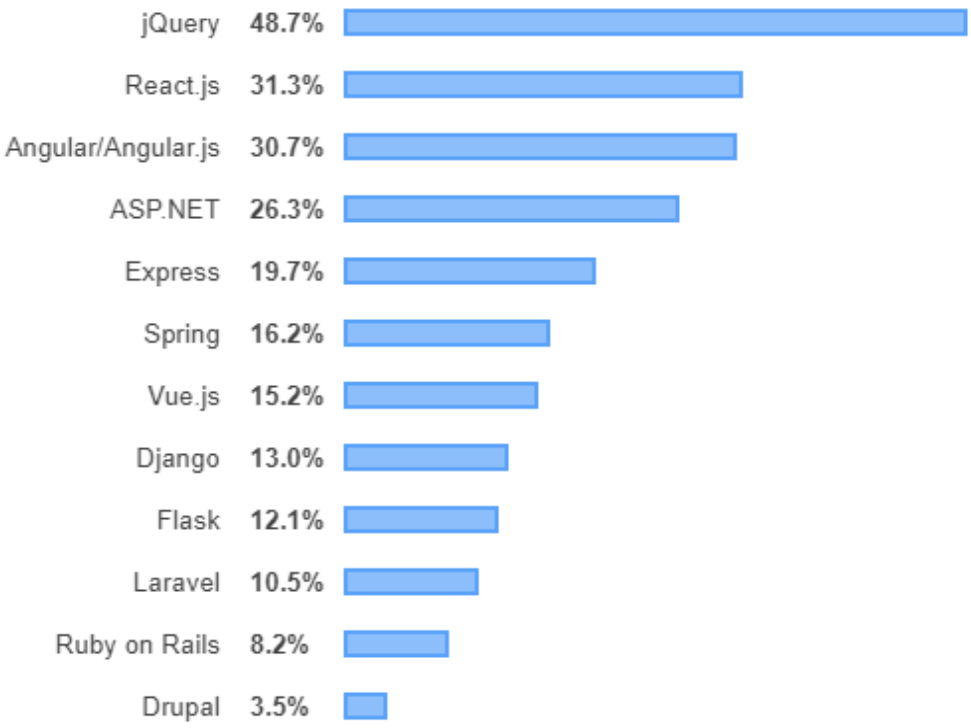


Fig. 11. Estadística de preferencia framewoeks según Stack Over Flow
Fuente:(Stack Overflow, 2019)

1.6.10 SQL Server

SQL Server está en la categoría de sistema de gestión de bases de datos relaciones, este sistema es propiedad de Microsoft, está pensado para un entorno empresarial(Miguel, 2019), este sistema de gestión de bases de datos cuenta con varias características destacables, se mencionan varias a continuación.

- Escalabilidad, seguridad y estabilidad.
- Soporte de transacciones SQL.
- Cuenta con un entorno gráfico.
- Permite administrar información de varios servidores de datos.
- Es posible realizar transacciones DDL y DML gráficamente.
- Posibilita el trabajo Cliente Servidor.

Este sistema de base de datos debido a su estabilidad, disponibilidad es altamente aceptado por los desarrolladores del ámbito empresarial, en la figura 12 se muestra estadísticas de aceptación según Stack Over Flow.

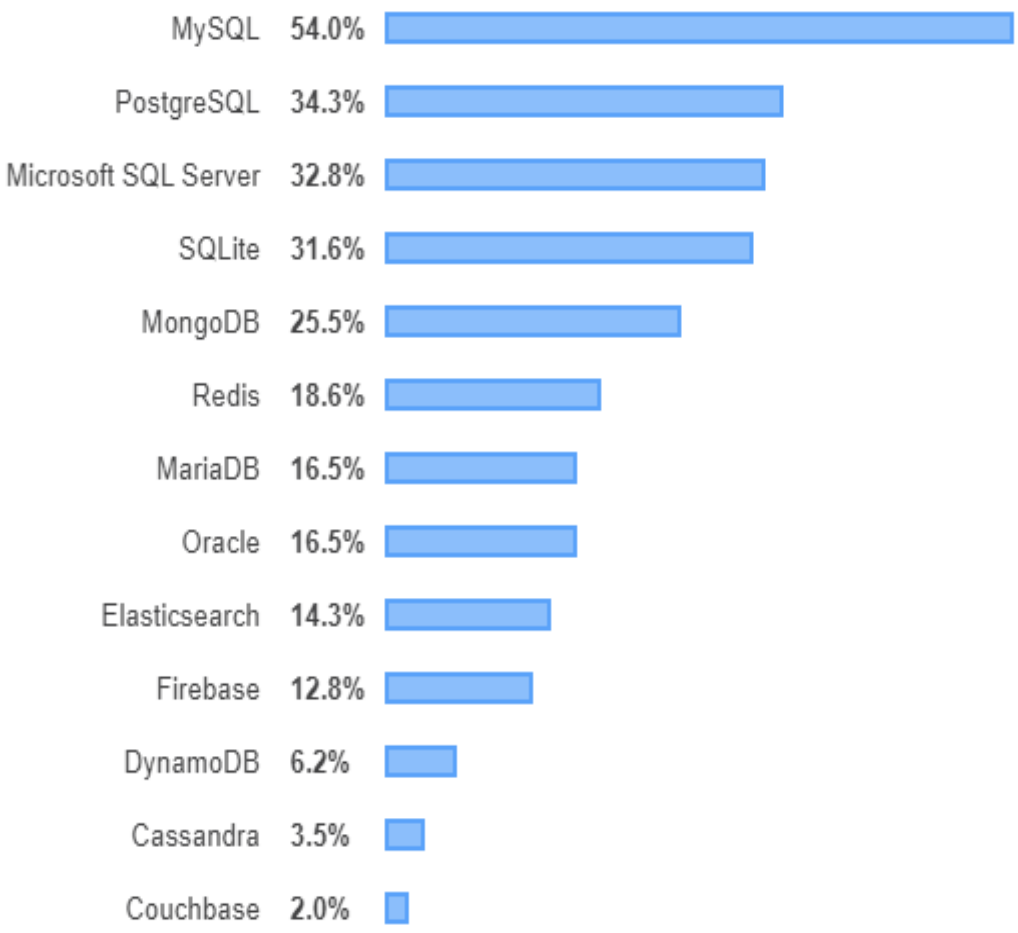


Fig. 12.. Estadísticas de preferencia de motores de base según Stack Over Flow
Fuente:(Stack Overflow, 2019)

1.7 Metodología Ágil Extreme Programming (XP)

1.7.1 Definición de Extreme Programming

Extreme Programming(XP) es una metodología ágil para el desarrollo y construcción de software, XP surgió como una metodología que centra sus bases en la simplicidad y agilidad, las metodologías de desarrollo tradicionales frente a XP suelen tornarse inefectivas y pesadas de llevar para un equipo de desarrollo(Joskowicz, 2018), además es importante mencionar que XP enfoca sus esfuerzos en cumplir los requerimientos del cliente de manera efectiva con vista clara en cuatro aspectos fundamentales de cualquier proyecto de software que son: costo, tiempo, calidad y alcance.

Extreme Programming sigue los manifiestos de las metodologías, a continuación, se enumeran estos cuatro principios.

1. Se valora a los individuos y las interacciones sobre los procesos y herramientas.
2. La documentación exhaustiva es subvalorada con respecto a las aplicaciones que funcionan.
3. La colaboración del cliente en negociaciones contractuales es sumamente valorada.
4. El seguimiento de un plan se subvalora con respecto a la factibilidad del cambio.

1.7.2 Objetivos que persigue Extreme Programming

El objetivo de todo desarrollo de software es entregar un producto de calidad, eficiente y sobre todo funcional, encuadrada en esa visión XP posee varios objetivos claros para conseguir un resultado integral favorable, algunos de estos se mencionan a continuación.

- Definir al cliente y sus requisitos de manera efectiva.
- Decidir las mejores prácticas de Ingeniería de Software de acuerdo con el proyecto específico.
- Maximizar la productividad de los desarrolladores.
- Motivar al equipo de desarrollo inculcar habilidades como una alta capacidad de aprender.

Extreme Programming más allá de los objetivos anteriormente mencionados, intenta desarrollar en el equipo a cargo del desarrollo varios valores como: simplicidad y practicidad, comunicación efectiva, realimentación y retroalimentación del desarrollo y coraje para cumplir con el objetivo de desarrollo trazado(Murilo Carneiro, Beltrán Castañón, Carneiro, & Romani, 2016).

1.7.3 Roles de la Metodología Extreme Programming

Extremme Prgramming cuenta para su desenvolvimiento con varios roles según Latelier Patricion en su trabajo denominado “Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP)“ (Letelier & Penadés, 2015), seguidamente se detalla en breves rasgos cada uno de estos roles.

- Programador

Es alguien que produce el código del sistema y escribe pruebas unitarias.

- Cliente

Realiza las historias de usuario y pruebas funcionales, con la finalidad de comprobar su validez en la implementación.

- Tester

Asesora al cliente en la redacción de pruebas funcionales, además es quien regularmente ejecuta las pruebas y expone los resultados al equipo.

- Tracker

Proporciona realimentación al equipo en el proceso de desarrollo ágil XP, estima el grado de acierto en las estimaciones realizadas y el tiempo real que se le ha dedicado al proyecto.

- Entrenador

El encargado y responsable del proceso en ambiente global, conoce a profundidad el proceso XP y provee al equipo las guías de la metodología ágil.

- Consultor

Posee un conocimiento específico en algún tópico fundamental para el proyecto.

- Gestor

Se encarga de vigilar que el equipo trabaje en condiciones adecuadas, además es el vínculo entre los programadores y el cliente.

1.7.4 Etapas y ciclo de Extreme Programming

Extreme Programming se desarrolla en ciclos y para su desenvolvimiento presenta varias etapas, en la figura 3 se muestran estas etapas en un gráfico, a continuación se hace una

breve referencia a cada una de ellas tomando como referencia a Joskowicz José en su trabajo “Reglas y Prácticas en eXtremme Programming”(Joskowicz, 2018).

- Exploración

Se exploran las diferentes posibilidades de arquitecturas para el proyecto, el cliente realiza a breve rasgos el panorama del proyecto y realiza historias de usuario, mientras los desarrolladores se familiarizan con las tecnologías y buenas prácticas necesarias.

Planificación de la entrega

El cliente establece la prioridad de cada historia de usuario, en base a esa prioridad se estima el tiempo y esfuerzo necesario para cumplir con cada una de las historias.

- Iteraciones

El plan de entrega se compone por iteraciones que suelen variar entre dos y tres semanas, por lo general en la primera iteración se establece la arquitectura del proyecto.

- Producción

Se trata de llevar el desarrollo al entorno del cliente, se suelen realizar varias pruebas para verificar la calidad de la pieza de software antes de realizarlo.

- Mantenimiento

Una vez se ha implementado la primera versión del proyecto en producción se realizan nuevas iteraciones incluyendo algunos cambios y reformas pedidas por el cliente estas reformas suelen ser incluidas para dar soporte al cliente.

- Muerte del Proyecto

Se concluye el proyecto una vez cumplidas las historias de usuario realizadas por el cliente, se realiza la documentación final del proyecto, esta etapa implica que la arquitectura del proyecto no tendrá más cambios.

1.7.5 Ventajas de programación en pares XP.

Extremme Programming promueve la programación en pares, es decir dos programadores codificando al tiempo en un mismo computador, se podría inferir que esto tendría un coste de tiempo desfavorable, pero se ha comprobado que el coste no supera un 15% en tiempo y que se mejora en gran medida la calidad del software(Joskowicz, 2018).

Se mencionan varias ventajas de la programación en pares:

- Se descubren de codificación al momento con gran eficacia.
- Estadísticamente los defectos que se llegan a encontrar en pruebas es menor.
- El código producido suele estar optimizado.
- El número de personas que conocen los pormenores de las implicaciones en el desarrollo aumenta.

1.8 Norma ISO 8583

1.8.1 Definición de la Norma ISO 8583

El estándar internacional ISO 8583 se aplica para los mensajes intercambiados los cuales suelen estar originados en tarjetas de transacciones financiares, cajeros automáticos ATM, y puntos de venta POS(ADMFactory, 2017).

La ISO 8583 precisa un flujo de comunicación y un formato de mensaje para que los sistemas puedan intercambiar respuestas y solicitudes de transacciones, por lo general estas solicitudes suelen ser generadas desde una tarjeta de crédito por el titular de esta, este estándar es utilizado y adaptado a cada red de pagos electrónicos algunos de los elementos de datos transmitidos suelen ser personalizados.

1.8.2 Estructura del mensaje ISO 8583

Un mensaje ISO8583 se compone de 3 partes, la ubicación de estos campos o partes suelen variar dependiendo de la versión de la norma, la versión de la norma que es mayormente usada es la ISO8583:1987.

A continuación, se explica cada una de las partes que componen la estructura ISO 8583:

- Indicador del tipo de mensaje

El indicador del tipo de mensaje es un campo que se compone de cuatro dígitos que indican la función que realiza el mensaje, el indicador de tipo de mensaje incluye la versión de ISO8583, el origen del mensaje y su función.

- Mapa de Bits

Un mapa de bits se constituye en un subcampo dentro del mensaje, este campo tiene la función de indicar que elementos de datos están presentes en la estructura del mensaje, un mensaje de datos debe contener por lo menos un mapa de bits.

- Elementos de datos.

Los elementos de datos son campos individuales, son fundamentales ya que llevan la información de la transacción, hay hasta 128 elementos de datos en la versión del estándar ISO8583:1987, cada elemento de datos tiene un formato y significado específico.

Ejemplo de un mensaje que se ha estructurado en base a la norma ISO8583:1987:

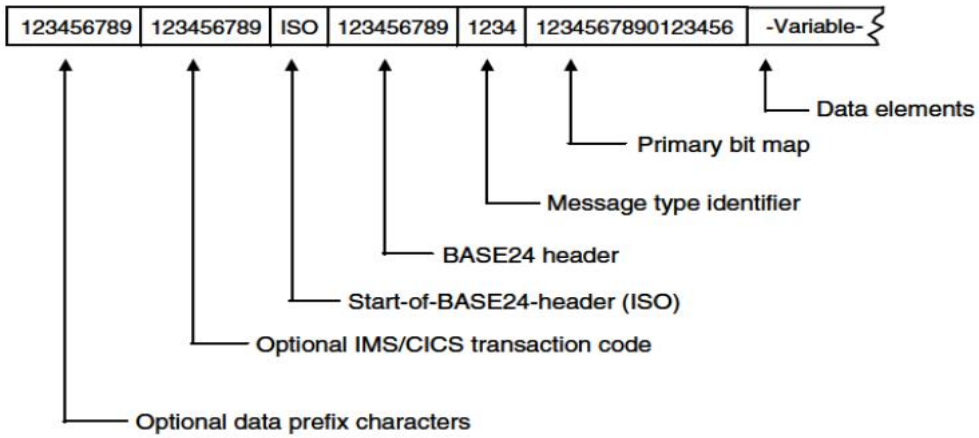


Fig. 13. Mensaje estructurado en base a la ISO8583
Fuente: (Jimenez, 2014)

1.9 ISO/IEC 25010

1.9.1 Definición ISO/IEC 25010

La norma ISO/IEC 25010 define varios parámetros de la calidad del software, entendiendo por calidad el nivel o grado en el que un producto de software satisface las necesidades de rendimiento, funcionalidad, mantenibilidad y seguridad(ISO 25010, 2019).

Fig. 14. Características de la ISO/IEC 25010
Fuente: (ISO 25010, 2019)

1.9.2 Eficiencia en el desempeño

La ISO/IEC 25010 contempla en una de sus características la eficiencia en el desempeño, es decir el rendimiento de la aplicación a continuación se define brevemente cada una de las subcaracterísticas:

- Comportamiento del tiempo: Nivel de respuesta y procesamiento en que los tiempos cumplen con los requisitos.
- Utilización de recursos: Nivel en que los recursos en sus diferentes tipos y cantidades al cumplir sus funcionalidades satisfacen los requisitos.
- Capacidad: Límites máximos de un parámetros o producto del sistema cumple con los requisitos.

1.10 The Open Web Application Security Project (OWASP)

OWASP se define como un proyecto abierto de seguridad de aplicaciones web, respaldado por una comunidad libre que está dedicada a permitir que mediante directrices las organizaciones desarrollen, adquieran, mantengan aplicaciones e interfaces de aplicación de programación segura(OWASP, 2017).

OWASP realiza recomendaciones en los siguientes aspectos:

- Inyección SQL
- Perdida de autenticación
- Exposición de datos sensibles
- Entidades Externas
- Pérdida de Control
- Configuración de control de acceso
- Configuración de seguridad incorrecta
- Cross-Site Scripting
- Deserialización Insegura
- Uso de Componentes con vulnerabilidades conocidas.

CAPÍTULO 2
Desarrollo

2.1 Planificación del Ecosistema de Software

El ecosistema de software se lo construye teniendo como especial objetivo la realización de web service eficientes, entre las tareas más fundamentales para la construcción de este ecosistema está la simulación las APIs de un Courier y simulación de APIs que permitan converger ese sistema Courier con un E-Commerce para ello fue necesario definir una arquitectura que permita la realización de los endpoints necesarios, acorde a los lineamientos y requerimientos levantados, en el presente capitulo se detallará todo lo concerniente al desarrollo realizado.

El ecosistema de software constara de tres sistemas los que se detallan en la tabla 3:

TABLA 4
SISTEMAS DEL ECOSISTEMA DE SOFTWARE

Sistema	Descripción
Ecommerce Admin	Sistema web api Restful que implementa seguridad jwt, trabaja sobre una base de datos Sql Server, está implementado en un hosting que soporta la tecnología .Net Core sus principales funciones a breves rasgos son: • Conocer compras aceptadas. • Registrar envíos por parte de un vendedor. • Registrar conformidad por parte de un comprador.
Courier	Sistema web api Restful que implementa seguridad jwt y trabajará sobre una base de datos Sql Server, está implementado en el cloud de Azure será construido con el framework .Net Core y cumple las siguientes funciones: • Generar un código de guía de transporte. • Realizar operaciones de tracking.
Aplicación Móvil	Diferencia a los actores del ecosistema de software, permite reconocer sus roles, además es la encargada de cohesionar el sistema Ecommerce Admin con el sistema Courier consumiendo sus API.

Fuente: Propia

2.1.1 Metodología de desarrollo

Para el desarrollo de los diferentes ecosistemas de software se implementó la metodología XP Xtreme Programming a continuación se da a conocer los detalles de cada una de sus etapas, cabe considerar que para la aplicación de la metodología se adaptó algunas de sus características, acordé a las posibilidades del proyecto.

- Roles del desarrollo:

Se definen los roles de la metodología que se consideran fundamentales para lo consecución de los objetivos del proyecto, se evidencia en la tabla 4.

TABLA 5
ROLES METODOLOGÍA EXTREMME PROGRAMMING

Rol	Integrante del Equipo
Gestor	Ing. MSc. Diego Xavier Trejo España
Consultor	
Entrenador	
Programador	Jorge Arturo Castro Querembás
Tester	

Fuente: Propia

2.1.2 Levantamiento de Requerimientos

El levantamiento de requerimientos se realizó en varias reuniones de trabajo, se conoció de manera holística el proyecto, como puntos importantes se mencionan los siguientes.

- El ecosistema de software a diseñar es parte de un macro ecosistema de software.
- El Ecommerce Admin para funcionamiento implementa varios módulos en el presente alcance se desarrollará uno de ellos.
- Será necesario simular el sistema de Courier teniendo como referencia a alguno de los pertenecientes a la ASEMEC.
- La aplicación móvil deberá ser capaz de consumir las Api Restful de los sistemas simulados.
- Las pruebas de software se realizarán para verificar la eficiencia de las API, se realizará específicamente pruebas de estrés.
- Se hace énfasis en aspectos de seguridad como la implementación de jwt y https en el medio de producción del Ecommerce Admin.

2.1.3 Flujograma de Actividades del Ecosistema de Software

En la consecución de los requerimientos del software se realizó un flujograma del proceso a implementar, esto con la finalidad de comprender como cada uno de los sistemas partes del ecosistema de software cohesionaban e interactuaban entre sí.

En la figura 15 se puede revisar el flujograma de resultados, que además de reflejar el proceso que el ecosistema de software realiza, permite la realización de las historias de usuario y la realización del modelo físico de la base de datos.

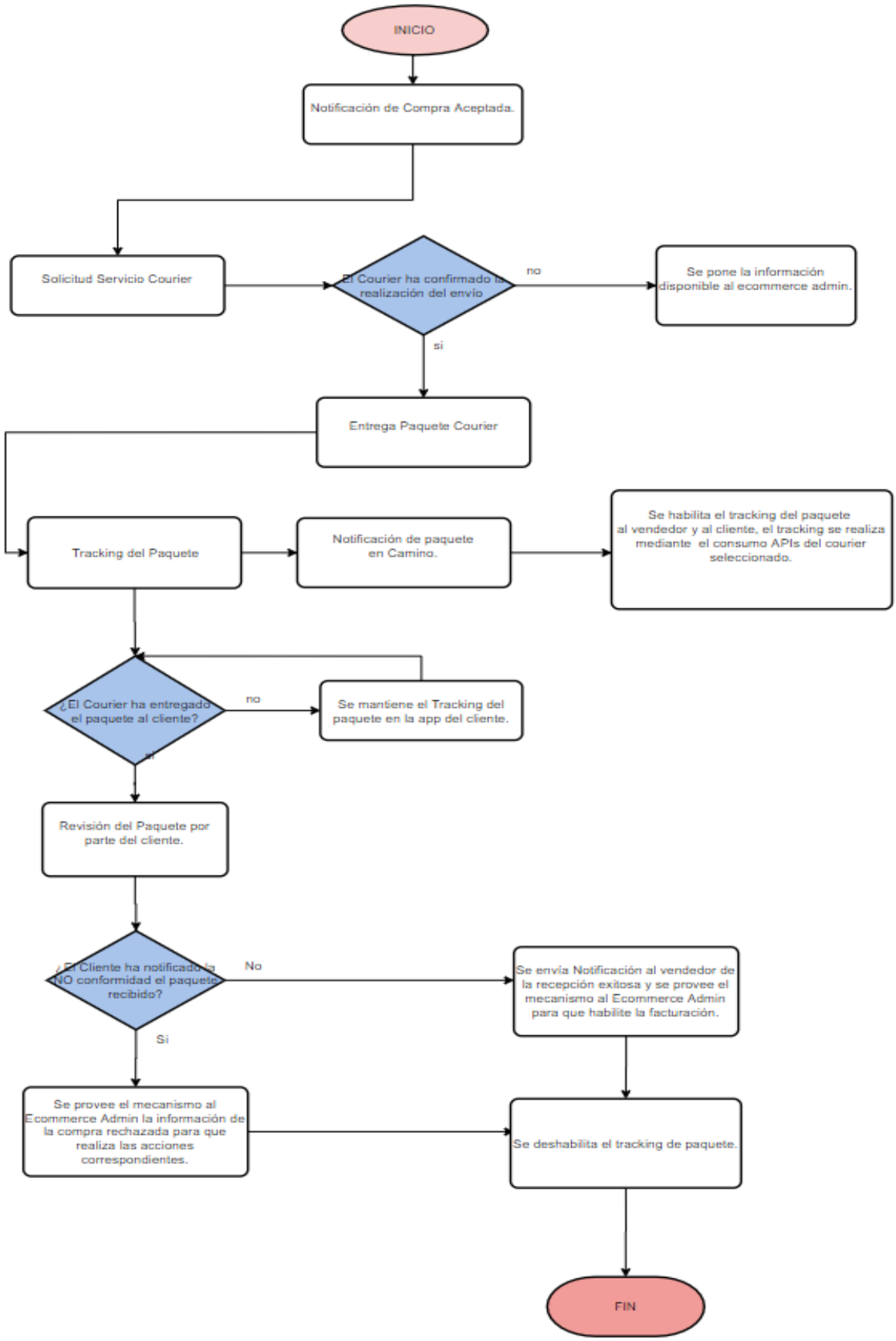


Fig. 15 Flujograma del Ecosistema de Software
Fuente: Propia

2.1.4 Historias de usuario

Se detalla cada una de las historias de usuario, en cada una se específica el sistema para el que pertenecen, una estimación en horas, descripción y grado de prioridad.

TABLA 6
HISTORIA DE USUARIO JSON WEB TOKEN

Historia de Usuario		
Código: REFE-001	**Sistema:** Ecommerce Admin	
Nombre: Implementar JWT Ecommerce Admin		
Prioridad: Alto	**Riesgo:** Alto	**Estimación:** 5
Descripción: El web service Api Restful del ecommerce admin deberá implementar jwt, el token será devuelto al usuario una vez que ingrese los siguientes datos: • Nombre de Usuario • Contraseña		
Pruebas de Aceptación: • El token tendrá una validez de dos horas. • Los claims del token incluirán: Rol de usuario Nombre de usuario		

Fuente: Propia

TABLA 7
HISTORIA DE USUARIO SWAGGER

Historia de Usuario		
Código: REFE-002	**Sistema:** Ecommerce Admin	
Nombre: Implementación de swagger al Api Rest		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 4
Descripción: Implementar Interfaz gráfica de swagger para acceder y dar información acerca de cada endpoint del Api Rest.		
Pruebas de Aceptación: • Todos los endpoints del Api debén estar documentados. • Swagger permitirá el acceso a los endpoints acorde al rol de usuario. • El único endpoint abierto será el endpoint para hacer login. • La interfaz gráfica de swagger estará disponible en un dominio de internet. • El dominio de internet deberá contar un certificado SSL. • Swagger deberá indicar oportunamente al usuario como debe colocar su JSON web token para acceder a los demás endpoints.		

Fuente: Propia

TABLA 8
HISTORIA DE USUARIO COLECCIONES POSTMAN ECOMMERCE ADMIN

Historia de Usuario

Código: REFE-003	**Sistema:** Ecommerce Admin	
Nombre: Implementación de colección Postman		
Prioridad: Medio	**Riesgo:** Medio	**Estimación:** 2

Descripción:
Implementar una colección de Postman que documente cada una de las peticiones del Ecommerce Admin.

Pruebas de Aceptación:
La colección de postman deberá tener las siguientes variables.

- Variable con el valor de la url del entorno de pruebas.
- Variable con el valor de la url del entorno de producción.
- Variable con el valor del jwt.
- Los endpoints deberán estar agrupados por folder de acuerdo con su función.
- Cada endpoint deberá tener una descripción y un ejemplo de la url para su acceso.

Fuente: Propia

TABLA 9
HISTORIA DE USUARIO LOGIN Y DATA USUARIOS

Historia de Usuario

Código: REFE-005	**Sistema:** Ecommerce Admin	
Nombre: Endpoints de login y data de usuarios.		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 3

Descripción:
Se realizará la implementación del endpoint para hacer login, además se necesita un endpoint que dependiendo del nickname de usuario devuelva los datos correspondientes a:

- Número de cédula
- Ruc
- Razón Social
- Nombres
- Apellidos
- Fecha de Nacimiento
- e-mail
- teléfono

Pruebas de Aceptación:

- El endpoint de login deberá devolver un jwt.
- El endpoint de data devolverá un objeto JSON con los datos dependiendo del usuario.
- El endpoint de data únicamente podrá ser accedido si el usuario tiene un token válido.

Fuente: Propia

TABLA 10
HISTORIA DE USUARIO COMPRAS ACEPTADAS

Historia de Usuario

Código: REFE-006 **Sistema:** Ecommerce Admin

Nombre: Implementación de endpoint a compras aceptadas simuladas.

Prioridad: Alta **Riesgo:** Alto **Estimación:** 3

Descripción:
Realizar un endpoint que simule compras aceptadas, este api es necesaria ya que es el punto de partida del proceso que realiza el ecosistema de software que se está diseñando, este endpoint tendrá los siguientes datos.

- Código postal comprador.
- Código postal vendedor.
- Peso volumétrico.

Este punto de acceso al web service tendrá las siguientes operaciones http:

- Get [nickname] TodasComprasAceptadas Vendedor
- Get [nickname] TodasComprasAceptadas Comprador
- Get [Objeto (nickname, idCA)] CompraAceptada Vendedor
- Get [Objeto (nickname, idCA)] CompraAceptada Comprador
- Post [Objeto CompraAceptada] Insertar CompraAceptada

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibless únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.

Fuente: Propia

TABLA 11
HISTORIA DE USUARIO SOLICITUDES COURIER

Historia de Usuario

Código: REFE-007 **Sistema:** Ecommerce Admin

Nombre: Implementación de endpoint de solicitudes de courier.

Prioridad: Alto **Riesgo:** Medio **Estimación:** 4

Descripción:
Se necesita realizar en el web service endpoints que le permitan al rol vendedor obtener la información de solicitud del courier, para ellos se realizará las siguientes operaciones http.

- Get [nicknameVendedor]
- Get [codeCompraAceptada]
- Get [int codeEntregaCourier]

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibles únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.

Fuente: Propia

TABLA 12
HISTORIA DE USUARIO RECEPCIONES DE PAQUETES

Historia de Usuario

Código: REFE-008	**Sistema:** Ecommerce Admin	
Nombre: Implementación de endpoint de recepciones de paquetes.		
Prioridad: Alto	**Riesgo:** Medio	**Estimación:** 3

Descripción:
Se necesita un endpoints en el web service que le permita al rol comprador registrar, actualizar la recepción de paquetes, además se requiere que el rol vendedor pueda obtener la información de recepción de paquetes, para ello se necesita las siguientes operaciones https.

- POST[ObjetoTrxRecepcionesPaquetes]
- PUT[ObjetoTrxRecepcionesPaquetes]
- Get [string nicknameVendedor]
- Get [string nicknameComprador]
- Get [int codeCompraAceptada] Comprador
- Get [string codeGuiaTransporte] Comprador
- Get [int codeCompraAceptada] Vendedor
- Get [string codeGuiaTransporte] Vendedor

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibles únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.

Fuente: Propia

TABLA 13
HISTORIA DE USUARIO TRANSACCIONES ISO 8583

Historia de Usuario

Código: REFE-009	**Sistema:** Ecommerce Admin	
Nombre: Implementación de endpoint para transacciones ISO 8583		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 5
Descripción:		

Se requiere para la solicitud del courier, la entrega al courier y la recepción del paquete endpoints que permitan obtener las transacciones con formato ISO 8583, para ellos es necesario implementar los siguientes endpoints:

- Get [codeSolicitudCourier]
- Get [codeEntregaCourier]
- Get [codeRecepcionPaquete]

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibles únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.
- Los endpoints implementarán todos los apartados de la ISO 8583.
- Los endpoints tendrán un id que les identifique.

Fuente: Propia

TABLA 14
HISTORIA DE USUARIO JSON WEB TOKEN COURIER

Historia de Usuario		
Código: RCFE-010	**Sistema:** Courier	
Nombre: Implementación de JWT en la Api Restful del Courier		
Prioridad: Media	**Riesgo:** Medio	**Estimación:** 3

Descripción:
El web service Api Restful del Courier deberá implementar jwt, el token será devuelto al usuario una vez que ingrese los siguientes datos:

- Nombre de Usuario
- Contraseña

Pruebas de Aceptación:

- El token tendrá una validez de dos horas.
- Los claims del token incluirán:
  Rol de usuario
  Nombre de usuario

Fuente: Propia

TABLA 15
HISTORIA DE USUARIO COLECCIÓN POSTMAN COURIER

Historia de Usuario		
Código: RCFE-011	**Sistema:** Courier	
Nombre: Realizar una colección de Postman para la Api del Courier		
Prioridad: Media	**Riesgo:** Medio	**Estimación:** 3

Descripción:

Implementar una colección de Postman que documente cada una de las peticiones del Courier.

Pruebas de Aceptación:
La colección de Postman deberá tener las siguientes variables.

- Variable con el valor de la url del entorno de pruebas.
- Variable con el valor de la url del entorno de producción.
- Variable con el valor del jwt.
- Los endpoints deberán estar agrupados por folder de acuerdo con su función.
- Cada endpoint deberá tener una descripción y un ejemplo de la url para su acceso.

Fuente: Propia

TABLA 16
HISTORIA DE USUARIO CREDENCIALES Y AUTENTICACIÓN COURIER

Historia de Usuario

Código: RCFE-012 **Sistema:** Courier

Nombre: Implementar endpoints de login Courier.

Prioridad: Alta **Riesgo:** Alto **Estimación:** 4

Descripción:
Se necesita realizar un endpoint que permita al courier logearse, para ello es necesario implemetar la siguiente operación http:

- POST [nickname, password] Autenticación usuario.

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- El endpoint de login deberá devolver un jwt.
- El endpoint de data devolverá un objeto JSON con los datos dependiendo del usuario.
- El endpoint de data únicamente podrá ser accedido si el usuario tiene un token válido.

Fuente: Propia

TABLA 17
HISTORIA DE USUARIO TRACKING GUÍAS DE TRANSPORTE COURIER

Historia de Usuario

Código: RCFE-013 **Sistema:** Courier

Nombre: Implementar endpoints de tracking

Prioridad: Alta **Riesgo:** Alto **Estimación:** 7

Descripción:
El courier debe ser capaz de tener un punto de acceso en el que se pueda conocer cuál es el estado de tracking de un paquete, a continuación, se detalla cada una de las etapas de tracking.

- Recolectar
- Recolectado
- En bodega

- En transito
- Zona entrega
- Entregado

Para cumplir con las acciones necesarias se deberá realizar las siguientes acciones y peticiones http:

- POST [codGuia] Tracking Guías
- Get Tracking Guías
- GET [ID Tracking Guías]
- PUT [ID Tracking Guías]
- DEL [ID Tracking Guías]

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibles únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.

Fuente: Propia

TABLA 18
HISTORIA DE USUARIO GUÍAS DE TRANSPORTE COURIER

Historia de Usuario		
Código: RCFE-014	**Usuario:** Courier	
Nombre: Implementar endpoints guías de transporte courier.		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 4

Descripción:
El courier necesita implementar endpoints en su api que le permitan insertar, actualizar y obtener guías de transporte, para ello deberá implementar las siguientes transacciones http:

- GET Courier Guías de Transporte
- GET [ID] Guía de transporte por ID
- GET [cod/{CodigoGuiaCourier}]
- GET [tracking/{CodigoGuia}] tracking guía
- POST [Objeto JSON]Courier Guías de Transporte
- PUT [Objeto JSON]Courier Guías de Transporte
- GET [detalles] tracking por guía
- GET [detalles/LC41347741]

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibles únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.

Fuente: Propia

TABLA 19
HISTORIA DE USUARIO DETALLE DE GUÍAS DE TRANSPORTE COURIER

Historia de Usuario

Código: RCFE-015	**Sistema:** Courier	
Nombre: Implementar endpoints detalle guías de transporte.		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 5

Descripción:
Se requiere endpoints que permitan la creación de detalles de guías de transporte, esto ya que el detalle indica para un paquete si han existido varios procesos de creación de guías de transporte, pero solo uno de los detalles se mantiene como activo, para ellos se requiere implementar las siguientes operaciones http:

- POST [codGuia] Detalle Guías Transporte
- GET Detalles Guías Transporte
- GET ID Detalle Guías Transporte
- PUT ID Detalle Guías Transporte

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibles únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.

Fuente: Propia

TABLA 20
HISTORIA DE USUARIO COLECCIÓN DE GUÍAS POR VENDEDOR

Historia de Usuario

Código: RCFE-016	**Sistema:** Courier	
Nombre: Implementación de endpoint de guías de transporte por vendedor.		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 6

Descripción:
El Courier deberá exponer un endpoint que le permita obtener varias guías de transporte, recibiendo los códigos de las guías de transporte requeridas, para ello se implementará la siguiente transacción http:

- GET [{CodigosGuíasTransporte}] etapa tracking mayor varias guías.

Importante: Cada uno de los puntos de acceso se explican por sí mismos, iniciando con el verbo http seguido de corchetes y entre ellos los parámetros que el endpoint deberá recibir.

Pruebas de Aceptación:

- Los endpoints deberán estar accesibles únicamente para el rol que corresponda.
- Los endpoints estarán protegidos por jwt.
- La implementación de los endpoints deberá estar protegida por https.

Fuente: Propia

TABLA 21
HISTORIA DE USUARIO SPLASH SCREEN Y AUTENTICACIÓN

Historia de Usuario		
Código: RMFE-017	**Sistema:** APP Móvil	
Nombre: Realizar splash screen Ecommerce Admin App		
Prioridad: Media	**Riesgo:** Bajo	**Estimación:** 3
Descripción: La aplicación deberá mostrar una pantalla personalizada al iniciar la aplicación por primera vez, tendrá el logo del Ecommerce Admin en un fondo blanco.		
Pruebas de Aceptación: • Los endpoints deberán estar accesibles únicamente para el rol que corresponda. • Los endpoints estarán protegidos por jwt. • La implementación de los endpoints deberá estar protegida por https.		

Fuente: Propia

TABLA 22
HISTORIA DE USUARIO LOGIN Y DASH BOARD USUARIOS

Historia de Usuario		
Código: RMFE-018	**Sistema:** APP Móvil	
Nombre: Realizar Login y Menús Principales de Navegación.		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 8
Descripción: La aplicación requerirá una pantalla de login en la cual se ingresará el usuario y contraseña. La aplicación que dependerá del rol deberá mostrar una pantalla con las acciones que el usuario puede realizar, un conjunto de botones en un screen con una imagen de portada sería lo ideal.		
Pruebas de Aceptación: • En caso de que el usuario no ingrese credenciales correctas se mostrará una alerta. • El usuario tendrá un número de intentos parametrizado al realizar el login. • En caso de que se supere el número de intentos se bloqueara el login por un tiempo determinado. • Los menús principales mostrarán botones dependiendo del rol. • Se verificará que el usuario tenga acceso a internet. • En el caso que el usuario no tenga una conexión activa de internet se mostrará una alerta.		

Fuente: Propia

TABLA 23
HISTORIA DE USUARIO VISUALIZACIÓN DATA USUARIOS

Historia de Usuario	
Código: RMFE-019	**Sistema:** APP Móvil
Nombre: Realizar la visualización de la data según el usuario.	

Prioridad: Media	**Riesgo:** Bajo	**Estimación:** 6

Descripción:
El usuario será capaz de visualizar sus datos personales, tendrá accesibilidad a esta funcionalidad desde el screen principal.

Pruebas de Aceptación:

- El usuario deberá visualizar sus datos.
- El acceso a los datos estará delimitado por rol y nickname.
- Se verificará que el usuario tenga acceso a internet.
- En el caso que el usuario no tenga una conexión activa de internet se mostrará una alerta.

Fuente: Propia

TABLA 24
HISTORIA DE USUARIO FUNCIONALIDAD COMPRAS ACEPTADAS APP

Historia de Usuario

Código: RMFE-020	**Sistema:** APP Móvil

Nombre: Realizar la visualización de compras aceptadas.

Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 10

Descripción:
El rol vendedor tendrá un listado de las compras aceptadas que vá a despachar, mientras que el rol comprador tendrá la posibilidad de visualizar un listado de las compras que ha realizado.

Pruebas de Aceptación:

- El rol vendedor visualizará las compras aceptadas en una lista.
- Las compras aceptadas se mostrarán en orden de fecha
- Se verificará que el usuario tenga acceso a internet.
- Sí el usuario no tenga una conexión activa de internet se mostrará una alerta.

Fuente: Propia

TABLA 25
HISTORIA DE USUARIO FUNCIONALIDAD SOLICITUD COURIER APP

Historia de Usuario

Código: RMFE-021	**Sistema:** APP Móvil

Nombre: Realizar la solicitud del courier por parte del Vendedor.

Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 6

Descripción:

Pruebas de Aceptación:

- El rol vendedor deberá tener la capacidad de realizar la solicitud del courier.
- La funcionalidad será accedida desde un botón en el listado de compras aceptadas.
- Se verificará que el usuario tenga acceso a internet.
- En el caso que el usuario no tenga una conexión activa de internet se mostrará una alerta.

Fuente: Propia

TABLA 26
HISTORIA DE USUARIO FUNCIONALIDAD ENTREGA COURIER APP

Historia de Usuario		
Código: RMFE-022	**Sistema:** APP Móvil	
Nombre: Registrar la entrega de paquete al Courier.		
Prioridad: Alto	**Riesgo:** Alto	**Estimación:** 7
Descripción: El rol vendedor necesita registrar el despacho del paquete mediante el courier al cliente para ello deberá implementarse esa funcionalidad.		
Pruebas de Aceptación: • El rol vendedor registrará la entrega del paquete al Courier en un screen de la App • El rol vendedor accederá al registro desde un listado. • Se verificará que el usuario tenga acceso a internet. • En el caso que el usuario no tenga una conexión activa de internet se mostrará una alerta.		

Fuente: Propia

TABLA 27
HISTORIA DE USUARIO FUNCIONALIDAD TRACKING DE PAQUETES

Historia de Usuario		
Código: RMFE-023	**Sistema:** APP Móvil	
Nombre: Realizar la visualización de estado de tracking.		
Prioridad: Media	**Riesgo:** Medio	**Estimación:** 6
Descripción: El rol vendedor y el rol comprador tendrán la posibilidad de visualizar el tracking de una compra aceptada en específico.		
Pruebas de Aceptación: • El rol vendedor y comprador podrán visualizar la etapa en la que se encuentra el tracking. • El rol vendedor podrá acceder el estado de tracking desde un botón de un listado. • Se verificará que el usuario tenga acceso a internet. • En el caso que el usuario no tenga una conexión activa de internet se mostrará una alerta.		

Fuente: Propia

TABLA 28
HISTORIA DE USUARIO FUNCIONALIDAD ACEPTACIÓN DE PAQUETES

Historia de Usuario		
Código: RMFE-018	**Sistema:** APP Móvil	
Nombre: Implementar el registro de aceptación de paquete.		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 6

Descripción:
El comprador tendrá la posibilidad de registrar la aceptación o no de un paquete, esto con la finalidad de que en el caso de no aceptación se lleve un registro.

Pruebas de Aceptación:

- El rol comprador visualizará un screen para realizar la aceptación de un paquete.
- El tiempo en el que el usuario puede aceptar el paquete será parametrizado.
- El usuario accederá a la aceptación de paquete por medio de un listado.
- Se verificará que el usuario tenga acceso a internet.
- En el caso que el usuario no tenga una conexión activa de internet se mostrará una alerta.

Fuente: Propia

TABLA 29
HISTORIA DE USUARIO NOTIFICACIONES PUSH APP

Historia de Usuario		
Código: RMFE-018	**Sistema:** APP Móvil	
Nombre: Implementar notificaciones push en las acciones.		
Prioridad: Alta	**Riesgo:** Alto	**Estimación:** 6

Descripción:
La aplicación móvil consume las Api de los sistemas Ecommerce Admin y Courier, mediante la aplicación interaccionan los sistemas del ecosistema de software por lo que se requiere notificaciones push para las siguientes acciones:

- El rol cliente recibe una notificación push al realizar el despacho de un paquete.
- El rol vendedor recibe una notificación push cuando el cliente ha recibido el paquete.
- El rol vendedor recibe una notificación push cuando el cliente a aceptado o no el paquete recibido.

Pruebas de Aceptación:

- Las notificaciones push se implementarán en un sistema de nube.
- Las notificaciones push se realizarán acorde el flujograma del proceso.
- Las notificaciones únicamente llegarán si el usuario tiene accesibilidad a internet.

Fuente: Propia

2.1.5 Roles del ecosistema de software

TABLA 30
ROLES DEL ECOSISTEMA DE SOFTWARE

Sistema	Rol
Ecommerce Admin	• Administrador. • Vendedor. • Comprador.
Courier	• Administrador.

Fuente: Propia

2.2 Fase de Exploración

2.2.1 Arquitectura de los sistemas pertenecientes al Ecosistema de Software.

La arquitectura de los sistemas que convergen en el sistema de software diseñado se diseñó en base a patrones de diseño y buenas prácticas a continuación se explica la arquitectura del ecosistema de software:

Arquitectura Sistemas de Apis RestFul Ecommerce Admin y Courier:

Para los sistema de APIs RestFul se realizó una aplicación con el patrón de diseño de MVC, el cual encuentra persistencia en una base de datos Sql Server, además se protege los endpoints de la aplicación JSON Web Tokens, en el caso del Ecommerce Admin se pone a disposición la información descriptiva de los puntos de acceso de las Api con Swagger UI, en la figura 16, se puede visualizar el esquema básico de la arquitectura para el Ecosistema de software, tomando en cuenta que el controller está protegido por jwt y el cliente es una aplicación móvil.

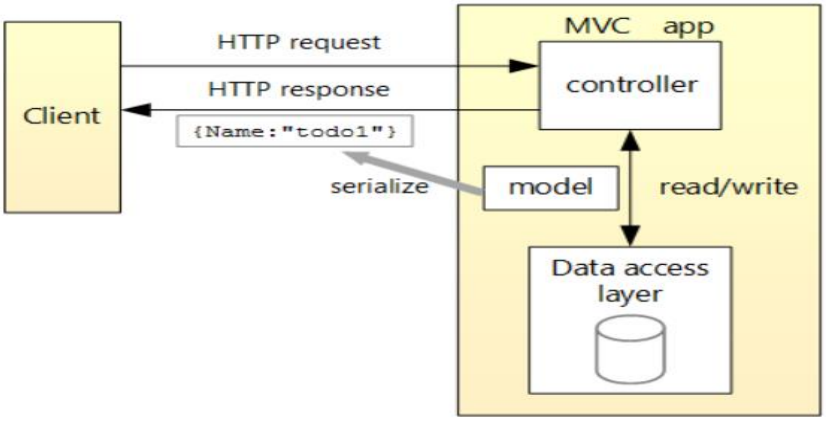


Fig. 16. Arquitectura APIs RestFul
Fuente: (Microsoft Docs, 2020)

Aplicación Móvil:

La aplicación Móvil dentro de la arquitectura que se muestra en la figura 16 es el cliente, su desarrollo se realizó con el patrón de diseño de software MVVM que se muestra en la figura 17.

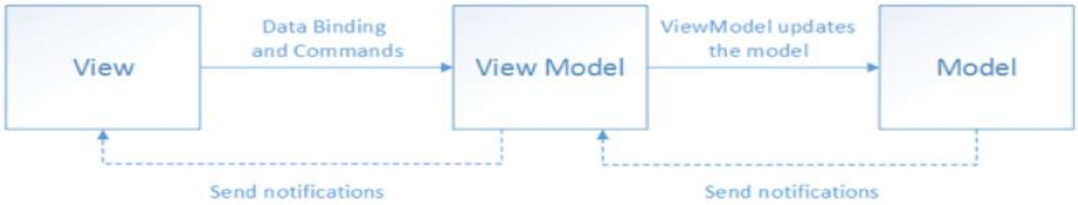


Fig. 17. Arquitectura Aplicación Móvil.
Fuente: (Microsoft Docs, 2017)

2.2.2 Diagrama entidad relación de la base de datos del Ecommerce Admin

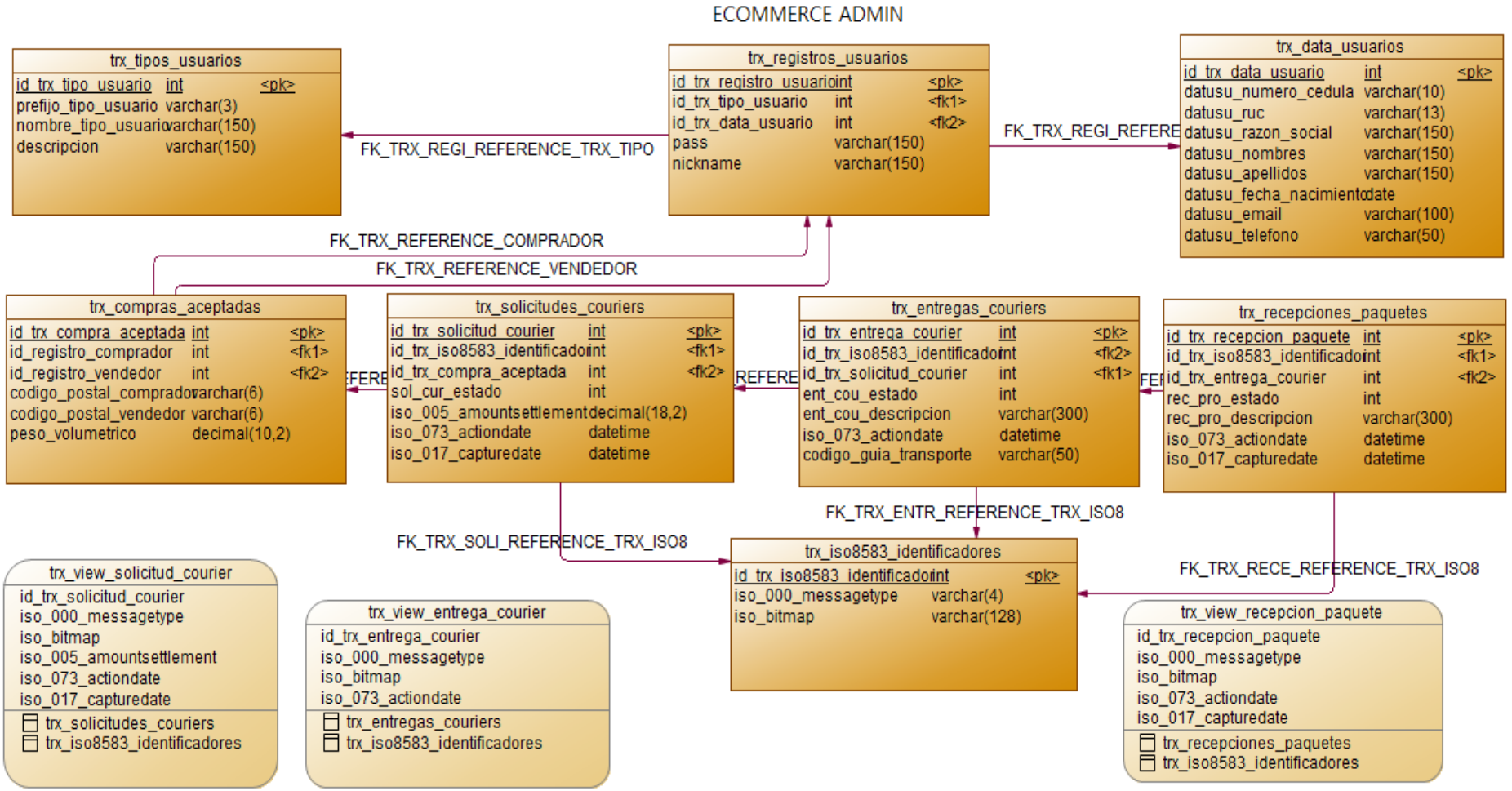


Fig. 18. Modelo Físico Ecommerce Courier
Fuente: Propia

2.2.3 Diagrama entidad relación de la base de datos del Courier

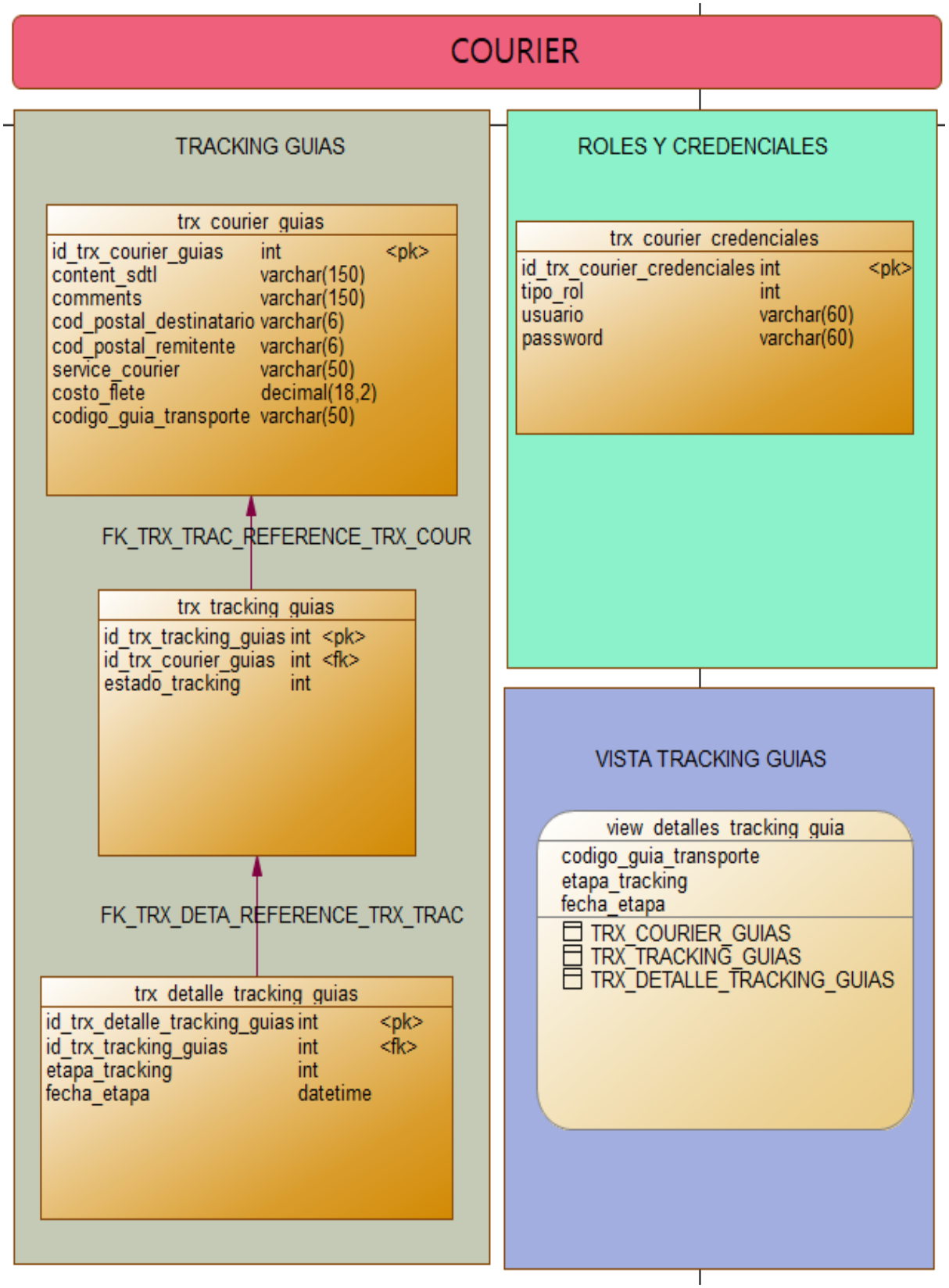


Fig. 19. Modelo Físico Courier
Fuente: Propia

2.3 Ecosistema de software en Producción

Para la puesta en producción del ecosistema de software se consideró que todos los componentes estén disponibles en la internet, pues de ello dependería que se puede estimar la eficiencia del servicio RestFul del Ecommerce Admin, que es una de las mayores prioridades del diseño y realización de este ecosistema de software, es por ello que se usó hospedaje en hosting y en sistemas de nube para la puesta en producción de los web services del Courier, el Ecommerce Admin y del entorno de pruebas, en la figura 20, se observa un esquema ilustrativo.

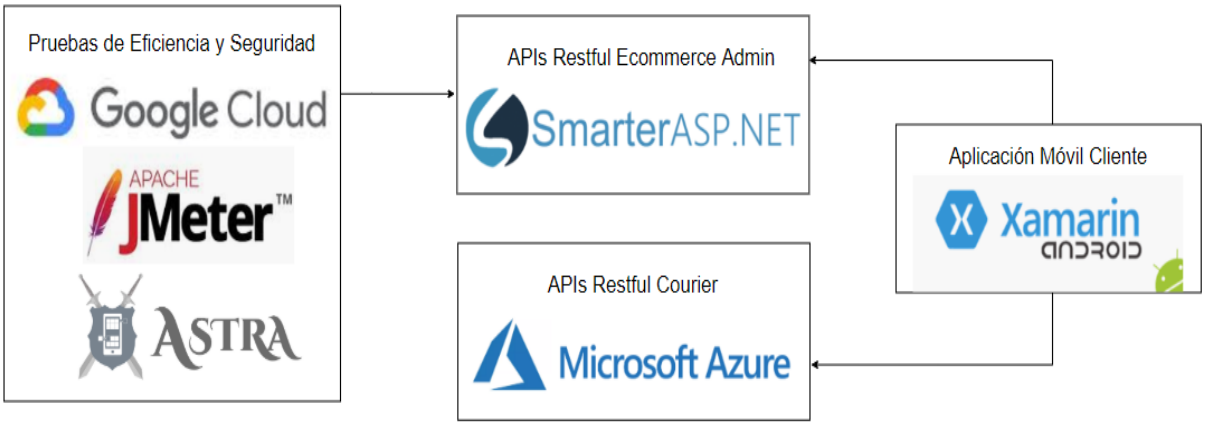


Fig. 20. Puesta en producción ecosistema de software.
Fuente: Propia

2.3.1 Puesta en producción Api Restful Ecommerce Admin.

El sistema del Ecommerce Admin se puso en producción en hosting compartido, en el anexo E, se puede apreciar algunas capturas de pantalla de un landing page que se implementó y de la UI de Swagger que posee el sistema.

La base de datos Sql Server del Ecommerce Admin, que se muestra en la figura 18, fue implementada en este hosting, las características técnicas que fueron contratadas en el hosting compartido para hospedar la aplicación y la base de datos se muestran a continuación en la tabla 30.

TABLA 31
CARACTERÍSTICAS TÉCNICAS DEL HOSPEDAJE DEL ECOMMERCE ADMIN

Sistema Ecommerce Admin	Carácterísticas del Hospedaje Smarterasp
APIs RestFul	• Aplicación .Net Core 3.1 • 256 Mb Ram • Certificado SSL 256 bits
Base de Datos	• MSSQL 2017. • Cuota de disco 600 Mb

Fuente: Propia

2.3.2 Puesta en producción Api Restful Courier

Para hospedar el sistema de APIs Restful del sistema simulado del courier se optó por una solución en la nube de Azure, en ese contexto se utilizó una instancia de App Service, este sistema a diferencia del Ecommerce Admin no implementa Swagger UI, únicamente accede a la base de datos del Courier, mediante, transacciones de tipo http devolviendo data en formato JSON.

La Base de Datos del Courier que se muestra en la figura 19, se implementó en un plan básico en el cloud de Azure, a continuación, en la tabla 31 se detallan los aspectos técnicos para el hospedaje del sistema del Courier simulado.

TABLA 32
CARACTERÍSTICAS TÉCNICAS DEL HOSPEDAJE DEL COURIER

Sistema Ecommerce Admin	Carácterísticas del Hospedaje Azure
APIs RestFul	• Aplicación .Net Core 3.1 • 1 Gb Ram
Base de Datos	• MSSQL 2019. • Cuota de disco 2 Gb

Fuente: Propia

2.3.3 Puesta en producción Aplicación Móvil

La aplicación móvil se desarrolló en Xamarin, esta aplicación está disponible para usar en los sistemas Android 5.0 Lollipop que corresponde a la API 21 de Android y ha sido diseñada para una versión destino de Android 9.0 Pie que corresponde a la API 28, esta aplicación por el momento no está disponible en tiendas de aplicaciones.

En el anexo E, se muestran varias capturas de pantalla de la aplicación en los dispositivos con los cuales se realizó las pruebas durante el desarrollo, a continuación, se listan los dispositivos en los que se puso a prueba la aplicación móvil.

- Samsung J700M, API 23.
- Samsung GT-UI9082L API 24
- Samsung A30 API 28

Los dispositivos físicos que se tuvo disponible durante el desarrollo son los anteriormente mencionados, no obstante, también se realizó pruebas con emuladores y diferentes versiones de las API de Android.

2.3.4 Puesta en producción Pruebas de Software

- Pruebas de Eficiencia:

Las pruebas de software se realizaron con Jmeter, fue necesario tomar en cuenta varios factores para la realización de las pruebas entre ellas el tipo de pruebas que se realizaría dentro de todo el abanico de posibles pruebas se realizó pruebas de estrés, es decir haría falta para tener una buena calidad que el cliente en el que se aloje el entorno de pruebas tenga buena capacidad de procesamiento, este dotado de la suficiente memoria principal RAM y el ancho de banda sea estable.

Se analizaron varias alternativas, finalmente se optó por hospedar el entorno de pruebas en una máquina virtual en el sistema de nube Google Cloud, para ello fue necesario instalar un sistema operativo Linux y mediante consola SSH instalar una interfaz gráfica, en el anexo A, se evidencia el script de instalación de la GUI, además, se instaló el JDK 1.8 de Java, necesario para que Jmeter pueda correr las pruebas de carga hasta llegar al estrés.

- Pruebas de Seguridad:

Para realizar las pruebas de seguridad se tuvo que buscar una herramienta que provea todas las bondades necesarias para realizar un testing que arroje las vulnerabilidades que pueden representar riesgos.

Para la búsqueda de herramientas que permitan realizar estas pruebas fue necesario tomar en cuenta los parámetros establecidos por OWASP, para tener un mejor contexto de los puntos clave que es necesario poner a prueba, entre varias opciones se tomó como herramienta de pentesting a Astra-Security, pues presta todas las facilidades para ejecutar un testing completo.

La puesta en producción de Astra-Security tiene varios prerrequisitos que se exponen, a continuación:

- Sistema Operativo Linux o MacOS
- Python 2.7
- Mongo DB 5.03

Al clonar el repositorio de GIT Hub de la herramienta se crearon los directorios y servicios necesarios, se obtuvo algunas dificultades entre ellas la versión de Mongo DB debe ser específicamente la 5.03, además se realizó el debug y la refactorización de varios de los archivos de Astra-Security, por ejemplo, en el archivo Astra.py se encontró algunos problemas con la identación que tuvieron que ser resueltos.

2.3.5 Características del ambiente de pruebas de eficiencia y seguridad

En la tabla siguiente se detallan las características técnicas de Hardware y software con las que contó la máquina que hospedó el entorno de pruebas de eficiencia y seguridad.

TABLA 33
CARACTERÍSTICAS TÉCNICAS DEL HOSPEDAJE DEL ENTORNO DE PRUEBAS

Entorno de Pruebas	Carácterísticas de la máquina virtual en Google Cloud
Hardware	• Alojado en la zona us-central1 • 8 núcleos de procesamiento serie estándar • 30Gb de memoria principal RAM • Ancho de Banda de Subida 960 Mb • Ancho de Banda de Bajada 800 Mb
Software	• SO Ubuntu 16.04 • JDK 1.8 Java • Jmeter 15.01 • NetData 1.23 • Python 7.2 • Mongo DB 5.03 • Astra-Security

Fuente: Propia

CAPÍTULO 3
Validación de resultados

3.1 Pruebas de eficiencia

Para el diseño de pruebas se analizaron varios aspectos entre ellos la herramienta que se usaría, el tipo de pruebas que se harían, como cohesionarían las pruebas para alcanzar los objetivos del proyecto y cuál sería la estrategia y el estándar con el que se evaluaría la eficiencia de los endpoints de interés en la API Restful del Ecommerce Admin.

3.1.1 Elección de la herramienta para realizar las pruebas de eficiencia

Existen varias herramientas para analizar pruebas de desempeño del software hay varias alternativas libres que coexisten con alternativas de pago, para la realización del presente proyecto se analizaron varias de esas herramientas con la finalidad de utilizar la mejor alternativa, a continuación, se detalla en formato de tabla cada una de las herramientas analizadas y sus características.

TABLA 34
HERRAMIENTAS PARA TEST DE DESEMPEÑO

Herramienta	Características
Gatling	• Versión Completa de Pago • Open Source • Unlimited testing multihilo • Graphical User Interface
JMeter	• Versión Completa Gratis • Open Source • Graphical User Interface • Reporte Html estadísticas Apdex • Unlimited testing multihilo • Soporte multi protocolo
The Grinder	• Versión Completa Gratis • Open Source • Graphical User Interface • Soporte multi protocolo • Unlimited testing multihilo

Fuente: Propia

Tras realizar un análisis de las principales características de las herramientas para realizar pruebas de desempeño, se encontró en JMeter la mejor alternativa, esta herramienta es completamente libre y gratis, además es ampliamente usada en la industria del software, otra de las principales razones para la elección de esta herramienta fue provee las estadísticas encontradas en base al indicador APDEX en formato html, además para su

ejecución solo es necesario instalar el jdk 1.8 de Java, lo cual en la puesta en producción sería de una manera bastante eficiente y sencilla.

3.1.2 Endpoints de Interés

El ecosistema de software diseñado, como se específica en el apartado 2.1, se compone dos sistemas API Restful, el sistema de Courier simulado no es de interés central, por otra parte de interés fundamental son las APIs del sistema Ecommerce Admin, se desarrollaron varios endpoints, sin embargo, la parte fundamental radica en conocer la eficiencia de los puntos de salida que representan transacciones y están basados en la ISO 8583, a continuación se detalla varios aspectos de las por las cuales estos endpoints son de interés.

- Interactuarán dentro un macroecosistema de software.
- Reflejan en términos económicos las acciones principales del Ecommerce Admin.
- Implementan el estándar ISO 8583.
- El desempeño de estos endpoints, refleja la eficiencia del sistema Ecommerce Admin.

Debido a los puntos anteriormente expuestos el diseño y realización de un plan de pruebas se aplicará en los endpoints relacionados con transacciones que implementan el estándar ISO 8583 y son parte del Ecommerce Admin.

3.1.3 Característica de capacidad de la Norma ISO 25010.

El desempeño de los endpoints de interés se evalúa acordé a la característica de capacidad de la norma ISO 25010, en específico esta característica se define como el grado en el cual los máximos límites de un sistema o producto en el caso particular del presente trabajo los endpoint de interés se desempeñan para cumplir con los requerimientos del ecosistema de software diseñado.

En base a la premisa anteriormente expuesta se define el tipo de pruebas a realizar y el diseño del plan para evidenciar el desempeño y llegar a una conclusión mesurable.

3.1.4 Tipo de pruebas y diseño de plan de pruebas de eficiencia.

Se ha establecido los puntos de salida de interés de las APIs del Ecommerce Admin, la herramienta para realizar las pruebas y la norma o estándar a aplicar, en base a ello se determina el tipo de pruebas a aplicar y se diseña el plan de pruebas que se aplicará.

Se realizan pruebas de carga, una prueba de carga consiste en poner una determinada carga de trabajo a un sistema y medir la capacidad que dicho sistema puede soportar, este tipo de prueba se realizará con diferentes cargas en varias repeticiones hasta llegar al estrés de los endpoints de interés.

El diseño y la estructura del plan de pruebas se basa en un grupo de hilos, que tendrá una petición http de tipo get por cada enpoint de la API del Ecommerce Admin que implementa la ISO 8583, además, la prueba implementa cinco receptores, los cuales permiten tener acceso a los datos recopilados por las pruebas, en formato estadístico y gráficos, en el anexo C, se muestra el plan de pruebas implementado en la herramienta JMeter.

3.1.5 Resultados Obtenidos

Se realizaron siete pruebas de carga hasta llegar al estrés de las API, se comenzó con una carga de 50 hilos en ejecución simultánea hasta cumplir 600 ciclos, la segunda fue una prueba de 100 hilos, a partir, de esta prueba se aumenta 100 hilos por cada prueba de carga, se recogieron los datos y se exportaron a html para obtener los valores del índice Apdex.

Se proporcionó al entorno de pruebas de un hardware lo suficientemente potente, ya que una de las principales preocupaciones fue evitar factores externos que puedan influir en los resultados es por ello que tal cual se muestra el punto 2.3.4, para la puesta en producción de las pruebas se utilizó una máquina virtual en Google Cloud para la puesta en marcha de las pruebas mediante Jmeter, por esa razón se hizo un monitoreo del uso del hardware del equipo cliente.

3.1.6 Resultados pruebas de Carga

Realizadas las pruebas de carga se presenta los cuadros de estadística de cada una las pruebas de carga realizadas, las columnas de las tablas corresponden a:

- Etiqueta: Endpoint al que se le realiza la carga.
- #Hilos: Cantidad de hilos concurrentes para la realización de la carga.
- Prom: En milisegundo el tiempo promedio de los resultados de carga.
- Min: En milisegundos el tiempo mínimo que un hilo concurrente a logrado cumplir con la petición http.
- Max: En milisegundos el tiempo máximo que un hilo concurrente a realizado la petición http.

- Rendimiento: Se representa como el número de transacciones o peticiones http que se han atendido por segundo.
- Recibido: En kilobytes sobre segundo (kb/s), la cantidad de información recibida.
- Enviado: En kilobytes sobre segundo (kb/s), la cantidad de información que ha sido enviada.

A continuación, en formato de tabla se muestra los resultados obtenidos para las pruebas de carga hasta llegar al estrés.

TABLA 35
PRUEBA 1 DE CARGA 50 HILOS CONCURRENTES

Peticiones	**Executiones**	**Tiempos de Respuesta (ms)**			**Rendimiento**	**Red (Kb/s)**	
Etiqueta	#Hilos	Prom	Min	Max	Transacciones/s	Recibido	Enviado
E. Courier	30000	76.90	46	3322	208.78	65.17	128.85
R. Paquete	30000	77.20	46	3218	208.82	74.78	129.29
S. Courier	30000	78.58	46	3305	207.12	79.84	128.24
TOTAL	90000	77.56	46	3322	620.80	218.47	383.96

Fuente: Reporte JMeter, VM Google Cloud

TABLA 36
PRUEBA 2 DE CARGA 100 HILOS CONCURRENTES

Peticiones	**Executiones**	**Tiempos de Respuesta (ms)**			**Rendimiento**	**Red (Kb/s)**	
Etiqueta	#Hilos	Prom	Min	Max	Transacciones/s	Recibido	Enviado
E. Courier	60000	110.09	46	1193	298.19	92.47	184.02
R. Paquete	60000	110.05	45	1144	298.50	106.27	184.82
S. Courier	60000	113.90	45	2922	294.41	112.88	182.28
TOTAL	180000	111.35	45	2922	882.67	308.80	545.92

Fuente: Reporte JMeter, VM Google Cloud

TABLA 37
PRUEBA 3 DE CARGA 200 HILOS CONCURRENTES

Peticiones	**Executiones**	**Tiempos de Respuesta (ms)**			**Rendimiento**	**Red (Kb/s)**	
Etiqueta	#Hilos	Prom	Min	Max	Transacciones/s	Recibido	Enviado
E. Courier	120000	109.19	35	1032	595.88	184.10	367.77
R. Paquete	120000	109.19	37	3237	595.89	211.44	368.94
S. Courier	120000	112.00	37	2489	583.93	225.46	365.25
TOTAL	360000	110.13	37	3237	1768.72	616.67	1093.94

Fuente: Reporte JMeter, VM Google Cloud

TABLA 38
PRUEBA 4 DE CARGA 300 HILOS CONCURRENTES

Peticiones	**Executiones**	**Tiempos de Respuesta (ms)**			**Rendimiento**	**Red (Kb/s)**	
Etiqueta	#Hilos	Prom	Min	Max	Transacciones/s	Recibido	Enviado
E. Courier	180000	440.46	38	4335	225.53	70.69	139.20
R. Paquete	180000	441.56	38	4798	225.54	81.02	139.64

S. Courier	180000	444.12	38	4350	224.96	86.96	139.28
TOTAL	540000	442.05	39	4798	674.81	238.26	417.36

Fuente: Reporte JMeter, VM Google Cloud

TABLA 39

PRUEBA 5 DE CARGA 400 HILOS CONCURRENTES

Peticiones	**Executiones**	**Tiempos de Respuesta (ms)**			**Rendimiento**	**Red (Kb/s)**	
Etiqueta	#Hilos	Prom	Min	Max	Transacciones/s	Recibido	Enviado
E. Courier	240000	285.16	45	4128	418.19	165.31	258.10
R. Paquete	240000	295.45	47	8632	414.30	180.76	256.51
S. Courier	240000	284.50	47	9155	418.24	176.06	258.95
TOTAL	720000	288.37	45	9155	1242.64	518.83	768.56

Fuente: Reporte JMeter, VM Google Cloud

TABLA 40

PRUEBA 6 DE CARGA 500 HILOS CONCURRENTES

Peticiones	**Executiones**	**Tiempos de Respuesta (ms)**			**Rendimiento**	**Red (Kb/s)**	
Etiqueta	#Hilos	Prom	Min	Max	Transacciones/s	Recibido	Enviado
E. Courier	3000000	341.63	37	9267	392.03	171.00	241.95
R. Paquete	3000000	342.46	37	9149	392.07	177.19	242.75
S. Courier	3000000	358.39	40	14708	389.70	179.79	241.28
TOTAL	9000000	347.49	37	14708	1168.93	525.82	722.97

Fuente: Reporte JMeter, VM Google Cloud

Tomadas las estadísticas de cada una de las pruebas, es necesario aclarar que para estas se han considerado todas las solicitudes indiferentemente del tipo de respuesta http que se ha generado, es decir estas estadísticas comprenden peticiones que han tenido éxito y también respuestas que no han tenido éxito, pero finalmente han obtenido una respuesta http 200 o 503, el propósito del presente análisis es estimar la eficiencia en el orden de ms, el tiempo en el que el API, ha sido capaz de recibir una petición y devolver una respuesta, con la finalidad de medir la eficiencia de los resultados se realiza el cálculo de distribución normal tomando una intervalo de confianza del 90% y estableciendo como promedio o también llamada media aritmética el cociente de la sumatoria de los totales de tiempos promedios de cada prueba sobre le número de pruebas realizadas, tomando lo anterior en cuenta, a continuación, se presentan los resultados obtenidos.

TABLA 41

RESULTADOS DE LAS PRUEBAS DE CARGA

Cálculo	**Resultado**
Sumatoria de tiempos promedios.	1276.95 ms
Media Aritmética	229.49 ms
Desviación Estándar	150.90 ms
Porcentaje de Intervalo de Confianza	90%
Límite Inferior del Intervalo de Confianza	218.46 ms
Límite superior del Intervalo de Confianza	240.52 ms

Fuente: Propia

Presentados los resultados, se describe que el 90% de las peticiones recibidas por los endpoints de interés fueron resueltos entre 218.46 ms y 240.52 ms hasta llegar al estrés.

3.1.7 Estrés de las API.

Se llegó al estrés de los endpoint de interés del API, en la séptima prueba de carga realizada, en el punto 3.1 no existe las estadísticas de una prueba 7, la razón es que al sobrecargar el API con 600 hilos concurrentes y esperar la respuesta de 3'000.000 de peticiones en 600 ciclos, dejó de responder por lo que se detuvo la prueba de carga llegando al estrés total de los endpoints de interés, el servidor esta alojado en un host compartido y con 256 mb de memoria principal respondió de una manera bastante aceptable, se detalla la cantidad de respuestas exitosas y fallidas en la siguiente para cada prueba y se presenta un gráfico.

TABLA 42
PORCENTAJES DE ÉXITO Y FALLO HASTA LLEGAR AL ESTRÉS DEL API

Prueba de Carga	Respuestas Exitosas	%Éxito	Respuestas Fallidas	%Fallo
1	90000	100%	0	0%
2	180000	100%	0	0%
3	360000	100%	0	0%
4	540000	99%	545	1%
5	402855	56%	317145	44%
6	308788	34%	591212	66%
7	0	0%	0	100%

Fuente: Reporte JMeter, VM Google Cloud

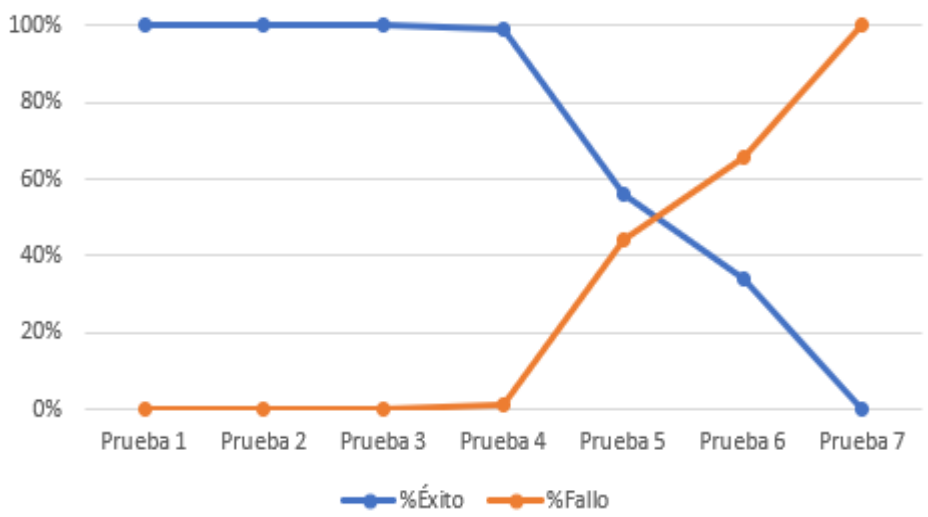


Fig. 21. Gráfica de líneas éxito y fallo.
Fuente: Propia

Al observar los resultados de la gráfica, se puede apreciar que el porcentaje de éxito en las 4 primeras pruebas fue cercano al 100%, es por esto por lo que se considera necesario para obtener una mejor perspectiva del rendimiento bajo determinadas condiciones de los

endpoints de interés analizar cuál es la respuesta media en ms y que resultados se obtiene según el índice desempeño de aplicación APDEX cuando existen 300 hilos concurrentes o menos, esto se realiza en el punto siguiente.

3.1.8 Índice de Desempeño de Aplicación APDEX.

Se realiza el análisis de los resultados del índice de desempeño APDEX, en base a las pruebas de carga que obtuvieron entre un 99% y 100%, es decir aquellas que implementaron 300 hilos concurrentes o menos.

APDEX está definido por la siguiente expresión matemática:

$$Apdex_T = \frac{Satisfied\ count + \frac{Tolerating\ count}{2}}{Total\ samples}$$

Fig. 22. Fórmula índice APDEX
Fuente: (Apdex, 2018)

Para realizar la toma de resultados se consideró un límite de tolerancia 500 ms y un límite de frustración de 1500 ms, a continuación, se presentan los resultados obtenidos.

TABLA 43
RESULTADOS ÍNDICE APDEX

Prueba de Carga	APDEX	%Éxito	Respuesta Promedio en ms
1	1.0	100%	77.56
2	0.996	100%	111.35
3	0.998	100%	110.13
4	0.821	99%	442.05
TOTAL	0.953	99.75%	185.27

Fuente: Reporte JMeter, VM Google Cloud

3.2 Pruebas de Seguridad

Los endpoints de interés del presente proyecto son aquellos que implementan transacciones con formato según la ISO 8583, la seguridad es de vital importancia para este tipo de transacciones que por lo general incluyen un valor económico, por esa razón con la herramienta Astra-Security que considera las recomendaciones fundamentales de OWASP se realizará pruebas automatizadas de pentesting. Astra-Security está desarrollada y disponibles para sistemas Linux y MacOS, requiere la versión de Python 2.7 y una base de datos no relacional Mongo DB.

El aseguramiento de los servicios web o APIs Restful deben considerar pruebas de inyección de dependencias, Cross-site scripting (XSS), Configuración Débil, CSRF y

pruebas de perdida de autenticación(Michelena, 2018), el desarrollo de las APIs Restful han requerido un análisis minucioso y se ha realizado una arquitectura basada en inyección de dependencias, enrutamiento pipeline en base a un middleware, el uso de configuración de servicios y patrones de diseño enfocados a mejorar la eficiencia y solventar los aspectos de seguridad, con la herramienta Astra-Security se realizó dos iteraciones de pruebas a continuación se detalla los resultados obtenidos y las acciones realizadas para un realizar un endurecimiento o hardening en aspectos de seguridad a los puntos de acceso de interés de las API que son.:

- Transacción ISO 8583 Solicitud Courier
- Transacción ISO 8583 Entrega Courier
- Transacción ISO 8583 Recepción Paquete

El análisis de Astra-Security abarca varios tipos de vulnerabilidades a continuación se presenta los check list de las iteraciones realizadas.

3.2.1 Astra-Security test 1

Primer análisis realizado a las APIs de interés con Astra-Security:

TABLA 44
CHECK LIST VULNERABILIDADES ANALIZADAS TEST 1

Tipo de vulnerabilidad	No Vulnerable	Vulnerable
SQL Injection	X	
Cross site scripting	X	
Information Leakage	X	
Broken Authentication	X	
CSRF		X
Rate Limit	X	
CORS misconfiguration	X	
JWT attack	X	
CRLF detection	X	
Blind XXE injection	X	

Fuente: Reporte Astra Linux VM Google Cloud

La iteración o test 1 realizado con Astra-Security, en general obtuvo buenos resultados con respecto a Broken Authentication y JWT attack que son las vulnerabilidades que se consideraron como fundamentales y en la etapa de desarrollo se puso énfasis en cubrir esas posibles debilidades implementando políticas de recursos de origen cruzado CORS, no obstante el análisis en la primera iteración con Astra-Security arrojó que las APIs de interés eran vulnerables a Cross-Site Request Forgery CSRF esta vulnerabilidad trata de la falsificación de solicitudes entre sitios, es decir un sitio diferente al autorizado pero con acceso al información temporal del navegador podría realizar transacciones no autorizadas.

Se solventó la vulnerabilidad CSRF, revisando las políticas de CORS y se aplicó a los endpoints de interés directivas ValidateAntiForgeryToken, en este punto es necesario tener presente que el marco de trabajo usado es .Net Core 3.1, ya que de esta tecnología se encontró solución a la vulnerabilidad parchando este problema, en específico se aplicó "Prevent Cross-Site Request Forgery (XSRF/CSRF) attacks in ASP.NET Core"(Anderson, 2019) preparando las API para una nueva iteración de test con Astra-Security.

3.2.2 Astra-Security test 2

Segundo análisis realizado a las APIs de interés con Astra-Security:

TABLA 45
CHECK LIST VULNERABILIDADES ANALIZADAS TEST 2

Tipo de vulnerabilidad	No Vulnerable	Vulnerable
SQL Injection	X	
Cross site scripting	X	
Information Leakage	X	
Broken Authentication	X	
CSRF	X	
Rate Limit	X	
CORS misconfiguration	X	
JWT attack	X	
CRLF detection	X	
Blind XXE injection	X	

Fuente: Reporte Astra Linux VM Google Cloud

La iteración test 2, tras realizar el parche a la vulnerabilidad CSRF, arrojó resultados positivos lo cual demuestra que los endpoints de interés son seguros y no está expuestos a vulnerabilidades esto corroborado con Astra-Security el cual basa sus pruebas en las acciones recomendadas por OWASP.

En el Anexo E, se pueden apreciar figuras que muestran las pruebas realizadas con Astra-Security.

3.3 Uso de recursos del entorno de pruebas

El uso de recursos del entorno de pruebas no constituye un factor fundamental en el presente proyecto, sin embargo, el entender cuál ha sido la demanda de recursos para llevar a cabo las pruebas de estrés que se le aplicaron a las APIs de interés de este trabajo, muestra una perspectiva clara de los requerimientos de hardware y software necesarios, para medir de manera satisfactoria factores de eficiencia y seguridad.

En el anexo F, se puede evidenciar de manera detallada el uso de recursos del entorno de pruebas constituido por una máquina virtual con sistema operativo Ubuntu 16.04 puesta en producción en Google Cloud.

3.4 Análisis e interpretación de resultados

Se realizaron pruebas de eficiencia y seguridad lo que permite evaluar a las APIs realizadas, a continuación, se presentan el análisis de los resultados obtenidos.

3.4.1 Análisis de los resultados obtenidos en Pruebas de Carga y Estrés

Se realizaron 6 iteraciones de pruebas de carga, el servidor que soporta esta carga cuenta con 256 Mb de memoria principal RAM, este es un dato relevante ya que se sobrecarga al servidor con hilos concurrentes, cada hilo representa un usuario, en un entorno exigente de producción se podría realizar un escalamiento vertical para dotar al servidor de mayor capacidad para soportar usuarios concurrentes, se realiza el análisis tomando en cuenta lo anterior antes expuesto.

El servicio de APIs Restful, se sobrecargo en su primera iteración con 50 hilos concurrentes, en su segunda iteración con 100 hilos, desde la tercera iteración el aumento fue de 100 hilos con 600 ciclos de carga, se denota que hasta la cuarta iteración los resultados fueron superiores o iguales al 99% de éxito soportando hasta 540000 peticiones, desde la quinta iteración la respuesta fue poco satisfactoria teniendo un éxito de 56% o menor, por lo que el análisis indica que con el hardware el sistema de APIs puede soportar hasta 300 hilos concurrente con una respuesta promedio de 185.27 ms.

Al realizar una estimación de distribución normal con los datos obtenidos de las 6 pruebas de carga realizadas y tomando en cuenta un intervalo de confianza del 90%, se denota un límite superior 240.52 y un límite inferior del 218.46 ms, se considera que 0.1 es el tiempo en que un usuario siente que está manipulando eficientemente una respuesta y 1000 ms es el límite en el que considera a la respuesta como fluida(Nielsen, 2016), la respuesta del API al no superar los 1000 ms puede ser considerada como eficiente.

El estrés se obtuvo cuando se sobrecargó las APIs con 600 hilos concurrentes, para soportar este tipo de carga sería necesario dotar al servidor con mejores componentes de hardware.

3.4.2 Análisis de resultados según los indicadores Apdex

El índice APDEX oscila entre 0 y 1 y se define como la medición de los datos para medir la satisfacción del usuario, las APIs son capaces de soportar con una respuesta eficiente

una carga de 300 ms, considerando la media aritmética de las 4 primeras pruebas de carga obtiene un valor de 0.953, APDEX entre más cercano a 1 indica mayor satisfacción por parte del usuario, en el caso de las APIs, debido a que el valor obtenido es cercano a 1, se infiere que la respuesta y eficiencia de los endpoints es satisfactoria.

3.4.3 Análisis de resultados pruebas de seguridad

Se realizaron pruebas de seguridad con Astra-Security, esta herramienta cubre el análisis de vulnerabilidades, según OWASP, en la primera iteración se encontró una vulnerabilidad CSRF, el framework .Net Core, establece un procedimiento para realizar el patch o parche de esta vulnerabilidad se realizó las modificaciones correspondientes y en una segunda iteración se pasó todos los análisis de vulnerabilidades. Con esto se puede catalogar a las APIs desarrolladas como seguras.

La arquitectura de software implementada en el contexto de seguridad puede catalogarse como segura, vulnerabilidades como la perdida de autenticación, inyección sql y ataques al JWT, se encuentran solventadas.

3.4.4 Análisis del uso de recursos del Cliente

El análisis del uso de recursos del cliente que albergó el entorno de pruebas no es un punto fundamental de la presente investigación, sin embargo, puede ser interesante para estudios similares tener una estimación aproximada de los recursos de hardware para llevar a cabo pruebas de software y hardware con solvencia, por ese motivo se menciona este factor.

Durante las pruebas de carga que fueron las que mayor exigencia requirió del hardware se denota que 2 de los 8 núcleos en el punto más álgido de la prueba tuvieron actividad, en aspectos de ancho de banda la subida durante las pruebas fue menor a 0.5 mb/s y 1mb/s de bajada, la memoria principal tuvo una exigencia aproximada de 1.3 Gb, por lo que se define que un hardware promedio podría solventar el alojamiento de pruebas de eficiencia y seguridad.

CONCLUSIONES

El presente proyecto se desarrolló como uno de los módulos que aportarán a la consecución del proyecto e investigación científica denominada ”Diseño de plataforma tecnológica de medio de pago local para el comercio electrónico en Ecuador, mediante la integración de infraestructuras de servicio tecnológico financiero (FINTEC)” (Trejo & Ortega, 2018), el desarrollo del prototipo realizado permite la interoperabilidad vendedor y Courier el cual es necesario para apalancar el comercio electrónico en Ecuador, los beneficiarios de este proyecto macro serán las Pymes y negocios pequeños del país.

En Ecuador la demanda de servicios de Ecommerce y plataformas de medios electrónicos se concentra en las principales ciudades del país con el 51% de la totalidad de compras por medios digitales que se realizan en el país, sin embargo, el acceder al comercio por medios electrónicos es deseable para apalancar todos los pequeños y medianos negocios en el Ecuador, parte importante del proceso de comercio electrónico es la interoperabilidad con sistemas de Courier, en el presente proyecto se desarrolló un prototipo de API que solventa la interoperabilidad con esos sistemas.

Los sistemas de Ecommerce tienen la necesidad de transaccionar montos económicos con sus clientes, por esa razón el prototipo desarrollado implementó endpoints los cuales son beneficiosos para su realización, estos endpoints del API implementan la estructura de mensajes para la realización de pagos por medios electrónicos indicada por la ISO 8583:1987 que es el estándar mayormente usado por instituciones financieras en el país, de esa manera se consiguió solventar esa necesidad.

El desarrollo del prototipo implementó una arquitectura sobre tecnologías .Net Core la cual cumple con las restricciones de Rest y permite proveer parámetros de eficiencia y seguridad, además se realizó el consumo del API mediante un prototipo de aplicación móvil comprobando el funcionamiento que tiene el módulo desarrollado.

Los parámetros de eficiencia y seguridad se evaluaron en marcados en el índice APDEX y en las recomendaciones de OWASP respectivamente, en cuanto a la eficiencia se concluye que el sistema EcommerceAdmin correspondiente a las APIs alcanzaron un resultado del 0.953 según el índice APDEX, al ser bastante cercano a uno y tener una respuesta en milisegundos menor a 1000 segundos se concluye que el sistema API desarrollado es eficiente, en cuanto a la seguridad se concluye después de realizar una auditoría interna en base a las recomendaciones de OWASP y haber sometido al sistema API a una auditoría externa realizada por el club de Hacking Ethical UTN que el sistema desarrollo provee las prestaciones de seguridad necesarias para ser calificada como segura.

RECOMENDACIONES

Se recomienda para la implementación de nuevos Endpoints del Api y escalabilidad del EcommerceAdmin, el cual fue el desarrollo principal de la presente tesis tomar como referencia la arquitectura desarrollada pues implementa las restricciones Restful, además una mejora que aportaría al API se conseguiría estudiando la posibilidad de implementar Refresh Token, ya que actualizaría el JWT implementado en el API en el caso de tener inicios de sesión prolongados.

Se recomienda en la línea de investigación del proyecto macro del cual es parte este proyecto, incluir un módulo que permita realizar bussines intelligent y cubos de información, ya que sería valioso un aporte en la consecución del proyecto pues permitirá desde etapas tempranas evaluar la factibilidad económica y eficiencia de la tecnología que se está implementando.

Se recomienda además en carreras de la FICA ligadas al desarrollo de software, proveer a los estudiantes una visión amplia de las posibles tecnologías para crear Interfaces de Programación Aplicada (API), como .NET Core, Node JS, entre otras. Dentro del desarrollo del presente proyecto si bien se tenía una noción de las bases de Rest, fue necesario adquirir conocimiento más profundo para llevar al proyecto a buen término, en la actualidad el desarrollo de servicios Rest es demandado en la industria, el fortalecer estas habilidades agrega sustancial valor de un programador en el mercado laboral.

GLOSARIO DE TÉRMINOS

Ecosistema de Software. – Es la interacción de aplicaciones de sistemas informáticos que interactúan, se acoplan y cohesionan en un contexto de buenas prácticas.

Interoperabilidad. – Es la característica o capacidad que un sistema informático posee para compartir e intercambiar datos con otro u otro sistema informático.

.Net Core. – Es un framework o marco de desarrollo de software el cual está basado en código abierto además de multiplaforma por que puede alojarse y usarse en sistemas operativos Mac OS, Linux y Windows.

Xamarin. – Es una plataforma que es utiliza especialmente en el desarrollo móvil, cuenta con varias bibliotecas de controles de usuario, además es multiplaforma y permite la realización de aplicaciones iOS, tvOS, watchOS, macOS, Windows.

Google Cloud. – Es una plataforma que ofrece soluciones informáticas de hospedaje de aplicaciones, arrendamiento de infraestructura en el área de las tecnologías de la información, ofrece un saldo para realizar la prueba de sus servicios.

Cloud Azure. – Azure es una nube pública que ofreces multiples servicios de pago por uso, su propietario es Microsoft, ofrece servicios de hospedaje de aplicaciones, soluciones de red y arrendamiento de infraestructura.

Smarterasp. – Es un hosting compartido, además ofrece el servicio de vps, soporta aplicaciones .Net, .Net Core, Php y librerías del lado del servidor como angular, react y blazor posee buena reputación en el hospedaje de soluciones informáticas de pequeña y mediana escala.

Web Service. – Tecnología usada para el intercambio de datos entre aplicaciones de igual o de diferente arquitectura y tecnología, por lo general está ligado a operaciones http y maneja formatos de texto plano como JSON y XML.

Apdex. – Es un índice de rendimiento de una aplicación informática, su principal propósito es convertir mediciones en datos que reflejen la satisfacción del usuario, analiza para informar el nivel o grado en que el rendimiento cumple las expectativas del usuario.

Pruebas de Stress. – Prueba de software que simula una carga extrema en el sistema, suele ser usado para identificar cuellos de botella y realizar pruebas de simulación de planes de contingencia en caso de una caída de un sistema informático.

REFERENCIAS

ADMFactory. (2017). ISO8583 flows, fields meaning and values. Recuperado 2 de marzo de 2020, de https://www.admfactory.com/iso8583-flows-data-elements-meaning-and-values/

Álvarez, C. (2016). ¿ Que es REST ? Recuperado 1 de marzo de 2020, de https://www.arquitecturajava.com/que-es-rest/

Anderson, R. (2019). Prevent Cross-Site Request Forgery (XSRF/CSRF) attacks in ASP.NET Core | Microsoft Docs. Recuperado 9 de agosto de 2020, de https://docs.microsoft.com/en-us/aspnet/core/security/anti-request-forgery?view=aspnetcore-3.1

Apdex. (2018). *APDEX OVERVIEW*. Recuperado de https://www.apdex.org/overview.html

Appleton, B. (2017). ¿Qué es la programación extrema (XP)? | Alianza ágil. Recuperado 16 de junio de 2019, de https://www.agilealliance.org/glossary/xp/#q=~(infinite~false~filters~(postType~(~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'xp))~searchTerm~'~sort~false~sortDirection~'asc~page~1)

Armando, C. (2018). Nosotros - ASEMEC. Recuperado 27 de febrero de 2020, de http://asemec.com.ec/nosotros

Baquero, J., & Blanch, A. (2017). ¿Qué son los web services y qué tecnología usar en su desarrollo? - Blog de arsys.es. Recuperado 29 de febrero de 2020, de https://www.arsys.es/blog/programacion/diseno-web/web-services-desarrollo/

Basantes, A., Gallegos, M., & Cathy, G. V. (2016). Comercio Electrónico. *Universidad Técnica del Norte*. Recuperado de https://ebookcentral.proquest.com/lib/bibliocauladechsp/reader.action?docID=3181747&query=el+comercio

Bravo, M. (2017). Ecosistemas de desarrollo de software: lineas de automatización. Recuperado 29 de febrero de 2020, de https://jbravomontero.wordpress.com/2012/02/18/ecosistemas-de-desarrollo-de-software-lineas-de-automatizacion/

Carolina, E. (2017). Envíos por courier crecieron tras dos años | El Comercio. Recuperado 27 de febrero de 2020, de El Comercio website: https://www.elcomercio.com/actualidad/envios-courier-crecieron-ecuador-negocios.html

CECE. (2018). Comportamiento de las transacciones no presenciales en ecuador 2018. *Cece*, 18-25.

Crespo, R. (2017). Swagger: Documenta APIs REST - Cómo construir microservicios con Spring Boot (IV) -Roberto Crespo. Recuperado 1 de marzo de 2020, de http://www.robertocrespo.net/kaizen/implementar-microservicios-spring-boot-iv-documentar-apis-rest-swagger/

Daniela, A., & Navarro, A. (2018). E-Commerce: un factor fundamental para el desarrollo empresarial en el Ecuador. *REVISTA CIENTÍFICA ECOCIENCIA*, 1-18. Recuperado de http://revistas.ecotec.edu.ec/index.php/ecociencia/article/view/156

Eduardo, P. (2019). Análisis de prefactibilidad de una tienda en línea de artículso deportivos como modelo de negoscion en Quito - Ecuador 2019. (PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR). https://doi.org/.1037//0033-2909.I26.1.78

Francesc Robusta, D. G. (2015). *E-logístics* (UPC). Recuperado de https://books.google.es/books?hl=es&lr=&id=lkBpBgAAQBAJ&oi=fnd&pg=PP5&dq=e+logistic&ots=7nYqZ9Ri07&sig=aGqATUdm6NS1fJuSRAA6Ih8VbUg#v=onepage&q&f=false

GAIBOR, S. (2015). *ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE ADMINISTRACIÓN Y CONTROL PARA EMPRESAS DE COURIER DEL PAÍS.* Recuperado de http://www.dspace.uce.edu.ec/bitstream/25000/5456/1/T-UCE-0011-231.pdf

Hidalgo, A. (2016). *Análisis y propuesta de mejoras al proceso de clasificación y distribución de envíos postales en DHL Express, sucursal Guayaquil*. Universidad Politéncica Salesiana.

ISO 25010. (2019). ISO 25010. Recuperado 12 de julio de 2020, de https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

Jimenez, P. (2014). *PROPUESTA DE CUMPLIMIENTO DE LAS EXIGENCIAS DEL ESTÁNDAR ISO 8583 DE 1993 PARA LA MIGRACIÓN DESDE EL ESTÁNDAR ISO 8583 DE 1987 EN LAS INSTITUCIONES FINANCIERAS; Y, REQU*. Pontificia Universidad Catolica del Ecuador.

Joskowicz, J. (2018). *Reglas y Prácticas en eXtreme Programming*.

Kotecha, K. (2018). Pruebas de API con Postman - Aubergine Solutions - Medium. Recuperado 1 de marzo de 2020, de https://medium.com/aubergine-solutions/api-

testing-using-postman-323670c89f6d

Laudon, K. C., Kearney, A. T., & Pueyrredon, M. (2014). *E-COMMERCE EN ECUADOR: Estado actual y sus perspectivas de crecimiento.* Recuperado de http://repositorio.educacionsuperior.gob.ec/bitstream/28000/1915/1/T-SENESCYT-01149.pdf

Letelier, P., & Penadés, C. (2015). *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)*. Recuperado de www.agileuniverse.com.

Marcos, M. (2016). ¿Qué es una API y para qué sirve? Recuperado 1 de marzo de 2020, de https://www.ticbeat.com/tecnologias/que-es-una-api-para-que-sirve/

Martín, M. (2015). Logística para un ecommerce: Consejos y recomendaciones para lograr el éxito. Recuperado 29 de febrero de 2020, de https://neoattack.com/logistica-ecommerce/

MDN. (2019). Métodos de petición HTTP - HTTP | MDN. Recuperado 29 de febrero de 2020, de https://developer.mozilla.org/es/docs/Web/HTTP/Methods

Michelena, C. (2018). *Desarrollo de servicios web REST «inseguros» para auto-aprendizaje en la explotación de vulnerabilidades.*

Microsoft. (2019). Sobre .NET Core | Microsoft Docs. Recuperado 1 de marzo de 2020, de https://docs.microsoft.com/es-es/dotnet/core/about

Microsoft Docs. (2017, julio 8). The Model-View-ViewModel Pattern - Xamarin. Recuperado 26 de julio de 2020, de https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm

Microsoft Docs. (2020, mayo 2). Web API with ASP.NET Core. Recuperado 26 de julio de 2020, de https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api?view=aspnetcore-3.1&tabs=visual-studio

Miguel, P. (2019). Qué es SQL Server | OpenWebinars. Recuperado 1 de marzo de 2020, de https://openwebinars.net/blog/que-es-sql-server/

Murilo Carneiro, G., Beltrán Castañón, A., Carneiro, M., & Romani, G. (2016). Fontes de Financiamento de Organizações de Microcrédito sem fins lucrativos: um estudo comparativo entre Brasil, Chile e Peru. *Técnica administrativa*, *5*(25), 1.

Nana, C. (2018). ¿Cuales son las ventajas y desventajas en el ecommerce? Recuperado 27 de febrero de 2020, de https://canarias-digital.com/ventajas-y-desventajas-del-comercio-electronico-e-commerce/

Narula, A. (2017, agosto 17). Agile Methodology : Extreme Programming(XP) – A testerthing. Recuperado 28 de febrero de 2020, de https://atesterthing.wordpress.com/2017/08/17/agile-methodology-extreme-programmingxp/

Nielsen, J. (2016). *Usability Engineering*. Recuperado de https://books.google.com.ec/books?id=DBOowF7LqIQC&printsec=frontcover&dq=inauthor:%22Jakob+Nielsen%22&hl=es&sa=X&ved=2ahUKEwjnpp201Y_rAhUMTN8KHZpGByYQ6AEwAHoECAQQAg#v=onepage&q&f=false

Olivera, J. (2017). *JWT: Json Web Token*.

ONU-CEPAL. (2016). Agenda 2030 y los Objetivos de Desarrollo Sostenible: una oportunidad para América Latina y el Caribe. *Publicación de las Naciones Unidas*, *Mayo*, 50. https://doi.org/10.1017/CBO9781107415324.004

OWASP. (2017). *Los diez riesgos más críticos en Aplicaciones Web*. Recuperado de https://github.com/OWASP/Top10/issues

Papazoglou, M. P., Jeusfeld, M. A., Weigand, H., & Jarke, M. (2016). Distributed, interoperable workflow support for electronic commerce. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *1402*, 192-204. https://doi.org/10.1007/bfb0053411

Pozo, C. (2018). ¿Qué es un ecosistema software? | Mi espacio. Recuperado 16 de junio de 2019, de http://manuelrecena.com/blog/archives/219

Red Hat. (2017). ¿Qué es una API? Recuperado 1 de marzo de 2020, de https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces

Rocío, V. (2017). ¿Qué Es El Servicio De Courier? | E-traders. Recuperado 27 de febrero de 2020, de http://www.etraders.cl/blog/que-es-el-servicio-de-courier/

Romero, P., & Mauricio, D. (2015). Revisión de modelos de adopción de E-commerce para pymes de países en desarrollo. *Revista de investigación de sistemas e informática, RISI 9(1), 69 - 90*, *9*(1), 69-90.

Sayago, J., & Flores, E. (2019). *Análisis Comparativo entre los Estándares Orientados a Servicios Web SOAP , REST y GRAPHQL Comparative Analysis between Standards Oriented to Web Services SOAP , REST and GRAPHQL*. *9*, 10-22.

Stack Overflow. (2019). Developer Survey 2019. Recuperado 1 de marzo de 2020, de https://insights.stackoverflow.com/survey/2019#technology

Trejo, D., & Ortega, C. *Diseño de plataforma tecnológica de medio de pago local para el comercio electrónico en Ecuador, mediante la integración de infraestructuras de servicio tecnológico financiero (FINTEC)*. , (2018).

Umbarila Méndez, D. M., & Romero Rodríguez, J. F. (2015). Sistema de comercio electrónico para la distribuidora de licores orodi en ambientes web con realidad aumentada en los catálogos de los productos. En *instname:Universidad Libre*.

Valenzuela, M. (2018). E-Commerce y Logística: Más juntos de lo que piensas - LAARCOURIER. Recuperado 28 de febrero de 2020, de https://www.laarcourier.com/e-commerce-y-logstica-ms-juntos-de-lo-que-piensas

Yzquierdo, R., & González, H. (2016). Interoperabilidad entre los sistemas informáticos. *VI Encuentro Internacional de Contabilidad, Auditoría y Finanzas*, (June 2009), 1-10. https://doi.org/10.13140/RG.2.1.1503.5607

ANEXOS

Anexo A. Maquetación de la Aplicación Móvil

Fig. 23. Maquetación App Móvil
Fuente: Propia

Anexo B. Procedimiento instalación de GUI en Ubuntu 16.04 Google Cloud

1. Registrarse y crear una máquina virtual Ubuntu 16.04 en Google Cloud.

2. Acceder por SSH a la máquina virtual y ejecutar los siguientes comandos:

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install gnome-shell
sudo apt-get install ubuntu-gnome-desktop
sudo apt-get install gnome-core
sudo apt-get install gnome-panel
sudo apt-get install gnome-themes-standard
sudo apt-get install adwaita-icon-theme-full adwaita-icon-theme
gsettings get org.gnome.metacity theme
gsettings set org.gnome.metacity theme 'Adwaita'
sudo apt-get install autocutsel
sudo apt-get install tightvncserver
touch ~/.Xresources
```

3. Instalar VNC Server:

```
Vncserver
vim /home/arturocastro_q/.vnc/xstartup
```

4.Editar el archivo xstartup y cambiar su contenido, se debería ver de la siguiente manera:

```
#!/bin/sh
autocutsel -fork
xrdb $HOME/.Xresources
xsetroot -solid grey
export XKL_XMODMAP_DISABLE=1
export XDG_CURRENT_DESKTOP="GNOME-Flashback:Unity"
export XDG_MENU_PREFIX="gnome-flashback-"
unset DBUS_SESSION_BUS_ADDRESS
gnome-session --session=gnome-flashback-metacity --disable-acceleration-check --debug &
```

5. Finalizar y reiniciar VNC Server:

```
vncserver -kill :1
vncserver -geometry 1600x900
```

6. Instalar Google Cloud SDK y Configurar el acceso vinculado a la máquina virtual:

```
gcloud compute ssh miusuario@nombremaquina --project nombreproyecto --zone
zonainstalacion --ssh-flag "-L 5901:localhost:5901"
```

7. Instalar y abrir TightVNC Viewer:
a) Ingresar Host: localhost:5901
b) Ingresar el password.

Anexo C. Plan de pruebas de eficiencia Jmeter

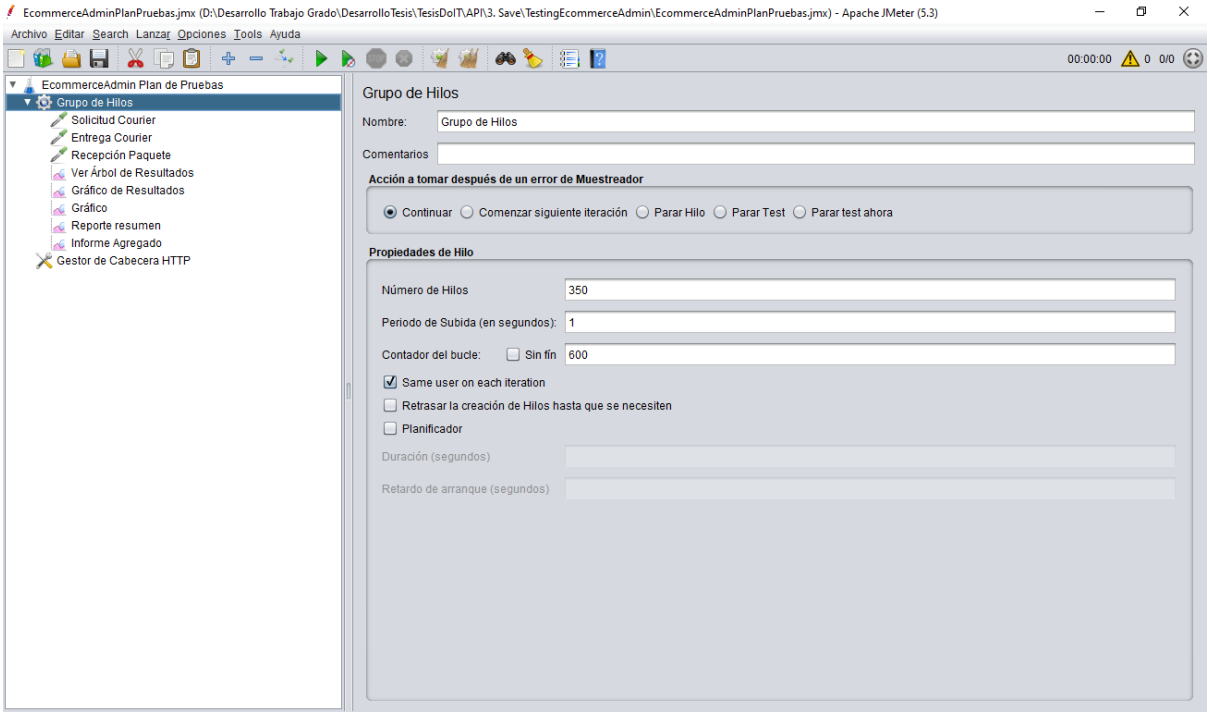


Fig. 24. Plan de pruebas Jmeter endpoints ISO 8583
Fuente: Propia

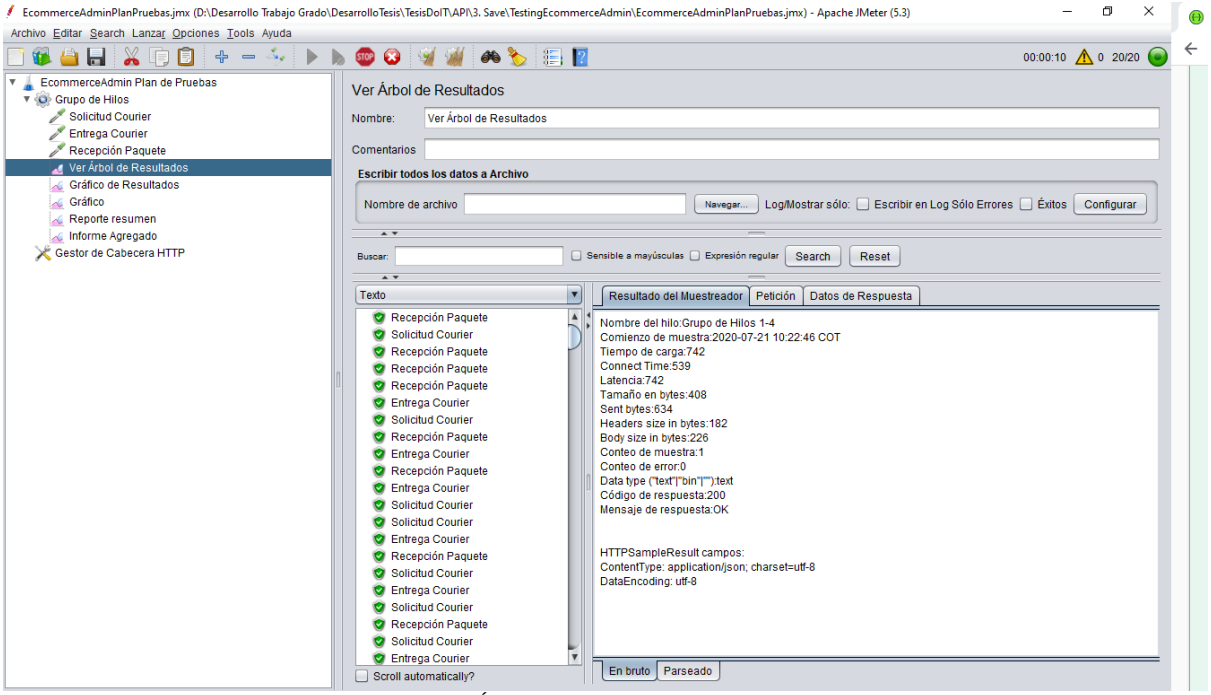
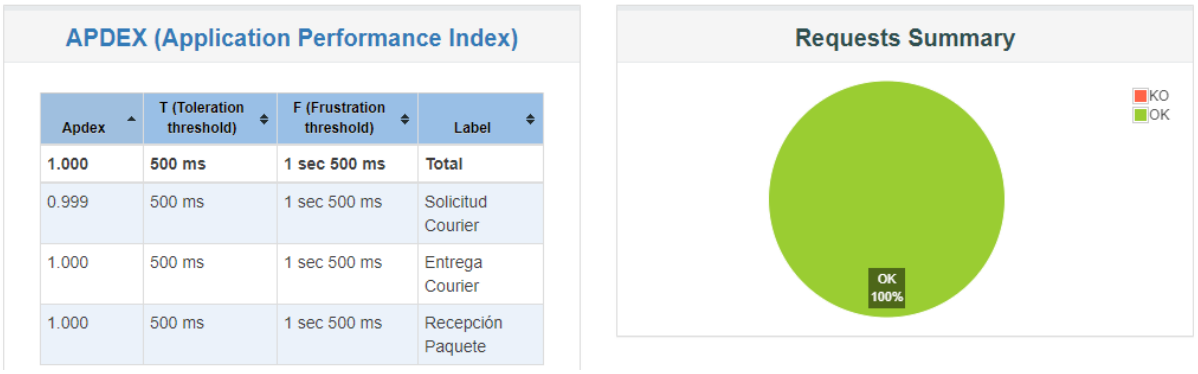


Fig. 25. Árbol de resultados pruebas Jmeter
Fuente: Propia

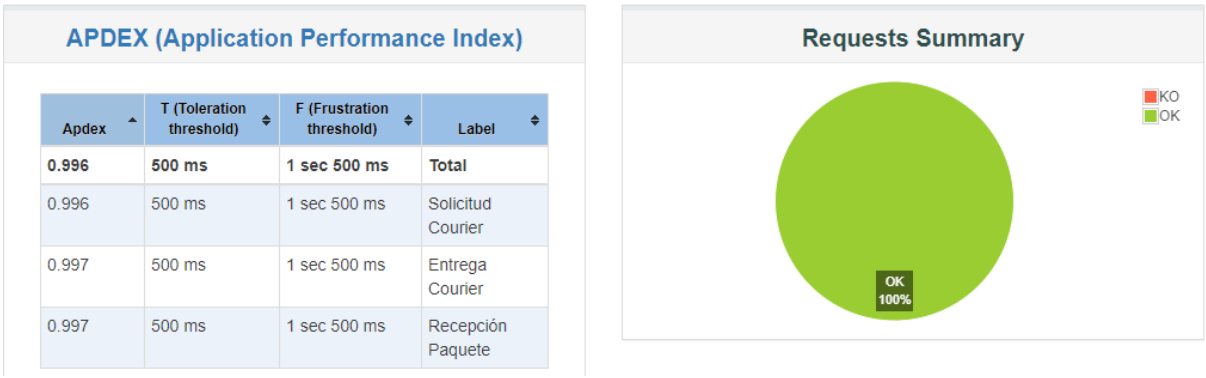
Anexo D. Resúmenes y gráficas de la prueba de estrés.



APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
1.000	**500 ms**	**1 sec 500 ms**	**Total**
0.999	500 ms	1 sec 500 ms	Solicitud Courier
1.000	500 ms	1 sec 500 ms	Entrega Courier
1.000	500 ms	1 sec 500 ms	Recepción Paquete

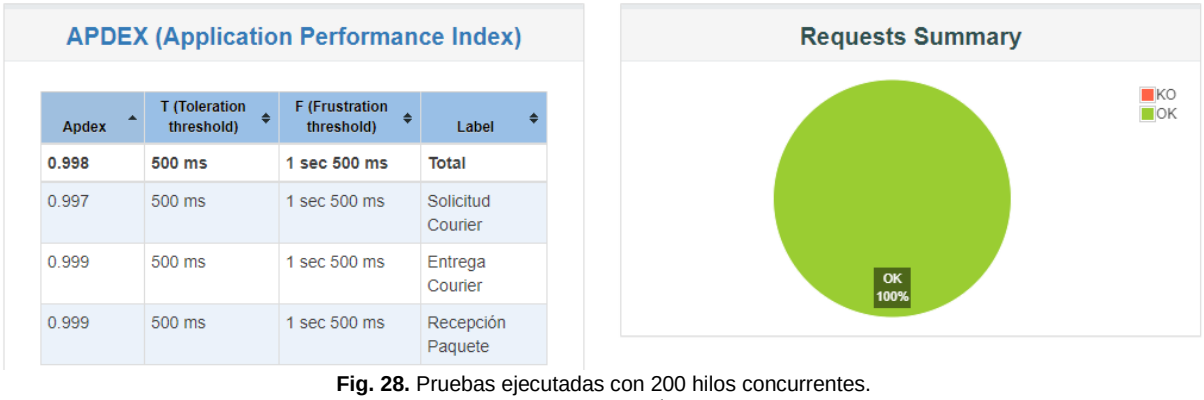
Fig. 26. Pruebas ejecutadas con 50 hilos concurrentes.
Fuente: Propia



APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.996	**500 ms**	**1 sec 500 ms**	**Total**
0.996	500 ms	1 sec 500 ms	Solicitud Courier
0.997	500 ms	1 sec 500 ms	Entrega Courier
0.997	500 ms	1 sec 500 ms	Recepción Paquete

Fig. 27. Pruebas ejecutadas con 100 hilos concurrentes.
Fuente: Propia



APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.998	**500 ms**	**1 sec 500 ms**	**Total**
0.997	500 ms	1 sec 500 ms	Solicitud Courier
0.999	500 ms	1 sec 500 ms	Entrega Courier
0.999	500 ms	1 sec 500 ms	Recepción Paquete

Fig. 28. Pruebas ejecutadas con 200 hilos concurrentes.
Fuente: Propia

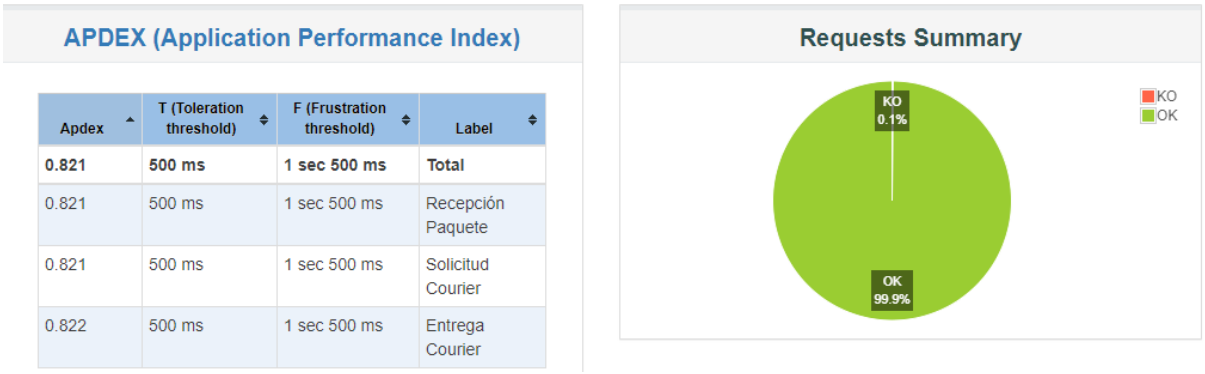


Fig. 29. Pruebas ejecutadas con 300 hilos concurrentes.
Fuente: Propia

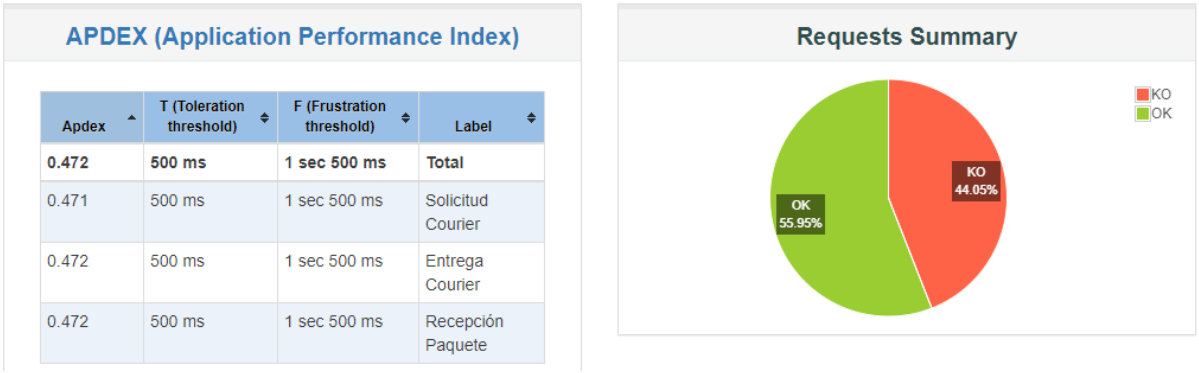


Fig. 30. Pruebas ejecutadas con 400 hilos concurrentes.
Fuente: Propia

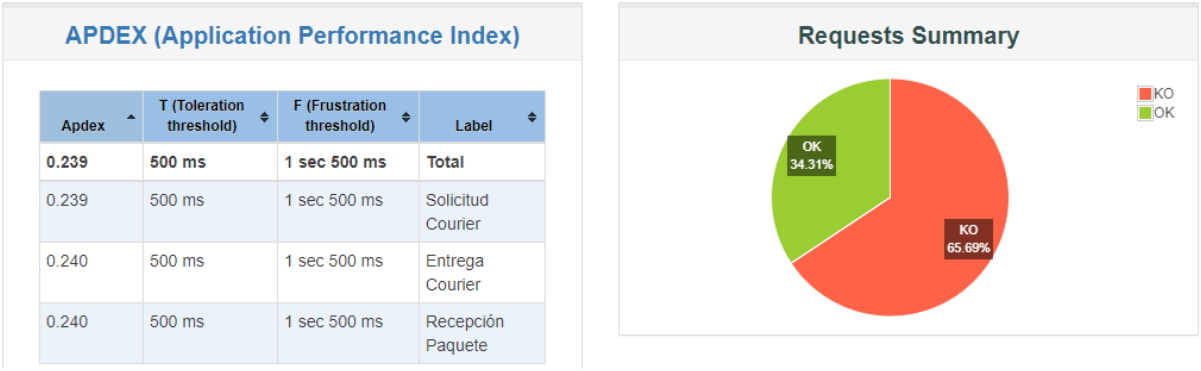


Fig. 31. Pruebas ejecutadas con 500 hilos concurrentes.
Fuente: Propia

Anexo E. Evidencia resultados pruebas de seguridad.

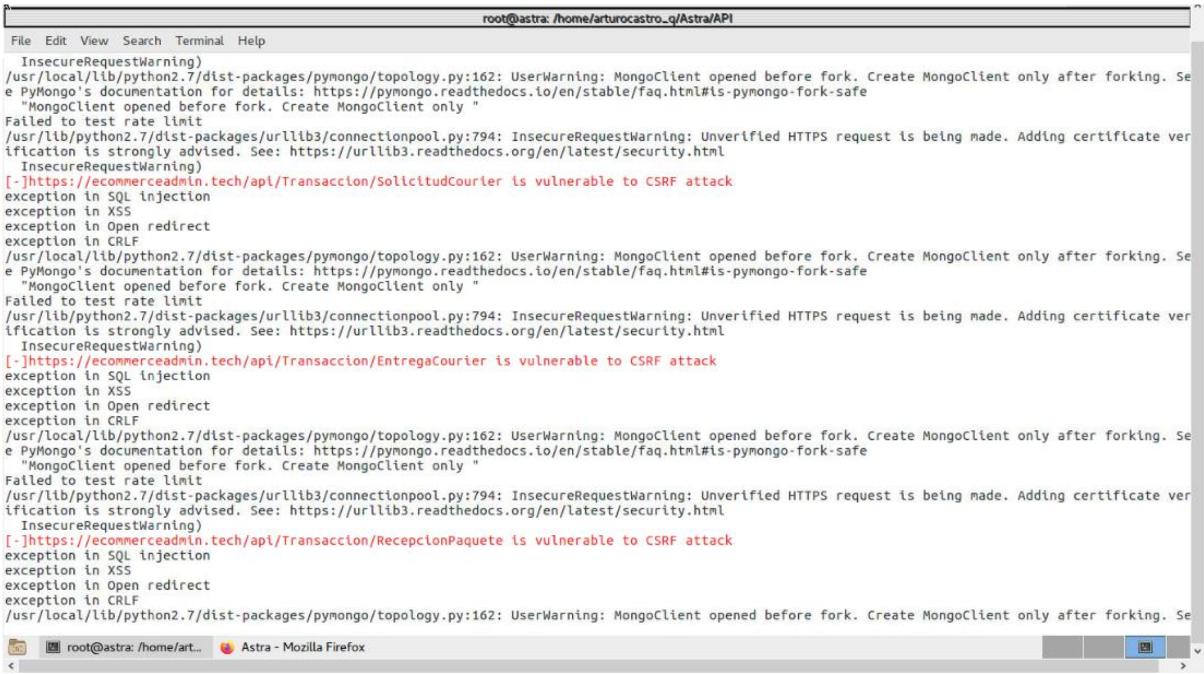


Fig. 32. Resultados Astra vulnerabilidad CSRF.
Fuente: Astra-Security en Google Cloud VM

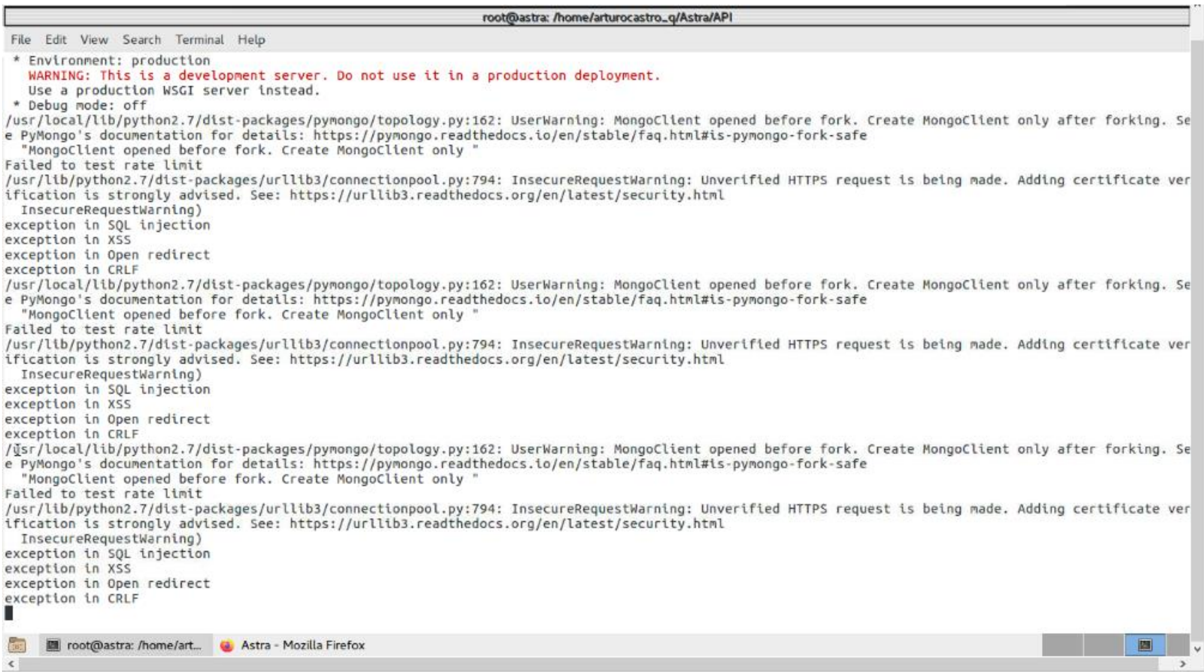


Fig. 33. Resultado Astra, no muestra vulnerabilidades
Fuente: Astra-Security en Google Cloud VM

Anexo F. Uso de recursos de la VM Ubuntu 16.04 en Google Cloud

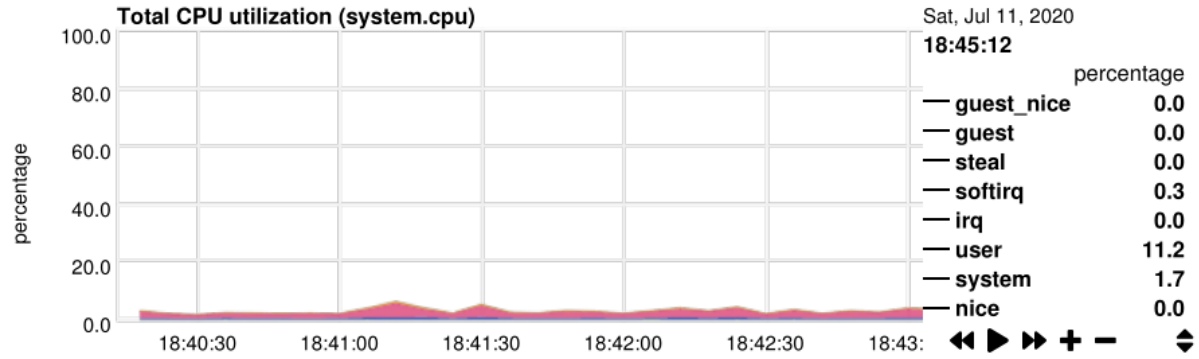


Fig. 34. Utilización CPU.
Fuente: Nerdata Dashboard Ubuntu .16.04

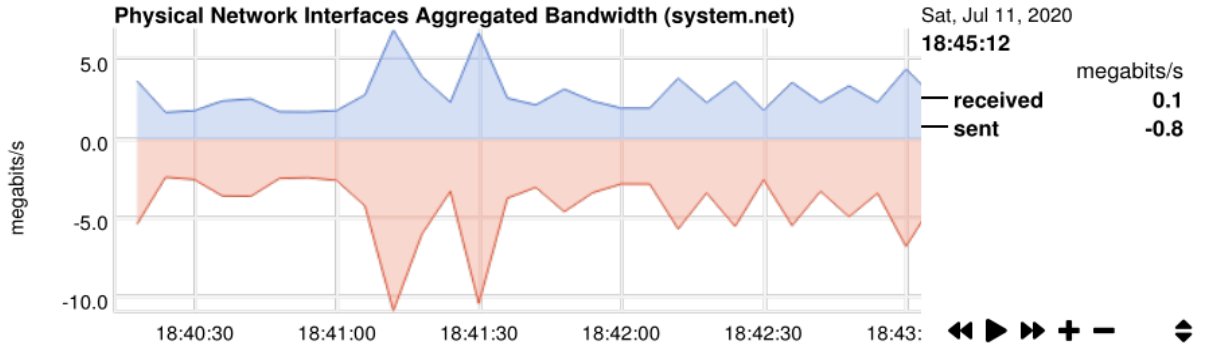


Fig. 35. Ancho de banda interfaces de red.
Fuente: Nerdata Dashboard Ubuntu .16.04

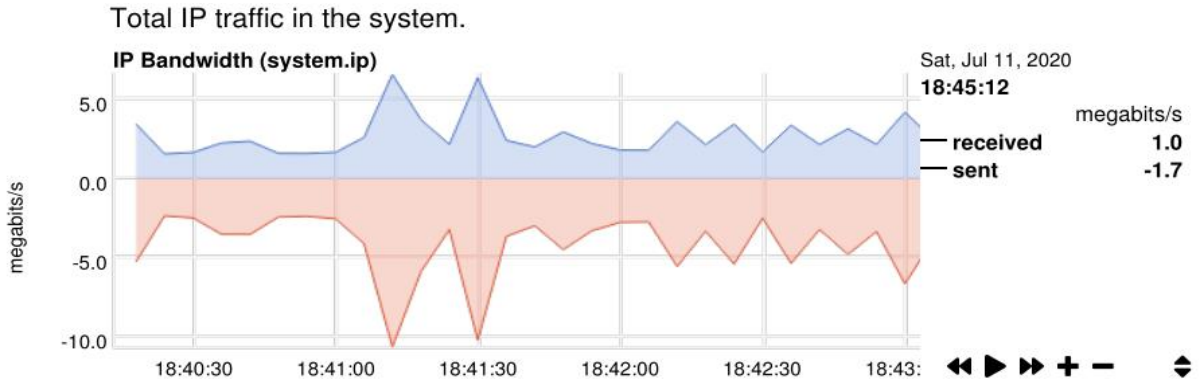


Fig. 36. Total de tráfico ip.
Fuente: Nerdata Dashboard Ubuntu .16.04

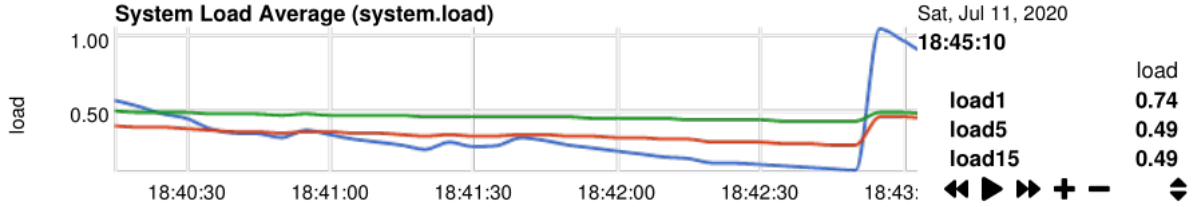


Fig. 37. Promedio de carga del sistema.
Fuente: Nerdata Dashboard Ubuntu .16.04

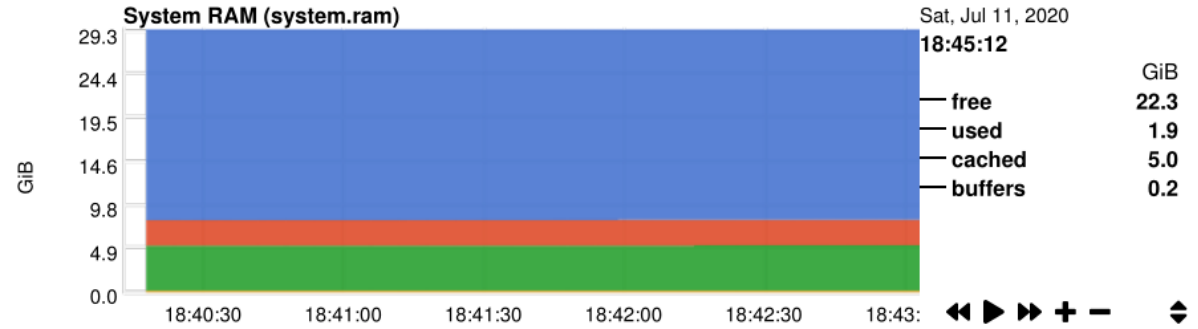


Fig. 38. Uso de memoria RAM.
Fuente: Nerdata Dashboard Ubuntu .16.04

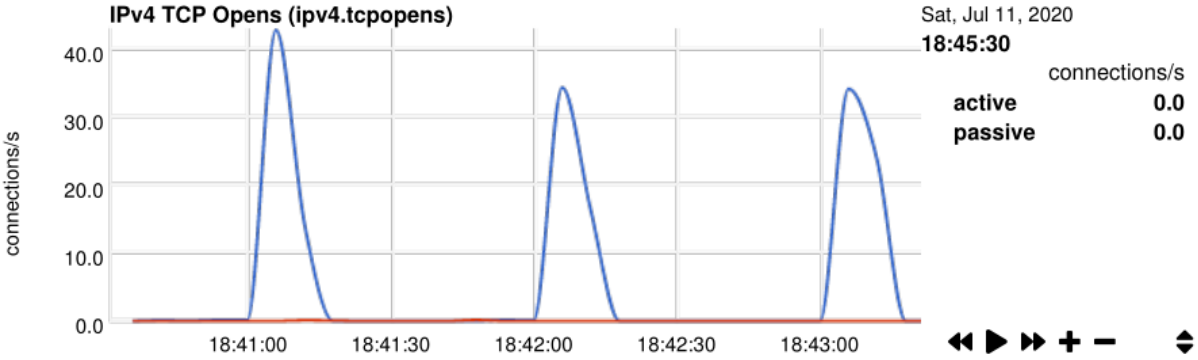


Fig. 39. Protocolo IPv4 TCP.
Fuente: Nerdata Dashboard Ubuntu .16.04

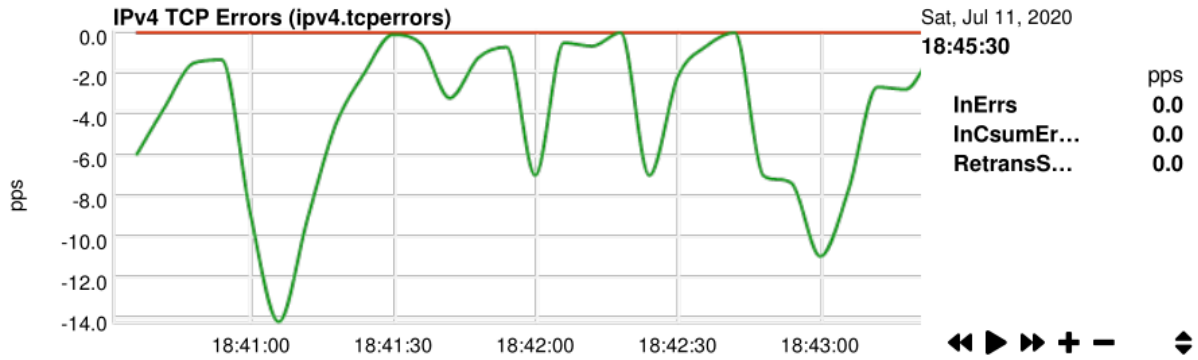


Fig. 40. Errores TCP IPv4.
Fuente: Nerdata Dashboard Ubuntu .16.04

Anexo G. Figuras de los sistemas en producción.

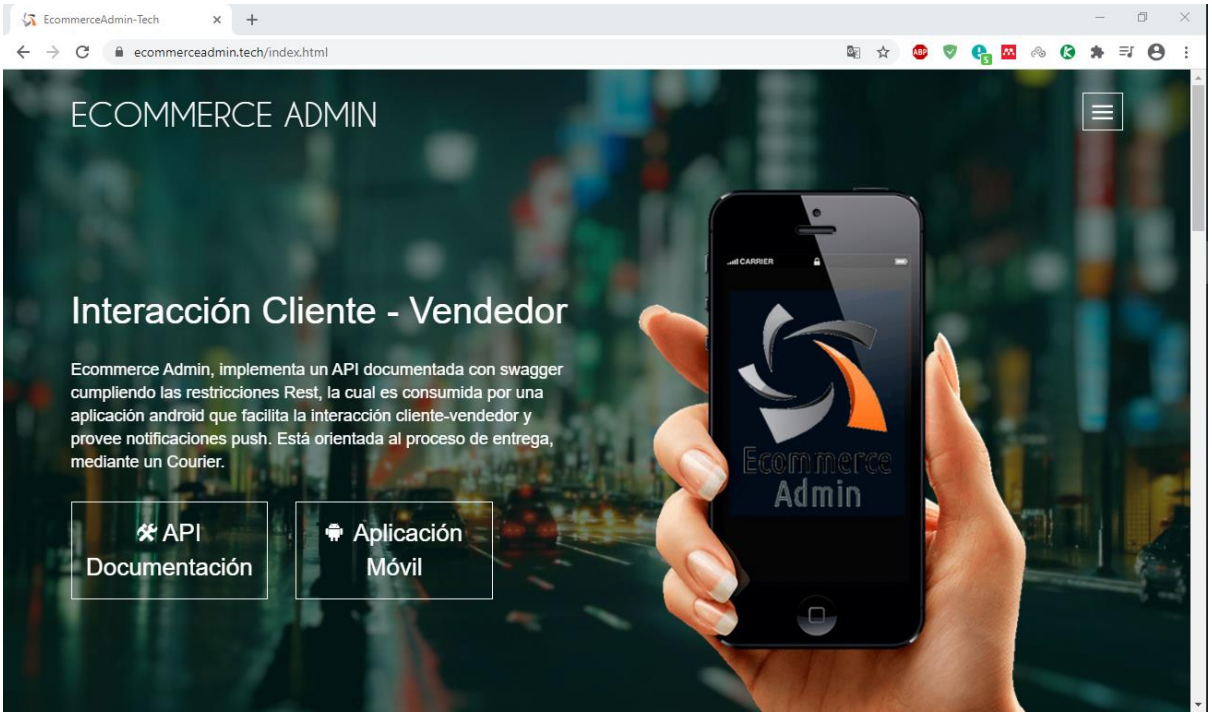


Fig. 41. Landing Page Ecommerce Admin.
Fuente: Propia

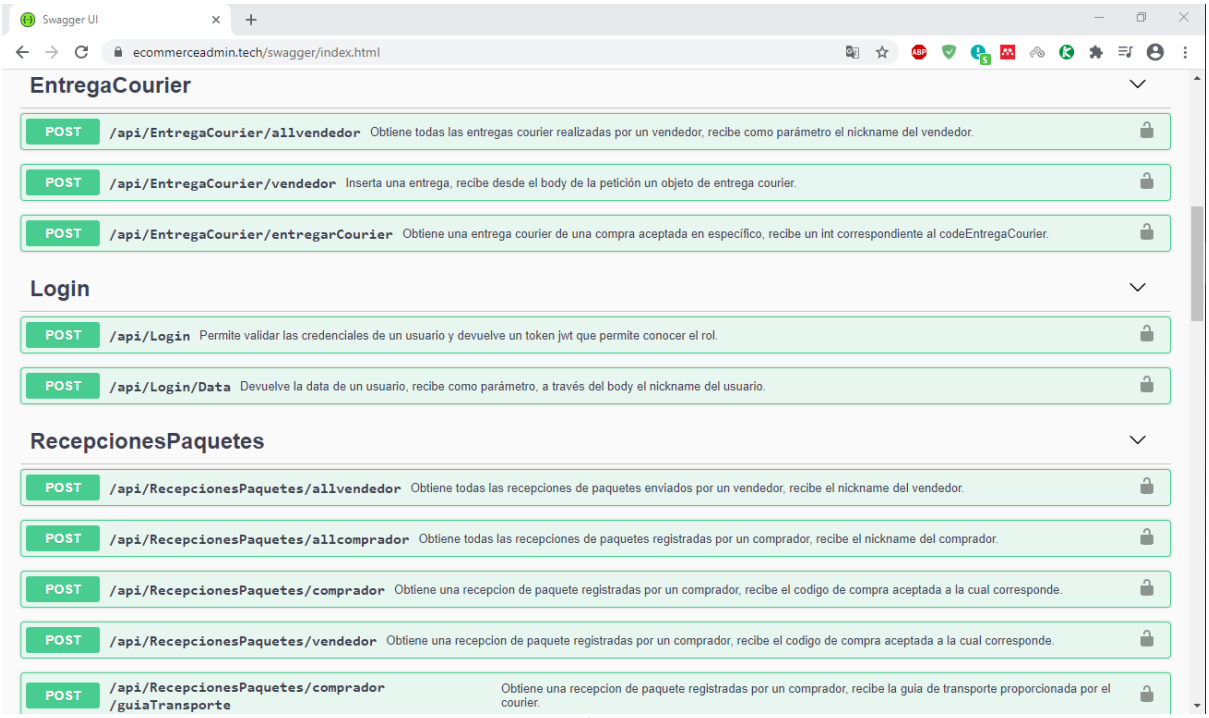


Fig. 42. UI Swagger exposición APIs Restful Ecommerce Admin.
Fuente: Propia

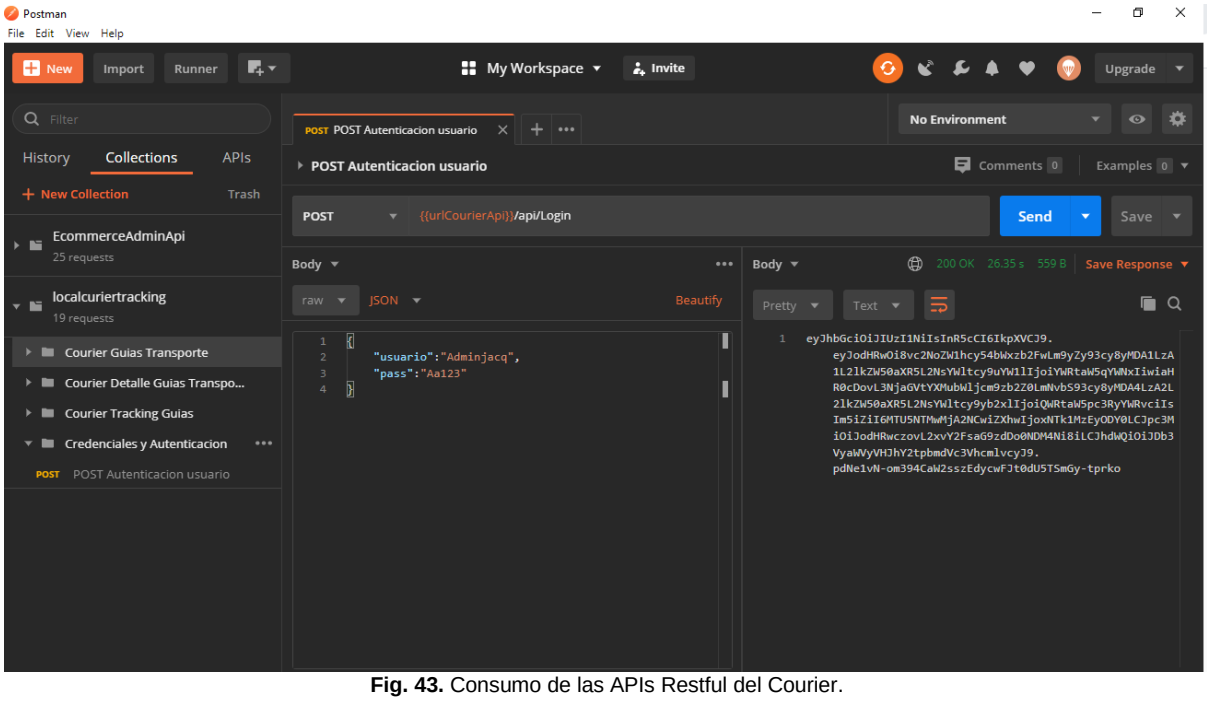


Fig. 43. Consumo de las APIs Restful del Courier.
Fuente: Propia

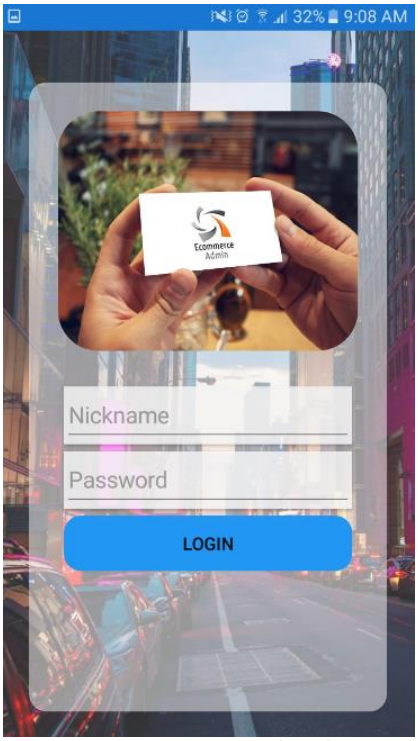


Fig. 44. Login Aplicación Móvil.
Fuente: Propia