



### **Tema**

# **ESPECIFICACIÓN DE REQUERIMIENTOS DE SISTEMA DE EUREKABANK**

### **Autores**

BARRIONUEVO ORDÓÑEZ LINDSAY DOMENIQUE

RIVERA LÓPEZ JOEL ALESSANDRO

YARANGA SUQUILLO LEONARDO JAVIER

### **Tutor**

Ing. Eduardo Mauricio Campaña Ortega

MIS.MDU.CCNA.CCIA.

PhD. (c) Ingeniería de Software PhD. (c) Seguridad Información

### **Fecha**

10/12/2025

# ESPECIFICACIÓN DE REQUERIMIENTOS DEL PROYECTO DE SISTEMA DE EUREKABANK

## ÍNDICE DE CONTENIDOS

ESPECIFICACIÓN DE REQUERIMIENTOS DEL PROYECTO DE SISTEMA DE EUREKABANK .....	1
1. Introducción .....	5
2. Propósito .....	5
3. Alcance.....	5
4. Definiciones, Acrónimos y Abreviaturas .....	6
4.1 Definiciones.....	6
4.2 Acrónimos y abreviaturas .....	8
5. Referencias .....	9
6. Visión General del Documento .....	10
7. Descripción General .....	10
7.1 Perspectiva del Producto .....	11
7.2 Funciones del Producto (software).....	11
7.3 Condiciones del Entorno .....	13
7.4 Características de los Usuarios.....	13
7.5 Interfaces Externas .....	15
7.6 Restricciones.....	15
7.7 Suposiciones y Dependencias .....	16
8. Especificación de Requerimientos.....	16
8.1 Requisitos comunes de las interfaces.....	16
8.1.1 Interfaces de usuario .....	16
8.1.2 Interfaces de hardware .....	16
8.1.3 Interfaces de software.....	18
8.1.4 Interfaces de comunicación .....	20
8.2 Requisitos Funcionales .....	20
8.2.1. Objetivo General .....	20
8.2.3. Actores.....	21
8.2.4. Lista de requisitos funcionales .....	22
8.2.5 Requisitos Funcionales .....	23
8.3 Requisitos No Funcionales.....	24

8.3.1 Objetivo General .....	24
8.4 Otros Requerimientos .....	26
8.4.1 Restricciones de Diseño .....	26
8.4.2 Restricciones de Hardware .....	27
8.4.3 Atributos de calidad .....	27
9. Requisitos de Rendimiento .....	27
9.1. Requisitos de Interfaces Externas.....	27
9.1.1. Interfaces de Usuario .....	27
9.1.2. Interfaces de Hardware.....	28
9.1.3. Interfaces de Software .....	28
9.1.4. Interfaces de Comunicación.....	28
9.1.5. Base de Datos .....	28

## ÍNDICE DE ILUSTRACIONES

Ilustración 1. Diagrama de casos de uso del sistema .....	13
---	----

## ÍNDICE DE TABLAS

Tabla 1. Definiciones.....	6
Tabla 2. Acrónimo y abreviaturas.....	8
Tabla 3. Referencias .....	9
Tabla 4. Usuario Administrador de Sistemas .....	14
Tabla 5. Usuario Gerente .....	14
Tabla 6. Usuario Cajero .....	14
Tabla 7. Usuario Cliente.....	15
Tabla 8. Requisitos de interfaces de hardware .....	16
Tabla 9. Interfaces de software .....	18
Tabla 10. Autenticación y Seguridad (Login).....	20
Tabla 11. Visibilidad Operativa (Cuentas) .....	20
Tabla 12. Auditoría (Movimientos).....	21
Tabla 13. Integridad Transaccional (Registro) .....	21
Tabla 14. Actor Cajero (Escritorio) .....	21
Tabla 15. Actor Gerente (Web) .....	21
Tabla 16. Actor Administrador (Consola) .....	21
Tabla 17. Actor Cliente (Móvil) .....	22
Tabla 18. Lista de requisitos funcionales .....	22
Tabla 19. Gestionar Login .....	23
Tabla 20. Obtener Cuentas con sus Clientes.....	23
Tabla 21. Obtener Movimientos de una Cuenta.....	23
Tabla 22. Registrar Movimiento .....	24
Tabla 23. Requerimiento no funcional Tiempo de respuesta .....	24
Tabla 24. Requerimiento no funcional utilización de colores.....	24
Tabla 25. Requerimiento no funcional ícono de operaciones.....	25
Tabla 26. Requerimiento no funcional métodos de acceso.....	25
Tabla 27. Requerimiento no funcional plataforma.....	25
Tabla 28. Requerimiento no funciona accesibilidad .....	25
Tabla 29. Requerimiento no funcional mantenimiento .....	26

## CASO DE ESTUDIO

Se desea desarrollar una aplicación para facilitar y comprender el proceso de una entidad financiera, sus movimientos y cómo trabaja. El "Eureka Bank" es una entidad financiera que requiere modernizar su plataforma tecnológica para centralizar y asegurar sus operaciones transaccionales. Actualmente, el banco maneja información crítica sobre Clientes, Cuentas de Ahorro, Movimientos Monetarios y Empleados, pero necesita una capa de servicios robusta que permita la interoperabilidad entre sus diferentes canales de atención como pueden ser cajeros, Ventanilla, Banca Web.

El funcionamiento operativo del banco que debe automatizar el sistema es el siguiente:

1. Gestión de Seguridad: El acceso a las operaciones sensibles está restringido. Todo empleado (Cajero, Gerente) debe autenticarse contra una base de datos centralizada para obtener permisos, dirigidos por el superadmin Monster.
2. Operatividad de Cuentas: Los clientes poseen cuentas bancarias asociadas a una sucursal específica. El sistema debe ser capaz de identificar no solo el saldo, sino también los datos del titular y el estado de la cuenta.
3. Transaccionalidad (Depósitos y Retiros):
  - Cuando un cliente realiza un depósito, el sistema debe identificar la cuenta, registrar la entrada de dinero y actualizar el saldo acumulado de forma atómica.
  - Cuando se solicita un retiro, el sistema debe validar estrictamente que el saldo disponible sea mayor o igual al monto solicitado. Si la validación es exitosa, se debita el monto y se genera un número de operación único.
4. Auditoría e Historial: Cada operación (movimiento) queda registrada con fecha, hora, empleado responsable y tipo de movimiento. Los clientes pueden solicitar un extracto (historial) para visualizar sus últimos movimientos ordenados cronológicamente.

El sistema será desarrollado con NetBeans como IDE, y María DB como base de datos. Adicionalmente, el proyecto técnico implica desarrollar esta lógica en una arquitectura SOA (Service Oriented Architecture) usando Servicios Web SOAP con Java (JAX-WS), desplegando la base de datos MariaDB en la nube de Google Cloud Platform para garantizar alta disponibilidad.

# 1. Introducción

Este documento de Especificación de Requerimientos de Software, ERS, ha sido elaborado siguiendo los lineamientos de la norma IEEE 830, adaptada para proyectos de integración de servicios. Constituye la documentación técnica fundamental previa al desarrollo e implementación de la capa de servicios del sistema Eureka Bank. El documento detalla la arquitectura, las interfaces y los comportamientos funcionales necesarios para construir una solución bancaria basada en el protocolo SOAP.

## 2. Propósito

El propósito principal de este documento es definir, listar y detallar exhaustivamente los requerimientos funcionales y no funcionales del sistema "Eureka Bank Service Layer". Este documento sirve como la fuente de verdad única para los desarrolladores Java, los arquitectos de base de datos y los gestores del proyecto. Su objetivo es eliminar la ambigüedad en la lógica de negocio bancaria, estableciendo claramente cómo deben comportarse los servicios web ante transacciones financieras críticas, asegurando que todas las partes interesadas, Stakeholders, tengan una visión unificada del producto final a desplegarse en Google Cloud. Este documento servirá como guía maestra para:

- Los desarrolladores del servicio SOAP en Java (Backend).
- Los desarrolladores de las interfaces de usuario en sus respectivas tecnologías (Java Swing, Web/HTML5, Consola, Android).
- Los arquitectos de infraestructura encargados del despliegue en Google Cloud.
- El equipo de QA para validar la correcta integración entre los clientes y el servicio.

Es principalmente preparado para establecer el escenario para la fase de diseño del proyecto. El entregable que está siendo elaborado es la primera versión del documento de visión para el proyecto enfocado en el sistema de "Eureka Bank".

## 3. Alcance

El presente trabajo se centra en la construcción del sistema Transaccional de Eureka Bank, tomando en cuenta su desarrollo Backend y Frontend, es decir, su lógica Full Stack. El sistema automatizará la comunicación entre las interfaces de usuario (Frontend) y los datos almacenados en la nube.

La implementación del sistema, una vez finalizada tendrá como beneficios la gestión de los siguientes módulos:

### 1. CAPA DE SERVICIOS (Backend)

- 1.1. Desarrollo de Web Services SOAP (JAX-WS) que exponen la lógica de negocio (Login, Cuentas, Movimientos)

### 2. CAPA DE DATOS (Cloud)

- 2.1. Implementación y gestión de la base de datos MariaDB en una instancia de Google Cloud SQL.

### 3. CAPA DE CLIENTE

- 3.1. Cliente Consola: Interfaz de línea de comandos (CLI) para ejecución rápida y bajo consumo de recursos.
- 3.2. Cliente Escritorio: Aplicación Java (Swing/JavaFX) para uso intensivo por parte de los cajeros del banco.
- 3.3. Cliente Web: Portal accesible vía navegador para consultas de gerencia y clientes.
- 3.4. Cliente Móvil: App Android que permite a los usuarios consultar sus saldos desde cualquier lugar.

El sistema permitirá realizar las siguientes funciones críticas en todas las plataformas: Autenticación de usuarios, Consulta de saldos y movimientos, Reportes de sucursal (Web/Escritorio) y Ejecución de transacciones monetarias (Escritorio/Consola).

## 4. Definiciones, Acrónimos y Abreviaturas

### 4.1 Definiciones

Tabla 1. Definiciones

TÉRMINO	DEFINICIÓN
Backend	Capa de acceso a datos y lógica de negocio que se ejecuta en el servidor. En este proyecto, corresponde al Servicio Web SOAP alojado en Apache Tomcat/GlassFish.
Cliente Pesado (Escritorio)	Aplicación desarrollada en Java Swing que se instala localmente en las estaciones de trabajo. Consume recursos del PC del usuario y se conecta al servidor para operaciones críticas.
Cliente Ligero (Web)	Aplicación nativa (Android APK) diseñada para ejecutarse en smartphones, optimizada para pantallas táctiles y redes de datos móviles (4G/5G).
Cliente Móvil	Detalle de cuotas mensuales con capital, intereses y saldo.
Cliente Consola (CLI)	Interfaz de Línea de Comandos. Aplicación minimalista basada en texto, utilizada por administradores para pruebas de conectividad y ejecución de operaciones sin carga gráfica.
Deploy (Despliegue)	Proceso de trasladar el código compilado (archivo .war) al servidor de aplicaciones y la estructura de base de datos a la nube para que el sistema esté operativo.

<b>Endpoint</b>	Punto de acceso específico (URL) donde el servicio web SOAP escucha y atiende las peticiones. Ejemplo: <code>http://ip-servidor:8080/EurekaBank/EurekaService</code> .
<b>Google Cloud SQL</b>	Servicio de base de datos relacional completamente administrado por Google, que permite configurar, mantener y administrar bases de datos MariaDB en la nube.
<b>JDBC (Java Database Connectivity)</b>	API estándar de Java que permite a las aplicaciones conectarse a bases de datos relacionales, enviar consultas SQL y procesar resultados.
<b>JAX-WS</b>	<i>Java API for XML Web Services</i> . Tecnología de Java utilizada para crear servicios web SOAP y clientes, facilitando la generación de contratos WSDL.
<b>Latencia</b>	Tiempo que tarda un paquete de datos en viajar desde el cliente (ej. Celular) hasta el servidor (Cloud) y regresar. Crítico para la experiencia de usuario.
<b>Marshalling / Unmarshalling</b>	Proceso de conversión de objetos Java en formato XML (Marshalling) para su envío por la red, y viceversa (Unmarshalling) al recibir una respuesta.
<b>Middleware</b>	Software que actúa como puente entre el sistema operativo/base de datos y las aplicaciones, en este caso, el servidor de aplicaciones que aloja el servicio SOAP.
<b>Pool de Conexiones</b>	Mecanismo que mantiene un caché de conexiones a la base de datos abiertas y listas para ser reutilizadas, mejorando el rendimiento ante múltiples usuarios simultáneos.
<b>Payload</b>	Carga útil. Contenido real del mensaje SOAP (ej. Número de Cuenta, Monto) excluyendo los encabezados de protocolo.
<b>Rollback</b>	Operación de base de datos que deshace una transacción en curso (ej. Depósito fallido), revirtiendo los datos a su estado anterior para garantizar integridad.
<b>SOAP (Simple Object Access Protocol)</b>	Protocolo de mensajería basado en XML para intercambiar información estructurada. Es independiente del sistema operativo y lenguaje de programación.
<b>Transacción ACID</b>	Conjunto de operaciones que garantizan Atomicidad (todo o nada), Consistencia (datos válidos), Aislamiento (sin interferencias) y Durabilidad (persistencia).
<b>WSDL (Web Services Description Language)</b>	Documento XML que describe la interfaz pública del servicio web, detallando métodos, parámetros, tipos de datos y ubicación del servicio.



<b>Gestionar</b>	Hacer las acciones o los trámites necesarios para conseguir o resolver una cosa.
<b>Automatizar</b>	Aplicar máquinas o procedimientos automáticos en la realización de un proceso o en una industria o empresa.
<b>Control de procesos</b>	El control es una de las principales actividades administrativas dentro de las organizaciones. El control es el proceso de verificar el desempeño de distintas áreas o funciones de una organización.
<b>Integral</b>	Comprende todos los aspectos o todas las partes necesarias para estar completo.
<b>Sostenibilidad</b>	Cualidad de sostenible, especialmente las características del desarrollo que asegura las necesidades del presente sin comprometer las necesidades de futuras generaciones.
<b>Requisito</b>	Condición necesaria para cumplir las expectativas de algo.
<b>Base de datos</b>	Una base de datos es una colección de información organizada de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que necesite. Las bases de datos tradicionales se organizan por campos, registros y archivos.
<b>Query/Consulta</b>	Un query en base de datos es una búsqueda o pedido de datos almacenados en una base de datos. En forma genérica, query también puede tratarse de una inserción, actualización, búsqueda y/o eliminación en una base de datos.
<b>MYSQL</b>	Sistema de gestión de bases de datos relacional desarrollado bajo licencia libre.
<b>IDE</b>	Aplicación informática que proporciona servicios integrales para facilitar el desarrollo de software.
<b>Netbeans</b>	Entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java.
<b>Java</b>	Entorno o plataforma de computación, capaz de ejecutar aplicaciones desarrolladas usando el lenguaje de programación Java

## 4.2 Acrónimos y abreviaturas

Tabla 2. Acrónimo y abreviaturas

<b>Acrónimo/Abreviatura</b>	<b>Significado</b>
<b>SOA</b>	Service Oriented Architecture (Arquitectura Orientada a Servicios).
<b>APK</b>	Android Package Kit (Formato de archivo de aplicaciones)

	Android).
<b>HTTP / HTTPS</b>	Hypertext Transfer Protocol / Secure (Protocolo de transporte web).
<b>CLI</b>	Command Line Interface (Interfaz de Línea de Comandos).
<b>SQL</b>	Structured Query Language (Lenguaje de Consulta Estructurado).
<b>DAO</b>	Data Access Object (Patrón de diseño para aislar la capa de datos).
<b>DTO</b>	Data Transfer Object (Objeto simple para transferir datos entre procesos).
<b>GCP</b>	Google Cloud Platform.
<b>Usuario</b>	Persona o cliente que usará aplicación WEB.
<b>ERS o SRS</b>	Especificación de requisitos de software, Software Requirements Specification
<b>API</b>	Application Programming Interface
<b>RF</b>	Requisito funcional
<b>RNF</b>	Requisito no funcional.
<b>DB</b>	Database (Base de Datos)
<b>CRUD</b>	Operaciones de base de datos (Create, Read, Update, Delete – Crear, Leer, Modificar, Eliminar)
<b>IDE</b>	Integrated Development Environment
<b>ECUD</b>	Especificación de Casos de Uso Detallado
<b>ROI</b>	Retorno de la inversión
<b>JSON</b>	JavaScript Object Notation.

## 5. Referencias

*Tabla 3. Referencias*

Referen	Título	Ruta	Fecha	Autor
---------	--------	------	-------	-------

Bibliografía					
1.	Standard IEEE 830-1998	Recommended Practice for Software Requirements Specifications. <a href="http://www.qualitatis.org">http://www.qualitatis.org</a>	1988	IEEE Computer Society	
2.	Video Tutorial Base	WEB SERVICE REST PARTE 01 (Base lógica del negocio) - <a href="https://www.youtube.com/watch?v=N6jRpOdR9IQ">https://www.youtube.com/watch?v=N6jRpOdR9IQ</a>	2017	Desarrolla Software	
3.	Manual de Usuario NetBeans	Documentación oficial del IDE NetBeans para desarrollo Java Web.	2020	Apache Software Foundation	
4.	Documentación Google Cloud	Cloud SQL for MariaDB / Networking in GCP.	2024	Google	
5.	W3C SOAP Specification	Simple Object Access Protocol (SOAP) 1.2.	2007	W3C	

## 6. Visión General del Documento

Este documento proporciona una descripción completa y detallada del sistema "Eureka Bank". Se estructura en tres módulos principales:

- Introducción y Contexto: Define el propósito, alcance y terminología (Secciones 1-5).
- Descripción General: Explica la arquitectura del sistema, los usuarios involucrados y el entorno operativo distribuido (Sección 7).
- Especificación Técnica: Detalla minuciosamente los requisitos funcionales (Casos de Uso), no funcionales (Rendimiento, Usabilidad) y restricciones de diseño para las cuatro plataformas cliente y el backend (Secciones 8-9).

## 7. Descripción General

El sistema "Eureka Bank" es una solución integral diseñada bajo el paradigma de Arquitectura Orientada a Servicios (SOA). Su núcleo es un servidor centralizado que expone lógica de negocio bancaria a través de Web Services SOAP. Esta arquitectura permite desacoplar la lógica de las interfaces, facilitando que múltiples canales (Web, Móvil, Escritorio, Consola) consuman los mismos datos y reglas de negocio, garantizando

consistencia y seguridad en todas las transacciones. Su función principal es actuar como un gestor inteligente de peticiones bancarias, recibiendo solicitudes desde diversos puntos de contacto y procesándolas de manera segura contra una base de datos alojada en la nube. El sistema no se limita a ser un simple repositorio de datos, sino que incorpora reglas de negocio activas que garantizan la integridad financiera de cada operación antes de que esta sea confirmada. El despliegue en nube asegura disponibilidad 24/7.

## **7.1 Perspectiva del Producto**

Desde una perspectiva arquitectónica, el sistema funciona como un *Middleware* transaccional que interconecta la infraestructura de datos con las interfaces de usuario. A diferencia de sistemas monolíticos tradicionales, esta solución desacopla completamente la lógica financiera de la presentación visual. Esto significa que el "producto" es, en esencia, un motor de procesamiento bancario capaz de interpretar intenciones de negocio, como mover dinero o consultar un saldo, y traducirlas en operaciones atómicas de base de datos. El sistema opera en un entorno distribuido, donde el procesamiento de las reglas de negocio ocurre en un servidor de aplicaciones Java, mientras que la persistencia de los datos reside en una instancia de alta disponibilidad en Google Cloud Platform. Esta separación garantiza que, independientemente del canal que utilice el cliente (Web, Móvil o Escritorio), las reglas de validación y seguridad sean idénticas y consistentes. El sistema no es un producto aislado, sino un ecosistema compuesto por cinco nodos interconectados:

- Servidor de Aplicaciones (Backend): Nodo central que ejecuta el código Java, procesa XML y gestiona transacciones.
- Servidor de Datos (Cloud): Nodo de persistencia en Google Cloud Platform (MariaDB) que almacena la información.
- Cliente Escritorio (Desktop): Nodo cliente en la red interna del banco (LAN), usado por cajeros para alta transaccionalidad.
- Cliente Web / Móvil: Nodos clientes externos (Internet), usados por clientes y gerentes para consultas.
- Cliente Consola: Nodo de administración y diagnóstico para el equipo de TI.

## **7.2 Funciones del Producto (software)**

La funcionalidad del sistema gira en torno a la gestión segura y eficiente de los activos financieros de los clientes, estructurándose en cuatro grandes capacidades operativas que definen el comportamiento del software.

En primer lugar, el sistema incorpora un módulo estricto de Control de Acceso y Autenticación (Login). Antes de permitir cualquier operación, el software tiene la capacidad de verificar la identidad digital de los empleados y usuarios. Esta función no solo compara credenciales, sino que establece una sesión segura que acompañará al usuario durante su interacción, garantizando que cada transacción posterior pueda ser auditada y vinculada a un responsable específico.

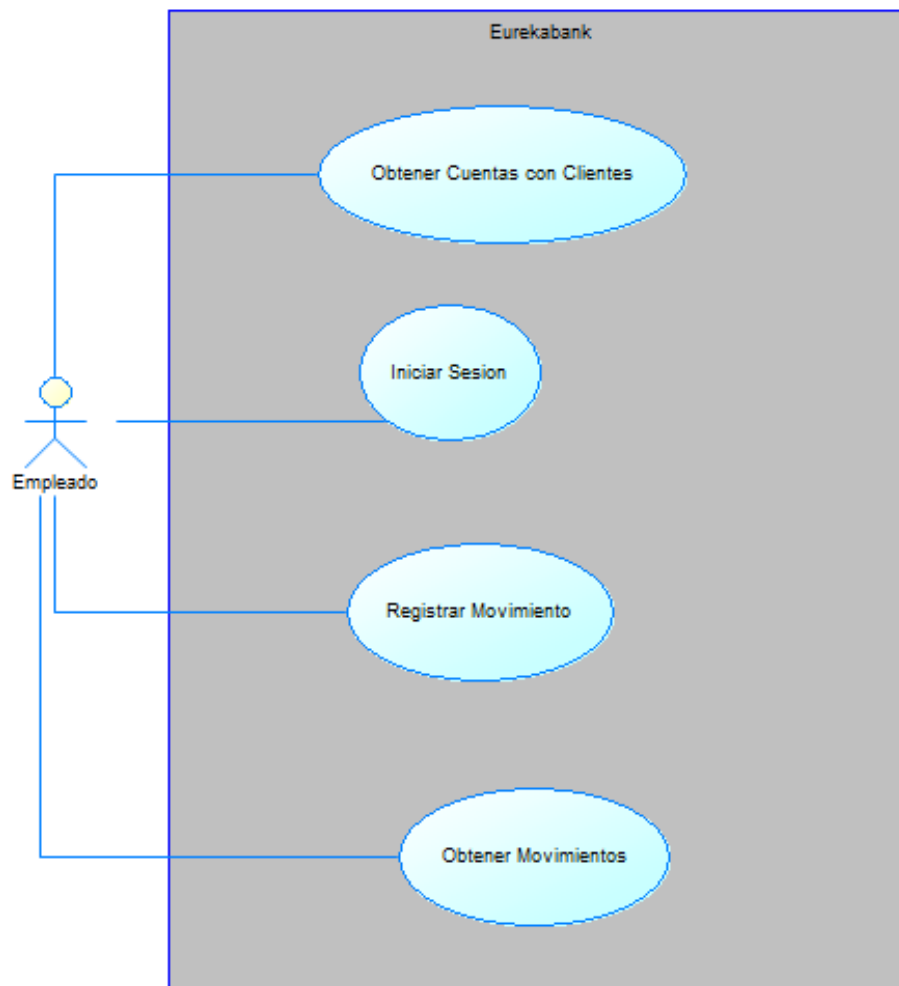
En segundo lugar, el sistema provee una capacidad de Visibilidad Financiera Consolidada

(Obtener Cuentas). Esta función permite a los gestores y gerentes obtener una radiografía completa de los productos bancarios asociados a una sucursal. El software es capaz de recuperar no solo los números de cuenta y sus saldos actuales, sino que cruza esta información en tiempo real con los datos maestros de los clientes —como nombres, cédulas y estado— para presentar un reporte integral. Es importante destacar que el sistema trata la información del cliente como datos de solo lectura; es decir, consume la información existente para darle contexto a las cuentas, pero no permite la modificación o creación de nuevos clientes como parte de este alcance funcional.

En tercer lugar, el software ofrece una función de Auditoría Transaccional Detallada (Obtener Movimientos). Esta funcionalidad permite reconstruir la historia financiera de una cuenta específica. Al invocar esta función, el sistema recupera cronológicamente cada operación que ha afectado el saldo, presentando detalles críticos como la fecha, la hora exacta, el monto involucrado y el tipo de operación. Esta capacidad es fundamental para la conciliación bancaria y para brindar transparencia al cliente final sobre el uso de sus fondos.

Finalmente, la función más crítica del sistema es el Motor de Procesamiento de Transacciones (Registrar Movimiento). Esta capacidad habilita al software para alterar los saldos de las cuentas mediante tres operaciones fundamentales: Depósitos, Retiros y Transferencias. El sistema no solo registra el dato, sino que ejecuta validaciones complejas en tiempo real, como verificar la existencia de fondos suficientes antes de un retiro o asegurar que una transferencia debite de una cuenta y acredite en otra de manera simultánea e indivisible. Esta funcionalidad es la encargada de mantener la consistencia contable del banco, asegurando que el dinero nunca se pierda ni se duplique ante fallos técnicos.

A continuación, casos de uso propuestos. Ilustración 1. Diagrama de casos de uso del sistema (incluye actores: Administrador, Cliente, resumidos como empleados; casos: Login, Obtener Cuentas, Obtener Movimientos, Registrar Movimiento).



*Ilustración 1. Diagrama de casos de uso del sistema*

### 7.3 Condiciones del Entorno

El software operará en un entorno web, móvil, escritorio y de consola, en los cuales existirán módulos, los mismos que serán administrados por el usuario. El entorno operativo del sistema requiere una infraestructura de red robusta y estable. Dado que la base de datos reside en la nube de Google, es condición indispensable que el servidor de aplicaciones mantenga una conexión a Internet de baja latencia y alta disponibilidad. Cualquier interrupción en la conectividad activará mecanismos de protección que revertirán las transacciones en curso para evitar inconsistencias. Asimismo, el entorno debe soportar protocolos de seguridad para proteger los datos sensibles mientras viajan desde las terminales de los cajeros o dispositivos móviles hasta el núcleo de procesamiento en la nube.

### 7.4 Características de los Usuarios

Los usuarios forman parte del personal de “taller de mantenimiento de vehículos”, el nivel de manejo de sistemas informáticos orientados a la web debe ser intermedio.

Tabla 4. Usuario Administrador de Sistemas

Tipo de usuario	Administrador de Sistemas
Formación	Ingeniero de Sistemas, Tecnólogo en Informática o carreras afines.
Habilidades	Conocimiento avanzado en administración de servidores Linux, bases de datos SQL, redes y seguridad informática. Capacidad para ejecutar scripts y comandos en terminal.
Actividades	Responsable de la gestión técnica de la plataforma. Utiliza principalmente el <b>Cliente Consola</b> para verificar la conectividad con Google Cloud, monitorear los logs de transacciones del servidor de aplicaciones (GlassFish/Tomcat) y realizar mantenimientos de la base de datos MariaDB. Gestiona los accesos de nivel superior.

Tabla 5. Usuario Gerente

Tipo de usuario	Gerente de Sucursal
Formación	Licenciatura en Administración de Empresas, Finanzas, Economía o Contabilidad.
Habilidades	Capacidad analítica para la interpretación de reportes financieros. Manejo intermedio de navegadores web y herramientas de ofimática. Visión estratégica del negocio.
Actividades	Utiliza el <b>Cliente Web</b> para supervisar la operatividad de la sucursal. Su función principal es consultar reportes consolidados de las cuentas activas y la información de los clientes asociados (Caso de Uso 1.1), con el fin de auditar el rendimiento y detectar anomalías en los saldos.

Tabla 6. Usuario Cajero

Tipo de usuario	Cajero de Ventanilla
Formación	Bachiller, Técnico Bancario o estudios universitarios en curso.
Habilidades	Alta velocidad y precisión en la digitación numérica. Conocimiento de procedimientos de caja, conteo de efectivo y validación de identidad. Manejo ágil de aplicaciones de escritorio (ventanas, atajos de teclado).
Actividades	Operador principal del sistema transaccional. Utiliza el <b>Cliente de Escritorio (Java Swing)</b> para procesar depósitos, retiros y transferencias en tiempo real (Caso de Uso 1.3). Es responsable de la entrada y salida física de dinero, validando los fondos en el sistema antes de entregar efectivo.

Tabla 7. Usuario Cliente

Tipo de usuario	Cliente Final
Formación	No requerida (Público en general).
Habilidades	Uso básico de dispositivos móviles inteligentes (Smartphones) y navegación en aplicaciones Android.
Actividades	Utiliza la <b>Aplicación Móvil</b> para realizar consultas de auto-servicio. Accede para verificar su saldo disponible y revisar el historial cronológico de sus últimos movimientos (Caso de Uso 1.2), sin necesidad de acudir a una agencia física.

## 7.5 Interfaces Externas

El sistema interactúa con el mundo exterior principalmente a través de su interfaz de servicios web SOAP. Esta interfaz actúa como un contrato estricto que define cómo deben enviarse las solicitudes y cómo se recibirán las respuestas, utilizando XML como lenguaje común. Adicionalmente, el sistema mantiene una interfaz de datos persistente mediante JDBC con el motor de base de datos MariaDB. Esta conexión es transparente para los usuarios y es gestionada íntegramente por el backend, encargándose de traducir los objetos de negocio Java en sentencias SQL optimizadas para la nube.

## 7.6 Restricciones

El desarrollo y operación del sistema están sujetos a las siguientes limitantes técnicas y de negocio:

- Hardware Cliente: Las PCs de los cajeros deben contar con al menos 4GB de RAM para ejecutar la JVM sin lentitud. Los móviles deben tener Android 8.0 o superior.
- Conectividad: Es mandatorio que el Servidor de Aplicaciones tenga salida a Internet estable para alcanzar la nube de Google. La latencia no debe superar los 200ms para garantizar la fluidez en caja.
- Tecnología Backend: El núcleo debe ser desarrollado exclusivamente en Java (JDK 1.8) utilizando JAX-WS para SOAP. No se permite el uso de frameworks REST.
- Base de Datos: El motor debe ser MariaDB versión 10.x alojado en Google Cloud. No se permite almacenamiento local en el servidor de aplicaciones por seguridad.
- Transaccionalidad: El sistema no debe permitir bajo ninguna circunstancia operaciones que generen inconsistencia (pérdida de dinero), obligando el uso de transacciones ACID.
- Hardware: Computador Core i5, 8GB de memoria RAM.
- Se trabaja con el sistema operativo Windows 11 tanto Home como Pro y Linux, distribuciones basadas en Debian.
- Navegador Web



- Se utilizará el lenguaje de programación Java.
- Para la elaboración del modelo conceptual, lógico y físico de la base de datos, se hará uso de la herramienta Power Designer.
- Los perfiles de los usuarios limitarán el uso de los módulos del sistema.
- La interfaz gráfica deberá ser amigable para el usuario y sencilla de utilizar.
- El sistema debe tener una conexión a internet estable que permita realizar transacciones sobre backend.

## 7.7 Suposiciones y Dependencias

En caso de que el sistema de “Eureka Bank” no cuente con el hardware detallado en los requisitos, existirá dos posibles soluciones:

- El personal del banco realiza la inversión para la adquisición del hardware necesario.
- Los requisitos cambiarán para ajustarse a la necesidad del cliente.

Si el cliente necesita realizar un cambio en algunos de los requisitos expuestos en el documento, se lo realizará con la anticipación de 7 días laborales; consecuentemente el requisito afectado tendrá un cambio tanto en el desarrollo como en el documento.

Se asume que la infraestructura de red del banco cuenta con las medidas de seguridad perimetral (Firewalls) adecuadas. Se depende directamente de la disponibilidad del servicio de Google Cloud Platform; una caída en la región de la nube afectará la operatividad total. Asimismo, se asume que los dispositivos móviles de los clientes cuentan con acceso a datos o WiFi para conectarse al servicio. Se depende de la librería mysql-connector-java o mariadb-java-client para la conectividad JDBC.

Otro factor importante por considerar es el sistema operativo con el que cuenta las computadoras donde se implementará el sistema, debido a que, si no es lo suficientemente bueno o estable, los requisitos sufrirán un cambio para posterior análisis de cómo solucionar esta problemática, o a su vez si los requisitos sufrirán un cambio y así poder acoplar al sistema operativo en funcionamiento. La conexión a internet es una parte esencial dentro del sistema a desarrollar, por ello es por lo que, si el cliente no cuenta con una conexión buena y estable, los requisitos sufrirán un cambio para ajustarse a lo que cuente la empresa.

## 8. Especificación de Requerimientos

### 8.1 Requisitos comunes de las interfaces

#### 8.1.1 Interfaces de usuario

El sistema, al ser multi-plataforma, presenta interfaces adaptadas a cada contexto. El entorno Web debe ser limpio y responsivo. El entorno de Escritorio debe priorizar la eficiencia con uso de teclado. El entorno Móvil debe ser táctil e intuitivo. El entorno de Consola debe ser textual y directo. Todas deben compartir la misma lógica de negocio provista por el backend.

#### 8.1.2 Interfaces de hardware

*Tabla 8. Requisitos de interfaces de hardware*

Nombre	Detalle	Marca	Características	Descripción	Precio
<b>Servidor de Aplicaciones</b>	Nodo central de procesamiento	Virtual (Cloud) o Físico (Dell/HP)	<ul style="list-style-type: none"> <li>- CPU: 4 vCores</li> <li>- RAM: 8 GB</li> <li>- Disco: SSD 50GB</li> </ul>	Ejecuta el contenedor de Servlets (Tomcat) y procesa las peticiones XML SOAP. Debe soportar alta concurrencia.	\$0
<b>Portátil</b>	Principal dispositivo de salida (interfaz), que muestra datos o información. Útil en caso de desarrollo y de cliente	Genérico / Lenovo / HP	<ul style="list-style-type: none"> <li>- CPU: Core i3 o sup.</li> <li>- RAM: 4GB min.</li> <li>- Teclado numérico</li> </ul>	Hardware donde se ejecuta el Cliente Escritorio (Swing). Debe permitir la digitación ágil de montos.	\$500 - \$900
<b>Monitor</b>	Principal dispositivo de salida (interfaz), que muestra datos o información. Es la Estación de trabajo cliente.	Acer	<ul style="list-style-type: none"> <li>- 24"</li> <li>- Resolución: Full HD (1920 x 1080)</li> <li>- Relación de aspecto: 16:9 Tiempo de Respuesta: 5 ms</li> <li>- Frecuencia de actualización: 60 Hz.</li> <li>- Colores Admitidos: 16.7 millones</li> <li>- Relación de Contraste: 100,000,000:1</li> <li>- Brillo: 250 cd/m<sup>2</sup></li> </ul>	El software deberá mostrar información al usuario a través de la pantalla.	\$116
<b>Mouse</b>	Dispositivo apuntador utilizado para facilitar el manejo de un entorno gráfico en una computadora.	Genius	Ratón óptico con sensor de 1000dpi.	El software debe interactuar con el movimiento del mouse y sus botones.	\$6
<b>Teclado</b>	Dispositivo o periférico de entrada	Genius	Estándar de teclas de 104/105/106 Puerto USB: Si Soporte Sistema Windows10 /Windows@7/Vista/XP Soporte Interfaz USB	El software deberá interactuar con las pulsaciones del teclado. Con el teclado el usuario digitara las peticiones que requiera.	\$10
<b>CPU</b>	La unidad central de procesamiento es el hardware dentro de un ordenador u	Acer	Procesador: Intel Core i5 2da Gen o superior. compatibles con un sistema operativo como Windows 10, Linux	En buen estado, el cual poseerá todos los controladores para poder manejar cada periférico de	\$399

	otros dispositivos		(...), Memoria Ram: 8GB	entrada y salida.	
<b>Dispositivo Móvil</b>	Terminal del cliente final	Genérico / Samsung / Xiaomi / Motorola	- Pantalla: 5"+ - RAM: 2GB -Conectividad 4G/WiFi	Dispositivo Android donde corre la App para consultas de saldo y movimientos.	\$150 - \$ 800
<b>Servidor de Base de Datos</b>	Instancia de Persistencia	Google Cloud Platform	-Conectividad 4G/WiFi - Tipo: e2-standard-2 - Almacenamiento: Persistente SSD	Instancia en la nube dedicada exclusivamente a ejecutar el motor MariaDB y almacenar la data financiera.	\$ --

### 8.1.3 Interfaces de software

Tabla 9. Interfaces de software

	Detalles	Definición	Propósito
<b>Sistema Operativo Servidor</b>	Nombre: Linux Ubuntu Server N.º Especificación: x86_64 N.º Versión: 20.04 LTS Fuente: ubuntu.com Distribución: Open Source Precio: Gratuito	Sistema operativo tipo Unix basado en Debian, diseñado para servidores y entornos de nube, conocido por su estabilidad y seguridad.	Proveer la plataforma base para ejecutar el servidor de aplicaciones y gestionar las conexiones de red con Google Cloud.
<b>Sistema Operativo Cliente</b>	Nombre: Microsoft Windows N.º Especificación: NT Kernel N.º Versión: 10 Pro / 11 Fuente: microsoft.com Distribución: Comercial Precio: ~\$199 USD	Sistema operativo con interfaz gráfica desarrollado por Microsoft, estándar en entornos corporativos y estaciones de trabajo.	Servir de entorno para las PC de los cajeros, ejecutando la máquina virtual de Java y soportando los periféricos bancarios.
<b>Sistema Operativo Móvil</b>	Nombre: Android N.º Especificación: API Level 26+ N.º Versión: 8.0 a 14.0 Fuente: android.com Distribución: Open Source Precio: Gratuito	Sistema operativo móvil basado en el núcleo Linux, diseñado principalmente para dispositivos con pantalla táctil como teléfonos inteligentes.	Permitir la ejecución de la App móvil de Eureka Bank en los dispositivos personales de los clientes para consultas remotas.
<b>Base de Datos</b>	Nombre: MariaDB Server N.º Especificación: SQL:2016	Sistema de gestión de bases de datos relacional derivado de MySQL, que	Almacenar de forma persistente y segura toda la información

	<p>N.º Versión: 10.5 Series</p> <p>Fuente: mariadb.org</p> <p>Distribución: GPL v2</p> <p>Precio: Gratuito</p>	garantiza la integridad de los datos y el soporte ACID.	transaccional (cuentas, clientes, movimientos) en la nube.
<b>Lenguaje de Programación</b>	<p>Nombre: Java JDK</p> <p>N.º Especificación: Java SE 8</p> <p>N.º Versión: 1.8.0_202+</p> <p>Fuente: oracle.com</p> <p>Distribución: Oracle/GPL</p> <p>Precio: Gratuito</p>	Lenguaje de programación orientado a objetos, concurrente y basado en clases, diseñado para tener la menor cantidad de dependencias posible.	Desarrollo unificado de la lógica del Backend (SOAP), y de los clientes de Escritorio, Móvil y Consola.
<b>Entorno de Desarrollo</b>	<p>Nombre: Apache NetBeans</p> <p>N.º Especificación: IDE Platform</p> <p>N.º Versión: 12.0 LTS</p> <p>Fuente: netbeans.apache.org</p> <p>Distribución: Apache 2.0</p> <p>Precio: Gratuito</p>	Entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java.	Herramienta para escribir código, diseñar interfaces gráficas, depurar errores y generar los ejecutables del proyecto.
<b>Herramienta de desarrollo</b>	<p>Nombre: NetBeans</p> <p>Número de especificación: 8</p> <p>Número de versión: 8.0.2</p> <p>Fuente: <a href="https://www.netbeanside.com/">https://www.netbeanside.com/</a></p> <p>Distribución: libre</p>	Diseñar y generar interfaces y el funcionamiento del programa mediante el lenguaje de programación Java.	Herramienta que se utiliza para desarrollar aplicaciones Web, Móvil y de Escritorio para diferentes lenguajes de programación como son Java, C++, Ruby y PHP entre otros
<b>Framework SOAP</b>	<p>Nombre: JAX-WS (Metro)</p> <p>N.º Especificación: JAX-WS 2.3</p> <p>N.º Versión: 2.3.1</p> <p>Fuente: javaee.github.io</p> <p>Distribución: CDDL/GPL</p> <p>Precio: Gratuito</p>	API de Java para crear servicios web XML (SOAP), facilitando el marshalling y unmarshalling de objetos.	Generar automáticamente los contratos WSDL y gestionar el intercambio de mensajes XML entre cliente y servidor.
<b>Diseño de modelo de base de datos</b>	<p>Nombre: Power Designer</p> <p>Desarrollador: SAP</p> <p>Version: 16.5</p> <p>Distribución: Pagada</p> <p>Precio: 99.98 \$</p>	Es una herramienta para el análisis, diseño inteligente y construcción sólida de una base de datos y un desarrollo orientado a modelos de datos a nivel físico y conceptual, que da	El propósito de este programa en el proyecto es diseñar un modelo relacional de base de datos

#### 8.1.4 Interfaces de comunicación

El sistema utilizará el protocolo HTTP sobre TCP/IP para transportar los mensajes SOAP. El servidor escuchará peticiones en el puerto 8080 (o 443 para SSL). La comunicación con la base de datos en la nube se realizará a través del puerto estándar 3306, utilizando sockets seguros.

**Extracción de datos:** Se obtiene la información de fuentes de origen creadas previamente como script SQL, posterior a ello, de los distintos clientes existentes de los movimientos realizados en base a las cuentas existentes. Durante la extracción, en los sistemas de bases de datos MariaDB se identifica cada movimiento y se lo extrae.

**Carga:** se escribe los datos en la base de datos. La fase de carga es el momento en el cual los datos de la fase anterior (transformación) son cargados en el sistema de destino, considerando que esta se realiza desde la nube, de Cloud Plataform.

### 8.2 Requisitos Funcionales

#### 8.2.1. Objetivo General

El Sistema "Eureka Bank" permitirá la gestión centralizada y segura de las operaciones bancarias (depósitos, retiros, consultas) a través de múltiples canales de atención, garantizando la integridad de los datos en la nube.

#### 8.2.2. Objetivos Específicos

A continuación, se muestran los objetivos específicos en el que se sustenta el proyecto de desarrollo de software.

Tabla 10. Autenticación y Seguridad (Login)

OBJ-001	Controlar el Acceso al Sistema
Descripción	Asegurar que solo personal autorizado (Empleados) y clientes validados puedan acceder a la información financiera, mediante un proceso de login único centralizado.
Importancia	Alta
Comentarios	

Tabla 11. Visibilidad Operativa (Cuentas)

OBJ-002	Centralizar Información de Cuentas
Descripción	Permitir a la gerencia obtener reportes inmediatos del estado de todas las cuentas de una sucursal, vinculando los datos financieros con la identidad del cliente.
Importancia	Alta

<b>Comentarios</b>	
--------------------	--

Tabla 12. Auditoría (Movimientos)

OBJ-003	Proveer Historial Transaccional
Descripción	Garantizar que cada movimiento quede registrado y sea consultable cronológicamente, brindando transparencia total al cliente sobre el uso de su dinero.
Importancia	Alta
Comentarios	

Tabla 13. Integridad Transaccional (Registro)

OBJ-004	Ejecutar Transacciones Seguras
Descripción	Procesar depósitos y retiros aplicando reglas de validación de fondos y bloqueos de base de datos (ACID) para evitar pérdidas o inconsistencias financieras.
Importancia	Crítica
Comentarios	

### 8.2.3. Actores

Tabla 14. Actor Cajero (Escritorio)

ACT-001	Cajero
Descripción	Empleado del banco encargado de operar la aplicación de Escritorio para ejecutar transacciones monetarias físicas (ingresos y egresos).
Comentarios	

Tabla 15. Actor Gerente (Web)

ACT-002	Gerente de Sucursal
Descripción	Supervisor encargado de monitorear el rendimiento de la sucursal y consultar listados de clientes y cuentas vía Web.
Comentarios	

Tabla 16. Actor Administrador (Consola)

ACT-003	Administrador IT
Descripción	Personal técnico que utiliza la consola para verificar la

	salud del sistema y realizar pruebas de conectividad.
<b>Comentarios</b>	

Tabla 17. Actor Cliente (Móvil)

<b>ACT-004</b>	<b>Cliente Final</b>
<b>Descripción</b>	Usuario propietario de la cuenta que utiliza la App Móvil para consultar su saldo y movimientos personales.
<b>Comentarios</b>	

#### 8.2.4. Lista de requisitos funcionales

A continuación, se presenta una lista con los requisitos funcionales del sistema “Eureka Bank. Considerar que por el momento, se tendrá un superadmin que realice todos los movimientos.

Tabla 18. Lista de requisitos funcionales

<b>Servicio de Eureka Bank</b>			
<i>Lista de requerimientos funcionales</i>			
<i>Código</i>	<i>Requerimiento</i>	<i>Caso de uso</i>	<i>Actor</i>
<i>RF – 01</i>	El sistema debe validar las credenciales de usuario (Usuario/Clave) contra la base de datos y retornar el perfil del empleado.	Login (CU 1.4)	Todos
<i>RF – 02</i>	El sistema debe permitir consultar todas las cuentas de una sucursal específica, mostrando saldo, moneda y nombre del cliente titular.	Obtener Cuentas (CU 1.1)	Administrador, Cliente
<i>RF – 03</i>	El sistema debe permitir consultar el historial detallado de movimientos de una cuenta, ordenados por fecha descendente.	Obtener Movimientos (CU 1.2)	Cliente / Cajero / Administrador
<i>RF – 04</i>	El sistema debe permitir registrar depósitos, retiros y transferencias, validando saldo suficiente y ejecutando la operación de forma atómica.	Registrar Movimiento (CU 1.3)	Cajero / Cliente

### 8.2.5 Requisitos Funcionales

Para el Sistema "Eureka Bank se ha identificado los siguientes requerimientos funcionales.

Tabla 19. Gestionar Login

<b>RQF-001</b>	<b>Gestionar Login</b>
<b>Descripción</b>	El sistema debe permitir recibir un usuario y contraseña desde cualquier cliente (Web, Móvil, Escritorio). Debe conectarse a la BD MariaDB, buscar en la tabla Empleado y verificar la coincidencia. Si es exitoso, retorna el objeto Empleado con su código de sucursal.
<b>Objetivo</b>	OBJ-001
<b>Importancia</b>	Alta
<b>Estado</b>	Aprobado
<b>Estabilidad</b>	Alta
<b>Comentarios</b>	

Tabla 20. Obtener Cuentas con sus Clientes

<b>RQF-002</b>	<b>Obtener Cuentas con Clientes</b>
<b>Descripción</b>	El sistema debe recibir un código de sucursal. Debe ejecutar una consulta SQL con INNER JOIN entre Cuenta y Cliente. Debe retornar una lista de objetos que contengan: Código de Cuenta, Saldo, Estado, Código de Cliente, Nombre y Apellido del Cliente.
<b>Objetivo</b>	OBJ-002
<b>Importancia</b>	Alta
<b>Estado</b>	Aprobado
<b>Estabilidad</b>	Alta
<b>Comentarios</b>	Solo lectura. No permite modificar datos del cliente.

Tabla 21. Obtener Movimientos de una Cuenta

<b>RQF-003</b>	<b>Obtener Movimientos</b>
<b>Descripción</b>	El sistema debe recibir un número de cuenta. Primero valida su existencia. Luego consulta la tabla Movimiento filtrando por la cuenta y ordenando por fecha reciente. Retorna una lista con: Número de Movimiento, Fecha, Tipo (Depósito/Retiro), Monto y Empleado responsable.
<b>Objetivo</b>	OBJ-003
<b>Importancia</b>	Alta
<b>Estado</b>	Aprobado
<b>Estabilidad</b>	Alta
<b>Comentarios</b>	



Tabla 22. Registrar Movimiento

RQF-004	Registrar Movimiento (Depósito/Retiro)
<b>Descripción</b>	<p>El sistema debe gestionar la información del movimiento realizado, mostrar saldo actual, que tipo de movimiento será. El sistema recibe Cuenta, Monto y Tipo.</p> <ol style="list-style-type: none"> <li>1. Inicia Transacción BD.</li> <li>2. Bloquea registro (SELECT FOR UPDATE).</li> <li>3. Valida Estado Activo.</li> <li>4. Si es Retiro, valida Saldo <math>\geq</math> Monto.</li> <li>5. Actualiza Saldo (UPDATE).</li> <li>6. Inserta Movimiento (INSERT).</li> <li>7. COMMIT.</li> </ol> <p>Si algo falla, hace ROLLBACK.</p>
<b>Objetivo</b>	OBJ-004
<b>Importancia</b>	Crítica
<b>Estado</b>	Aprobado
<b>Estabilidad</b>	Alta
<b>Comentarios</b>	Manejo estricto de excepciones ACID.

## 8.3 Requisitos No Funcionales

### 8.3.1 Objetivo General

Para el Sistema de “Eureka Bank” se ha identificado los siguientes requerimientos no funcionales en relación con la tecnología de la información, siendo el objetivo principal garantizar que el sistema Eureka Bank opere con altos estándares de calidad técnica.

Tabla 23. Requerimiento no funcional Tiempo de respuesta

RNF-001	Tiempo de Respuesta
<b>Descripción</b>	El sistema debe procesar transacciones críticas (Depósitos/Retiros) en menos de 3 segundos, incluyendo la latencia de red a Google Cloud. Las consultas deben responder en menos de 1 segundo.
<b>Importancia</b>	Alta
<b>Estado</b>	Aprobado

Tabla 24. Requerimiento no funcional utilización de colores

RNF-002	Utilización de Colores
<b>Descripción</b>	Las interfaces gráficas (Web, Móvil, Escritorio) deben usar una paleta consistente que no supere una gama de más de tres colores o que sean monocromáticos o del círculo cromático. Los montos

	negativos o de retiro deben resaltarse en Rojo.
<b>Importancia</b>	Media
<b>Estado</b>	Aprobado

Tabla 25. Requerimiento no funcional ícono de operaciones

<b>RNF-003</b>	<b>Íconos e Identidad</b>
<b>Descripción</b>	Uso de íconos estándar para acciones comunes (Lupa para buscar, Disco para guardar) en todas las plataformas para facilitar la usabilidad.
<b>Importancia</b>	Media
<b>Estado</b>	Aprobado

Tabla 26. Requerimiento no funcional métodos de acceso

<b>RNF-004</b>	<b>Métodos de Acceso</b>
<b>Descripción</b>	El acceso al sistema requiere obligatoriamente autenticación por usuario y contraseña. No se permite acceso anónimo a ningún servicio.
<b>Importancia</b>	Alta
<b>Estado</b>	Aprobado

Tabla 27. Requerimiento no funcional plataforma

<b>RNF-005</b>	<b>Plataforma Estándar</b>
<b>Descripción</b>	El backend debe correr sobre la Máquina Virtual de Java (JVM) y la base de datos debe ser relacional compatible con SQL estándar (MariaDB).
<b>Importancia</b>	Alta
<b>Estado</b>	Aprobado

Tabla 28. Requerimiento no funciona accesibilidad

<b>RNF-006</b>	<b>Accesibilidad</b>
<b>Descripción</b>	Las interfaces deben ser claras y permitir la navegación intuitiva. La App Móvil debe soportar fuentes escalables.
<b>Importancia</b>	Media
<b>Estado</b>	Aprobado

Tabla 29. *Requerimiento no funcional mantenimiento*

RNF-007	Documentación y Mantenimiento
<b>Descripción</b>	El sistema deberá tener un manual de usuario para facilitar el mantenimiento que se realizará en oportunidades futuras. El código debe estar documentado (Javadoc). Se debe entregar manual de despliegue para el servidor y el script SQL de creación de base de datos.
<b>Importancia</b>	Alta
<b>Estado</b>	Aprobado

- El sistema contará con un nivel de acceso, basado en autenticación usuario y contraseña para un superadministrador.
- El sistema contará con un nombre de usuario y contraseña personal
- El sistema tendrá mecanismos de respaldo de información automáticos, los cuales aumentarán la integridad de estos al usar una base de datos de Cloud.
- Emitirá mensajes de alertas cuando el usuario realice un procedimiento erróneo en la ejecución del sistema.
- Emitirá advertencias y mensajes de confirmación en el momento de registrar, modificar y eliminar detalles de los movimientos.
- Las Búsquedas se realizarán clasificándolas según: Fecha, cliente o movimiento.
- El administrador podrá consultar el estado de las cuentas con montos disponibles de cada cliente.
- El administrador puede buscar datos de todas las cuentas.

## 8.4 Otros Requerimientos

### 8.4.1 Restricciones de Diseño

Las restricciones de diseño son las siguientes:

- No sobrecargar con muchas imágenes la parte principal del usuario, limitarla a logos, e imágenes llamativas de cada tipo de movimiento, por lo demás se trabajará solamente con íconos para mantener la semiótica.
- Las imágenes deben ser en formato png de acuerdo al tipo de espacio, así que variarán los pixeles, pero no deben superar los 250 x 250px.
- Se mantenga el orden del menú principal, submenú y pantalla de operación como container del sistema.
- Formato minimalista, colores bajos que no canse la visión del usuario.
- Arquitectura SOA pura: lógica en el servidor, presentación en el cliente.
- Diseño Responsivo en cada cliente a excepción del de consola, es decir, para móvil, escritorio y web.

#### 8.4.2 Restricciones de Hardware

- El servidor debe tener redundancia de disco para evitar pérdida de datos en caso de fallo físico (manejado por GCP).
- Los equipos disponibles deben tener las siguientes características:
  - o Procesador Intel Core i5 o superior,
  - o 8 GB de RAM o superior,
  - o Disco de 250 GB o superior,
  - o Sistema Operativo Windows 10 u 11 versión Home o Pro, además puede ser Linux con distribuciones Debian.

Por este motivo es necesario procurar que el sistema se adapte a dichas capacidades, condicionado a una mejora inmediata de los mismos.

#### 8.4.3 Atributos de calidad

**Fiabilidad.** - El sistema, ofrecerá garantía de funcionar correctamente y cumplir con los requisitos presentados previamente. Integridad de datos garantizada al 100%.

**Portabilidad.** - Al tratarse de una aplicación orientada a distintos clientes, se puede ejecutar en cualquier dispositivo electrónico que posea un navegador y una conexión a internet.

**Seguridad.** - El sistema tendrá mecanismos que brindan la posibilidad de que los datos sean resguardados satisfactoriamente. Se usarán los siguientes mecanismos de acceso a los datos:

- Cada usuario tiene una cuenta, pero la cuenta general la dirige el super administrador.
- Para utilizar el sistema, es necesario introducir un usuario y una contraseña, ambos previamente creados y ejecutados por un script SQL.

**Disponibilidad.** - Los usuarios podrán acceder al sistema durante las 24 horas del día, respaldado por su base de datos en la nube.

## 9. Requisitos de Rendimiento

### 9.1. Requisitos de Interfaces Externas

#### 9.1.1. Interfaces de Usuario

Los formularios deben validar tipos de datos (números vs letras) antes de enviar la petición SOAP para reducir carga al servidor.

Las interfaces que se elaboran en las plataformas web, escritorio y móvil incluyen:

- Botones representativos para elegir las opciones de tipo de movimiento.

- Pestañas específicas para cada movimiento a manera de modales.
- Mensajes informativos que mantengan al tanto de si una operación se realizó exitosamente.
- Mensajes de error representativos en caso de que fallen los procesos, que permitan saber al usuario lo que pasó.
- Formularios para el ingreso, modificación, actualización y búsqueda de datos, para completar las operaciones CRUD.
- Considerar colores que lleven a la semiótica de las acciones, como gris o rojo para cancelar, el color principal para operaciones de éxito.
- Utilización de Hovers e ícono de cursores para dar un feed visual al usuario.

### 9.1.2. Interfaces de Hardware

**La pantalla del monitor/ teléfono.** - el software muestra información al usuario a través de la pantalla del monitor o del móvil, considerando los distintos tipos de dispositivos.

**Ratón.** - el software interactúa con el movimiento y los botones del ratón. El ratón activa las zonas de entrada de datos, botones de comando y selección de las opciones de los menús.

**Teclado.** - el software interactúa con el sistema pulsando el teclado en el momento de ingresar los datos.

**Servidor de aplicaciones.** - Debe gestionar eficientemente la memoria (Heap Size) para soportar la creación de objetos XML grandes durante reportes masivos.

### 9.1.3. Interfaces de Software

- Sistema Operativo: Compatible con Windows 10/11 (Clientes PC), Android 8-14 (Móvil), Linux (Servidor).
- Navegador Web: Chrome, Mozilla, Safari, Opera.

### 9.1.4. Interfaces de Comunicación

Los servidores y clientes se comunicarán mediante el protocolo SOAP sobre HTTP/S, utilizando puertos estándar de internet (80, 443, 8080) para facilitar el paso a través de firewalls corporativos.

### 9.1.5. Base de Datos

El sistema dispondrá de comunicación persistente y segura a la base de datos "EurekaBank" en Google Cloud, utilizando un pool de conexiones JDBC optimizado para soportar múltiples hilos de ejecución simultáneos.

## RECOMENDACIONES

- Se recomienda implementar un certificado SSL en el servidor de aplicaciones para cifrar el tráfico XML y proteger los datos sensibles de los clientes.
- Es importante realizar pruebas de carga (Stress Testing) sobre los métodos SOAP antes del lanzamiento para calibrar el pool de conexiones a la base de datos en la nube.
- Se sugiere desarrollar librerías cliente (Stubs) compartidas para los proyectos de Consola y Escritorio, evitando la duplicidad de código en la generación de peticiones SOAP.

## CONCLUSIONES

- El modelo del sistema Eureka Bank cumple con los requerimientos de modernización al integrar múltiples canales (Web, Móvil, Escritorio) bajo una arquitectura centralizada y robusta.
- La especificación detallada de los requisitos funcionales transaccionales (Login, Movimientos) asegura que el equipo de desarrollo tiene una hoja de ruta clara para implementar la lógica de negocio crítica en Java.
- El uso de estándares como SOAP y SQL en la nube garantiza la interoperabilidad, escalabilidad y seguridad de la solución a largo plazo.

## BIBLIOGRAFÍA

IEEE Computer Society. (1998). Standard IEEE 830: Recommended Practice for Software Requirements Specifications. <https://ieeexplore.ieee.org/document/720574>

Desarrolla Software. (2017). Tutorial Web Services Java y Base de Datos. YouTube. <https://www.youtube.com/watch?v=N6jRpOdR9IQ>

Google Cloud. (2024). Documentación Oficial Cloud SQL. <https://cloud.google.com/sql/docs>

Oracle. (2023). Java Documentation. [https://www.w3schools.com/java/java\\_ref\\_reference.asp](https://www.w3schools.com/java/java_ref_reference.asp)