



Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Facultad de Ingeniería de Sistemas e Informática

Escuela Profesional de Ingeniería de Sistemas

**Solución aplicada para la migración de una plataforma
monolítica on premise a una plataforma cloud
utilizando componentes de Amazon Web Services y su
marco teórico Well Architected Framework**

TRABAJO DE SUFICIENCIA PROFESIONAL

Para optar el Título Profesional de Ingeniero de Sistemas

AUTOR

César Junior CHÁVEZ GUTIERREZ

ASESOR

Cesar Alberto MOLINA NEYRA

Lima, Perú

2022



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

Referencia bibliográfica

Chávez, C. (2022). *Solución aplicada para la migración de una plataforma monolítica on premise a una plataforma cloud utilizando componentes de Amazon Web Services y su marco teórico Well Architected Framework*. [Trabajo de suficiencia profesional de pregrado, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Escuela Profesional de Ingeniería de Software]. Repositorio institucional Cybertesis UNMSM.

Metadatos complementarios

Datos de autor	
Nombres y apellidos	CÉSAR JUNIOR CHÁVEZ GUTIERREZ
Tipo de documento de identidad	DNI
Número de documento de identidad	77174525
URL de ORCID	https://orcid.org/0000-0003-4817-6198
Datos de asesor	
Nombres y apellidos	César Alberto Molina Neyra
Tipo de documento de identidad	DNI
Número de documento de identidad	40553679
URL de ORCID	https://orcid.org/0000-0002-8018-7973
Datos del jurado	
Presidente del jurado	
Nombres y apellidos	Santiago Domingo Moquillaza Henríquez
Tipo de documento	DNI
Número de documento de identidad	08280889
Miembro del jurado 1	
Nombres y apellidos	Carlos Edmundo Navarro Depaz
Tipo de documento	DNI
Número de documento de identidad	08482690
Miembro del jurado 2	
Nombres y apellidos	César Alberto Molina Neyra
Tipo de documento	DNI
Número de documento de identidad	40553679
Datos de investigación	
Línea de investigación	No aplica

Grupo de investigación	No aplica
Agencia de financiamiento	Propio
Ubicación geográfica de la investigación	País: Perú Departamento: Lima Provincia: Lima Distrito: Cercado de Lima Jr. Carlos Amezaga No. 375 Universidad Nacional Mayor de San Marcos Latitud: -12.0564232 Longitud: -77.0843327
Año o rango de años en que se realizó la investigación	2021
URL de disciplinas OCDE	2.02.04 -- Ingeniería de sistemas y comunicaciones https://purl.org/pe-repo/ocde/ford#2.02.04



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
Escuela Profesional de Ingeniería de Software

Acta Virtual de Sustentación
del Trabajo de Suficiencia Profesional

Siendo las 16:00 horas del día 15 de enero del año 2022, se reunieron virtualmente los docentes designados como Miembros de Jurado del Trabajo de Suficiencia Profesional, presidido por el Mg. Moquillaza Henríquez Santiago Domingo (Presidente), Dr. Navarro Depaz Carlos Edmundo (Miembro) y el Ing. Molina Neyra Cesar Alberto (Miembro Asesor), usando la plataforma Meet (<https://meet.google.com/jiy-yahj-fza>), para la sustentación virtual del Trabajo de Suficiencia Profesional intitulado: **“SOLUCIÓN APLICADA PARA LA MIGRACIÓN DE UNA PLATAFORMA MONOLÍTICA ON PREMISE A UNA PLATAFORMA CLOUD UTILIZANDO COMPONENTES DE AMAZON WEB SERVICES Y SU MARCO TEÓRICO WELL ARCHITECTED FRAMEWORK”**, por el Bachiller **Chávez Gutierrez César Junior**; para obtener el Título Profesional de Ingeniero de Software.

Acto seguido de la exposición del Trabajo de Suficiencia Profesional, el Presidente invitó al Bachiller a dar las respuestas a las preguntas establecidas por los miembros del Jurado.

El Bachiller en el curso de sus intervenciones demostró pleno dominio del tema, al responder con acierto y fluidez a las observaciones y preguntas formuladas por los señores miembros del Jurado.

Finalmente habiéndose efectuado la calificación correspondiente por los miembros del Jurado, el Bachiller obtuvo la nota de **17 DIECISIETE**.

A continuación el Presidente de Jurados el Mg. Moquillaza Henríquez Santiago Domingo, declara al Bachiller **Ingeniero de Software**.

Siendo las 16:55 horas, se levantó la sesión.

Presidente

Mg. Moquillaza Henríquez Santiago Domingo

Miembro

Dr. Navarro Depaz Carlos Edmundo

Miembro Asesor

Ing. Molina Neyra Cesar Alberto

DEDICATORIA

A toda mi familia y amigos, que me han apoyado durante toda mi vida académica, en especial a mi abuela Francisca, que sin su guía no sería el profesional que aspiro a convertirme, gracias a todos por su incondicional ayuda y consejos.

AGRADECIMIENTOS

A la consultora Applying, por siempre exigirme cada vez más como profesional y por brindarme oportunidades de crecimiento personal.

A mi asesor Cesar Molina Neyra, por su guía, asesoramiento y consejos brindados durante el desarrollo de este informe.

UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA ESCUELA
PROFESIONAL DE INGENIERÍA DE SISTEMAS

**Solución aplicada para la migración de una plataforma monolítica on
premise a una plataforma cloud utilizando componentes de Amazon Web
Services y su marco teórico Well Architected Framework**

Autor: CHÁVEZ GUTIERREZ, César Junior

Asesor: MOLINA NEYRA, César Alberto

Título: Tesis para optar el Título de Ingeniero de Software

Fecha: Febrero del 2022

RESUMEN

El presente informe de suficiencia profesional detalla la implementación de la migración de un sistema en premisa y su base de datos, a una plataforma en la nube, refactorizando sus principales funcionalidades en servicios web privados, considerando que la migración incluía cambiar el código fuente del antiguo sistema de Visual Basic a Java y del cambio de base de datos de una base de datos relacional a la nueva base de datos no relacional. Para la implementación del desarrollo de software se usó de la metodología de la cascada y para el diseño de la nueva arquitectura en la nube se utilizó el marco teórico de Amazon Web Services, conocido como Well Architected Framework, herramienta que sirvió para la correcta selección de servicios Amazon para cada uno de los componentes de la nueva arquitectura tanto de redes, seguridad, base de datos, cómputo. Siendo durante la fase final del proyecto el punto donde se pudo constatar la reducción de costos mensuales de mantenimiento del sistema en un 57,235% y una mejora en el tiempo de respuesta de los tres servicios desarrollados en un 0,9727%, 37,6% y 32.142%.

PALABRAS CLAVES: Servicios web, Amazon Web Services, Metodología de la cascada, SCRUM, sistemas en premisa

NATIONAL UNIVERSITY OF SAN MARCOS
SCHOOL OF SYSTEMS ENGINEERING AND COMPUTER PROFESSIONAL
SCHOOL OF SYSTEMS ENGINEERING

Applied solution for migration from a monolithic on-premise platform to a cloud platform using Amazon Web Services components and its Well Architected Framework

Author: CHÁVEZ GUTIERREZ, César Junior
Advisor: MOLINA NEYRA, César Alberto
Title: Thesis to choose the Software Engineer Degree
Date: February del 2022

ABSTRACT

This professional sufficiency report details the implementation of the migration of an on-premises system and its database, to a cloud platform, refactoring its main functionalities in private web services, considering that the migration included changing the source code of the old system from Visual Basic to Java and the database change from a relational database to the new non-relational database. For the implementation of software development, the waterfall methodology was used and for the design of the new architecture in the cloud, the theoretical framework of Amazon Web Services, known as Well Architected Framework, was used, a tool that served for the correct selection of Amazon services for each of the components of the new architecture, such as networks, security, databases, and computing. Being during the final phase of the project the point where it was possible to verify the reduction of monthly maintenance costs of the system by 57.235% and an improvement in the response time of the three services developed by 0.9727%, 37.6% and 32.142%.

KEY WORDS: Web Services, Amazon Web Services, Waterfall model, on-premises systems

INDICE GENERAL

RESUMEN	vi
ABSTRACT	vii
ÍNDICE DE TABLAS	x
ÍNDICE DE FIGURAS.....	xi
INTRODUCCIÓN	1
CAPÍTULO I TRAYECTORIA PROFESIONAL	3
CAPÍTULO II CONTEXTO EN EL QUE SE DESARROLLO LA EXPERIENCIA	9
2.1. Empresa – actividad que realiza.....	9
2.2. Misión	9
2.3. Visión	9
2.4. Organización de la empresa	10
2.5. Área, cargo y funciones desempeñadas	10
2.6. Experiencia profesional realizada en la organización	11
CAPÍTULO III ACTIVIDADES DESARROLLADAS	13
3.1. Situación problemática	13
3.1.1. Definición del problema.....	14
3.2. Solución.....	15
3.2.1. Objetivos.....	15
3.2.2. Alcance	15
3.2.3. Etapas y metodología	17

3.2.4. Fundamentos utilizados	19
3.2.4.1. Fundamentos utilizados para el desarrollo del proyecto de migración	19
3.2.5. Implementación de las áreas de procesos y buenas practicas.....	22
3.3. Evaluación.....	34
3.3.1. Evaluación Económica	34
3.3.2. Beneficios Obtenidos	35
CAPÍTULO IV REFLEXION CRITICA DE LA EXPERENCIA	39
CAPÍTULO V CONCLUSIONES Y RECOMENDACIONES	41
5.1. Conclusiones	41
5.2. Recomendaciones	42
FUENTES DE INFORMACIÓN.....	43
ANEXOS.....	44
ANEXO 1. Plan de trabajo	46
ANEXO 2. Casos de pruebas funcionales	48
ANEXO 3. Estructura de API's	56

ÍNDICE DE TABLAS

<i>Tabla 1 Formación Académica</i>	<i>3</i>
<i>Tabla 2 Experiencia en RS Techonology SAC.....</i>	<i>3</i>
<i>Tabla 3 Experiencia en Consulting & Management SAC.....</i>	<i>4</i>
<i>Tabla 4 Experiencia en Applying Consulting SAC.....</i>	<i>4</i>
<i>Tabla 5 Certificados.....</i>	<i>8</i>
<i>Tabla 6 Estimación del costo del proyecto.....</i>	<i>35</i>
Tabla 7 Comparación entre los tiempos de respuesta del nuevo y antiguo con 150 usuarios concurrentes por segundo para el servicio de metada..	35
Tabla 8 Comparación entre los tiempos de respuesta del nuevo y antiguo con 70 usuarios concurrentes por segundo para el servicio de fraccionamiento	36
Tabla 9 Comparación entre los tiempos de respuesta del nuevo y antiguo con 70 usuarios concurrentes por segundo para el servicio de cotización	36
Tabla 10 Comparación entre los costos mensuales del antiguo y el nuevo sistema de cotizaciones.....	37
Tabla 11 Detalle de costo por servicio de AWS mensual.....	37

ÍNDICE DE FIGURAS

Figura 1 Organigrama de la Empresa Applying Consulting SAC.	10
Figura 2 Modelo de la cascada.	18
Figura 3 AWS - 5 Pilares del Well Architected Framework.....	21
Figura 4 Diagrama de software para los nuevos servicios de cotizaciones.	25
Figura 5 Diagrama físico de la base de datos para los nuevos servicios de cotizaciones.	26
Figura 6 Diagrama de secuencia para los nuevos servicios de cotizaciones	27
Figura 7 Diagrama de componentes para los nuevos servicios de cotizaciones	28
Figura 8 Matriz de integración Frontend/Backend	29
Figura 9 Diagrama de infraestructuras AWS del nuevo sistema de cotizaciones	32

INTRODUCCIÓN

El presente informe de suficiencia profesional para alcanzar el título profesional de Ingeniero de Software, describe la implementación de una solución de migración de plataformas, desde un sistema monolítico en premisa a un conjunto de servicios web desplegados con una arquitectura en la nube de alta disponibilidad utilizando Amazon Web Services (AWS) y su marco teórico de buenas prácticas de arquitectura para el diseño y operación sistemas en la nube.

La necesidad de migración de la plataforma del sistema surgió debido a los cambios de negocio que la empresa cliente estaba atravesando debido al contexto mundial de la pandemia, el antiguo sistema de cotización en premisa no se ajustaba a los nuevos costos operacionales que requería para su funcionamiento, además que la demanda de uso por otros sistemas externos de la misma empresa era muy limitada al ser un sistema en premisa. Es debido a esta necesidad de migrar el sistema a una plataforma en la nube, siendo AWS la herramienta escogida por el cliente para el despliegue de esta nueva versión de servicios. El presente informe esta subdivido en los siguientes capítulos:

El Capítulo I – Trayectoria profesional: se detalla los conocimientos y experiencias laborales adquiridos póstumos a la obtención del grado de bachiller, así como un cuadro que sirve como línea de tiempo para brindar mayor detalle de las empresas, posiciones laborales, logros y proyectos en las que el autor estuvo comprometido profesionalmente.

El Capítulo II – Contexto en el que se desarrolló la experiencia: se detalla información de la empresa en la que el autor laboró y desarrolló el proyecto expuesto en este informe, información como la visión, misión y organización de la empresa, también se detalla los cargos desarrollados, funciones y experiencias profesionales desarrolladas mientras el autor permanecía en la empresa.

El Capítulo III – Actividades desarrollada: se detalla inicialmente el contexto de la situación problemática seguido de los objetivos general y específicos, el alcance que tuvo el proyecto y la metodología con la que se trabajó. Finalmente se citan los fundamentos utilizados y la implementación de los procesos que conllevo al desarrollo total de todo el proyecto.

El Capítulo IV – Reflexión crítica de la experiencia: el autor detalla una reflexión y autocrítica sobre todo lo desarrollado durante el proyecto, tanto decisiones positivas como negativas que se tuvieron que tomar para sobre llevar impases durante las fases de desarrollo del proyecto.

Finalmente, el capítulo V conclusiones y recomendaciones, el autor detalla una lista de conclusiones que señalan los resultados finales del proyecto desarrollado y de igual manera el autor elaboró una lista de recomendaciones que señalan pautas a seguir para una futura ampliación o continuación del proyecto desarrollado y puntos que tener en consideración para desarrollos de proyectos de similar naturaleza.

CAPÍTULO I TRAYECTORIA PROFESIONAL

Profesional con el grado de bachiller en Ingeniería de Software de la Universidad Nacional Mayor de San Marcos (UNMSM), con 3 años de experiencia en la construcción y diseño de modelos de arquitecturas para sistemas ERP's en plataformas cloud, especializado en el uso de Amazon Web Services (AWS). Experiencia en los puestos como líder de equipo de desarrollo backend, arquitecto de software para plataformas cloud y como administrador de bases de datos SQL (Oracle, SQL Server, Postgrest y MySql) y NoSql (MongoDB y DynamoDB). Experiencia en la implementación de soluciones integrales para sistemas de recursos empresariales (ERP) así como para la gestión de relaciones de clientes (CRM), diseño de indicadores de negocios (KPI's), reportes empresariales y dashboards.

Certificado con las credenciales de AWS Solutions Architect Associate y AWS Developer Associate, especialista en ejecución de proyectos con metodologías ágiles (SCRUM), gestión y liderazgo (Managment 3.0).

Formación Académica

Tabla 1 Formación Académica

2013-2018 Grado académico de Bachiller en Ingeniería de Software Universidad Nacional Mayor de San Marcos - Facultad de Ingeniería de Sistemas e Informática - Escuela de Ingeniería de Software

Fuente. Elaboración propia

Experiencia Profesional

Tabla 2 Experiencia en RS Techonology SAC

RS Technology

Mar 2018 – Jun 2018	
Puesto	Analista programador
Rol	Técnico programador
Proyecto	PROMART
Funciones	Desarrollo y mantenimiento en los sistemas de impresión de viñetas, despacho, control de stock de las sedes Promart a nivel nacional realizado en .NET MVC4 y Visual Basic.

Nota: Elaboración propia

Tabla 3 Experiencia en Consulting & Management SAC

Consulting & management SAC	
Jun 2018 – Set 2018	
Puesto	Analista programador, administrador y gestor de base de datos
Rol	Líder de equipo de desarrollo backend
Proyecto	AVIO SYSTEM
Funciones	- Liderar el equipo de 4 desarrolladores backend, en la construcción de un nuevo sistema para la empresa de AVIO SA, sistema basado en los procesos de gestión de recursos organizacionales, gestión de marco regulatorio, reportes de servicios brindados a clientes. Sistema desarrollado en .NET - Implementación del modelo de datos en SQL Server 2014 y la administración del mismo.

Fuente. Elaboración propia

Tabla 4 Experiencia en Applying Consulting SAC

Applying consulting SAC	
Set 2018 – Ene 2019	
Puesto	Analista programador fullstack, analista de sistemas BI
Rol	Líder de equipo en el análisis y recolección de información para sistemas de negocios
Proyecto	PMO

Funciones	• Encargado en el desarrollo de nuevos módulos para el proceso de pre matricula online en todas las sedes de Innova School a nivel nacional para el periodo verano 2019-2020. • Elaboración de servicios desplegados en servidores Windows (IIS) para la elaboración de procesos ETL con el fin de generar reportes semanales basados en indicadores de avance en el proceso de negocio del cliente.
------------------	---

Ene 2019 – May 2019

Puesto	Analista programador fullstack, administrador y gestor de base de datos
Rol	Líder de equipo de desarrollo backend
Proyecto	GESTOR DE TURNOS

Funciones	• Encargado en el desarrollo de nuevos módulos para el proceso de registro, programación de horarios y recursos administrativos del personal para la empresa de fondos colectivos Panderó. • Elaboración de servicios web utilizando servicios de computo en la nube como Elastic Computing Services (ECS) para el despliegue de servidores Linux, herramienta de AWS • Desarrollo de módulos generación de reportes basados en indicadores de productividad del personal laboral, utilizando el framework de php, Laravel, y como motor de base de datos PostgreSQL
------------------	--

May 2019 – Ene 2020

Puesto	AWS cloud solution architect, analista programador fullstack
Rol	Arquitecto de infraestructura de sistemas en la nube de AWS
Proyecto	Última Milla
Funciones	• Encargado del diseño de la infraestructura de despliegue en AWS, basado en la necesidad de demanda de uso del sistema

por parte de sus usuarios implementado con los correctos componentes que AWS dispone.

- Coordinador con el equipo de desarrollo para las fechas de despliegue en los ambientes correspondientes de acuerdo a la agenda pactada, así como el monitoreo de incidencias a nivel de infraestructura.
- Desarrollador de servicios backend en Node.js para los módulos administración de recursos de personal para la empresa de logística RANSA.

Ene 2020 – Ago 2020

Puesto AWS cloud solution architect, analista programador fullstack

Rol Arquitecto de infraestructura de sistemas en la nube de AWS

Proyecto DSV COTIZADOR

- Funciones**
- Encargado de la migración del antiguo sistema en premisa a la nueva plataforma en la nube, diseñador de la nueva infraestructura de despliegue en AWS, basado en las nuevas necesidades de demanda en uso del sistema por parte de sus usuarios implementado con los correctos componentes que AWS dispone.
 - Responsable de coordinar con los interesados del negocio para la captura de los nuevos requerimientos en la nueva plataforma y rediseño de los casos de uso del antiguo sistema en premisa, así como con el equipo de desarrollo para las fechas de despliegue en los ambientes correspondientes de acuerdo a la agenda pactada, y también del monitoreo de incidencias a nivel de infraestructura.
 - Desarrollador de servicios backend utilizando Spring como framework de Java para los módulos administración de recursos de personal

Ago 2020 – Mar 2021

Puesto	AWS cloud solution architect, analista programador fullstack
Rol	Arquitecto de infraestructura de sistemas en la nube de AWS
Proyecto	ILLARY
Funciones	• Encargado del diseñado de la infraestructura serverless (sin servidores) de despliegue en AWS, enfocado únicamente en procesos background utilizando principalmente los servicios de AWS como: Lambdas, Simple Queue Service (SQS), Simple Email Services (SES) y DyanamoDB como base de datos. • Responsable de coordinar con los interesados del negocio para la captura de requerimientos en la plataforma y diseño de los casos de uso para los procesos brackground del sistema, así como con el equipo de desarrollo para las fechas de despliegue en los ambientes correspondientes de acuerdo a la agenda pactada, y también del monitoreo de incidencias a nivel de infraestructura. • Desarrollador de servicios backend utilizando el lenguaje de programación Python para las funciones almacenadas en los lambdas de los módulos de procesamiento y actualización de información en las pólizas de seguros adquiridos por los clientes.

Mar 2021 - Actualidad

Puesto	AWS cloud solution architect, analista programador fullstack
Rol	Arquitecto de infraestructura de sistemas en la nube de AWS
Proyecto	RAF
Funciones	• Encargado del diseñado de la infraestructura serverless (sin servidores) de despliegue en AWS, enfocado en procesos background y servicios web (API) utilizando principalmente los servicios de AWS como: Lambdas, Simple Queue Service (SQS), Simple Notification Services (SES), Management Message Broker Service (MQ) y DyanamoDB como base de datos.

- Responsable de coordinar con los interesados del negocio para la captura de requerimientos en la plataforma y diseño de los casos de uso para los procesos background y servicios web del sistema, así como con el equipo de desarrollo para las fechas de despliegue en los ambientes correspondientes de acuerdo a la agenda pactada, y también del monitoreo de incidencias a nivel de infraestructura.
- Desarrollador de servicios backend utilizando el lenguaje de programación Python para las funciones almacenadas en los lambdas de los módulos de gestión y actualización de estados de las pólizas de seguros adquiridos por los clientes.

Fuente. Elaboración propia

Otros Estudios

Tabla 5 Certificados

Certificados	
SCRUM	• Scrum Fundamentals Certified – SCRUMstudy – Julio 26, 2019
	• Scrum Developer Certified – SCRUMstudy – Agosto 19, 2019
Management 3.0	• Management 3.0 foundation workshop – MANAGEMENT 3.0 – Diciembre 2019
Amazon Web Services	• AWS Certified Cloud Practitioner – AWS – Enero 03, 2020
	• AWS Certified Solutions Architect Associate – AWS – Setiembre 21, 2020
	• AWS Certified Developer Associate – AWS – Abril 08, 2021

Fuente. Elaboración propia

CAPÍTULO II CONTEXTO EN EL QUE SE DESARROLLO LA EXPERIENCIA

2.1. Empresa – actividad que realiza

“Applying Consulting es una empresa consultora peruana de tecnología especializada en Migración y Modernización de aplicaciones en AWS, tanto a nivel de Infraestructura y Plataforma como en el Software en sí.” (Applying Consulting, 2020).

Datos de la empresa

- RUC: 20451577175
- Razón Social: APPLYING CONSULTING S.A.C.
- Dirección: Calle Las Orquídeas 444, Piso 7, San Isidro

2.2. Misión

“Nuestra misión es acelerar la adopción de Tecnología AWS de las empresas, de cualquier rubro, de cualquier tamaño y en cualquier momento de su Transformación Digital. Nos inspira ayudar a las empresas comprometidas con la transformación de sus negocios” (Applying Consulting, 2020).

2.3. Visión

“Ser el mejor partner peruano en Migración y Modernización de aplicaciones en AWS y uno de los referentes (TOP 10) de América Latina para el 2024” (Applying Consulting, 2020).

2.4. Organización de la empresa

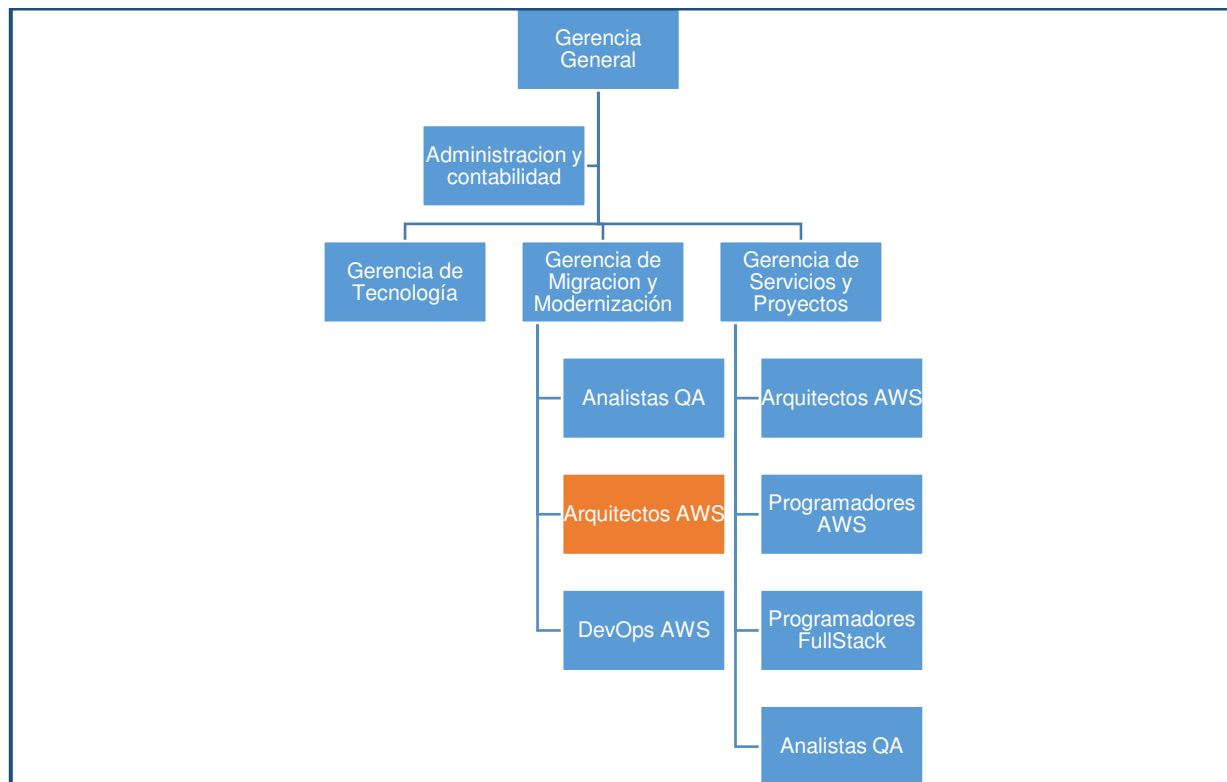


Figura 1 Organigrama de la Empresa Applying Consulting SAC.

Fuente. Elaboración propia

2.5. Área, cargo y funciones desempeñadas

El autor presente de este informe de experiencia profesional se desempeñó como analista programador fullstack y como AWS Cloud Architect en los equipos de trabajo de desarrollo web y arquitectos de software respectivamente.

Las funciones realizadas por el autor durante el periodo que asistió a su empresa laboral fueron las siguientes:

Diseño de infraestructuras de despliegue en AWS para los diversos proyectos solicitados, basado en la necesidad de demanda de uso del sistema por parte de sus usuarios implementado con los correctos componentes que AWS dispone.

Coordinación con el equipo de desarrollo para fechas de despliegue en los diversos ambientes de las cuentas AWS de los clientes.

Monitoreo, control y atención de incidencias a nivel de infraestructura de servicios AWS.

Gestionar de plan de actividades para el equipo de desarrollo de servicios backend.

Coordinaciones con los clientes para la captura de requerimientos, así como la estimación de las mismas y el diseño de planes de trabajos entre los equipos de trabajo involucrados.

Analizar los posibles riesgos que podrían surgir mientras se van desarrollando los proyectos y tomar medidas correctivas para alinear las agendas de las partes involucradas.

2.6. Experiencia profesional realizada en la organización

Durante todo el periodo que el autor se encontró laborando en la empresa Applying Consulting SAC tuvo las siguientes experiencias:

Diseñó la infraestructura en la nube de AWS de múltiples proyectos de clientes bajo el cumplimiento estricto de los requerimientos solicitados.

Participó en el planeamiento de actividades para los equipos de desarrollo de servicios web.

Participó en el equipo de control de incidencias de infraestructura cloud, así como en la elaboración de reportes de errores frecuentes por cada proyecto.

Gestionó y modeló diversas bases de datos relacionales y no relacionales de los diversos proyectos de los clientes.

Elaboró planes de migración para plataformas on premise a plataformas de alta disponibilidad en la nube de AWS.

CAPÍTULO III ACTIVIDADES DESARROLLADAS

3.1. Situación problemática

La necesidad de las empresas por contar con todos sus sistemas integrados siempre está en aumento, y en su mayoría la solución optada es la digitalización de sus sistemas o por la migración de los mismos a la nube. Sin embargo, los procesos para realizar migraciones de sistemas a la nube o de convertir ciertas funcionalidades de sistemas en servicios web que puedan ser utilizados de forma pública o privada es compleja y se deben de tener muchos factores en cuenta, como la concurrencia de usuarios, la resiliencia, la disponibilidad, los costos de los servicios usados, entre otros. Es común que las empresas no hagan un análisis general de lo que implicaría hacer este tipo de migraciones, y se terminaría haciendo migraciones de sus sistemas sin saber si el proyecto es rentable a mediano o largo plazo.

“Un estudio realizado por una Bizagi, empresa tecnológica colombiana, arrojó que el 60% de las compañías en el mundo ve necesario agilizar sus operaciones para ser competitivos. La manera de lograrlo, indicaron, será impulsar la digitalización de sus procesos” (Comercio, 2016).

Complementando la idea de la necesidad de las empresas por ser más competitivas, reducir costos y adaptarse a los nuevos desafíos generados por la pandemia, Ricardo Márquez, presidente de la Sociedad Nacional de Industrias (SNI) comentó para una entrevista para El Comercio, lo siguiente: “El que quiere seguir tiene que invertir en tecnología digital o se quedará en el camino y cerrará”, en el mismo artículo periodístico también se señaló información relacionada brindada por la PNUD, Programa de las Naciones Unidas para el Desarrollo, este indicó que “un 60% de empresas rediseñó sus procesos y un 40% los digitalizó en los últimos meses”. (Artica, 2020)

3.1.1. Definición del problema

El principal problema que radica en el actual sistema de cotizaciones de pólizas de vida es la interconectividad y disponibilidad que tiene con otros sistemas, al ser un sistema monolítico on premise aislado de los otros sistemas con los que cuenta la empresa aseguradora, las tres principales funcionalidades que tiene el sistema de cotizaciones no pueden interactuar con algún servicio externo, es por ello que el cliente requiere que estas funcionalidades del sistema de cotizaciones puedan ser consumido por otras aplicaciones internas de la empresa, como servicios web privados dentro de su misma red. Sin embargo, existen otros problemas que el sistema de cotizaciones acarrea y que deberían de atenderse mientras se soluciona el problema principal mencionado anteriormente, estos problemas específicos son los siguientes:

- Los altos costos de mantenimiento que el sistema de cotizaciones del cliente genera mensualmente, esto debido a que es un sistema desarrollado en Visual Basic desplegado en un servidor Windows por lo que genera un costo por licencia; adicionalmente la base de datos Oracle que también adiciona otro costo también de licencia, dando como resultado un costo superior a los 200 dólares mensuales en mantenimiento del sistema.
- Los tiempos de respuesta para procesar las cotizaciones solicitadas por los usuarios superan los ocho segundos por lo que se busca reducir este tiempo a no más de los cinco segundos.
- La concurrencia de cientos usuarios al mismo tiempo, actualmente el sistema genera inconsistencias en la creación de cotizaciones cuando más de 30 usuarios usan el sistema simultáneamente y el tiempo de respuesta aumenta superando los 30 segundos.

Se demostrará que con la migración final del proyecto a la nueva plataforma cloud de AWS los costos de mantenimiento se reduzcan notoriamente, al menos en un 40%, especialmente con cambio de base de datos que impactará positivamente en el aspecto económico y de igual manera que los nuevos servicios desarrollados

puedan estar disponibles dentro de la red privada de AWS del cliente, así como contar con un mejor performance en tiempos de respuesta y concurrencia de usuarios, para un futuro uso por otros sistemas del cliente que lo requieran.

3.2. Solución

3.2.1. Objetivos

Objetivo General

Implementar la migración de las principales funcionalidades de un sistema on premise a una plataforma cloud usando componentes AWS y su marco teórico Well Architected Framework (WAF).

Objetivos Específicos

- Reducir un 40% los costos de mantenimiento del sistema de cotizaciones mensualmente migrando toda la base de datos Oracle on premise del antiguo sistema de cotizaciones a la nueva base de datos NoSQL, DynamoDB, de Amazon, así como el cambio de código fuente del sistema, de Visual Basic a Java.
- Mejorar un tiempo de respuesta no mayor a los cinco segundos para el servicio de cotizar seguros de vida, en comparación a los ocho segundos de la versión anterior, y de promedio menor a los dos segundos para los otros dos servicios.
- Mejorar el soporte de la concurrencia de hasta 60 usuarios por segundo en la nueva infraestructura de sistemas desplegada.

3.2.2. Alcance

Alcance Funcional

El presente proyecto comprende la migración de servicios on premise a una plataforma cloud usando AWS, la cual establece los siguientes alcances:

Para el código fuente:

- El código fuente de la primera versión del sistema de cotizaciones se migró de Visual Basic a Java usando el framework de Spring.
- Se desarrolló tres servicios que resumieron las principales funcionalidades del antiguo sistema
- Se implementó el Modelo Vista Controlador como patrón de arquitectura de software.

Para la base de datos:

- Se migró únicamente la data de las tablas maestras de la antigua base de datos a la nueva que se desplegó en la nube.
- Se realizó una desnormalización del antiguo modelo de base datos, en Oracle, para el diseño de la nueva base de datos NoSQL desarrollada con DynamoDB

Para la conectividad:

- Las nuevas apis desarrolladas solo debían ser invocadas por otros sistemas o servicios que se encuentran dentro de la misma red privada del cliente de Amazon.

Para la seguridad:

- Se utilizó el servicio Amazon Cognito, servicio de control de accesos de Amazon, para la autenticación de los usuarios que requieran invocar a cualquiera de las nuevas apis desarrolladas.
- Se utilizó el servicio AWS Web Application Firewall, servicio de seguridad de Amazon, para la protección contra ataques DDoS, como inyecciones SQL o scripting.

Para la calidad:

- Se realizó pruebas de humo, de carga y de estrés para garantizar el correcto funcionamiento de las nuevas apis, para un máximo de 4200 usuarios concurrentes por minuto.

3.2.3. Etapas y metodología

3.2.3.1. Fases

Se tienen a continuación las fases de análisis, implementación y ejecución que se utilizaron para realizar la migración de la antigua plataforma on premise del sistema de cotizaciones a una plataforma cloud:

1. Kick Off
2. Análisis y diseño del software
3. Análisis y diseño de la infraestructura AWS
4. Implementación
5. Transición
6. Lanzamiento

En adición, durante la fase de implementación mencionada previamente, se utilizó la metodología SCRUM.

3.2.3.2. Modelo de proceso descriptivo

En mención al estudio realizado por Pressman (2010), manifiesta lo siguiente:

El modelo de la cascada, a veces llamado ciclo de vida clásico, sugiere un enfoque sistemático y secuencial para el desarrollo del software, que comienza con la especificación de los requerimientos por parte del cliente y avanza a través de planeación, modelado, construcción y despliegue, para concluir con el apoyo del software terminado.



Figura 2 Modelo de la cascada.

Fuente. Datos tomados de Pressman (2010)

Para el desarrollo del presente proyecto, el equipo de trabajo del autor utilizó el modelo de la cascada debido a la limitada disponibilidad del cliente para tener reuniones continuas con el equipo de desarrollo, esto además de que para la fase implementación ya se contaba con la mayoría de documentación necesario para llevarla a cabo, siendo esta la fase más iterativa del proyecto. En adición el modelo de la cascada se ajustaba mejor a la metodología clásica con la que el cliente estaba acostumbrado a trabajar y con la que ya habría llevado a cabo proyectos anteriores con la consultora.

Durante el desarrollo del proyecto el equipo de trabajo del autor contó con los siguientes roles:

- Jefe de proyecto: Encargado de la dirección, planificación y monitoreo de las acciones del equipo de trabajo durante cada una de las fases del proyecto.
- Desarrollador backend: Encargado del desarrollo de las actividades de programación de API's, así como de la documentación de la ficha técnica en la sección de los servicios web expuestos, diseño de base de datos, diagramas de componentes, secuencia y despliegue.

- Arquitecto AWS: Encargado del diseño de la infraestructura en la nube de AWS de la nueva versión del sistema de cotizaciones.
- Desarrollador AWS: Encargado del desarrollo de actividades de despliegue de ambientes, pruebas de concepto, y configuraciones de servicios AWS.
- QA Tester: Encargado de diseñar las pruebas de funcionales, así como de llevar a cabo las pruebas de carga, estrés y humo durante la fase implementación en el periodo de pruebas.
- QA Documentador: Encargado de recolectar y dar el formato correcto de documentación a los resultados obtenidos durante el periodo de pruebas.

3.2.4. Fundamentos utilizados

3.2.4.1. Fundamentos utilizados para el desarrollo del proyecto de migración

Modelo de la cascada

“Hay veces en las que los requerimientos para cierto problema se comprenden bien: cuando el trabajo desde la comunicación hasta el despliegue fluye en forma razonablemente lineal. Ocasiones cuando deben hacerse adaptaciones o mejoras bien definidas a un sistema ya existente” (Pressman, 2010).

El modelo la cascada atiende los enfoques de proyectos que tienen claro los requerimientos que se tienen que desarrollar, además de mantener un orden secuencial en el desarrollo de todas sus fases o etapas. Sin embargo, el modelo de cascada tiene sus limitaciones y fallas, estas son algunas de ellas:

1. Es poco común que durante todas las fases de desarrollo del proyecto se sigan flujos secuenciales, pese que el modelo permite repeticiones, estas se realizan

de manera indirecta, por lo que los cambios terminan confundiendo al equipo de trabajo mientras se va avanzando el proyecto.

2. Es reiterativo que el cliente tenga problemas de concentrar todos los requerimientos de forma detallada durante las fases iniciales del proyecto. Para el modelo de cascada es vital que se pueda tener todo claro pues al modelo se le dificulta manejar y controlar la incertidumbre natural que es propia de la mayoría de los proyectos.
3. Paciencia por parte del cliente, el modelo puede presentar un avance del proyecto solamente durante sus fases finales, lo que podría generar incertidumbre prolongada durante cada una de sus fases previas, y aún más crítico sería que se presente un error durante sus primeras presentaciones a puertas de culminar el proyecto.

SCRUM

“El marco del SCRUM se basa en la manera en que trabajan realmente los equipos y les proporciona las herramientas para organizarse y mejorar rápidamente tanto la velocidad como la calidad del trabajo” (Sutherland, 2018).

La metodología de SCRUM también permite a los equipos de trabajo sincerar los avances actuales que se tienen mediante las reuniones diarias del equipo “daily’s”, permitiendo correcciones o alineamientos con los objetivos de trabajo pactados antes de terminar un sprint.

Aunque el proyecto presentado por el autor no trabajó bajo la metodología de SCRUM, hubo ciertas características propias de esta metodología que se sí se ejecutaron, una de ellas fueron los “daily’s”, reuniones diarias, que ayudaron ver el continuo avance del equipo de desarrollo durante toda la fase de implementación, dando un panorama actualizado del estatus de avance. Otra característica adoptada por el equipo fue el de la mejora continua, cuando un miembro del equipo terminaba sus actividades durante el periodo de una semana, otro miembro del equipo la

revisaba y validaba que lo desarrollado cumpla con el requerimiento relacionado, de no cumplirlo a cabalidad se reportaba lo fallido y se corregía.

Well Architected Framework (WAF)

“El Marco de Buena Arquitectura de AWS describe los conceptos claves, principios de diseño y prácticas recomendadas de arquitectura para diseñar y ejecutar cargas de trabajo en la nube” (Amazon Web Services, Inc., 2021).

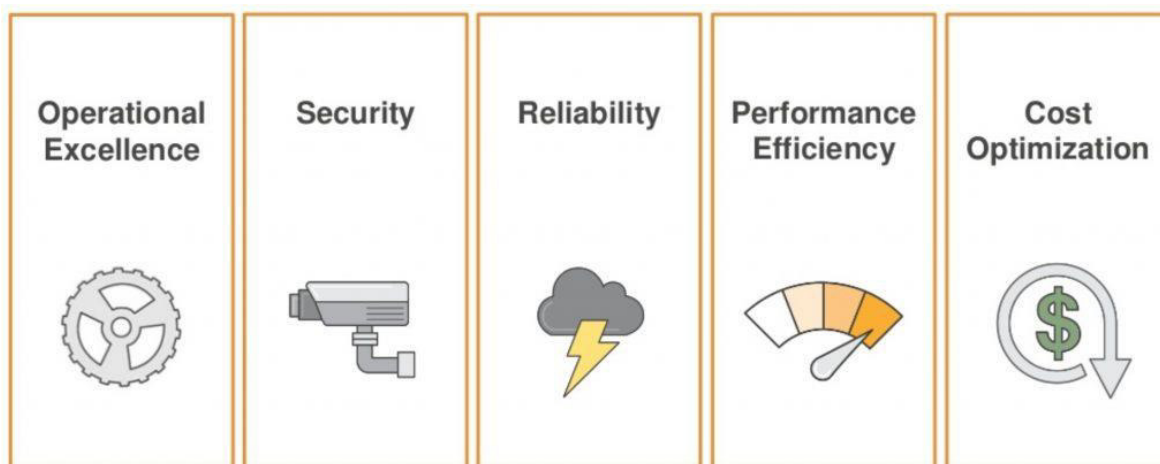


Figura 3 AWS - 5 Pilares del Well Architected Framework.

Fuentes. Datos tomados de Gupta (2020)

El WAF ayuda a los arquitectos a diseñar infraestructuras, de alto rendimiento, eficientes y eficientes; este marco referencial de trabajo diseñado por Amazon se basa en cinco pilares, estos son:

1. **Pilar de excelencia operativa:** Se enfoca en el monitoreo y ejecución de sistemas con la finalidad de generar valor empresarial y optimizar reiterativamente los procesos y procedimientos. Por ejemplo, sus principales temas incluyen la automatización de cambios, respuestas de eventos y la estandarización de operaciones diarias.
2. **Pilar de seguridad:** Se enfoca en la protección de información y en los sistemas. Por ejemplo, sus principales temas incluyen, la confidencialidad,

integridad de datos, identificación y autenticación de usuarios; por último, controles para la identificación de eventos de seguridad.

3. Pilar de fiabilidad: Se enfoca en el aseguramiento en las que las cargas de trabajo cumplan con sus funciones previstas de manera correcta y consistente durante el momento que se espera que lo haga. Por ejemplo, sus principales temas incluyen el diseño de sistemas distribuidos, planificación de recuperación y el cómo para la gestión de cambio.
4. Pilar de eficiencia del rendimiento: Se enfoca en el uso eficaz de los diversos recursos de TI y procesamiento. Por ejemplo, sus principales temas incluyen la selección de tamaños y tipos de recursos adecuados para el cumplimiento de requisitos de carga de trabajo, monitoreo de rendimiento, así como la toma de decisiones para mantener la eficacia a medida que las necesidades de la empresa vayan creciendo.
5. Pilar de optimización de costos: Se enfoca principalmente en evitar los gastos incensarios. Por ejemplo, sus principales temas incluyen el entendimiento y el control del origen de gastos, la selección adecuada de los tipos y cantidades de los recursos, compresión de los gastos a lo largo del tiempo, así como su escalado para la satisfacción de necesidades de la empresa sin caer en los gastos excesivos que podrían llevar su crecimiento.

3.2.5. Implementación de las áreas de procesos y buenas practicas

La ejecución del proyecto de migración a una plataforma en la nube se desarrolló con la participación del autor bajo el rol de AWS Cloud Architect, en las siguientes 6 fases:

3.2.5.1. Fase 0: Kick Off

Durante esta fase, terminado la reunión con la asistencia de los stakeholders, como los jefes de proyectos (cliente y Applying), líder técnico (cliente), líder de

infraestructura (cliente), líder funcional (cliente) y el arquitecto AWS (Applying) se definieron los siguientes puntos:

- Objetivos del proyecto
- Alcance del proyecto
- Hitos del proyecto
- Organigrama del proyecto
- Premisas del proyecto

Como punto adicional se revisó y aprobó el plan de trabajo (Anexo 1) del proyecto, como parte del primer entregable.

3.2.5.2. Fase 1: Análisis y diseño del Software

Durante esta fase, debido a la naturaleza del proyecto se necesitaron los siguientes prerequisites que el cliente debía entregar:

- Entrega de estándares de desarrollo: documento que contiene los estándares de programación con los que los sistemas del cliente tienen que cumplir, estándares como la nomenclatura de variables, funciones, el tamaño de archivos, nomenclatura para las tablas en sus bases de datos, procedimientos almacenados, índices, entre otros.
- Entrega de accesos: documento que contiene las credenciales de los usuarios necesarios para acceder a los recursos para el desarrollo del proyecto como: base de datos, VPN, repositorio Git, consola AWS
- Entrega del ambiente DEV: para empezar el desarrollo del proyecto era necesario que el cliente disponga una cuenta para la consola AWS además de configurar roles, usuarios y permisos necesarios para el equipo de desarrolladores.
- Entrega de línea gráfica: documento que contiene los mockups web y mobile con el que el nuevo sistema contar.

- Entrega del ambiente DEV del sistema antiguo: entrega de las fuentes de código del antiguo sistema de cotizaciones para la migración y el análisis de las funcionalidades principales.

Adicionalmente de los entregables por parte del cliente, el equipo de proyecto llevo a cabo un análisis de requerimientos y construcción de las historias de usuario, así como la transferencia de conocimientos entre los miembros del equipo de desarrollares y arquitectos para el cumplimiento de objetivos en agenda.

Para el cierre de la primera fase del proyecto se tuvieron que presentar los entregables:

- Diseño del software: documento que brinda detalle sobre los diagramas de arquitectura de software, de componentes, de secuencia, físico de la base de datos.
- Diseño de casos de prueba: documento que brinda detalle de los casos de pruebas funcionales identificados para los nuevos servicios del sistema migrado, los casos de pruebas funcionales se encuentran clasificados por cada API (Anexo 2).
- Diseño de API's en swagger: documento que detalla la estructura de los servicios REST, estructura de los request, response, tipo de datos, cabeceras, parámetros de entrada (query param, path param), tipo de protocolo de autenticación (Anexo 3).
- Matriz de integración Fronted/Backend: Documento que detalla la asociación entre los mockups del frontend y los nuevos servicios, clasificados por historia de usuario.

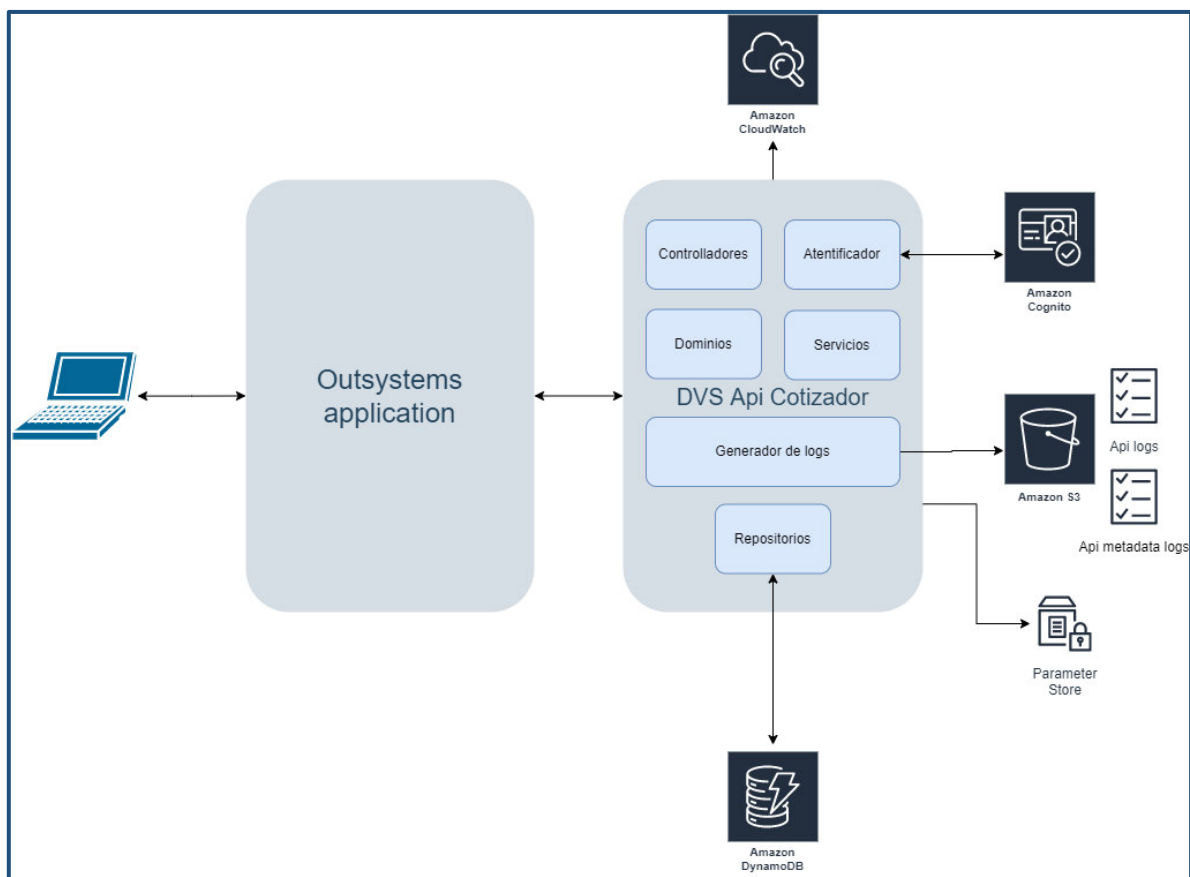


Figura 4 Diagrama de software para los nuevos servicios de cotizaciones.

Fuente. Elaboración propia

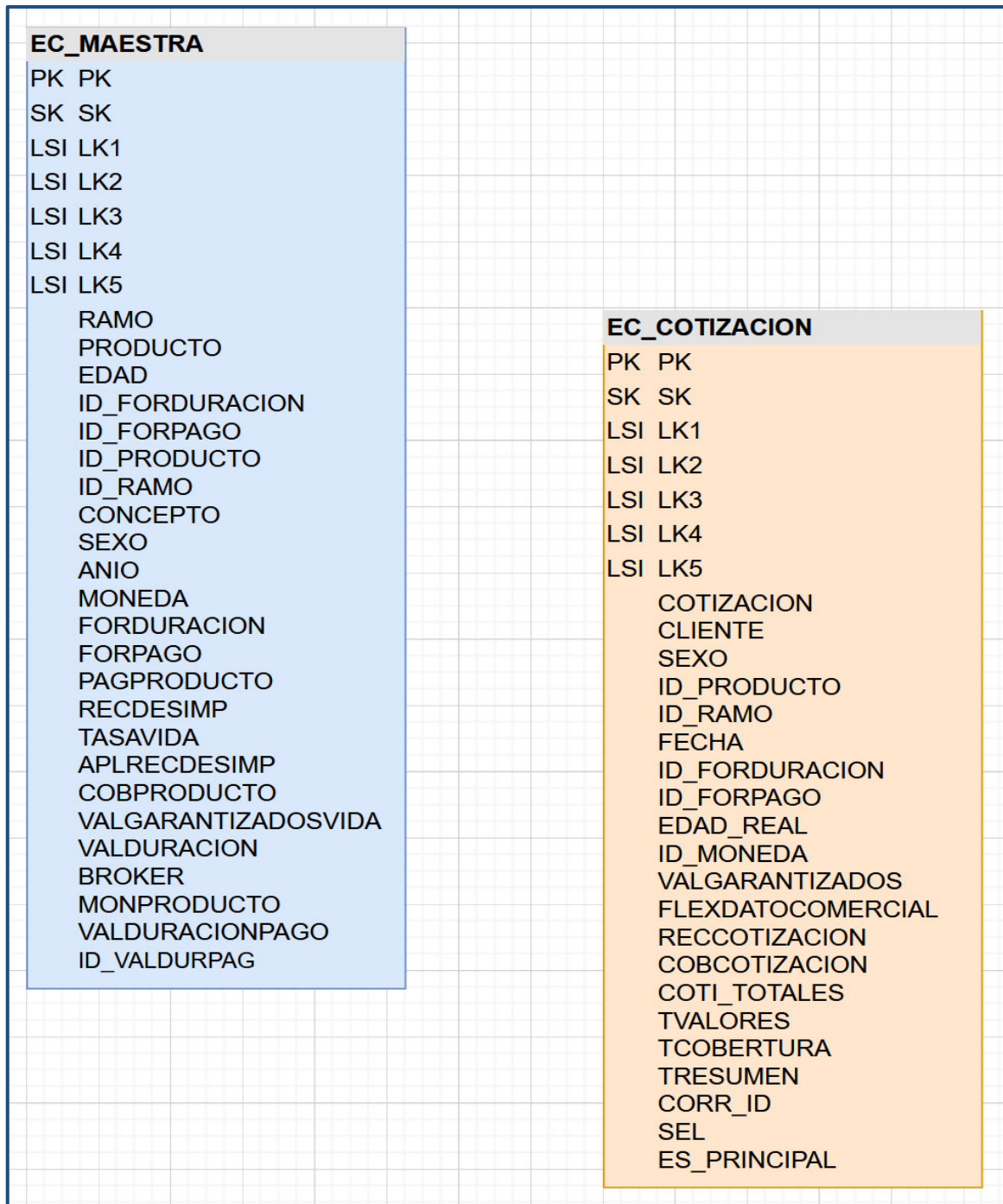


Figura 5 Diagrama físico de la base de datos para los nuevos servicios de cotizaciones.

Fuente. Elaboración propia

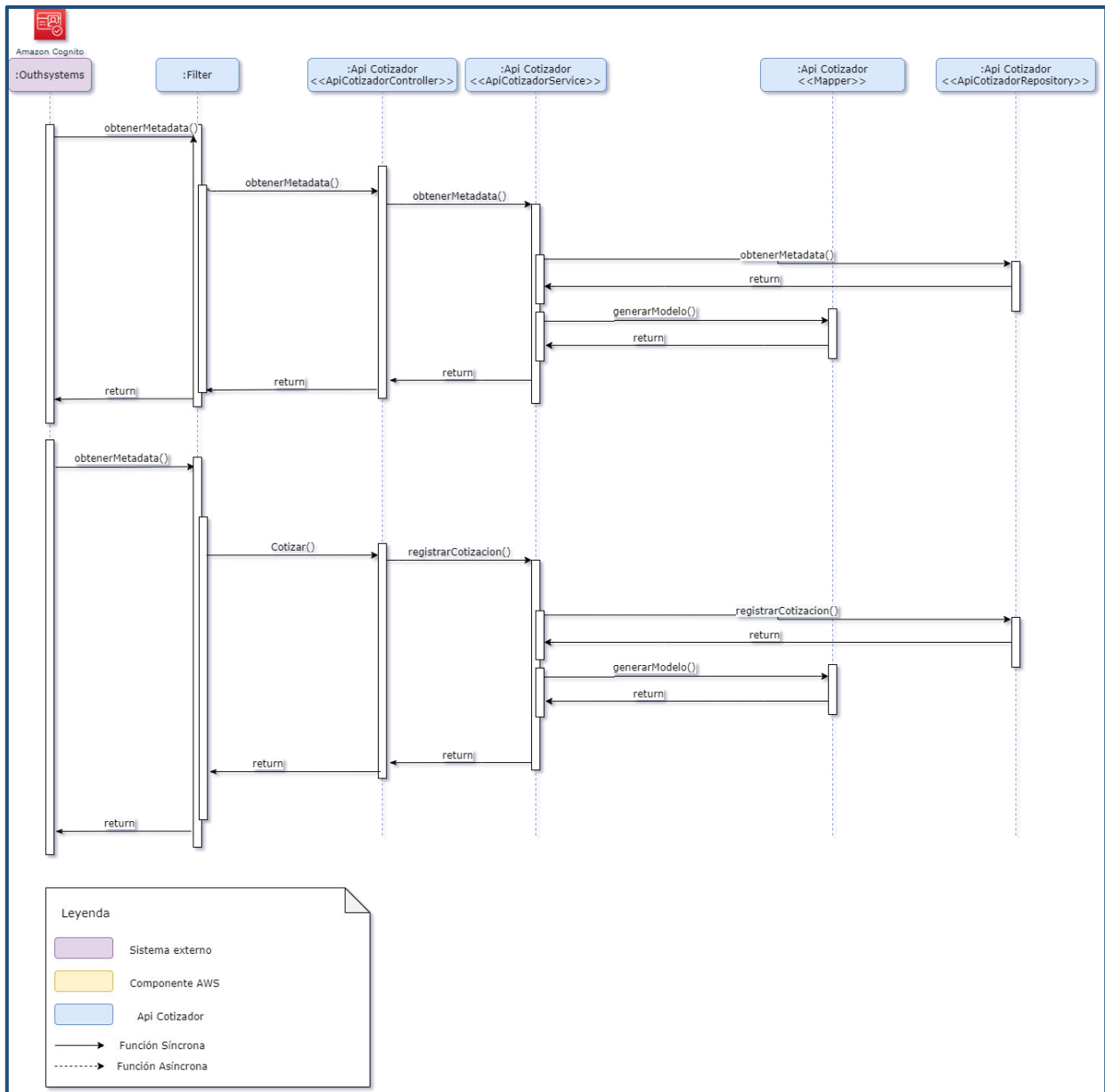


Figura 6 Diagrama de secuencia para los nuevos servicios de cotizaciones

Fuente. Elaboración propia

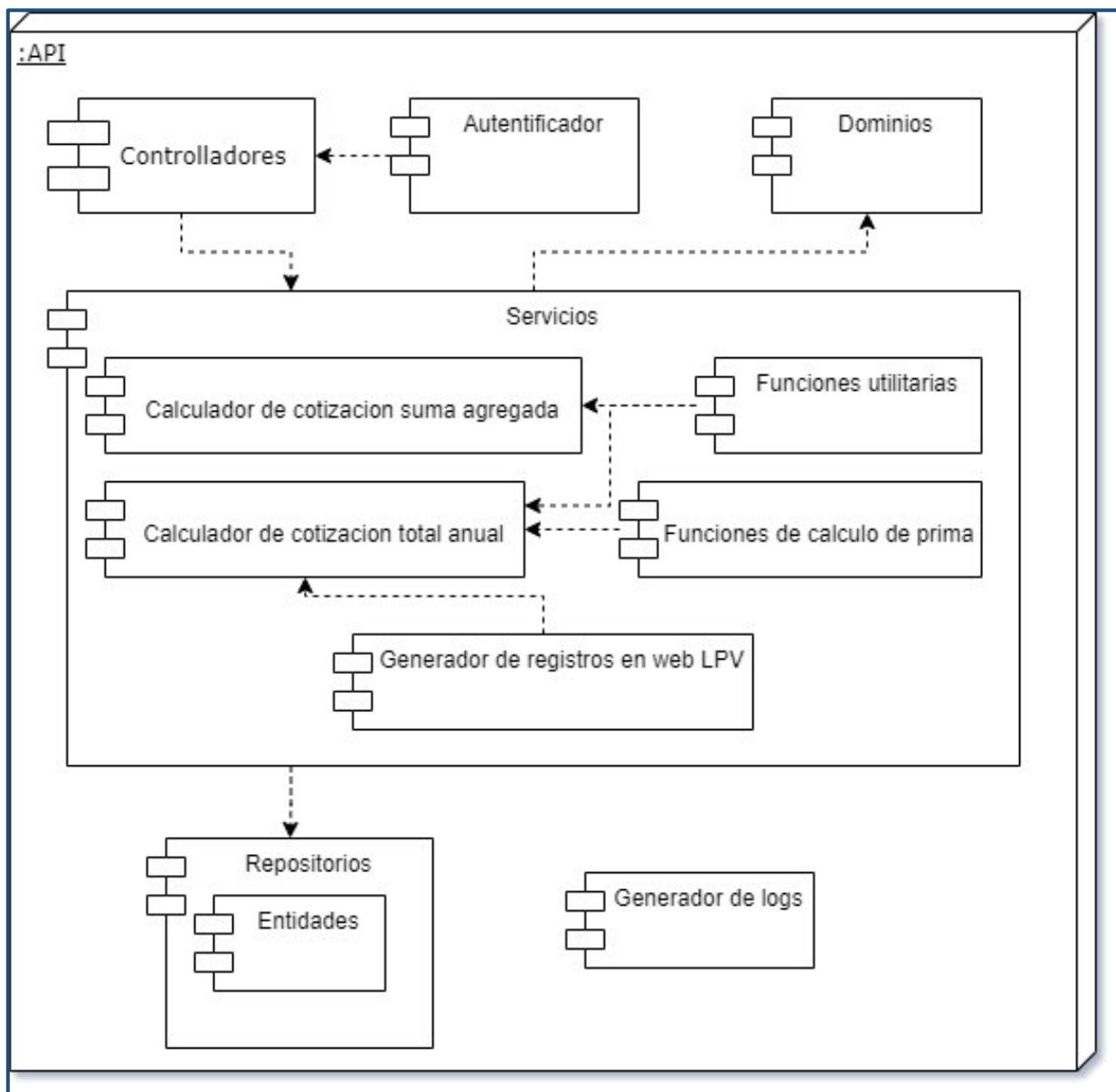


Figura 7 Diagrama de componentes para los nuevos servicios de cotizaciones

Fuente. Elaboración propia

Dominio	Subdominio	HU	Prototipo	METODO	RECURSO		
						/metadata	/quotation
					GET	POST	GET
CORE-LOGIST	METADATA	HU-01	image 005		X		
CORE-LOGIST	COTIZACION	HU-02	image 006			X	
CORE-LOGIST	COTIZACION	HU-03	image 006			X	
CORE-LOGIST	COTIZACION	HU-04	image 006				X
CORE-LOGIST	COTIZACION	HU-05	image 007			x	
CORE-LOGIST	COTIZACION	HU-06	image 008			x	

Figura 8 Matriz de integración Frontend/Backend

Fuente. Elaboración propia

Después de presentar los entregables anteriores, se realizó una reunión con el equipo del cliente para la revisión y confirmación de diseño de software, seguidamente una vez subsanado alguna observación o ajuste, se prosiguió con la conformidad y la continuación de la siguiente fase.

3.2.5.3. Fase 2: Análisis y diseño de la infraestructura AWS

Durante esta fase, al igual que en la fase anterior fueron necesarios algunos entregables por parte del cliente para cumplir con requerimientos básicos en base a su infraestructura en la nube, entregables como los siguientes:

- Entrega de requerimientos técnicos: documento en el que se definieron los componentes básicos para el diseño de la seguridad en la infraestructura, data en tránsito y en reposo; componentes básicos en el monitoreo, observabilidad y operación de la infraestructura; y por último los componentes necesarios para la alta disponibilidad y su la tolerancia a fallos.

Una vez presentada los requerimientos técnicos por el cliente, se programaba una reunión para ambos equipos, a las que tenía que asistir los jefes del proyecto y los líderes técnicos de ambas partes, con la finalidad de revisar la existencia de algún requerimiento que se encontrase fuera del alcance inicial. Seguidamente después de haber establecido la conformidad de los requerimientos de la infraestructura AWS se prosiguió con el diseño de despliegue en AWS, documento que consistió en definir algunos parámetros como etiquetas de negocio para los componentes que se tenían a usar; de igual manera que los procesos anteriores se realizó una reunión de ambos equipos con los mismos responsables del proyecto para la revisión y confirmación del diseño de despliegue AWS. A continuación, se detalla las actividades y entregables a cargo del equipo de migración:

- Creación de cloudformation: actividad que consistió en crear un archivo de extensión yaml que contiene todos los componentes AWS que la nueva plataforma va a contener, con este archivo se puede desplegar fácil y automáticamente en uno o varios ambientes dentro de una cuenta de AWS que tenga los permisos necesarios para el despliegue. Basado en un enfoque de infraestructura como código, permite que durante las fases de despliegue se pueda parametrizar los componentes usados y agilizar el proceso.
- Estimación de costos AWS: documento que detalla (precio, unidades, y consideraciones) una estimación de todos los gastos realizados por los servicios AWS utilizados mensualmente.

- Integración de OutSystems y backend en AWS: actividad que consistió en evaluar las configuraciones necesarias para establecer la conectividad entre los nuevos servicios migrados y el nuevo sistema de OutSystems.
- Pruebas técnicas unitarias: actividad que consistió en realizar pruebas de concepto en conectividad y de verificación para la comunicación efectiva entre los servicios backend (servicios web) y el frontend (nueva interfaz de OutSystems) del proyecto.

Finalmente, después de haber realizado las actividades y presentado los entregables mencionados anteriormente, se realizó una reunión con el equipo del cliente para la revisión y confirmación del diseño de infraestructura AWS, seguidamente una vez subsanado alguna observación o ajuste, se prosiguió con la conformidad y la continuación de la siguiente fase.

3.2.5.4. Fase 3: Implementación

Durante esta fase, el equipo de migración se enfocó principalmente en desarrollar dos actividades, la primera actividad fue el desarrollo de software que consistió en los siguientes puntos:

- **Desarrollo Frontend:** actividad que se encargó de diseñar la interfaz gráfica de la nueva plataforma web del sistema de cotizaciones migrado, teniendo que esta actividad no la realizó el equipo de migración a la que pertenecía el autor, si hubo coordinaciones para coordinar la integración con el backend para los servicios REST.
- **Desarrollo Backend:** actividad que se encargó de la refactorización de las antiguas funcionalidades del sistema de cotizaciones en nuevos servicios REST, así como en la construcción de la nueva arquitectura de software MVC.

Y finalmente la segunda actividad que fueron las pruebas, que consistió en los siguientes puntos:

- **Pruebas Unitarias:** actividad que consistió en realizar la comprobación de del correcto funcionamiento de los servicios REST desarrollados por el equipo de desarrollo para la futura integración con el frontend.
- **Pruebas QC:** actividad que consistió en la realización de las pruebas funcionales del documento previamente definido en la fase 1 de Análisis y diseño del software, por parte del equipo de calidad, estas pruebas fueron realizadas en un ambiente propio de QC dentro de la cuenta AWS del cliente, ambiente que el equipo de infraestructura desplegó gracias a la plantilla de cloudformation que se entregó en la fase 2 de análisis y diseño de la infraestructura AWS, donde finalmente se presentó un documento de evidencias de los resultados.

3.2.5.5. Fase 4: Transición

Durante esta fase, se preparó el ambiente QA mediante un nuevo despliegue con la plantilla de cloudformation que se entregó en la fase 2 de análisis y diseño de la infraestructura AWS, con este nuevo ambiente el equipo de pruebas del cliente realizó su propio plan de pruebas para validar el correcto funcionamiento de todas las combinaciones de las cotizaciones que podría realizar el nuevo sistema, así como la realización de pruebas funcionales y no funcionales, pruebas de humo , pruebas de carga y pruebas de estrés, pruebas de seguridad. En la finalización de esta fase el cliente presentó un informe (Anexo 4) con los resultados de las pruebas no funcionales que detallaron si el sistema cumplía con requerimientos mínimos para el futuro pase al ambiente de producción.

3.2.5.6. Fase 5: Lanzamiento

Durante esta última fase, se realizó el acompañamiento para el despliegue del ambiente de producción, de la misma manera se hizo una revisión y actualización de los documentos de despliegue y el documento de ficha técnica, así como una reunión con el equipo técnico del cliente con la finalidad de realizar una transferencia de conocimientos técnicos sobre los nuevos servicios REST y la nueva plataforma desplegada.

3.3. Evaluación

3.3.1. Evaluación Económica

Los costos del presente proyecto se dividen en relación a las horas trabajadas por cada uno de los roles involucrados durante todas las fases de trabajo. Cabe mencionar que los costos mostrados en la siguiente tabla, son referenciales al costo original.

Tabla 6 Estimación del costo del proyecto

Recursos	Cantidad	Costo (hora)	Duración (horas)	Costo total
Jefe de Proyecto	1	USD \$25	240	USD \$6000
Backend Dev	2	USD \$25	568	USD \$14200
QA Tester	1	USD \$25	216	USD \$5400
AWS Architect	1	USD \$50	248	USD \$12400
AWS Developer	1	USD \$35	80	USD \$2800
QA Documentador	1	USD \$25	32	USD \$800
			Total	USD \$41600

Fuente. Elaboración propia

3.3.2. Beneficios Obtenidos

Uno de los beneficios obtenidos al finalizar el presente proyecto fue la mejora en la performance de tiempo de respuesta por los servicios migrados a la nube. En la figura se puede apreciar la comparación de tiempos con respecto a cada uno de los servicios desarrollados, se puede apreciar que el servicio de obtención de “metadata” tiene un tiempo de respuesta de 0.993 segundos para 150 usuarios concurrentes. De igual manera se obtuvo una mejora en los tiempos de respuesta para los servicios de “cotización” y “fraccionamiento” en 4.389 segundos y 1.560 segundos respectivamente para 70 usuarios concurrentes.

Tabla 7 Comparación entre los tiempos de respuesta del nuevo y antiguo con 150 usuarios concurrentes por segundo para el servicio de metada

Escenario	Nuevo Cotizador	Antiguo Cotizador	Resultado
-----------	-----------------	-------------------	-----------

	Tiempo de respuesta máximo permitido Seg.	Tiempo de respuesta Seg.	Tiempo de respuesta Seg.	% de Mejora
Metadata	5	0,993	1,1	+0,9727%

Fuente. Elaboración propia

Tabla 8 Comparación entre los tiempos de respuesta del nuevo y antiguo con 70 usuarios concurrentes por segundo para el servicio de fraccionamiento

Escenario	Nuevo Cotizador		Antiguo Cotizador	
	Tiempo de respuesta máximo permitido Seg.	Tiempo de respuesta Seg.	Tiempo de respuesta Seg.	% de Mejora
Fraccionamiento	5	1,560	2,5	+37,6%

Fuente. Elaboración propia

Tabla 9 Comparación entre los tiempos de respuesta del nuevo y antiguo con 70 usuarios concurrentes por segundo para el servicio de cotización

Escenario	Nuevo Cotizador		Antiguo Cotizador	
	Tiempo de respuesta máximo permitido Seg.	Tiempo de respuesta Seg.	Tiempo de respuesta Seg.	% de Mejora

Cotización	5	4.275	6.3	+32.142%
-------------------	---	-------	-----	----------

Fuente. Elaboración propia

Y por el otro lado se obtuvo el beneficio en el ahorro de costos por los servicios consumidos mensualmente, por un lado, con la nueva plataforma desplegada se gasta un promedio mensual de 87.78 dólares mientras que con la plataforma anterior se estima que el costo mensual era de 200 dólares. Los factores determinantes en el ahorro mensual de servicios consumidos mensualmente se deben a que ya no se estarían pagando varias licencias, como de la de un servidor Windows y una licencia de base de datos Oracle.

Tabla 10 Comparación entre los costos mensuales del antiguo y el nuevo sistema de cotizaciones

Escenario	Nuevo Cotizador	Antiguo Cotizador	
	Gastos mensuales con servicios AWS dólares	Gastos mensuales dólares	% de Mejora
Cotización	85,53 USD	≈200 USD	+57,235%

Fuente. Elaboración propia

Tabla 11 Detalle de costo por servicio de AWS mensual

Servicio	Costo
AWS	dólares
CloudWatch	0,6

Elastic Container Registry	0,02
Elastic Load Balancer	0,62
Key Management Service	30,4
Route 53	0,1
DynamoDB	17,75
Cognito	0,00
Fargate	36,04
TOTAL	85,53

Fuente. Elaboración propia

CAPÍTULO IV REFLEXION CRITICA DE LA EXPERENCIA

El autor del presente trabajo de suficiencia profesional participó en el proyecto de migración de un sistema monolítico on premise hacia una plataforma cloud usando Amazon Web Services y el marco teórico de Well Architected Framework bajo los roles de arquitecto de soluciones cloud y líder del equipo de desarrollo de software, para el cliente de pólizas de seguros.

El autor estuvo presente en todas fases de desarrollo del proyecto a excepción de la fase del kick off, el autor tuvo una directa participación en cada una de las fases póstumas a la inicial, que le permitió realizar ajustes en la infraestructura si se presentaba alguna desviación que no se ajustara a los requerimientos y objetivos planteados del proyecto. El autor acompañó hasta la fase de despliegue y soporte una vez se terminó de publicar el nuevo sistema en la plataforma cloud de AWS.

Los certificados adquiridos como arquitecto y desarrollador asociado de AWS contribuyeron de manera significativa para que el autor pueda levantar todas las observaciones de infraestructura que al inicio del proyecto no estuvieron bien definidas y de poder realizar pruebas de concepto de AWS para constatar resultados al momento de optar por un recurso de AWS u otro.

Una de las experiencias adquiridas por el autor durante el desarrollo del proyecto fue la necesidad de separar un tiempo de análisis para las pruebas de concepto a nivel de arquitectura de infraestructura, que podrían darse durante fase de implementación, las pruebas de concepto desarrolladas en el proyecto ayudaron a aclarar el panorama del alcance a nivel de conectividad que iba a tener el nuevo sistema, que al final fueron cruciales para redefinir el alcance del proyecto y

especificar las nuevas condiciones de conexión para los nuevos servicios desarrollados. Otra experiencia adquirida por el autor fue el trabajo con las bases de datos NoSQL, de la cual el autor tuvo que tomarse un tiempo de análisis para el correcto diseño del nuevo modelo de base de datos, cabe recalcar que, si bien se realizó una estimación de tiempo para la migración, esta estimación se tuvo que aumentar debido a que su implementación también implicaba nuevas tecnologías para su desarrollo a nivel de código.

La experiencia adquirida por el autor durante todas las fases de trabajo del proyecto de migración lo ayudará para futuros escenarios de similar complejidad en el área de migraciones de plataformas, así como para desarrollo profesional como arquitecto de soluciones cloud de Amazon Web Services y como futuro ingeniero de software.

CAPÍTULO V CONCLUSIONES Y RECOMENDACIONES

Para concluir el presente trabajo de suficiencia profesional se detallará en este capítulo las conclusiones y recomendaciones más importantes logradas, relacionados al cumplimiento de objetivos, con la finalidad de dar continuidad al proyecto para que se pueda seguir a expandiendo a futuro.

5.1. Conclusiones

- El uso del marco teórico de Amazon, Well Architected Framework, ayudó en la selección correcta de recursos AWS para el diseño de la nueva infraestructura cloud, la operatividad eficiente de unos recursos de AWS sobre otros permitió mejorar el performance de los nuevos servicios desarrollados, de igual esta selección eficiente permitió reducir costos mensuales de infraestructura.
- Se mejoró el tiempo de respuesta en los tres nuevos servicios web desarrollados al migrar las funcionalidades a la plataforma cloud de AWS, cada uno de ellos mejoró en un 0,9727%, 37,6% y 32.142% respectivamente.
- Se redujo en un 57,235% los costos de recursos computacionales para nueva infraestructura cloud del sistema de cotizaciones con la migración realizada. El desuso de recursos de compute con licencia fue clave en este ahorro, así como el cambio de tipo de base de datos de relacional a no relacional.
- Se implementó una infraestructura cloud en AWS capaz de soportar la concurrencia de hasta 4200 usuarios por minuto para 2 de los tres servicios desarrollados (API de cotización y fraccionamiento) y de hasta 9000 usuarios concurrentes para el tercero (API de metadata).

- Se desplegaron tres nuevas API's dentro de la red privada de AWS del cliente, estando disponibles para cualquier otro sistema dentro de la misma red privada de manera inmediata.

5.2. Recomendaciones

- Se recomienda realizar pruebas de concepto, atendiendo las principales dudas que se puedan tener durante la fase de análisis, para evitar demoras o controles de cambio durante la fase de implementación. Darles unas horas a las pruebas de concepto podrían aliviar en el área de costos del proyecto, así como evitar impactar en otras actividades del calendario.
- Para la migración de información, se recomienda tener un especial cuidado con las migraciones de data entre las bases de datos, en especial si el tipo de base de datos es diferente, algunos motores de base de datos no soportan o no realizan este tipo de transferencia de datos por lo que vendría a ser necesario elaborar un plan de migración entre ellas, e incorporar este plan en el calendario de actividades.
- Para la conectividad, se recomienda incorporar a nivel de infraestructura de AWS, la oportunidad de que los servicios desarrollados puedan ser consumidos públicamente, claramente bajo una capa de seguridad acorde a las políticas de la empresa cliente, con la finalidad de que el sistema pueda llegar a más y diferentes tipos de usuarios y recolectar cada vez más información de muchos otros sistemas para ser explotada a futuro con el uso de Big Data.

FUENTES DE INFORMACIÓN

Amazon Web Services, Inc. (2021). *AWS Well-Architected*. Obtenido de AWS Well-Architected: <https://aws.amazon.com/es/architecture/well-architected/?wa-lens-whitepapers.sort-by=item.additionalFields.sortDate&wa-lens-whitepapers.sort-order=desc>

Applying Consulting. (2020). <https://www.applying.pe/>.

Artica, J. (16 de Octubre de 2020). SNI: “Los negocios que quieren seguir en el mercado tendrán que invertir en tecnología digital”. *El Comercio*.

Comercio, E. (13 de Septiembre de 2016). El 60% de empresas ve necesario agilizar sus operaciones. *El Comercio*.

Gupta, R. (30 de Julio de 2020). <https://tridentsys.net>. Obtenido de <https://tridentsys.net>: <https://tridentsys.net/aws-well-architected-framework/>

Pressman, R. S. (2010). Ingeniería del Software. Un enfoque práctico. En R. S. Pressman, *Ingeniería del Software. Un enfoque práctico* (págs. 33-34). México: McGraw Hill.

Sutherland, J. (2018). Scrum. En J. Sutherland, *Scrum* (pág. 23). Barcelona: Editorial Ariel.

GLOSARIO

- **API**

Conocido por sus siglas, Interfaz de Programación de Aplicaciones, este recurso permite la comunicación de servicios entre sistemas de diferentes orígenes.

- **Base de datos NoSQL o no relacional**

Bases de datos que difieren en el modelo convencional, como una de sus principales características es que no utilizan el lenguaje SQL para la ejecución de consultas, otra característica es que su data no tiene que almacenarse necesariamente en tablas y que permite que sus modelos de almacenamiento sean de estructura dinámica.

- **DynamoDB**

Base de datos no relacional de Amazon Web Services.

- **Amazon Web Services**

Es un proveedor de servicios en la nube, como almacenamiento, base de datos, recursos de computo, redes, seguridad de infraestructura entre muchos más dedicados al cloud computing.

- **Desnormalización**

Proceso que busca optimizar el procesamiento de búsquedas en una base de datos por medio de agregar redundancia en las tablas, simplificando relaciones y compactándolas entre sí.

- **Cognito**

Servicio de AWS que permite implementar de manera rápida el registro, inicio de sesión y control de acceso de usuarios a sus respectivas aplicaciones sean web o móviles.

- **Kick off**

Reunión preliminar al inicio de un proyecto en la que se presentan las partes como el jefe de proyecto y su equipo con las partes del cliente.

- **Daily**

Reunión de corto tiempo realizada al inicio de cada jornada de laboral con los miembros de trabajo de un proyecto, en la que cada integrante comenta las tareas realizadas el día anterior, los pendientes, obstrucciones de avances; concluyendo su participación comentando las actividades con las que se estaría desarrollando ese mismo día.

- **CloudFormation**

Servicio de AWS que permite modelar, diseñar, aprovisionar recursos de AWS de la infraestructura cloud de un sistema mediante código.

- **Cotización**

Acción realizada por el usuario para realizar la cotización de una póliza de seguro de vida basado en las características de un cliente, factores como la edad, si es fumador o no, el sexo, entre otras variables más, terminan dando como resultado el costo de una póliza de seguro y la cobertura que esta conllevaría.

ANEXOS

ANEXO 1. Plan de trabajo

ID	Nombre	Duración	Esfuerzo
1	DSV Sistema Cotizador	62días	1708horas
2	KickOff	1día	0hora
3	Gestión del Proyecto	60días	240horas
4	Análisis y Diseño del Software	7.5días	152horas
5	Entrega de Prerequisitos Técnicos	1día	8horas
6	Entrega de Estándares de Desarrollo (Frontend, Backend, BDs, Seguridad)	0.2día	1.6horas
7	Entrega de Accesos (Git, VPN, AWS, BD, Pipeline, Dominios, API de Login)	0.2día	1.6horas
8	Entrega de Ambiente DEV	0.2día	1.6horas
9	Entrega de Línea Gráfica + Mockups Web, Mobile	0.2día	1.6horas
10	Entrega de Ambiente DEV - Cotizador Actual	0.2día	1.6horas
11	Análisis de Requerimientos y Transferencia de Conocimiento	1día	24horas
12	Diseño del Software (Diagrama de Arquitectura, Componentes, Secuencia)	1día	16horas
13	Diseño de Casos de Prueba (Funcional y Técnico)	2días	16horas
14	Diseño de APIs en Swagger	2días	32horas
15	Matriz de Integración Frontend / Backend	1día	16horas
16	Confirmación de Estimación de Implementación	1día	0hora
17	Creación de Stack para Backend	0.5día	8horas
18	Revisión y Confirmación de Diseño de Software	1día	32horas
19	Diagrama de Arquitectura, Componentes, Secuencia	0.25día	2horas
20	Casos de Prueba (Funcional y Técnico)	0.25día	2horas
21	Diseño de APIs en Swagger	0.25día	2horas
22	Stacks Frontend + Backend	0.25día	2horas
23	Soporte de Revisión y Confirmación	1día	24horas
24	Análisis y Diseño de Infraestructura AWS	7.5días	156horas
25	Entrega de Requerimientos Técnicos	1día	8horas
26	Seguridad de Infraestructura y Software (En tránsito, En reposo)	0.25día	2horas
27	Operación, Monitoreo y Observabilidad	0.25día	2horas

28	Alta Disponibilidad y Tolerancia a Fallos	0.5día	4horas
29	Revisión y Análisis de Requerimientos Técnicos	0.5día	12horas
30	Diseño de Despliegue en AWS	1día	8horas
31	Revisión y Confirmación de Diseño AWS	1día	16horas
32	Creación de CloudFormation	3.5días	28horas
33	Estimación de Costos AWS (Mensuales)	0.5día	4horas
34	Integración con Stack OutSystems y Backend en AWS (Monitoreo, Observabilidad)	1día	16horas
35	Diseño e Implementación de Pipelines (DevOps)	1día	24horas
36	Pruebas Técnicas Unitarias	0.5día	8horas
37	Revisión y Confirmación de Diseño AWS	1día	32horas
38	Revisión de Infraestructura AWS	0.5día	4horas
39	Revisión de Diseño de Seguridad	0.5día	4horas
40	Soporte de Revisión y Confirmación	1día	24horas
41	Implementación	23días	872horas
42	Despliegue y Validación de Servicios en DEV	1día	8horas
43	Desarrollo del Software	12días	512horas
44	Configuración Inicial	2días	32horas
45	Desarrollo de Servicios REST (mockups)	2días	16horas
46	Mapeo de Integraciones desde OutSystems	2días	16horas
47	Desarrollo Frontend (OutSystems)	10días	80horas
48	Integración con APIs y Validaciones de Negocio	10días	80horas
49	Desarrollo Backend (15 APIs)	10días	320horas
50	Implementación de APIs, Validaciones de Negocio, Redis, Cognito	10días	320horas
51	Desarrollo de Casos de Prueba	10días	80horas
52	Diseño de Pruebas Automatizadas de APIs	10días	80horas
53	Pruebas Unitarias	5días	160horas
54	Ejecución de Pruebas con OutSystems	5días	40horas
55	Ejecución de Pruebas Funcionales de APIs	2.5días	20horas
56	Ejecución de Pruebas Técnicas de APIs	2.5días	20horas
57	Soporte a Pruebas	5días	80horas
58	Pruebas QC (Applying)	5días	192horas
59	Entrega de Ambiente QC	1día	8horas
60	Despliegue y Validación de Servicios en QC	1día	8horas
61	Ejecución de Pruebas Funcionales	4días	32horas
62	Ejecución de Pruebas Técnicas	4días	32horas
63	Generación de Evidencia de Pruebas	4días	32horas
64	Soporte a Pruebas	5días	80horas
65	Transición	19días	224horas
66	Entrega de Ambiente QA	1día	8horas
67	Generación de Procedimiento de Despliegue	1día	24horas

68	Despliegue y Validación de Servicios en QA	2días	16horas
69	Soporte a Despliegue de Servicios en QA	2días	16horas
70	Pruebas QA (Cliente)	15días	160horas
71	Ejecución de Pruebas Funcionales	5días	0hora
72	Ejecución de Pruebas Técnicas	5días	0hora
73	Ejecución de Pruebas de Seguridad	5días	40horas
74	Soporte a Pruebas	15días	120horas
75	Lanzamiento	4días	64horas
76	Generación de Procedimiento de Despliegue	1día	24horas
77	Actualización de Documentación Técnica	1día	24horas
78	Transferencia de Conocimiento	1día	8horas
79	Pase a Producción	1día	8horas

ANEXO 2. Casos de pruebas funcionales

Requerimiento	Código	Funcionalidad / Escenario / Caso	Resultado Esperado
Ingresar datos del seguro	CDP01	En caso que no se encuentre el parametro "token" y los parametros: "x-id-aplicacion", "x-id-usuario", "x-modulo" cuente con informacion correcta, cuando el usuario haga una peticion al servicio "metadata"	El servicio devolvera un mensaje de error y el codigo de response sera 401
	CDP02	En caso que no se encuentre el parametro "x-id-aplicacion" y los parametros: "token", "x-id-usuario", "x-modulo" cuente con informacion correcta, cuando el usuario haga una peticion al servicio "metadata"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP03	En caso que no se encuentre el parametro "x-id-usuario" y los parametros: "x-id-aplicacion", "token", "x-modulo" cuente con informacion correcta, cuando el usuario haga una peticion al servicio "metadata"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP04	En caso que no se encuentre el parametro "x-modulo" y los parametros: "x-id-aplicacion", "token", "x-id-usuario" cuente con informacion correcta, cuando el	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje"

Realizar la cotización del seguro		usuario haga una petición al servicio "metadata"	adicionalmente el código de respuesta será 500
	CDP05	En caso que se coloquen correctamente los parámetros, cuando el usuario haga una petición al servicio "metadata"	El servicio devolverá un mensaje de "Success", el código de respuesta será 200 y los objetos: "productos" (array), "tipoContratante"(array) y "tipoCotizacion"(array)
	CDP06	En caso que no se encuentre el parámetro "idRamo" y los parámetros: "idProducto", "idMoneda", "duracion", "fechaEfecto", "fechaFin", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con información correcta, cuando el usuario haga una petición "Cotizacion"	El servicio devolverá un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el código de respuesta será 500
	CDP07	En caso que no se encuentre el parámetro "idProducto" y los parámetros: "idRamo", "idMoneda", "duracion", "fechaEfecto", "fechaFin", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con información correcta, cuando el usuario haga una petición "Cotizacion"	El servicio devolverá un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el código de respuesta será 500
	CDP08	En caso que no se encuentre el parámetro "idMoneda" y los parámetros: "idRamo", "idProducto", "duracion", "fechaEfecto", "fechaFin", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "esFumador", "tipoContratante", "correo",	El servicio devolverá un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el código de respuesta será 500

		"telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	
	CDP09	En caso que no se encuentre el parametro "fechaEfecto" y los parametros: "idRamo", "idProducto", "idMoneda", "duracion", "fechaFin", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP10	En caso que no se encuentre el parametro "fechaFin" y los parametros: "idRamo", "idProducto", "idMoneda", "duracion", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP11	En caso que no se encuentre el parametro "idTipoCotizacion" y los parametros: "idRamo", "idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP12	En caso que no se encuentre el parametro "montoBase", "edadActuarial", "edadReal",	El servicio devolvera un mensaje de error junto con los objetos:

		"idBroker", "idFordur", "desDuracion", y los parametros: "idRamo","idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	"codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP13	En caso que no se encuentre el parametro "fechaNacimiento" y los parametros:"idRamo","idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP14	En caso que no se encuentre el parametro "sexo" y los parametros: "idRamo","idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "fechaNacimiento", "esFumador", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP15	En caso que no se encuentre el parametro "esFumador" y los parametros: "idRamo","idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo",	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500

		"fechaNacimiento", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	
	CDP16	En caso que no se encuentre el parametro "tipoContratante" y los parametros: "idRamo", "idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "correo", "telefono", "ip" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP17	En caso que el parametro "fechaNacimiento" de como resultado ser una persona menor a 18 años cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP18	En caso que el parametro "fechaFin" no sume los años seleccionados en la vigencia cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP19	En caso que el parametro "montoBase" no contenga el minimo o maximo permitido cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP20	En caso que el parametro "idRamo" no se encuentre en la BD cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje"

			adicionalmente el codigo de response sera 500
	CDP21	En caso que el parametro "idProducto" no se encuentre en la BD cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP22	En caso que no se encuentre el parametro "numeroDocumento" y los parametros: "idRamo", "idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "sexo", "fechaNacimiento", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de "Success", el codigo de response sera 200 y los objetos: "coberturas", "detalleValorGarantizado" y "fraccionesCotizacion"
	CDP23	En caso que no se encuentre el parametro "nombreCompleto" y los parametros: "idRamo", "idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "numeroDocumento", "sexo", "fechaNacimiento", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" cuenten con informacion correcta, cuando el usuario haga una peticion "Cotizacion"	El servicio devolvera un mensaje de "Success", el codigo de response sera 200 y los objetos: "coberturas", "detalleValorGarantizado" y "fraccionesCotizacion"
	CDP24	En caso que no se obtenga el token correctamente cuando el usuario haga una peticion del servicio "Cotizacion"	El servicio devolvera un mensaje de error "Solicitud no autorizada. Es necesario autenticar para obtener la respuesta solicitada.

Visualizar "Fraccionamiento"			Detalle:" y el codigo de response sera 401
	CDP25	En caso que se coloquen los parametros: "idRamo", "idProducto", "idMoneda", "duracion", "fechaFin", "fechaEfecto", "idTipoCotizacion", "montoBase", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "edadActuarial", "edadReal", "idBroker", "idFordur", "desDuracion", "nombreCompleto", "numeroDocumento", "sexo", "fechaNacimiento", "tipoContratante", "correo", "telefono", "ip", "duracionPag", "idFordurPag" correctamente, cuando el usuario haga una peticion del servicio "Cotizacion"	El servicio devolvera un mensaje de "Success", el codigo de response sera 200 y los objetos: "coberturas" (array), "detalleValorGarantizado" (array) y "fraccionesCotizacion" (array)
	CDP26	En caso que no se encuentre el parametro "token" y el parametro: "x-id-aplicacion", "x-id-usuario", "x-modulo", "idRamo", "idProducto", "id" (cotizacion), "concepto", "marcado", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP27	En caso no se encuentre el parametro "x-id-aplicacion" y el parametro: "token", "x-id-usuario", "x-modulo", "idRamo", "idProducto", "id" (cotizacion), "concepto", "marcado", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP28	En caso no se encuentre el parametro "x-id-usuario" y el parametro: "token", "x-id-aplicacion", "x-modulo", "idRamo", "idProducto", "id" (cotizacion), "concepto", "marcado", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP29	En caso no se encuentre el parametro "x-modulo" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "idRamo", "idProducto", "id" (cotizacion), "concepto", "marcado", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el

		usuario haga una peticion del servicio "Fraccionamiento"	codigo de response sera 500
	CDP30	En caso que no se encuentre el parametro "id" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "idRamo", "idProducto", "concepto", "marcado", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP31	En caso que el parametro "id" no exista en la BD y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "idRamo", "idProducto", "concepto", "marcado", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP32	En caso que no se encuentre el parametro "concepto" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "idRamo", "idProducto", "id", "marcado", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP33	En caso que no se encuentre el parametro "marcado" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "concepto", "idRamo", "idProducto", "id", "idMoneda", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP34	En caso que no se encuentre el parametro "idMoneda" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "idRamo", "idProducto", "concepto", "marcado", "id", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP35	En caso que no se encuentre el parametro "tipoCliente" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo",	El servicio devolvera un mensaje de error junto con los objetos: "codigoError",

		"idRamo", "idProducto", "concepto", "marcado", "idMoneda", "id" cuentan con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	"datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP36	En caso que no se encuentre el parametro "idRamo" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "idProducto", "concepto", "marcado", "idMoneda", "id", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP37	En caso que no se encuentre el parametro "idProducto" y el parametro: "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "idRamo", "concepto", "marcado", "idMoneda", "id", "tipoCliente" cuenten con informacion correcta, cuando el usuario haga una peticion del servicio "Fraccionamiento"	El servicio devolvera un mensaje de error junto con los objetos: "codigoError", "datoRecibido" y "mensaje" adicionalmente el codigo de response sera 500
	CDP38	Al momento de realizar la peticion del servicio "Fraccionamiento" se enviaron los parametros "token", "x-id-aplicacion", "x-id-usuario", "x-modulo", "idRamo", "concepto", "id" (cotizacion) "concepto", "marcado", "idMoneda" y "tipoCliente" correctamente.	El servicio devolvera un mensaje de "Success", el codigo de response sera 200 y un array de objetos con los atributos "periodo" y "valor"

ANEXO 3. Estructura de API's

Path	/cotizacion/
Método	POST
Descripción	Realizar y registrar una cotización
Resumen	El servicio se resolverá devolviendo un objeto "CotizacionResponseBean" para una respuesta exitosa; como se muestra en la parte inferior del servicio, la estructura de cada uno de los objetos que lo componen siendo estos: un arreglo del objeto "Cobertura", un arreglo del objeto "detalleValorGarantizados", un arreglo del objeto "valoresGarantizados" y un arreglo del objeto

	“fraccionesCotizacion”. Y para una respuesta fallida se resolverá devolviendo un objeto “ErrorResponse”. Para la resolución del servicio el Request debe de contar con un objeto “AutenticacionHeaderParametros” en el Header y un objeto “RequestParametrosCotizacion” en el RequestBody
Header-parámetros	<pre>{ "token": string, "x-id-aplicacion": string, "x-id-usuario": string, "x-modulo": string }</pre>
RequestBody	<pre>{ "cotizacion": { "idRamo": integer, "idProducto": integer, "idMoneda": integer, "duracion": integer, "duracionPag": integer, "fechaEfecto": string, "fechaFin": string, "idTipoCotizacion": integer, "montoBase": number, "edadActuarial": integer, "edadReal": integer, "idBroker": long, "idFordur": integer, "idFordurPag": integer, "desDuracion": string }, "cliente": { "nombreCompleto": string, "numeroDocumento": string, "sexo": integer, "fechaNacimiento": string, "esFumador": integer, "tipoContratante": integer, "correo": string, "telefono": string, "ip": string } }</pre>
Response-Success	<pre>{ "id": integer, "primaAnual": number, "sumaAsegurada": number, "usuarioCreacion": string, "fechaCreacion": string, "coberturas": [{ "correlativold": integer, "esPrincipal": string, </pre>

	<pre> "habilitadoRB":boolean, "nombre": string, "sumaAsegurada":number "primaTotalAnual": number, "idConcepto": integer, "seleccionado":string, "orden": integer }], "detalleValorGarantizados": [{ "intervaloPeriodo": string, "intervaloPeriodoNom":string, "valoresGarantizados": [{ "edad": integer, "anio":integer, "primaAcumulada": number, "valorAsegurado": number, "montoRescate": number, "montoSaldado": number, "anioProrrogado": integer, "montoProrrogado": integer }] }], "fraccionesCotizacion": [{ "periodo": string, "valor":number, "idForpago":integer, "desConcepto":string }] } </pre>
Response-Error	<pre> { "codigoError": string, "tipoError":string, "datoRecibido": string, "mensaje": string, "idlog":string } </pre>
Cotizacion	<pre> { "idRamo": integer, "idProducto": integer, "idMoneda": integer, "duracion": integer, "duracionPag": integer, "fechaEfecto": string, "fechaFin": string, "idTipoCotizacion": integer, "montoBase": number, </pre>

	<pre> "edadActuarial": integer, "edadReal": integer, "idBroker": long, "idFordur": integer, "idFordurPag": integer, "desDuracion": string } </pre>
Cliente	<pre> { "nombreCompleto": string, "numeroDocumento": string, "sexo": integer, "fechaNacimiento": string, "esFumador": integer, "tipoContratante": integer, "correo": string, "telefono": string, "ip": string } </pre>
CotizacionResponseBean	<pre> { id:integer primaAnual:string sumaAsegurada:string usuarioCreacion:string fechaCreacion:string coberturas:[{ "correlativold": integer, "esPrincipal":string, "habilitadoRB":boolean, "nombre": string, "sumaAsegurada":number "primaTotalAnual": number, "idConcepto": integer, "seleccionado":string, "orden": integer }] detalleValorGarantizados: [{ intervaloPeriodo:string valoresGarantizados :[{ edad:string primaAcumulada:string valorRescate:string seguroSaldado:string anioProrrogado:string seguroProrrogado:string }] }] } </pre>

Cobertura	<pre>{ "correlativold": integer, "esPrincipal":string, "habilitadoRB":boolean, "nombre": string, "sumaAsegurada":number "primaTotalAnual": number, "idConcepto": integer, "seleccionado":string, "orden": integer }</pre>
DetalleValorGarantizado	<pre>{ "intervaloPeriodo": string, "intervaloPeriodoNom":string, "valoresGarantizados": [{ "edad": integer, "anio":integer, "primaAcumulada": number, "valorAsegurado": number, "montoRescate": number, "montoSaldado": number, "anioProrrogado": integer, "montoProrrogado": integer }] }</pre>
ValorGarantizado	<pre>{ "edad": integer, "anio":integer, "primaAcumulada": number, "valorAsegurado": number, "montoRescate": number, "montoSaldado": number, "anioProrrogado": integer, "montoProrrogado": integer }</pre>
FraccionCotizacion	<pre>{ "periodo": string, "valor": number, "idForpago":integer, "desConcepto":string }</pre>
ErrorResponse	<pre>{ "codigoError": string, "tipoError":string, "datoRecibido": string, "mensaje": string, "idlog":string }</pre>

AutenticacionHeaderParametros	<pre>{ "token": string, "x-id-aplicacion": string, "x-id-usuario": string, "x-modulo": string }</pre>
RequestParametrosCotizacion	<pre>{ "cotizacion": { "idRamo": integer, "idProducto": integer, "idMoneda": integer, "duracion": integer, "duracionPag": integer, "fechaEfecto": string, "fechaFin": string, "idTipoCotizacion": integer, "montoBase": number, "edadActuarial": integer, "edadReal": integer, "idBroker": long, "idFordur": integer, "idFordurPag": integer, "desDuracion": string }, "cliente": { "nombreCompleto": string, "numeroDocumento": string, "sexo": integer, "fechaNacimiento": string, "esFumador": integer, "tipoContratante": integer, "correo": string, "telefono": string, "ip": string } }</pre>

Path	/ {id} /fraccionamiento
Método	GET
Descripción	Realiza el cálculo del fraccionamiento de suma asegurada o la prima total anual en periodos de anual, semestral, trimestral y mensual.
Resumen	El servicio se resolverá devolviendo un arreglo del objeto "Coberturas" y el arreglo del objeto "fraccionamientos" para una respuesta exitosa. Y para una respuesta fallida se resolverá devolviendo un objeto "ErrorResponse".

	Para la resolución del servicio el Request debe de contar con un objeto "AutenticacionHeaderParametros" en el Header y los parametros "token", "x-id-aplicacion", "x-id-usuario" y "x-modulo" ambos requeridos
Header-parámetros	<pre>{ "token": string, "x-id-aplicacion": string, "x-id-usuario": string, "x-modulo": string }</pre>
Query-parámetros	<pre>id : integer - path - required concepto :number - query - required marcado: string - query - required idMoneda: integer - query - required tipoCliente: string - query - required</pre>
Response-Success	<pre>[{ "coberturas": [{ "correlativoId": integer, "esPrincipal":string, "habilitadoRB":boolean, "nombre": string, "sumaAsegurada":number "primaTotalAnual": number, "idConcepto": integer, "seleccionado":string, "orden": integer }], "fraccionamientos": [{ "periodo": string, "valor": number, "idForpago": integer }] }]</pre>
Response-Error	<pre>{ "codigoError": string, "tipoError":string, "datoRecibido": string, "mensaje": string, "idlog":string }</pre>

FraccionCotizacion	{ "periodo": string, "valor": number, "idForpago":integer, "desConcepto":string }
ErrorResponse	{ "codigoError": string, "tipoError":string, "datoRecibido": string, "mensaje": string, "idlog":string }
AutenticacionHeaderParametros	{ "token": string, "x-id-aplicacion": string, "x-id-usuario": string, "x-modulo": string }

Path	/metadata
Método	GET
Descripción	Realiza la obtención de la lista de productos, tipo de personas(natural o juridica), tipo de monedas y pagos-vigencia.
Resumen	El servicio se resolverá devolviendo un objeto "Metadata" para una respuesta exitosa; como se muestra en la parte inferior del servicio, la estructura de cada uno de los objetos que lo componen siendo estos: un arreglo del objeto "Productos", un arreglo del objeto "vigencias", un arreglo del objeto "monedas", un arreglo del objeto "TipoContratante" y un arreglo del objeto "TipoCotizacion". Y para una respuesta fallida se resolverá devolviendo un objeto "ErrorResponse". Para la resolución del servicio el Request debe de contar con un objeto "AutenticacionHeaderParametros" en el Header.
Header-parámetros	{ "token": string, "x-id-aplicacion": string, "x-id-usuario": string, "x-modulo": string }

RequestBody	-
Response-Success	<pre> { "productos": [{ "nombre": string, "id": string, "valor": string, "vigencias": [{ "nombre": string, "nombrePagos": string, "valor": string, "valorPagos": string, "idProducto": integer, "idRamo": integer, "idForDuracion": integer, "duracion": integer, "idForDuracionPag": integer, "duracionPag": integer }], "monedas": [{ "nombre": string, "id": integer }] }], "tipoContratante": [{ "nombre": string, "id": integer }], "tipoCotizacion": [{ "nombre": string, "id": integer }] } </pre>
Response-Error	<pre> { "codigoError": string, "tipoError": string, "datoRecibido": string, "mensaje": string, "idlog": string } </pre>

Metadata	<pre> { "productos": [{ "nombre": string, "id": string, "valor": string, "vigencias": [{ "nombre": string, "nombrePagos": string, "valor": string, "valorPagos": string, "idProducto": integer, "idRamo": integer, "idForDuracion": integer, "duracion": integer, "idForDuracionPag": integer, "duracionPag": integer }], "monedas": [{ "nombre": string, "id": integer }] }], "tipoContratante": [{ "nombre": string, "id": integer }], "tipoCotizacion": [{ "nombre": string, "id": integer }] } </pre>
Producto	<pre> { "productos": [{ "nombre": string, "id": string, "valor": string, "vigencias": [{ "nombre": string, "valor": string, </pre>

	<pre> "idProducto": integer, "idRamo": integer, "idForDuracion": integer, "duracion": integer }], "monedas": [{ "nombre": string, "id": integer }] }], "tipoContratante": [{ "nombre": string, "id": integer }], "tipoCotizacion": [{ "nombre": string, "id": integer }] } </pre>
Vigencia	<pre> { "nombre": string, "nombrePagos": string, "valor": string, "valorPagos": string, "idProducto": integer, "idRamo": integer, "idForDuracion": integer, "duracion": integer, "idForDuracionPag": integer, "duracionPag": integer } </pre>
Moneda	<pre> { "nombre": string, "id": integer } </pre>
TipoContratante	<pre> { "nombre": string, "id": integer } </pre>
TipoCotizacion	<pre> { </pre>

	<pre> "nombre": string, "id": integer } </pre>
ErrorResponse	<pre> { "codigoError": string, "tipoError":string, "datoRecibido": string, "mensaje": string, "idlog":string } </pre>
AutenticacionHeaderParametros	<pre> { "token": string, "x-id-aplicacion": string, "x-id-usuario": string, "x-modulo": string } </pre>

ANEXO 4. Fragmento de los resultados finales de las pruebas QA

7. Conclusiones generales

Las **pruebas de carga** (Escenario 1) en los 3 endpoint (Metadata, Cotización y Fraccionamiento) del servicio web [REDACTED] Cotización, el resultado es **CONFORME**, ya que los resultados en tiempos de respuesta son menores a los esperados; y los porcentajes de errores tampoco superan el porcentaje máximo permitido. A continuación, se muestran en detalle los resultados en cada escenario de cada endpoint.

7.1. LP - Metadata:

Para el endpoint **Metadata**, los resultados son siguientes.

❖ Escenario 1: 70 Usuarios (Prueba de Carga): **CONFORME**

Escenario	1era Ejecución			
	Tiempo de repuesta esperado Seg	Tiempo de respuesta Seg	%Error Máximo permitido	%Error obtenido
Metadata	5	1.532	5%	0.56%

Tabla 9. Resultados de la prueba con 70 usuarios

❖ Escenario 2: 150 Usuarios (Prueba de estrés): **Informativo**

Escenario	1era Ejecución			
	Tiempo de repuesta esperado Seg	Tiempo de respuesta Seg	%Error Máximo permitido	%Error obtenido
Metadata	5	0.993	5%	0.23%

Tabla 10. Resultados de la prueba con 150 usuarios

7.2. LP - Cotización:

Para el endpoint **Cotización**, los resultados son siguientes.

❖ Escenario 1: 70 Usuarios (Prueba de Carga): **CONFORME**

Escenario	1era Ejecución			
	Tiempo de repuesta esperado Seg	Tiempo de respuesta Seg	%Error Máximo permitido	%Error obtenido
Cotización	5	4.389	5%	0.26%

Tabla 11. Resultados de la prueba con 70 usuarios

❖ Escenario 2: 150 Usuarios (Prueba de estrés): **Informativo**

Escenario	1era Ejecución			
	Tiempo de repuesta esperado Seg	Tiempo de respuesta Seg	%Error Máximo permitido	%Error obtenido
Cotización	5	10.379	5%	11.76%

Tabla 12. Resultados de la prueba con 150 usuarios

Nota: Este endpoint de cotización no soporta más de 100 usuarios concurrentes; ergo, con más de 100 usuarios provoca tiempos mayores de respuesta y mayor porcentaje de errores.

7.3. LP - Fraccionamiento

Para el endpoint **Fraccionamiento**, los resultados son siguientes.

❖ Escenario 1: 70 Usuarios (Prueba de Carga): **CONFORME**

Escenario	1era Ejecución			
	Tiempo de repuesta esperado Seg	Tiempo de respuesta Seg	%Error Máximo permitido	%Error obtenido
Fraccionamiento	5	1.560	5%	0%

Tabla 13. Resultados de la prueba con 70 usuarios

❖ Escenario 2: 150 Usuarios (Prueba de estrés): **Informativo**

Escenario	1era Ejecución			
	Tiempo de repuesta esperado Seg	Tiempo de respuesta Seg	%Error Máximo permitido	%Error obtenido
Fraccionamiento	5	4.275	5%	0.21

Tabla 14. Resultados de la prueba con 150 usuarios