

**UNIVERSIDAD PERUANA LOS ANDES**

**FACULTAD DE INGENIERÍA**

Escuela Profesional de Ingeniería de Sistemas y Computación



**TESIS**

**Proceso de Desarrollo de Software basado en  
DevOps para las aplicaciones Web de una  
institución Financiera**

**PRESENTADO POR:**

**Bach. ANTÓN SEBASTIÁN, MAGALLY EDITH**

**Línea de Investigación de la Universidad:**

**Nuevas Tecnologías y Procesos**

**Línea de Investigación de la Escuela Profesional:**

**Ingeniería de Software**

**PARA OPTAR TÍTULO PROFESIONAL DE:**

**INGENIERA DE SISTEMAS Y COMPUTACIÓN**

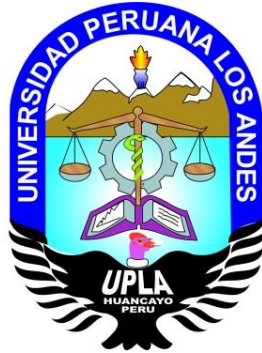
**LIMA- PERÚ**

**2019**

**UNIVERSIDAD PERUANA LOS ANDES**

**FACULTAD DE INGENIERÍA**

Escuela Profesional de Ingeniería de Sistemas y Computación



**TESIS**

**Proceso de Desarrollo de Software basado  
en DevOps para las aplicaciones Web de  
una institución Financiera**

**PRESENTADO POR:**

**Bach. ANTÓN SEBASTIÁN, MAGALLY EDITH**

**PARA OPTAR TÍTULO PROFESIONAL DE:  
INGENIERA DE SISTEMAS Y COMPUTACIÓN**

**LIMA- PERÚ  
2019**

## **ASESORES**

---

### **ASESOR METODOLÓGICO**

MG. LUIS ALBERTO TORRES CABANILLAS

---

### **ASESOR TEMÁTICO**

DRA. KARIN CORINA ROJAS ROMERO

*Dedico este trabajo a mi familia, por su comprensión y apoyo incondicional a lo largo de mi carrera profesional y en cada etapa de mi vida.*

## **AGRADECIMIENTO**

Este informe es el resultado de mi trabajo de investigación de tesis realizado en el período de 2019, como parte de mi estudio en la carrera de Computación e Informática en la Universidad Peruana de los Andes. Estoy muy satisfecha con el progreso y el resultado de este estudio, que no fue posible sin la cooperación y el apoyo de varias personas y sus organizaciones. Me complace tomar esto como una oportunidad para agradecerles su asistencia y contribución para este estudio.

Debo mi más profunda gratitud a Dios, por ser la fuerza más grande que me motiva e impulsa a seguir avanzando y aprovechando cada nueva oportunidad que se presenta.

Gracias a mi Asesora, la Dra. Karin Rojas, quien me ha apoyado de muchas maneras. Sin su aliento, confianza en mi trabajo y retroalimentación oportuna y constructiva, este estudio apenas se habría completado.

Así mismo estoy agradecida a mi asesor, el Dr. Luis Torres, por haber ofrecido su ayuda en mi tesis cuando lo pedí.

Es un honor para mí reconocer las diversas ayudas que he recibido de todos los miembros de mi equipo de trabajo, les agradezco sus sugerencias y comentarios a lo largo de la planificación y el desarrollo de este trabajo de tesis. Su buena disposición para dedicar su tiempo tan generosamente ha sido muy apreciada.

Un agradecimiento muy especial a mi madre y hermanas, por brindarme la motivación y el entusiasmo durante todo el período de esta tesis. Por último, no puedo agradecer lo suficiente a mi esposo, Marco, por su inestimable amor, aliento y apoyo incomparable en todas las formas posibles a lo largo del período de mi trabajo de tesis.

## **JURADOS DE SUSTENTACIÓN**

---

### **PRESIDENTE**

DR.CASIO AURELIO TORREZ LÓPEZ

---

### **PRIMER JURADO**

ING. JESSICA VILCHEZ GUTARRA

---

### **SEGUNDO JURADO**

ING. RAFAEL EDWIN GORDILLO FLORES

---

### **TERCER JURADO**

ING. ALEX ALBERT ZUÑIGA MANRQUE

---

### **SECRETARIO DOCENTE**

MG.. MIGUEL ÁNGEL CARLOS CANALES

## ÍNDICE

1. FALSA PORTADA.....	ii
2. HOJA CON EL NOMBRE DEL ASESOR.....	iii
3. DEDICATORIA Y AGRADECIMIENTO.....	iv
4. HOJA DE CONFORMIDAD DE LOS JURADOS.....	vi
5. ÍNDICE.....	vii
5.1.ÍNDICE DE TABLAS.....	x
5.2.ÍNDICE DE FIGURAS.....	xi
6. RESUMEN.....	xiv
ABSTRACT.....	xv
7. INTRODUCCIÓN.....	xvi

## CAPÍTULO I

<b>EL PROBLEMA DE INVESTIGACIÓN.....</b>	<b>17</b>
1.1. Planteamiento del problema.....	17
1.2. Formulación y sistematización del problema.....	18
1.2.1. Problema General.....	18
1.2.2. Problemas Específicos.....	18
1.3. Justificación.....	19
1.3.1. Práctica.....	19
1.3.2. Metodológica.....	19
1.4. Delimitaciones.....	20
1.4.1. Espacial.....	20
1.4.2. Temporal.....	20
1.4.3. Económica.....	20
1.5. Limitaciones.....	20
1.6. Objetivos.....	20
1.6.1. Objetivo General.....	20
1.6.2. Objetivos Específicos.....	21

## **CAPÍTULO II**

<b>MARCO TEÓRICO.....</b>	<b>22</b>
2.1. Antecedentes.....	22
2.2. Marco conceptual.....	24
2.3. Definición de términos.....	43
2.4. Hipótesis.....	47
2.4.1. Hipótesis General.....	47
2.4.2. Hipótesis Específicas.....	47
2.5. Variables.....	48
2.5.1. Definición Conceptual de la variable.....	48
2.5.2. Definición operacional de la variable.....	48
2.5.3. Operacionalización de la variable.....	48

## **CAPÍTULO III**

<b>METODOLOGÍA.....</b>	<b>50</b>
3.1. Método de investigación.....	50
3.2. Tipo de investigación.....	50
3.3. Nivel de investigación.....	50
3.4. Diseño de investigación.....	51
3.5. Población y muestra.....	51
3.6. Técnicas e instrumentos de recolección de datos.....	51
3.7. Procesamiento de información.....	52
3.8. Técnicas y análisis de datos.....	52

## **CAPÍTULO IV**

<b>RESULTADOS.....</b>	<b>58</b>
4.1. Estadística Descriptiva.....	58
4.2. Análisis Inferencial - Prueba de Hipótesis.....	64



## **CAPÍTULO V**

<b>DISCUSIÓN DE RESULTADOS.....</b>	<b>69</b>
<b>CONCLUSIONES.....</b>	<b>70</b>
<b>RECOMENDACIONES.....</b>	<b>71</b>
<b>REFERENCIAS BIBLIOGRÁFICAS.....</b>	<b>72</b>
<b>ANEXOS.....</b>	<b>74</b>
Anexo N° 01: MATRIZ DE CONSISTENCIA.....	75
Anexo N° 02: INSTRUMENTO DE INVESTIGACIÓN.....	77
Anexo N° 04: BASE DE DATOS DEL PRE TEST Y POST TEST.....	83
Anexo N° 05: TECNOLOGÍAS Y SERVICIOS.....	85

## ÍNDICE DE TABLAS

Tabla 01. Operacionalización de la variable Dependiente.....	49
Tabla 2. Validez del instrumento según expertos.....	54
Tabla 3. Escala de valores de confiabilidad.....	55
Tabla 4. Aplicativos Web (Pre Test) - Estadística Descriptiva.....	58
Tabla 5. Rendimiento del Proceso (Pre Test) - Estadística Descriptiva.....	59
Tabla 6. Calidad del Proceso (Pre Test) - Estadística Descriptiva.....	60
Tabla 7. Aplicativos Web (Post Test) - Estadística Descriptiva.....	61
Tabla 8. Rendimiento del Proceso (Post Test) - Estadística Descriptiva.....	62
Tabla 9. Calidad del Proceso (Post Test) - Estadística Descriptiva.....	63

## ÍNDICE DE FIGURAS

Figura 1. Procesos que deben estar integrados en DevOps.....	25
Figura 2. Arquitecturas de aplicación monolítica y microservicio.....	27
Figura 3. Etapas del ciclo de vida DevOps.....	28
Figura 4. Modelo de Madurez DevOps – Mohamed.....	29
Figura 5. Modelo de Madurez DevOps – Bucena y Kirikova.....	30
Figura 6. Modelo de Madurez DevOps - Feijter.....	33
Figura 7. Métricas y dimensiones DevOps.....	34
Figura 8. Áreas del proceso de Ingeniería de Software.....	42
Figura 9. Puntos claves de las organizaciones.....	42
Figura 10. Confiabilidad del instrumento – Alfa de Cronbach - Pre Test.....	55
Figura 11. Confiabilidad del instrumento – Alfa de Cronbach - Post Test.....	56
Figura 12. Aplicativos Web (Pretest) - Estadística Descriptiva.....	58
Figura 13. Rendimiento del proceso (Pretest) - Estadística Descriptiva.....	59
Figura 14. Calidad del Proceso (Pretest) - Estadística Descriptiva.....	60
Figura 15. Aplicaciones web (Post test) - Estadística Descriptiva.....	61
Figura 16. Rendimiento del proceso (Post test) - Estadística Descriptiva.....	62
Figura 17. Calidad del proceso (Post test) - Estadística Descriptiva.....	63
Figura 18. Resultados Wilcoxon para el Aplicativo Web.....	64
Figura 19. Resultados de Wilcoxon para el Rendimiento del Proceso .....	66
Figura 20. Resultados de Wilcoxon para la Calidad del aplicativo.....	68
Figura 21. Aspecto de función Lambda.....	85
Figura 22. Aspecto de función Lambda II.....	87
Figura 23. Puntos de enlace para flujo DynamoDB.....	88
Figura 24. Exportando datos de DynamoDB hacia Amazon S3.....	89
Figura 25. Importando datos de Amazon S3 hacia DynamoDB.....	89
Figura 26. Componentes principales de Amazon Cognito.....	90
Figura 27. Acceso a Recursos con API Gateway y Lambda.....	90
Figura 28. Proceso de Confirmación de una cuenta de usuario en Cognito.....	91
Figura 29. Creación de usuarios en Cognito.....	92
Figura 30. Contenido de usuarios en Cognito.....	92
Figura 31. Creación de grupos en Cognito.....	92

Figura 32. Creación de grupos en Cognito II.....	93
Figura 33. Flujo de autenticación Cognito.....	93
Figura 34. Creación de bucket S3.....	94
Figura 35. Control de versiones en S3.....	94
Figura 36. Flujo de Configuración CloudFront.....	95
Figura 37. Funciones Lambda con CloudFront.....	96
Figura 38. Eventos de CloudFront para disparar funciones Lambda.....	96
Figura 39. Flujo redirección de tráfico al dominio con Route 53.....	97
Figura 40. Comprobación del estado del recurso con Route 53.....	98
Figura 41. Métricas CloudWatch admitidas por ACM PCA.....	99
Figura 42. Líneas de código 1 – Configuración React.....	100
Figura 43. Demo API Pagos.....	113
Figura 44. Estructura configuración raíz Seed.....	114
Figura 45. Administración de servicios y escenarios Seed.....	114
Figura 46. Interfaz cuenta Netlify.....	115
Figura 47. Configuración de cuentas asociadas en Netlify.....	115
Figura 48. Configuración de cuentas asociadas GitHub.....	115
Figura 49. Repositorios en Github.....	116
Figura 50. Pantalla base de Login.....	117
Figura 51. Opción de Registro y autenticación.....	117
Figura 52. Pantalla con campos de Autenticación Login.....	118
Figura 53. Validación de Login.....	118
Figura 54. Opción de Logout.....	119
Figura 55. Pantalla de Registro de usuario.....	120
Figura 56. Pantalla de código de verificación luego del Signup.....	120
Figura 57. Código de verificación recibido.....	121
Figura 58. Ingreso de código de verificación.....	121
Figura 59. Validación del código de verificación.....	121
Figura 60. Funcionalidad de ingreso de notas de la aplicación.....	122
Figura 61. Opción de creación de nueva nota.....	122
Figura 62. Ventana de carga de archivos.....	122
Figura 63. Nota con archivo adjunto.....	123
Figura 64. Procesamiento de creación de la nota.....	123

Figura 65. Vista de nota creada.....	123
Figura 66. Ingreso al detalle de la nota creada.....	124
Figura 67. Opciones del detalle de nota creada.....	124
Figura 68. Procesamiento Guardar nota.....	124
Figura 69. Opción eliminar nota.....	125
Figura 70. Confirmación de eliminación de nota .....	125
Figura 71. Pantalla sin notas.....	125
Figura 72. Escenarios del flujo de trabajo.....	126
Figura 73. Servicios contenidos en un Escenario Productivo.....	126

## RESUMEN

El trabajo de investigación tuvo como problema general ¿En qué medida el proceso de desarrollo de software basado en DevOps mejora la obtención de aplicaciones web en una institución Financiera?, el objetivo general fue determinar en qué medida el proceso de desarrollo de software basado en DevOps mejora la obtención de aplicaciones web en una institución Financiera y la hipótesis general que se verificó fue: “El proceso de desarrollo de software basado en DevOps mejora la obtención de las aplicaciones web en una institución Financiera”.

El método general de investigación fue el científico, con enfoque cuantitativo, tipo de investigación aplicada, nivel explicativo, diseño pre experimental de corte longitudinal. La población estuvo conformada por 10 trabajadores del equipo de desarrollo de la entidad Inteligo.Bank., no se realizó la técnica del muestreo debido a que la población es pequeña.

Los resultados validaron como cierta la hipótesis “El proceso de desarrollo de software basado en DevOps mejora la obtención de los aplicativos web en una institución Financiera”, ya que existió percepción de mejora en los encuestados, existiendo diferencia significativa en el Post test en relación al Pre test, con un  $p\text{valor} < 0.05$

Palabras clave: DevOps, aplicación web, tecnología.

## ABSTRACT

The research work had as a general problem: To what extent does the DevOps-based software development process improve the obtaining of web applications in a financial institution? The general objective was to determine to what extent the DevOps-based software development process improves the obtaining of web applications in a Financial Institution and the general hypothesis that was verified was: “The software development process based on DevOps improves the obtaining of web applications in a Financial Institution”.

The general research method was the scientist, with quantitative approach, type of applied research, explanatory level, pre-experimental design of longitudinal cut. The population was made up of 10 workers from the Inteligo Bank development team. The sampling technique was not performed because the population is small.

The results validated as true the hypothesis “The software development process based on DevOps improves the obtaining of the web applications in a Financial institution”, since there was a perception of improvement in the respondents, there being significant difference in the Post test in relation to the Pre test, with a value  $<0.05$

Keywords: DevOps, web application, technology.

## INTRODUCCIÓN

En un mundo empresarial cada vez más competitivo, complejo y ambiguo, que compite en un entorno dominado por la continua evolución de los canales digitales y dispositivos móviles, la única estrategia manejable es adaptarse para sobrevivir. Por ello el motivo de la presente investigación tiene como propósito implementar DevOps en el proceso de Desarrollo de Software de las aplicaciones web de una institución Financiera.

La investigación consta de cinco capítulos organizados de la siguiente manera:

En el Capítulo I se desarrolló el Problema de Investigación, detalla la realidad existente y enfoca la situación del problema, la justificación del estudio, la delimitación del problema, así como los objetivos logrados.

En el Capítulo II se desarrolló el Marco Teórico, en este capítulo se describen los antecedentes, iniciando con antecedente internacionales y terminando con los nacionales los cuales sirvieron como guía para el desarrollo de la investigación. Luego pasamos al marco conceptual que son conocimientos y términos de la investigación, la definición de términos, el planteamiento de la hipótesis y las variables de investigación.

En el Capítulo III se desarrolló la Metodología, que comprende una breve descripción del tipo, nivel y diseño de investigación; luego se presenta la población y muestra, las técnicas e instrumentos de recolección de datos, el procesamiento de información y finalmente las técnicas y análisis de datos.

En el Capítulo IV se presentan los Resultados, capítulo en el que se analizan los resultados obtenidos.

En el Capítulo V se presenta la discusión de los resultados para poder validar la hipótesis general.

Finalmente se presentan las conclusiones, las recomendaciones, las referencias bibliográficas y los anexos. En los anexos se encuentra un mayor detalle de las herramientas disponibles y que forman parte de una amplia cantidad de artefactos a disponer para este tipo de implementación.



# CAPÍTULO I

## EL PROBLEMA DE INVESTIGACIÓN

### 1.1 Planteamiento del Problema

“El acelerado avance de las tecnologías de información y las redes informáticas como Internet ha facultado que ésta, se convierta en una herramienta empresarial incalculable debido a la conectividad casi universal que ofrece” (Mahmood, 2009, p.30).

A nivel mundial, la revolución originada por la prestación de servicios a través de la red, transformó el campo de la banca electrónica, que es un medio innovador para la comunicación entre la institución financiera y el usuario final; con ello creció la necesidad de repotenciar las aplicaciones web, para llegar a cada uno de los principales bancos a nivel mundial, tal es así que las instituciones financieras, tienen como reto, ofrecer disponibilidad de sus servicios con entregas más rápidas y asegurando la calidad de sus productos.

Para definir la trascendencia de las aplicaciones, Ravichandran, Taylor, Waterhouse, sostiene que:

En la economía de aplicaciones la transformación de hoy está dada por la evolución de los canales digitales, muchas organizaciones se están proyectando como proveedores de software y servicios digitales. Se prevé que la población mundial de usuarios de banca móvil se duplicará para el 2020 a 1.800 millones, lo que representa más del 25 por ciento de la población mundial. Como resultado, los bancos se centran cada vez más en el avance de las aplicaciones basadas en web y móviles. (2016, p. 5).

Todo aquello se traduce en tecnologías que cambian rápidamente y que han alcanzado todos los ámbitos locales, en el que no somos ajenos; tecnologías donde el cliente se ha convertido en parte vital que se necesita satisfacer y donde las empresas financieras necesitan desarrollar formas innovadoras, implantando nuevas arquitecturas y herramientas que ayudarán a lograr sus objetivos.

Uno de los principales problemas que se tenía en el proceso de desarrollo de los aplicativos de la institución financiera Inteligo.Bank, se generaba al estar el área de desarrollo totalmente aislada del área de operaciones, lo que conllevaba a la generación de una enorme traba en el despliegue de los productos terminados, ya que existían pérdidas de tiempo por coordinación, comités, listas y “pulls” de paso a producción, a esto debemos agregar que no existía un control de versiones óptimo, además de la existencia de una gran cantidad de errores que se presentaban en el despliegue del producto, desde los ambientes previos hacia el ambiente productivo, a consecuencia de la desintegración en los equipos de personas involucradas .

Lo que se logró con esta investigación fue demostrar el efecto de mejora que significó implantar DevOps en el proceso de desarrollo de los aplicativos web, y lo beneficioso del resultado obtenido para sus dimensiones rendimiento del proceso y calidad del proceso, al adoptar la integración continua y el despliegue continuo dentro del proceso y tomando como referencia los Modelos CMMI (Capability Maturity Model Integration) que nos provee de un glosario de buenas prácticas que tratan las actividades de desarrollo aplicadas a productos y servicios.

Para demostrar estos resultados se utilizó como instrumento, el uso de cuestionarios que midieron las principales dimensiones de los aplicativos web relacionados con el proceso de su desarrollo basado en DevOps, por lo que se entenderá que el instrumento hace énfasis en la evaluación de los resultados para la obtención del producto.

## **1.2 Formulación y sistematización del Problema**

### **1.2.1. Problema General**

¿En qué medida el proceso de desarrollo de software basado en DevOps mejora la obtención de aplicaciones web en una institución Financiera?

### **1.2.2. Problemas Específicos**

a. ¿Cuál es el efecto del proceso de desarrollo de software basado en DevOps para la optimización en el rendimiento del proceso de las

aplicaciones web de una institución financiera?

- b. ¿Cómo influye el proceso de desarrollo de software basado en DevOps en la optimización de la calidad del proceso de las aplicaciones web de una institución Financiera?

### **1.3 Justificación**

#### **1.3.1. Práctica**

Tafur Raúl e Izaguirre Manuel, define:

“Son usos prácticos: el uso de instrumentos para resolver problemas de índole técnica, por ejemplo, resolver problemas humanos, diseñar textos, representar actividades, etc.” (2014, p.117).

La investigación planteada sirve para señalar las pautas en base a un enfoque heurístico en la implementación de DevOps en una institución financiera de Lima Metropolitana para ayudar en la gestión de sus procesos agilizando de esta manera la producción del software y puesta en producción del mismo.

#### **1.3.2. Metodológica**

Tafur Raúl e Izaguirre Manuel, definen:

“La justificación metodológica se realiza en razón que el investigador propone como novedad o aporte la formulación de un nuevo método técnica, sea para el conocimiento de la realidad, para la transformación en conjunto de fenómenos para viabilizar un nuevo acceso a la realidad” (2014, p. 117).

Los resultados de la investigación se traducirán como una guía de Implementación de Devops que podría ser utilizada por cualquier tipo de negocio a fin de reestructurar sus procesos lo que se traduciría en entregas efectivas y rápidas de sus productos.

Así mismo el estudio ayudará a tener claro entendimiento de la relación existente entre las metodologías Ágiles y DevOps con el único objetivo de generar valor a los procesos y con ellos a los negocios.

## **1.4 Delimitación del Problema**

### **1.4.1 Delimitación Espacial**

El estudio llevado a cabo tuvo como entorno de aplicación, los equipos de desarrollo de software que se encargan de la puesta en marcha de los aplicativos web de una institución financiera ubicada en Av.Rivera Navarrete 501, Edificio Capital, Piso 17, San Isidro - Lima

### **1.4.2 Delimitación Temporal**

La investigación propuesta aplicará en el primer periodo del año 2019, desde enero hasta abril. En este periodo de tiempo se fueron identificando los principales problemas existentes que perjudicaban la entrega y puesta en producción de los aplicativos web.

### **1.4.3 Delimitación Económica**

Esta investigación fue realizada con recursos propios.

## **1.5 Limitaciones**

Este punto refiere a cuan factible es la aplicación de los resultados a otros entornos. El alcance de la presente investigación en todas las fases del proyecto abarca a instituciones financieras comparables en tamaño. Así mismo tomamos como marco de referencia el grupo correspondiente al equipo de desarrollo de aplicativos web de una institución financiera, con conocimientos y experiencia en el giro del negocio financiero.

## **1.6 Objetivos**

### **1.6.1. Objetivo General**

Determinar en qué medida el proceso de desarrollo de software basado en DevOps mejora la obtención de las aplicaciones web en una institución Financiera.

### **1.6.2. Objetivos Específicos**

- a. Precisar el efecto del proceso de desarrollo de software basado en DevOps para la optimización de rendimiento del proceso de las aplicaciones web de una institución Financiera.
  
- b. Definir cómo influye el proceso de desarrollo de software basado en DevOps en la optimización de la calidad del proceso de las aplicaciones web de una institución Financiera.

## CAPÍTULO II

### MARCO TEÓRICO

#### 2.1. Antecedentes

**DEVI (2018)**, en su tesis titulada “**DevOps implementation framework for Agile-based large financial organizations.**” **Utecht Trecht University**. El objetivo es desarrollar un marco de implementación que se adecúe a la implementación del desarrollo ágil. Utiliza el método Cuantitativo incluyendo la revisión de literatura a fin de comprender los principios y organización de impulsores a fin de adoptarlos. El resultado de la Tesis es mostrar diferentes artefactos que responden preguntas que resultan de utilidad para la evaluación de cuán importante es la relación entre la implementación de DevOps y Agile y el impacto directo que esto tiene sobre la obtención de los objetivos de la organización. El aporte de esta tesis radica en la importancia que cumple la transformación digital dentro de las organizaciones utilizando Agil y DevOps.

**LWAKATAREE (2017)**. En su tesis titulada “**DevOps adoption and implementation in software development practice. Concept, practices, benefits and challenges**”. **University of Oulu**. El objetivo general es detallar las prácticas, beneficios y retos del enfoque DevOps. Los métodos utilizados para el estudio son: mapeo sistemático, revisión multivocal de la literatura, investigación de casos de estudio. Estos métodos de investigación fueron seleccionados por el autor al decidir que son los que mejor se ajustan por el tipo de análisis cualitativo. Los resultados muestran como la adopción de implementación de DevOps es aplicable a la mayoría de las arquitecturas sin discriminar por el tamaño de los Proyectos. Lo que esta tesis aporta a la presente investigación es la comprensión conceptual de DevOps, y sus principales características.

**PEKKI (2017)**, en su tesis titulada “**Implementing modern DevOps development environment for training Case: N4S@JAMK**”. **JAMK University of Applied Sciences**. El objetivo de la tesis fue explorar las últimas tecnologías con lo cual

poder desarrollar un entorno DevOps novedoso. La metodología es de enfoque Cuantitativo, método deductivo, tipo aplicada, diseño Pre experimental. El resultado de la tesis es la brindar los pasos necesarios para la configuración de un entorno de desarrollo con herramientas open source. Esta tesis aporta en la investigación brindando un modelo para iniciar la implementación en un entorno correctamente estabilizado.

**DE FEIJTER (2017). En su tesis: “Towards the adoption of DevOps in software product organizations: A maturity model approach”. Utrecht University.** Esta investigación tiene como objetivo la creación de un modelo de competencia DevOps desde una Perspectiva amplia, pero que proporciona detalles suficientes para dar a las empresas que desarrollan productos de software, una comprensión de las áreas en las que concentrarse para adoptar DevOps. La metodología es de enfoque Cuantitativo, método deductivo, tipo aplicada, diseño Pre experimental. El resultado de la tesis es la obtención de un modelo de competencia basado en las perspectivas, áreas de enfoque y capacidades de DevOps. Esta tesis aporta un nuevo modelo de implementación de DevOps dirigido a las empresas cuyo producto es el software.

**FIERRO (2015). En su tesis: “Heurísticas para Evaluar la Usabilidad de Aplicaciones Web Bancarias”. Pontificia Universidad Católica del Perú.** El objetivo general del documento fue investigar acerca de las aplicaciones de la banca por Internet y la importancia del otorgar canales por Internet, que se ajusten a las necesidades de los clientes de las instituciones bancarias. La metodología propuesta está basada en el instrumento de evaluación propuesto por Nielsen, pero con una adaptación simple. Los resultados obtenidos mostraron lo importante que es no sólo contar con los canales adecuados, sino el importante papel que juega la disponibilidad y confiabilidad de los canales ofrecidos.

Lo que esta tesis aportó a la presente investigación es conocer la importancia demostrada en la construcción de aplicativos seguros y con alto grado de aporte que suma valor tanto al negocio como a la experiencia de los usuarios.

**MUÑOZ (2015).** Tesis “**Aplicación de herramientas DevOps en entornos de Desarrollo Web**”. **Universidad Politécnica de Valencia**. El objetivo de la tesis fue mejorar los procesos realizados por el equipo de desarrollo de la empresa en estudio. La metodología es de enfoque Cuantitativo, método deductivo, tipo aplicada, diseño Pre experimental. El resultado de la tesis fue mejorar la eficiencia de un departamento de desarrollo que utiliza el método tradicional, utilizando para ello herramientas open source apoyándose en la metodología DevOps. Esta tesis aporta en la investigación, brindando información que reafirma que los problemas identificados en las áreas del proceso de desarrollo de software son básicamente las mismas para todas las empresas del rubro y nos ayuda a atender el uso de determinadas herramientas desconocidas por la autora de la presente tesis.

**SANDOVAL (2015).** En su tesis: “**A Case Study in Enabling DevOps Using Docker. Master of Science in Engineering**”. **University of Texas at Austin**. El objetivo del documento es evaluar las diferentes características que posee Docker, así mismo orientar en cómo construir e implementar este tipo de contenedores. La metodología es de enfoque Cuantitativo, método deductivo, tipo aplicada, diseño Pre experimental. El resultado fue la construcción de un contenedor Docker en un Sistema de producción demostrando su adecuada tecnología para este caso específico. El aporte de la tesis a nuestra investigación radica en que nos muestra y nos clarifica la pila de redes que pueden generarse al utilizar Docker como contenedor.

## **2.2. Marco Conceptual**

### **2.2.1. Variable Independiente DevOps**

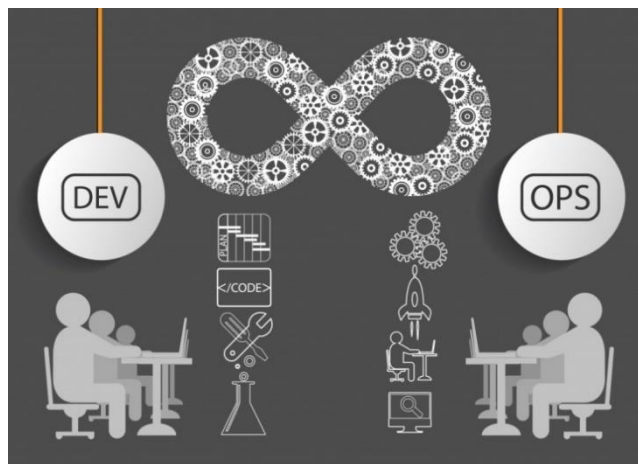
Según Ravichandran (2016), definió:

Los conceptos subyacentes abarcados por DevOps, representan una reformulación sísmica en el contexto de la producción y el soporte de software. En lugar de mantener aplicaciones de ingeniería discretas ("Dev") y gestión de TI ("Ops"), DevOps dicta el uso de equipos más pequeños con experiencia multifuncional para mejorar la funcionalidad del software y los procesos utilizados para entregarlo.



IBM (2017), definió:

DevOps es un enfoque que busca la colaboración entre líneas de negocio, desarrollo y operaciones de TI. Es un funcionamiento empresarial que faculta la entrega continua, el despliegue continuo y la supervisión continua de aplicaciones. Reduce el tiempo necesario para tratar el feedback de los clientes. El desarrollo y las operaciones, e incluso las pruebas, antes se organizaban en silos. DevOps las reúne para mejorar la agilidad.



**Figura 1.** Procesos que deben estar integrados en DevOps.

**Fuente:** <https://www.panel.es/portfolio>

### **Principios DevOps**

1. **Acción centrada en el cliente**  
Tomar medidas centralizadas en el cliente a fin de que éste invierta en productos y servicios.
2. **Responsabilidad de extremo a extremo**  
Otorga soporte del rendimiento durante todo el periodo de vida del producto.
3. **Mejora continua**  
Continuamente se acelera la mejora de productos y/o servicios a fin de minimizar los desperdicios.
4. **Automatización global**  
Pilar de DevOps no sólo en el desarrollo del software sino también con

alcance en todo el panorama de la infraestructura.

5. Trabajar como un solo equipo

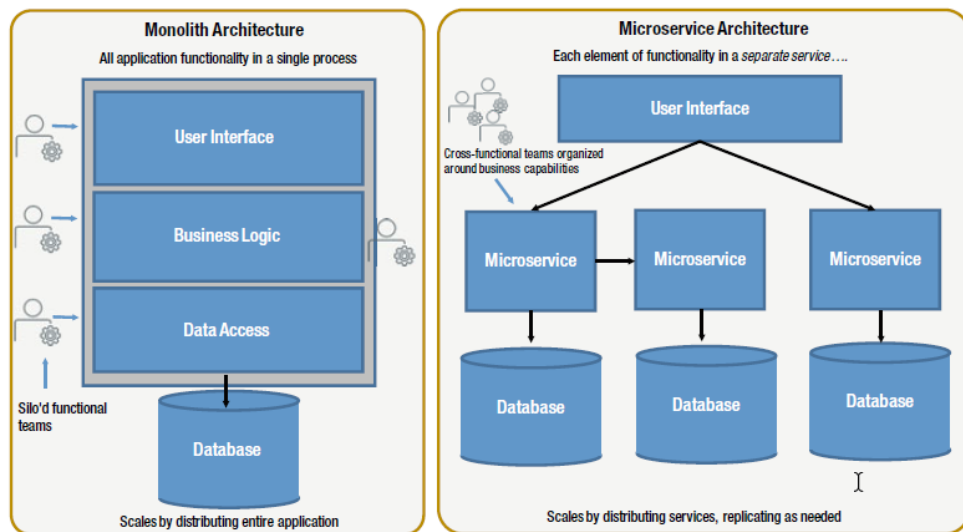
En donde cada rol está establecido, y sólo se busca es trabajar como uno solo en total colaboración.

6. Monitoreo y prueba de todo

Es importante que el equipo tenga un sólido monitoreo y procedimiento de pruebas.

Ravichandran (2016), menciona como características más relevantes, lo siguiente:

- Uso de lenguajes de programación políglota –equipos autónomos a poder seleccionar los lenguajes más adecuados para su fin.
- Infraestructura programable – permite a los equipos la construcción de códigos para administrar las configuraciones así como automatizar el aprovisionamiento de la infraestructura que apoye el control de versiones y las pruebas automatizadas con la finalidad de reducir los errores.
- Software de código abierto – independientemente del beneficio de contar con uso bajo y/o gratuito, también hay que considerar el aumento de la agilidad proporcionada. Su uso indirectamente otorga como beneficio el desarrollo colaborativo de la comunidad, con herramientas que simplifican y automatizan diversas tareas.
- Nube y plataforma como servicio – con la llegada de la virtualización de servidores y de las plataformas como servicio, se proporcionan pilas completas de tecnología en la nube para el desarrollo y las pruebas.
- Contenedores – comprenden todo un entorno en ejecución, que incluye la aplicación, además de todas sus dependencias, bibliotecas y archivos de configuración necesarios para ejecutarla, todo en un solo paquete.



**Figura 2.** Arquitecturas de aplicación monolítica y microservicio.

**Fuente:** DevOps for digital Leaders (2016)

### Ciclo de vida DevOps

#### 1. Desarrollo

En esta etapa el desarrollo se lleva a cabo constantemente dividiéndose en pequeños ciclos de desarrollo, lo que beneficia al equipo para acelerar la construcción y el proceso de entrega.

#### 2. Prueba

El equipo de QA es el encargado de identificar errores en la nueva pieza de código y que serán corregidos de inmediato, valiéndose para ello de herramientas de automatización.

#### 3. Integración

En esta etapa, la funcionalidad se integra al código vigente y se llevan a cabo las pruebas. El desarrollo continuo sólo es posible debido a la integración y pruebas continuas.

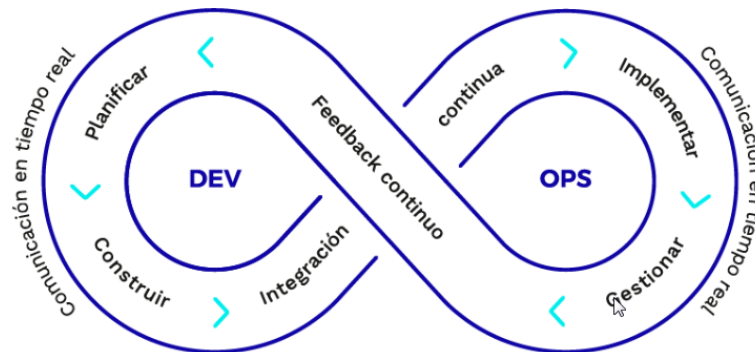
#### 4. Despliegue

Se realiza de forma que cualquier cambio realizado (en cualquier momento) no afecte el código ni funcionalidades existentes.

#### 5. Monitoreo

En esta fase el equipo de operaciones se encargará del comportamiento

inadecuado del sistema y errores en producción.



**Figura 3.** Etapas del ciclo de vida DevOps.

Fuente: tomado de <https://www.autentia.com>

Según Ravichandran (2016), en la actual era se distingue abiertamente los Modelos para la disrupción del mercado:

#### **Modelo de Madurez por Mohamed (2015):**

Basado en el modelo de madurez de CMMI para el marco de madurez de los procesos de software. Considera 5 niveles de madurez, mientras que cada nivel defiende algunos criterios en términos de cuatro dimensiones como son: Calidad, Automatización, Comunicación / Colaboración, Gobernabilidad.

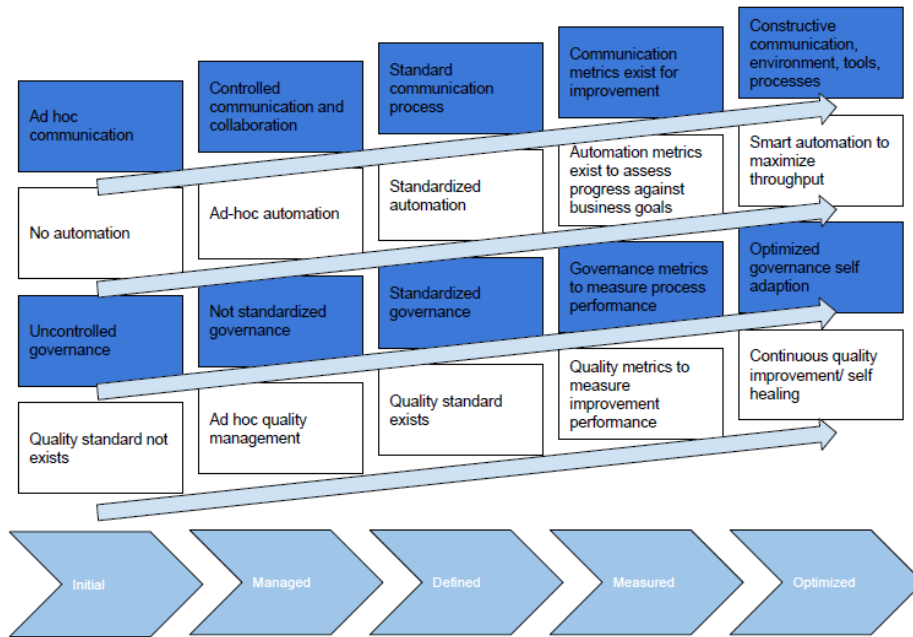
El proceso está organizado de manera que el modelo en su nivel inicial está determinado por la carencia de un proceso controlado con respecto a cada una de las dimensiones, mientras que el nivel final determina la presencia centrada en la mejora de las mismas dimensiones.

Primera dimensión: la Comunicación, en su nivel inicial, se caracteriza por no tener procesos / herramientas / roles que permitan la comunicación efectiva entre las partes interesadas. Los siguientes niveles de madurez definen cambios incrementales a fin de mejorar la comunicación.

Segunda dimensión: la Automatización alienta a las organizaciones a implementar la automatización que esté alineado con los objetivos de la

organización.

Tercera y Cuarta dimensión: hacen referencia a la especificación de procesos de gobierno claros y actividades de control de calidad apoyadas en herramientas y procesos aptos.



**Figura 4.** Modelo de Madurez DevOps.

**Fuente:** DevOps maturity model adapted from (Mohamed 2016,p.26)

### **Modelo de Madurez por Bucena y Kirikova (2017)**

Propone un modelo de madurez que abarca 5 niveles:

Nivel Inicial (Nivel 1), Nivel Repetible (Nivel 2), Nivel Definido (Nivel 3),  
Nivel Administrado (Nivel 4), Nivel Optimizado (Nivel 5).

Todos estos niveles van se encuentran definidos tomando 4 áreas empresariales como son:

- Tecnología
- Proceso
- Personas
- Cultura

Area	ID	Initial level (1)	Repeatable level (2)	Defined level (3)	Managed level (4)	Optimized level (5)
TECHNOLOGY	T1	Environments are provisioned manually	All environment configurations are externalized and versioned	Virtualization used if applicable	All environments managed effectively	Environment provisioning fully automated
	T2	Manual tests or minimal automation	Functional test automation	Triggered automated tests	Smoked tests and dashboard shared with Op.t.	Chaos Monkey
	T3	Data migration unversioned and performed manually	Changes to DB done with automated scripts versioned with application	DB changes performed automatically as part of deployment process	DB upgrades and rollbacks tested with every deployment	Feedback from DB performance after each release
	T4	Manual deployment	Build automation	Non-production deployment automation	Production deployment automation	Op.t. and Dev.t. regularly collaborate to manage risks and reduce cycle time
	T5	Manual processes for building software/ No artifact versioning	Regular automated build and testing. Any builds can be recreated from source	Automated build and test cycle every time a change is committed	Build metrics gathered, made visible and taken into account	Continuous work on process improvement, better visibility, faster feedback
	T6	No collaboration tools	Project planning tool	Team/ toolset integration	Knowledge management tool	-
	T7	No software configuration management (SCM)	Standardized SCM	Configuration is delivered together with code	Self-healing tools	-
	T8	No or minimal monitoring	Core monitoring	Integrated monitoring	Analytics/ Intelligence	-
	T9	No tools or minimal tool usage for issue tracking	All issue and bug reports are tracked	Issue reporting automatization and monitoring	Activities based on received feedback and data	Continuous delivery process
	PROCESS	PR 1	Inconsistent delivery process	Scheduled delivery process	Automated delivery process	Frequent delivery process
PR 2		Ad-hoc development	Scrum development	Agile development	Lean development	Continuous testing
PR 3		Ad-hoc testing	Requirement based testing	Integrated testing	Qualitative testing	Organized performance management
PR 4		Inconsistent project management	Project & requirement management	Integrated project management	Quantitative proj. management	-
PR 5		Deployment and development documenta-	Development documentation and relevant	Regular validation of the documentation	Documentation process and structure	-

		tion is not available or is out of date	configuration files are up-to-date	and related configuration descriptions are provided	update based on gathered experience and quality requirements	
	PR 6	Uncontrolled or reactive processes (not applied management)	Processes are managed, but not standardized	Processes are standardized across organization	Visibility & predictability of entire process & performance	Highly optimized & integrated processes
PEOPLE	P1	Teams organized around skillsets	Team organized around deliveries	Team organized around projects	Team organized around products/business lines	Interdisciplinary teams organized around KPIs
	P2	Ad-hoc learning	Team learning	Value stream learning	X-process learning	External learning
	P3	Ad-hoc approach regarding competences development	Competences are developed with the help of training and development	Analysis of exiting competences and future development	Mentor usage	Continuous capability improvement
CULTURE	C1	Restricted communication	Rapid intra-team (inside) communication	Rapid communication between teams (inter-team)	Frequent, collaborative communication	Rapid feedback
	C2	Uncommunicated vision	Clear delivery requirements	Clear project requirements	Clear product/business line requirements	Clear organization requirements
	C3	Lack of awareness of how culture is impacting day-to-day business	Awareness of aspects in culture that may help or hinder day-to-day business	Cultural traits that support business strategies have been identified	Culture viewed as an asset to be managed	Desired elements of the culture are identified, ingrained and sustainable, thus creating "the way we"
	C4	Poor, ad-hoc communication and coordination	Managed communication	Active collaboration	Collaboration based on process measurement, which allows to identify bottlenecks and inefficiencies	-
	C5	Sub-innovating/no innovations	Innovations by necessity	Innovation by design	Strategic innovation	-

**Figura 5.** Modelo de Madurez DevOps.

**Fuente:** A sample fragment from the DevOps maturity model of (Bucena and Kirikova 2017, p.7)

### **Modelo de Madurez por Feijter (2018)**

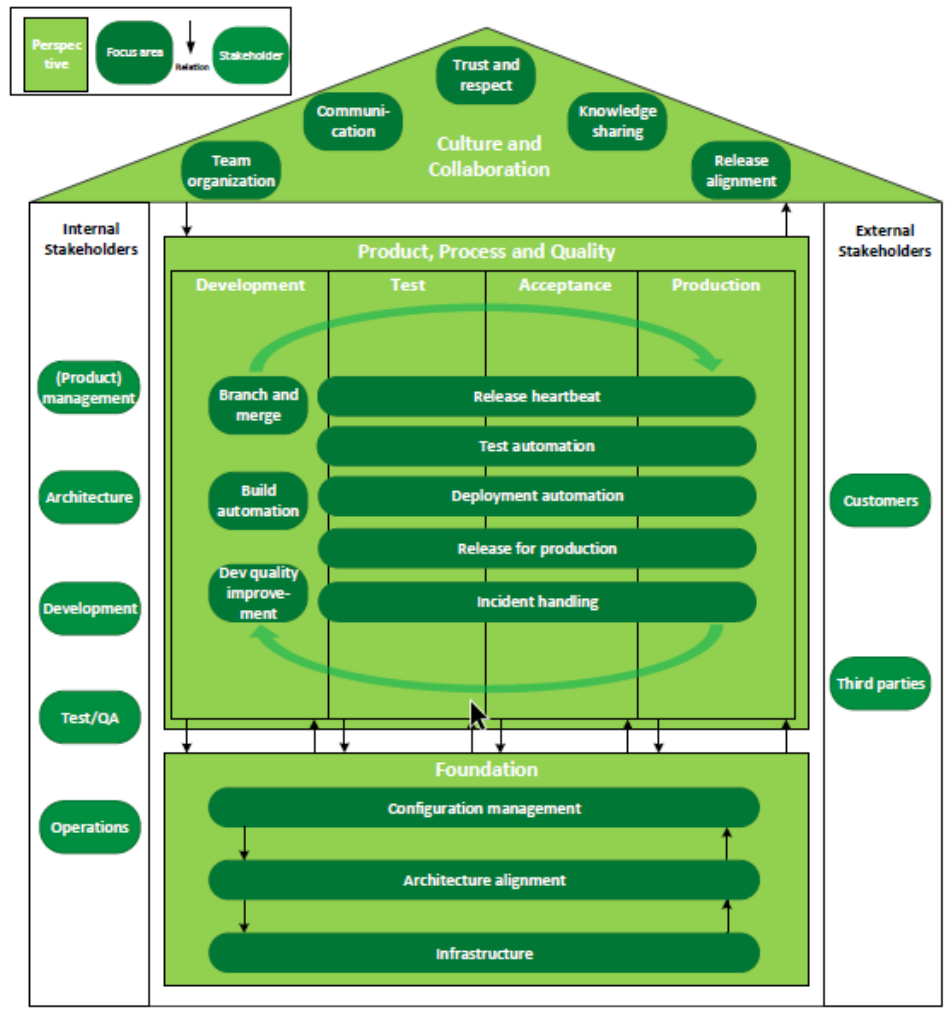
Estos modelos son específicos para las organizaciones que producen productos software estándar para ser utilizados por varios clientes, por lo que son diferentes de las organizaciones que producen software personalizado y/o para un cliente específico.

Las áreas de enfoque se agrupan en las siguientes perspectivas:

- Cultura y colaboración: esta perspectiva se coloca en la parte superior ya que resulta la más prominente, ya que la organización en sí debe estar en su lugar para realizar su trabajo. Lo ideal es que las personas interdisciplinarias compartan conocimientos, confíen y se respeten mutuamente, trabajen en equipos y exista una alineación con las dependencias internas y externas para poder implementar el software en el tiempo planificado.
- Producto, proceso y calidad: esta perspectiva intenta visualizar el proceso de liberación de un producto y los bucles de retroalimentación que residen en este proceso. Al ver el producto, el proceso y la calidad se puede discernir que varias áreas de enfoque atraviesan cuatro entornos que juntos representan el DTAP (Desarrollo, Prueba, Aceptación y Producción). Observando los entornos, se puede percibir que el software se pone continuamente en producción y al mismo tiempo los datos de producción se envían hacia la administración de productos para cumplir con los deseos de los clientes.
- Fundación: cada entorno tiene una arquitectura técnica, que está relacionada con la arquitectura del software. La infraestructura representa de forma inherente todos los entornos que son una representación de la infraestructura. Las tres áreas de enfoque de la parte inferior están asociadas entre sí; es decir la infraestructura de aprovisionamiento con los elementos de configuración correctos para preparar el entorno para la implementación requiere la gestión de la configuración.

Cada una de las áreas de enfoque se identifica con varias capacidades que se ubican en la matriz de madurez según los datos recopilados.





**Figura 6.** Modelo de Madurez DevOps.

**Fuente:** DevOps Competence Model adapted from (Feijter et al.2018, p.248)

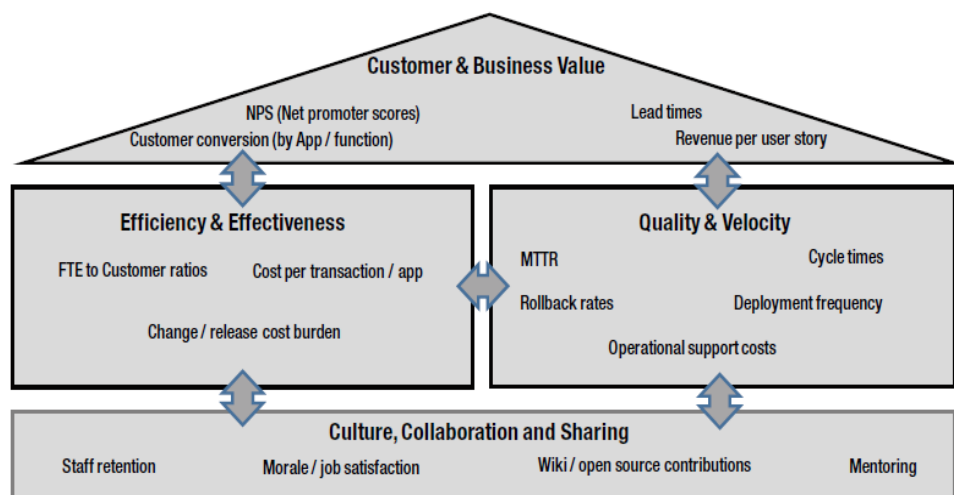
**Dimensiones de la variable independiente: Proceso de desarrollo de software basado en DevOps:**

Para Ravichandran (2016), las dimensiones aplicables a DevOps pueden ser nuevas para las organizaciones (por ejemplo, la velocidad de implementación, la tasa de cambio y la capacidad de respuesta del cliente), por lo que es importante pensar de manera general cómo los cambios en las prácticas de trabajo, el proceso y la tecnología pueden respaldar estos objetivos:

- Personas: las métricas relacionadas con el personal pueden ser las más difíciles de recopilar, pero siguen siendo indicadores de cambio potentes. Se debe prestar mucha atención a las métricas internas, como las tasas de retención de personal y la capacitación, junto con la formación de

mentores y conocimientos (por ejemplo, contribuciones de código abierto y desarrollo de wiki).

- Proceso: es importante considerar cómo las prácticas existentes ayudarán o dificultarán el logro de nuevos objetivos, prestando especial atención a los cuellos de botella existentes (por ejemplo, seguridad).
- Las auditorías realizadas solo después de las pruebas tendrán un impacto en las tasas de implementación).
- Tecnología: las buenas métricas son aquellas que ayudan a los equipos a impulsar mejoras, incluso después de fallas (por ejemplo, cuál es el porcentaje de versiones fallidas y qué porcentaje de éstas se debió a defectos de código, procesamiento manual, errores de configuración, etc.).



*i*

**Figura 7.** Métricas y dimensiones DevOps.

Fuente: DevOps for digital Leaders.

### 2.2.2 Variable Dependiente: Aplicaciones Web

Dominios de aplicación de Software:

Pressman (2010), definió que existen siete dominios de categorías, a fin de poder contextualizar nuestra variable a su dominio de aplicación definido:

- Software de sistemas, se trata de códigos de programación escritos para ser usados como servicios de otros programas, como por ejemplo: componentes de sistemas operativos, software de redes, procesadores de

telecomunicaciones.

- Software de aplicación, programas elaborados a fin de cubrir una necesidad específica de negocio, además de usarse para aplicaciones convencionales de procesamiento de datos, este software es utilizado para controlar en tiempo real determinadas funciones de negocio.
- Software de ingeniería y ciencias, se trata de programas que abordan distintas ramas de las ciencias como las matemáticas, química , física; en las que podemos hallar información y permiten la interacción a través de herramientas creadas para este fin. Podemos mencionar: Autocad (planos), P-Plan (cálculo de estructuras), Civil-Cad (diseño de caminos), ect.
- Software incrustado, es usado para implementar y controlar funciones y características para el sistema en sí y usuarios finales, ya que reside dentro de un producto o sistema proporcionando capacidad de control y funcionamiento.
- Software de línea de productos, es elaborado para ofrecer sistemas que comparten características específicas de uso para muchos consumidores diferentes, se centra en algún mercado limitado y particular (por ejemplo, control del inventario de productos) o se dirige a mercados masivos de consumidores (procesamiento de textos, hojas de cálculo, gráficas por computadora, multimedia, entretenimiento, administración de base de datos y aplicaciones para finanzas personales o de negocios)., buscando siempre la reutilización sistemática aprovechando artefactos construídos previamente.
- Aplicaciones web, focalizadas en redes que agrupan una amplia gama de aplicaciones tales como su integración con base de datos corporativas, y aplicaciones de negocios.
- Software de inteligencia artificial, centrado en el uso de algoritmos no numéricos que buscan la simulación de procesos (pp. 6-7).

Los sistemas y aplicaciones basadas en web han pasado de ser un simple conjunto de contenidos de información a sistemas refinados con una

funcionalidad complicada y contenidos en varios medios. A pesar que los aplicativos web tienen particularidades y requerimientos singulares, son software.

En los inicios de la Red Mundial (1990 – 1995), los sitios web consistían en una serie de archivos de hipertexto vinculados, que presentaban texto y gráficas de forma limitada.

Con el pasar del tiempo y con el aumento del HTML a través de mecanismos de desarrollo tales como XML y Java, proporcionó a los ingenieros brindar capacidad amplia de cómputo así como contenido de información.

“Las webapps pertenecen a una categorías de software con una combinación entre publicaciones impresas y desarrollo de software, entre la mercadotecnia y la computación, entre las comunicaciones internas y las relaciones exteriores y entre el arte y la tecnología” (Pressman, 2010, p. 9).

Como atributos comunes de las webapps, podemos mencionar:

- Intensa utilización de redes, un aplicativo web se aloja en una red y atiende exigencias diversas de clientes, permitiendo el acceso y comunicación.
- Concurrencia, capacidad de respuesta en el ingreso de los clientes a la web, situaciones o momentos en que los usuarios ingresen al mismo tiempo.
- Carga impredecible, tiene referencia con el número de usuarios que puede soportar la aplicación.
- Rendimiento, vinculado con el procesamiento del aplicativo en el acceso y tiempo de respuesta a un determinado aplicativo.
- Disponibilidad, es la disponibilidad del aplicativo para el acceso en el momento que se requiera por parte del usuario final.
- Orientación a los datos, generalmente las webapps son utilizadas para acceder a información que existe en repositorios de datos que no

pertenecen al ambiente global web (aplicaciones financieras, comercio electrónico).

- Contenido sensible, es importante la índole y esencia estética del contenido.
- Evolución continua, la evolución de este tipo de aplicaciones es continua, lo que la diferencia de cualquier otro tipo de aplicación convencional de software que se actualizan con cada petición muchas veces.
- Inmediatez, contar con herramientas modernas hace posible la producción de aplicativos web, en mucho menos tiempo.
- Seguridad, se refiere a la necesidad de implementar medidas de seguridad que permitan proteger el contenido sensible y proteger la transmisión de datos.
- Estética, es muy importante la apariencia y percepción de los usuarios ante el aplicativo, forma parte tan importante como lo es el diseño técnico.

## **MODELADO DE REQUERIMIENTO PARA WEBAPPS**

Según Pressman:

El grado de profundidad de la configuración de los requerimientos, depende de los factores:

- Tamaño y complejidad del aumento de la webapp.
- Identificar los requerimientos a fin de determinar el número de personas involucradas en los mismos.
- Grupo necesario para cubrir los requerimientos solicitados.
- Cohesión y conocimiento previo de los miembros que conformarán el equipo.
- Determinar el grado de dependencia de la organización en relación con el éxito del proyecto. (2010, p. 174).

## **MODELO DE LA INTERACCIÓN DE LA WEBAPP**

Pressman, define que:

Los aplicativos web, casi en su totalidad cubren el rol de comunicación entre la funcionalidad y el usuario final, por lo que dicho modelo consta de alguno de

los siguientes elementos:

- Casos de uso
- Diagramas de secuencia
- Diagramas de estado
- Prototipos de la interfaz de usuario. (2010, pp. 177-178)

## **MODELO FUNCIONAL PARA LOS APLICATIVOS WEB**

Para Pressman:

El modelo funcional enfrenta dos elementos de procesamiento, cada uno de los cuales representa un modelo de conceptualización diferente:

1. Funciones observables por los usuarios, es el resultado visible de una función implementada por la institución financiera y que pone a disponibilidad del cliente final. Mostrando el resultado de las operaciones que devuelven las clases de análisis pero a partir de los datos ingresados por el cliente. Podríamos tomar como ejemplo el Simulador de préstamos, simulador de proyección de ahorros, etc.
2. Operaciones contenidas en las clases de análisis, estas operaciones manipulan los atributos de clase y se mezclan como clases que cooperan con otras para lograr algún objetivo. La mayor parte de la dificultad de los aplicativos no reside en las funciones que cubren sino en la naturaleza de la información a la que se accede. (2010, pp. 178 - 179).

## **MODELO DE CONFIGURACIÓN PARA LOS APLICATIVOS WEB:**

En muchos de los sucesos, este modelo no pasa de ser una lista de características del lado del servidor y del lado del usuario. No obstante, para webapps más complicadas, son varios los obstáculos de configuración (por ejemplo, distribuir la carga en diversos servidores, arquitecturas caché, repositorios de datos remotos, diferentes servidores que atienden a varios objetos en la misma página web, etc.) que influyen en el análisis y diseño.

## **MODELO DE LA NAVEGACIÓN:**

Hay que tomar en cuenta y simular como navegará cada tipo de usuario, de modo de colocar más visibles y de forma más directa las funcionalidades principales y que son de más uso por el cliente, centrándose en los requisitos

de navegación generales.

Tomar en cuenta:

- Prioridad en la presentación de las funcionalidades.
- Necesidad del negocio de forzar al uso de determinadas funciones del aplicativo web.
- Navegación hacia grupo de elementos o hacia elementos específicos.
- El diseño de la navegación va a estar orientada a brindar disponibilidad de las funciones más comunes a las que accede el cliente.
- Manejo de vínculos externos desde los aplicativos web.

### **Dimensiones de la variable dependiente: Aplicativos Web**

Las dimensiones consideradas para la medición de resultados han sido tomadas, basándonos en el área de gestión de proyectos, detalladas en el modelo CMMI: Gestión Cuantitativa del Proyecto (Gestión de Proyectos en el nivel de madurez 4) en las que prima las dimensiones de Calidad y Rendimiento del proceso.

“El área de proceso Gestión Cuantitativa del Proyecto establece objetivos para la calidad y el rendimiento del proceso, redacta un proceso definido que puede ayudar a lograr esos objetivos y gestiona cuantitativamente el proyecto. Los objetivos de calidad y de rendimiento del proceso del proyecto están basados en los objetivos establecidos por la organización y el cliente” (CMMI® para Desarrollo, Versión 1.3,2010, p.68).

### **Dimensión Calidad del Proceso**

“La calidad de un sistema o producto está muy influenciada por la calidad del proceso empleado para desarrollarlo y mantenerlo” (CMMI® para Desarrollo, Versión 1.3,2010, p.29)

Según CMMI (2010) explica:

Es el aseguramiento de la entrega de productos y servicios de alta calidad, ya que proporciona la visibilidad apropiada a todos los niveles de gestión, además de ofrecer una realimentación sobre los procesos y productos de trabajos

asociados, durante toda la vida del proyecto. (CMMI® para Desarrollo, Versión 1.3, 2010, p. 99).

El aseguramiento de la calidad del proceso busca su implantación en cada uno de los proyectos de las organizaciones, así mismo contar con la suficiente independencia para otorgar objetividad al identificar y comunicar las no conformidades que puedan existir dentro de los proyectos.

Indicadores:

- a. Tiempo medio entre fallos.
- b. Número y gravedad de los defectos en el producto liberado.
- c. Uso de recursos críticos.
- d. Número y gravedad de las reclamaciones del cliente respecto al servicio proporcionado.

### **Dimensión Rendimiento del proceso**

Según CMMI (2010) explica:

Refiere a las pruebas de rendimiento para determinar la respuesta del sistema bajo condiciones particulares de trabajo, buscando monitorear y medir la capacidad del sistema de utilizar los recursos de forma eficiente y evaluar la calidad ofrecida por los componentes

Indicadores:

- a. Porcentaje de tiempo de retrabajo.
- b. Porcentaje de defectos hallados en las actividades de verificación del producto.
- c. Tasa de defectos no detectados.
- d. Número y gravedad de defectos encontrados.
- e. Porcentaje de tiempo no productivo.
- f. Tiempo medio de resolución de problemas en el entorno productivo.

### **CMMI (Capability Maturity Model Integration):**

Los modelos CMMI, proporcionan un conjunto de guías para la obtención de



mejores productos y servicios, lo que se traduce en recopilación de buenas prácticas que facilitan a las organizaciones, mejoras de sus procesos a fin de abarcar las necesidades de los clientes y usuarios finales.

Existen cinco áreas en el proceso de Ingeniería de CMMI-DEV, que son:

- Integración del producto (PI).

Es el área de proceso, donde se conjugan los factores del producto y se comprueban las interfaces para aseverar que satisfacen los requisitos de interfaz proporcionados por el área de proceso Desarrollo de Requisitos.

- Desarrollo de Requisitos (RD)

El área de Desarrollo de Requisitos (RD), es la encargada de reconocer las exigencias del cliente y transformarlas en requerimientos para el producto. Este conjunto de requerimientos es analizado a fin de encontrar soluciones conceptuales de alto nivel, para luego instaurar un conglomerado inicial de elementos del producto, en términos que el desarrollador pueda interpretar y utilizar. Otorga los requisitos al área Solución Técnica y al área de proceso de Integración del producto.

- Solución Técnica (TS).

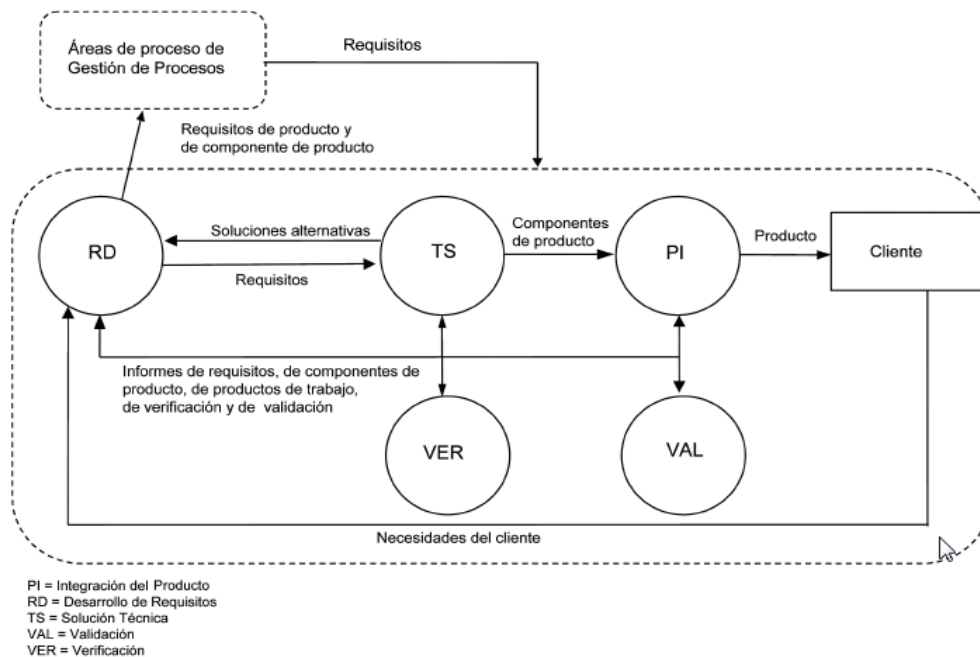
Es donde los requisitos proporcionados por el área de Desarrollo de Requisitos, pasan a formar la arquitectura del producto, se desarrollan los paquetes de datos técnicos examinando soluciones alternativas para elegir el diseño adecuado basado en principios establecidos.

- Validación (VAL).

El área de Validación autentifica de forma incremental los productos frente a los requerimientos del cliente. Abarca la aprobación de productos, componentes, productos de trabajo intermedio seleccionados y de procesos.

- Verificación (VER).

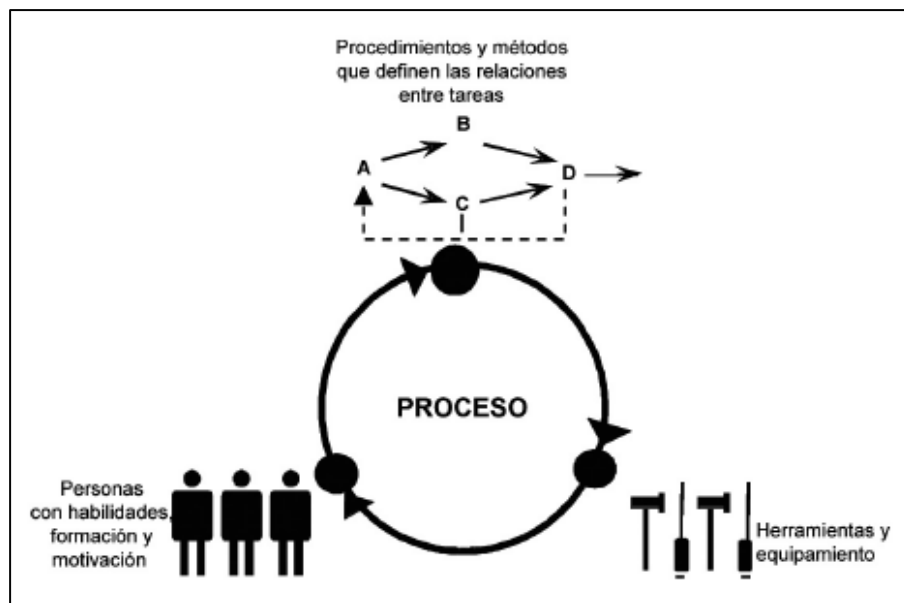
El área de Verificación se encarga de escoger los productos de trabajo y métodos de validación que serán utilizados para comprobar los productos de trabajo frente a los requerimientos especificados. Es un proceso incremental que empieza con la verificación de elementos de un producto y finaliza con la revisión de los productos ensamblados por completo.



**Figura 8.** Áreas del proceso de Ingeniería de Software.  
**Fuente:** CMMI® para Desarrollo, Versión 1.3

### Acerca de la mejora de los procesos

Software Engineering Institute (SEI), identifica tres puntos claves a partir de los cuales las organizaciones e instituciones pueden centrarse a fin de mejorar su actividad.



**Figura 9.** Puntos claves de las organizaciones  
**Fuente:** CMMI® para Desarrollo, Versión 1.3

### 2.3. Definición de términos

**Software**, es una agrupación de programas de cómputo, prácticas, normas, registros y datos relacionados que conforman las operaciones de un Sistema computacional.

**Fases**, Etapa o estado de un proceso, fenómeno natural o histórico, doctrina, negocio, etc.

**Iterativo**, equivale a realizar varias veces un proceso con la finalidad de conseguir un resultado esperado u objetivo. Cada acción de repetir el proceso es denominado una "iteración", y los resultados de una iteración se utilizan como punto de partida para la siguiente iteración.

**Entrega continua**, es una perspectiva de la ingeniería del software en que los grupos de desarrollo generan software en ciclos reducidos, garantizando que el software puede ser desplegado en cualquier momento, de forma confiable. Enfoca la construcción, revisión, y despliegue del software de modo más rápido y más constante. Esta óptica ayuda en disminuir los costos, tiempo, y posibles problemas de los despliegues de versiones por medio de la liberación de versiones más incrementales a aplicaciones en producción. Un proceso directo y repetible de liberación es determinante para una entrega continua.

**Integración Continua**, La integración continua es una práctica de desarrollo de software donde los participantes de un equipo integran su trabajo frecuentemente, al menos una vez diariamente, lo que conlleva a múltiples integraciones por día. Cada integración se comprueba mediante una compilación automatizada para identificar errores de integración en el menor tiempo posible. Muchos equipos concluyen que esta práctica reduce significativamente la integración y permite que el equipo desarrolle un software cohesivo más efectivamente.

**Automatización de procesos**, la automatización de procesos, se refiere al proceso de reducción, mejora y hacer automáticos los procesos clave que impulsan a una

organización, teniendo como fin principal la reducción de los costos mediante la integración de aplicaciones, minimizando la mano de obra, acelerando el tiempo de ejecución de las actividades y reemplazando los procesos manuales con aplicaciones de software.

**Feedback**, viene a ser un procedimiento de control de sistemas; en el cual los resultados que se obtienen son reingresados al sistema con la finalidad de comprobar y mejorar su funcionamiento.

**Arquitectura de software**, también denominada arquitectura lógica, se basa en un conjunto de modelos e ideas abstractas relacionadas que proporcionan un marco definido y claro para interactuar con el código fuente del software.

**Compatibilidad de componentes**, es la circunstancia que hace que un programa y un sistema, arquitectura o aplicación consigan funcionar correctamente, tanto directa o indirectamente (mediante un algoritmo). A este conjunto ordenado y finito de operaciones, que hace que un programa logre ser comprendido por un sistema, arquitectura o aplicación es llamado emulador, el cual es un traductor entre el programa y el sistema, arquitectura o aplicación.

**Fintech**, Financial Technology (Tecnología Financiera) tiene como base la aplicación de la tecnología a los servicios financieros, identificando y estudiando los problemas a fin de otorgar soluciones.

**Entornos de desarrollo**, es la unión de herramientas que automatiza o soporta al menos una gran parte de la tareas del desarrollo: análisis de requisitos, diseño de arquitectura, diseño detallado, codificación, pruebas de unidades, pruebas de integración y validación, gestión de configuración, mantenimiento, etc. Las herramientas deben estar bien acopladas, permitiendo inter operar entre ellas.

Están formados por una colección de instrumentos (hardware, software, procedimientos,etc.) que ayudan a facilitar o hacer automáticas las actividades de desarrollo.

## **Aplicaciones sin Servidor**

Tradicionalmente, en las aplicaciones web creadas e implementadas, se tiene determinado grado de verificación sobre las solicitudes HTTP que se dirigen al servidor. Nuestra aplicación se ejecuta en ese servidor y somos responsables de abastecer y gestionar los recursos, es aquí donde surgen algunos inconvenientes:

1. Se tiene que pagar por mantener activo el servidor, inclusive cuando no se atiende solicitud alguna.
2. Nos hacemos cargo del tiempo de actividad y conservación del servidor y todos sus recursos.
3. Nos convertimos en los encargados de aplicar las actualizaciones de seguridad adecuadas al servidor.
4. Conforme nuestra escala de uso fluctúa también necesitamos gestionar la ampliación de escala del servidor. Y como resultado, configurar la escala hacia abajo cuando no se tenga tanto uso.

Para instituciones más pequeñas y desarrolladores individuales, esto puede ser muy complicado de manejar y termina distrayendo al equipo de desarrollo de las tareas de mayor relevancia que es el desarrollar y brindar mantenimiento a la aplicación actual. En instituciones de mayor envergadura, esto es dirigido por el equipo de infraestructura y, generalmente, no es obligación del desarrollador individual. No obstante, los procesos necesarios para apoyar esto pueden terminar haciendo más lentos los tiempos de desarrollo. Por este motivo, hemos estado buscando solución a estos problemas y aquí es donde entra en juego el servidor.

## **Computación sin servidor**

Se refiere a un procedimiento de ejecución en el que el proveedor de la nube (Google Cloud, AWS o Azure) se hace cargo de ejecutar un fragmento de código mediante la asignación dinámica de los recursos. Y sólo se realiza el pago por la cantidad de recursos utilizados para realizar la ejecución de dicho código.

El código por lo general se ejecuta en contenedores sin estado que pueden ponerse en marcha por una variedad de eventos que incluyen alertas de monitoreo, solicitudes http, servicios de colas , eventos de base de datos, carga de archivos, eventos

programados (trabajos cron), etc. El código enviado al proveedor de la nube para la ejecución, es por lo general en la forma de una función. Por lo tanto, serverless a veces se denomina "Funciones como un servicio" o "FaaS". A continuación se presentan las ofertas de FaaS de los principales proveedores de la nube:

- AWS: AWS Lambda
- Microsoft Azure
- Google Cloud

Dado que su código se ejecutará como funciones individuales, hay algunas consideraciones que debemos tener en cuenta.

### **Microservicios**

El mayor cambio que se afronta durante la transición a un ambiente sin servidores es que las aplicaciones requieren ser diseñada en forma de funciones, en los entornos sin servidor, generalmente se necesita la adopción de una arquitectura apoyada en microservicios. Puede resolver esto corriendo toda la aplicación dentro de una sola función como un monolito y manejando el enrutamiento usted mismo. Pero esto no se recomienda ya que es mejor reducir el tamaño de sus funciones.

### **Funciones sin estado**

Sus funciones por lo general se ejecutan dentro de contenedores seguros sin estado. Esto significa que no podrá ejecutar el código en su servidor de aplicaciones que se ejecuta mucho después de que un evento se haya completado o use un contexto de ejecución anterior para atender una solicitud. Debe aceptarse y tomarse conciencia que su función se invoca de nuevo cada vez.

### **Arranques fríos**

Dado que sus procedimientos se ejecutan dentro de un contenedor que se activa bajo petición para responder a un evento, existe cierta demora asociada con él. Esto se conoce como un arranque en frío. Es posible que su contenedor se mantenga por un tiempo después de que su función haya finalizado su ejecución. Si se realiza la

petición de otro evento en este periodo, este responde mucho más rápido y esto generalmente se conoce como un arranque en caliente.

El tiempo de duración de los arranques en frío depende de la configuración del proveedor de nube específico. En AWS Lambda puede variar desde unos pocos cientos de milisegundos hasta unos pocos segundos. Puede depender del tiempo de ejecución utilizado, el tamaño de la función (como un paquete) y, por supuesto, el proveedor de la nube en cuestión. Los arranques en frío han mejorado drásticamente a lo largo de los años, ya que los proveedores de servicios en la nube han mejorado mucho en la optimización para tiempos de respuesta más bajos.

Además de mejorar sus funciones, puede valerse de trucos simples como una función automatizada separada para llamar su función cada pocos minutos para mantenerla caliente. El Framework sin servidor que vamos a usar tiene algunos complementos para ayudar a mantener sus funciones calientes.

## **2.4. Hipótesis**

### **2.4.1. Hipótesis General**

El proceso de desarrollo de software basado en DevOps mejora la obtención de las aplicaciones web en una institución Financiera.

### **2.4.2. Hipótesis Específicas**

- a. El proceso de desarrollo de software basado en DevOps optimiza el rendimiento del proceso en las aplicaciones web de una institución Financiera.
- b. El proceso de desarrollo de software basado en DevOps optimiza la calidad del proceso de las aplicaciones web de una institución Financiera.

## 2.5. Variables

### 2.5.1. Definición conceptual de la variable

**Variable Independiente:** Proceso de desarrollo basado en DevOps

Según Ravichandran (2016), define:

“En el contexto actual, en el que la mayoría de las organizaciones se mueven para adoptar una entrega más ágil y eficiente, y prácticas de gestión de TI que satisfagan las expectativas cambiantes de los clientes y que ofrece las estrategias de crecimiento más rápido y ampliamente adoptadas y que llevan la eficiencia al ciclo de vida de las aplicaciones, se encuentra la metodología DevOps”.

**Variable Dependiente:** Aplicativos Web

Pressman (2010), define:

“Aplicaciones web: conocidos también como webapps, esta clasificación de software enfocado en redes reúne una amplia escala de aplicaciones. En su forma más simple, son poco más que una agrupación de archivos de hipertexto relacionados que presentan información con uso de texto y gráficas limitadas. Sin embargo, desde que surgió Web 2.0, las webapps están avanzando hacia ambientes computacionales complejos que no sólo ofrecen características aisladas, funciones de cómputo y contenido para el usuario final, sino que también están integradas con bases de datos corporativas y aplicaciones de negocios”.

### 2.5.2. Definición operacional de la variable

**Variable dependiente**

Son aplicaciones de fácil acceso, ya que sólo es necesario contar con conexión a Internet o intranet, además son independientes de los sistemas operativos y sobre todo son de mantenimiento fácil.

### 2.5.3. Operacionalización de la variable

**Variable Dependiente**

Es un proceso de causa-efecto que consiste en descomponer, utilizando el método deductivo, las variables del problema a investigar, empezando desde



lo más general hasta lo más específico a través de las dimensiones e indicadores.

**Tabla 01.**

*Operacionalización de la variable Dependiente Aplicaciones web*

VARIABLE	DIMENSIÓN	INDICADOR	ESCALA	NIVEL Y RANGO
Aplicaciones Web	Rendimiento del proceso	c. Porcentaje tiempo de retrabajo	1. Muy bajo	Nivel Bajo
		d. Porcentaje defectos hallados en la verificación del producto	2. Bajo	< 27-35 >
		e. Tasa de defectos no detectados	3. Medio	Nivel Medio
		f. Número y gravedad de defectos encontrados	4. Alto	<17-26 >
		g. Porcentaje de tiempo no productivo	5. Muy alto	Nivel Alto
		h. Tiempo medio de resolución de problemas en ambiente productivo		< 7-16 >
		i. Frecuencia de despliegue		
	Calidad del Proceso	j. Tiempo medio entre fallos.	1. Muy bajo	Nivel Bajo
		k. Número y gravedad de defectos en el producto liberado.	2. Bajo	< 27-35 >
		l. Uso de recursos críticos	3. Medio	Nivel Medio
		m. Numero y gravedad de reclamaciones del cliente por el servicio.	4. Alto	<17-26 >
		n. Porcentaje de éxito en test automático	Muy alto	Nivel Alto
		o. Defectos hallados en producción		< 7-16 >
		p. Seguridad de los aplicativos		

Fuente: Elaboración propia.

## CAPÍTULO III

### METODOLOGÍA

#### 3.1. Método de Investigación

El método general de investigación de investigación fue el científico: deductivo, al ser una estrategia de razonamiento utilizada para obtener conclusiones lógicas a partir de una serie de premisas o principios.

En este sentido, es un proceso de pensamiento que va de lo general (leyes o principios) a lo particular (fenómenos o hechos concretos).

Según el método deductivo, la resolución se halla dentro de las propias premisas referidas, dicho de otro modo, la conclusión es consecuencia de estas; es un proceso de causa-efecto que consiste en descomponer deductivamente las variables del problema a investigar comenzando desde lo más general hasta lo más específico mediante las dimensiones e indicadores.

#### 3.2. Tipo de investigación

La investigación es de tipo aplicada con enfoque Cuantitativo. Aplicada, porque el problema está identificado, por lo que utiliza la investigación para dar respuesta a preguntas específicas. Según Hernández, Fernández y Baptista (2014), define que “la investigación aplicada considera planteamientos para evaluar, comparar, interpretar, establecer precedentes y determinar causalidad y sus aplicaciones”.

En este tipo de investigación el énfasis del estudio está en la conclusión práctica de problemas. Se enfoca específicamente en cómo se pueden aplicar las teorías generales. Su motivación va hacia la solución de los problemas que se plantean en un determinado momento.

#### 3.3. Nivel de investigación

El nivel de investigación es Explicativa. Según Hernandez, Fernandez y Baptista (2014), define que “las investigaciones de nivel explicativa pretenden establecer las causas de los sucesos o fenómenos que se estudian”.

### **3.4. Diseño de investigación**

El diseño de la investigación es Pre experimental de corte longitudinal. Según Hernandez, Fernández y Baptista (2014), define “el diseño pre experimental como un tipo de diseño de investigación que administra estímulos o tratamientos y/o intervenciones que tienen un grado de control mínimo”.

### **3.5. Población y muestra**

#### **3.5.1. Población**

Chávez, define:

“La población se define como universo de la investigación, sobre el cual se pretende generalizar los resultados” (2017).

En esta investigación, la población está conformada los 10 integrantes del equipo de desarrollo de aplicativos web de la institución.

#### **3.5.2. Muestra**

La muestra se considera censal al haberse seleccionado el 100% de la población al considerarla un número manejable de sujetos. En este sentido Ramirez (2017), establece que la muestra censal es aquella donde todas la unidades de investigación fueron tomadas como muestra.

### **3.6. Técnicas e instrumentos de recolección de datos**

Hurtado (2016), define:

“La técnica tiene que ver con los procedimientos utilizados, para la recolección de los datos, es decir el cómo. Éstos pueden ser de observación documental, observación, encuesta y técnicas sociométricas entre otras”

Para Tamayo y Tamayo (2007), define:

“Una técnica valiosa y significativa, es la encuesta, que recoge información directa o indirecta formulando preguntas, las cuales son formadas y llenadas por un empadronador, frente a quien le responde”.

Para Hernández Fernández (2006), define:

“El cuestionario es un recurso que se mide a través de un instrumento, estos deben estar basados en referencias teóricas por los indicadores suficientes para medirlos, donde se miden según actitudes, registros del contenido (análisis de

contenido) y observación”.

Para la recolección de la información se opta por el instrumento de elaboración de cuestionario, el mismo que fue adaptado a 14 preguntas de tipo cerradas. Se utilizó la valoración en la escala de Likert de 5 puntos, se realizó una serie de enunciados sobre el tema en cuestión, en los cuales los individuos mostrarán su grado de acuerdo o desacuerdo para todas las preguntas formuladas.

La variable respuesta Rendimiento del Proceso (RP), se evaluó utilizando 7 preguntas.

La variable Calidad del aplicativo (CA) fue evaluó utilizando también el mismo número de preguntas.

### **3.7. Procesamiento de la información**

Asesoría de investigación, se llevó a cabo previa coordinación con los asesores a fin de que se den las pautas necesarias para trazar los objetivos y definir la metodología a utilizar.

Recopilación de información bibliográfica, revisión de investigaciones relacionadas al tema, tanto en el rubro de instituciones financieras así como en otro tipo de negocios a fin de conocer el manejo y análisis de las diferentes problemáticas existentes.

Instrumento experimental, se aplicó un cuestionario ajustado para evaluar las variables establecidas.

### **3.8. Técnicas y análisis de datos**

#### **Técnicas**

Cuestionario previamente validado en el juicio de expertos.

#### **Análisis de datos**

Contrastación de hipótesis a partir de tabla de datos y que buscan rechazar la hipótesis nula. Se efectuaron comparaciones de las muestras relacionadas. Al ser una prueba no paramétrica se empleó la prueba de los rangos con signo de Wilcoxon, cuyo detalle se presenta en el siguiente capítulo donde se encuentran los Resultados.

### **Ficha técnica del instrumento**

Nombre	: Cuestionario para medir los Aplicativos Web
Autor	: Elaborado por Magally Antón
Año	: 2019
Objetivo	: Determinar el efecto del proceso de desarrollo de software basado en DevOps para la obtención de las aplicaciones web.
Lugar de aplicación	: Institución Financiera de Lima Metropolitana
Forma de aplicación	: Directa a trabajadores
Duración de la Aplicación:	15 min.

Descripción del instrumento: Cuestionario destinado a trabajadores que conforman el equipo de desarrollo de aplicaciones web de la institución financiera. Para la variable Rendimiento del proceso, el cuestionario estuvo constituido por 7 preguntas y otras 7 preguntas para la dimensión Calidad del proceso; las respuestas del cuestionario estuvieron estructuradas bajo la escala Likert, considerando cinco categorías: Muy bajo = 1 punto, Bajo = 2 puntos, Medio = 3 puntos, Alto = 4 puntos, y Muy alto = 5 puntos

### **Procedimiento de puntuación, Escala de Baremización:**

Para el instrumento completo de 14 preguntas:

Nivel Bajo < 52 – 70 >

Nivel Medio < 33 – 51 >

Nivel Alto < 14 – 32 >

Para las dimensiones:

Nivel Bajo < 27 – 35 >

Nivel Medio < 17 – 26 >

Nivel Alto < 7 – 16 >

### 3.8.1 Validación y confiabilidad de los instrumentos

#### Validación del instrumento

La validación de un instrumento, en términos generales, se refiere al grado en que un instrumento realmente mide la variable que pretende medir (Hernández, Fernández y Baptista, 2010).

Para determinar la validez del instrumento, se sometió a consideraciones de juicio de expertos. Según Hernández, Fernández y Baptista (2010), el juicio de expertos consiste en preguntar a personas expertas acerca de la pertinencia, relevancia, claridad y suficiencia de cada uno de los ítems, en el caso del instrumento.

**Tabla 2.**

Validez del instrumento para los Aplicativos Web del proceso según expertos.

Experto	El instrumento presenta				Condición final
	Pertinencia	Relevancia	Claridad	Suficiencia	
Juez 1	si	si	si	si	Aplicable
Juez 2	si	si	si	si	Aplicable
Juez 3	si	si	si	si	Aplicable

Fuente: Elaboración propia.

La tabla muestra que los expertos consideraron el instrumento como aplicables por contener ítems pertinentes, relevantes, claros y suficientes para garantizar la medición válida de la variable Aplicativos Web.

### 3.8.2 Confiabilidad del instrumento

El instrumento de recolección de datos que presentaron ítems con opciones politómicas, fueron evaluados a través del coeficiente alfa de Cronbach con el fin de determinar su consistencia interna, analizando la correlación media de cada ítem con todas las demás que integran dicho instrumento. Se aplicó la

prueba piloto y después de analizó mediante el Alfa de Cronbach con la ayuda del software estadístico SPSS versión 25.

**Tabla 3.**  
*Escala de valores para determinar la confiabilidad (Herrera 1998)*

Valor	Confiabilidad
0,53 a menos	Confiabilidad nula
0,54 a 0,59	Confiabilidad baja
0,60 a 0,65	Confiable
0,66 a 0,71	Muy confiable
0,72 a 0,99	Excelente confiabilidad
1.00	Confiabilidad perfecta

Fuente: Elaboración propia.

Resumen de procesamiento de casos			
		N	%
Casos	Válido	10	100,0
	Excluido <sup>a</sup>	0	,0
	Total	10	100,0

a. La eliminación por lista se basa en todas las variables del procedimiento.

Estadísticas de fiabilidad	
Alfa de Cronbach	N de elementos
,839	14

**Figura 10.** Confiabilidad del instrumento – Alfa de Cronbach - Pre Test

**Fuente:** Elaboración propia con SPSS v.25

Para evaluar la confiabilidad de la prueba, se empleó el método de consistencia interna, a través del coeficiente Alfa de Cronbach, se halló un

valor de 0.839 para el instrumento indicando que la escala presenta un nivel de excelente confiabilidad para el PRE TEST.

<b>Resumen de procesamiento de casos</b>			
		N	%
Casos	Válido	10	100,0
	Excluido <sup>a</sup>	0	,0
	Total	10	100,0

a. La eliminación por lista se basa en todas las variables del procedimiento.

<b>Estadísticas de fiabilidad</b>		
Alfa de Cronbach	N de elementos	
,809	14	

**Figura 11.** Confiabilidad del instrumento – Alfa de Cronbach - Post Test

**Fuente:** Elaboración propia con SPSS v.25

Para evaluar la confiabilidad de la prueba, se empleó el método de consistencia interna, a través del coeficiente Alfa de Cronbach, se halló un valor de 0.809 para el instrumento indicando que la escala presenta un nivel de excelente confiabilidad para el POST TEST.

### 3.8.3 Métodos de análisis de datos

El procedimiento para la recolección de datos siguió los siguientes pasos:

Se inició con la aplicación del instrumento, siguiendo las indicaciones establecida en las respectivas fichas técnicas. Se solicitó el permiso respectivo para la aplicación del cuestionario. El tiempo aproximado que tomó el instrumento para ser respondido fue de 15 minutos.

Posteriormente, con los datos obtenidos se elaboró la matriz de datos, se transformaron los valores según las escalas establecidas y se procedió con el debido análisis con la finalidad de presentar las conclusiones y recomendaciones y de esta manera preparar el informe final.



Para el análisis y presentación de los datos obtenidos en la investigación, se empleó la estadística descriptiva e inferencial. Estos resultados fueron representados utilizando figuras estadísticas para poder visualizar y comprender mejor la investigación.

En un primer momento se procedió a organizar y ordenar la información recopilada en una base de datos, posteriormente se analizaron empleando el software SPSS versión 25, que permitió establecer el porcentaje de incidencias en las respuestas obtenidas, Para la contratación de las hipótesis se empleó la prueba no paramétrica Wilcoxon, pues la variable de estudio era cuantitativa y medidas en una escala ordinal.

Cabe mencionar que en la presente investigación que una vez definida la variable como una variable cualitativa, el estadístico no paramétrico que se utilizó permitió contrastar, aceptar o rechazar las hipótesis.

#### **3.8.4 Aspectos Éticos**

Los datos indicados en esta investigación fueron recogidos del grupo de investigación y se procesaron de forma adecuada sin adulteraciones.

Los integrantes del equipo que participaron en este cuestionario, no fueron mencionados, se han tomado las reservas del caso para evitar información dañina en contra de las personas de la institución que han colaborado con esta investigación.

De igual forma el marco teórico se recolectó de acuerdo a los parámetros establecidos e indicados para realizar este tipo de estudio, evitando copia de otras investigaciones.

Finalmente los resultados de la investigación no han sido adulteradas o plagiadas de otras investigaciones haciéndose un buen uso de la investigación en beneficio de todos.

## CAPÍTULO IV

### RESULTADOS

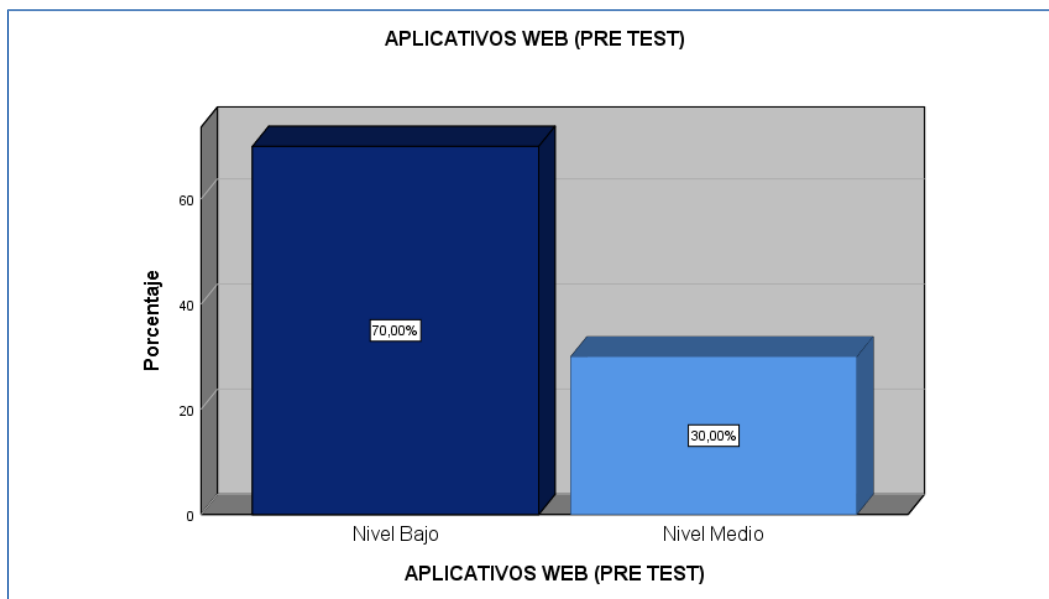
#### 4.1. Estadística Descriptiva

##### 4.1.1. Resultados del Pre-Test: Variable Aplicativos Web (Pre Test)

**Tabla 4.**  
Aplicativos Web (Pre Test)

		Frecuencia	Porcentaje		Porcentaje acumulado
Válido	Nivel Bajo	7	70,0	70,0	70,0
	Nivel Medio	3	30,0	30,0	100,0
	Total	10	100,0	100,0	

Fuente: elaboración propia con SPSS v.25



**Figura 12.** Aplicativos Web (Pretest)

Fuente: Elaboración propia con SPSS v.25

#### Interpretación

De acuerdo a la tabla 4 y figura 12, muestran los resultados desde la percepción de los encuestados, que el 70.0% tienen un nivel bajo, el 30% tiene un nivel medio, en las aplicaciones Web antes de realizar proceso de Desarrollo de Software basado en DevOps, de una institución Financiera.

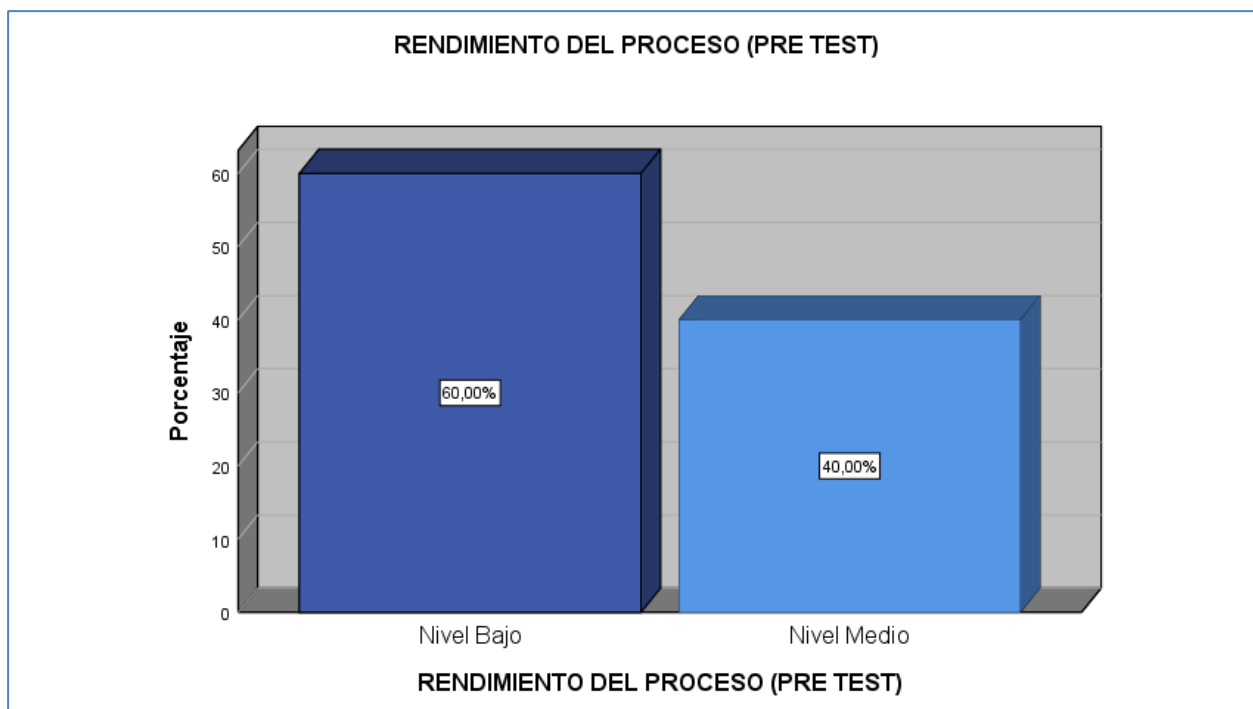
#### 4.1.2. Dimensión: Rendimiento del proceso (Pre Test)

**Tabla 5.**

Rendimiento del proceso (Pre Test)

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válido	Nivel Bajo	6	60,0	60,0	60,0
	Nivel Medio	4	40,0	40,0	100,0
	Total	10	100,0	100,0	

Fuente: elaboración propia con SPSS v.25



**Figura 13.** Rendimiento del proceso (Pretest)

Fuente: Elaboración propia con SPSS v.25

#### Interpretación

De acuerdo a la tabla 5 y figura 13, muestran los resultados desde la percepción de los encuestados, que el 60.0% tienen un nivel bajo, el 40% tiene un nivel medio, en las aplicaciones Web en la dimensión rendimiento del proceso, antes de realizar proceso de Desarrollo de Software basado en DevOps, de una institución Financiera.

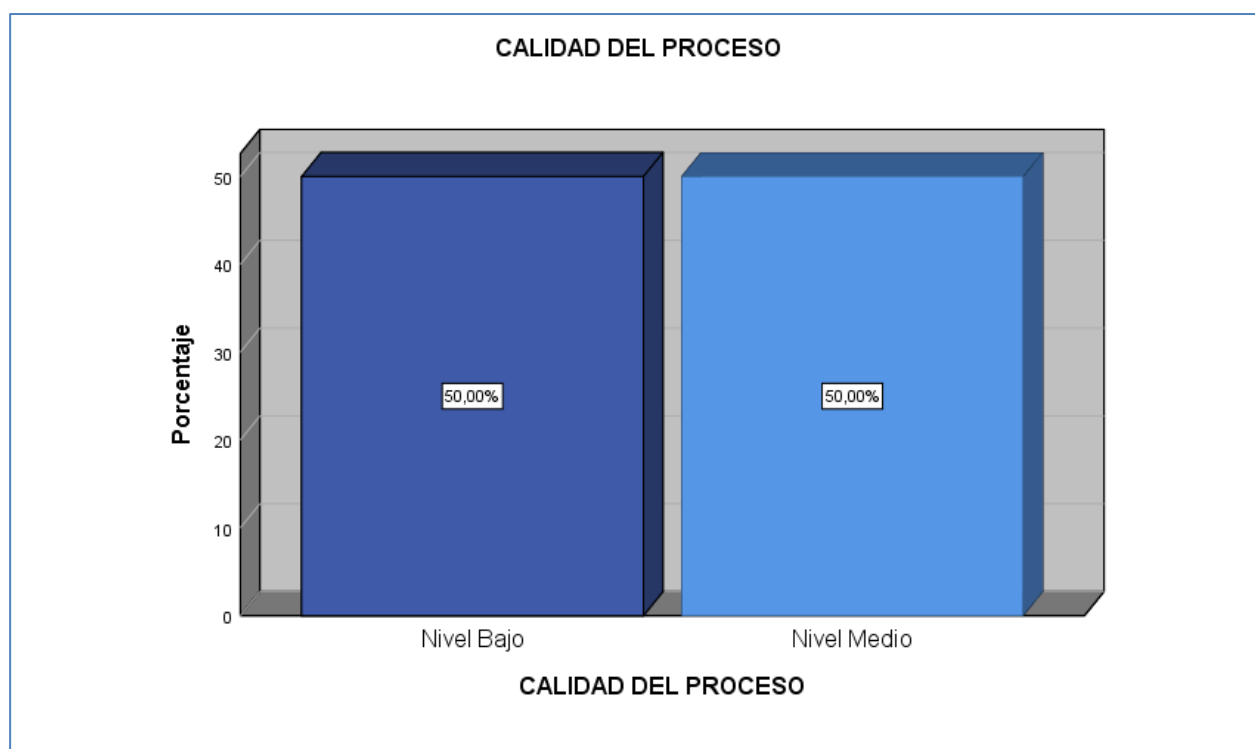
### 4.1.3. Dimensión: Calidad del proceso (Pre Test)

**Tabla 6.**

Calidad del proceso (Pre Test)

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válido	Nivel Bajo	5	50,0	50,0	50,0
	Nivel Medio	5	50,0	50,0	100,0
	Total	10	100,0	100,0	

Fuente: elaboración propia con SPSS v.25



**Figura 14.** Calidad del Proceso (Pretest)

Fuente: Elaboración propia con SPSS v.25

#### Interpretación

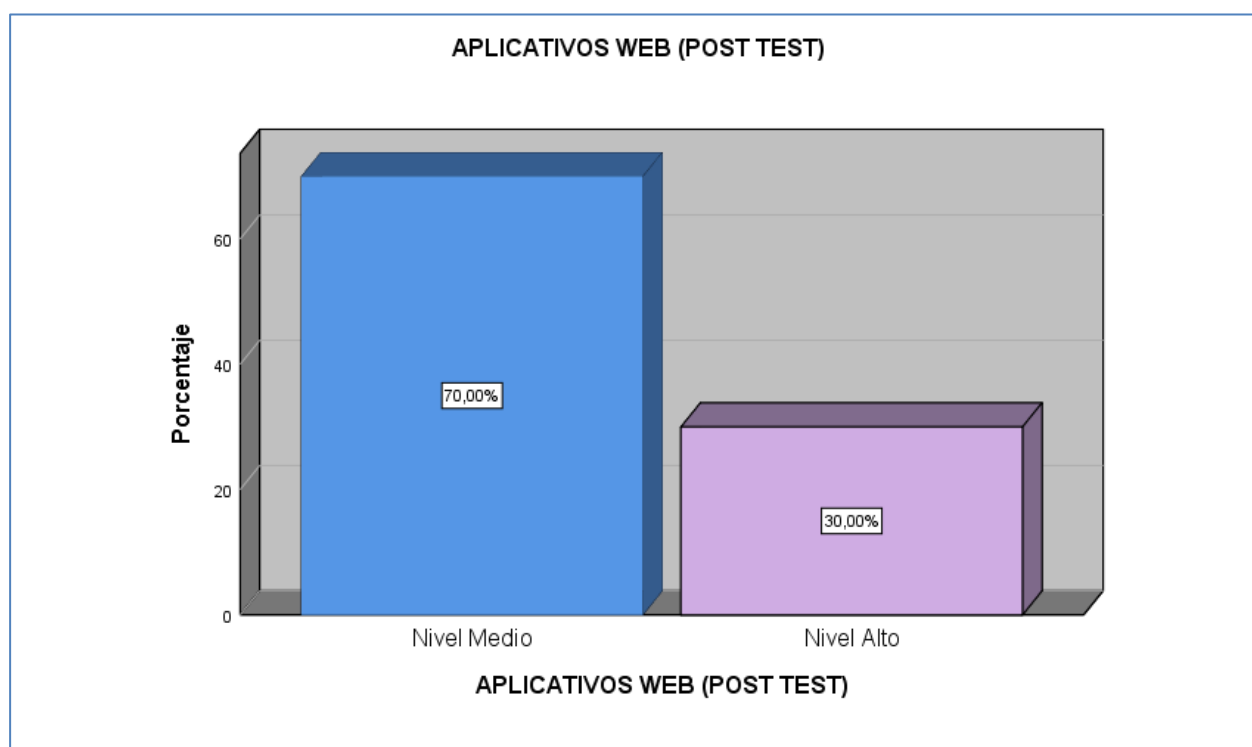
De acuerdo a la tabla 6 y figura 14, muestran los resultados desde la percepción de los encuestados, que el 50.0% tienen un nivel bajo, el 50% tiene un nivel medio, en las aplicaciones Web en la dimensión Calidad del Proceso, antes de realizar proceso de Desarrollo de Software basado en DevOps, de una institución Financiera.

#### 4.1.4. Variable Aplicativos Web (Post Test)

**Tabla 7.**  
Aplicativos Web (Post Test)

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válido	Nivel Medio	7	70,0	70,0	70,0
	Nivel Alto	3	30,0	30,0	100,0
	Total	10	100,0	100,0	

Fuente: elaboración propia con SPSS v.25



**Figura 15.** Aplicaciones web (Post test)

Fuente: Elaboración propia con SPSS v.25

#### Interpretación

De acuerdo a la tabla 7 y figura 15, muestran los resultados desde la percepción de los encuestados, que el 70.0% tienen un nivel medio, el 30% tiene un nivel alto, en las aplicaciones Web, después de realizar proceso de Desarrollo de Software basado en DevOps, de una institución Financiera.

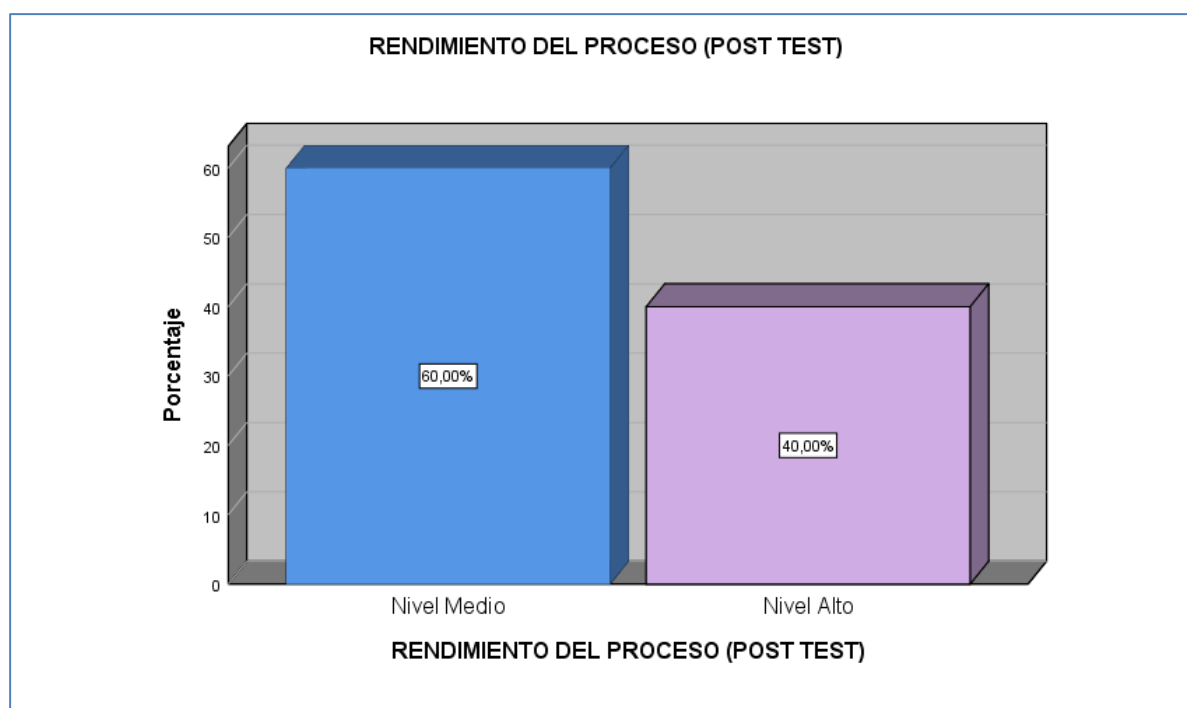
#### 4.1.5. Dimensión: Rendimiento del proceso (Post Test)

**Tabla 8.**

Rendimiento del proceso (Post Test)

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válido	Nivel Medio	6	60,0	60,0	60,0
	Nivel Alto	4	40,0	40,0	100,0
	Total	10	100,0	100,0	

Fuente: elaboración propia con SPSS v.25



**Figura 16.** Rendimiento del proceso (Post test)

**Fuente:** Elaboración propia con SPSS v.25

#### Interpretación

De acuerdo a la tabla 8 y figura 16, muestran los resultados desde la percepción de los encuestados, que el 60.0% tienen un nivel medio, el 40% tiene un nivel alto, en las aplicaciones Web en su dimensión rendimiento del proceso, después de realizar proceso de Desarrollo de Software basado en DevOps, de una institución Financiera.

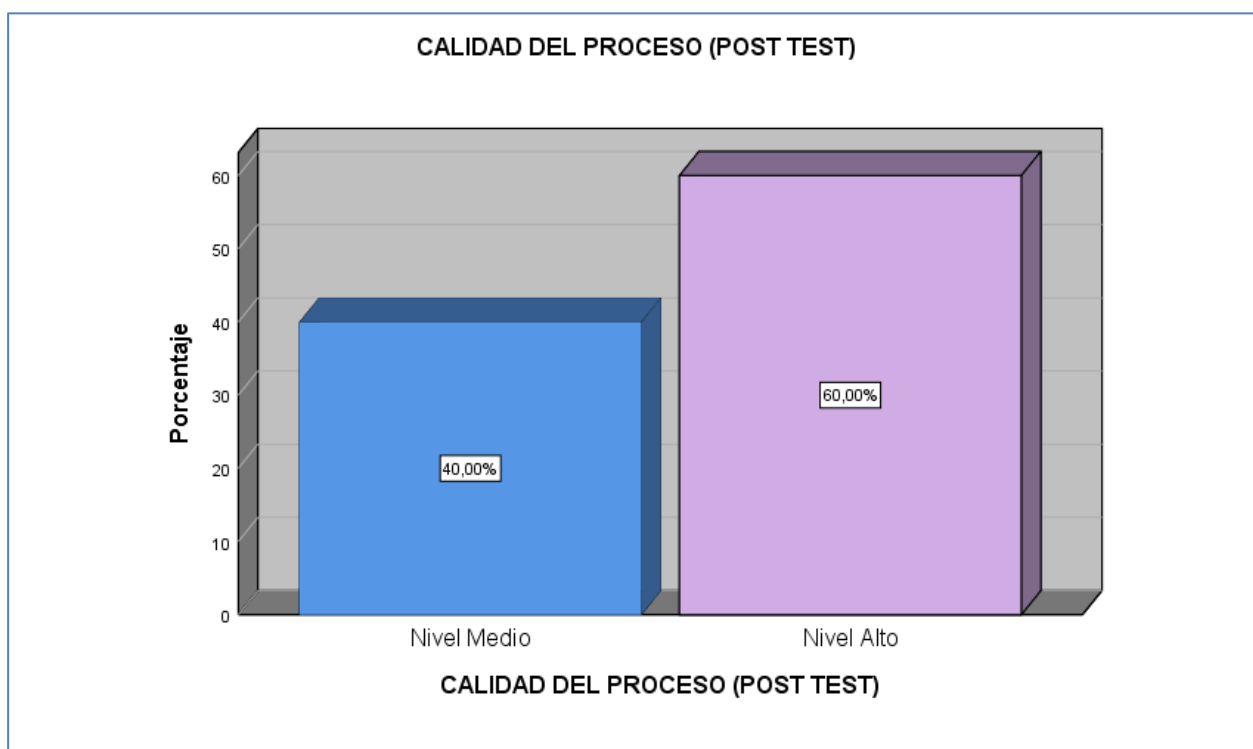
#### 4.1.6. Dimensión: Calidad del proceso (Post Test)

**Tabla 9.**

Calidad del proceso (Post Test)

		Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Válido	Nivel Medio	4	40,0	40,0	40,0
	Nivel Alto	6	60,0	60,0	100,0
	Total	10	100,0	100,0	

Fuente: elaboración propia con SPSS v.25



**Figura 17.** Calidad del proceso (Post test)

Fuente: Elaboración propia con SPSS v.25

#### Interpretación

De acuerdo a la tabla 8 y figura 6, muestran los resultados desde la percepción de los encuestados, que el 40.0% tienen un nivel medio, el 60% tiene un nivel alto, en las aplicaciones Web en su dimensión calidad del proceso, después de realizar proceso de Desarrollo de Software basado en DevOps, de una institución Financiera.

## 4.2. Análisis Inferencial – Prueba de Hipótesis

### Hipótesis general

H0. El proceso de desarrollo de software basado en DevOps no mejora la obtención de las aplicaciones web en una institución Financiera.

H1. El proceso de desarrollo de software basado en DevOps mejora la obtención de las aplicaciones web en una institución Financiera.

**Nivel de Significación** Se ha considerado  $\alpha = 0.05$

**Regla de decisión:** Si  $p \geq \alpha$ , se acepta H0; Si  $p < \alpha$ , se rechaza H0

Pvalor = Sig. Asintótica bilateral

**Prueba de estadística:** Debido a que las variables son de escala ordinal, utilizamos el procedimiento estadístico de Wilcoxon de la estadística paramétrica debido a que son el mismo grupo de observación, para determinar el efecto de la variable independiente sobre la dependiente.

Prueba de rangos con signo de Wilcoxon				
Rangos				
		N	Rango promedio	Suma de rangos
APLICACIONES WEB (POST TEST) - APLICACIONES WEB (PRE TEST)	Rangos negativos	0 <sup>a</sup>	,00	,00
	Rangos positivos	10 <sup>b</sup>	5,50	55,00
	Empates	0 <sup>c</sup>		
	Total	10		

a. APLICACIONES WEB (POST TEST) < APLICACIONES WEB (PRE TEST)  
b. APLICACIONES WEB (POST TEST) > APLICACIONES WEB (PRE TEST)  
c. APLICACIONES WEB (POST TEST) = APLICACIONES WEB (PRE TEST)

**Estadísticos de prueba<sup>a</sup>**

APLICACIONES WEB (POST TEST) - APLICACIONES WEB (PRE TEST)	
Z	-3,162 <sup>b</sup>
Sig. asintótica(bilateral)	,002

a. Prueba de rangos con signo de Wilcoxon  
b. Se basa en rangos negativos.

**Figura 18.** Resultados de Wilcoxon para el Aplicativo Web – Relación Pre y Post Test

**Fuente:** Elaboración propia con SPSS v.25



Según la Figura 18, el resultado fue significativo, con un  $p\text{valor}=\text{Sig}=0.002<0.05$ . La variable independiente proceso de desarrollo de software basado en DevOps tiene un efecto significativo sobre la variable dependiente aplicaciones Web. Por lo tanto, se rechaza la hipótesis de nula y se acepta la hipótesis de investigación. En conclusión, el proceso de desarrollo de software basado en DevOps mejora la obtención de las aplicaciones web en una institución Financiera.

### **Hipótesis específica 1**

H0. El proceso de desarrollo de software basado en DevOps no optimiza el rendimiento del proceso en las aplicaciones web de una institución Financiera.

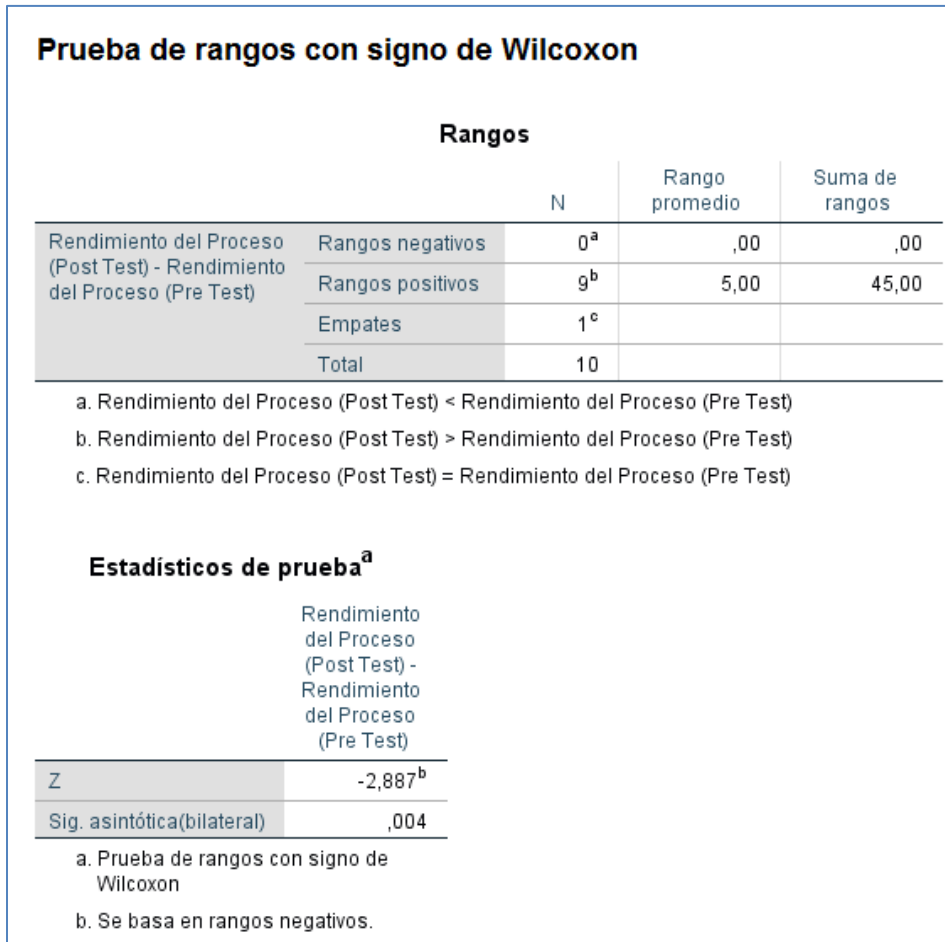
H1. El proceso de desarrollo de software basado en DevOps optimiza el rendimiento del proceso en las aplicaciones web de una institución Financiera.

**Nivel de Significación** Se ha considerado  $\alpha=0.05$

**Regla de decisión:** Si  $p \geq \alpha$ , se acepta H0; Si  $p < \alpha$ , se rechaza H0

Pvalor = Sig. Asintótica bilateral

**Prueba de estadística:** Debido a que las variables son de escala ordinal, utilizamos el procedimiento estadístico de Wilcoxon de la estadística paramétrica debido a que son el mismo grupo de observación, para determinar el efecto de la variable independiente sobre la dependiente.



**Figura 19.** Resultados de Wilcoxon para el Rendimiento del Proceso – Relación Pre y Post Test

**Fuente:** Elaboración propia con SPSS v.25

Según la Figura 19, el resultado fue significativo, con un pvalor=Sig=0.004<0.05. La variable independiente proceso de desarrollo de software basado en DevOps tiene un efecto significativo sobre la variable dependiente aplicaciones Web en la dimensión rendimiento del proceso. Por lo tanto, se rechaza la hipótesis de nula y se acepta la hipótesis de investigación. En conclusión, el proceso de desarrollo de software basado en DevOps optimiza el rendimiento del proceso en las aplicaciones web de una institución Financiera.

## **Hipótesis específica 2**

H0. El proceso de desarrollo de software basado en DevOps no optimiza la calidad del proceso de las aplicaciones web de una institución Financiera.

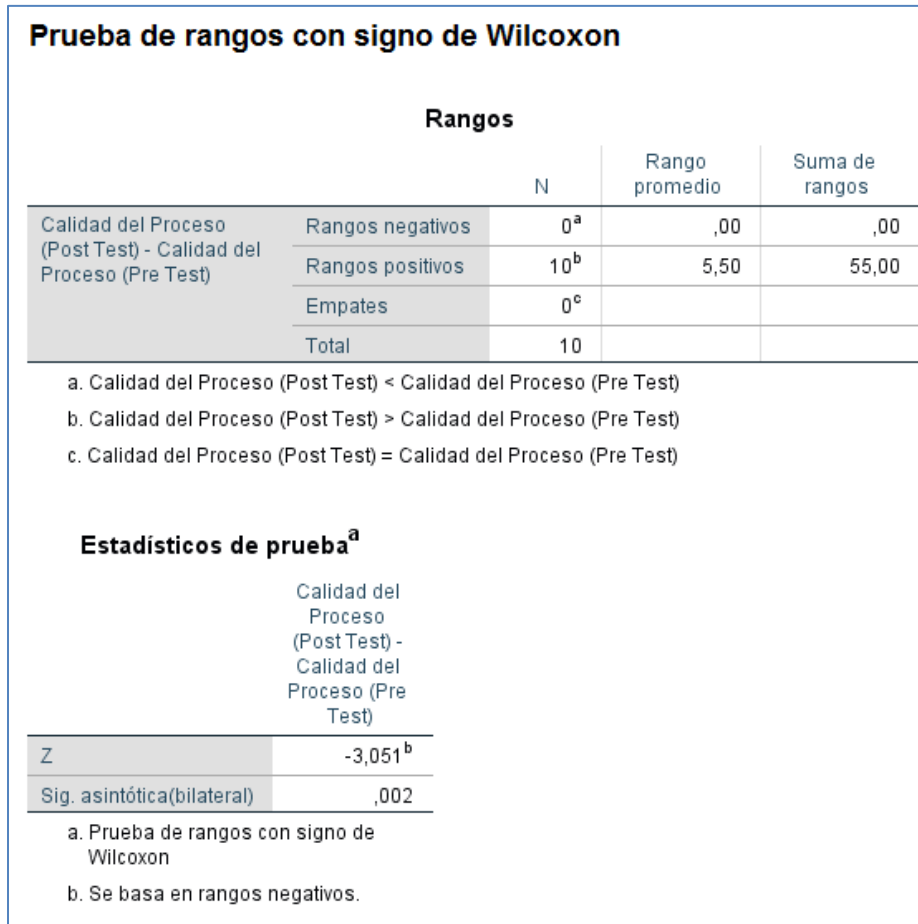
H1. El proceso de desarrollo de software basado en DevOps optimiza la calidad del proceso de las aplicaciones web de una institución Financiera

**Nivel de Significación** Se ha considerado  $\alpha= 0.05$

**Regla de decisión:** Si  $p \geq \alpha$ , se acepta H0; Si  $p < \alpha$ , se rechaza H0

Pvalor = Sig. Asintótica bilateral

**Prueba de estadística:** Debido a que las variables son de escala ordinal, utilizamos el procedimiento estadístico de Wilcoxon de la estadística paramétrica debido a que son el mismo grupo de observación, para determinar el efecto de la variable independiente sobre la dependiente.



**Figura 20.** Resultados de Wilcoxon para la Calidad del Proceso – Relación Pre y Post Test  
**Fuente:** Elaboración propia con SPSS v.25

Según la Figura 20, el resultado fue significativo, con un pvalor=Sig=0.002<0.05. La variable independiente proceso de desarrollo de software basado en DevOps tiene un efecto significativo sobre la variable dependiente aplicaciones Web en la dimensión calidad del proceso. Por lo tanto, se rechaza la hipótesis de nula y se acepta la hipótesis de investigación. En conclusión, el proceso de desarrollo de software basado en DevOps optimiza la calidad del proceso de las aplicaciones web de una institución Financiera.

## CAPÍTULO V

### DISCUSIÓN DE RESULTADOS

En base a los resultados como evidencias del trabajo empírico realizado, los objetivos de la investigación, antecedentes y el marco teórico realizado, se puede precisar que se cumplió el objetivo general de la investigación al determinar las diferencias en el pre y post test al aplicarse el proceso de desarrollo de software basado en DevOps mejora la obtención de los aplicativos web en una institución Financiera, tal como se corrobora con el trabajo de Muñoz (2015), cuyo enfoque fue cuantitativo y de tipo aplicada con un diseño pre experimental. Mejorando la eficiencia del departamento de desarrollo que utiliza el método tradicional, utilizando para ello herramientas open source apoyándose en la metodología DevOps. En tal sentido, IBM indicó que se reduce el tiempo necesario para tratar el feedback de los clientes. Asimismo, en el desarrollo y las operaciones, e incluso las pruebas, antes se organizaban en silos, DevOps las reúne para mejorar la agilidad. Asimismo, la investigación de Sandoval (2015), cuyos resultados fueron la construcción de un contenedor Docker en un Sistema de producción demostrando su adecuada tecnología para este caso específico y que apoya los resultados de la presente investigación.

Respecto rendimiento del proceso en las aplicaciones web, se cumplió el objetivo específico, donde se expresan las diferencias del pre y post aplicación del proceso de desarrollo de software basado en DevOps, en ese sentido, CMMI (2010), el rendimiento está vinculado con el procesamiento del aplicativo en el acceso y tiempo de respuesta a un determinado producto.

Respecto a la calidad del proceso de las aplicaciones web, se determinó el objetivo específico correspondiente, donde se evidencia las diferencias significativas del pre y post test en la aplicación del proceso de desarrollo de software basado en DevOps. Según CMMI (2010) es importante la calidad del proceso así como la naturaleza estética del contenido. Asimismo, en la evolución continua de este tipo de aplicaciones que se diferencia de cualquier otro tipo de aplicación convencional de software.

## CONCLUSIONES

1. El proceso de desarrollo de software basado en DevOps mejora la obtención de los aplicativos web en una institución Financiera, debido a que existe evidencia estadística suficiente para considerar la prueba significativa con un pvalor  $=0.002 < 0.05$ , y que se ve apoyado con un incremento en las entregas de las aplicaciones y despliegues del producto continuamente.
2. El proceso de desarrollo de software basado en DevOps optimiza el rendimiento del proceso en las aplicaciones web de una institución Financiera, debido a que existe evidencia estadística suficiente para considerar la prueba significativa con un pvalor  $=0.004 < 0.05$ , en donde existe reducción del tiempo de desarrollo de las aplicaciones, aumento de la regularidad y cumplimiento de entregas del producto, aumento de la productividad.
3. El proceso de desarrollo de software basado en DevOps optimiza la calidad de proceso de las aplicaciones web de una institución Financiera debido a que se existe evidencia estadística suficiente para considerar la prueba significativa con un pvalor  $=0.002 < 0.05$ , en donde se tiene como resultados el aumento de la eficiencia y eficacia del proceso, aumento de la aceptación por parte de los clientes, reducción del tiempo invertido en cambios para los aplicativos, mejora en el uso de recursos en la organización.

## **RECOMENDACIONES**

1. Programar charlas y seminarios a fin de procurar la integración y entendimiento de los equipos de Desarrollo y Operaciones, que centralice la visión de éxito de los proyectos como consecuencia del resultado de esfuerzos compartidos en donde uno refuerce al otro ante la claridad de competencias del otro.
2. Se recomienda capacitar a los integrantes de los equipos de desarrollo en el uso y configuración de herramientas tanto open source como de pago a fin de que puedan continuar con el aprovechamiento de éstas nuevas tecnologías y adecuarlas a los diferentes tipos de proyectos en los que participen, ya que ha sido demostrado que las dependencias con mayor disponibilidad agilizan de forma notable el proceso de desarrollo de las aplicaciones.
3. Continuar aprovechando las herramientas existentes ya que permiten disponer de tiempo antes invertido en el mantenimiento de la infraestructura y dedicarlo a las tareas propias del desarrollo y cumplimiento de los requerimientos de forma más eficiente y efectiva.

## REFERENCIAS BIBLIOGRÁFICAS

1. PRESSMAN, Roger S. Ingeniería del Software. Un enfoque práctico. 7ma.ed. México: McGraw-Hill Interamericana Editores, S.A. DE C.V., 2010, 805 pp.  
ISBN: 978-607-15-0314-5
2. SOMMERVILLE, Ian. Software Engineering. 9na.ed. United States of America: Pearson Education, Inc., 2011, 790 pp.  
ISBN 10: 0-13-703515-2 / ISBN 13: 978-0-13-703515-1
3. MAHAPATRA, R. Software Engineering. New Delhi: Khana Book Publishing, 2016, 345 pp.  
ISBN: 978-93-82609-68-1
4. SHARMA, Sanjeev. The DevOps Adoption Playbook: A Guide to Adopting DevOps in a Multi- Speed IT Enterprise. United States of America: John Wiley & Sons, Inc., 2016, 368 pp.  
ISBN: 978-1-119-30874-4
5. The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security In Techonology Organizations by Kin Gene [et al.]. United States of America: Digital Bindery, 2016, 184 pp.
6. Software Engineering Institute. CMMI® para Desarrollo, Versión 1.3. United States of America: Carnegie Mellon University, 2010, 555 pp.
7. RAVICHANDRAN, Aruna, TAYLOR, Kieran y WATERHOUSE, Peter. DevOps for Digital Leaders. Reignite Business with a Modern DevOps – Enabled Software Factory. USA: CA Technologies, 2016, 179 pp.  
ISBN-13 (electronic): 978-1-4842-1842-6
8. HERNÁNDEZ, Roberto, FERNÁNDEZ, Carlos, BAPTISTA, Pilar. Metodología de la Investigación. 6ª.ed. México: McGraw-Hill Interamericana Editores, S.A. DE C.V.,



2014, 634 pp.

ISBN: 978-1-4562-2396-0

9. SHARMA, Sanjeev, COYNE, Bernie. DevOps for Dummies. 2nd IBM Limited Edition. USA: John Wiley & Sons, Inc., 2015, 76 pp.  
ISBN: 978-1-119-04705-6 (pbk); ISBN: 978-1-119-04729-2 (ebk)
10. DEVI, Anitha. DevOps implementation framework for Agile-based large financial organizations. Tesis (Master of Science). Países Bajos: Utrecht University, 2018, 123 pp.
11. FRANÇA, B.B.N.D., JUNIOR, H.J., and TRAVASSOS, G.H., 2016. Characterizing DevOps by Hearing Multiple Voices. In Proceedings of the 30th Brazilian Symposium on Software Engineering. Brazil: Maringá - PR, Brazil 2016, ACM, 53-62.
12. LWAKATAREE, Lucy. DevOps adoption and implementation in software development practice. Concept, practices, benefits and challenges. Tesis (Doctoral Training). Finland: University of Oulu, 2017, 114 pp.
13. PEKKI, Juho. Implementing modern DevOps development environment for training Case: N4S@JAMK. Tesis. Jamk University of Applied Sciences, 2017, 84 pp.
14. MUÑOZ, Simón. Aplicación de herramientas DevOps en entornos de Desarrollo Web. Tesis. España: Universidad Politécnica de Valencia, 2015. 113 pp.
15. DE FEIJTER, Rico. En su tesis: “Towards the adoption of DevOps in software product organizations: A maturity model approach”. Technical Report UU-CS-2017-009. Países Bajos: Utrecht University, 2017, 173 pp.
16. SANDOVAL, Robert. A Case Study in Enabling DevOps using Docker. Tesis Master of Science in Engineering. Texas: University of Texas at Austin, 2015, 45 pp.

## **ANEXOS**

## ANEXO 01: MATRIZ DE CONSISTENCIA

**Título:** Proceso de Desarrollo de Software basado en DevOps para las aplicaciones Web de una institución Financiera

**Autor:** Antón Sebastián, Magally Edith

PROBLEMA	OBJETIVO	HIPÓTESIS	OPERACIONALIZACIÓN		
			VARIABLE	INDICADORES	METODOLOGÍA
<p><b>Problema General:</b></p> <p>¿En qué medida el proceso de desarrollo de software basado en DevOps mejora la obtención de aplicaciones web en una institución Financiera?</p>	<p><b>Objetivo General:</b></p> <p>Determinar en qué medida el proceso de desarrollo de software basado en DevOps mejora la obtención de aplicaciones web en una institución Financiera.</p>	<p><b>Hipótesis General:</b></p> <p>El proceso de desarrollo de software basado en DevOps mejora la obtención de las aplicaciones web en una institución Financiera.</p>	<p><b>Variable Independiente:</b></p> <p>Proceso de desarrollo de software basado en DevOps</p>	<p><b>Dimensiones e indicadores de la variable Independiente:</b></p> <ol style="list-style-type: none"> <li>1. Cliente y valor de negocio <ul style="list-style-type: none"> <li>- Ingresos por historia de usuarios.</li> <li>- Plazos de entrega</li> </ul> </li> <li>2. Eficiencia y Efectividad <ul style="list-style-type: none"> <li>- Velocidad de implementación</li> <li>- Reducción de costos</li> <li>- Costos por transacción</li> </ul> </li> <li>3. Calidad y velocidad <ul style="list-style-type: none"> <li>- Tasas de reversión</li> <li>- Frecuencia de despliegues</li> <li>- Tiempos por ciclo</li> <li>- Costo de soporte operativo</li> </ul> </li> </ol>	<p><b>Tipo:</b></p> <p>Aplicada</p> <p><b>Enfoque:</b></p> <p>Cuantitativo de corte longitudinal.</p> <p><b>Diseño:</b></p> <p>Pre Experimental</p> <p><b>Población:</b></p> <p>Integrantes del equipo de desarrollo de aplicaciones web de la institución financiera.</p> <p><b>Técnica de recolección de datos:</b></p> <p>Encuesta / Cuestionario</p> <p><b>Ficha bibliográfica:</b></p> <p>Tesis Artículos científicos</p>
<p><b>Problemas específicos:</b></p> <p>a. ¿Cuál es el efecto del proceso de desarrollo de software basado en DevOps para la optimización en el rendimiento del proceso de las aplicaciones web de una institución financiera?</p> <p>b. ¿Cómo influye el proceso de desarrollo</p>	<p><b>Objetivos Específicos:</b></p> <p>a. Precisar el efecto del proceso de desarrollo de software basado en DevOps para la optimización de rendimiento del proceso de las aplicaciones web de una institución Financiera.</p> <p>b. Definir cómo influye el proceso de</p>	<p><b>Hipótesis Específicas:</b></p> <p>a.El proceso de desarrollo de software basado en DevOps optimiza el rendimiento del proceso en las aplicaciones web de una institución Financiera .</p> <p>b.El proceso de desarrollo de software basado en</p>	<p><b>Variable Dependiente:</b></p> <p>Aplicaciones web de</p>	<p><b>Dimensiones e indicadores de la variable Dependiente:</b></p> <ol style="list-style-type: none"> <li>1. Rendimiento del proceso</li> </ol>	<p><b>Fuentes:</b></p> <p>Bibliografía Internet</p>

---

de software basado en DevOps en la optimización de la calidad del proceso de las aplicaciones web de una institución financiera.?

desarrollo de software basado en DevOps en la optimización de la calidad del proceso de las aplicaciones web de una institución Financiera.

DevOps optimiza la calidad del proceso de las aplicaciones web de una institución Financiera .

una institución financiera.

- Tiempo de retrabajo.
- Defectos hallados en la verificación del producto.
- Defectos no detectados.
- Número y gravedad de defectos encontrados.
- Tiempo no productivo.
- Tiempo medio de resolución de problemas en el entorno productivo.
- Frecuencia de despliegue.

## 2. Calidad del proceso:

- Tiempo medio entre fallos.
  - Número y gravedad de los defectos del producto liberado.
  - Uso de recursos críticos.
  - Número y gravedad de las reclamaciones del cliente por el servicio proporcionado.
  - Porcentaje de éxito en Test automáticos
  - Defectos encontrados en el entorno productivo.
  - Seguridad de los aplicativos
-

## ANEXO 02: INSTRUMENTO DE INVESTIGACIÓN

### CERTIFICADO DE VALIDEZ DE CONTENIDO DEL INSTRUMENTO QUE MIDE LOS APLICATIVOS WEB DE UNA INSTITUCIÓN FINANCIERA

DIMENSIÓN: RENDIMIENTO DEL PROCESO	Pertinencia <sup>1</sup>		Relevancia <sup>2</sup>		Claridad <sup>3</sup>	
	Si	No	Si	No	Si	No
1. ¿Cómo percibe el porcentaje de tiempo de retrabajo durante el tiempo de desarrollo de los aplicativos web?	X		X		X	
2. ¿Cómo considera el porcentaje de defectos hallados en las actividades de verificación de los aplicativos web?	X		X		X	
3. ¿Cómo califica la tasa de defectos no detectados en los aplicativos web?	X		X		X	
4. ¿Cuál es la valoración que le da al número y gravedad de defectos encontrados en el producto final?	X		X		X	
5. ¿Qué valoración le otorga al porcentaje de tiempo no productivo durante el proceso de desarrollo de los aplicativos web?	X		X		X	
6. ¿Considera que el tiempo medio de resolución de problemas en el entorno productivo está por encima de lo estimado?	X		X		X	
7. ¿Cómo percibe la dependencia de otros equipos para poder realizar los despliegues hacia el ambiente productivo?	X		X		X	
<b>DIMENSIÓN: CALIDAD DEL APLICATIVO</b>						
8. ¿Cómo califica Ud. el tiempo medio entre fallos para la disponibilidad de los servicios en los aplicativos web?	X		X		X	
9. ¿Cómo considera el número y gravedad de los defectos de los aplicativos web liberados?	X		X		X	
10. ¿Cómo considera el uso de recursos críticos como roll back para mitigar los errores de los aplicativos en producción?	X		X		X	
11. ¿Cómo califica el número y gravedad de las reclamaciones del cliente por el servicio proporcionado en el desarrollo de los aplicativos web?	X		X		X	
12. ¿El porcentaje de insatisfacción de los test automáticos en la evaluación de los aplicativos antes y después de su pase al ambiente productivo?	X		X		X	
13. ¿Qué valoración le otorga al porcentaje de defectos encontrados a los aplicativos web en el ambiente productivo?	X		X		X	

14. ¿ Cómo considera el porcentaje de mejoras pendientes de implementar en la seguridad de los aplicativos web en los ambientes productivos?	X		X		X	
--	---	--	---	--	---	--

Observaciones (precisar si hay suficiencia): EXISTE Suficiencia

Opinión de aplicabilidad:    Aplicable [ ]        Aplicable después de corregir [ ]        No aplicable [ ]

Apellidos y nombres del juez validador. Dr/ Mg:    Mg. Luis Torres Cabanillas        DNI:08404690

Especialidad del validador: Ing. Estadístico Nro. 49863

04 de mayo del 2019

- <sup>1</sup>Pertinencia: El ítem corresponde al concepto teórico formulado.
- <sup>2</sup>Relevancia: El ítem es apropiado para representar al componente o dimensión específica del constructo
- <sup>3</sup>Claridad: Se entiende sin dificultad alguna el enunciado del ítem, es conciso, exacto y directo

**Nota:** Suficiencia, se dice suficiencia cuando los ítems planteados son suficientes para medir la dimensión

  
 -----  
**Firma del Experto Informante.**

**CERTIFICADO DE VALIDEZ DE CONTENIDO DEL INSTRUMENTO QUE MIDE LOS APLICATIVOS WEB DE UNA INSTITUCIÓN FINANCIERA**

DIMENSIÓN: RENDIMIENTO DEL PROCESO	Pertinencia <sup>1</sup>		Relevancia <sup>2</sup>		Claridad <sup>3</sup>	
	Si	No	Si	No	Si	No
1. ¿Cómo percibe el porcentaje de tiempo de retrabajo durante el tiempo de desarrollo de los aplicativos web?	X		X		X	
2. ¿Cómo considera el porcentaje de defectos hallados en las actividades de verificación de los aplicativos web?	X		X		X	
3. ¿Cómo califica la tasa de defectos no detectados en los aplicativos web?	X		X		X	
4. ¿Cuál es la valoración que le da al número y gravedad de defectos encontrados en el producto final?	X		X		X	
5. ¿Qué valoración le otorga al porcentaje de tiempo no productivo durante el proceso de desarrollo de los aplicativos web?	X		X		X	
6. ¿Considera que el tiempo medio de resolución de problemas en el entorno productivo está por encima de lo estimado?	X		X		X	
7. ¿Cómo percibe la dependencia de otros equipos para poder realizar los despliegues hacia el ambiente productivo?	X		X		X	
<b>DIMENSIÓN: CALIDAD DEL APLICATIVO</b>						
8. ¿Cómo califica Ud. el tiempo medio entre fallos para la disponibilidad de los servicios en los aplicativos web?	X		X		X	
9. ¿Cómo considera el número y gravedad de los defectos de los aplicativos web liberados?	X		X		X	
10. ¿Cómo considera el uso de recursos críticos como roll back para mitigar los errores de los aplicativos en producción?	X		X		X	
11. ¿Cómo califica el número y gravedad de las reclamaciones del cliente por el servicio proporcionado en el desarrollo de los aplicativos web?	X		X		X	
12. ¿El porcentaje de insatisfacción de los test automáticos en la evaluación de los aplicativos antes y después de su pase al ambiente productivo?	X		X		X	
13. ¿Qué valoración le otorga al porcentaje de defectos encontrados a los aplicativos web en el ambiente productivo?	X		X		X	

14. ¿ Cómo considera el porcentaje de mejoras pendientes de implementar en la seguridad de los aplicativos web en los ambientes productivos?

X		X		X	
---	--	---	--	---	--

Observaciones (precisar si hay suficiencia): Suficiente

Opinión de aplicabilidad:    Aplicable [ ]    Aplicable después de corregir [ ]    No aplicable [ ]

Apellidos y nombres del juez validador. Dr/ Mg:    Dra. Karin Corina Rojas Romero    DNI: 32645104

Especialidad del validador: Ing. Computación y Sistemas CIP. 110497

04 de mayo del 2019

- <sup>1</sup>**Pertinencia:** El ítem corresponde al concepto teórico formulado.
- <sup>2</sup>**Relevancia:** El ítem es apropiado para representar al componente o dimensión específica del constructo
- <sup>3</sup>**Claridad:** Se entiende sin dificultad alguna el enunciado del ítem, es conciso, exacto y directo

**Nota:** Suficiencia, se dice suficiencia cuando los ítems planteados son suficientes para medir la dimensión

  
Dra. Karin C. Rojas Romero  
ING. COMP. Y SISTEMAS  
R. CIP. 110497

-----  
**Firma del Experto Informante.**



**CERTIFICADO DE VALIDEZ DE CONTENIDO DEL INSTRUMENTO QUE MIDE LOS APLICATIVOS WEB DE UNA INSTITUCIÓN FINANCIERA**

DIMENSIÓN: RENDIMIENTO DEL PROCESO	Pertinencia <sup>1</sup>		Relevancia <sup>2</sup>		Claridad <sup>3</sup>	
	Si	No	Si	No	Si	No
1. ¿Cómo percibe el porcentaje de tiempo de retrabajo durante el tiempo de desarrollo de los aplicativos web?	X		X		X	
2. ¿Cómo considera el porcentaje de defectos hallados en las actividades de verificación de los aplicativos web?	X		X		X	
3. ¿Cómo califica la tasa de defectos no detectados en los aplicativos web?	X		X		X	
4. ¿Cuál es la valoración que le da al número y gravedad de defectos encontrados en el producto final?	X		X		X	
5. ¿Qué valoración le otorga al porcentaje de tiempo no productivo durante el proceso de desarrollo de los aplicativos web?	X		X		X	
6. ¿Considera que el tiempo medio de resolución de problemas en el entorno productivo está por encima de lo estimado?	X		X		X	
7. ¿Cómo percibe la dependencia de otros equipos para poder realizar los despliegues hacia el ambiente productivo?	X		X		X	
<b>DIMENSIÓN: CALIDAD DEL APLICATIVO</b>						
8. ¿Cómo califica Ud. el tiempo medio entre fallos para la disponibilidad de los servicios en los aplicativos web?	X		X		X	
9. ¿Cómo considera el número y gravedad de los defectos de los aplicativos web liberados?	X		X		X	
10. ¿Cómo considera el uso de recursos críticos como roll back para mitigar los errores de los aplicativos en producción?	X		X		X	
11. ¿Cómo califica el número y gravedad de las reclamaciones del cliente por el servicio proporcionado en el desarrollo de los aplicativos web?	X		X		X	
12. ¿El porcentaje de insatisfacción de los test automáticos en la evaluación de los aplicativos antes y después de su pase al ambiente productivo?	X		X		X	
13. ¿Qué valoración le otorga al porcentaje de defectos encontrados a los aplicativos web en el ambiente productivo?	X		X		X	

14. ¿ Cómo considera el porcentaje de mejoras pendientes de implementar en la seguridad de los aplicativos web en los ambientes productivos?	X		X		X	
--	---	--	---	--	---	--

Observaciones (precisar si hay suficiencia): Suficiente

Opinión de aplicabilidad:    Aplicable [ ]    Aplicable después de corregir [ ]    No aplicable [ ]

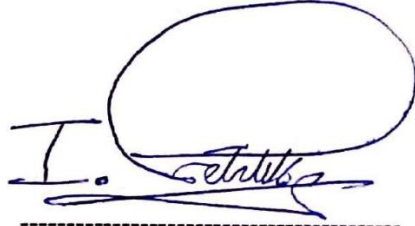
Apellidos y nombres del juez validador. Dr/ Mg:    Mg. Ivan Carlo Petrik Azabache    DNI: 10140461

Especialidad del validador: Ingeniería de Computación y Sistemas CIP. 91445

04 de mayo del 2019

- <sup>1</sup>**Pertinencia:** El ítem corresponde al concepto teórico formulado.
- <sup>2</sup>**Relevancia:** El ítem es apropiado para representar al componente o dimensión específica del constructo
- <sup>3</sup>**Claridad:** Se entiende sin dificultad alguna el enunciado del ítem, es conciso, exacto y directo

**Nota:** Suficiencia, se dice suficiencia cuando los ítems planteados son suficientes para medir la dimensión



Firma del Experto Informante.

**ANEXO 03: BASE DE DATOS**

**APLICACIONES WEB - PRE TEST**

	RENDIMIENTO DEL PROCESO							CALIDAD DEL APLICATIVO							Y	Y1	Y2	y	y1	y2
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14						
Per1	5	5	4	4	5	5	5	5	5	5	5	5	5	5	68	33	35	1	1	1
Per2	4	4	4	4	4	5	5	4	5	4	5	5	4	4	61	30	31	1	1	1
Per3	5	4	4	5	4	4	5	4	4	4	5	5	5	5	63	31	32	1	1	1
Per4	5	5	5	5	3	5	5	3	5	3	5	3	3	3	58	33	25	1	1	2
Per5	4	3	3	4	3	3	5	5	5	4	5	5	5	5	59	25	34	1	2	1
Per6	3	5	5	5	3	5	5	5	4	3	3	4	3	4	57	31	26	1	1	2
Per7	3	3	3	4	3	3	5	3	3	3	4	4	5	5	51	24	27	2	2	1
Per8	5	5	5	4	5	5	5	3	3	3	3	3	3	3	55	34	21	1	1	2
Per9	3	3	3	4	3	5	5	3	3	3	3	3	3	3	47	26	21	2	2	2
Per10	3	3	3	3	3	4	5	3	3	4	3	4	3	3	47	24	23	2	2	2

## RECOPIACIÓN DE RESULTADOS DEL CUESTIONARIO APLICADO

### APLICACIONES WEB - POST-TEST

	RENDIMIENTO DEL PROCESO							CALIDAD DEL APLICATIVO							Y	Y1	Y2	y	y1	y2
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14						
Per1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	42	21	21	2	2	2
Per2	2	2	2	2	2	3	3	3	3	3	3	3	3	3	37	16	21	2	3	2
Per3	3	3	3	3	2	2	3	2	2	2	3	3	3	3	37	19	18	2	2	2
Per4	3	3	3	3	2	3	3	2	3	2	3	2	2	2	36	20	16	2	2	3
Per5	3	2	2	2	2	2	3	3	3	2	3	3	3	3	36	16	20	2	3	2
Per6	2	3	3	3	2	3	3	3	2	2	2	2	2	2	34	19	15	2	2	3
Per7	2	2	2	2	1	2	3	2	2	2	2	2	3	3	30	14	16	3	3	3
Per8	3	3	3	3	3	3	3	2	2	2	2	2	2	2	35	21	14	2	2	3
Per9	2	2	2	3	2	3	3	2	2	2	2	2	2	2	31	17	14	3	2	3
Per10	2	2	2	2	2	3	3	2	2	2	2	2	2	2	30	16	14	3	3	3

## ANEXO 05: TECNOLOGÍAS Y SERVICIOS

### 1. Lambda:

AWS Lambda es un servicio informático sin servidor proporcionado por AWS.

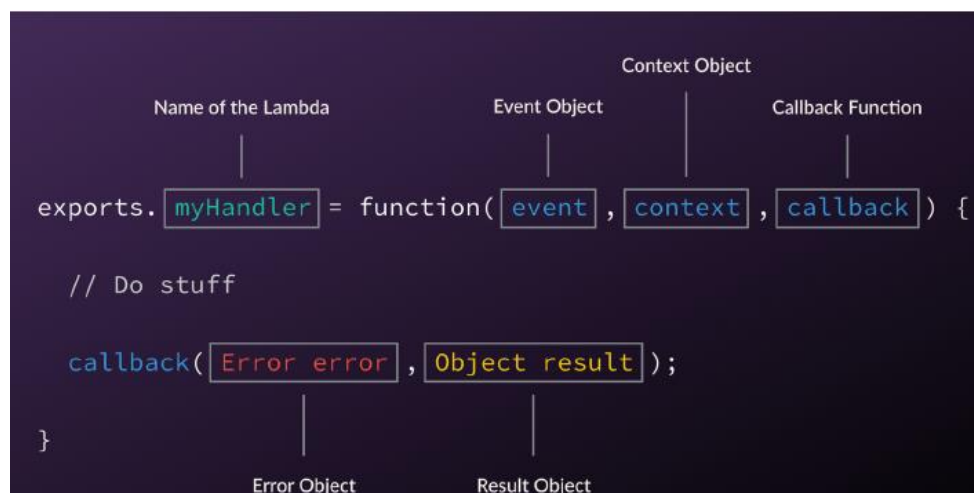
Permite ejecutar código sin preocuparse de los servidores en donde sólo paga el tiempo que se consume, sin cargo alguno por código que no se ejecute.

Permite ejecutar código para casi cualquier tipo de aplicación o servicio backend.

Especificaciones Lambda:

Lambda soporta las siguientes runtimes.

17. Node.js: v8.10 and v6.10
18. Java 8
19. Python: 3.6 and 2.7
20. .NET Core: 1.0.1 and 2.0
21. Go 1.x
22. Ruby 2.5
23. Rust
- 24.



*Figura 21.* Aspecto de función Lambda

*Fuente:* Serverless (2019)

Aquí `myHandler` es el nombre de nuestra función Lambda. El objeto `event` contiene toda la información sobre el evento que desencadenó este Lambda. En el caso de una

solicitud HTTP, habrá información sobre la solicitud HTTP específica. El objeto `context` contiene información sobre el tiempo de ejecución en el que se está ejecutando nuestra función Lambda. Después hacemos todo el trabajo dentro de nuestra función Lambda, simplemente llamamos a la función de `callback` con los resultados (o el error) y AWS responderá a la solicitud HTTP con ella.

### Funciones de embalaje

Las funciones Lambda se deben empaquetar y enviar a AWS. Este suele ser un proceso que comprime la función y todas sus dependencias y la carga en un contenedor S3. Y dejar que AWS sepa que desea utilizar este paquete cuando se produce un evento específico. Para ayudarnos con este proceso utilizamos el Framework Serverless.

### Modelo de ejecución

AWS se encarga por completo del contenedor y los recursos que utiliza para ejecutar nuestra función. Se activa cuando se produce un evento y se apaga si no se está utilizando. Si se realizan solicitudes adicionales mientras se está sirviendo el evento original, se abre un nuevo contenedor para atender una solicitud. Esto significa que si estamos experimentando un pico de uso, el proveedor de la nube simplemente crea múltiples instancias del contenedor con nuestra función para atender esas solicitudes.

Cada solicitud (o evento) es atendida por una sola instancia de una función Lambda. Esto significa que no va a manejar solicitudes concurrentes en su código. AWS abre un contenedor cada vez que hay una nueva solicitud. Hace algunas optimizaciones aquí. Se mantendrá en el contenedor durante unos minutos (5 a 15 minutos según la carga) para que pueda responder a las solicitudes posteriores sin un arranque en frío.

### Funciones sin estado

El modelo de ejecución anterior hace que Lambda funcione efectivamente sin estado. Esto significa que cada vez que su función Lambda se activa por un evento, se invoca en un entorno completamente nuevo. No tienes acceso al contexto de ejecución del evento anterior.

Sin embargo, debido a la optimización mencionada anteriormente, la función Lambda

real se invoca solo una vez por instanciación de contenedor. Recordemos que nuestras funciones se ejecutan dentro de contenedores. Entonces, cuando se invoca una función por primera vez, se ejecuta todo el código en nuestra función de manejador y se invoca la función de manejador. Si el contenedor aún está disponible para solicitudes posteriores, su función se invocará y no el código que lo rodea.

Por ejemplo, el método `createNewDbConnection` se llama una vez por instanciación de contenedor y no cada vez que se invoca la función Lambda. La función `myHandler` por otro lado se llama en cada invocación.

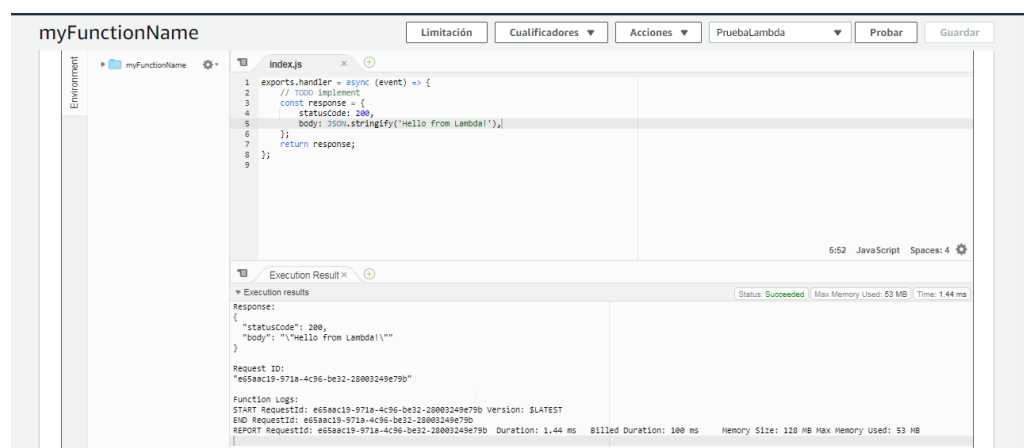
```
var dbConnection = createNewDbConnection();

exports.myHandler = function(event, context, callback) {
  var result = dbConnection.makeQuery();
  callback(null, result);
};
```

## Precios

El punto principal por lo que las aplicaciones sin servidor son preferidas por sobre las aplicaciones alojadas en un servidor tradicional son:

1. Bajo mantenimiento
2. Bajo costo
3. Fácil de escalar



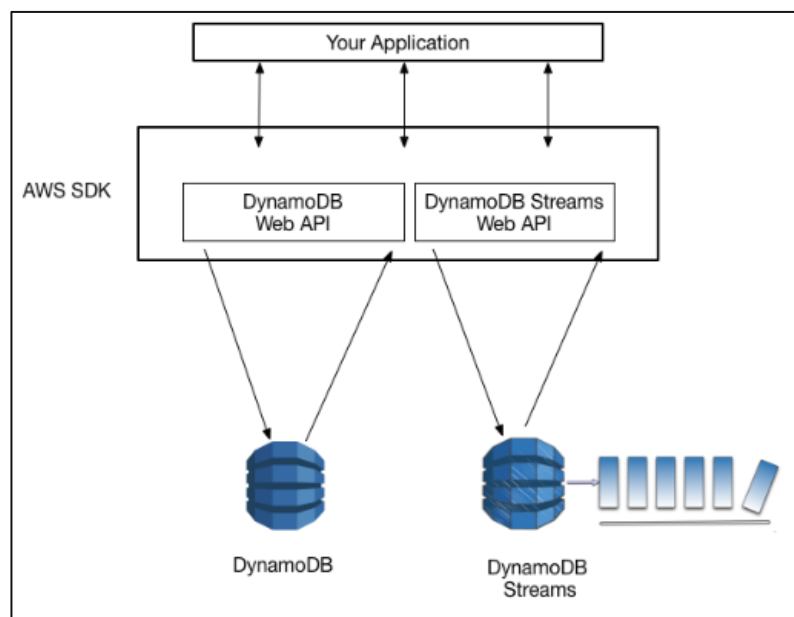
**Figura 22.** Aspecto de función Lambda II

**Fuente:** Servedless (2019)

## 2. Dynamo DB

Amazon DynamoDB es una base de datos NoSQL totalmente administrada que proporciona un rendimiento rápido y predecible con una escalabilidad perfecta. Similar a otras bases de datos, DynamoDB almacena los datos en tablas. Cada tabla contiene varios elementos, y cada elemento se compone de uno o más atributos, así mismo utilizan modelos alternativos de administración de datos como por ejemplo clave-valor o almacenamiento de documentos.

AWS mantiene puntos de enlace distintos para DynamoDB y Flujos de DynamoDB. Para usar las tablas y los índices de la base de datos, la aplicación tendrá que obtener acceso a un punto de enlace de DynamoDB. Para leer y procesar los registros de Flujos de DynamoDB, la aplicación tendrá que obtener acceso a un punto de enlace de Flujos de DynamoDB situado en la misma región.

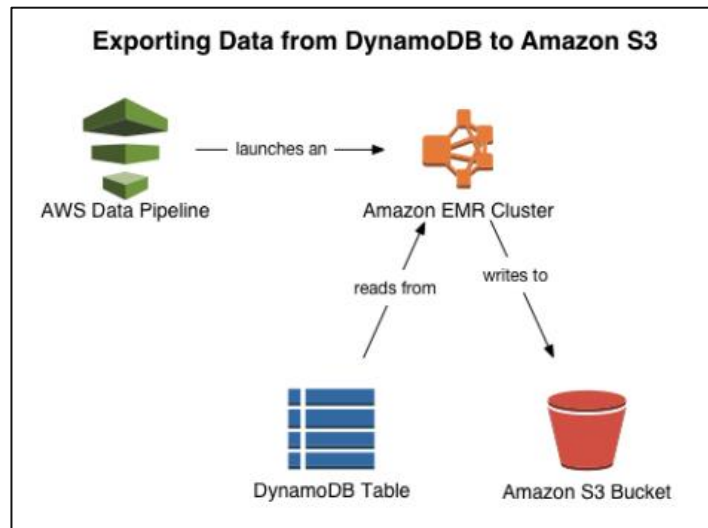


**Figura 23.** Puntos de enlace para flujo DynamoDB

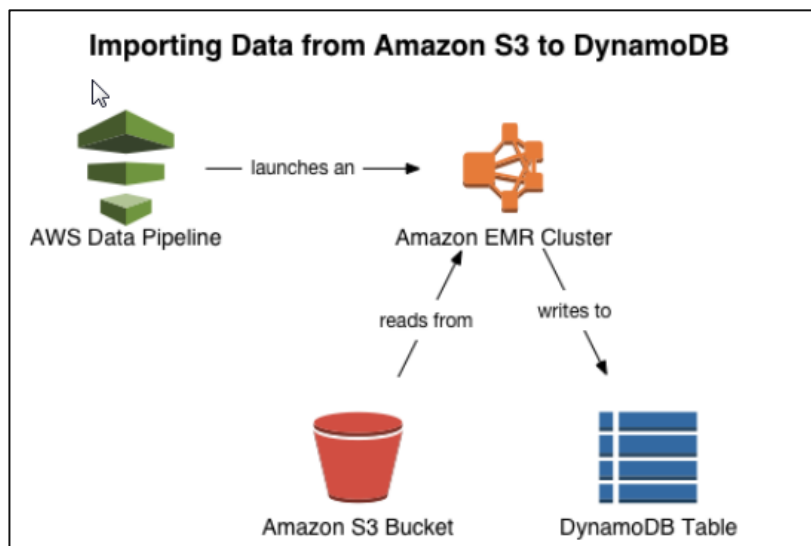
**Fuente:** Amazon Web Services

Si se trata de seguridad Amazon ofrece un entorno con mayor seguridad, nos referimos a Amazon Virtual Private Cloud (Amazon VPC) que no es más que una nube virtual privada aislada de otras redes incluida la red pública de Internet y en las cuales puede controlar un rango de direcciones IP, Gateway, configuraciones de seguridad, entre otros.





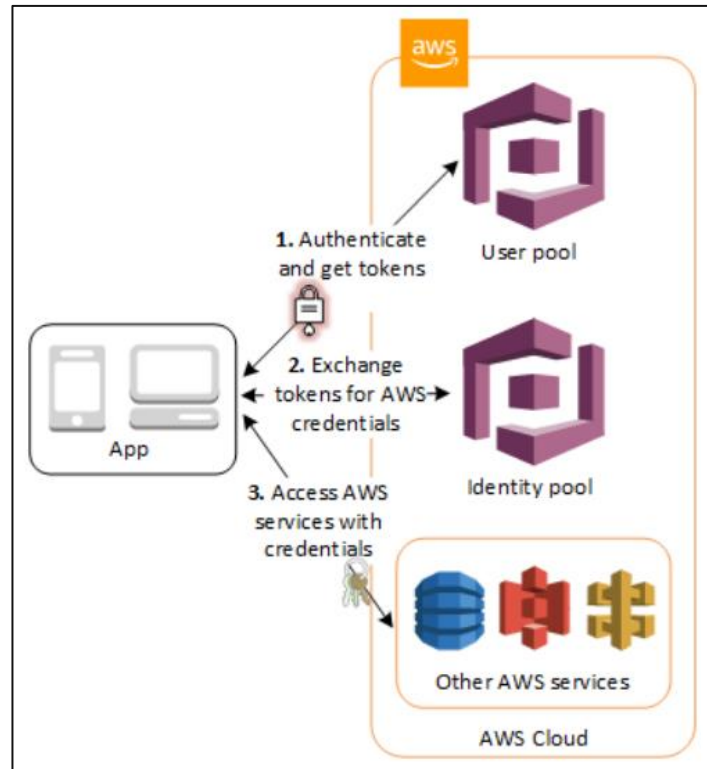
**Figura 24.** Exportando datos de DynamoDB hacia Amazon S3  
**Fuente:** Amazon Web Services (pp. 938)



**Figura 25.** Importando datos de Amazon S3 hacia DynamoDB  
**Fuente:** Amazon Web Services (pp. 938)

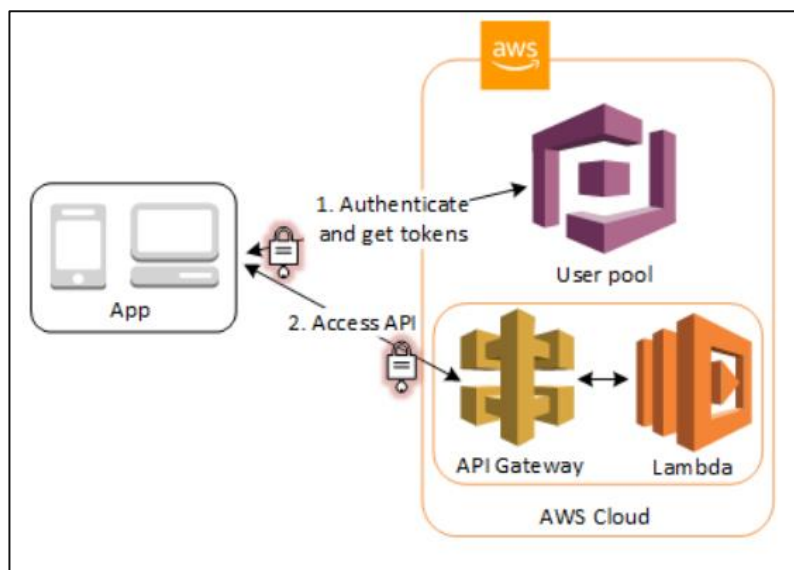
### 3. Cognito:

Amazon Cognito proporciona autenticación, autorización y administración de usuarios para las aplicaciones móviles y web.



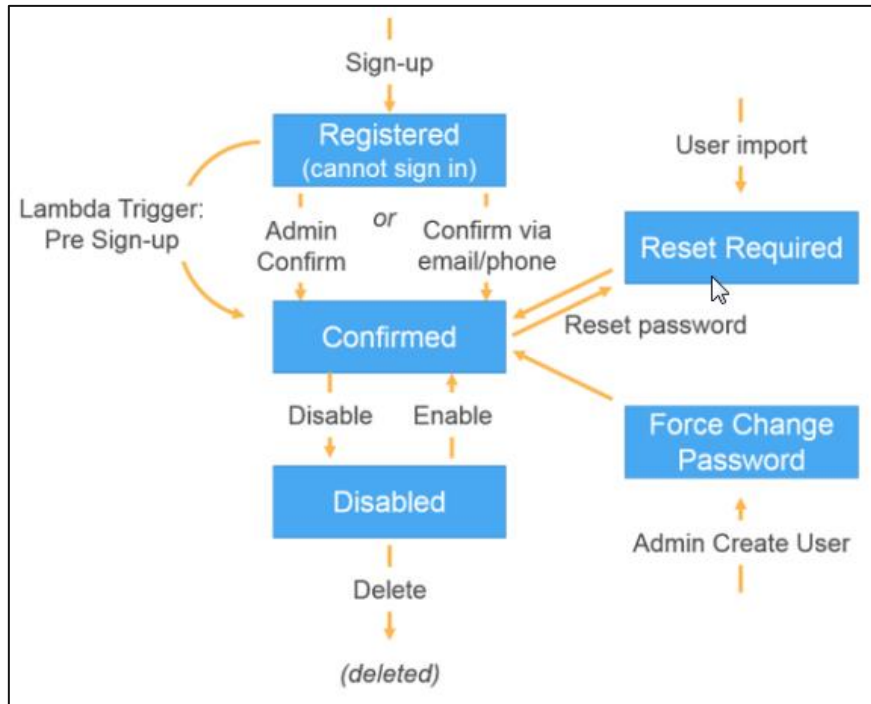
**Figura 26.** Componentes principales de Amazon Cognito.

**Fuente:** Amazon Cognito (p. 7)



**Figura 27.** Acceso a Recursos con API Gateway y Lambda.

**Fuente:** Amazon Cognito (p. 14)



**Figura 28.** Proceso de Confirmación de una cuenta de usuario en Cognito.

**Fuente:** Amazon Cognito (p. 14)

- *Estado Registrada* (sin confirmar), para los nuevos usuarios que se inscriben, la el usuario está habilitado pero no confirmado.
- *Estado confirmada*, cuenta confirmada por lo que el usuario está habilitado para iniciar sesión.
- *Estado restablecimiento de contraseña requerido*, cuenta de usuario confirmada, pero el usuario debe solicitar un código a fin de poder restablecer su contraseña.
- *Estado obligar a cambiar de contraseña*, cuenta de usuario está confirmada y puede ingresar con clave temporal, pero la primera vez que inicie sesión debe cambiar su contraseña.

**Create user**

**Username (Required)**

Send an invitation to this new user?

SMS (default)  Email

**Temporary password**

**Phone number**

Mark phone number as verified?

**Email**

Mark email as verified?

Create user

**Figura 29.** Creación de usuarios en Cognito.  
**Fuente:** Amazon Cognito (p. 190)

Users Groups

Import users Create user User name Search for value...

Username	Enabled	Account status	Email verified	Phone number verified	Updated	Created
7c0dc801ee66	Enabled	CONFIRMED	true	-	Nov 19, 2018 7:44:48 PM	Nov 19, 2018 7:44:48 PM
75f3de26d2b5	Enabled	CONFIRMED	true	true	Nov 7, 2018 8:59:35 PM	Nov 7, 2018 8:59:35 PM

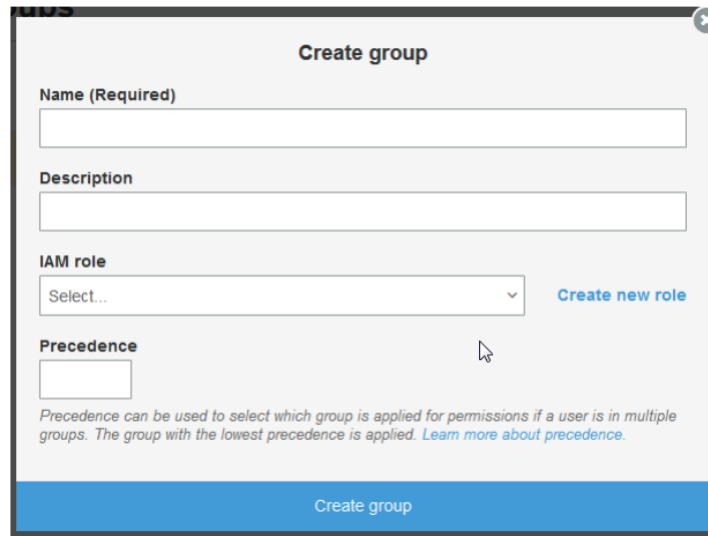
**Figura 30.** Contenido de usuarios en Cognito.  
**Fuente:** Amazon Cognito (p. 189)

Users Groups

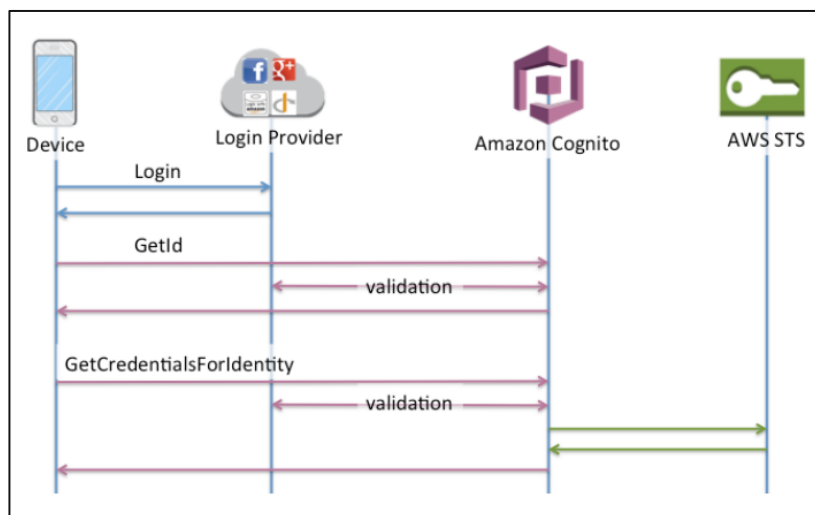
Create group

Group Name	Description	Precedence	Updated	Created
Group001	Admin	1		

**Figura 31.** Creación de grupos en Cognito.  
**Fuente:** Amazon Cognito (p. 192)



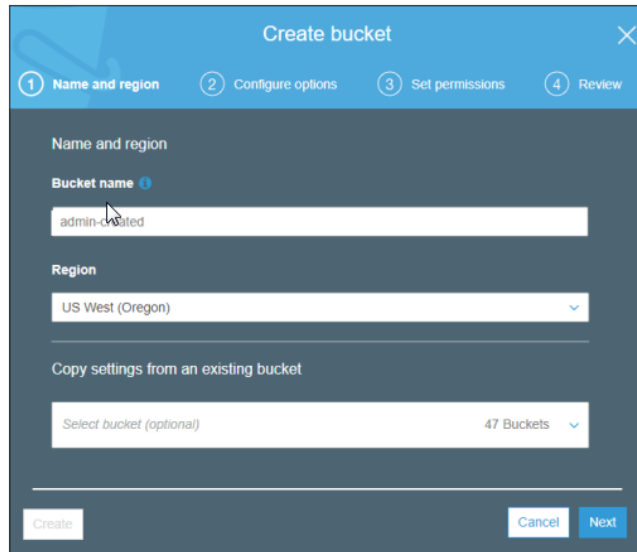
**Figura 32.** Creación de grupos en Cognito II.  
**Fuente:** Amazon Cognito (p. 192)



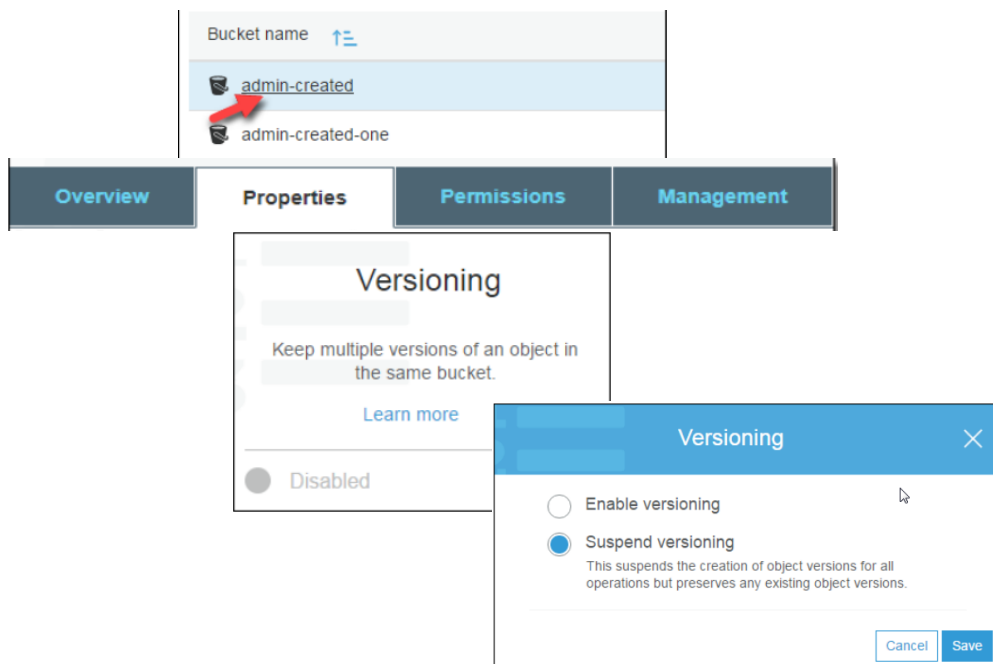
**Figura 33.** Flujo de autenticación Cognito.  
**Fuente:** Amazon Cognito (p. 258)

#### 4. S3:

Facilita la tarea de recopilar, almacenar y analizar datos a gran escala sea cual sea su formato. Proporciona almacenamiento casi ilimitado en internet, permite administrar buckets, objetos y carpetas en Amazon S3 valiéndonos de la consola de AWS.



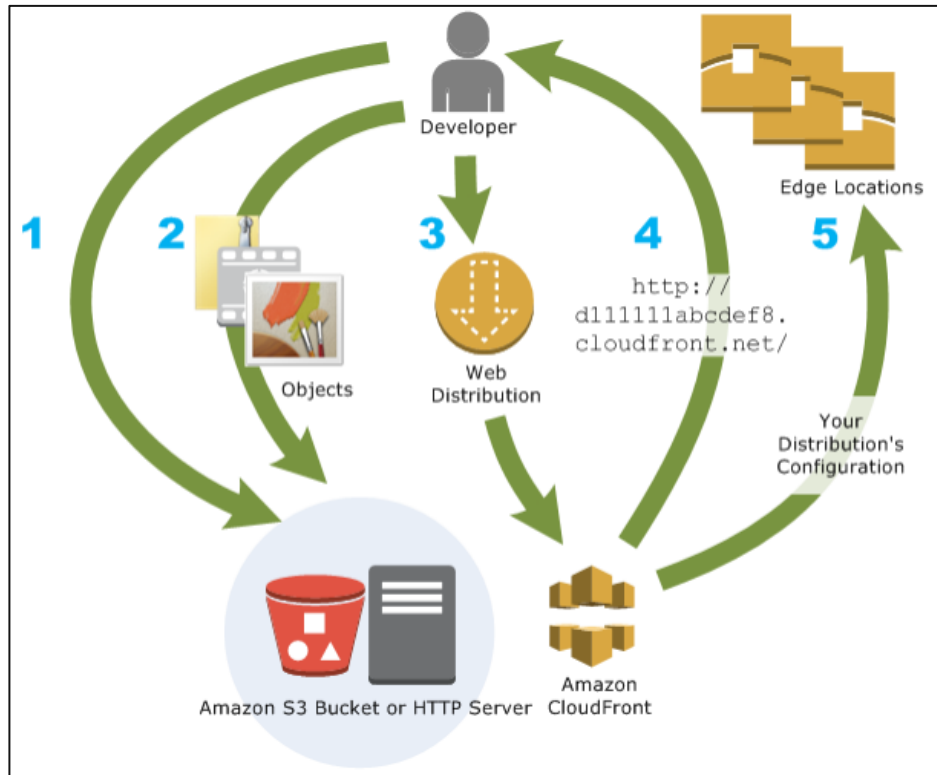
**Figura 34.** Creación de bucket S3.  
**Fuente:** Amazon Simple Storage Service (p. 9)



**Figura 35.** Control de versiones en S3.  
**Fuente:** Amazon Simple Storage Service (p. 17)

## 5. CloudFront

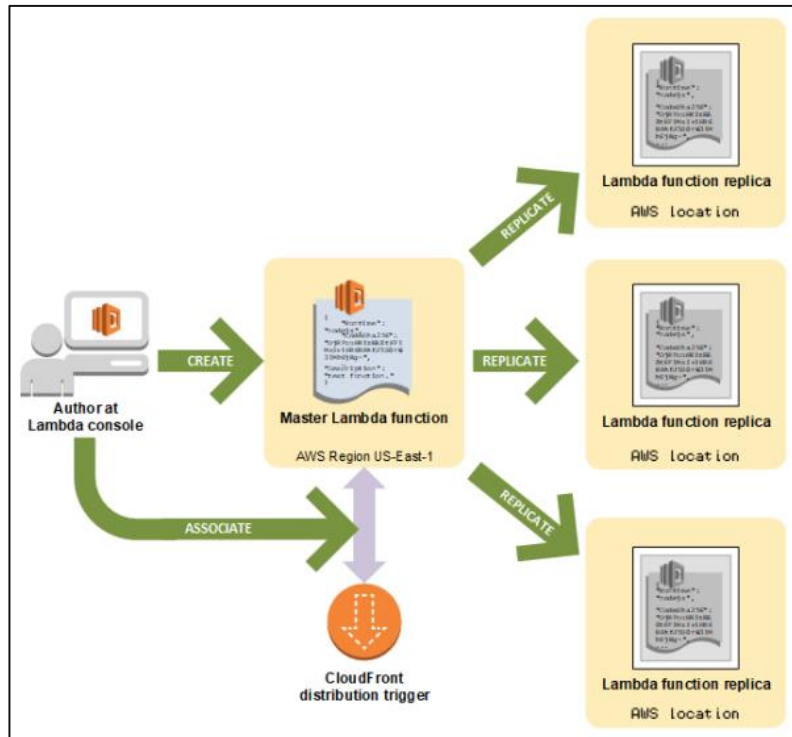
Utiliza ubicaciones de borde, que es una red mundial de centro de datos, que ofrece el mínimo retraso y asegura la distribución del contenido con el mejor desempeño factible.



**Figura 36.** Flujo de Configuración CloudFront.

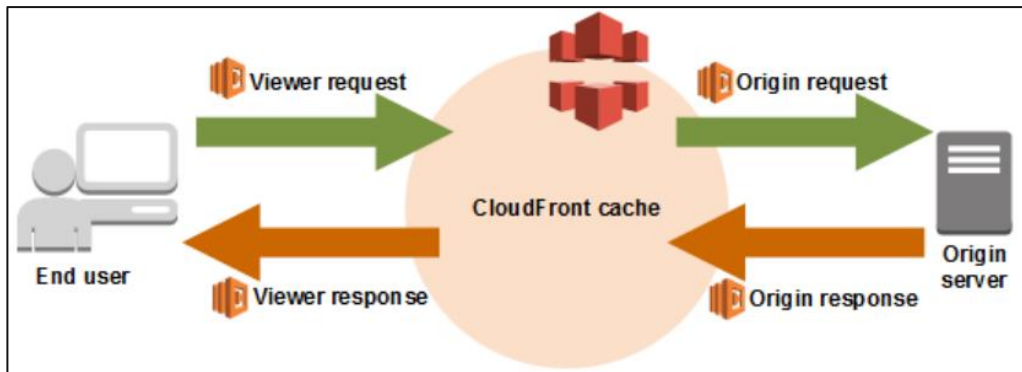
**Fuente:** Amazon Cloud Front (p. 11)

1. Especificación de los servidores de origen, como el bucket de Amazon S3 desde donde CloudFront accederá a sus archivos que serán distribuidos desde ubicaciones de borde.
2. Carga de archivos en los servidores de origen, dichos archivos se conocen como objetos que por lo general vienen a ser páginas web, archivos, imágenes, etc.
3. Creación de la distribución indicando desde que servidores se obtendrán los archivos solicitados por los usuarios a través de su aplicación o web.
4. Nombre de dominio que es asignado a la nueva distribución que se puede apreciar desde la consola de CloudFront.
5. Envío de la configuración de la distribución a las ubicaciones de borde dispersas geográficamente en donde se almacenan en caché todas las copias de los objetos.



**Figura 37.** Funciones Lambda con CloudFront.

*Fuente:* Amazon Cloud Front (p. 306)



**Figura 38.** Eventos de CloudFront para disparar funciones Lambda.

*Fuente:* Amazon CloudFront (p. 326)

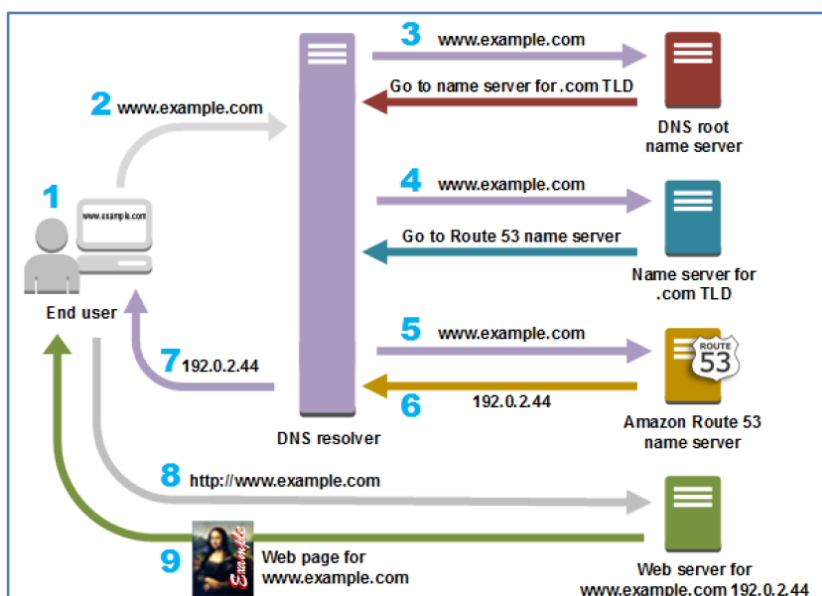
25. Solicitud del Espectador
  26. Solicitud al Origen
  27. Respuesta del Origen
  28. Respuesta al Espectador
6. Route 53

Es un servicio web DNS de alta disponibilidad, que tiene las siguientes funciones



principales:

29. Registro de nombre de dominio
30. Direccionar el tráfico de Internet hacia los recursos del dominio
31. Comprobar el estado del recurso



**Figura 39.** Flujo redirección de tráfico al dominio con Route 53.

**Fuente:** Amazon Route 53 (p. 4)

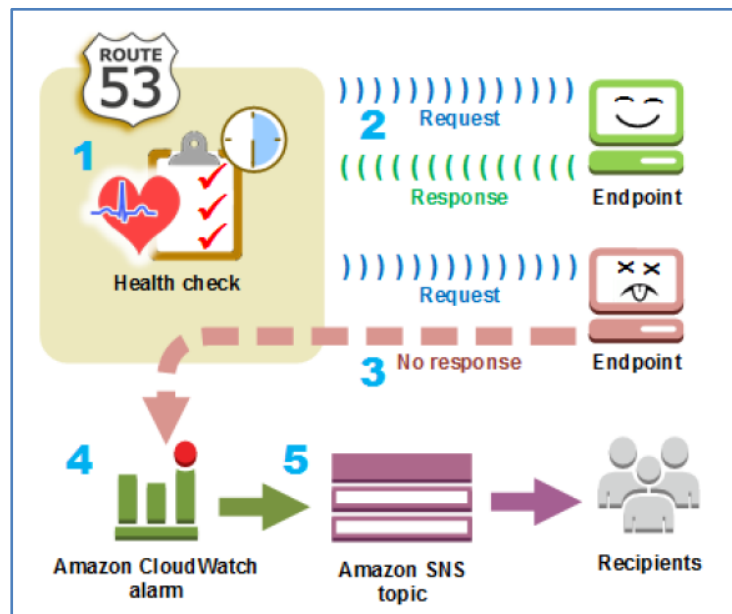
1. Usuario ingresa el nombre del dominio en la web.
2. Solicitud es enviada al servicio de solicitud de nombres DNS que es administrado por el proveedor de Internet.
3. El servicio de solicitud de nombre DNS, reenvía la solicitud al servidor de nombres raíz DNS.
4. Servicio de resolución de nombres reenvía solicitud a uno de los servidores TLD con el nombre del dominio. El servidor con el nombre del dominio responde la solicitud con los nombres de los 4 servidores de nombres de Route 53 que están asociados a los nombres del dominio ingresado.
5. El servicio de resolución de nombres almacena en caché los 4 servidores en Route 53, ocasionando que la próxima vez que alguien navegue en el mismo dominio omita los pasos 3 y 4. El tiempo que los servidores de nombres pueden almacenarse en caché suele ser durante 2 días.
6. Servicio de resolución de nombres DNS elige un servidor de nombres de Route 53

y reenvía la solicitud del dominio al servidor de nombres.

7. Servidor de nombres Route 53 busca la zona hospedada del dominio, al obtener el valor asociado como la dirección IP, devuelve dicha información al servicio de resolución de nombres DNS.

Al tener el servicio DNS, la dirección IP éste devuelve el valor al navegador.

8. El navegador envía la solicitud del dominio a la IP que se obtuvo por el servicio de resolución de nombre DNS.
9. El navegador web u otro recurso en la dirección IP devuelve la página web ingresada al navegador y éste muestra la página.



**Figura 40.** Comprobación del estado del recurso con Route 53.

**Fuente:** Amazon Route 53 (p. 6)

## 7. Certificate Manager

AWS Certificate Manager (ACM), facilita la tarea de crear y administrar los certificados SSL/TLS públicos para sus aplicaciones.

Métrica	Descripción
<code>CRLGenerated</code>	Se crea una lista de revocación de certificados (CRL). Esta métrica solo se aplica a una CA privada.
<code>MisconfiguredCRLBucket</code>	El bucket de S3 especificado para la CRL no está configurado correctamente. Compruebe la política del bucket. Esta métrica solo se aplica a una CA privada.
<code>Time</code>	Hora a la que se emitió el certificado. Esta métrica se aplica únicamente a la operación <a href="#">IssueCertificate</a> .
<code>Success</code>	Especifica si un certificado se ha emitido correctamente. Esta métrica se aplica únicamente a la operación <code>IssueCertificate</code> .
<code>Failure</code>	Indica que se ha producido un error en una operación. Esta métrica se aplica únicamente a la operación <code>IssueCertificate</code> .

**Figura 41.** Métricas CloudWatch admitidas por ACM PCA.  
**Fuente:** AWS Certificate Manager Private Certificate Authority (p. 16)

## 8. React.js

Es una librería de JavaScript que nos ayuda a desarrollar componentes de forma sencilla y modular haciendo uso de los últimos estándares. Es una herramienta idónea para el desarrollo de interfaces de usuario y aplicaciones Frontend.

```
npx create-react-app my-app
cd my-app
npm start
```

```

// To Do app example
import React from 'react';

class Todos extends React.Component {
  state = { team: "React", todos: [] };

  render() {
    let todoItems = this.state.todos.map(({todoId, text}) => {
      return <li key={todoId}>{text}</li>;
    });

    return (
      <div style={styles}>
        <h1>{this.state.team} Todos</h1>
        <ul>
          {todoItems}
        </ul>

        <form>
          <input ref="newTodo" type="text" placeholder="What do you want to do?" />
          <input type="submit" value="Save" />
        </form>
      </div>
    );
  }
}

let styles = {
  margin: "0 auto",
  width: "25%"
};

export default Todos;

```

*Figura 42.* Líneas de código 1 – Configuración React.  
*Fuente:* Serveless (p. 16)

## 9. React Router

Create React configura muchas cosas de forma predeterminada pero como estamos creando una aplicación de una sola página vamos a utilizar React Router para manejarlos por nosotros.

```
$ npm install react-router-dom --save
```

Aunque no tenemos ninguna ruta configurada en nuestra aplicación, podemos poner en marcha la estructura básica. Nuestra aplicación actualmente se ejecuta desde el componente APP en src / App.js. Vamos a utilizar este componente como el contenedor de toda nuestra aplicación. Para hacer eso, encapsularemos nuestro componente de aplicación dentro de un enrutador.

Reemplace el siguiente código en src/App.js :

```
ReactDOM.render(<App />, document.getElementById('root'));
```

Con esto:

```
ReactDOM.render(  
  <Router>  
    <App />  
  
  </Router>,  
  document.getElementById("root")  
) ;
```

Importar en el encabezado de src/index.js :

```
import { BrowserRouter as Router } from "react-router-dom";
```

## 10. Bootstrap

Una gran parte de las aplicaciones web de escritura es tener un kit de UI para ayudar a crear la interfaz de la aplicación. Vamos a utilizar Bootstrap para nuestra aplicación de toma de notas. Mientras que Bootstrap se puede usar directamente con React; la forma preferida es usarlo con el paquete React-Bootstrap. Esto hace que nuestro marcado sea mucho más sencillo de implementar y entender.

```
$ npm install react-bootstrap@0.32.4 --save
```

```
<link rel="stylesheet"  
  href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min."
```

### **Creación de contenedores:**

Para crear nuestra aplicación, necesitaremos crear algunas páginas para cargar, editar, crear notas. Para ello colocaremos el Chrome exterior de nuestra aplicación en un componente de nivel superior que representarán a las distintas páginas y a los que llamaremos contenedores.

### **Añadir barra de navegación**

Para ello usaremos el componente Navbar React-Bootstrap .

Para empezar, puede eliminar el src / logo.svg que se encuentra allí mediante la aplicación Crear React.

```
$ rm src/logo.svg
```

Eliminar el código del interior de src/App.js con lo siguiente:

```
import React, { Component } from "react";
import { Link } from "react-router-dom";
import { Navbar } from "react-bootstrap";
import "./App.css";

class App extends Component {
  render() {
    return (
      <div className="App container">
        <Navbar fluid collapseOnSelect>
          <Navbar.Header>
            <Navbar.Brand>
              <Link to="/">Scratch</Link>
            </Navbar.Brand>
            <Navbar.Toggle />
          </Navbar.Header>
        </Navbar>
      </div>
    );
  }
}

export default App;
```

Agregar enlaces a la barra de navegación: Esto dirigirá a los usuarios a iniciar sesión o registrarse en nuestra aplicación cuando la visiten por primera vez.

Reemplazar el método `render` con lo siguiente:

```

render() {
  return (
    <div className="App container">
      <Navbar fluid collapseOnSelect>
        <Navbar.Header>
          <Navbar.Brand>
            <Link to="/">Scratch</Link>
          </Navbar.Brand>
          <Navbar.Toggle />
        </Navbar.Header>
        <Navbar.Collapse>
          <Nav pullRight>
            <NavItem href="/signup">Signup</NavItem>
            <NavItem href="/login">Login</NavItem>
          </Nav>
        </Navbar.Collapse>
      </Navbar>
      <Routes />
    </div>
  );
}

```

Reemplace react-router-dom and react-bootstrap en src/App.js

```

import { Link } from "react-router-dom";
import { Nav, Navbar, NavItem } from "react-bootstrap";

```

## AWS Amplify

Es una biblioteca JavaScript de código abierto para aplicaciones web diseñadas para trabajar desde la nube, brindando componentes de UI personalizables para trabajar con otros servicios de AWS.

Usaremos esta biblioteca para permitir que nuestra aplicación React, se comunique con todos los recursos AWS que se han creado y conectarnos sin problemas a nuestro backend.

```
$ npm install aws-amplify --save
```

Esto agrega el paquete npm y añade las dependencias al paquete package.json

**Crear configuración:**

Con la referencia de todos los recursos creados anteriormente. Archivo src/config.js

```
export default {
  s3: {
    REGION: "YOUR_S3_UPLOADS_BUCKET_REGION",
    BUCKET: "YOUR_S3_UPLOADS_BUCKET_NAME"
  },
  apiGateway: {
    REGION: "YOUR_API_GATEWAY_REGION",
    URL: "YOUR_API_GATEWAY_URL"
  },
  cognito: {
    REGION: "YOUR_COGNITO_REGION",
    USER_POOL_ID: "YOUR_COGNITO_USER_POOL_ID",
    APP_CLIENT_ID: "YOUR_COGNITO_APP_CLIENT_ID",
    IDENTITY_POOL_ID: "YOUR_IDENTITY_POOL_ID"
  }
};
```

Importar agregando el siguiente encabezado en src/index.js

```
import Amplify from "aws-amplify";
```

Importar la configuración creada arriba, agregar lo siguiente en la cabecera de src/index.js



```
import config from "./config";
```

```
Amplify.configure({
  Auth: {
    mandatorySignIn: true,
    region: config.cognito.REGION,
    userPoolId: config.cognito.USER_POOL_ID,
    identityPoolId: config.cognito.IDENTITY_POOL_ID,
    userPoolWebClientId: config.cognito.APP_CLIENT_ID
  },
  Storage: {
    region: config.s3.REGION,
    bucket: config.s3.BUCKET,
    identityPoolId: config.cognito.IDENTITY_POOL_ID
  },
  API: {
    endpoints: [
      {
        name: "notes",
        endpoint: config.apiGateway.URL,
        region: config.apiGateway.REGION
      }
    ]
  }
});
```

Creación de página de autenticación – crear un nuevo archivo src/containers/Login.js

```
import React, { Component } from "react";
import { Button, FormGroup, FormControl, ControlLabel } from "react-
bootstrap";
import "./Login.css";

export default class Login extends Component {
  constructor(props) {
    super(props);

    this.state = {
      email: "",
      password: ""
    };
  }

  validateForm() {
    return this.state.email.length > 0 && this.state.password.length >
0;
  }

  handleChange = event => {
```

```

    this.setState({
      [event.target.id]: event.target.value
    });
  }

  handleSubmit = event => {
    event.preventDefault();
  }

  render() {
    return (
      <div className="Login">
        <form onSubmit={this.handleSubmit}>
          <FormGroup controlId="email" bsSize="large">
            <ControlLabel>Email</ControlLabel>
            <FormControl
              autoFocus
              type="email"
              value={this.state.email}
              onChange={this.handleChange}
            />
          </FormGroup>
          <FormGroup controlId="password" bsSize="large">
            <ControlLabel>Password</ControlLabel>
            <FormControl
              value={this.state.password}
              onChange={this.handleChange}
              type="password"
            />
          </FormGroup>
          <Button
            block
            bsSize="large"
            disabled={!this.validateForm()}
            type="submit"
          >
            Login
          </Button>
        </form>
      </div>
    );
  }
}

```

## Autenticación con AWS Cognito

Utilizaremos AWS Amplify para el inicio de sesión de Amazon Cognito, para ello

Simplemente reemplace nuestro método handleSubmit de marcador de posición en src / containers / Login.js con lo siguiente:

```
handleSubmit = async event => {
  event.preventDefault();

  try {
    await Auth.signIn(this.state.email, this.state.password);
    alert("Logged in");
  } catch (e) {
    alert(e.message);
  }
}
```

Almacenando estado de inicio de sesión en la aplicación, en src/App.js :

```
constructor(props) {
  super(props);

  this.state = {
    isAuthenticated: false
  };
}

userHasAuthenticated = authenticated => {
  this.setState({ isAuthenticated: authenticated });
}
```

Crear botón de Cierre de sesión:

```
<LinkContainer to="/signup">
  <NavItem>Signup</NavItem>
</LinkContainer>
<LinkContainer to="/login">
  <NavItem>Login</NavItem>
</LinkContainer>
```

```

{this.state.isAuthenticated
  ? <NavItem onClick={this.handleLogout}>Logout</NavItem>
  : <Fragment>
    <LinkContainer to="/signup">
      <NavItem>Signup</NavItem>
    </LinkContainer>
    <LinkContainer to="/login">
      <NavItem>Login</NavItem>
    </LinkContainer>
  </Fragment>
}

```

Formulario de Registro básico:

```

import React, { Component } from "react";
import {
  HelpBlock,
  FormGroup,
  FormControl,
  ControlLabel
} from "react-bootstrap";
import LoaderButton from "../components/LoaderButton";
import "./Signup.css";

export default class Signup extends Component {
  constructor(props) {
    super(props);

    this.state = {
      isLoading: false,
      email: "",
      password: "",
      confirmPassword: "",
      confirmationCode: "",
      newUser: null
    };
  }

  validateForm() {
    return (

```

```

    this.state.email.length > 0 &&
    this.state.password.length > 0 &&
    this.state.password === this.state.confirmPassword
  );
}

validateConfirmationForm() {
  return this.state.confirmationCode.length > 0;
}

handleChange = event => {
  this.setState({
    [event.target.id]: event.target.value
  });
}

handleSubmit = async event => {
  event.preventDefault();

  this.setState({ isLoading: true });

  this.setState({ newUser: "test" });

  this.setState({ isLoading: false });
}

handleConfirmationSubmit = async event => {
  event.preventDefault();

  this.setState({ isLoading: true });
}

renderConfirmationForm() {
  return (
    <form onSubmit={this.handleConfirmationSubmit}>
      <FormGroup controlId="confirmationCode" bsSize="large">
        <ControlLabel>Confirmation Code</ControlLabel>
        <FormControl

```

```

        autoFocus
        type="tel"
        value={this.state.confirmationCode}
        onChange={this.handleChange}
    />
    <HelpBlock>Please check your email for the code.</HelpBlock>
</FormGroup>
<LoaderButton
    block
    bsSize="large"
    disabled={!this.validateConfirmationForm()}
    type="submit"
    isLoading={this.state.isLoading}
    text="Verify"
    loadingText="Verifying..."
/>
</form>
);
}

renderForm() {
    return (
        <form onSubmit={this.handleSubmit}>
            <FormGroup controlId="email" bsSize="large">
                <ControlLabel>Email</ControlLabel>
                <FormControl
                    autoFocus
                    type="email"
                    value={this.state.email}
                    onChange={this.handleChange}
                />
            </FormGroup>
            <FormGroup controlId="password" bsSize="large">
                <ControlLabel>Password</ControlLabel>
                <FormControl
                    value={this.state.password}
                    onChange={this.handleChange}
                    type="password"

```

```
    />
  </FormGroup>
  <FormGroup controlId="confirmPassword" bsSize="large">
    <ControlLabel>Confirm Password</ControlLabel>
    <FormControl
      value={this.state.confirmPassword}
      onChange={this.handleChange}
      type="password"
    />
  </FormGroup>
  <LoaderButton
    block
    bsSize="large"
    disabled={!this.validateForm()}
    type="submit"
    isLoading={this.state.isLoading}
    text="Signup"
    loadingText="Signing up..."
  />
</form>
);
}

render() {
  return (
    <div className="signup">
      {this.state.newUser === null
        ? this.renderForm()
        : this.renderConfirmationForm()}
    </div>
  );
}
}
```

## 11. Stripe

Es una plataforma que desarrolla herramienta con alta capacidad y que se acomodan a cualquier tipo de comercio electrónico, eliminando la dificultad.



```
1 // Solicita la biblioteca de Stripe con clave secreta para pruebas.
2 const stripe = require('stripe' 6.31.1 )('sk_test_BQokikJOvBiI2HLWgH4oIfQ2');
3
4 // Crea un pago a partir de un token de tarjeta de prueba.
5 const charge = await stripe.charges.create({
6   amount: 3000,
7   currency: 'usd',
8   source: 'tok_amex',
9   description: 'Mi primer pago'
10 });
11
12 // Haz clic en "▶ run" para probar este código en modo real y crear tu primer pago.
```

Save on RunKit Node 8 help run

Object

- # amount: 3000
- # amount\_refunded: 0
- ∅ application: null
- ∅ application\_fee: null
- ∅ application\_fee\_amount: null
- « balance\_transaction: "txn\_1ETz9U2eZvKYlo2CQ35Ynk0s"
- ▶ ∅ billing\_details: Object {address: Object {city: null, country: null, line1: null, line2: null, postal\_code: null, state: null}, ...}
- TF captured: true
- # created: 1556404668
- « currency: "usd"
- ∅ customer: null
- « description: "Mi primer pago"
- ∅ destination: null
- ∅ dispute: null
- ∅ failure\_code: null
- ∅ failure\_message: null
- ▶ ∅ fraud\_details: Object {}
- « id: "ch\_1ETz9U2eZvKYlo2CkccVtsKN"
- ∅ invoice: null

**Figura 43.** Demo API Pagos.

**Fuente:** <https://stripe.com/>

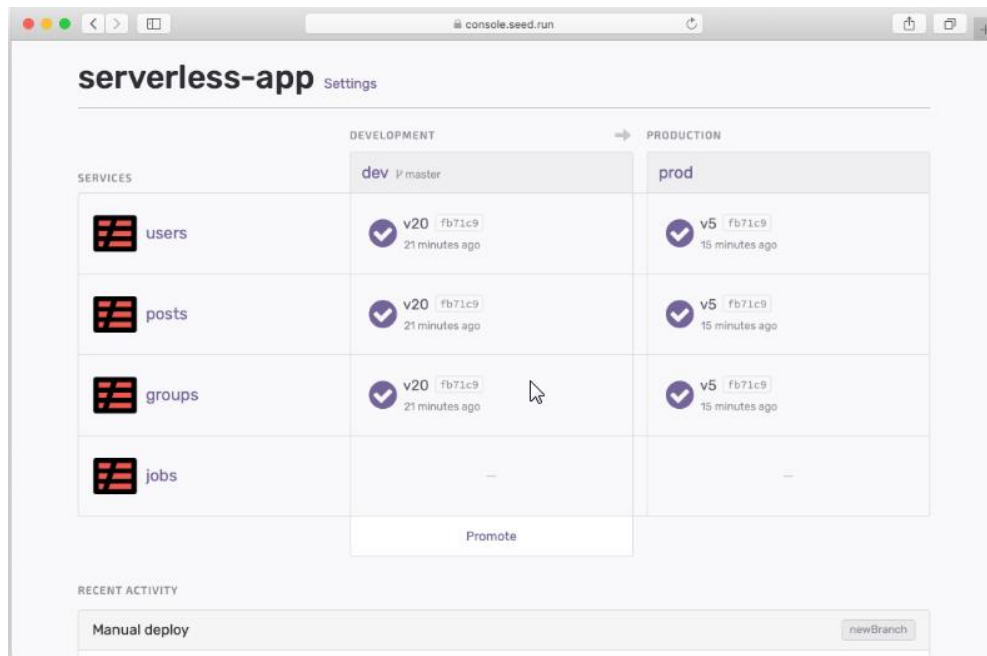
## 12. Seed

Canalización de código hecho especialmente para aplicativos Framework Serverless en AWS. Nos ayuda en la administración de las pipelines, administración de los entornos y el control de las implementaciones para este tipo de proyectos.



**Figura 44.** Estructura configuración raíz Seed.

**Fuente:** <https://seed.run/>



**Figura 45.** Administración de servicios y escenarios Seed.

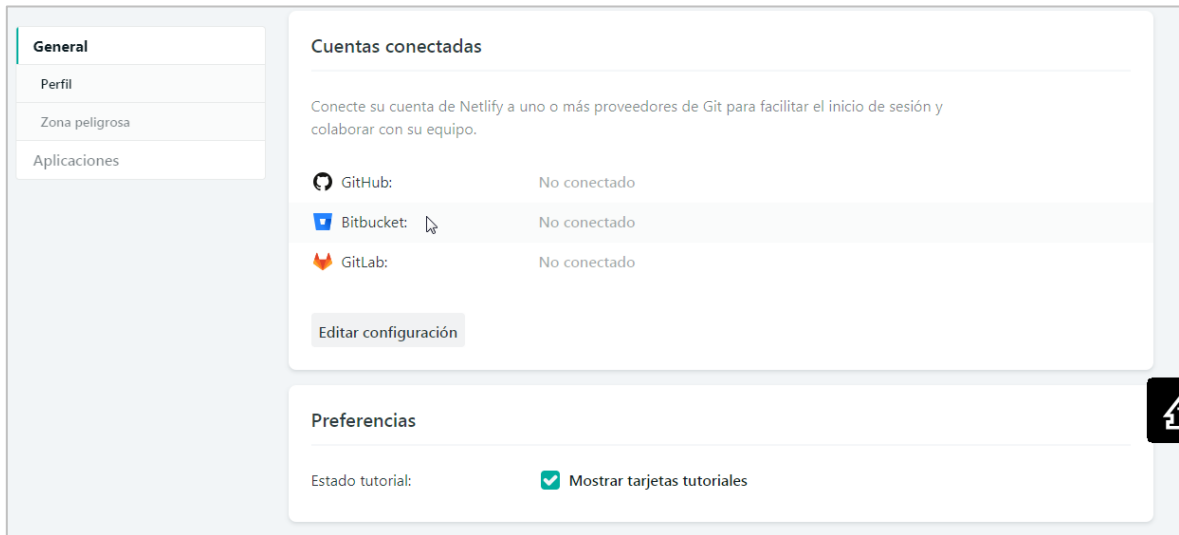
**Fuente:** <https://seed.run/>

### 13. Netlify

Abarca la construcción, implementación a la vez que gestiona proyectos web modernos. Ofrece un marco de trabajo que concerta y lleva a cabo la implementación global, integración continua y HTTPS automático.



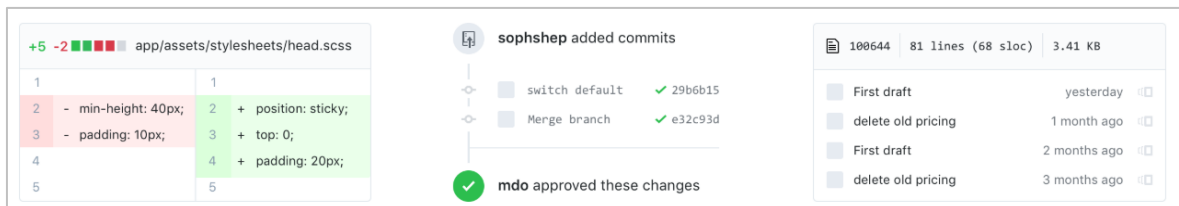
**Figura 46.** Interfaz cuenta Netlify.  
Fuente: <https://netlify.com/>



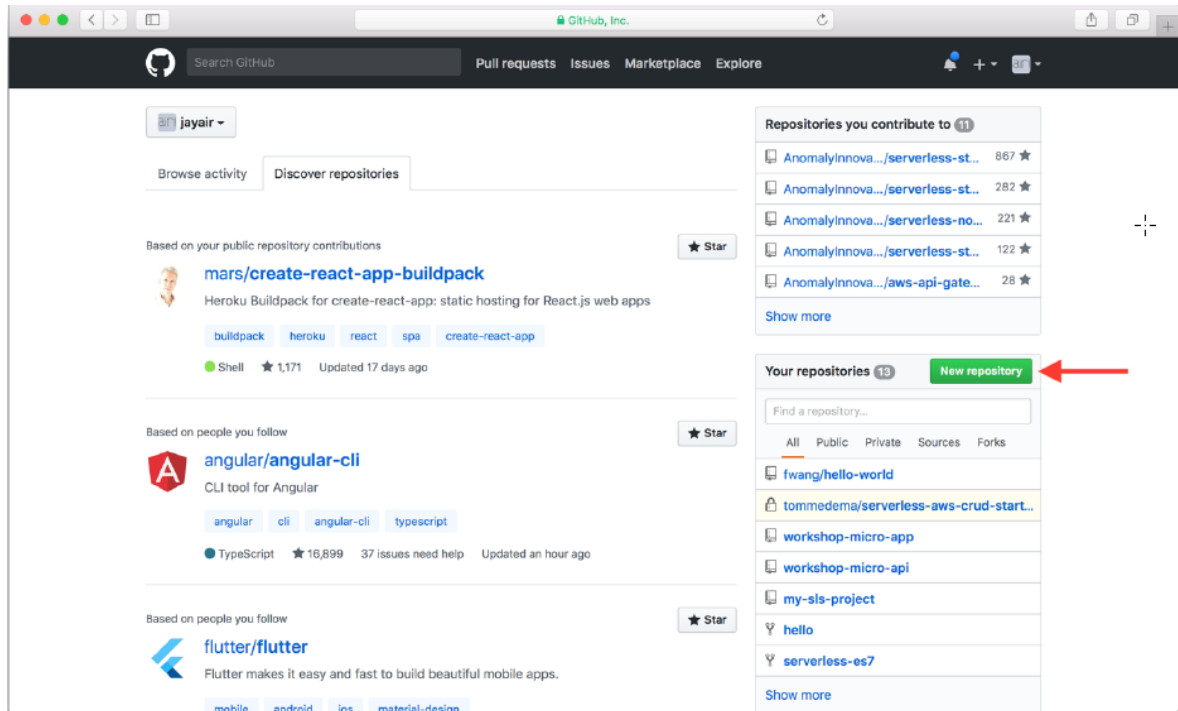
**Figura 47.** Configuración de cuentas asociadas en Netlify.  
Fuente: <https://netlify.com/>

## 14. GitHub

Nos permite alojar código y revisarlo ayudando así la gestión de proyectos y la construcción de software.



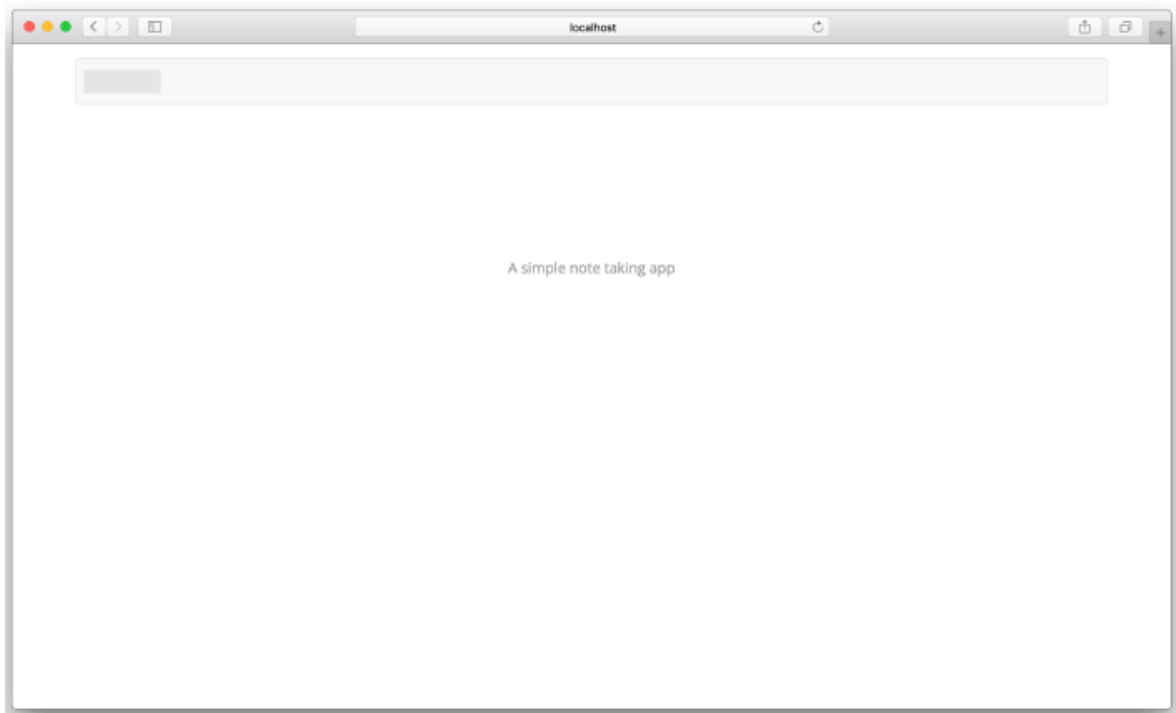
**Figura 48.** Configuración de cuentas asociadas GitHub.  
Fuente: <https://github.com/>



**Figura 49.** Repositorios en Github.

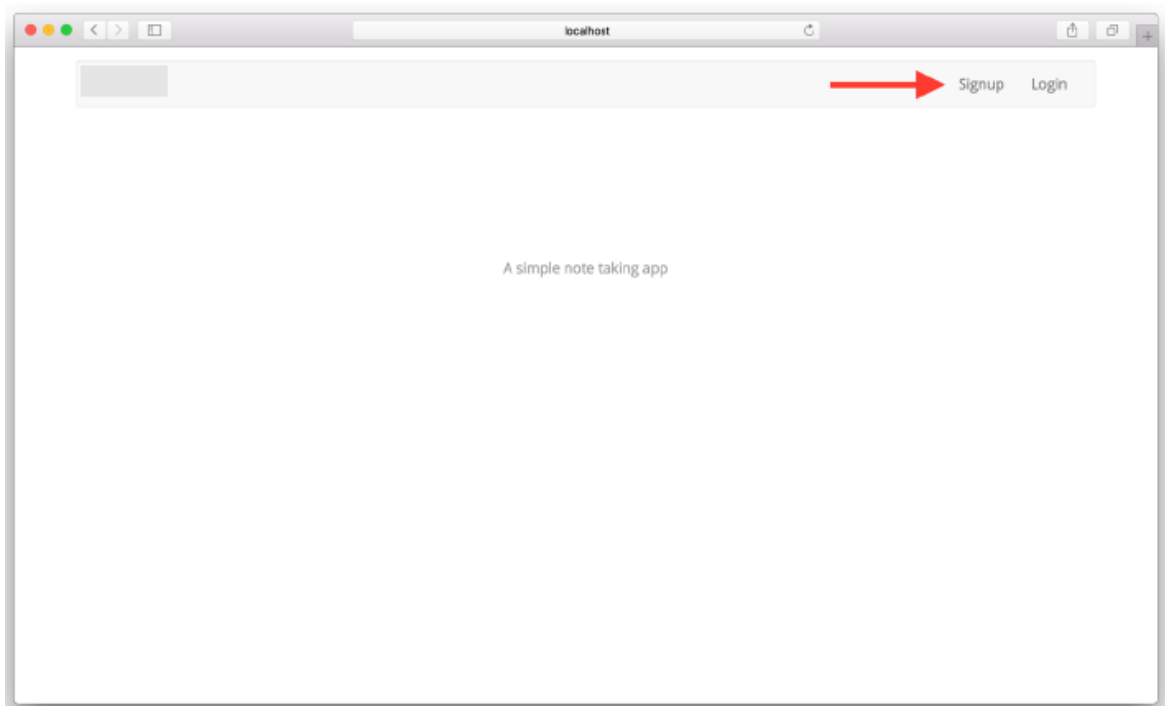
Fuente: <https://github.com/>

## PANTALLAS FUNCIONALES FASE 1



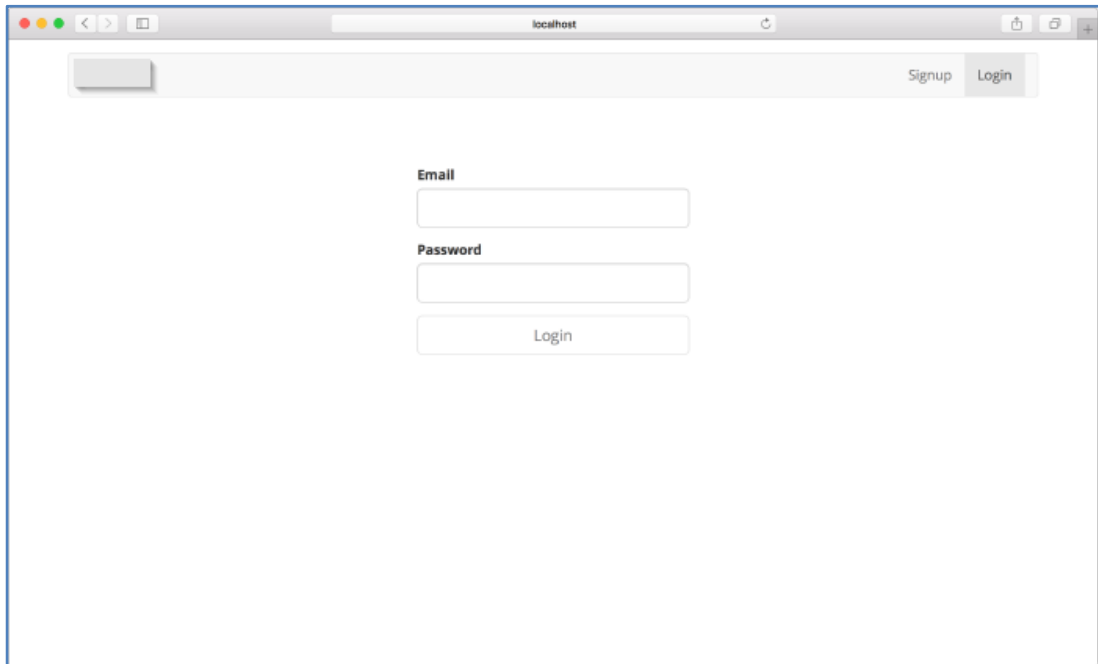
**Figura 50.** Pantalla base de Login

*Fuente:* propia con Serverless

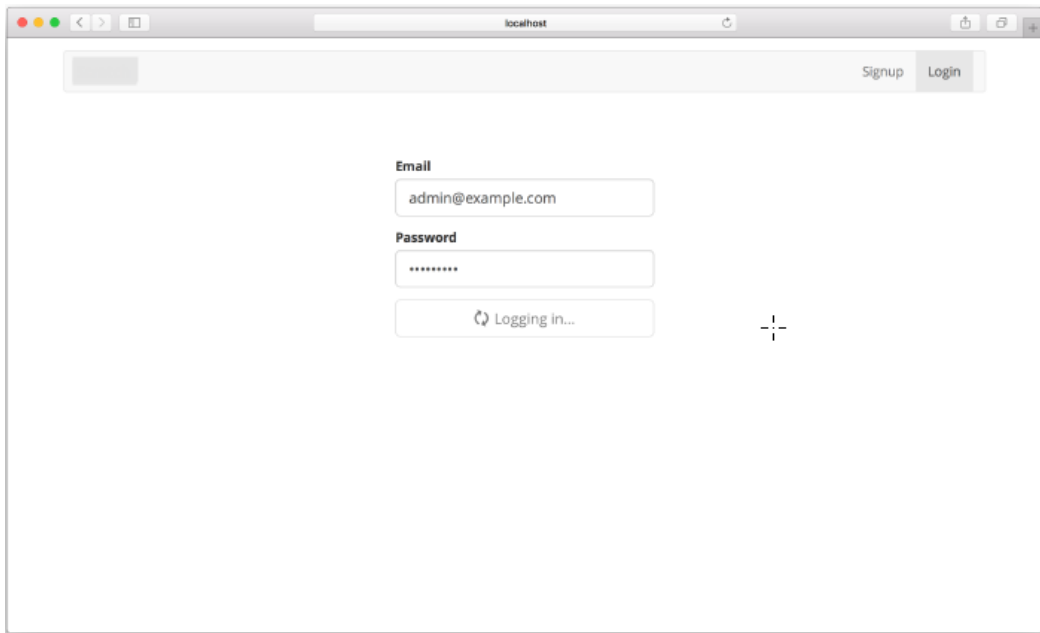


**Figura 51.** Opción de Registro y autenticación.

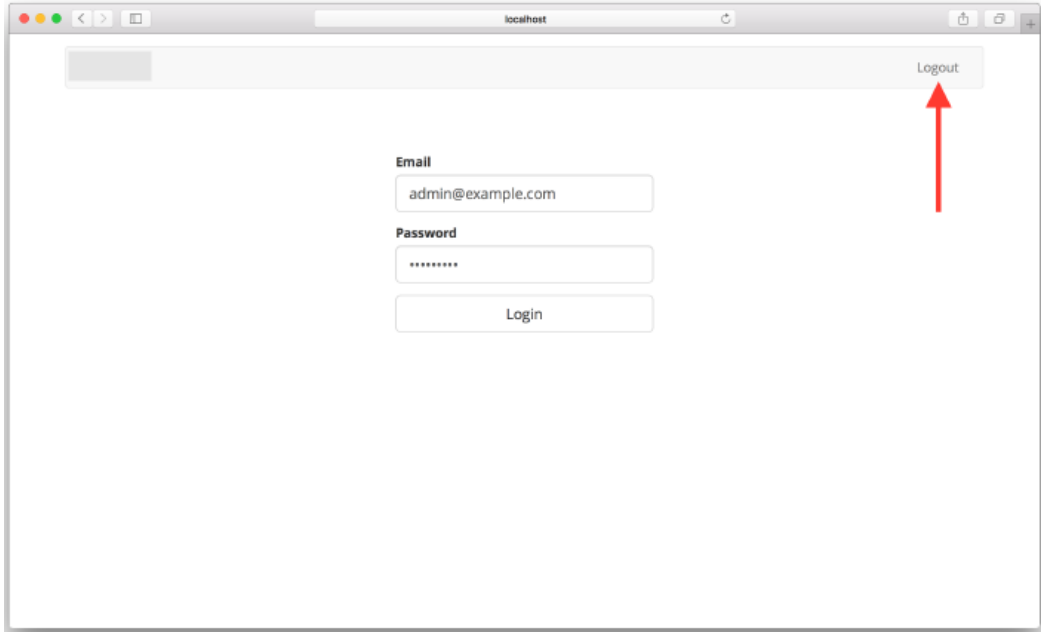
*Fuente:* propia con Serverless



**Figura 52.** Pantalla con campos de Autenticación Login  
*Fuente:* propia con Serveless



**Figura 53.** Validación de Login  
*Fuente:* propia con Serveless



**Figura 54.** Opción de Logout  
*Fuente:* propia con Serveless

## PANTALLAS FUNCIONALES FASE 2

Signup Login

Email  
magally.anton@gmail.com

Password  
.....

Confirm Password  
.....

Signup

**Figura 55.** Pantalla de Registro de usuario.

**Fuente:** propia con Serveless

Signup Login

Confirmation Code

.....

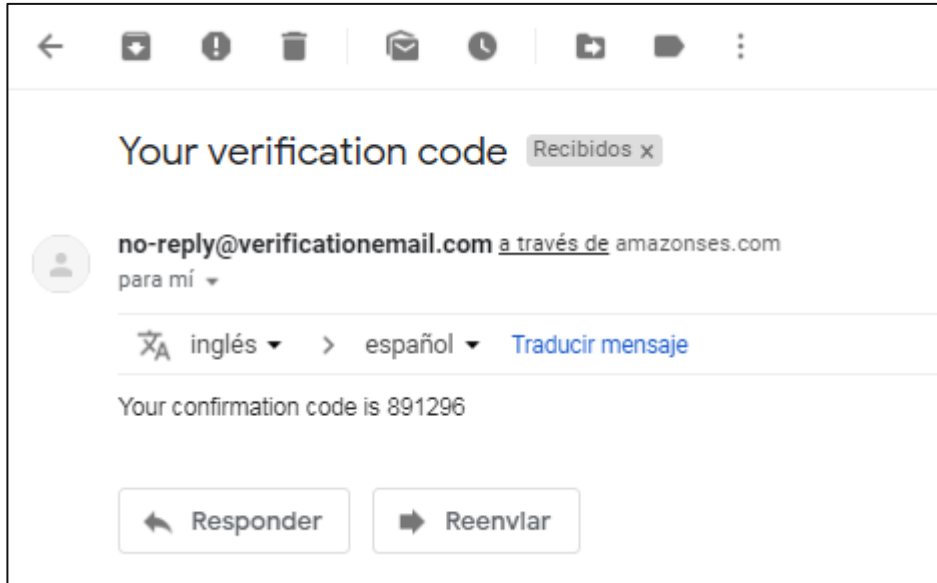
Please check your email for the code.

Verify

**Figura 56.** Pantalla de código de verificación luego del Signup

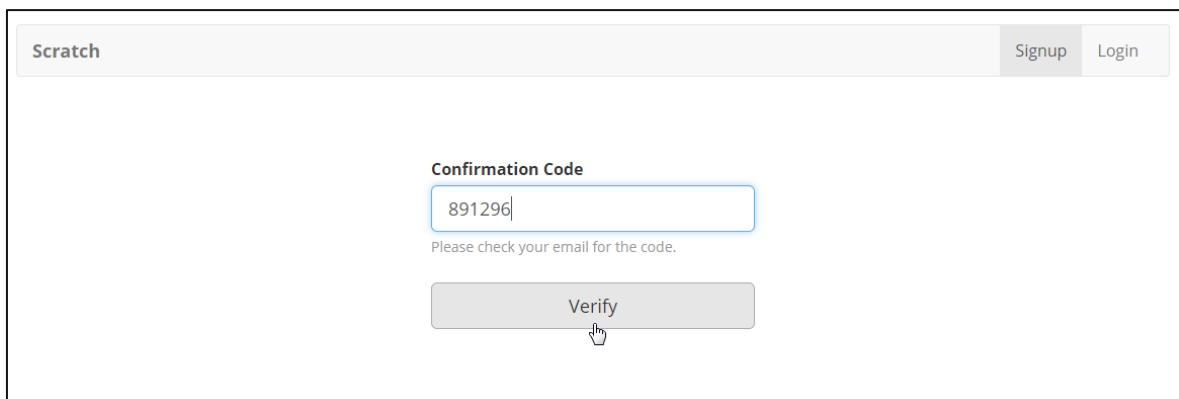
**Fuente:** propia con Serveless





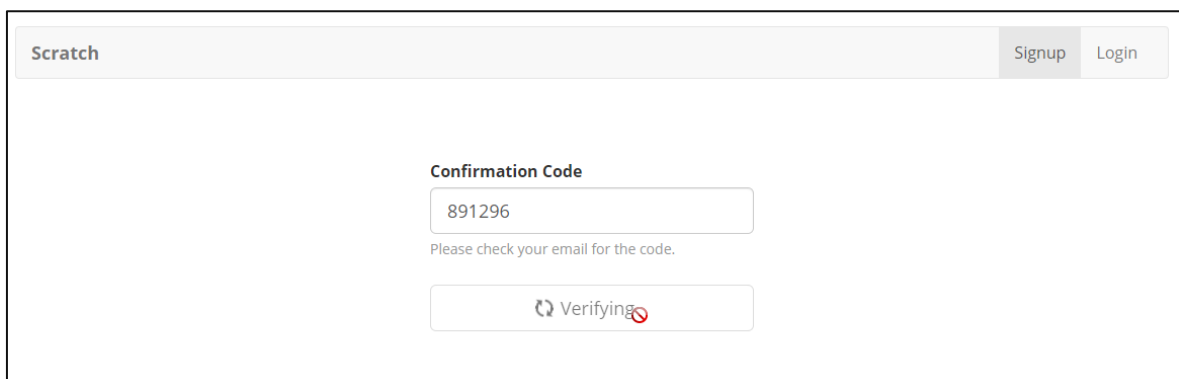
**Figura 57.** Código de verificación recibido

*Fuente:* propia con Serveless



**Figura 58.** Ingreso de código de verificación.

*Fuente:* propia con Serveless



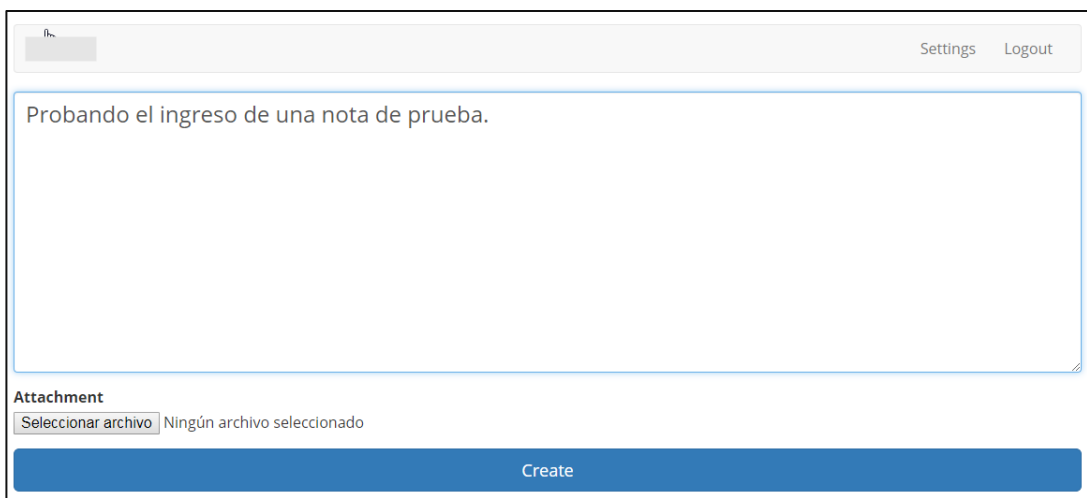
**Figura 59.** Validación del código de verificación.

*Fuente:* propia con Serveless



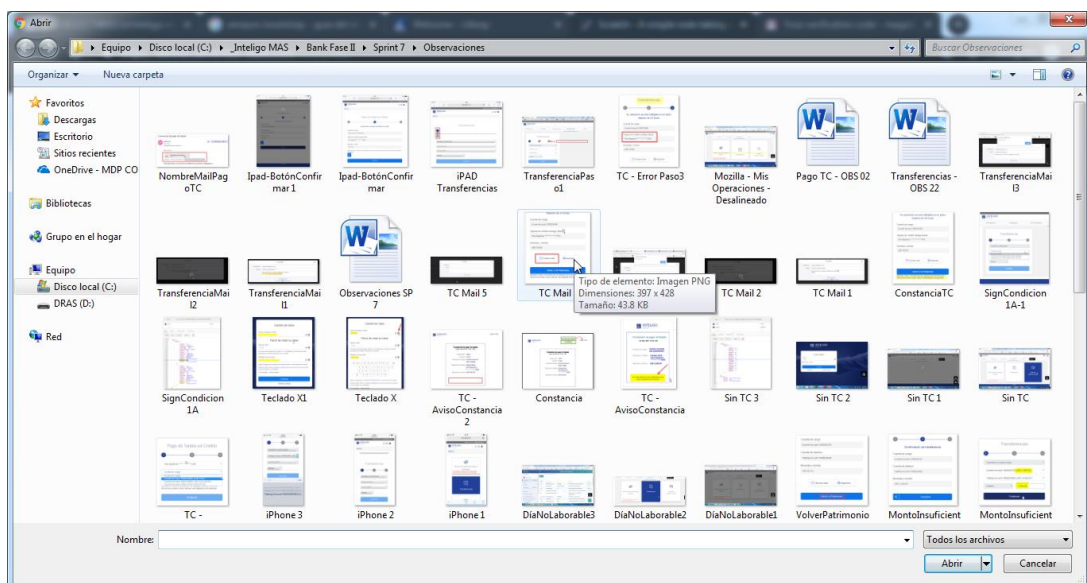
**Figura 60.** Funcionalidad de ingreso de notas de la aplicación.

*Fuente:* propia con Serveless



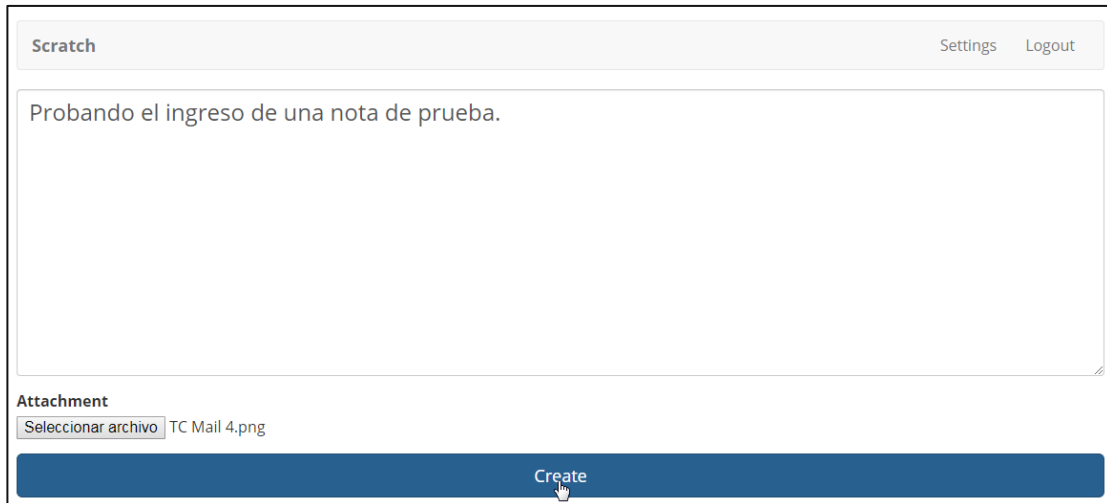
**Figura 61.** Opción de creación de nueva nota.

*Fuente:* propia con Serveless



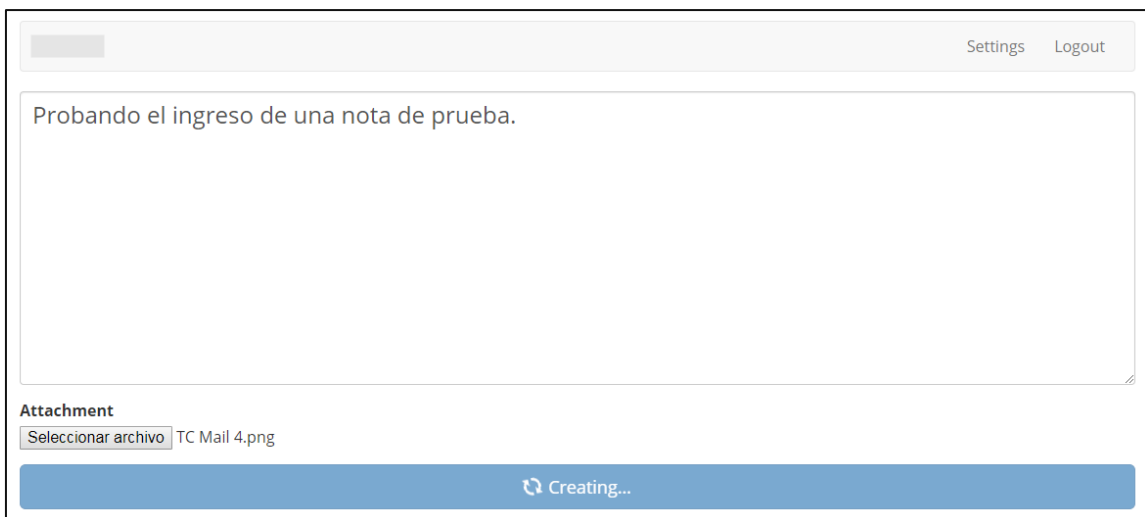
**Figura 62.** Ventana de carga de archivos.

*Fuente:* propia con Serveless



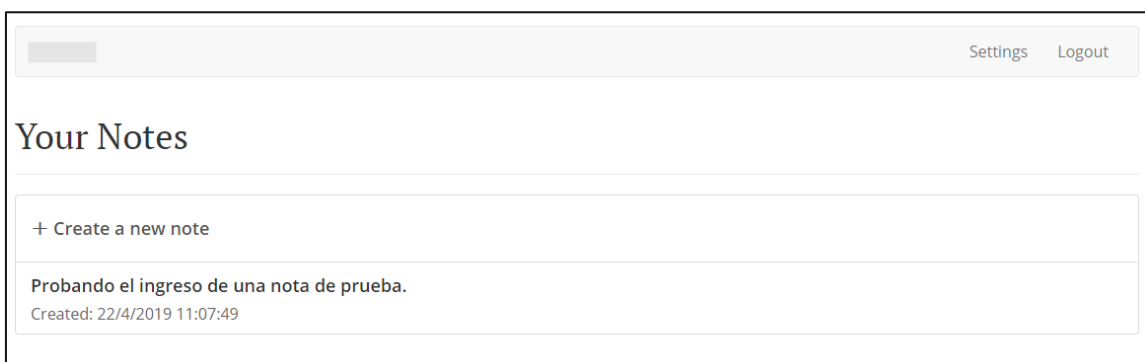
**Figura 63.** Nota con archivo adjunto.

*Fuente:* propia con Serveless



**Figura 64.** Procesamiento de creación de la nota.

*Fuente:* propia con Serveless



**Figura 65.** Vista de nota creada.

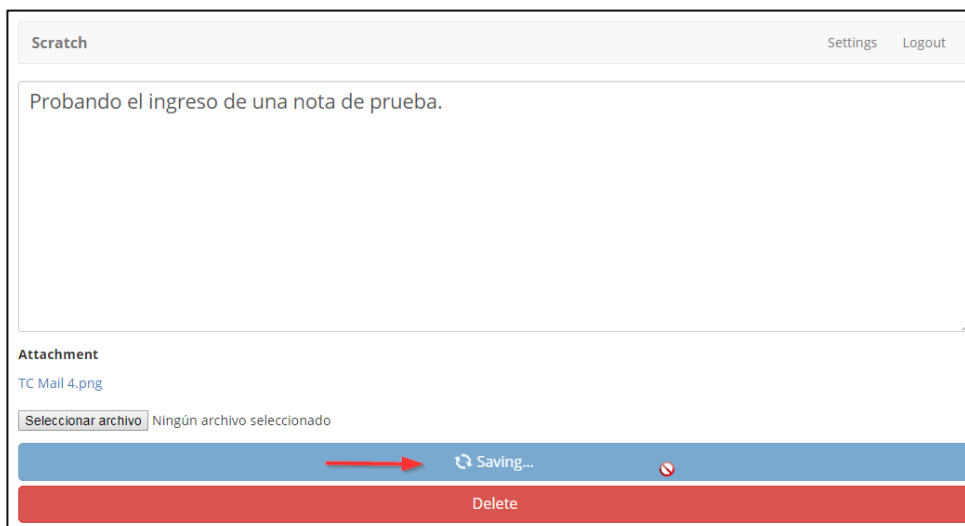
*Fuente:* propia con Serveless



**Figura 66.** Ingreso al detalle de la nota creada.  
**Fuente:** propia con Serveless



**Figura 67.** Opciones del detalle de nota creada.  
**Fuente:** propia con Serveless



**Figura 68.** Procesamiento Guardar nota.  
**Fuente:** propia con Serveless



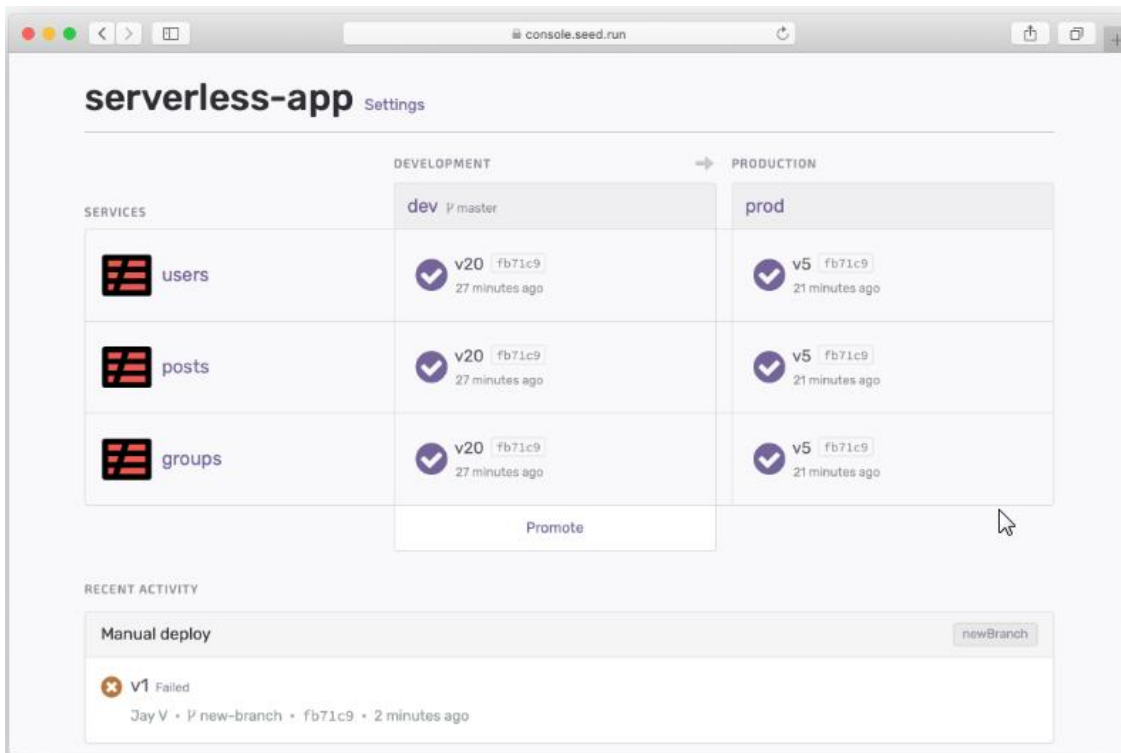
**Figura 69.** Opción eliminar nota.  
*Fuente:* propia con Serveless



**Figura 70.** Confirmación de eliminación de nota.  
*Fuente:* propia con Serveless

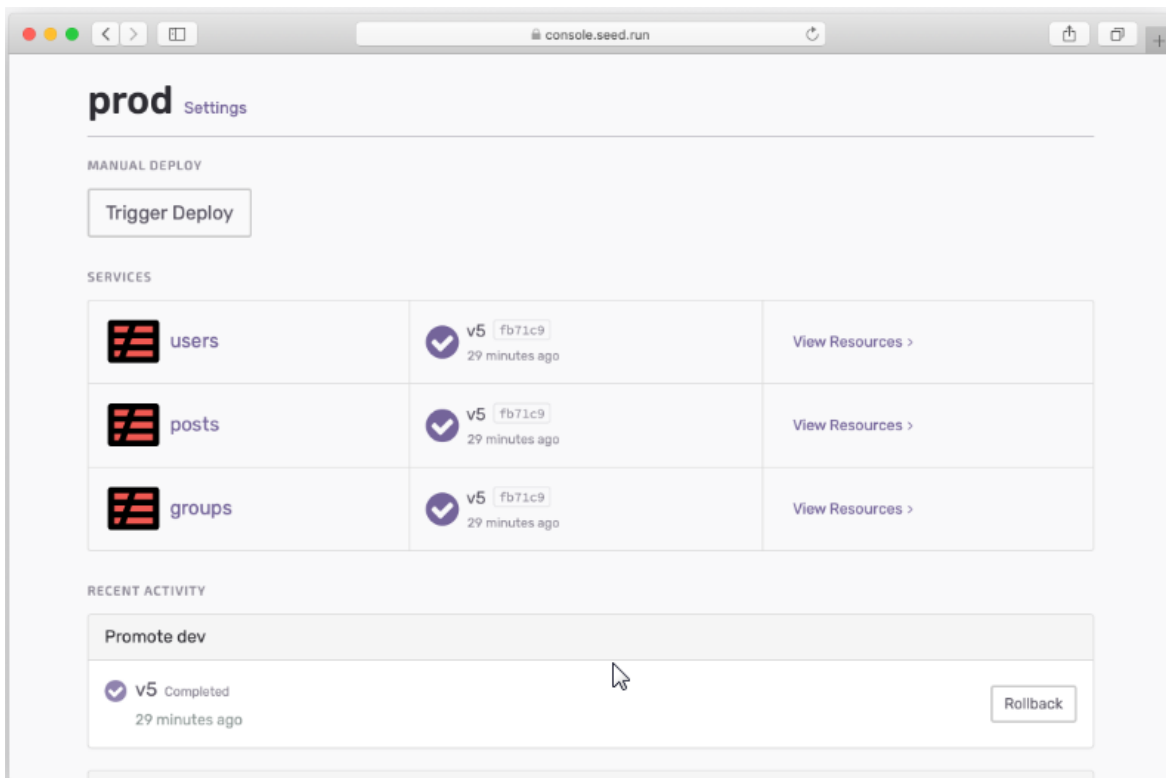


**Figura 71.** Pantalla sin notas.  
*Fuente:* propia con Serveless



**Figura 72.** Escenarios del flujo de trabajo.

*Fuente:* propia con Serveless



**Figura 73.** Servicios contenidos en un Escenario Productivo.

*Fuente:* propia con Serveless