DISEÑO DE UNA ARQUITECTURA DE MICROSERVICIOS CON CLOUD COMPUTING PARA UNA APLICACIÓN DE COMERCIO ELECTRÓNICO

Proyecto de Investigación

Juan José Rojas Pérez
CC 1005480248
Liseth Daniela Sandoval Perez
CC 1005105874

UNIDADES TECNOLÓGICAS DE SANTANDER
CIENCIAS NATURALES E INGENIERÍAS
INGENIERÍA DE SISTEMAS
BUCARAMANGA Y FECHA (16, septiembre del 2024)

Diseño de una Arquitectura de Microservicios con Cloud Computing para una Aplicación de Comercio Electrónico

Proyecto de Investigación

Juan José Rojas Pérez
CC 1005480248
Liseth Daniela Sandoval Perez
CC 1005105874

Trabajo de Grado para optar al título de
Ingeniero de Sistemas
DIRECTOR
Sergio Alexander Galvis Silva
Grupo de investigación – GRIIS

UNIDADES TECNOLÓGICAS DE SANTANDER
CIENCIAS NATURALES E INGENIERÍAS
INGENIERÍA DE SISTEMAS
BUCARAMANGA Y FECHA DE PRESENTACIÓN: 16-09-24

Nota de Aceptación

Aprobado en cumplimiento de los requisitos exigidos por las Unidades Tecnológicas de Santander, para optar al título en Ingeniero de sistemas
Según acta de comité de trabajos de grado No **31** del 26 de septiembre del 2024.
Evaluador: Diego Álvarez.

Firma del Evaluador

Sergio Alexander Galvis Silva

Firma del director

DEDICATORIA

Queremos dedicar este trabajo de investigación a nuestras familias por ser parte fundamental en cada paso de nuestra carrera universitaria. Su apoyo inquebrantable, confianza, amor y constante motivación han sido pilares indispensables para alcanzar el éxito y la culminación de esta tesis. A ellos les debemos nuestro más profundo agradecimiento por estar presentes, guiándonos y alentándonos durante todo este proceso académico. Su amor y apoyo han sido nuestro motor para lograr nuestros objetivos. Sin su constante respaldo, este logro no habría sido posible.

Gracias por ser nuestra inspiración y por creer en nosotros en todo momento.

AGRADECIMIENTOS

En primer lugar, nos gustaría expresar nuestro más profundo agradecimiento a nuestras familias por su invaluable apoyo durante el transcurso de nuestra carrera universitaria. Su paciencia, compromiso, reconocimiento de nuestros logros y su presencia en momentos de alegría, estrés, tristeza y adversidad fueron la base de nuestro crecimiento y desarrollo personal. En segundo lugar, le debo mi más sincero agradecimiento a nuestro compañero de proyecto (Juan José Rojas Pérez – Liseth Daniela Sandoval Pérez), por compartir de la mano y ser parte de uno de los momentos más importante o feliz de todo joven universitario. Gracias por todo lo vivido durante nuestra etapa universitaria y por su apoyo incondicional, dedicación y determinación en este esfuerzo conjunto para lograr el éxito y culminar este proyecto, sin su respaldo, confianza y su ánimo inquebrantable, habría sido mucho más difícil alcanzar estas metas. También le agradecemos a cada docente, al rector Omar Lengerke Pérez y a las unidades tecnológicas de Santander por recibirnos en su campus, brindarnos el tiempo y conocimiento a lo largo de nuestro recorrido académico. A nuestro director de proyecto, le agradecemos por su valioso aporte, orientación, conocimiento, compromiso y dedicación para la finalización de este trabajo.

Por último, pero no menos importante a todos aquellos que nos siguieron, formaron parte y participaron en este maravilloso proceso, especialmente a nuestros amigos o compañeros de clase. Les agradecemos por su continuo apoyo, por compartir experiencias y por ser uno de los pilares más importantes a lo largo de este viaje. Apreciamos enormemente todas sus contribuciones al éxito de nuestras carreras universitarias y la valiosa ayuda que brindaron para completar esta etapa.

TABLA DE CONTENIDO

RESUMEN EJECUTIVO 14

INTRODUCCIÓN 15

1. DESCRIPCIÓN DEL TRABAJO DE INVESTIGACIÓN 16

1.1. PLANTEAMIENTO DEL PROBLEMA 16
1.2. JUSTIFICACIÓN 18
1.3. OBJETIVOS 19
1.3.1. OBJETIVO GENERAL 19
1.3.2. OBJETIVOS ESPECÍFICOS 19
1.4. ESTADO DEL ARTE 20
1.4.1. A NIVEL LOCAL O REGIONAL 20
1.4.2. A NIVEL NACIONAL 22
1.4.3. A NIVEL INTERNACIONAL 23

2. MARCO REFERENCIAL 26

2.1. MARCO TEÓRICO 26
2.1.1. ARQUITECTURA DE SOFTWARE 33
2.1.2. MICROSERVICIOS 34
2.1.3. PLATAFORMA EN LA NUBE 38
2.1.4. CONTENEDORES 45
2.1.5. ORQUESTADORES 49
2.2. MARCO LEGAL 52
2.2.1. HABEAS DATA 52
2.2.2. CONPES 3856 DE 2016 52

2.2.3. LEY 1273 DE 2009 52
2.2.4. LEY ESTATUTARIA 1581 DE 2012 53
2.2.5. LICENCIAS AWS Y REGULACIÓN 53
2.2.6. DECRETO 1377 DE 2013 53
2.2.7. LEY 527 DE 1999 53
2.2.8. LEY 1480 DE 2011 54
2.2.9. SEGURIDAD EN MICROSERVICIOS Y CLOUD COMPUTING 54
2.3. MARCO CONCEPTUAL 57
2.3.1. CLOUD COMPUTING 57
2.3.2. MICROSERVICIOS 59
2.3.3. ARQUITECTURA DE SOFTWARE 60
2.3.4. COMERCIO ELECTRÓNICO 61
2.3.5. CONTENEDORES Y ORQUESTACIÓN 62

3. DISEÑO DE LA INVESTIGACIÓN 64

3.1. VARIABLES DE ESTUDIO 65
3.2. HIPÓTESIS 67
3.3. TÉCNICAS 68
3.4. PROCEDIMIENTO O FASES 68
3.4.1. DISEÑO DEL INSTRUMENTO DE RECOLECCIÓN: 68
3.4.2. ANÁLISIS DE DATOS: 69
3.4.3. CONCLUSIONES Y RECOMENDACIONES: 69
3.5. LIMITACIONES 70

4. DESARROLLO DEL TRABAJO DE GRADO 71

4.1. DEFINICIÓN DE REQUERIMIENTOS 72

4.1.1. REQUERIMIENTOS FUNCIONALES:..... 72
4.1.2. REQUERIMIENTOS NO FUNCIONALES: 73
4.2. DISEÑO Y ELABORACIÓN DE LA ARQUITECTURA 73
4.2.1. DIAGRAMA DE CASOS DE USO 73
4.2.2. SELECCIÓN E INTEGRACIÓN DE COMPONENTES..... 74
4.2.2.1 ALMACENAMIENTO Y ENTREGA DE CONTENIDO 74
4.2.2.2 GESTIÓN DE SEGURIDAD Y ACCESO 75
4.2.2.3 ORQUESTACIÓN DE CONTENEDORES Y MICROSERVICIOS 76
4.2.2.4 MONITOREO Y LOGÍSTICA..... 76
4.2.2.5 GESTIÓN DE DATOS Y RESPALDO 76
4.2.3. PLANTEAMIENTO DE LA ARQUITECTURA..... 77
4.3. CASOS DE USO Y EVALUACIÓN DE LA ARQUITECTURA..... 89
4.3.1. OPTIMIZACIÓN DEL ALMACENAMIENTO Y CONTENIDO: 89
4.3.2. GESTIÓN DE SERVICIOS CLAVE:..... 89
4.3.3. ESCALABILIDAD Y FLEXIBILIDAD OPERATIVA: 89
4.3.4. SEGURIDAD Y MONITOREO EFICIENTE: 90
4.3.5. RESPALDO Y RECUPERACIÓN:..... 90
4.4. ANÁLISIS DE COSTOS 90
4.4.1. SUPUESTOS PARA LA EVALUACIÓN DE COSTOS 91
4.4.2. COSTOS DETALLADOS POR SERVICIO 92
4.4.3. COSTOS POR USO..... 95

5. RESULTADOS..... 97

5.1. JUSTIFICACIÓN DE LA INVERSIÓN EN AWS..... 99
5.2. COMPARATIVAS..... 101
5.2.1. COMPARACIÓN CON OTRAS ARQUITECTURAS DE MICROSERVICIOS 101

5.2.2. COMPARACIÓN CON ALTERNATIVAS DE BAJO COSTO 101
5.2.3. LIGTHSAIL (ARQUITECTURA MONOLÍTICA) 102
5.2.4. EVENT- DRIVEN ARCHITECTURE 107

6. CONCLUSIONES 114

7. RECOMENDACIONES 116

8. REFERENCIAS BIBLIOGRÁFICAS 117

9. APÉNDICES 127

10. ANEXOS 128

LISTA DE FIGURAS

FIGURA 1. LA EVOLUCIÓN DE LAS PÁGINAS WEB. 27

FIGURA 2. CRECIMIENTO DEL E-COMMERCE EN AMÉRICA LATINA: ESTIMACIONES PARA 2023 Y 2028 28

FIGURA 3. EVOLUCIÓN DE LAS VENTAS DE E-COMMERCE DE ADIDAS.COM: 2014-2022 29

FIGURA 4. COMPARATIVA DE LA ADQUISICIÓN DE SERVICIOS DE COMPUTACIÓN EN LA NUBE EN EL SECTOR MANUFACTURERO DEL REINO UNIDO (2018 VS 2020) 32

FIGURA 5. ESTABILIDAD EN LOS MICROSERVICIOS 37

FIGURA 6. DIAGRAMA DE UN CONTENEDOR DOCKER CON SUS COMPONENTES. 46

FIGURA 7. DIAGRAMA DE UN CONTENEDOR BASADO EN LXC 48

FIGURA 8. SERVIDORES SEPARADOS EJECUTANDO UNA CARGA DE TRABAJO DIFERENTE ENTRE SÍ. 49

FIGURA 9. ARQUITECTURA DE TIPO “CLUSTER” USANDO KUBERNETES. 50

FIGURA 10. GESTIÓN DE IDENTIDADES Y ACCESOS EN UNA ARQUITECTURA DE MICROSERVICIOS. 55

FIGURA 11. MÉTODOS DE TRABAJO PARA PLATAFORMAS DE CLOUD COMPUTING. 57

FIGURA 12. ELEMENTOS CLAVE EN LA COMPUTACIÓN EN LA NUBE .. 59

FIGURA 13. EJEMPLO DE MICROSERVICIOS COMPARADO CON UN SERVICIO MONOLÍTICO. 60

FIGURA 14. ARQUITECTURA DEL CLÚSTER DE KUBERNETES CON LONGHORN 63

FIGURA 15. FLUJO DE PROCESOS LOGÍSTICOS 71

FIGURA 16. DIAGRAMA DE CASOS DE USO PLANTEADO PARA LA ARQUITECTURA. 74

FIGURA 17. DIAGRAMA PLANTEADO CON LOS RECURSOS EXPUESTOS ANTERIORMENTE..... 77

FIGURA 18. COMPONENTE “FIREWALL”..... 78

FIGURA 19. COMPONENTE “DNS” 79

FIGURA 20. COMPONENTE “FRONT-END” 80

FIGURA 21. COMPONENTE “DB FRONT-END” 82

FIGURA 22. COMPONENTE “BACK-END” 83

FIGURA 23. MÓDULO PRINCIPAL PARA BACK-END. 84

FIGURA 24. MÓDULO DE GESTIÓN DEL USUARIO. 85

FIGURA 25. MÓDULO DE ADMINISTRACIÓN. 86

FIGURA 26. COMPONENTE “ALMACENAMIENTO PARA BACK-END” 88

LISTA DE TABLAS

TABLA 1. COMPARACIÓN DE PRECIOS DE SERVICIOS DE AMAZON WEB SERVICES (AWS) (2022-2024) 41

TABLA 2. COMPARACIÓN DE PRECIOS DE SERVICIOS DE MICROSOFT AZURE (2022-2024) 43

TABLA 3. COMPARACIÓN DE PRECIOS DE SERVICIOS DE GCP (2022-2024) 44

TABLA 4. ESTUDIO DE MÉTRICAS Y PATRONES DE SEGURIDAD EN MICROSERVICIOS..... 56

TABLA 5. COSTOS MENSUALES DE AMAZON S3 92

TABLA 6. COSTOS MENSUALES DE AMAZON CLOUDFRONT 93

TABLA 7. COSTOS MENSUALES DE AMAZON EC2 93

TABLA 8. COSTOS MENSUALES DE AMAZON RDS 94

TABLA 9. COSTOS MENSUALES DE AWS WAF 94

TABLA 10. COSTOS MENSUALES DE AMAZON CLOUDWATCH 95

TABLA 11. COSTOS DE LOS COMPONENTES POR USO. 95

TABLA 12. COMPARATIVA CON LA ARQUITECTURA LIGHTSAIL 103

TABLA 13. COMPARATIVA DE ARQUITECTURAS 108

RESUMEN EJECUTIVO

El objetivo principal de este trabajo de grado fue diseñar una arquitectura de software basada en microservicios y computación en la nube (Cloud Computing) que garantizara escalabilidad, rendimiento y disponibilidad con una inversión inicial mínima. Esta solución es para aquellos comercios electrónicos que están comenzando o creciendo en este mundo y buscan mantener costos operativos bajos mientras aseguran resiliencia ante fallos, mejora continua y adaptabilidad.

Para lograr esto, se adoptó una metodología que incluyó la evaluación de documentos y tesis relacionadas a la investigación para entender su comportamiento, cómo trabajan o funcionan en conjunto estas tecnologías. Este análisis ayudó a definir los casos de uso y seleccionar las soluciones tecnológicas más adecuadas para el proyecto. Como resultado, se logró plantear una arquitectura eficiente y escalable que permitiera adaptarse rápidamente a los cambios en la demanda del mercado, al igual que ofrecer una base sólida para futuras expansiones, manteniendo un equilibrio entre costo y rendimiento.

PALABRAS CLAVE. Microservicios, Cloud Computing, comercio electrónico y arquitectura de software.

INTRODUCCIÓN

El comercio electrónico se destaca por su dinamismo y la forma en la que los emprendedores buscan constantemente maximizar sus inversiones o ampliar el alcance del negocio. Aunque la adquisición de productos y las estrategias de marketing son efectivas hasta cierto punto, el éxito a largo plazo depende crucialmente de un pilar tan importante como una infraestructura tecnológica robusta que respalde las operaciones del comercio digital. Por eso, la combinación estratégica de microservicios y Cloud Computing es una solución integral y eficiente que ayuda a las empresas a minimizar costos operativos asociados a la infraestructura tecnológica y garantiza la escalabilidad necesaria para adaptarse rápidamente a las nuevas demandas del mercado. La flexibilidad inherente de los microservicios y la capacidad de escalabilidad de la nube ofrecen una respuesta efectiva a los desafíos de un entorno empresarial altamente competitivo o en constante cambio.

En conclusión, con este enfoque estratégico las empresas pueden optimizar la eficiencia operativa, reducir costos, así como garantizar una experiencia de usuario fluida y segura. Combinación que fortalece la infraestructura tecnológica y sirve como un catalizador para el crecimiento e innovación del comercio electrónico, proporcionando así las herramientas necesarias para competir eficazmente y mantenerse relevantes en el mercado.

1. DESCRIPCIÓN DEL TRABAJO DE INVESTIGACIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA

En el mundo del comercio electrónico, muchos emprendedores buscan hacer una inversión significativa de su capital disponible tanto en la adquisición de productos para vender como en estrategias de marketing dirigidas a ampliar el alcance de las ventas hacia un público mayor. Se desea invertir en lo que un emprendedor ve más tangible de una u otra forma, ya que para que su negocio prospere debe tener los recursos mencionados anteriormente para poder alcanzar un flujo de ventas apropiado para su objetivo.

Sin embargo, esta estrategia centrada en la adquisición de productos y marketing puede resultar limitada si no se acompaña de una infraestructura tecnológica sólida que respalde las operaciones del comercio electrónico a futuro. En un entorno altamente competitivo y en constante evolución, es esencial contar con tecnologías que permitan una gestión eficiente del inventario, una experiencia de usuario fluida y segura, así como una escalabilidad sin contratiempos. La capacidad de adaptarse rápidamente a las demandas del mercado y a las preferencias cambiantes de los consumidores es fundamental para mantener una gran ventaja.

Un e-commerce puede realizar una inversión mínima en infraestructura inicial de microservicios y Cloud Computing debido a su flexibilidad en los modelos de pago, ofreciendo opciones más flexibles y aprovechando la versatilidad de la plataforma. La escalabilidad de la nube evita costos excesivos al ajustarse a las necesidades del negocio, mientras que los microservicios ofrecen desarrollo modular y ágil, reduciendo gastos de mantenimiento. Esta combinación brinda una solución rentable y eficiente para establecer una infraestructura sólida, adaptándose al

crecimiento del negocio sin sobrecargar recursos no utilizados, lo que resulta esencial en un entorno competitivo y en constante cambio. Dicho esto, surge la siguiente pregunta:

¿Cuáles son los beneficios y desafíos clave al adoptar una arquitectura en microservicios para la minimización de costos y la escalabilidad como objetivos principales?

1.2. JUSTIFICACIÓN

Al considerar la implementación de un comercio electrónico, surgen requisitos indispensables e inherentes al negocio como un carrito de compras, sección de productos y una versión móvil del sitio. Para cumplir con estos requisitos, la adopción de microservicios y computación en la nube emerge como una opción relevante por su capacidad para ofrecer flexibilidad, escalabilidad y eficiencia en la gestión de recursos, lo que no solo establece un precedente sólido para futuras expansiones, sino que también permite ampliar las capacidades de los servicios según las necesidades que surjan a medida que el negocio o emprendimiento crezca. Esta estrategia beneficiará tanto de manera temprana como a largo plazo a los e-commerce, asegurando una infraestructura robusta y adaptable que puede evolucionar con el crecimiento del negocio. Sin embargo, es esencial realizar una inversión inicial en la infraestructura tecnológica del proyecto para asegurar su viabilidad y éxito continuo.

La adopción de Cloud Computing proporciona una infraestructura versátil y bajo demanda que se ajusta dinámicamente a los cambios en la carga de trabajo, optimizando así los recursos y reduciendo los costos operativos. La escalabilidad horizontal inherente a la nube permite una distribución eficiente del trabajo, garantizando la disponibilidad y el rendimiento de la aplicación incluso en períodos de tráfico elevado. Además, debido a su naturaleza distribuida y orientada a los microservicios, facilita la integración con sistemas externos y la implementación de funcionalidades adicionales a través de una API, lo que permite una mayor flexibilidad y extensibilidad del sistema de comercio electrónico, asegurando una mayor adaptabilidad a los cambios en los requisitos del negocio y una capacidad de innovación continua.

En términos de impacto en la investigación o la educación en las UTS, esta propuesta alimenta el avance en áreas como la arquitectura de software distribuido, la ingeniería de sistemas en la nube y la gestión de servicios de TI. De esta manera, prepara a los estudiantes para enfrentar los desafíos técnicos y operativos del mundo empresarial moderno, mejorando su empleabilidad o contribuyendo al desarrollo de soluciones tecnológicas tanto innovadoras como eficientes en el ámbito del comercio electrónico y más allá.

1.3. OBJETIVOS

1.3.1. OBJETIVO GENERAL

Diseñar una arquitectura de software en microservicios y Cloud Computing para un comercio electrónico en su etapa inicial que asegure escalabilidad con una inversión mínima.

1.3.2. OBJETIVOS ESPECÍFICOS

- Definir los casos de uso dentro de un sistema de comercio electrónico mediante el análisis de las características y requisitos específicos de un negocio en su etapa inicial.
- Analizar las tecnologías de microservicios y Cloud Computing disponibles en el mercado para un comercio electrónico, evaluando sus características y comparando sus ventajas y desventajas.
- Proponer una arquitectura de software basada en microservicios y Cloud Computing, teniendo en cuenta diversos casos de uso específicos dentro del sistema de comercio electrónico.

- Evaluar la arquitectura propuesta en términos de escalabilidad, rendimiento, seguridad y adaptabilidad, sin necesidad de una implementación directa e identificando áreas de mejora para su optimización.

1.4. ESTADO DEL ARTE

La implementación de microservicios y computación en la nube tiene importancia crítica en una amplia gama de proyectos de investigación y aplicaciones empresariales contemporáneas, abarcando desde plataformas de streaming hasta las redes sociales. No obstante, su ámbito de aplicación primordial y el enfoque central del presente estudio reside en su aplicación al comercio electrónico.

1.4.1. A NIVEL LOCAL O REGIONAL

- En la tesis titulada "Propuesta de un Ecosistema tecnológico para la administración en eventos de salud pública, basado en Arquitectura Cloud Computing. Caso de estudio Santander", Meneses (2019) realizó un análisis del sistema de salud pública en Colombia, así como el uso de las Tecnologías de la Información y Comunicación (TICS) como herramienta fundamental en los procesos de salud. Además, estudió detalladamente el uso del Cloud Computing como proveedor de infraestructura informática, servicios, plataformas y aplicaciones. Esta revisión le permitió establecer bases sólidas para la propuesta de una arquitectura de un ecosistema cloud destinado al apoyo de eventos de salud pública. La propuesta se presenta como una herramienta sumamente útil en la vigilancia de sucesos o circunstancias que pueden incidir en la situación de salud de una comunidad. La orientación específica del proyecto ha dado como resultado un producto con características destacables, tales como modelos de intercambio de datos

estandarizados, provisión de información crucial para la salud de la población, herramientas que facilitan la obtención, procesamiento y consolidación de información necesaria para la toma de decisiones, reducción de esfuerzos de integración tanto para usuarios como para proveedores, y altos estándares de seguridad en el manejo de información y flujo del proceso.

- En el desarrollo de este estudio se buscó determinar el modelo en la nube más adecuado para las empresas tecnológicas en la región de Bucaramanga, con el propósito de aumentar su competitividad, optimizar costos y mejorar procesos técnicos. Se identificaron necesidades específicas de optimización y reducción de costos en estas empresas, organizando la información recopilada según criterios comunes y proponiendo modelos de computación en la nube adaptados a cada grupo de necesidades. Se encontró que, aunque las empresas tienen bases sólidas en conocimientos sobre la nube, su implementación no se ajusta completamente a lo esperado, posiblemente debido a la falta de confianza en esta tecnología, a pesar de que han obtenido beneficios notables con su uso actual. Se concluyó que tanto IaaS como SaaS son opciones viables para mejorar acciones y reducir costos en infraestructura y software, resaltando la importancia de capacitar al personal para maximizar los beneficios de estas tecnologías. (Uribe, 2019).

- Este estudio aborda el diseño y la evaluación de un prototipo de almacenamiento de datos en Cloud Computing, utilizando AWS y NextCloud, con el objetivo de promover el uso de herramientas libres en pequeñas y medianas empresas (Pymes). Se destaca la usabilidad, accesibilidad, gestión de infraestructura, seguridad y eficiencia de este enfoque. La

implementación del prototipo cumplió con los objetivos planteados, asegurando el uso de software libre y proporcionando almacenamiento en la nube con alto grado de satisfacción y funcionalidad para las empresas evaluadas. Este enfoque se presenta como una solución de bajo costo, fácil implementación y beneficioso para el sector productivo de las Pymes, contribuyendo al estado del arte en el campo de la tecnología y la gestión de datos en la nube. (Quevedo, 2018).

1.4.2. A NIVEL NACIONAL

- Se diseño, implementó y desplegó una arquitectura de software usando microservicios y Cloud Computing para un E-commerce, donde mediante el uso de los módulos llamados Lambda de AWS, se constituyeron 6 módulos que realizan la función de data center, Front-End, y demás funciones claves de software. Gracias a los servicios Lambda anteriormente mencionados, se pueden ejecutar funciones clave a nivel de desarrollo Back/Front-End, algunos de los Lambdas usan dos son: EKS, VPC, API Gateway, AWS WAF, etc. (Arango Ortiz, A. F. (2022).

- El presente estudio expone el caso de la transformación de un sistema de información de arquitectura monolítica a microservicios. Se basa en la creación, implementación y validación de un modelo de evolución, desarrollado a partir del análisis arquitectónico detallado y los requisitos técnicos y operativos de la Dirección Nacional de Admisiones de la Universidad Nacional de Colombia. Este modelo prioriza los módulos del sistema mediante análisis de dependencias y el impacto positivo previsto de la evolución, adaptando tres patrones de arquitectura como estrategia

principal de manera iterativa e incremental, sin interrumpir la operación del sistema ni su dependencia. Las conclusiones destacan que este modelo permite la transición de manera transitiva, manteniendo la disponibilidad y seguridad del sistema, y favoreciendo la flexibilidad y el análisis experto sobre el sistema durante el proceso de evolución arquitectónica. (Moreno, 2022).

- El proyecto analiza en profundidad la evolución de las tecnologías de información, especialmente el cloud computing, dentro del ámbito empresarial en Colombia. Se enfoca en detallar las ventajas y los desafíos que estas tecnologías enfrentan en el contexto local, subrayando su impacto en la innovación de productos y servicios, la optimización de procesos internos y la modernización de la infraestructura tecnológica de las empresas. Se concluye enfáticamente que las organizaciones colombianas deben dar pasos decididos hacia la adopción de tecnologías avanzadas para cosechar beneficios económicos y competir efectivamente en los mercados globales. La flexibilidad y la facilidad de implementación de estas tecnologías en la nube se destacan como factores determinantes para impulsar la innovación y la competitividad empresarial en el contexto actual.

1.4.3. A NIVEL INTERNACIONAL

- Actualmente, hay tanto soluciones como problemas que han traído las nuevas propuestas en tecnología, en estas se busca, realizar acciones que sin la tecnología de otra forma serían imposibles de realizar o reemplazar tareas que antes requerían de algún operario o persona que le hiciera frente a dicha labor. El comercio electrónico no es una excepción, para suplir la falta

de comunicación entre el cliente y el vendedor, se desarrolló un aplicativo SaaS (Software as a Servi- ce) con bases de microservicios para mejorar la brecha entre el cliente y todo el proceso que con lleva la entrega de sus pedidos, mediante un software de envío y seguimiento de pedidos, integrando un sistema de mensajería instantánea con los diversos transportistas, centralizando así la comunicación para que el usuario tenga la mejor experiencia posible en su compra (Soler, 2023).

- La popularidad creciente de la nube como solución tecnológica se debe a diversos motivos, como la eliminación de la necesidad de servidores o centros de datos locales, y la prescindencia de personal especializado en la gestión de sistemas operativos Linux. Los proyectos de comercio electrónico se benefician especialmente del bajo costo y la alta rentabilidad gracias al uso bajo demanda de los proveedores de servicios en la nube. Además, la escalabilidad, flexibilidad y el potencial de crecimiento empresarial son aspectos destacados. Sin embargo, surgen desafíos como la privacidad de los datos y la portabilidad de funciones entre servidores (Vijai y Nivetha, 2020).

- Dentro de las oportunidades que nos ofrece el Cloud Computing, se encuentra el uso de Kubernetes, una plataforma de código abierto diseñada para automatizar la implementación, escalado y gestión de aplicaciones en contenedores, Este sistema facilita la orquestación y el manejo eficiente de las cargas de trabajo en entornos de contenedores. Apoyándose en el uso de los Kubernetes, se desarrolló un aplicativo basado en Microservicios con la finalidad de apoyar a una empresa dedicada al comercio electrónico, resaltando parte de sus resultados, se encuentra una integración exitosa con

el entorno ya existente de la empresa, optimizando tiempos de carga, obteniendo una base escalable para el futuro y demás beneficios (Paredes, 2021).

2. MARCO REFERENCIAL

2.1. MARCO TEÓRICO

Para ser explícitos en el tema de investigación, es importante indagar y exponer la relevancia e importancia del contexto histórico, los estudios relevantes o previos y la evolución que ha tenido a lo largo de los años.

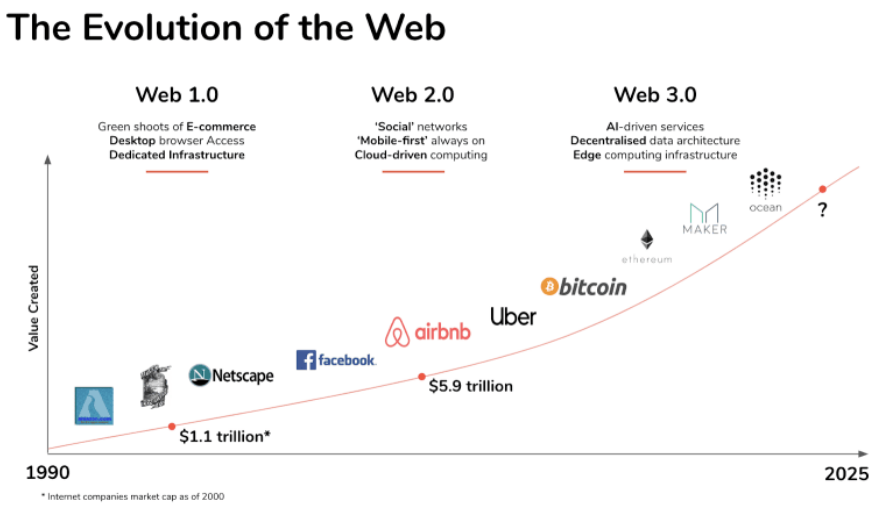
Para iniciar, hablaremos de la innovación y de las mejoras de competitividad que las empresas buscaban mediante el uso de las herramientas informáticas o tecnológicas para ayudarse con la optimización de los procesos internos, las mejoras en la rentabilidad del negocio y el aumento de visibilidad en cuanto a sus productos o servicios. Dándose así, en las últimas décadas del siglo XX, la popularización en ventas por catálogo, pues la misma ofrecía comodidad tanto para comerciantes como para compradores, ya que permitía realizar transacciones sin la necesidad de salir de casa.

Otro avance crucial en el comercio electrónico ocurrió en 1960 con la creación de una plataforma (Electronic Data Interchange, más conocida con sus siglas EDI), que facilitaba a las empresas transferir datos financieros de manera electrónica. Avance que continuó en los años 80 con la introducción de las primeras transacciones electrónicas de ventas y el Teleshopping, etapa donde la televisión jugó un papel importante al permitir la promoción y venta de productos a través de un medio accesible y masivo.

Pero sin duda el avance más grande en la historia del comercio electrónico fue la revolución del internet con la llegada de World Wide Web en los 90 y la creación de las primeras páginas web, pues esto mejoró la accesibilidad de los usuarios para conocer nuevas marcas y realizar compras. En 1995 nacen dos compañías muy

importantes como: eBay y Amazon y tres años después aparece PayPal, ficha clave para el desarrollo del pago electrónico en dichas plataformas. La expansión del comercio electrónico continuó con la llegada de la Web 2.0 en 2005, en donde las plataformas pasaron a tener un papel más interactivo y "amigable" con el consumidor.

Figura 1. La evolución de las páginas web.



Nota. Tomado de The Evolution of the Web [Imagen], Pragati Verma, 2021

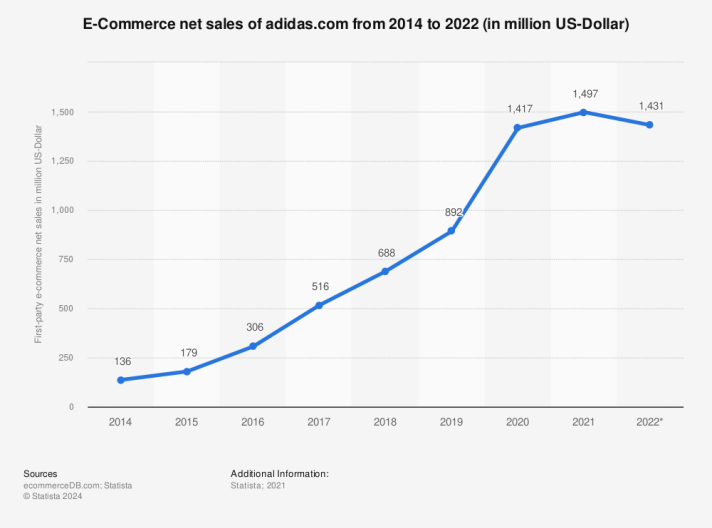
Las innovaciones de la Web 3.0, las actualizaciones de software y optimizaciones UX ayudaron a los e-commerce a convertirse en una de las plataformas más importantes de la actualidad debido a la posibilidad de acceder desde cualquier dispositivo móvil a una tienda online y la sencillez del proceso de compra significa un gran porcentaje de ventas anuales en muchas empresas.

Figura 2. Crecimiento del E-commerce en América Latina: Estimaciones para 2023 y 2028

Nota. Tomado de El boom del e-commerce latinoamericano [infografía], Statista Market Insigh ts, 2023.

En otras palabras, las grandes innovaciones de las últimas décadas, el incremento en el uso de los dispositivos móviles y la sencillez al realizar compras por Internet ha dado lugar a que los e-commerce se hayan convertido en plataformas por excelencia dentro de lo que cabe en la historia del comercio electrónico, ya que las mismas convirtieron la venta digital en algo cotidiano y cómodo tanto para compañías como para los clientes, motivo que incrementa y despierta en los usuarios el gusto y la preferencia en las compras online. Entre las herramientas que permiten la efectividad en este proceso está el correo electrónico, fundamental para la comunicación entre empresas y consumidores, y la facturación electrónica para automatizar la compra y la gestión del cobro.

Figura 3. Evolución de las Ventas de E-Commerce de adidas.com: 2014-2022



Nota: Tomado de Ventas netas de comercio electrónico de adidas.com de 2014 a 2022 (en millones de dólares estadounidenses).

Es por eso que, con el aumento de la popularidad y el uso de los comercios electrónicos, las empresas comenzaron a explorar funcionalidades más avanzadas como carritos de compras, pagos en línea y nuevas secciones. Estos avances no solo mejoraron la experiencia de usuario, sino que también incrementó significativamente la complejidad de las aplicaciones, debido a que el enfoque técnico de las arquitecturas de software se basaba en sistemas monolíticos, donde todo se estructuraba y ejecutaba en un solo servicio.

Este enfoque presentaba desafíos importantes para las empresas, ya que el mantenimiento se volvía cada vez más difícil a medida que el código base crecía, dado que cada componente dependía del funcionamiento de otro. A medida que las aplicaciones se volvían más grandes y complejas, el número de líneas de código también aumentaba, lo que ralentizaba el rendimiento del aplicativo, reducía la

compresión y dificultaba el trabajo de los desarrolladores. Asimismo, cualquier ajuste, modificación o mejora requería de una validación general de todo el programa para evitar dañar algún comportamiento o flujo. Proceso que implicaba reconstruir y desplegar la aplicación en su totalidad, lo que causaba retrasos en las entregas, ineficiencia y aumento de errores. Todo esto, en conjunto, obstaculizaba el desarrollo de una empresa en el mercado.

Es por ello que el crecimiento exponencial del internet y las nuevas tecnologías se convirtieron en un factor importante ya que los mismos han obligado a los negocios a responder y adaptarse constantemente a grandes cambios para mantener su competitividad en el mercado. En respuesta a estas demandas y necesidades, a nivel empresarial se ha producido una evolución significativa en la forma de desarrollar, diseñar, implementar y gestionar aplicaciones con el fin de poder encontrar soluciones rápidas para automatizar los procesos internos y aplicar estrategias de comercialización que garanticen el éxito de la empresa. Razón por la que las empresas pasaron de usar aplicaciones monolíticas a optar por arquitecturas de microservicios.

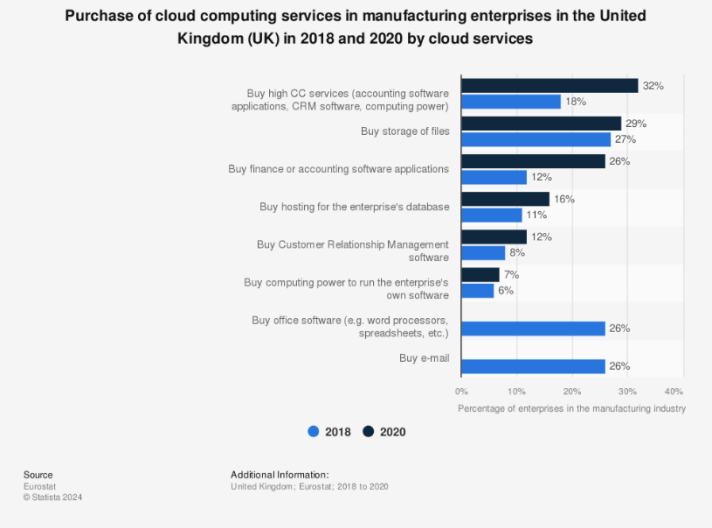
Según Newman (2019), los microservicios son una arquitectura que descompone aplicaciones grandes y complejas en componentes más pequeños y manejables. Cada componente puede desarrollarse, desplegarse y escalarse de manera independiente, permitiendo que las aplicaciones mantengan un alto rendimiento y puedan manejar muchas transacciones y usuarios de manera eficiente. De manera similar, Fowler y Lewis (2019) afirman que las arquitecturas de microservicios son especialmente eficaces para mejorar la escalabilidad y la resiliencia de las aplicaciones de comercio electrónico al permitir que diferentes partes de la aplicación evolucionen de manera independiente.

Por otro lado, como se ha explicado anteriormente y en respuesta a las constantes exigencias del mercado, surgieron nuevas necesidades como la capacidad de consultar o realizar ajustes desde distintos lugares de acceso, el uso eficiente del tiempo compartido, mayor seguridad de los datos y una escabilidad superior, entre otras. En ese sentido, la computación en la nube cobra relevancia y se convierte en un complemento esencial para la arquitectura de microservicios.

La combinación ofrece a las empresas beneficios en comunicación, crecimiento, escalabilidad y un entorno virtual que permite a los usuarios acceder a servicios y aplicaciones desde cualquier lugar. Conforme a lo que nos expone Adams (2020) en su investigación, la integración de microservicios con cloud computing potencia aún más sus beneficios al proporcionar una infraestructura escalable y flexible que puede ajustarse según la demanda. Pues, la computación en la nube ofrece recursos bajo demanda y una infraestructura gestionada que soporta la implementación continua y la entrega rápida de servicios.

El estudio de Villamizar et al. (2019) demostraron que el uso de plataformas de cloud computing como AWS, Google Cloud y Azure permite que las aplicaciones de comercio electrónico escalen automáticamente y gestionen el tráfico de manera efectiva, lo que es fundamental durante picos de demanda como los días de ventas especiales. Basado en esto, se puede afirmar que la integración de cloud computing no solo mejora la capacidad de respuesta y el rendimiento de las aplicaciones de comercio electrónico, sino que también garantiza una experiencia de usuario consistente y fiable durante periodos de alta demanda.

Figura 4. Comparativa de la Adquisición de Servicios de Computación en la Nube en el Sector Manufacturero del Reino Unido (2018 vs 2020)



Nota. Adaptado de Statista. (2021). Compra de servicios de computación en la nube en empresas manufactureras en el Reino Unido (UK) en 2018 y 2020 por parte de servicios en la nube [Gráfico].

Al momento de implementar microservicios y cloud computing, es importante considerar detenidamente los requerimientos específicos de la empresa, ya que, aunque ofrecen numerosas ventajas, también presenta desafíos. En base a la investigación de Kritikos y Plexousakis (2020), las ventajas de adoptar este enfoque incluyen la escalabilidad horizontal, la resiliencia mejorada y la capacidad de desplegar actualizaciones sin tiempos de inactividad. Sin embargo, Basiri et al. (2019) destacan que los desafíos asociados es la complejidad en la gestión de la infraestructura, la necesidad de una estrategia sólida de orquestación y la integración de servicios. Esto enfatiza otro aspecto relevante para el proyecto como lo es el uso de herramientas de orquestación como Kubernetes.

Según el estudio de Burns et al. (2020), los Kubernetes permiten automatizar la implementación, la escalabilidad y la gestión de aplicaciones en contenedores, facilitando así la adopción de microservicios en entornos de Cloud Computing. Por esta razón, es fundamental considerar las orquestaciones cuando se quiere gestionar de manera eficiente aplicaciones distribuidas y complejas para optimizar los beneficios de una arquitectura de microservicios y asegurar una integración fluida con la infraestructura en la nube.

Para concluir, varios estudios han documentado la implementación exitosa de arquitecturas de microservicios con cloud computing en aplicaciones de comercio electrónico. Por ejemplo, una investigación realizada por Taibi et al. (2020) mostró cómo una gran empresa de retail logró reducir significativamente los tiempos de despliegue y mejorar la resiliencia de su plataforma de comercio electrónico mediante la adopción de microservicios y cloud computing. Otro estudio de Sharma et al. (2021) destacó cómo la transición de una arquitectura monolítica a una basada en microservicios en la nube permitió a una empresa de tecnología escalar sus operaciones globales rápidamente y con mayor eficiencia.

Para profundizar más en el tema de investigación, es fundamental comprender lo siguiente:

2.1.1. ARQUITECTURA DE SOFTWARE

La arquitectura de software define la estructura de un sistema, comprendiendo sus componentes, sus relaciones y las propiedades que estas relaciones manifiestan según Bass et al. (2012), la arquitectura de software se puede entender como las decisiones significativas sobre la organización del sistema, que incluyen la selección

de los elementos estructurales y sus interfaces, así como su comportamiento y sus interacciones.

Esta arquitectura también involucra aspectos como la modularidad, la escalabilidad y la mantenibilidad del software. Su importancia radica en que establece los cimientos sobre los cuales se desarrollará el sistema, influyendo directamente en su calidad y en su capacidad para satisfacer los requisitos funcionales y no funcionales. En palabras de Shaw y Garlan (1996), la arquitectura proporciona un marco para la toma de decisiones y la gestión de la complejidad, permitiendo que diferentes partes del sistema se desarrollen y evolucionen de manera coherente y controlada.

El Software Engineering Institute (SEI) subraya que una arquitectura bien diseñada es fundamental no solo para desarrollar productos de calidad, sino también para modernizar sistemas heredados y evaluar las consecuencias del endeudamiento técnico. Los cursos y publicaciones del SEI, como "Software Architecture in Practice" y "Designing Software Architectures", ofrecen conocimientos esenciales sobre definición, diseño, documentación y evaluación de arquitecturas (Software Engineering Institute, 2023). Complementando esta visión, Garlan y Perry (1995), amplían el enfoque al destacar que la arquitectura de software no solo se enfoca en la estructura estática de los componentes, sino también en sus interacciones dinámicas y patrones de comunicación, cruciales para la performance y la robustez del sistema.

2.1.2. MICROSERVICIOS

Tal como lo describen Fowler y Lewis (2014), el estilo arquitectónico de microservicios implica desarrollar una única aplicación como un conjunto de pequeños servicios, cada uno ejecutándose en su propio proceso y comunicándose

con mecanismos livianos, como una API de recursos HTTP. Los servicios se enfocan en funciones empresariales y pueden desplegarse de manera independiente, gracias a herramientas de automatización que simplifican su implementación. La gestión centralizada de estos servicios es mínima, lo que permite que sean desarrollados en distintos lenguajes de programación y que utilicen variadas tecnologías de almacenamiento de datos. Esta independencia y flexibilidad permiten adaptar cada servicio a las necesidades específicas del negocio, mejorando su eficiencia y capacidad de respuesta.

Normalmente, las aplicaciones empresariales se dividen en tres componentes principales: una interfaz de usuario del lado del cliente (como páginas HTML y JavaScript ejecutadas en un navegador del dispositivo), una base de datos (que consiste en varias tablas integradas en un sistema de gestión de bases de datos, generalmente relacional), y una aplicación del lado del servidor. Esta última se encarga de manejar las solicitudes HTTP, ejecutar la lógica de dominio, acceder o actualizar datos en la base de datos y generar vistas HTML para enviar al navegador.

En una arquitectura de microservicios, estas responsabilidades se dividen en múltiples servicios pequeños e independientes, cada uno de los cuales maneja una parte específica de la funcionalidad de la aplicación. Esto significa que cada microservicio puede manejar solicitudes HTTP, ejecutar su propia lógica de dominio y acceder o actualizar sus propios datos en la base de datos. Por otro lado, en el enfoque monolítico, todas estas funciones están unificadas en un único ejecutable lógico, lo que implica que cualquier modificación en el sistema requiere la construcción y despliegue de una nueva versión completa de la aplicación del lado del servidor.

De esta manera, mientras que el enfoque de microservicios permite separar y manejar cada componente de manera independiente, el enfoque monolítico trata todas las funciones como un solo bloque, dificultando la escalabilidad y flexibilidad del sistema.

2.1.2.1 ESTRUCTURA MONOLÍTICA VS MICROSERVICIOS

Para entender mejor los beneficios de adoptar una arquitectura de microservicios, es esencial compararla con una arquitectura monolítica tradicional. En esta última, todo el proceso lógico se concentra en un único sistema, lo cual ha sido una práctica común y natural en el desarrollo de sistemas informáticos.

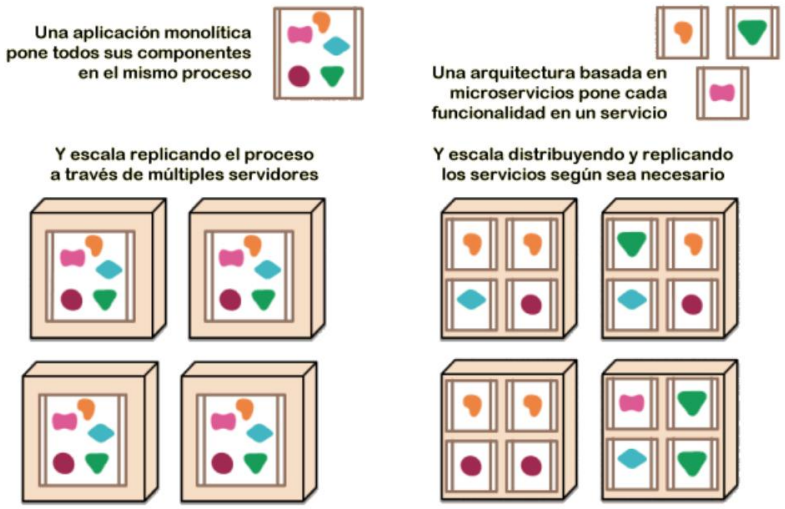
En una arquitectura monolítica, la aplicación se organiza en componentes internos como clases, funciones y espacios de nombres, aprovechando las capacidades del lenguaje de programación para estructurar el sistema de manera lógica y modular. Esta organización facilita la ejecución, simplifica las pruebas de la aplicación en el entorno de desarrollo y asegura una implementación coherente de nuevas funcionalidades. No obstante, con el tiempo pueden surgir desafíos significativos, como la rigidez y el acoplamiento inherente, que dificultan el mantenimiento, la escalabilidad y la adaptación a nuevos requisitos. Esto complica la evolución del sistema y aumenta el riesgo de errores durante las actualizaciones.

Por ello, estas aplicaciones pueden volverse complejas y difíciles de mantener, especialmente cuando se despliegan en entornos como la nube. Los ciclos de cambio están estrechamente vinculados, de modo que cualquier modificación, incluso en una pequeña parte de la aplicación, requiere la reconstrucción y despliegue de todo el sistema. Esto dificulta la implementación de cambios que deberían afectar únicamente a un módulo específico dentro del aplicativo. Además,

la escalabilidad de una aplicación monolítica requiere escalar toda la aplicación en lugar de permitir la expansión individual de partes que requieren recursos adicionales.

Estas limitaciones han impulsado la búsqueda de enfoques alternativos, como la arquitectura de microservicios, que ofrece una mayor modularidad, flexibilidad y escalabilidad en el desarrollo y mantenimiento de aplicaciones.

Figura 5. Estabilidad en los microservicios



Nota. Iranzo, V. (2018). Escalabilidad en los microservicios [Figura].

La imagen expuesta anteriormente ilustra cómo una aplicación monolítica concentra todas sus funcionalidades en un único proceso o sistema integral. Esto significa que todas las partes de la aplicación, desde la interfaz de usuario hasta la lógica de negocio y la base de datos, están integradas y operan como una entidad indivisible. En términos de escalabilidad, una aplicación monolítica tiende a replicarse en múltiples servidores para manejar cargas más pesadas o para mejorar la disponibilidad y el rendimiento.

Por otro lado, una arquitectura de microservicios brinda una solución a estas limitaciones al descomponer cada elemento funcional en servicios separados e independientes. Cada uno se enfoca en una tarea específica o un conjunto de tareas relacionadas, como la autenticación de usuarios, la gestión de productos o el procesamiento de pagos. Estos servicios se comunican entre sí a través de mecanismos ligeros, como APIs RESTful, y pueden ser desarrollados, probados, desplegados y escalados de manera autónoma.

La escalabilidad en este enfoque se logra distribuyendo los servicios a través de múltiples servidores, ajustando o replicando el número de instancias de cada servicio según las necesidades de la aplicación y la carga de trabajo en tiempo real. Así, se obtiene mayor flexibilidad, agilidad y capacidad para escalar partes específicas de la aplicación de manera más eficiente que en un entorno monolítico.

2.1.3. PLATAFORMA EN LA NUBE

Un proveedor de servicios en la nube es una empresa externa que proporciona una serie de servicios en la nube, como plataformas (PaaS), infraestructura (IaaS), aplicaciones (SaaS) o almacenamiento persistente. La característica distintiva de estos servicios es que las organizaciones solo pagan por los servicios que realmente consumen, siguiendo el modelo de pago por uso (Iranzo, 2018). Algunos beneficios que ofrecen estos proveedores a las empresas involucran la escalabilidad, flexibilidad al utilizar diferentes centros de datos, la seguridad proporcionada por la replicación de datos y la conveniencia de no tener que preocuparse por la gestión de servidores físicos. Esto posibilita que las empresas se enfoquen en sus competencias principales y en el desarrollo de sus aplicaciones, en lugar de gestionar infraestructuras complejas.

A continuación, se detallan tres de los principales proveedores de servicios en la nube:

2.1.3.1 AMAZON WEB SERVICES

De acuerdo con el artículo “Review on Amazon Web Service (AWS), Microsoft Azure & Google Cloud Platform (GCP) Services” de Mufti et al. (2021), Amazon Web Services (AWS) se presenta como uno de los pioneros en el mercado de servicios en la nube desde su lanzamiento en 2006. AWS ofrece una amplia gama de servicios informáticos como almacenamiento en la nube, servicios de bases de datos, análisis, redes, Internet de las cosas (IOT), computación móvil y servicios empresariales. Estos servicios están diseñados para cubrir diversas necesidades, proporcionando soluciones que le ayudan a las organizaciones a crecer más rápido, reducir costos y expandir sus negocios.

Complementando esta información, Mathew (2014) señala en su artículo de “Overview of Amazon Web Services” que AWS se compone de servicios en la nube que pueden ser combinarse y adaptarse a las necesidades específicas de cada negocio u organización. Se organizan en categorías como cómputo, redes, almacenamiento y distribución de contenido, bases de datos, análisis, servicios de aplicaciones, despliegue y gestión, así como móvil y aplicaciones. Siendo una de las plataformas de nube más antiguas y reconocidas en el mercado, AWS cuenta con una red global de 63 zonas de disponibilidad en todo el mundo, lo que garantiza alta disponibilidad, resiliencia, baja latencia y la capacidad de implementar soluciones robustas y escalables con alta fiabilidad.

Entre las características más relevantes que ofrece este proveedor y que se abordarán en el desarrollo del trabajo de grado, se encuentran:

- **Elastic Load Balancing (ELB):** Servicio que distribuye automáticamente el tráfico de aplicaciones entrante entre varias instancias de Amazon EC2. Mejora la tolerancia a fallos al equilibrar de manera fluida la capacidad de carga necesaria en respuesta al tráfico entrante. Además, detecta automáticamente instancias defectuosas y redirige el tráfico hacia instancias saludables hasta que las problemáticas sean restauradas. Los clientes pueden activar ELB en una sola zona de disponibilidad o en varias zonas para lograr un rendimiento de aplicación más estable y confiable.

- **AWS Lambda**: Es un servicio de cómputo que ejecuta código en respuesta a eventos y gestiona los recursos informáticos de manera automática, facilitando la creación de aplicaciones que responden rápidamente a nueva información. AWS Lambda comienza a ejecutar su código en cuestión de milisegundos tras un evento, como la carga de una imagen, una actividad en la aplicación, un clic en un sitio web o la salida de un dispositivo conectado. También es posible desarrollar nuevos servicios back-end, donde los recursos informáticos se activan automáticamente en función a solicitudes personalizadas.

- **Auto Scaling:** Ajusta automáticamente la capacidad de las instancias de Amazon EC2 hacia arriba o hacia abajo dependiendo de las condiciones definidas. Este servicio asegura que el número de instancias en uso de Amazon EC2 aumente de manera fluida durante altos picos de demanda para mantener el rendimiento y disminuirlo automáticamente durante periodos de baja demanda para minimizar costos. Auto Scaling es especialmente adecuado para aplicaciones que experimentan variabilidad horaria, diaria o semanal en el uso.

Para finalizar con este apartado, el modelo de costos de Amazon se destaca por su competitividad en comparación con otros proveedores de servicios en la nube. Los precios se basan en el pago por hora, lo que brinda una mayor flexibilidad al pagar únicamente por el tiempo real de uso de los servicios. Además, ofrece un plan gratuito con capacidad y recursos informáticos limitados, lo que resulta útil para personas y nuevas empresas que desean probarlos antes de realizar una compra. El modelo de cotización es bajo demanda y se adapta a diferentes criterios como intereses, deportes, juegos y retención. Los pagos se pueden realizar mensualmente, de acuerdo con en el uso del servicio horario y se puede calcular una estimación de costos utilizando la calculadora de precios (Kamal et al., 2020).

Tabla 1. Comparación de precios de servicios de Amazon Web Services (AWS) (2022-2024)

Year	On-Demand EC2 (m5.large)	Reserved Instances (3-year, all upfront, m5.large)	S3 Standard Storage (per GB)	RDS (db.m5.large, On-Demand)
2022	$0.096 per hour	$0.052 per hour	$0.023	$0.10 per hour
2023	$0.092 per hour	$0.050 per hour	$0.023	$0.09 per hour
2024	$0.090 per hour	$0.048 per hour	$0.023	$0.09 per hour

Nota. Amazon Web Services, Inc. (s.f.). Price reductions and cost analysis across several AWS services. AWS Partner Network Blog.

2.1.3.2 MICROSOFT AZURE

Andersson (2023) explica en su obra “Learning Microsoft Azure” el notable crecimiento de Azure como una de las principales plataformas de servicios en la nube, respaldada por Microsoft. Aunque Azure comenzó su trayectoria después de sus competidores, ha logrado consolidarse como uno de los líderes en el mercado global de la computación en la nube. La plataforma ofrece una amplia gama de servicios en la nube, que abarcan categorías como inteligencia artificial, aprendizaje automático, análisis, blockchain, contenedores de cálculo y computación sin servidor. Asimismo, cuenta con servicios para bases de datos, herramientas para desarrolladores, DevOps, gestión de identidad, integración, Internet de las Cosas (IoT), computación de borde, computación cuántica, soluciones de gestión de la nube, servicios de comunicación y medios de Azure, migración híbrida de Azure, realidad mixta, aplicaciones móviles, redes, seguridad, almacenamiento, servicios web y escritorio virtual de Windows, entre otros. Sin embargo, su singularidad radica en la oferta exclusiva de productos de Microsoft y su integración de servicios en la nube, puesto que esta integración proporciona una amplia gama de productos y servicios avanzados, lo que convierte a Azure en una solución atractiva y estratégica para diversas necesidades empresariales.

En cuanto a los precios, Microsoft Azure adopta un modelo de precios variable que se ajusta a los tipos de productos y servicios utilizados por el equipo de desarrollo. Como se detalla en el trabajo de Kamal et al. (2020), Azure ofrece opciones de pago por adelantado o mensual con tarifas calculadas por cada minuto de uso. Esto demuestra su flexibilidad en base a las necesidades del cliente, ya que el costo se determina de acuerdo con los compromisos a corto plazo y bajo demanda.

Tabla 2. Comparación de precios de servicios de Microsoft Azure (2022-2024)

Año	Servicio	Precio (por hora)
2022	Azure App Service (Standard Plan)	$0.10
2022	Azure Security Center (Standard Tier)	$0.02 por nodo/hora
2023	Azure SQL Database (Business Critical)	$1.50 por DTU
2024	Azure Cosmos DB (Multiregional)	$0.008-$0.16 por RU/s
2024	Azure App Service (Premium Plan)	$0.20

Nota. Executech. (s. f.). Azure Pricing: Compute, Networking, Storage.

2.1.3.3 GOOGLE CLOUD PLATFORM (GCP)

En su artículo "*Highlight the Features of AWS, GCP and Microsoft Azure that Have an Impact when Choosing a Cloud Service Provider*", Kamal et al. (2020) describen a Google Cloud Platform (GCP) como un conjunto de servicios de computación en la nube, que opera sobre la misma infraestructura que utiliza Google para sus propios servicios emblemáticos, como YouTube y Google Search. Este entorno permite a los usuarios realizar diversas tareas administrativas en la nube, como almacenamiento de información, cálculo de datos, análisis de datos y aprendizaje automático, mediante un sencillo registro con tarjeta de crédito o detalles bancarios.

GCP se distingue no solo por su infraestructura robusta, sino también por su plataforma de gestión de infraestructura y entornos de computación sin servidor, lo que facilita la creación de aplicaciones sin la necesidad de gestionar servidores físicos. En abril de 2008, Google lanzó App Engine como una plataforma destinada al desarrollo y alojamiento de aplicaciones web en sus centros de datos, con un

enfoque en servicios de computación distribuida. La creciente popularidad de App Engine desde noviembre de 2011 llevó a Google a introducir y expandir varios servicios en la nube dentro de esta plataforma.

Con la evolución continua de GCP, Google ha posicionado a GCP como una opción atractiva para organizaciones que buscan una plataforma escalable, segura y respaldada por la vasta experiencia de Google en tecnologías avanzadas.

Tabla 3. Comparación de precios de servicios de GCP (2022-2024)

Año	Porcentaje de Mercado
2022	10%
2023	10%
2024	10%

Nota. Google Cloud. (s. f.). Compare AWS and Azure services to Google Cloud.

Google Cloud Platform combina la infraestructura en la nube abierta de GCP con G Suite, las versiones empresariales de Android y Chrome OS, así como interfaces de programación de aplicaciones (API) orientadas a la inteligencia artificial (IA) y servicios de mapeo empresarial.

2.1.4. CONTENEDORES

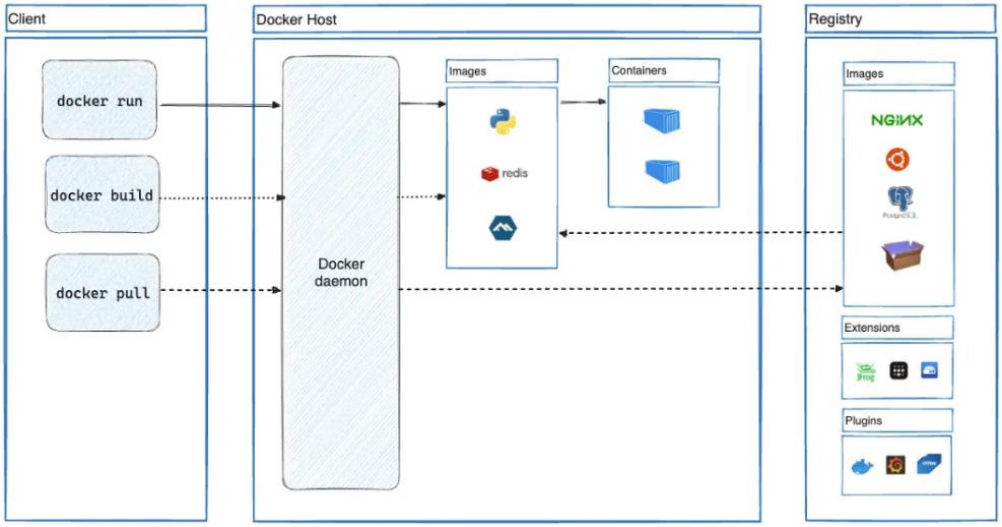
2.1.4.1 DOCKER

Docker ha transformado la manera en que se crean, despliegan y gestionan aplicaciones mediante la contenerización. De acuerdo a Docker (2020), esta tecnología simplifica el proceso al empaquetar una aplicación con todas sus dependencias en un único contenedor. De esta forma, los usuarios pueden ejecutar aplicaciones en un entorno virtual que incluye todos los elementos necesarios, como archivos, bibliotecas y otros componentes.

Esta capacidad de contenerización no solo simplifica la gestión de aplicaciones, sino que también facilita la portabilidad entre diferentes entornos. Esto se debe a que garantiza que una aplicación se ejecute de manera consistente en cualquier entorno, desde el desarrollo local hasta los entornos de prueba y producción, eliminando problemas relacionados con las diferencias entre los entornos de ejecución. Además, juega un papel vital en los procesos de DevOps, ya que facilita las compilaciones automatizadas de software y los despliegues continuos. La integración con los pipelines de CI/CD (Integración Continua y Despliegue Continuo) permite una mayor eficiencia en el ciclo de vida del desarrollo, reduciendo el tiempo necesario para la integración y el despliegue de nuevas versiones de aplicaciones.

Su funcionalidad se asemeja al montaje de bloques de Lego, en el sentido de que encapsula aplicaciones en imágenes con todas sus dependencias. Estas imágenes, que contiene tanto la aplicación como su entorno de ejecución, se distribuyen a través del Registro de Docker, lo que facilita la personalización y el intercambio de entornos sin problemas.

Figura 6. Diagrama de un contenedor Docker con sus componentes.



Nota. Docker Docs. (s. f.). Descripción general de Docker.

Una ventaja clave de Docker es su eficiencia en términos de tamaño y uso de recursos. Los contenedores e imágenes son significativamente más ligeros que las máquinas virtuales tradicionales, lo que ahorra espacio en entornos en la nube. Además, al incluir solo los datos necesarios para la aplicación, Docker optimiza el uso de los recursos de hardware y genera ahorros económicos.

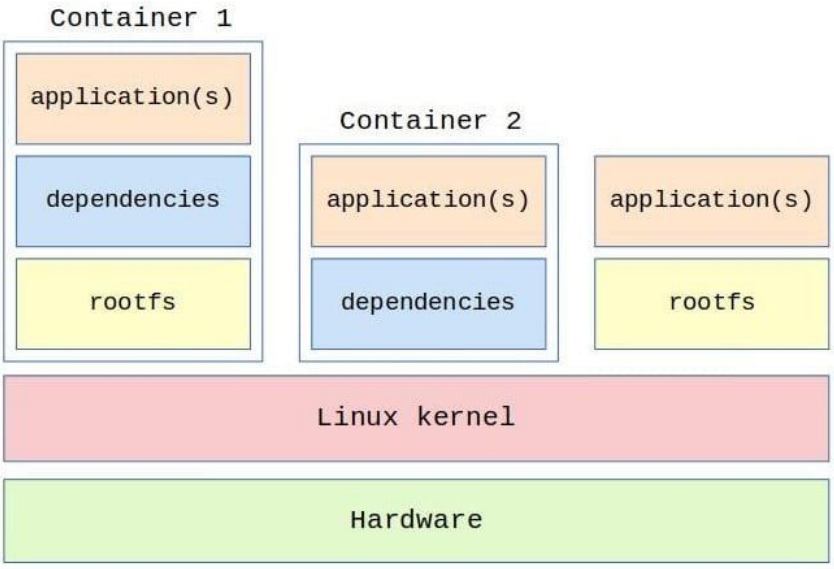
Por otra parte, facilita la gestión de aplicaciones mediante Docker files, que son scripts concisos con instrucciones o pasos para crear y lanzar imágenes. Esta característica simplifica el traslado de aplicaciones entre distintos entornos, acelera el desarrollo al permitir pruebas independientes de componentes de proyectos y respalda la escalabilidad bajo demanda de aplicaciones y servicios.

2.1.4.2 CONTENEDORES LINUX (LXC)

La historia de Linux LXC se remonta a más de una década, aunque su adopción generalizada y madurez tecnológica han sido más recientes. Al principio, la virtualización a nivel de sistema operativo, como la ofrecida por LXC, enfrentó desafíos debido a las capacidades de los hipervisores como KVM y Xen, que resolvieron muchas limitaciones del kernel de Linux en ese momento. Sin embargo, con la evolución de tecnologías como los espacios de nombres del kernel y los grupos de control (cgroups), la virtualización ligera mediante contenedores se hizo viable (Ivanov, 2017). Los espacios de nombres del kernel facilitan la compartimentación efectiva de los recursos del sistema al aislar los procesos dentro de cada contenedor, asegurando que cada uno tenga su propia vista del sistema operativo sin interferir con otros contenedores o el host. Por su parte, los cgroups proporcionan un control granular de la asignación de recursos como CPU y memoria, lo que permite administrar eficientemente el rendimiento de cada contenedor.

Antes de la implementación de estas tecnologías, la asignación de CPU y memoria a los procesos era menos eficiente, los procesos competían por los recursos disponibles y los bloqueos de E/S podían causar la pérdida de ciclos valiosos de CPU. Aunque la llegada de programadores de CPU más avanzados como el Completely Fair Scheduler (CFS), mejoró la asignación de recursos, todavía existían limitaciones en el control granular de prioridades y asignación de memoria. Con la integración de cgroups y mejoras en la programación de recursos, LXC logró un control más ligero, preciso y eficiente en comparación a la virtualización tradicional, marcando un hito en la evolución de la virtualización ligera al ofrecer una infraestructura más ágil y adaptable.

Figura 7. Diagrama de un contenedor basado en LXC

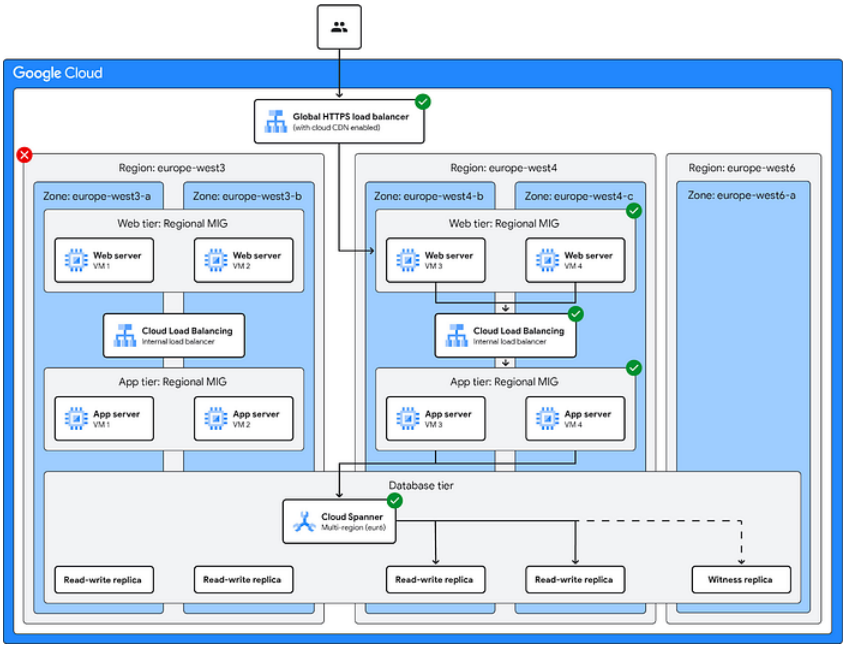


Nota. UbuntuPIT. (s. f.). Everything You Need to Know About Linux Containers (LXC).

La virtualización basada en hipervisor y los contenedores abordaron problemas al proporcionar aislamiento de recursos y entornos más ligeros para ejecutar aplicaciones. Los contenedores permiten una mayor eficiencia al compartir el mismo kernel del sistema operativo entre múltiples contenedores, lo que reduce la sobrecarga y mejora el rendimiento general del sistema.

No obstante, surgieron complicaciones al ejecutar diferentes cargas de trabajo en el mismo servidor físico, como impactos negativos en otros servicios debido a fallos o sobrecargas de recursos. La solución fue separar las aplicaciones en grupos de servidores o la combinación de enfoques de virtualización basados en hipervisor y contenedores para optimizar la utilización de recursos y evitar interrupciones en el funcionamiento de los sistemas.

Figura 8. Servidores separados ejecutando una carga de trabajo diferente entre sí.



Nota. Romin Irani. (2023, 1–15 de enero). Google Cloud Technology Nuggets.

Linux LXC y la virtualización de contenedores han evolucionado para ofrecer beneficios significativos en términos de eficiencia de recursos, aislamiento de entornos y gestión simplificada de aplicaciones, aunque también han enfrentado desafíos en la optimización del rendimiento y la prevención de impactos negativos entre cargas de trabajo diversas.

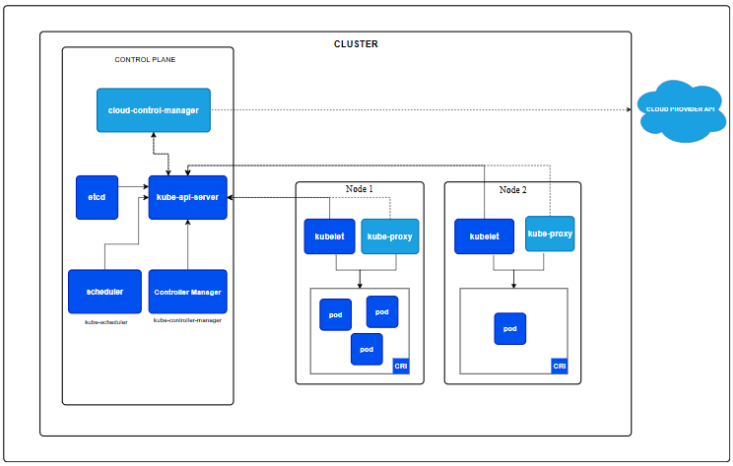
2.1.5. ORQUESTADORES

2.1.5.1 KUBERNETES

Es un sistema de orquestación de contenedores de código abierto que ha revolucionado la manera en que se despliegan, gestionan y escalan las aplicaciones

en la nube. Burns, Beda y Hightower (2019) señalan que Kubernetes fue desarrollado inicialmente por Google y luego donado a la Cloud Native Computing Foundation (CNCF), s convirtiéndose en una pieza fundamental en la infraestructura de TI moderna debido a su capacidad para automatizar la implementación, escalabilidad y operaciones de aplicaciones en contenedores.

Figura 9. Arquitectura de tipo "cluster" usando Kubernetes.



Nota. Kubernetes. (s. f.). Arquitectura de Kubernetes.

Desde su lanzamiento en 2014, ha evolucionado considerablemente. La versión 1.0 se lanzó en 2015 y marcó el inicio de su adopción masiva, la cual desde el 2019, ha mejorado significativamente en seguridad, escalabilidad y facilidad de uso. Estas mejoras se reflejan en las versiones posteriores, las cuales han integrado características avanzadas como el soporte para múltiples arquitecturas de nodos y la inclusión de herramientas para la gestión de políticas de seguridad (Hightower et al., 2020).

Su arquitectura se basa en un modelo maestro-esclavo, donde el nodo maestro gestiona el clúster y los nodos de trabajo ejecutan las aplicaciones en contenedores.

Los componentes principales de un clúster de Kubernetes incluyen (Burns, Beda, & Hightower, 2019; Goasguen & Jocha, 2019):

- ***API Server***: Punto de entrada para todas las operaciones administrativas en el clúster.
- ***etcd***: Almacén de datos distribuido que almacena toda la información del clúster.
- ***Scheduler***: Asigna contenedores a los nodos de trabajo.
- ***Controller Manager***: Ejecuta controladores que gestionan las operaciones del clúster.
- ***Kubelet***: Agente que corre en cada nodo de trabajo, asegurando que los contenedores están corriendo en los pods.
- ***Kube Proxy***: Mantiene las reglas de red en los nodos de trabajo.

Entre las principales ventajas de Kubernetes se destacan la automatización del despliegue, escalado y gestión de aplicaciones, la escalabilidad eficiente de aplicaciones, la portabilidad o el movimiento de aplicaciones entre diferentes entornos de ejecución y la simplificación de la gestión de configuración y secretos de las aplicaciones (Burns et al., 2021). A pesar de sus beneficios, enfrenta desafíos relacionados con la complejidad de su configuración y gestión. Sin embargo, las mejoras continuas y el apoyo de la comunidad han mitigado estos problemas a través de herramientas complementarias como Helm para la gestión de paquetes y operadores para la automatización de tareas complejas (Hightower et al., 2020).

2.2. MARCO LEGAL

2.2.1. HABEAS DATA

El Habeas Data es una garantía constitucional que protege el derecho de las personas a conocer, actualizar y corregir sus datos personales de archivos o bases de datos en entidades públicas y privadas. Esta protección es esencial para cualquier plataforma de comercio electrónico que valora la privacidad y la seguridad de la información de sus usuarios. El usuario puede ejercer sus derechos legales gracias a la implementación de estas medidas y acciones, lo que aumenta la confianza en la plataforma.

2.2.2. CONPES 3856 DE 2016

Establece pautas para la protección de la información y la ciberseguridad, que incluyen temas relacionados con el comercio electrónico en términos de seguridad de datos y transacciones. Se centra en establecer normas para proteger los datos personales y controlar las actividades comerciales en línea, con el fin de promover la confianza y la seguridad en el comercio electrónico en Colombia.

2.2.3. LEY 1273 DE 2009

La Ley 1273 de 2009 penaliza delitos informáticos relacionados con el acceso, uso, daño, interceptación, hurto y manejo indebido de información digital. Por lo tanto, implementar estas medidas de seguridad en un E-Commerce evita ataques y accesos no autorizados para proteger los datos de los clientes y la integridad del sistema.

2.2.4. LEY ESTATUTARIA 1581 DE 2012

Normativa que establece el proceso general de protección de datos personales, es decir, reconoce y protege el derecho de toda persona a conocer, actualizar y rectificar información relacionada con la recolección, almacenamiento, uso, circulación y cancelación de datos personales en base de datos o archivos tanto de entidades públicas y privadas.

2.2.5. LICENCIAS AWS Y REGULACIÓN

Al utilizar servicios de Cloud Computing como AWS, es fundamental asegurarse que las licencias y los acuerdos de nivel de servicio (SLA) cumplan con las leyes locales y las mejores prácticas internacionales. Por ejemplo, AWS ofrece certificaciones y auditorías de seguridad que pueden ayudar a demostrar que la infraestructura cumple con los requisitos legales y de seguridad mencionados anteriormente.

2.2.6. DECRETO 1377 DE 2013

Este decreto complementa la Ley 1581 de 2012 y establece pautas adicionales para la protección de datos personales. Es necesario asegurar que los datos personales de un E-commerce estén protegidos y que se cumpla con los derechos de los titulares de los datos, incluyendo el acceso, rectificación, y supresión de información.

2.2.7. LEY 527 DE 1999

El acceso a los mensajes de datos, las firmas digitales, los contratos electrónicos y entidades de certificación están reguladas por la Ley 527 de 1999, la cual establece las bases para el comercio electrónico. Esta ley garantiza que las transacciones y

comunicaciones realizadas sean legales y seguras, asegurando la integridad y disponibilidad de los datos.

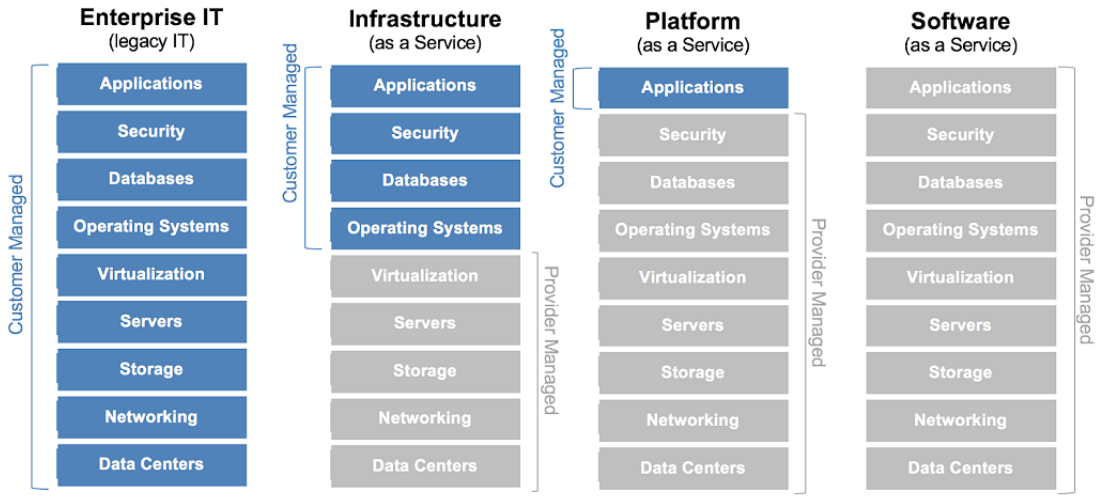
2.2.8. LEY 1480 DE 2011

Regula la protección de los derechos de los consumidores en las transacciones electrónicas, centrándose en la protección de datos personales y la transparencia en el manejo de la información. Sus objetivos son proteger, promover y garantizar la efectividad y el libre ejercicio de los derechos de los consumidores, así como el respeto a su dignidad e intereses económicos. Ley esencial para los comercios electrónicos porque le brinda a los usuarios seguridad y confianza, los cuales son elementos clave para el éxito de las plataformas en línea.

2.2.9. SEGURIDAD EN MICROSERVICIOS Y CLOUD COMPUTING

Di Francesco et al. (2020), enfatizan que la seguridad en arquitecturas de microservicios debe abordar aspectos como la autenticación, autorización, encriptación y monitoreo de servicios. Los microservicios deben estar diseñados para ser seguros por defecto, aplicando prácticas de seguridad como la defensa en profundidad y el principio de menor privilegio, con el fin de que cada componente de la arquitectura opere bajo estrictas condiciones de seguridad, mitigando riesgos y protegiendo la integridad del sistema.

Figura 10. Gestión de identidades y accesos en una arquitectura de microservicios.



Nota. Cloud Information Center. (s. f.). Cloud Security.

En cuanto a la seguridad en la nube, es importante que esta incluya la protección de datos, la gestión de identidades y accesos, así como el cumplimento de normativas y estándares de la industria. Tal y como indica Jamsa (2020) en su estudio, las soluciones en la nube deben integrar medidas de seguridad robustas para proteger tanto las aplicaciones como los datos sensibles. Esta integración no solo asegura la confidencialidad de la información, sino que también ayuda a mitigar riesgos y a responder a amenazas emergentes, promoviendo un entorno seguro y confiable para las partes interesadas.

Además, las aplicaciones de comercio electrónico deben cumplir con las regulaciones locales e internacionales relacionadas con la privacidad y protección de datos, tales como el GDPR en Europa y la CCPA en California (Gartner, 2020).

Cumplir con estas normativas es fundamental para proteger la información personal de los usuarios y mantener la confianza en el mercado global.

Tabla 4. Estudio de Métricas y Patrones de Seguridad en Microservicios

Nombre del Artículo	Sector	Características
A Survey on Systems Metrics	Nominal, Ordinal, Interval, Ratio	Las miden por escalas: Nominal, Ordinaria, Intervalo.
Structural Coupling for Microservices	Microservices	La mayoría de estas métricas puede ser adecuada para microservicios guiada por el acoplamiento y cohesión.
Microservices Metrics	Microservices	Estas métricas se basan en mediciones manuales con base a un conjunto de propiedades y no están validadas empíricamente.
Quality metrics for web application development	Web	Métricas orientadas a aplicaciones web para proveer el desarrollo a empresas y desarrolladores, basadas en encuestas y revisiones bibliográficas.
Evaluation of Microservice Architectures: A Metric and Tool-Based Approach	Microservices	Realizan derivaciones de principios y métricas para realizar evaluaciones, utilizando el enfoque de ingeniería inversa y datos de comunicación dinámica.

Automated Software Architecture Security Risk Analysis	Object-Oriented Designs	Utilizando el lenguaje de restricción de objetos (OCL), analiza un sistema para localizar coincidencias para los escenarios de ataque.
Signatures for Object-Oriented Designs	Object-Oriented Designs	Capturan los detalles de seguridad usando UML y SecDSVL.
Security Metrics for Object-Oriented Designs	Object-Oriented Designs	Métricas orientadas a objetos basadas en las propiedades de diseño de calidad del software.

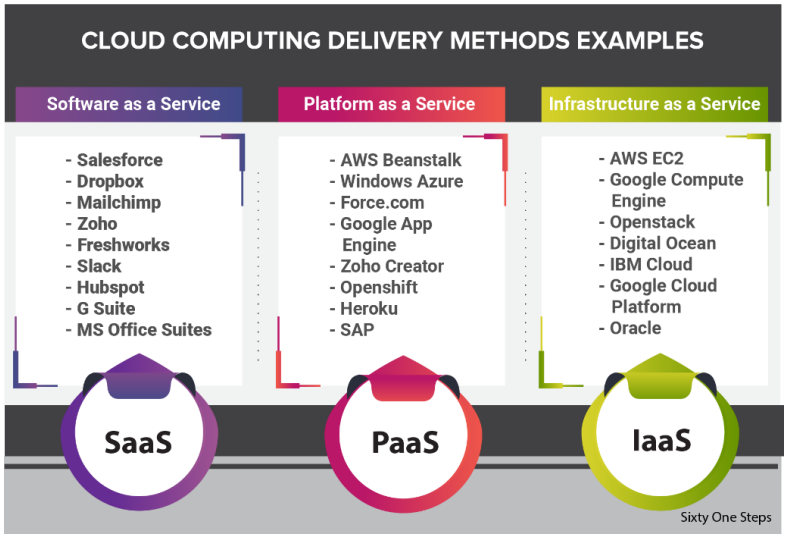
Nota. Penagos Sanchez, J. V. (2023). Estudio de Métricas y Patrones de Seguridad en Microservicios.

2.3. MARCO CONCEPTUAL

2.3.1. CLOUD COMPUTING

La computación en la nube es la entrega de recursos informáticos, como servidores, almacenamiento, bases de datos, redes, software y análisis, a través de Internet ("la nube") para ofrecer una innovación más rápida, recursos flexibles y economías de escala, como señala Marinescu (2020) en su investigación. En otras palabras, se trata de una tecnología que facilita el hospedaje, el acceso remoto o la conectividad desde cualquier lugar o dispositivo a diversos softwares, aplicaciones, archivos, entre otros.

Figura 11. Métodos de trabajo para plataformas de Cloud Computing.

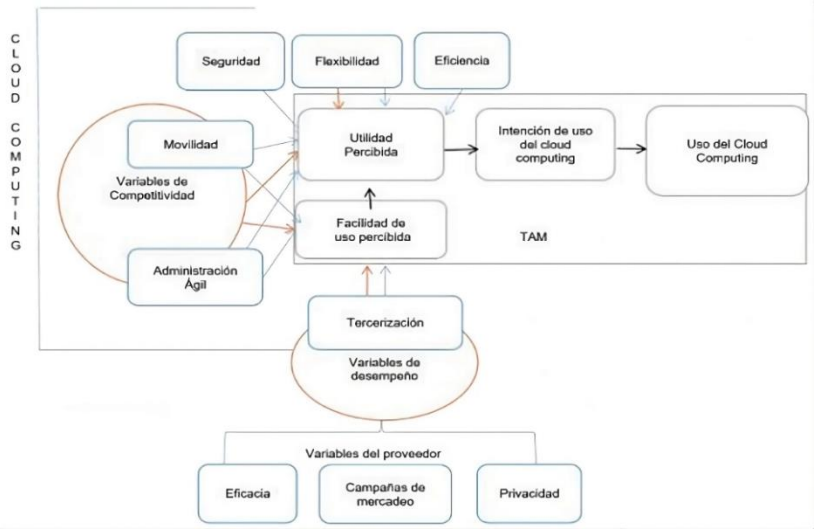


Nota. Sixty One Steps. (s. f.). What is Cloud Technology? Know More about - SaaS, PaaS, IaaS.

Este modelo brinda a las empresas la posibilidad de acceder a una infraestructura tecnológica avanzada sin la necesidad de realizar grandes inversiones iniciales, facilitando así la adaptación y el crecimiento continuo del negocio. El concepto es importante para nuestro proyecto porque la computación en la nube proporciona una infraestructura escalable y flexible, ideal para el despliegue de microservicios. Como indican Sill et al. (2021), ofrece capacidades de autoescalado y gestión automatizada, lo que facilita la administración de aplicaciones complejas de comercio electrónico. Por lo tanto, la combinación de microservicios y Cloud Computing asegura una arquitectura robusta y adaptable a el crecimiento del negocio y las nuevas demandas del mercado.

Los principales modelos de servicio en Cloud Computing son IaaS (Infraestructura como Servicio), PaaS (Plataforma como Servicio) y SaaS (Software como Servicio). Los modelos de despliegue incluyen nube pública, privada, híbrida y multicloud (Garg & Versteeg, 2021).

Figura 12. Elementos Clave en la Computación en la Nube

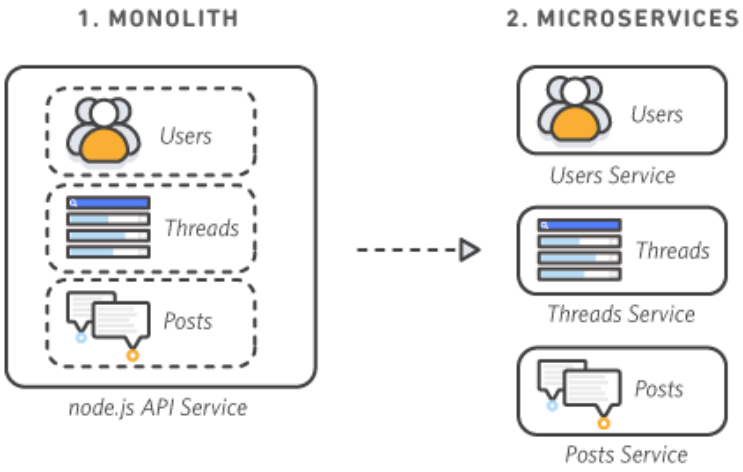


Nota. Patiño, J., y Valencia, A. (2019). Propuesta de modelo estratégico de adopción de la computación en la nube en las PYME [Figura].

2.3.2. MICROSERVICIOS

Son un enfoque arquitectónico diseñado para construir y gestionar aplicaciones mediante la división de un conjunto de servicios autónomos o pequeños que trabajan en conjunto para llevar a cabo tareas específicas, facilitando una mayor flexibilidad y control sobre cada componente del sistema. Tal y como expone Dragoni et al. (2020), esta arquitectura de software descompone una aplicación en un conjunto de servicios pequeños e independientes, cada uno ejecutando en su propio proceso y comunicándose con otros servicios a través de interfaces bien definidas, típicamente API HTTP.

Figura 13. Ejemplo de microservicios comparado con un servicio monolítico.



Nota. Amazon. (s. f.). Arquitectura monolítica en comparación con la arquitectura de microservicios.

Su importancia para los comercios electrónicos radica por su capacidad para mejorar la escalabilidad, la resiliencia y la capacidad de implementación continua. Los microservicios permiten que diferentes equipos desarrollen, desplieguen y escalen servicios de manera independiente, lo cual es crucial para manejar picos de tráfico y actualizar funcionalidades sin afectar a toda la aplicación (Newman, 2021).

2.3.3. ARQUITECTURA DE SOFTWARE

Se basa en patrones y directrices que deben seguirse al desarrollar una aplicación o programa, para proporcionar un marco estructural que facilita a desarrolladores, analistas y al equipo a cumplir los requisitos del sistema con una base coherente para su implementación. Según Bass et al. (2020), estos patrones no solo

proporcionan un marco para la estructuración de la aplicación, sino que también aseguran que todos los componentes del sistema estén alineados en términos de comunicación y funcionalidad.

La arquitectura de software es fundamental para diseñar una solución de e-commerce que integre microservicios y Cloud Computing para demostrar a lo largo de la investigación cómo estas tecnologías trabajan perfectamente y pueden ser beneficiosas en términos de escalabilidad, adaptabilidad y competitividad en el mercado. La arquitectura proporciona una base sólida para comprender que componentes del sistema que se utilizaran y cómo se comunican entre sí. Además, define el enfoque general del desarrollo, establece fundamentos para futuras implementaciones y asegura que la aplicación pueda evolucionar eficientemente para satisfacer cambios o necesidades emergentes.

2.3.4. COMERCIO ELECTRÓNICO

Se refiere al negocio de compra y venta de bienes o servicios a través de internet. Su relevancia para esta investigación radica en que es el ámbito donde se aplicará la arquitectura propuesta. Dado que el comercio electrónico está en constante evolución y crecimiento, contar con una arquitectura de software robusta y flexible es crucial para aprovechar las oportunidades de este gran mercado.

La rentabilidad del e-commerce en la economía digital subraya la necesidad de una solución que pueda adaptarse a cambios en la demanda y en las tecnologías. Por lo que, implementar microservicios y Cloud Computing no solo facilita una mayor escalabilidad y adaptabilidad, sino que también optimiza la gestión de recursos y mejora la experiencia del usuario, asegurando que la plataforma pueda mantenerse competitiva y eficiente en un entorno en constante cambio.

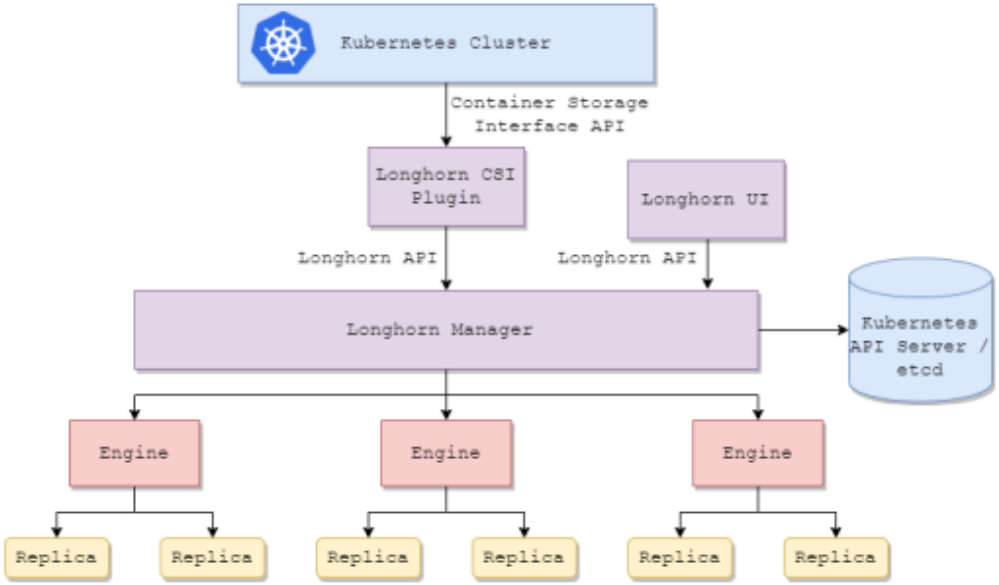
2.3.5. CONTENEDORES Y ORQUESTACIÓN

Los contenedores son una tecnología o unidad de software que encapsula una aplicación con sus dependencias, bibliotecas y todo su entorno de ejecución, con el fin de que la aplicación tenga todo lo que necesita para ejecutarse en cualquier entorno sin problemas. Como señala Merkel (2014), es un método de virtualización a nivel de sistema operativo que permite empaquetar una aplicación y sus dependencias en un contenedor, garantizando que se ejecute de manera consistente en cualquier entorno.

Por otra parte, la orquestación de contenedores se refiere a la automatización del ciclo de vida de los contenedores, que abarca su despliegue, eliminación, desarrollo, gestión, escalado y operación. Su objetivo es simplificar la administración de contenedores en todas sus etapas. Según Burns et al. (2021), Kubernetes es la herramienta de orquestación más utilizada, proporcionando funcionalidades avanzadas para gestionar aplicaciones en contenedores a gran escala.

Para el proyecto, es esencial considerar estos dos conceptos porque los contenedores y la orquestación son fundamentales para el despliegue y la gestión de arquitecturas de microservicios en la nube. Como se explicó anteriormente, los contenedores permiten empaquetar y ejecutar aplicaciones en cualquier entorno, mientras que la orquestación automatiza su ciclo de vida, incluyendo despliegue, escalado y gestión. Según Kim et al. (2020), esta combinación facilita la implementación continua y la escalabilidad de las aplicaciones de comercio electrónico, garantizando alta disponibilidad y rendimiento. Además, la orquestación asegura que las aplicaciones puedan adaptarse rápidamente a cambios en la demanda y mantener un funcionamiento óptimo en todo momento.

Figura 14. Arquitectura del Clúster de Kubernetes con Longhorn



Nota. Mejia, C. (2022). Arquitectura Longhorn [Figura]. Biblioteca digital udea.

3. DISEÑO DE LA INVESTIGACIÓN

El proyecto se llevó a cabo bajo un diseño de investigación de tipo exploratoria y explicativa con un enfoque cualitativo. Su objetivo fue establecer una base sólida para justificar cómo la implementación de microservicios y cloud computing impactaba la escalabilidad, disponibilidad y rendimiento de un e-commerce en su etapa inicial. Esta metodología facilitó la identificación de los recursos necesarios para diseñar una arquitectura adaptable y competitiva ante las demandas del mercado, además de explorar y definir las relaciones causales entre estas variables. Dicho de otra manera, el estudio no solo analizó el impacto individual de cada tecnología en estas métricas, sino que también investigó cómo interactúan entre sí para influir en el desempeño general de la arquitectura de software para comercio electrónico.

El enfoque cualitativo se implementó mediante el análisis de casos de estudio y la revisión de tesis, proyectos de grado y artículos científicos relacionadas con el tema, disponibles en bases de datos académicas como Google Scholar. Esto permitió comparar distintas teorías y enfoques, ofreciendo una visión detallada de cómo estas tecnologías pueden mejorar u optimizar la arquitectura de software para e-commerce.

La diversidad de estas fuentes garantizó una representación amplia y precisa, abarcando diferentes tamaños de empresas o sectores que contribuyeron a una evaluación más robusta y generalizable. Este enfoque permitió obtener conclusiones basadas en el análisis teórico de las dinámicas y beneficios asociados con la implementación de microservicios y Cloud Computing, identificando los aspectos clave que justifican su adopción, valor y efectividad.

En otras palabras, el enfoque metodológico se basó en la revisión, observación, comparación y análisis de estudios académicos previos para establecer una base sólida que facilitara la evaluación del impacto de los microservicios y el Cloud Computing en comercios electrónicos. Cada hipótesis se formuló para evaluar una dimensión específica de cómo la implementación de estas tecnologías influye positiva o negativamente en aspectos clave como la escalabilidad, la disponibilidad y el rendimiento de las aplicaciones, así como en sus capacidades operativas y su eficiencia. Por ello, se empleó el método deductivo para probar las hipótesis y validar las ventajas de utilizar estas tecnologías, proporcionando evidencia sólida que respalde su adopción en el diseño y desarrollo de arquitecturas de software para e-commerce.

3.1. VARIABLES DE ESTUDIO

- **Variable Independiente:** Arquitectura de Microservicios y Cloud Computing. La adopción de estas tecnologías se considera una variable independiente debido a su papel en la transformación de las aplicaciones de comercio electrónico. Los microservicios permiten dividir una aplicación en componentes autónomos, cada uno con su propia funcionalidad específica, lo que facilita el desarrollo, despliegue y mantenimiento independiente de cada servicio (Newman, 2015). Este enfoque modular no solo incrementa la eficiencia operativa, sino que también permite a las empresas adaptar rápidamente sus aplicaciones a las cambiantes demandas del mercado.

  Por otro lado, el Cloud Computing proporciona una infraestructura escalable y flexible que puede ajustarse dinámicamente a las necesidades de la aplicación. Servicios como Amazon Web Services (AWS), Google Cloud Platform (GCP) y Microsoft Azure ofrecen recursos bajo demanda, lo que

permite a las aplicaciones escalar horizontalmente sin necesidad de grandes inversiones iniciales en infraestructura (Armbrust et al., 2010). La combinación de microservicios y Cloud Computing crea un entorno altamente adaptable y eficiente que facilita la gestión de grandes volúmenes de datos y transacciones, esenciales para el éxito en el comercio electrónico.

- **Variables Dependientes:** Escalabilidad, Disponibilidad y Rendimiento en aplicaciones de comercio electrónico.

  En el presente estudio, se analizaron tres variables dependientes esenciales: la escalabilidad, disponibilidad y rendimiento, las cuales son importantes ya que representan los principales indicadores del éxito de una arquitectura de software en aplicaciones de comercio electrónico. Estas métricas proporcionan una visión integral de la eficacia y eficiencia de las arquitecturas modernas en el contexto del comercio electrónico.

  La escalabilidad se refiere a la capacidad de la aplicación para manejar un incremento en la carga de trabajo sin comprometer su desempeño, lo que es fundamental durante picos de tráfico, como eventos o fechas de ventas especiales. Las arquitecturas de microservicios permiten escalar de manera más eficiente al posibilitar la escalabilidad independiente de cada componente, optimizando el uso de recursos y reduciendo costos operativos (Fowler & Lewis, 2014). Esto significa que, en lugar de escalar toda la aplicación, solo se escalan los servicios que experimentan una mayor demanda y deben recibir las mejoras.

  La disponibilidad, por su parte, es la que garantiza que la aplicación esté operativa y accesible en todo momento. En este entorno, la alta disponibilidad es importante para evitar la pérdida de ventas y mantener la satisfacción del cliente. Las soluciones basadas en Cloud Computing ofrecen herramientas avanzadas para la redundancia y recuperación ante desastres, a diferencia

de las soluciones on-premises que pueden requerir una planificación y ejecución compleja para su recuperación, lo que minimiza significativamente el tiempo de inactividad (Google Cloud, 2018).

Por último, el rendimiento mide la eficacia de la aplicación en términos de tiempo de respuesta y capacidad de procesamiento, un rendimiento óptimo es crucial para proporcionar una experiencia de usuario fluida y mejorar la tasa de conversión o la proporción de visitantes que realizan una compra. Los microservicios y el Cloud Computing permiten optimizar y gestionar cada servicio de forma independiente, resultando en tiempos de respuesta más rápidos y una mayor resiliencia frente a fallos en comparación con las arquitecturas monolíticas (Microsoft Azure, 2019).

Estos factores combinados hacen que la escalabilidad, disponibilidad y rendimiento sean métricas clave para evaluar el impacto de las arquitecturas de microservicios y Cloud Computing en aplicaciones de e-commerce.

3.2. HIPÓTESIS

Las hipótesis formuladas para esta investigación son:

- H1: La implementación de microservicios en la nube permite a las aplicaciones de e-commerce escalar de manera más eficiente. En lugar de escalar toda la aplicación, cada microservicio puede escalar independientemente según sus necesidades específicas. Esto no solo mejora la escalabilidad general, sino que también optimiza el uso de recursos, lo cual es crítico durante picos de tráfico, como eventos de ventas especiales.
- H2: La adopción de Cloud Computing ofrece redundancia, balanceo de carga y recuperación ante desastres de manera más eficiente en comparación con

las soluciones on-premises. Las plataformas de nube como AWS, GCP y Azure proporcionan herramientas y servicios que minimizan el tiempo de inactividad mediante actualizaciones automáticas, replicación de datos y escalado automático, asegurando una alta disponibilidad.

- H3: Las arquitecturas basadas en microservicios y Cloud Computing permiten un rendimiento superior debido a la capacidad de optimizar y gestionar cada microservicio de forma independiente. Esto resulta en tiempos de respuesta más rápidos, una mejor experiencia del usuario y una mayor resiliencia frente a fallos en comparación con las aplicaciones monolíticas, donde un fallo en un componente puede afectar a toda la aplicación.

3.3. TÉCNICAS

Casos de estudio: Se realizó un análisis detallado de empresas o comercios electrónicos que migraron a arquitecturas de microservicios en la nube, para la recopilar datos históricos de rendimiento y disponibilidad, así como información sobre el proceso de migración.

3.4. PROCEDIMIENTO O FASES

3.4.1. DISEÑO DEL INSTRUMENTO DE RECOLECCIÓN:

Se desarrollaron criterios de selección para identificar proyectos de grado o artículos científicos relevantes que ofrecieran análisis comparativos y observaciones relevantes sobre la implementación de microservicios y Cloud Computing en un comercio electrónico. En estos criterios se tuvieron en cuenta documentos con cierta cantidad de antigüedad, tener el mismo contexto y haber sido testeados por plataformas reconocidas como AWS o Azure.

3.4.2. ANÁLISIS DE DATOS:

- **Análisis Descriptivo**: Se examinaron las características y patrones emergentes en las implementaciones de microservicios y Cloud Computing en e-commerce. Este análisis proporcionó una visión detallada y contextualizada de cómo estas tecnologías se están utilizando y percibiendo en la práctica.
- **Análisis de Contenido**: Se exploraron las narrativas y experiencias de los usuarios y expertos para identificar temas o tendencias clave relacionadas con la adopción de microservicios y Cloud Computing. Este enfoque permitió comprender las implicaciones y beneficios percibidos desde una perspectiva cualitativa.
- **Evaluación de Percepciones**: Se analizaron las opiniones sobre las mejoras y desafíos asociados con las tecnologías implementadas, proporcionando una base teórica para argumentar su impacto y efectividad en el contexto del e-commerce.

3.4.3. CONCLUSIONES Y RECOMENDACIONES:

A partir de los resultados obtenidos, se formularon las conclusiones de la investigación. También se elaboraron recomendaciones prácticas para la implementación de arquitecturas de microservicios y Cloud Computing en el ámbito del comercio electrónico.

3.5. LIMITACIONES

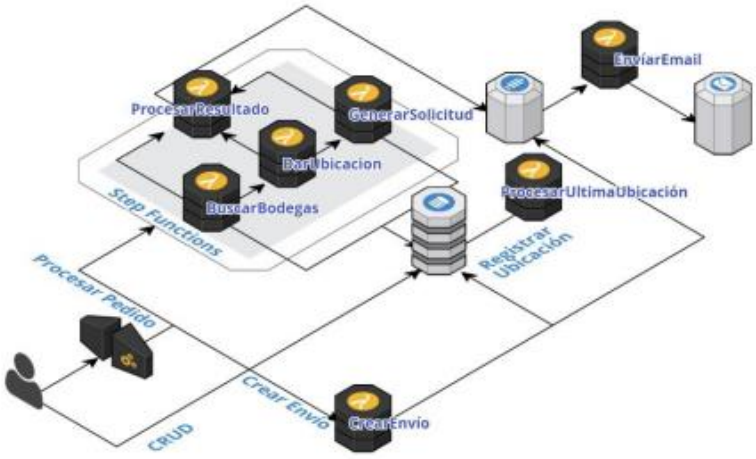
Durante el proceso de investigación, se identificaron las siguientes limitaciones:

- Disponibilidad de Datos: La disponibilidad y accesibilidad de estudios relevantes pudieron ser limitantes, especialmente cuando los datos históricos no estaban detalladamente documentados.
- Varianza en la Implementación: Las diferencias en la implementación y documentación de microservicios y Cloud Computing en los estudios pudieron afectar los resultados, dado que no todas las implementaciones son iguales o suplen las mismas necesidades.

4. DESARROLLO DEL TRABAJO DE GRADO

Para el diseño de la arquitectura de software basada en microservicios y Cloud Computing, se optó por utilizar servicios de Amazon Web Services (AWS) para garantizar la escalabilidad, la disponibilidad y el rendimiento óptimo de un comercio electrónico en su fase inicial. De manera resumida, esta arquitectura utiliza Amazon S3 para el almacenamiento de archivos estáticos como imágenes y documentos, mientras que Amazon CloudFront se ocupa de la entrega rápida y eficiente del contenido, Amazon Route 53 gestiona el DNS y Amazon API Gateway facilita la gestión de APIs. Por otro lado, Amazon ECS y EKS soportan la orquestación de contenedores, los mecanismos de seguridad contra amenazas se implementan con AWS WAF y Amazon Cognito protege las aplicaciones y gestiona la autenticación de usuarios. Además, Amazon CloudWatch y AWS X-Ray se encargan de la monitorización y AWS Backup gestiona las copias de seguridad.

Figura 15. Flujo de Procesos Logísticos



Nota. Castro Duarte, J. S. (2017). Arquitecturas de software sin servidor: un caso de estudio para el despliegue en amazon Web Services, Microsoft azure e IBM Bluemix.

En el desarrollo de este proyecto, se siguieron diversas etapas y técnicas para cumplir con los objetivos establecidos y resolver la problemática planteada. A continuación, se detalla el enfoque general y las fases consideradas para el planteamiento de la arquitectura.

4.1. DEFINICIÓN DE REQUERIMIENTOS

En esta etapa, se establecieron los requerimientos necesarios para abordar aspectos importantes que garanticen una arquitectura robusta y eficiente capaz de manejar grandes volúmenes de tráfico y datos de un comercio electrónico.

4.1.1. REQUERIMIENTOS FUNCIONALES:

- Escalabilidad: La plataforma debe poder adaptarse o escalar en respuesta al aumento en el número de usuarios y transacciones sin necesidad de una intervención manual, asegurando que los recursos tecnológicos disponibles siempre estén alineados a la creciente demanda.
- Disponibilidad: Los usuarios deben tener acceso a la plataforma desde cualquier lugar y momento, incluso durante periodos con alto tráfico. Esto para no generar caídas o retrasos innecesarios.
- Rendimiento: La plataforma debe entregar contenido como imágenes y videos de forma rápida y eficiente a usuarios en todo el mundo, asegurando que las páginas se carguen sin demoras y ofrezcan una experiencia de usuario fluida.

4.1.2. REQUERIMIENTOS NO FUNCIONALES:

- Seguridad de Datos: Es importante proteger los datos sensibles de los usuarios mediante sistemas de autenticación robustos y políticas de acceso restringidas.
- Resiliencia y Recuperación: La arquitectura debe contar con mecanismos que le permitan recuperarse rápidamente en caso de algún fallo para minimizar el impacto del negocio y asegurar la continuidad del servicio.
- Optimización de Costos: La plataforma debe aprovechar los recursos de manera eficiente, maximizando el retorno de inversión y minimizando costos.

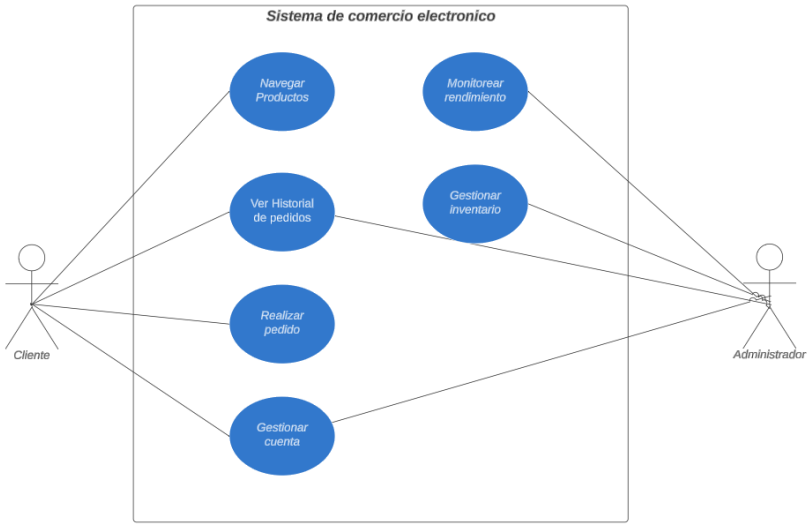
4.2. DISEÑO Y ELABORACIÓN DE LA ARQUITECTURA

Basado en los requerimientos establecidos anteriormente, se dio inicio al diseño de una arquitectura de software para un comercio electrónico.

4.2.1. DIAGRAMA DE CASOS DE USO

El desarrollo del diagrama de casos de uso es el primer paso en este proceso y proporciona una visión general de las interacciones entre los actores externos y el sistema. El diagrama planteado (Figura 16) muestra las diferentes formas en que los actores interactúan con el sistema, identificando los procesos y funcionalidades que se requieren para cumplir con lo establecido inicialmente.

Figura 16. Diagrama de casos de uso planteado para la arquitectura.



Nota. Del autor

4.2.2. SELECCIÓN E INTEGRACIÓN DE COMPONENTES

En esta sección, se detalla cómo se integraron y cuál es la función de cada componente de AWS de la arquitectura planteada. Es decir, se explica qué servicios se utilizaron, cómo se implementaron y su contribución para cumplir con los requisitos del sistema, ya que cada servicio aborda un aspecto diferente en el sistema.

4.2.2.1 Almacenamiento y entrega de Contenido:

Los componentes clave que ayudan a realizar los despliegues en la infraestructura para cumplir con los requisitos de disponibilidad, rendimiento y escalabilidad son:

- **Amazon S3:** Se usa para almacenar archivos estáticos, lo que resulta en una combinación de alta disponibilidad y durabilidad. S3 asegura la disponibilidad de los archivos en cualquier momento, incluso durante altos picos de tráfico.
- **AWS CloudFront:** Es configurado para las entregas eficientes de contenido a través de ubicaciones de borde y gestionado con AWS Certificate Manager (ACM) para la seguridad de SSL/TLS. Además, puede manejar una gran cantidad de solicitudes.
- **AWS Route 53:** Gestiona el DNS para asegura un enrutamiento rápido y fiable de las solicitudes. contribuye a la disponibilidad al garantizar que las solicitudes se dirijan al servidor correcto, incluso durante altos volúmenes de tráfico.

4.2.2.2 GESTIÓN DE SEGURIDAD Y ACCESO:

Los servicios que brindan seguridad, protección y monitoreo de la infraestructura son:

- **AWS WAF:** Protege las aplicaciones contra amenazas comunes mediante la implementación de reglas de firewall, lo que ayuda a prevenir ataques y mejorar la seguridad de datos.
- **AWS Cognito:** Se utiliza para la autenticación y autorización de usuarios, lo cual garantiza la gestión segura de identidades.
- **Geoblocking:** Restringe el acceso basado en la ubicación geográfica, mejorando la seguridad de los datos al limitar el acceso a la plataforma desde ubicaciones no deseadas.

4.2.2.3 ORQUESTACIÓN DE CONTENEDORES Y MICROSERVICIOS:

Los servicios necesarios para el desarrollo y despliegue eficiente de microservicios son:

- **AWS ECS y EKS:** Gestiona y organiza los contenedores, lo que facilita el despliegue y la expansión de microservicios. Ambos servicios ajustan automáticamente la cantidad de recursos según la demanda.
- **AWS API Gateway:** Facilita la creación, publicación, mantenimiento, vigilancia y protección de APIs a cualquier escala. Para garantizar una integración fluida entre los servicios, funciona como puerta de enlace para los microservicios.

4.2.2.4 MONITOREO Y LOGÍSTICA:

Los servicios que proporcionan visibilidad, control del rendimiento y salud de la infraestructura son:

- **AWS CloudWatch:** Proporciona información sobre el rendimiento y la salud de la infraestructura al monitorear y gestionar registros.
- **AWS X-Ray:** Ayuda en el análisis y la depuración de aplicaciones distribuidas para encontrar problemas de rendimiento.

4.2.2.5 GESTIÓN DE DATOS Y RESPALDO:

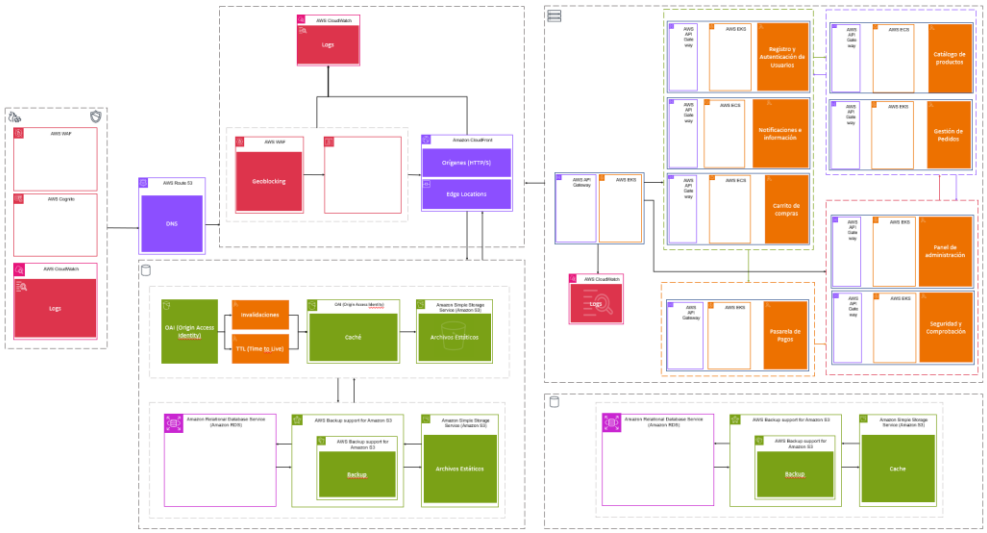
Los siguientes servicios garantizan la seguridad de los datos y su acceso, así como la capacidad de recuperación en caso de fallas:

- **Amazon RDS:** Administra las bases de datos relacionales y garantiza el almacenamiento de datos.
- **AWS Backup:** Asegura la recuperación de datos con soporte para respaldos de Amazon S3.

4.2.3. PLANTEAMIENTO DE LA ARQUITECTURA

El diagrama adjunto ilustra la arquitectura de software diseñada para un comercio electrónico con microservicios y Cloud Computing en AWS. Esta representación gráfica muestra la arquitectura dividida en componentes para que sea más fácil entender el funcionamiento de cada uno por separado. Además, cada componente está ligado a elementos gráficos que permiten conocer el flujo de trabajo, cómo se comunican o interactúan entre sí para lograr los objetivos del sistema.

Figura 17. Diagrama planteado con los recursos expuestos anteriormente

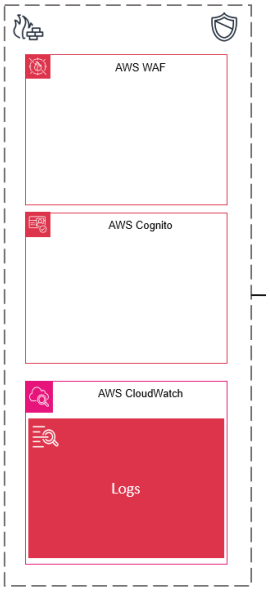


Nota. Del Autor

Como se puede evidenciar en la figura, cada componente tiene una responsabilidad específica y bien definida. Algunos se encargan de la representación de elementos gráficos, el acceso independiente a los datos de cada uno de los módulos y otros encapsulan servicios de seguridad, entre otros. A continuación, se describe el flujo de datos y la función específica de cada componente en el diagrama:

4.2.3.1 FIREWALL

Figura 18. Componente "Firewall"



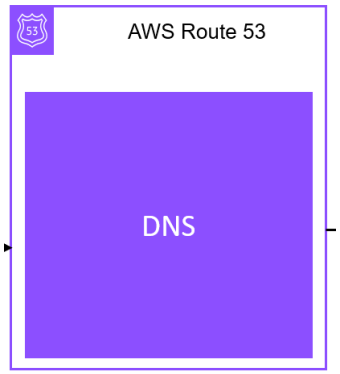
Nota. Del Autor

AWS WAF (Web Application Firewall) actúa como un componente crítico para la protección de aplicaciones web al filtrar y monitorear el tráfico HTTP/HTTPS que llega a las aplicaciones. Su funcionamiento se basa en la definición de reglas que permiten o bloquean solicitudes según criterios específicos, tales como direcciones IP, cabeceras HTTP, cadenas de consulta y cuerpos de solicitudes. AWS WAF utiliza listas de control de acceso y patrones predefinidos para mitigar ataques comunes como inyecciones SQL, cross-site scripting (XSS) y ataques de denegación de servicio (DoS). Además, se integra con otros servicios de AWS, como Amazon CloudFront y Application Load Balancer, para proporcionar una capa adicional de seguridad al aplicar políticas personalizadas y ajustadas a las necesidades específicas de la aplicación, lo que contribuye a mantener la integridad y disponibilidad del entorno en la nube.

Este componente es el primer filtro de seguridad, junto con dos adicionales dentro del componente Front-End y los sistemas de seguridad nativos de cada módulo adicional, que actúa como una puerta de seguridad y revisión de solicitudes del aplicativo para proteger toda la infraestructura.

4.2.3.2 DNS

Figura 19. Componente "DNS"



Nota. Del Autor

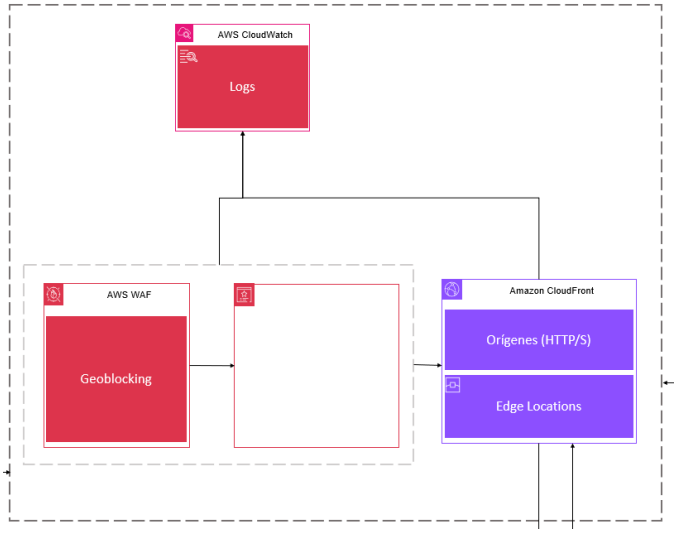
Un servicio de DNS llamado AWS Route 53 facilita la traducción de nombres de dominio en direcciones IP, permitiendo así la resolución de nombres de dominio para aplicaciones web. Ofrece varias políticas de enrutamiento, como la basada en latencia, geográfica y de failover, que dirigen el tráfico de manera eficiente según la proximidad, la disponibilidad o la ubicación del usuario. Además, permite el registro y la gestión de dominios, integrándose con otros servicios de AWS como Elastic Load Balancing y Amazon CloudFront para una gestión centralizada. Sus capacidades de monitoreo y chequeo de salud aseguran la alta disponibilidad al redirigir el tráfico en caso de fallos, optimizando la resiliencia y el rendimiento de las aplicaciones distribuidas en la nube.

A través de la función de AWS Route 53, se puede acceder al aplicativo. Como se muestra en el diagrama, solo se permite el acceso al Front-End, mientras que los demás componentes solo pueden acceder si el sistema lo necesita. Esto evita la posibilidad de fugas mediante el uso de cualquier otro tipo de solicitud DNS fuera de lo establecido por el Firewall anteriormente establecido.

4.2.3.3 FRONT-END

Dentro de la arquitectura en la esquina superior izquierda, AWS CloudWatch se presenta como un servicio dedicado al monitoreo de recursos y aplicaciones en tiempo real dentro de AWS, mientras proporciona métricas, alarmas y registros para una gestión proactiva del rendimiento de recursos y aplicaciones.

Figura 20. Componente "Front-End"



Nota. Del Autor

Justo debajo, AWS WAF actúa como un firewall de aplicaciones web y se representa con un ícono de escudo para simbolizar su función de protección contra ataques

comunes. La sección de “Geoblocking” dentro de AWS WAF indica la aplicación de reglas de bloqueo geográfico para restringir el acceso basado en la ubicación del usuario.

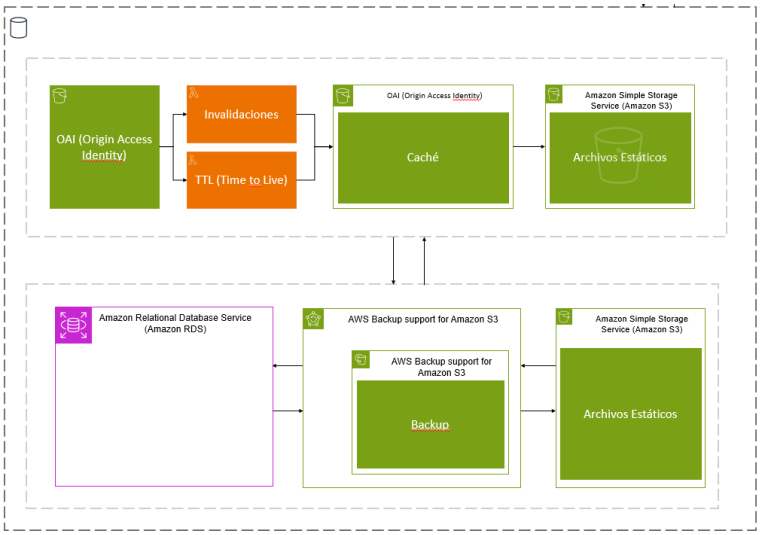
A la derecha se encuentra Amazon CloudFront, un servicio de red de distribución de contenido (CDN) representado por el color morado y dividido en dos componentes principales: los orígenes (HTTP/S) son los servidores donde se aloja el contenido y las Edge Locations, son servidores distribuidos globalmente que almacenan en caché el contenido cerca de los usuarios para reducir la latencia y mejorar el rendimiento.

Este componente se encarga del renderizado de la aplicación, se comunica con la base de datos específica para este componente para acceder a elementos gráficos que sean necesarios, datos pequeños y otros elementos temporales para el componente de Front-End. Luego, se comunica con el componente Back-End, donde se encuentra estructurado el sistema de microservicios para cada una de las funciones del e-commerce.

4.2.3.4 ALMACENAMIENTO PARA FRONT-END

Para comenzar, el diagrama a continuación muestra que el OAI (Origin Access Identity) es un elemento esencial para el almacenamiento porque proporciona un mecanismo para asegurar el acceso a los objetos almacenados en Amazon S3 a través de CloudFront sin hacerlos públicos, mejorando así la seguridad y el control de acceso.

Figura 21. Componente "DB Front-End"



Nota. Del Autor

El TTL (Time to Live) configura el tiempo que un objeto permanece almacenado en caché en una Edge Location de Amazon CloudFront antes de ser refrescado desde el origen. Esto optimiza el rendimiento al reducir las solicitudes al servidor de origen, aunque puede retrasar la actualización de cambios en el contenido. Las invalidaciones ayudan a eliminar objetos de la caché antes de que expire el TTL, lo que obliga a actualizar inmediatamente el contenido a la versión más reciente.

Por otro lado, Amazon RDS facilita la gestión de bases de datos SQL y NoSQL, mientras que AWS Backup automatiza y programa copias de seguridad para los objetos en Amazon S3, garantizando la recuperación de datos en caso de fallos. Amazon S3 ofrece almacenamiento escalable y seguro para datos como imágenes o archivos con alta disponibilidad y durabilidad. Al trabajar en conjunto, estos componentes optimizan la entrega de contenido, aumentan la seguridad y permiten un control efectivo sobre el almacenamiento y el respaldo de datos.

4.2.3.5 BACK-END

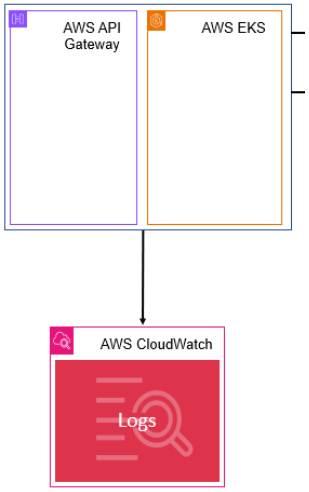
Figura 22. Componente "Back-End"

Nota. Del Autor

El componente Back-End (ver figura 22) se beneficia significativamente de la arquitectura de microservicios al descomponer las funciones complejas en servicios independientes y especializados que interactúan entre sí. Esta descomposición no solo mejora la escalabilidad y flexibilidad del sistema, sino que también facilita el mantenimiento y la implementación continua.

En otras palabras, Cada microservicio funciona de manera autónoma, lo que permite actualizar o escalar solo los componentes específicos según la demanda sin afectar a otras partes del sistema. Esto es fundamental para los comercios electrónicos, donde varios módulos, como la autenticación de usuarios, el catálogo de productos, la gestión de pedidos y el carrito de compras pueden experimentar diferentes niveles de uso a lo largo del tiempo.

Figura 23. Módulo principal para Back-End.

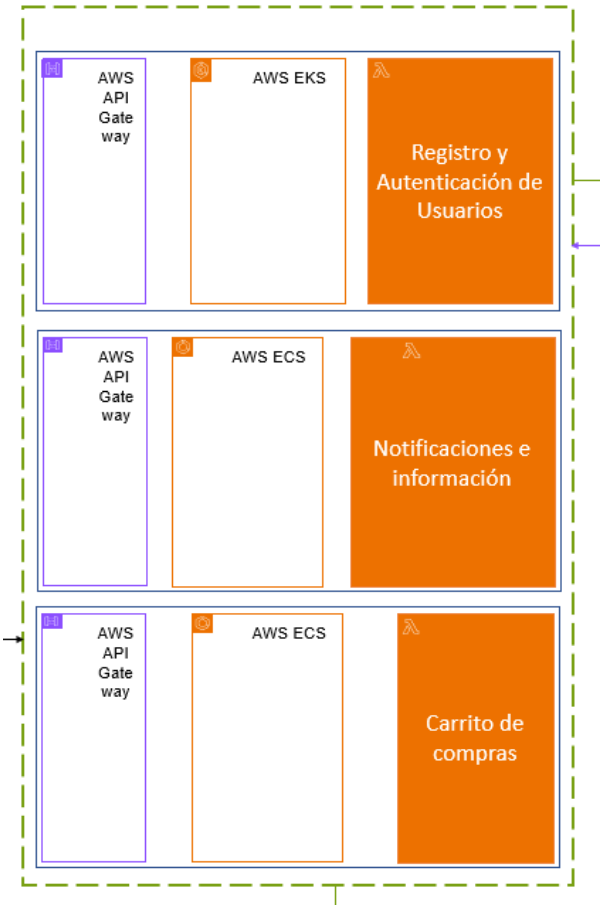


Nota. Del Autor

El módulo principal actúa como punto de partida para iniciar el flujo de trabajo del Back-End, utiliza AWS API Gateway para obtener los datos de forma eficiente y cuenta con un componente CloudWatch para realizar Logs o registros de todos los módulos según sea necesario.

Los dos módulos principales del esquema del Back-End se encargan de gestionar los dos tipos de usuarios que acceden al E-Commerce: el administrador se encarga de revisar y controlar todos los módulos y el usuario puede acceder a los datos de los productos y otros elementos como filtros, precios, etc. El módulo del usuario (ver figura 24) se comunica con el módulo de productos para obtener la lista de productos en Stock o disponibles en ese momento, así como con el módulo de la pasarela de pagos para informarle que productos adquirió el usuario.

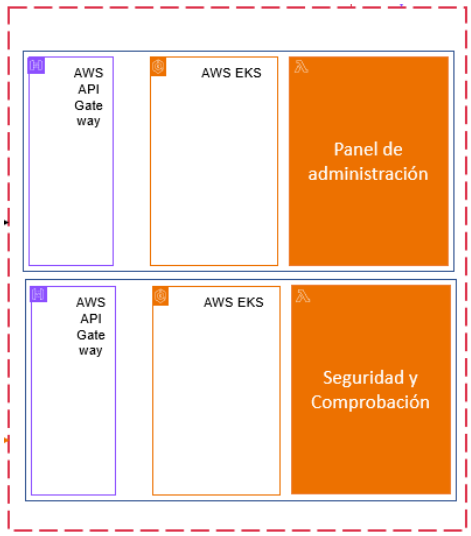
Figura 24. Módulo de gestión del usuario.



Nota. Del Autor

Por otra parte, el módulo de administración se comunica con todos los demás para poder recibir información sobre cada uno de los elementos que suceden en el flujo de trabajo en dado caso de que se necesiten cambios, el panel de administración contiene la información y funciones necesarias para auditar de forma completa todo el flujo de trabajo del Back-End.

Figura 25. Módulo de administración.



Nota. Del Autor

Dependiendo de sus roles dentro del sistema, las flechas entre módulos muestran cómo se conectan y el flujo de información entre ellos. Este diagrama evita el acceso no autorizado al módulo de administración, garantizando un estándar de seguridad alto.

Para explicar la funcionalidad de cada componente que se muestra en el diagrama, tenemos que:

El catálogo de productos almacena la información de los productos disponibles en una base de datos como Amazon DynamoDB, lo que facilita el acceso rápido y eficiente a la información de los productos, importante para una experiencia de usuario fluida y actualizada. Mientras tanto, la gestión de pedidos procesa los pedidos de los usuarios realizando cálculos, verificaciones de inventario y actualizaciones de estado de pedido, asegurándose de que los pedidos se procesen

de manera precisa y eficiente, coordinando la disponibilidad de productos, al igual que proporcionando actualizaciones de estado de pedido en tiempo real.

Además, las notificaciones e información del usuario utilizan Amazon SNS para enviar notificaciones y actualizaciones relevantes a los usuarios, manteniendo a los usuarios informados sobre el estado de sus pedidos y otros eventos importantes, mejorando la comunicación y la experiencia del usuario.

El carrito de compras registra temporalmente los productos seleccionados por el usuario antes de la compra, a menudo usando Amazon RDS. De esta manera, los usuarios pueden revisar y modificar sus selecciones antes de finalizar la compra, facilitando una experiencia de compra más flexible. Finalmente, el controlador principal es responsable de coordinar las acciones entre los componentes, como verificar la disponibilidad de productos, calcular precios y actualizar el estado del pedido, también maneja la interacción con el carrito de compras y la notificación a los usuarios, actuando como el núcleo de la lógica de negocio en el Back-End e integrando los componentes del sistema para proporcionar una experiencia de usuario coherente y eficiente.

Los componentes de AWS más utilizados son:

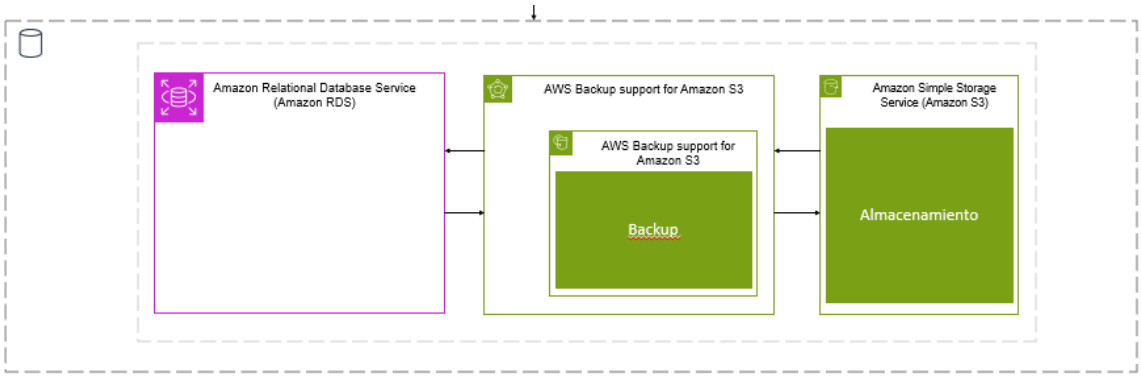
- API Gateway: Expone APIs para que los clientes interactúen con los servicios backend, facilitando la comunicación entre los microservicios y las interfaces de usuario.
- AWS Lambda: Ejecuta código sin servidor en respuesta a eventos, como el procesamiento de pedidos, permitiendo una ejecución escalable y bajo demanda.

- Amazon EKS: Despliega instancias virtuales para ejecutar aplicaciones, proporcionando flexibilidad y escalabilidad en la infraestructura.
- Amazon ECS: Es un servicio de orquestación de contenedores que proporciona una solución fácil de usar para ejecutar cargas de trabajo de contenedores que ayuda a implementar, administrar y escalar aplicaciones de contenedores.
- Amazon S3: Almacena objetos en la nube, como imágenes de productos y archivos estáticos, ofreciendo un almacenamiento escalable y duradero.

Como resultado, la combinación de todos estos servicios de AWS y la arquitectura de microservicios crea un backend robusto, escalable y eficiente que administra de manera efectiva los usuarios, productos, pedidos y notificaciones, garantizando una experiencia de usuario fluida y un sistema altamente disponible o seguro.

4.2.3.6 ALMACENAMIENTO PARA BACK-END

Figura 26. Componente "Almacenamiento para Back-End"



Nota. Del Autor

El diagrama anterior muestra el flujo de procesos relacionados con la gestión de datos y las copias de seguridad en los servicios de Amazon Web Services (AWS). El primer cuadro a la izquierda muestra el servicio de base de datos relacional de

Amazon, que es responsable de la gestión de la base de datos. Posteriormente, se conecta al soporte de AWS Backup para almacenar copias de seguridad de Amazon RDS en Amazon S3 mediante AWS Backup. De esta manera, los datos se almacenan en Amazon S3 como se muestra en el tercer cuadro.

4.3. CASOS DE USO Y EVALUACIÓN DE LA ARQUITECTURA

4.3.1. OPTIMIZACIÓN DEL ALMACENAMIENTO Y CONTENIDO:

Los contenidos como imágenes y videos se almacenan en una infraestructura sólida que permite su distribución global de manera eficiente. Esto asegura que los usuarios reciban el contenido rápidamente, sin importar su ubicación y mejorando significativamente la experiencia del cliente. La satisfacción del usuario aumenta como resultado de esta rapidez en la entrega, lo que aumenta la tasa de conversión para el negocio.

4.3.2. GESTIÓN DE SERVICIOS CLAVE:

Los servicios encargados de manejar la dirección web y las interacciones dentro de la plataforma permiten que los diferentes componentes del comercio electrónico se comuniquen eficazmente. Esto garantiza que el carrito de compras, el procesamiento de pagos y otras funciones clave funcionen de manera fluida, lo que es esencial para mantener una operación comercial eficiente y sin problemas.

4.3.3. ESCALABILIDAD Y FLEXIBILIDAD OPERATIVA:

La plataforma está diseñada para adaptarse automáticamente a la carga de trabajo, asegurando que la operación comercial funcione sin interrupciones a pesar del aumento de la demanda. Esto no solo garantiza la capacidad de respuesta del

sistema, sino que también permite que el negocio se expanda sin preocuparse por los límites tecnológicos.

4.3.4. SEGURIDAD Y MONITOREO EFICIENTE:

La plataforma cuenta con mecanismos de seguridad que protegen contra amenazas y garantizan que solo los usuarios autorizados accedan a los recursos. Además, un sistema de monitoreo avanzado permite una supervisión continua de la operación, lo que permite detectar y resolver rápidamente los problemas que surjan. Esto es fundamental para mantener la confianza de los usuarios y evitar pérdidas operativas.

4.3.5. RESPALDO Y RECUPERACIÓN:

Los datos críticos de la plataforma están respaldados y pueden ser recuperados rápidamente en caso de cualquier eventualidad. Esto reduce el riesgo de interrupciones prolongadas, protegiendo tanto las operaciones como la reputación del negocio. Para mantener la confianza del cliente y garantizar la operación continua del comercio electrónico, es esencial tener una capacidad de recuperación rápida.

4.4. ANÁLISIS DE COSTOS

Las pequeñas y medianas empresas (PYMEs) en Colombia, al igual que en otros países, se enfrentan a limitaciones presupuestarias y maximizar el rendimiento de sus inversiones tecnológicas. En ese contexto, se presenta una arquitectura de software basada en microservicios y Cloud Computing con AWS como una solución eficaz y escalable para optimizar las operaciones de los comercios electrónicos.

Además, es una opción atractiva para las PYMEs colombianas porque pueden adaptarse rápidamente a las demandas del mercado.

A continuación, se muestra un análisis detallado de los costos de los componentes expuestos anteriormente:

4.4.1. SUPUESTOS PARA LA EVALUACIÓN DE COSTOS

- Usuarios activos: 10,000 usuarios mensuales.
- Solicitudes HTTP/S: 100,000 solicitudes diarias.
- Almacenamiento de Contenido: 500 GB en Amazon S3 para imágenes, videos y otros activos estáticos.
- Transferencia de Datos: 1 TB mensual.
- ECS/EKS Clusters: 10 contenedores con 2 vCPUs y 4 GB de RAM por contenedor.
- Base de Datos: Amazon RDS (MySQL) con una instancia db.t3.micro y 20 GB de almacenamiento.
- Seguridad y Monitoreo: AWS WAF con protección básica y AWS CloudWatch para monitoreo.

4.4.2. COSTOS DETALLADOS POR SERVICIO

4.4.2.1 AMAZON S3 (ALMACENAMIENTO Y TRANSFERENCIA DE DATOS):

Tabla 5. Costos mensuales de Amazon S3

Detalle	Costo por unidad	Cantidad	Costo total mensual (USD)	Costo total mensual (aprox. COP)
Almacenamiento inicial	$0.023/GB/mes	500 GB	$11.50	≈ $42,000 COP
Transferencia de datos	$0.09/GB	1 TB	$90	≈ $330,000 COP
Total			**$101.50**	**$420,000 COP**

Nota. Amazon Web Services. (s.f.). Amazon S3 Pricing.

4.4.2.2 AWS CLOUDFRONT (CDN):

Tabla 6. Costos mensuales de Amazon CloudFront

Detalle	Costo por unidad	Cantidad	Costo total mensual (USD)	Costo total mensual (aprox. COP)
Solicitudes HTTP/S	$0.0075 por 10,000 solicitudes	3,000,000 solicitudes	$22.50	≈ $90,000 COP
Transferencia de datos	$0.085/GB	1 TB	$85	≈ $340,000 COP
Total			**$107.50**	**$450,000 COP**

Nota. Amazon Web Services. (s.f.). Amazon CloudFront Pricing.

4.4.2.3 AMAZON ECS/EKS (CONTENEDORES):

Tabla 7. Costos mensuales de Amazon EC2

Detalle	Costo por unidad	Cantidad	Costo total mensual (USD)	Costo total mensual (aprox. COP)
Instancia t3.small	$0.04656/hora	10 instancias x 24 horas/día x 30 días	$339.79	≈ $1,420,000 COP

Nota. Amazon Web Services. (s.f.). Amazon EC2 Pricing.

4.4.2.4 AMAZON RDS (MYSQL):

Tabla 8. Costos mensuales de Amazon RDS

Detalle	Costo por unidad	Cantidad	Costo total mensual (USD)	Costo total mensual (aprox. COP)
Instancia db.t3.micro	$0.017/hora	1 instancia x 24 horas/día x 30 días	$12.41	≈ $52,000 COP
Almacenamiento	$0.10/GB/mes	20 GB	$2	≈ $8,000 COP
Total			**$14.41**	**$60,000 COP**

Nota. Amazon Web Services. (s.f.). Amazon RDS Pricing.

4.4.2.5 AWS WAF (SEGURIDAD):

Tabla 9. Costos mensuales de AWS WAF

Detalle	Costo por unidad (USD)	Cantidad	Costo total mensual (USD)	Costo total mensual (aprox. COP)
Costo fijo mensual por regla web ACL	$5,00	1	$5,00	≈ $20,120 COP
Costo fijo mensual por regla	$1,00	3	$3,00	≈ $12,072 COP
Solicitudes inspeccionadas	$0,60	10	$6,00	≈ $24,144 COP
Total			**$14,00**	**$25,000 COP**

Nota. Amazon Web Services. (s.f.). AWS WAF Pricing.

4.4.2.6 AWS CLOUDWATCH (MONITOREO):

Tabla 10. Costos mensuales de Amazon CloudWatch

Detalle	Costo por unidad	Cantidad	Costo total mensual (USD)	Costo total mensual (aprox. COP)
Métricas	$0.30 por métrica	10 métricas	$3	≈ $12,000 COP
Logs	$0.50 por GB de datos ingresados	10 GB	$5	≈ $21,000 COP
Total			**$8**	**$33,000 COP**

Nota. Amazon Web Services. (s.f.). Amazon CloudWatch Pricing.

4.4.3. COSTOS POR USO

Tabla 11. Costos de los componentes por uso.

Componente	Costo en USD	Costo en COP
Amazon S3	$101.50	$420,000
AWS CloudFront	$107.50	$450,000
Amazon ECS/EKS	$339.79	$1,420,000
Amazon RDS	$14.41	$60,000
Amazon WAF	$5.60	$25,000
AWS CloudWatch	$8.00	$33,000
Total aproximado	**$576.80**	**$2,408,000**

Fuente: AWS Pricing Calculator. (s.f.). Recuperado de https://calculator.aws/#/addService

Numerosos estudios respaldan la implementación de Cloud Computing en las PYMEs colombianas, destacando sus beneficios en términos de agilidad, reducción

de costos operativos y competitividad. Además, demuestran que las soluciones en la nube no solo aumentan la capacidad de innovación y respuesta del mercado, sino que también permiten a las PYME competir en igualdad de condiciones con las grandes empresas. Según investigaciones realizadas por la Universidad Nacional de Colombia, las pequeñas y medianas empresas (PYMES) que utilizan servicios en la nube pueden incrementar su capacidad de innovación y adaptarse a las demandas del mercado. De manera similar, estudios publicados en *The Journal of Cloud Computing* destacan que las tecnologías en la nube hacen que el acceso a herramientas sofisticadas sea más accesible, lo que permite a las pequeñas y medianas empresas competir en igualdad de condiciones con las grandes empresas.

En pocas palabras, la propuesta de implementar AWS satisface las demandas y restricciones financieras de las PYMEs colombianas al proporcionar una infraestructura confiable y escalable a un precio razonable, lo que maximiza el retorno de inversión en el contexto económico local y permite a los emprendedores colombianos acceder a tecnología de clase mundial, la cual es importante para el crecimiento y competitividad.

5. RESULTADOS

A pesar de no haberse implementado en un entorno práctico, la arquitectura sigue las mejores prácticas y se adhiere principalmente a los principios establecidos principalmente a través de la teoría, lo que le permite demostrar sus ventajas o superioridad en comparación con arquitecturas monolíticas y otras arquitecturas de microservicios que no optimizan el uso de servicios en la nube. Para los comercios electrónicos que necesitan evolucionar y expandirse en un mercado altamente dinámico, la inversión en una infraestructura sólida como la que ofrece AWS es una decisión acertada a largo plazo.

Como se mencionó anteriormente, la arquitectura propuesta está diseñada con servicios de AWS, que facilitan la gestión, simplifican la complejidad operativa y garantizan una escalabilidad eficiente en comparación con otras alternativas menos sofisticadas y más económicas. A pesar de que esta implementación inicial pueda resultar más costosa, los beneficios a largo plazo en términos de escalabilidad, reducción de la complejidad operativa y mejora de la resiliencia justifican completamente dicha inversión. Estos componentes no solo garantizan un rendimiento consistente y de alta calidad para los usuarios, sino que también facilitan el crecimiento del negocio sin la necesidad de reconstruir la infraestructura desde cero a medida que se expande.

Uno de los principales beneficios de la arquitectura presentada es su enfoque en el desacoplamiento y modularidad. Los microservicios permiten que cada funcionalidad de la aplicación sea independiente, lo que facilita la escalabilidad y el mantenimiento del sistema, a diferencia de una arquitectura monolítica, donde las actualizaciones y mejoras requieren el despliegue de todo el sistema, aumentando el riesgo de errores y el tiempo de inactividad (Newman, 2021). Es por ello que la arquitectura diseñada aprovecha eso para facilitar la implementación de nuevas funciones sin afectar a las ya existentes, lo cual es crucial para adaptar el comercio electrónico en un entorno altamente competitivo y cambiante.

En otras palabras, la escalabilidad es otra ventaja de la arquitectura presentada porque permite agregar más instancias de microservicios según la demanda en lugar de aumentar la capacidad de los servidores existentes (escalabilidad vertical). Esto es ideal para manejar picos en el tráfico, que son comunes en las plataformas de comercio electrónico durante eventos. La capacidad de escalar de manera eficiente es una ventaja significativa frente a arquitecturas monolíticas y algunas arquitecturas de microservicios que no están optimizadas para la nube, donde la escalabilidad suele ser más compleja y costosa (Desina, 2023).

Para las plataformas de comercio electrónico que experimentan fluctuaciones significativas de demanda durante eventos de alta demanda como promociones o temporadas festivas, la arquitectura de AWS permite un escalado automático en respuesta a esos picos de tráfico. Esta capacidad de escalar sin intervención manual es crucial para diferenciarse de las arquitecturas más simples, que requieren un proceso manual o semi-automatizado con mayor riesgo de fallas y tiempos de inactividad (Oyova Software, LLC).

Por esa razón, esa capacidad de escalado automático ayuda a las empresas que quieren expandirse al optimizar el uso de recursos y mejorar la experiencia del usuario al garantizar la disponibilidad y el rendimiento constante de la plataforma, independientemente del volumen de tráfico. Aunque la arquitectura propuesta no ha sido implementada en un entorno real y la falta de pruebas prácticas es una limitación evidente para evaluar su rendimiento o eficacia en términos de funcionamiento, su diseño teórico demuestra una comprensión sólida de los principios fundamentales de la arquitectura de microservicios en la nube. Esta propuesta es sólida en términos conceptuales, una vez que se implemente, se espera que esta arquitectura cumpla con los requisitos planteados y brinde un marco escalable o resiliente para futuras expansiones.

5.1. JUSTIFICACIÓN DE LA INVERSIÓN EN AWS

Una de las principales ventajas de utilizar AWS radica en la reducción de la complejidad operativa. Servicios como AWS Lambda y Amazon RDS eliminan la necesidad de gestionar manualmente servidores y bases de datos, lo que no solo disminuye la carga de trabajo de los equipos técnicos, sino que también reduce la posibilidad de errores humanos. Esta gestión automatizada es crucial para empresas que buscan enfocar sus recursos en el desarrollo de sus productos y servicios en lugar de en la administración de infraestructura (Oyova Software, LLC).

En contraste, las arquitecturas basadas en soluciones más económicas, como el uso de contenedores gestionados en servidores propios o en plataformas más asequibles, pueden requerir una administración más intensiva. Aunque estas alternativas pueden ser adecuadas para pequeñas empresas con presupuestos limitados, introducen una mayor complejidad en la gestión diaria y requieren

personal especializado para garantizar un funcionamiento eficiente y seguro (Oyova Software, LLC).

Amazon Web Services (AWS) ofrece una infraestructura altamente resiliente gracias a su red distribuida y sus servicios avanzados de recuperación automática ante fallos. Esta infraestructura permite que la plataforma siga operativa incluso cuando ocurren fallos en componentes individuales, lo que reduce el impacto en los usuarios finales. En comparación, las arquitecturas más económicas pueden implementar redundancias, pero no garantizan el mismo nivel de continuidad del negocio, lo que puede ser un riesgo significativo para las plataformas de comercio electrónico que requieren disponibilidad continua para mantener sus ingresos (Oyova Software, LLC).

Para justificar la inversión en una arquitectura de microservicios utilizando AWS, especialmente en el caso de un comercio electrónico que busca evitar una alta inversión inicial, es esencial considerar los beneficios a largo plazo relacionados con la escalabilidad, la resiliencia y la simplificación operativa. La arquitectura basada en microservicios en AWS es particularmente destacable por su resiliencia porque al utilizar servicios como AWS Lambda, Amazon ECS y Amazon EKS, se obtienen capacidades avanzadas de alta disponibilidad y mecanismos eficaces de recuperación automática. Este enfoque minimiza el impacto de fallos individuales, asegurando que la aplicación permanezca operativa incluso si un servicio específico falla. Esto contrasta con las arquitecturas monolíticas, donde un fallo en un módulo puede afectar a todo el sistema, o con arquitecturas de microservicios que no emplean herramientas adecuadas de orquestación y monitoreo, lo que puede conducir a tiempos de inactividad prolongados (Richardson, 2021).

Además, la implementación de microservicios en AWS permite gestionar varios grupos de servicios mediante sistemas EKS (Elastic Kubernetes Service) o ECS (Elastic Container Service), según las necesidades específicas. Esta capacidad de distribuir cargas de trabajo de acuerdo con la complejidad y demanda de los servicios en momentos concretos contribuye a una operación más eficiente y resiliente, garantizando mayor flexibilidad y capacidad de respuesta ante diferentes escenarios operativos.

5.2. COMPARATIVAS

5.2.1. COMPARACIÓN CON OTRAS ARQUITECTURAS DE MICROSERVICIOS

Esta arquitectura se diferencia significativamente de otras implementaciones similares, como las que utilizan contenedores en lugar de funciones sin servidor. Las arquitecturas basadas en contenedores requieren una mayor gestión de infraestructura y una mayor complejidad en la configuración de redes y almacenamiento. Por lo que, el diseño con servicios gestionados de AWS reduce la carga operativa, lo que permite a los desarrolladores concentrarse más en la lógica de negocios que en la infraestructura subyacente (Hasselbring, 2020).

5.2.2. COMPARACIÓN CON ALTERNATIVAS DE BAJO COSTO

Las alternativas de bajo costo para la arquitectura de microservicios, como implementaciones en servidores locales o nubes más económicas, pueden parecer atractivas inicialmente debido a los menores costos iniciales. Sin embargo, estas opciones suelen conllevar una complejidad operativa significativa, particularmente en la gestión de la infraestructura, la escalabilidad y la resiliencia.

En un entorno de comercio electrónico, donde las demandas pueden fluctuar drásticamente y la necesidad de alta disponibilidad es crítica, la falta de características avanzadas como el Auto Scaling y el Elastic Load Balancing de AWS puede limitar la capacidad de respuesta a picos de tráfico, lo que afecta directamente la experiencia del usuario y los ingresos. Asimismo, la resiliencia es menor en estos entornos, ya que suelen depender de configuraciones manuales y menos robustas para manejar fallos.

5.2.3. LIGTHSAIL (ARQUITECTURA MONOLÍTICA)

La arquitectura LigthSail de AWS se presenta como una opción simplificada y económica para desarrolladores o empresas que buscan soluciones de hosting sin la necesidad de lidiar con la complejidad de los servicios avanzados de AWS. LigthSail combina computación, almacenamiento y redes en una plataforma fácil de usar, diseñada para proyectos con requisitos de menor escala. Ofrece funciones básicas, como el almacenamiento de archivos mediante bloques de almacenamiento adjuntos a instancias, opciones limitadas de Content Delivery Network (CDN) y servicios DNS integrados. Pero, LigthSail carece de capacidades avanzadas como la gestión de APIs, la orquestación de contenedores y el monitoreo detallado, características que están disponibles en la arquitectura de microservicios de AWS. Por ejemplo, en LigthSail, la seguridad se limita a opciones básicas de firewall, los servicios de backup y el monitoreo son menos robustos en comparación con lo que se puede lograr mediante AWS WAF, AWS CloudWatch y AWS X-Ray en una arquitectura de microservicios.

5.2.3.1 LIGTHSAIL VS MICROSERVICIOS EN AWS

Al comparar la arquitectura LigthSail con una arquitectura de microservicios en AWS, se observa una diferencia significativa en términos de escalabilidad,

flexibilidad y capacidad de manejo de aplicaciones complejas. Mientras que LigthSail es adecuada para proyectos más pequeños o menos complejos, la arquitectura de microservicios de AWS está diseñada para aplicaciones de comercio electrónico que demandan alta escalabilidad, rendimiento y seguridad avanzada. Una empresa que opere una plataforma de comercio electrónico con tráfico variable que necesite integrar múltiples servicios como gestión de inventarios, pasarelas de pago y herramientas de marketing en tiempo real, encontraría en los microservicios una solución idónea para escalar cada componente de manera independiente, permitiendo un manejo eficiente de los recursos y una mejora en la resiliencia del sistema (Dehghantanha et al., 2019). En este sentido, la superioridad que presenta la arquitectura de microservicios de AWS radica en su capacidad de personalización y adaptación a las necesidades específicas del proyecto, superando ampliamente a LigthSail en este aspecto. Una organización que necesite asegurar la entrega rápida y segura de contenido en distintas regiones geográficas podría utilizar AWS CloudFront junto con Amazon S3 para la distribución de archivos estáticos, mientras que AWS Cognito gestionaría la autenticación y autorización de usuarios.

Estos servicios, junto con la orquestación avanzada de contenedores mediante AWS ECS y EKS, permiten una escalabilidad dinámica y una flexibilidad inigualable que LigthSail no puede ofrecer. Varios estudios han demostrado que la adopción de microservicios y cloud computing en plataformas de comercio electrónico no solo mejora la eficiencia operativa, sino que también reduce los tiempos de inactividad, aumentando así la satisfacción del cliente (Mazzocchi, 2020).

Tabla 12. Comparativa con la Arquitectura LightSail

Aspecto	Arquitectura de Microservicios en AWS	Arquitectura LigthSail
Enfoque General y Complejidad	Complejidad Alta: Utiliza una variedad de servicios especializados como Amazon S3, AWS CloudFront, AWS Route 53, AWS API Gateway, AWS ECS, AWS EKS, AWS WAF, AWS Cognito, AWS CloudWatch, AWS X-Ray y AWS Backup. Es ideal para aplicaciones que requieren alta escalabilidad y rendimiento.	Complejidad Baja a Media: LigthSail proporciona una experiencia más simplificada con opciones integradas para computación, almacenamiento y redes. Es ideal para aplicaciones simples y para desarrolladores que buscan una solución económica y fácil de administrar.
Servicios y Componentes	Incluye servicios como Amazon S3 (almacenamiento), AWS CloudFront (CDN), AWS Route 53 (DNS), AWS API Gateway (gestión de APIs), AWS ECS y EKS (orquestación de contenedores), AWS WAF y Cognito (seguridad), AWS CloudWatch y X-Ray (monitoreo), y AWS Backup (respaldo de datos).	Incluye almacenamiento mediante bloques adjuntos, una opción básica de CDN, un servicio DNS integrado, pero carece de soluciones avanzadas para gestión de APIs, orquestación de contenedores, servicios de seguridad y monitoreo comparables a los de AWS Microservices.

Escalabilidad y Flexibilidad	Ofrece alta escalabilidad y flexibilidad, permitiendo configurar componentes y servicios para satisfacer necesidades específicas. Es capaz de escalar dinámicamente según la demanda, lo que es crucial para aplicaciones de gran escala.	En comparación con la arquitectura de microservicios, tiene una escalabilidad limitada. Las dimensiones de las instancias pueden modificarse, pero la escala dinámica es más limitada. La flexibilidad disminuye y hay menos opciones de configuración avanzada.
Seguridad	Seguridad avanzada a través de AWS WAF y AWS Cognito, que ofrecen protección contra amenazas web y gestión de usuarios. Los servicios de monitoreo como AWS CloudWatch y X-Ray permiten una supervisión detallada y respuesta a incidentes.	Aunque ofrece opciones de firewall básicas, carece de la protección de los servicios de seguridad avanzados de AWS. En comparación con AWS CloudWatch y X-Ray, el monitoreo es limitado y las capacidades de análisis y depuración son más limitadas.
Costos	Debido al uso de múltiples servicios avanzados y la necesidad de escalabilidad, aumentan los costos. Dependiendo del uso y la	Ofrece precios más predecibles y económicos, con una tarifa plana mensual. Es adecuado para proyectos con menos requisitos de

	complejidad de la arquitectura, estos pueden superar los $500 al mes.	escalabilidad y un presupuesto limitado.

Dicho esto, la arquitectura de software basada en microservicios y Cloud Computing utilizando servicios avanzados de AWS es ideal para aplicaciones de comercio electrónico que requieren alta escalabilidad, rendimiento, y seguridad avanzada. Ofrece una flexibilidad y personalización extensiva, adecuada para proyectos complejos y de gran envergadura. Por otro lado, LigthSail es una opción más simple y económica, adecuada para aplicaciones más pequeñas o menos complejas que ofrece una solución integral con menor capacidad de personalización y escalabilidad en comparación con los servicios especializados de AWS. La elección entre estas dos arquitecturas dependerá de las necesidades específicas del proyecto, el presupuesto disponible y el nivel de complejidad requerido.

Aunque LigthSail es una solución atractiva para proyectos más simples o presupuestos limitados, la arquitectura de microservicios de AWS es claramente superior para empresas que buscan construir plataformas de comercio electrónico robustas, escalables y seguras. La capacidad de integrar una gama extensa de servicios especializados, como AWS WAF para protección avanzada contra amenazas y AWS CloudWatch para monitoreo en tiempo real, asegura que las aplicaciones no solo cumplan con las demandas actuales, sino que también estén preparadas para escalar en el futuro. Así, para proyectos que requieren una alta disponibilidad, rendimiento y personalización, la arquitectura de microservicios en

AWS es la elección recomendada, respaldada por casos de éxito en la industria y evidencia empírica (Xu et al., 2021).

5.2.4. EVENT- DRIVEN ARCHITECTURE

5.2.4.1 EVENT-DRIVEN VS MICROSERVICIOS EN AWS

En la arquitectura de microservicios, una aplicación se descompone en servicios pequeños e independientes, cada uno encargado de una función específica. Estos microservicios se comunican a través de APIs y pueden ser desplegados, escalados y actualizados de manera independiente. Este enfoque promueve la modularidad, escalabilidad y la implementación continua, lo que permite a los equipos de desarrollo trabajar en diferentes partes de la aplicación de manera simultánea. Componentes clave de esta arquitectura incluyen servicios independientes, API Gateway, orquestadores de contenedores como AWS ECS o EKS, bases de datos independientes para cada microservicio y herramientas de monitoreo y seguridad avanzadas como AWS CloudWatch y AWS WAF.

Por otro lado, la arquitectura basada en eventos se centra en la comunicación asíncrona entre componentes del sistema a través de eventos. Estos eventos son mensajes que indican que algo ha ocurrido, y los componentes pueden reaccionar a estos eventos generando nuevos eventos o realizando acciones específicas. EDA se apoya en colas y brokers de eventos para gestionar la comunicación, lo que permite un alto grado de desacoplamiento, asíncronía y flexibilidad en la respuesta a eventos en tiempo real. Los componentes clave de esta arquitectura incluyen productores de eventos, brokers de eventos como Amazon SNS, Kafka o RabbitMQ, consumidores de eventos y procesadores de eventos que realizan acciones basadas en los eventos.

Tabla 13. Comparativa de arquitecturas

Criterio	Arquitectura basada en Microservicios AWS	Event-Driven Architecture
Definición y Concepto General	Una aplicación se divide en servicios pequeños e independientes que se comunican a través de APIs. Cada microservicio está diseñado para realizar una función específica y puede ser desplegado, escalado y actualizado de forma independiente (Newman, 2022).	Los eventos son mensajes que indican que algo ha ocurrido y permiten que los componentes del sistema se comuniquen entre sí, lo que les permite responder a estos eventos y generar nuevos. Esta arquitectura es altamente asíncrona y se basa en la comunicación mediante colas y brokers de eventos. (Fowler, 2022).
Objetivo	Modularidad, escalabilidad, e implementación continua.	Desacoplamiento, asíncronía, y flexibilidad en la respuesta a eventos.
Componentes Clave	Servicios Independientes: Cada servicio realiza una función específica (e.g.,	Productores de Eventos: Generan eventos cuando ocurre algo relevante.

	gestión de usuarios, procesamiento de pagos). API Gateway: Maneja las solicitudes y enruta a los microservicios correspondientes. Orquestación de Contenedores: Utiliza herramientas como AWS ECS o EKS. Bases de Datos Independientes: Cada microservicio puede tener su propia base de datos. Monitoreo y Seguridad: Herramientas como AWS CloudWatch y WAF.	Brokers de Eventos: Servicios como Amazon SNS, Kafka o RabbitMQ que gestionan y distribuyen eventos. Consumidores de Eventos: Escuchan y responden a eventos específicos. Procesadores de Eventos: Realizan acciones basadas en eventos, como actualizar bases de datos o invocar otros servicios.
Ventajas	Cada servicio puede escalarse de forma independiente según su carga. Los equipos pueden trabajar en diferentes microservicios simultáneamente y cada servicio puede usar	La arquitectura tiene componentes desacoplados, lo que aumenta la flexibilidad y la escalabilidad para manejar picos de carga y fallos de manera efectiva. La asincronía de los servicios permite que funcionen de

	diferentes tecnologías o lenguajes de programación (Thönes, 2022).	manera asíncrona, lo que mejora el rendimiento y la capacidad de respuesta (Pardede, 2022).
Desventajas	La configuración de la infraestructura para la gestión de microservicios requiere un gran cuidado. La comunicación entre servicios puede generar latencia y sobrecarga de red, mientras el manejo de datos distribuidos puede ser complejo y desafiante (Wolff, 2022).	Diseñar y gestionar la infraestructura de eventos puede ser complejo. La naturaleza asíncrona y distribuida dificulta el seguimiento o la resolución de problemas. Además, la coherencia de los datos puede ser un desafío (Krzywinski & Altman, 2022).
Adecuación para Casos de Uso	Adecuado para aplicaciones con múltiples funcionalidades que necesitan ser independientes y desplegables de manera continua. Ejemplos: plataformas de comercio electrónico, sistemas de	Ideal para sistemas que requieren una alta respuesta a eventos y acciones en tiempo real. Ejemplos: sistemas de procesamiento de pagos en tiempo real, sistemas de recomendaciones, plataformas de análisis de

	gestión empresarial, aplicaciones de redes sociales. Se adapta bien a aplicaciones que requieren diferentes tipos de servicios con distintas necesidades de escalabilidad (Richardson, 2022).	datos en streaming. Crucial para aplicaciones donde la reacción a eventos y la desacoplación entre servicios son esenciales (Chen, 2022).
Ejemplo Comparativo en el Contexto de Comercio Electrónico	Una plataforma de comercio electrónico con microservicios para gestión de usuarios, procesamiento de pagos y gestión de productos. Cada microservicio puede ser escalado y mantenido de manera independiente.	Un sistema de comercio electrónico en el que los servicios de backend procesan los eventos creados por cada acción del usuario, como agregar un producto al carrito. Los eventos se pueden utilizar para realizar análisis en tiempo real, generar notificaciones y actualizar inventarios.

Esta arquitectura de microservicios en AWS se destaca por su capacidad para manejar aplicaciones complejas con múltiples funcionalidades, cada una de las

cuales puede ser gestionada y escalada de manera independiente. La modularidad no solo permite a las organizaciones escalar sus aplicaciones de manera eficiente, sino que también facilita el desarrollo y la implementación continua, lo que es crucial en entornos dinámicos como el comercio electrónico. Además, la flexibilidad en la elección de tecnologías y la posibilidad de que cada microservicio utilice su propia base de datos ofrecen una ventaja significativa en términos de personalización y optimización de recursos.

En comparación a la arquitectura basada en eventos que ofrece un alto grado de desacoplamiento y flexibilidad, pero su implementación puede ser más compleja y desafiante, especialmente en términos de consistencia de datos y depuración. Mientras que EDA es ideal para aplicaciones que requieren una respuesta rápida a eventos en tiempo real, como sistemas de recomendaciones o análisis de datos en streaming, la arquitectura de microservicios es más adecuada para aplicaciones que necesitan una alta disponibilidad, seguridad avanzada y una gestión eficiente de múltiples servicios independientes.

Es por ello que al comparar una arquitectura basada en microservicios con una arquitectura basada en eventos (Event-Driven Architecture, EDA) puede ser útil para entender cómo cada enfoque maneja diferentes aspectos del desarrollo y operación de aplicaciones, como la escalabilidad, flexibilidad, y manejo de eventos. Ambas arquitecturas tienen sus propias ventajas y son adecuadas para diferentes tipos de proyectos. La elección entre ellas dependerá de las necesidades específicas del proyecto y los objetivos que se deseen alcanzar. Sin embargo, en el contexto de aplicaciones de comercio electrónico que requieren alta escalabilidad, rendimiento y seguridad, la arquitectura de microservicios en AWS demuestra ser superior. Su capacidad para manejar la complejidad de manera eficiente, junto con la flexibilidad

y robustez que ofrece, la convierte en la opción preferida para proyectos de gran envergadura y con altos requisitos de disponibilidad y resiliencia (Cloud Institute, 2023).

6. CONCLUSIONES

Uno de los resultados más relevantes del trabajo de grado es la importancia de entender las características de un negocio en su fase inicial para diseñar una arquitectura tecnológica que esté alineada a sus necesidades. Durante el desarrollo del diagrama se observó que la falta de claridad de estos requisitos y no contemplar cambios a futuros en el diseño arquitectónico puede llevar a resultados ineficientes por falta de una base sólida. Para este desafío, se utilizó un enfoque iterativo basado en la revisión constante de los requisitos establecidos para ajustar los diagramas a medida que avanzaba el proyecto sin problemas.

Asimismo, se identificó que, en la etapa de análisis y elección de las tecnologías disponibles para microservicios y Cloud Computing, aunque existen numerosas opciones con ventajas fuertes y rentables, la sobreabundancia de estas puede dificultar la elección de las tecnologías adecuadas para cumplir con un caso particular, como el manejo de grandes volúmenes de datos durante períodos de alta ocupación. Por ello, se realizó una evaluación comparativa para elegir las tecnologías más apropiadas, teniendo en cuenta los requerimientos técnicos y comerciales identificados inicialmente en base a uno de los problemas más comunes en el comercio electrónico. Esto aseguró que la elección tecnológica estuviera alineada no solo con las necesidades actuales, sino también con la capacidad de adaptación futura del sistema.

En cuanto a la propuesta de la arquitectura y el diagrama planteado para la tesis, se puede decir que demostró ser una solución robusta porque un diseño modular facilita la escalabilidad y el mantenimiento. Sin embargo, este análisis reveló que la arquitectura puesta en marcha podría enfrentar desafíos relacionados con la integración de servicios y la gestión de la complejidad inherente a una arquitectura

distribuida, dado que al ser componentes independientes requiere de mayor coordinación y monitoreo. Por esa razón, se incorporaron de patrones de diseño específicos, como API Gateways y la utilización de servicios de orquestación de contenedores para hacer que sea más manejable y tenga mejor rendimiento sin perder la alineación de los objetivos.

Por último, pero no menos importante, la evaluación de la arquitectura propuesta en términos de escalabilidad, rendimiento, seguridad y adaptabilidad reveló que, aunque la arquitectura es adecuada para manejar las demandas esperadas del sistema, hay áreas que podrían beneficiarse de mejoras adicionales. La identificación de estas facilitó la propuesta de futuras optimizaciones que podrían llevarse a cabo en fases posteriores del desarrollo. Estas conclusiones destacan la necesidad de un enfoque adaptable y dinámico para lograr arquitecturas tecnológicas en un entorno competitivo y cambiante como lo es el comercio electrónico. Dado que la implementación de mejoras continuas garantizará que la arquitectura siga siendo efectiva y relevante a largo plazo.

7. RECOMENDACIONES

Para futuros trabajos, se recomiendan varias prácticas para garantizar la efectividad y el éxito en la implementación de una arquitectura tecnológica robusta y adecuada. En primera instancia, hay que reconocer la importancia de validar la migración a AWS Lambda y Amazon EKS para una estabilidad eficiente y gestionar la carga de trabajo de manera dinámica, ajustando los recursos en función de la demanda y asegurando que los microservicios puedan crecer conforme a las necesidades del negocio. Para la optimización de costos, también es importante considerar el uso de modelos de pago por uso, como los ofrecidos por AWS Lambda y Amazon S3 juntos. En cuanto a la seguridad, es recomendable seguir indagando en soluciones como AWS WAF y Amazon Cognito para proteger las aplicaciones contra accesos no autorizados y ataques.

Finalmente, pero no menos importante, es indispensable observar que el alcance de este trabajo de grado era el diseño de una arquitectura y no centrarse en la implementación directa de la misma, sin embargo, se considera apropiado el complementar este apartado y llevarlo a la parte práctica para validar el comportamiento de la arquitectura en un entorno real, asegurando que la propuesta sea efectiva y esté alineada con los objetivos del negocio. Como recomendación se exalta el llevar a cabo una evaluación detallada de la arquitectura propuesta, realizar pruebas de rendimiento para evaluar el comportamiento de la arquitectura bajo diferentes cargas y condiciones, para permitir ajustar y optimizar el diseño antes de su despliegue completo.

8. REFERENCIAS BIBLIOGRÁFICAS

Soler, A. (2023). *Aplicación SaaS basada en microservicios para la gestión de pedidos en empresas de comercio electrónico*. riunet. https://riunet.upv.es/bitstream/handle/10251/196558/Soler%20-%20SaaS%20Application%20based%20on%20Microservices%20for%20Order%20Management%20in%20E-commerce%20Businesses.pdf?sequence=2&isAllowed=y

Meneses, J. (2019). *Propuesta de un ecosistema tecnológico para la administración en eventos de salud pública, basado en Arquitectura Cloud Computing. Caso de estudio Santander. Repositorio Universidad Autónoma de Bucaramanga*. https://repository.unab.edu.co/bitstream/handle/20.500.12749/3380/2019_Tesis_Jathinson_Meneses_Mendoza.pdf?sequence=1&isAllowed=y

Uribe, C. (2019). *Definición de un modelo de adopción a Cloud Computing adaptable a las PYMES de base tecnológica de la región de Bucaramanga*. Repositorio Universidad Autónoma de Bucaramanga. https://repository.unab.edu.co/bitstream/handle/20.500.12749/7275/2019_Tesis_Camilo_Andres_Uribe_Santos.pdf?sequence=1&isAllowed=y

Quevedo, M. (2018). *Prototipo de almacenamiento de datos sobre cloud computing mediante el uso de herramientas de software libre, para pequeñas y medianas empresas (Pymes)*. Repositorio Universidad Autónoma de Bucaramanga. https://repository.unab.edu.co/bitstream/handle/20.500.12749/3438/2018_Tesis_Martha_Liliana_Quevedo.pdf?sequence=1&isAllowed=y

Moreno, S. (2022). *Modelo de evolución arquitectónica de monolito a microservicios para el sistema de información ISys de la Dirección Nacional de Admisiones de la Universidad Nacional de Colombia*. Universidad Nacional de Colombia. Repositorio Universidad Nacional. https://repositorio.unal.edu.co/bitstream/handle/unal/81788/1020817052.2022.pdf?sequence=1&isAllowed=y

Macedo Pereira, A. (2020). Uso de una arquitectura basada en eventos como capa de comunicación para microservicios. Tesis de maestría, Pontificia Universidad Católica del Perú. https://tesis.pucp.edu.pe/repositorio//handle/20.500.12404/16979

Vijai, C., & Nivetha, P. (2020). *E-commerce on cloud: opportunities and challenges. SSRN*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3758721

Paredes, D. (2021). *Implementación De Microservicios Con Kubernetes Para Mejorar La Escalabilidad En La Arquitectura De La Aplicación En Una Empresa Dedicada Al Comercio Electrónico*. Repositorio Institucional de la Universidad Nacional Tecnológica de Lima Sur. https://repositorio.untels.edu.pe/jspui/bitstream/123456789/876/1/T088A_46524197_T.pdf

Ivanov, K. (2017). *Containerization with LXC*. Packt Publishing. https://www.google.com.co/books/edition/Containerization_with_LXC/vVMoDwAAQBAJ?hl=es-419&gbpv=1

Grau, M. (2023). Análisis de la eficiencia energética en edificios residenciales: Un estudio de caso en Barcelona (Trabajo de fin de grado). Universitat Politècnica de Catalunya.

https://upcommons.upc.edu/bitstream/handle/2117/396645/Degree_thesis_Marc_Grau.pdf?sequence=3&isAllowed=y

Shaw, M., & Garlan, D. (1996). *Software architecture: Perspectives on an emerging discipline*. Digital library. https://dl.acm.org/doi/abs/10.5555/231003

Garlan, D., & Perry, D. (1995). Introduction to the special issue on software architecture. *ACM Transactions on Software Engineering and Methodology, 4*(4), 275-286. https://doi.org/10.1145/224591.224593

Lewis, J., & Fowler, M. (2014). *Microservices*. martinfowler. https://martinfowler.com/articles/microservices.html

Kamal, M. A., Raza, H. W., Alam, M. M., & Mohd, M. (2020). *Highlight the features of AWS, GCP and Microsoft Azure that have an impact when choosing a cloud service provider*. researchgate. https://www.researchgate.net/profile/Muhammad-Ayoub-Kamal/publication/340173446_Highlight_the_Features_of_AWS_GCP_and_Microsoft_Azure_that_Have_an_Impact_when_Choosing_a_Cloud_Service_Provider/links/5e7c397c92851caef49d994f/Hig2hlight-the-Features-of-AWS-GCP-and-Microsoft-Azure-that-Have-an-Impact-when-Choosing-a-Cloud-Service-Provider.pdf.

Newman, S. (2021). Building Microservices. Estados Unidos: O'Reilly Media. https://books.google.com.co/books?hl=es&lr=&id=ZvM5EAAAQBAJ&oi=fnd&pg=PT8&dq=S.+Newman.+Building+Microservices.&ots=ui5jdvcGTt&sig=UHYRP0LItriTCgB3BSe112ftyns&redir_esc=y#v=onepage&q=S.%20Newman.%20Building%20Microservices.&f=false

Mathew, S. (2014). Overview of Amazon Web Services. https://www.sysfore.com/Assets/PDF/aws-overview.pdf

Christensen, H. B. (2023). Software Architecture in Practice. [SNIPPET] Software Architecture in Practice. Page 1. https://edisciplinas.usp.br/pluginfile.php/5922722/mod_resource/content/1/2013%20-%20Book%20-%20Bass%20%20Kazman-Software%20Architecture%20in%20Practice%20%281%29.pdf

Software Engineering Institute. (2023). Software Architecture. Retrieved from https://resources.sei.cmu.edu/asset_files/FactSheet/2010_010_001_513810.pdf

Andersson, J. C. (2023). Learning Microsoft Azure. " O'Reilly Media, Inc.".

Docker, I. (2020). Docker. https://www.docker.com/what-docker.

https://books.google.com.co/books?hl=es&lr=&id=II73rBDXCYC&oi=fnd&pg=PT15&dq=Bass,+Clements+y+Kazman+(2012)+&ots=ZpRAQFjuVC&sig=73riX62K1GnGHii6g44M6xzP4aE&redir_esc=y#v=onepage&q=Bass%2C%20Clements%20y%20Kazman%20(2012)&f=false

Rodríguez, J. M. (2020). El impacto de la inteligencia artificial en la educación superior. Revista Iberoamericana de Ciencia y Tecnología, 15(2), 45-62. https://doi.org/10.4108/eai.27-2-2020.2303255

Bass, L., Clements, P., & Kazman, R. (2020). Software Architecture in Practice (4th ed.). Addison-Wesley. https://www.oreilly.com/library/view/software-architecture-in/9780136886039/

Burns, B., Beda, J., & Hightower, K. (2019). Kubernetes: Up and Running: Dive into the Future of Infrastructure (2ª ed.). O'Reilly Media. https://www.oreilly.com/library/view/kubernetes-up-and/9781492046526

https://eddiejackson.net/azure/Kubernetes_book.pdf

Garrison, J., & Nova, K. (2019). Cloud Native Infrastructure: Patterns for Scalable Infrastructure and Applications in a Dynamic Environment. O'Reilly Media. https://www.oreilly.com/library/view/cloud-native-infrastructure/9781491973847

Burns, B., Beda, J., Hightower, K., & Evenson, L. (2022). *Kubernetes: up and running: Dive into the Future of Infrastructure* (3ª ed.). O'Reilly Media. https://books.google.com.co/books?hl=es&lr=&id=KeB-EAAAQBAJ&oi=fnd&pg=PT17&dq=Kubernetes:+Up+and+Running&ots=VaOTCLnpYc&sig=r1Sb2-iM7EMi8MmNqJBvmUrFKc0&redir_esc=y#v=onepage&q&f=false

Burns, B., Beda, J., & Hightower, K. (2021). Kubernetes Best Practices: Blueprints for Building Successful Applications on Kubernetes. O'Reilly Media. https://www.oreilly.com/library/view/kubernetes-best-practices/9781492056478

The Cloud Native Computing Foundation (CNCF). (2020). Kubernetes Annual Report 2020. CNCF. https://www.cncf.io/wp-content/uploads/2021/01/Kubernetes-Annual-Report-2020.pdf

Adams, R. (2020). Cloud Computing: Concepts, Technology & Architecture. Prentice Hall. (Nota: Este autor no parece estar relacionado con este título en 2020, verificar si es correcto.)

Basiri, A., Heydarnoori, A., & Hassanzadeh, N. (2019). Challenges of Microservice Architecture: A Survey on the State of the Practice. IEEE Software, 36(2), 75-81. https://ieeexplore.ieee.org/document/8652203

Fowler, M., & Lewis, J. (2019). Microservices: A Definition of This New Architectural Term. MartinFowler.com. https://martinfowler.com/articles/microservices.html

Kritikos, K., & Plexousakis, D. (2020). Towards Cloud-based Elasticity Control for Service Applications. Future Generation Computer Systems, 100, 752-762. https://www.sciencedirect.com/science/article/abs/pii/S0167739X1831043X

Newman, S. (2019). Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith. O'Reilly Media. https://www.oreilly.com/library/view/monolith-to-microservices/9781492047834/

Sharma, N., Gupta, A., & Singh, A. (2021). Transitioning from Monolith to Microservices: A Case Study. Journal of Cloud Computing, 10(1), 33-48. https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-021-00230-x

Taibi, D., Systä, T., & Martinez-Fernandez, S. (2020). Observability in Microservice Architectures. Journal of Systems and Software, 170, 110719. https://www.sciencedirect.com/science/article/abs/pii/S0164121220301875

Villamizar, M., Garcés, O., Ochoa, L., Castro, H., Salamanca, L., Verano, M., Casallas, R., Gil, S., Valencia, C., Zambrano, A., & Lang, M. (2019). Cost Comparison of Running Web Applications in the Cloud Using Monolithic, Microservice, and Serverless Architectures. IEEE Access, 7, 5600-5613. https://ieeexplore.ieee.org/document/8624564

Bass, L., Weber, I., & Zhu, L. (2020). DevOps: A Software Architect's Perspective. Addison-Wesley. https://www.pearson.com/store/p/devops/P100000693853/9780134049849

Di Francesco, P., Malavolta, I., & Lago, P. (2020). Research on Architecting Microservices: Trends, Focus, and Potential for Industrial Adoption. IEEE Software, 35(3), 44-52. https://ieeexplore.ieee.org/document/8453246

Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustafin, R., & Safina, L. (2020). Microservices: Yesterday, Today, and Tomorrow. Present and Ulterior Software Engineering, 195-216. https://link.springer.com/chapter/10.1007/978-3-030-00262-6_9

Fitzgerald, B., & Stol, K. J. (2020). Continuous Software Engineering: A Roadmap and Agenda. Journal of Systems and Software, 123(4), 176-189. https://www.sciencedirect.com/science/article/abs/pii/S0164121218302623

Garg, S., & Versteeg, S. (2021). Cloud Computing: Principles and Paradigms. Wiley. https://www.wiley.com/en-us/Cloud+Computing%3A+Principles+and+Paradigms-p-9781118002209

Gartner. (2020). Hype Cycle for Cloud Security. Gartner Research. https://www.gartner.com/en/documents/3984962/hype-cycle-for-cloud-security-2020

Jamsa, K. (2020). Cloud Computing: SaaS, PaaS, IaaS, Virtualization, Business Models, Mobile, Security and More. Jones & Bartlett Learning. https://www.amazon.com/Cloud-Computing-Virtualization-Business-Security/dp/1284091506

Kim, H., Woo, J., & Lee, J. (2020). Container-based Cloud Platform for Running Microservices. Journal of Cloud Computing, 9(1), 14-29. https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-020-00160-4

Marinescu, D. C. (2020). Cloud Computing: Theory and Practice. Morgan Kaufmann. https://www.elsevier.com/books/cloud-computing/marinescu/978-0-12-812810-7

Universidad Nacional de Colombia. (2022). Impacto del Cloud Computing en la Competitividad de las PYMEs. Revista Colombiana de Computación.

The Journal of Cloud Computing. (2021). Cloud Technologies and their Impact on Small and Medium Enterprises. https://journalofcloudcomputing.springeropen.com/

Amazon Web Services (AWS) Documentation. Best Practices and Reference Architectures. https://docs.aws.amazon.com/

Newman, S. (2019). Building Microservices: Designing Fine-Grained Systems. O'Reilly Media. https://www.oreilly.com/library/view/building-microservices/9781491950340/

Fowler, M., & Lewis, J. (2019). Microservices: A Definition of This New Architectural Term. MartinFowler.com. https://martinfowler.com/articles/microservices.html

Erl, T., Puttini, R., & Mahmood, Z. (2013). Cloud Computing: Concepts, Technology & Architecture. Prentice Hall. https://www.pearson.com/store/p/cloud-computing-concepts-technology-architecture/P100000150007/9780133387522

Chowdary Desina, G. (2023). Evaluating the Impact of Cloud-Based Microservices Architecture on Application Performance. arXiv.

Hasselbring, W. (2020). Microservices for Scalability: Keynote Talk Abstract. Communications in Computer and Information Science.

Newman, S. (2021). Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith. O'Reilly Media.

Richardson, C. (2021). Microservices Patterns: With Examples in Java. Manning Publications.

Chen, C. (2022). Event-Driven Architecture Patterns. O'Reilly Media.

Fowler, M. (2022). Patterns of Event-Driven Architecture. ThoughtWorks.

Krzywinski, M., & Altman, N. (2022). Best Practices in Asynchronous System Design. Nature Methods, 19(3), 215-221.

Newman, S. (2022). Building Microservices: Designing Fine-Grained Systems. O'Reilly Media.

Pardede, E. (2022). Scalability and Resilience in Event-Driven Systems. IEEE Software, 39(5), 14-18.

Richardson, C. (2022). Microservices Patterns: With examples in Java. Manning Publications.

Thönes, J. (2022). Microservices: An Overview. IEEE Software, 33(1), 113-116.

Wolff, E. (2022). Microservices: Flexible Software Architecture. Addison-Wesley.

https://ieeexplore.ieee.org/document/8624564

Constitución Política de Colombia 1991. https://www.sic.gov.co/manejo-de-informacion-personal#:~:text=%C2%BFQu%C3%A9%20es%20el%20derecho%20de,de%20naturaleza%20p%C3%BAblica%20o%20privada.

9. APÉNDICES

10. ANEXOS