Cajero Automático Multi Moneda



Documento de Arquitectura

Raphael Muñoz S.

rmunozs@grupobisa.com

Tabla de contenido

[1. Introducción 4](#_Toc156929698)

[1.1 Objetivo del Sistema 4](#_Toc156929699)

[1.2 Alcance del Documento 4](#_Toc156929700)

[1.3 Contexto del Proyecto 4](#_Toc156929701)

[1.4 Audiencia del Documento 4](#_Toc156929702)

[2. Decisiones de Arquitectura 4](#_Toc156929703)

[2.1 Arquitectura de Microservicios: 4](#_Toc156929704)

[2.2 Gestión de Cuentas 4](#_Toc156929705)

[2.3 Gestión de Transacciones 5](#_Toc156929706)

[2.4 Gestión de Tasa de Cambio: 5](#_Toc156929707)

[2.5 Microservicios de respaldo 5](#_Toc156929708)

[2.6 Comunicación entre Microservicios: 5](#_Toc156929709)

[2.7 Registro de microservicios 5](#_Toc156929710)

[2.8 Almacenamiento de Datos: 5](#_Toc156929711)

[2.9 Seguridad: 5](#_Toc156929712)

[2.10 Manejo de Transacciones: 6](#_Toc156929713)

[2.11 Escalabilidad: 6](#_Toc156929714)

[2.12 Manejo de Errores y Resiliencia: 6](#_Toc156929715)

[2.13 Integrabilidad: 6](#_Toc156929716)

[2.14 Interfaz de Usuario: 6](#_Toc156929717)

[2.15 Cumplimiento Normativo: 6](#_Toc156929718)

[3. Atributos de Calidad 6](#_Toc156929719)

[3.1 Disponibilidad: 6](#_Toc156929720)

[3.2 Rendimiento: 6](#_Toc156929721)

[3.3 Seguridad: 7](#_Toc156929722)

[3.4 Fiabilidad: 7](#_Toc156929723)

[3.5 Mantenibilidad: 7](#_Toc156929724)

[3.6 Escalabilidad: 7](#_Toc156929725)

[3.7 Usabilidad: 7](#_Toc156929726)

[3.8 Auditoría y Trazabilidad: 7](#_Toc156929727)

[3.9 Integrabilidad: 7](#_Toc156929728)

[3.10 Cumplimiento Normativo: 7](#_Toc156929729)

[3.11 Consistencia de Datos: 8](#_Toc156929730)

[3.12 Tiempo de Respuesta del Sistema: 8](#_Toc156929731)

[4. Escenarios 8](#_Toc156929732)

[5. Vista Lógica 14](#_Toc156929733)

[6. Vista de desarrollo 16](#_Toc156929734)

[7. Vista de proceso 19](#_Toc156929735)

[8. Vista física 22](#_Toc156929736)

[9. Vista de casos de uso 25](#_Toc156929737)

# Introducción

Este documento de arquitectura tiene como objetivo proporcionar una visión general detallada de la estructura y diseño del sistema de cajero automático multimoneda, el cual será implementado utilizando microservicios en la nube de Amazon Web Services (AWS). Este sistema está diseñado para ofrecer a los usuarios la capacidad de realizar diversas operaciones financieras, incluyendo depósitos, retiros y conversiones de moneda, gestionando cuentas en dos monedas distintas: el boliviano y el dólar estadounidense.

## Objetivo del Sistema

El objetivo principal del sistema de cajero automático multimoneda es proporcionar a los usuarios una experiencia financiera eficiente y segura, permitiéndoles gestionar sus cuentas y realizar transacciones en dos monedas diferentes. La implementación a través de microservicios en AWS busca lograr una arquitectura modular, escalable y altamente disponible, asegurando la confiabilidad y el rendimiento del sistema.

## Alcance del Documento

Este documento abordará los aspectos clave de la arquitectura del sistema, incluyendo la identificación de los microservicios, la descripción de las clases relevantes, la definición de atributos de calidad, la presentación de un diagrama de despliegue y otros elementos esenciales para comprender la estructura y el funcionamiento del sistema.

## Contexto del Proyecto

El sistema se concibe como una solución integral para usuarios que desean gestionar sus finanzas en dos monedas diferentes, proporcionando una interfaz de usuario intuitiva y funcionalidades que abarcan desde la creación de cuentas hasta la realización de operaciones de cambio de divisas. La implementación en AWS se selecciona para aprovechar los servicios en la nube que facilitan la escalabilidad, la redundancia y la eficiencia operativa.

## Audiencia del Documento

Este documento está destinado a arquitectos de sistemas, desarrolladores, gerentes de proyecto y otros interesados en comprender la estructura y los principios detrás del diseño del sistema de cajero automático multimoneda en AWS. Proporcionará una base sólida para la implementación, pruebas y mantenimiento del sistema.

# Decisiones de Arquitectura

## Arquitectura de Microservicios:

Se optó por una arquitectura de microservicios para modularizar el sistema en unidades independientes, cada una enfocada en una funcionalidad específica. Los microservicios permiten el desarrollo, la implementación y el escalado independientes, lo que facilita la evolución continua del sistema.

## Gestión de Cuentas

Se define un microservicio de consulta que permita a los usuarios ver el saldo de sus cuentas, el tipo de cambio y las operaciones realizadas.

## Gestión de Transacciones

Se define un microservicio de transacción que permita a los usuarios realizar operaciones de retiro, depósito y transferencia en moneda local o extranjera.

## Gestión de Tasa de Cambio:

Las tasas de cambio se manejarán mediante un microservicio específico, utilizando Amazon DynamoDB para almacenar y actualizar las tasas. Se restringe el uso de solo dos monedas (bolivianos y dólar estadounidense) para simplificar la gestión de tasas de cambio y por las restricciones físicas de la disponibilidad de bandejas en el cajero electrónico.

## Microservicios de respaldo

Adicionalmente a los microservicios principales se define un conjunto de microservicios adicionales para tareas de respaldo a la aplicación.

* Un microservicio de autenticación que valide la identidad de los usuarios mediante el uso de una tarjeta y un PIN.
* Un microservicio de notificación que envíe mensajes de confirmación, alertas y recibos a los usuarios por correo electrónico o SMS.
* Un microservicio de administración que permita a los operadores del sistema gestionar los cajeros automáticos, los límites de operación, las comisiones y las tasas de cambio.

## Comunicación entre Microservicios:

La comunicación entre microservicios se realizará a través de Amazon API Gateway, que actúa como punto de entrada único para las solicitudes. Esto simplifica la exposición y gestión de las APIs, permitiendo una comunicación segura y eficiente entre los microservicios.

## Registro de microservicios

Para registrar y localizar los microservicios, se utilizará un patrón de descubrimiento de servicios permitiendo a los microservicios encontrar y comunicarse entre sí de forma dinámica y tolerante a fallos. Para implementar el descubrimiento de servicios, se podría utilizar Eureka, otro componente de Spring Cloud Netflix que ofrece un servidor y un cliente de registro y descubrimiento.

## Almacenamiento de Datos:

Se decidió utilizar Amazon RDS para la gestión de datos relacionales, aprovechando su capacidad de escala, gestión automática y backups periódicos. Amazon DynamoDB se seleccionó para almacenar datos NoSQL que requieren acceso rápido y escalabilidad eficiente.

## Seguridad:

La seguridad se gestionará mediante Amazon IAM para el control de accesos y roles, y la autenticación se realizará a través de estrategias robustas, como autenticación de dos factores en las operaciones críticas. Además, se implementarán prácticas de cifrado de extremo a extremo para proteger la confidencialidad de los datos.

## Manejo de Transacciones:

Se implementará un mecanismo de control de concurrencia en las transacciones para garantizar la consistencia de los datos en situaciones de concurrencia. Además, se registrarán todas las transacciones en una base de datos dedicada para permitir el seguimiento y la auditoría.

## Escalabilidad:

La arquitectura se diseñará para la escalabilidad horizontal y vertical. La implementación en AWS facilita la escalabilidad mediante la asignación de recursos adicionales según sea necesario, y se aprovechará la capacidad de autoscaling para gestionar automáticamente la carga variable.

## Manejo de Errores y Resiliencia:

Se implementarán estrategias robustas para el manejo de errores, incluyendo mecanismos de retry y recuperación ante fallas. Además, se podrían establecer alarmas en Amazon CloudWatch para detectar y abordar proactivamente problemas operativos.

## Integrabilidad:

La arquitectura se diseñará considerando la facilidad de integración con servicios externos, como proveedores de tasas de cambio. El uso de estándares y protocolos comunes facilitará la interoperabilidad.

## Interfaz de Usuario:

La interfaz de usuario se desarrollará como una aplicación web, proporcionando una experiencia intuitiva y accesible para los usuarios. Se utilizarán prácticas de desarrollo centradas en el usuario para garantizar la usabilidad y la satisfacción del cliente.

## Cumplimiento Normativo:

Se establecerá un marco de cumplimiento normativo para garantizar que el sistema cumpla con las leyes y regulaciones aplicables en el ámbito financiero y de seguridad de la información. Se realizarán auditorías regulares para evaluar y actualizar los procesos de cumplimiento.

# Atributos de Calidad

Los atributos de calidad son características no funcionales que definen la efectividad del sistema en términos de sus características operativas, de rendimiento y de comportamiento. En el diseño del sistema de cajero automático multimoneda en AWS, se han identificado y definido los siguientes atributos de calidad como fundamentales para el éxito del proyecto

## Disponibilidad:

Definición: El sistema debe estar disponible en todo momento, minimizando el tiempo de inactividad.

Objetivo: Lograr una disponibilidad del 99.9% para garantizar la accesibilidad constante a las funcionalidades del sistema.

## Rendimiento:

Definición: El sistema debe cumplir con tiempos de respuesta y tasas de procesamiento específicos.

Objetivo: Mantener tiempos de respuesta inferiores a 3 segundos para operaciones críticas, asegurando una experiencia de usuario ágil y eficiente.

## Seguridad:

Definición: Proteger la integridad, confidencialidad y disponibilidad de la información del usuario.

Objetivo: Implementar prácticas de seguridad como autenticación de dos factores, cifrado de extremo a extremo y controles de acceso estrictos.

## Fiabilidad:

Definición: El sistema debe funcionar de manera confiable y consistente a lo largo del tiempo.

Objetivo: Garantizar una alta confiabilidad mediante pruebas exhaustivas, implementación de recuperación ante fallas y monitoreo proactivo.

## Mantenibilidad:

Definición: Facilitar la implementación de cambios y actualizaciones en el sistema.

Objetivo: Adoptar prácticas de desarrollo y diseño que faciliten la mantenibilidad, como código modular, documentación detallada y pruebas automáticas.

## Escalabilidad:

Definición: Permitir que el sistema crezca para manejar aumentos en la carga sin degradación significativa del rendimiento.

Objetivo: Diseñar la arquitectura para permitir la escalabilidad horizontal y vertical, utilizando servicios de AWS como autoscaling para gestionar automáticamente la carga variable.

## Usabilidad:

Definición: La interfaz de usuario debe ser intuitiva y fácil de usar.

Objetivo: Obtener una puntuación de satisfacción del usuario superior al 90% en encuestas semestrales, mejorando continuamente la experiencia del usuario.

## Auditoría y Trazabilidad:

Definición: Registrar y permitir el seguimiento detallado de las transacciones y actividades del sistema.

Objetivo: Proporcionar informes de auditoría en menos de 24 horas, garantizando la transparencia y el cumplimiento normativo.

## Integrabilidad:

Definición: Facilitar la integración con servicios externos.

Objetivo: Implementar estándares y protocolos para la interoperabilidad, permitiendo una integración sin problemas con proveedores de tasas de cambio externos.

## Cumplimiento Normativo:

Definición: Asegurar que el sistema cumple con las regulaciones y normativas aplicables.

Objetivo: Realizar auditorías regulares para evaluar y actualizar los procesos de cumplimiento en un plazo de 30 días ante cambios normativos.

## Consistencia de Datos:

Definición: Garantizar la coherencia e integridad de los datos en todas las transacciones.

Objetivo: Implementar mecanismos de control de concurrencia y corrección de desviaciones en la consistencia de datos en menos de 1 hora.

## Tiempo de Respuesta del Sistema:

Definición: Cumplir con requisitos específicos de tiempo de respuesta para operaciones críticas.

Objetivo: Establecer un tiempo de respuesta promedio del sistema por debajo de los 5 segundos para cualquier operación crítica.

# Escenarios

Escenario 1: Un usuario quiere consultar su saldo en ambas monedas.

Justificación: este escenario muestra una de las funcionalidades básicas del sistema, que es permitir a los usuarios ver el saldo de sus cuentas en moneda local y extranjera.

Atributos de calidad: usabilidad, rendimiento, seguridad.

Estímulos: el usuario inserta su tarjeta y digita su PIN en el cajero automático, luego selecciona la opción de consulta de saldo en el menú principal.

Respuesta: el sistema valida la tarjeta y el PIN del usuario, luego accede a la base de datos y muestra el saldo de las cuentas del usuario en moneda local y extranjera en la pantalla del cajero automático.

Decisiones de arquitectura: se utiliza un microservicio de autenticación que se encarga de validar la identidad de los usuarios mediante el uso de una tarjeta y un PIN. Se utiliza un microservicio de consulta que se encarga de acceder a la base de datos y mostrar el saldo de las cuentas de los usuarios. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al usuario seleccionar la opción de consulta de saldo fácilmente.

Medición: se mide el tiempo de respuesta del sistema desde que el usuario selecciona la opción de consulta de saldo hasta que se muestra el saldo en la pantalla. Se mide el nivel de satisfacción del usuario con la interfaz de usuario y la información mostrada. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes.

Escenario 2: Un usuario quiere retirar dinero en moneda extranjera.

Justificación: este escenario muestra una de las funcionalidades diferenciadoras del sistema, que es permitir a los usuarios retirar dinero en moneda extranjera a una tasa de cambio especificada.

Atributos de calidad: usabilidad, rendimiento, seguridad, disponibilidad.

Estímulos: el usuario inserta su tarjeta y digita su PIN en el cajero automático, luego selecciona la opción de retiro en el menú principal, luego selecciona la moneda extranjera y el monto que desea retirar.

Respuesta: el sistema valida la tarjeta y el PIN del usuario, luego consulta la tasa de cambio vigente y calcula el monto equivalente en moneda local, luego verifica que el usuario tenga saldo suficiente en su cuenta y que el cajero automático tenga efectivo disponible en la moneda extranjera, luego dispensa el dinero al usuario y actualiza el saldo de su cuenta y el inventario del cajero automático, luego emite un recibo al usuario con los detalles de la operación.

Decisiones de arquitectura: se utiliza un microservicio de autenticación que se encarga de validar la identidad de los usuarios mediante el uso de una tarjeta y un PIN. Se utiliza un microservicio de transacción que se encarga de realizar las operaciones de retiro, depósito y transferencia en moneda local o extranjera. Se utiliza un microservicio de tasas de cambio que se encarga de manejar las tasas de cambio entre las dos monedas y de consultarlas por otros microservicios. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al usuario seleccionar la opción de retiro, la moneda y el monto fácilmente. Se utiliza un mecanismo de reciclaje que permite al cajero automático dispensar y recibir efectivo en ambas monedas.

Medición: se mide el tiempo de respuesta del sistema desde que el usuario selecciona la opción de retiro hasta que se dispensa el dinero. Se mide el nivel de satisfacción del usuario con la interfaz de usuario, la información mostrada y el recibo emitido. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes. Se mide el nivel de disponibilidad del sistema frente a posibles fallas o interrupciones.

Escenario 3: Un usuario quiere depositar dinero en moneda local.

Justificación: este escenario muestra una de las funcionalidades básicas del sistema, que es permitir a los usuarios depositar dinero en moneda local en sus cuentas.

Atributos de calidad: usabilidad, rendimiento, seguridad, disponibilidad.

Estímulos: el usuario inserta su tarjeta y digita su PIN en el cajero automático, luego selecciona la opción de depósito en el menú principal, luego selecciona la cuenta donde desea depositar el dinero, luego ingresa el dinero en el cajero automático.

Respuesta: el sistema valida la tarjeta y el PIN del usuario, luego verifica que la cuenta seleccionada sea válida y que el dinero ingresado sea auténtico y de la moneda local, luego acredita el dinero en la cuenta del usuario y actualiza el saldo de su cuenta y el inventario del cajero automático, luego emite un recibo al usuario con los detalles de la operación.

Decisiones de arquitectura: se utiliza un microservicio de autenticación que se encarga de validar la identidad de los usuarios mediante el uso de una tarjeta y un PIN. Se utiliza un microservicio de transacción que se encarga de realizar las operaciones de retiro, depósito y transferencia en moneda local o extranjera. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al usuario seleccionar la opción de depósito, la cuenta y el monto fácilmente. Se utiliza un mecanismo de reciclaje que permite al cajero automático dispensar y recibir efectivo en ambas monedas. Se utiliza un mecanismo de detección de billetes falsos que rechaza el dinero que no sea auténtico o de la moneda local.

Medición: se mide el tiempo de respuesta del sistema desde que el usuario selecciona la opción de depósito hasta que se acredita el dinero. Se mide el nivel de satisfacción del usuario con la interfaz de usuario, la información mostrada y el recibo emitido. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes. Se mide el nivel de disponibilidad del sistema frente a posibles fallas o interrupciones.

Escenario 4: Un usuario quiere transferir dinero en moneda extranjera a otra cuenta.

Justificación: este escenario muestra una de las funcionalidades diferenciadoras del sistema, que es permitir a los usuarios transferir dinero en moneda extranjera a otra cuenta a una tasa de cambio especificada.

Atributos de calidad: usabilidad, rendimiento, seguridad, disponibilidad.

Estímulos: el usuario inserta su tarjeta y digita su PIN en el cajero automático, luego selecciona la opción de transferencia en el menú principal, luego selecciona la cuenta de origen y la cuenta de destino, luego selecciona la moneda extranjera y el monto que desea transferir.

Respuesta: el sistema valida la tarjeta y el PIN del usuario, luego consulta la tasa de cambio vigente y calcula el monto equivalente en moneda local, luego verifica que el usuario tenga saldo suficiente en su cuenta de origen y que la cuenta de destino sea válida, luego debita el dinero de la cuenta de origen y lo acredita en la cuenta de destino, luego actualiza el saldo de ambas cuentas, luego emite un recibo al usuario con los detalles de la operación.

Decisiones de arquitectura: se utiliza un microservicio de autenticación que se encarga de validar la identidad de los usuarios mediante el uso de una tarjeta y un PIN. Se utiliza un microservicio de transacción que se encarga de realizar las operaciones de retiro, depósito y transferencia en moneda local o extranjera. Se utiliza un microservicio de tasas de cambio que se encarga de manejar las tasas de cambio entre las dos monedas y de consultarlas por otros microservicios. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al usuario seleccionar la opción de transferencia, la cuenta de origen, la cuenta de destino, la moneda y el monto fácilmente.

Medición: se mide el tiempo de respuesta del sistema desde que el usuario selecciona la opción de transferencia hasta que se acredita el dinero en la cuenta de destino. Se mide el nivel de satisfacción del usuario con la interfaz de usuario, la información mostrada y el recibo emitido. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes. Se mide el nivel de disponibilidad del sistema frente a posibles fallas o interrupciones.

Escenario 5: Un operador quiere actualizar las tasas de cambio entre las dos monedas.

Justificación: este escenario muestra una de las funcionalidades de administración del sistema, que es permitir a los operadores actualizar las tasas de cambio entre las dos monedas según el mercado.

Atributos de calidad: usabilidad, rendimiento, seguridad, consistencia.

Estímulos: el operador accede al sistema mediante un navegador web, luego ingresa su usuario y contraseña, luego selecciona la opción de gestión de tasas de cambio en el menú principal, luego ingresa las nuevas tasas de cambio entre las dos monedas y confirma la operación.

Respuesta: el sistema valida el usuario y la contraseña del operador, luego verifica que tenga permisos para modificar las tasas de cambio, luego actualiza las tasas de cambio en la base de datos y las notifica a los demás microservicios, luego muestra un mensaje de confirmación al operador con los detalles de la operación.

Decisiones de arquitectura: se utiliza un microservicio de administración que se encarga de permitir a los operadores gestionar los cajeros automáticos, los límites de operación, las comisiones y las tasas de cambio. Se utiliza un microservicio de tasas de cambio que se encarga de manejar las tasas de cambio entre las dos monedas y de consultarlas por otros microservicios. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al operador seleccionar la opción de gestión de tasas de cambio, ingresar las nuevas tasas de cambio y confirmar la operación fácilmente. Se utiliza un mecanismo de sincronización que permite mantener la consistencia de las tasas de cambio entre los microservicios y la base de datos.

Medición: se mide el tiempo de respuesta del sistema desde que el operador ingresa las nuevas tasas de cambio hasta que se muestra el mensaje de confirmación. Se mide el nivel de satisfacción del operador con la interfaz de usuario y la información mostrada. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes. Se mide el nivel de consistencia del sistema frente a posibles inconsistencias o conflictos.

Escenario 6: Un hacker quiere acceder a la base de datos del sistema y robar la información de los usuarios y las operaciones.

Justificación: este escenario muestra una de las amenazas a las que se enfrenta el sistema, que es el intento de acceso no autorizado a la base de datos por parte de un agente malicioso.

Atributos de calidad: seguridad, disponibilidad, confidencialidad, integridad.

Estímulos: el hacker utiliza una herramienta de escaneo de puertos para detectar los servicios que están expuestos en el sistema, luego utiliza una herramienta de fuerza bruta para intentar adivinar las credenciales de acceso a la base de datos, luego utiliza una herramienta de inyección SQL para intentar extraer la información de la base de datos.

Respuesta: el sistema detecta los intentos de escaneo de puertos y de fuerza bruta y bloquea las direcciones IP del hacker, luego verifica las credenciales de acceso a la base de datos y rechaza las que no sean válidas, luego valida las consultas SQL y evita las que contengan código malicioso, luego registra los eventos de seguridad y alerta a los administradores del sistema, luego mantiene la disponibilidad, la confidencialidad y la integridad de la información de la base de datos.

Decisiones de arquitectura: se utiliza un firewall que se encarga de filtrar el tráfico entrante y saliente del sistema y de bloquear los puertos y las direcciones IP sospechosas. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos que se encarga de cifrar los datos y de autenticar las conexiones. Se utiliza un mecanismo de encriptación que se encarga de proteger la información sensible de la base de datos. Se utiliza un mecanismo de validación que se encarga de evitar las consultas SQL mal formadas o maliciosas. Se utiliza un mecanismo de registro y alerta que se encarga de monitorear y notificar los eventos de seguridad.

Medición: se mide el número y la frecuencia de los intentos de acceso no autorizado a la base de datos. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes. Se mide el nivel de disponibilidad, confidencialidad e integridad del sistema frente a posibles interrupciones, filtraciones o alteraciones.

Escenario 7: Un usuario quiere recibir un recibo por correo electrónico de la operación que realizó en el cajero automático.

Justificación: este escenario muestra una de las funcionalidades complementarias del sistema, que es permitir a los usuarios recibir un recibo por correo electrónico de la operación que realizaron en el cajero automático.

Atributos de calidad: usabilidad, rendimiento, seguridad, disponibilidad.

Estímulos: el usuario inserta su tarjeta y digita su PIN en el cajero automático, luego realiza una operación de retiro, depósito, transferencia o consulta, luego selecciona la opción de recibir un recibo por correo electrónico en el menú final, luego ingresa su dirección de correo electrónico y confirma la operación.

Respuesta: el sistema valida la tarjeta y el PIN del usuario, luego realiza la operación solicitada por el usuario, luego verifica que la dirección de correo electrónico ingresada por el usuario sea válida, luego envía un recibo por correo electrónico al usuario con los detalles de la operación, luego muestra un mensaje de confirmación al usuario en la pantalla del cajero automático.

Decisiones de arquitectura: se utiliza un microservicio de autenticación que se encarga de validar la identidad de los usuarios mediante el uso de una tarjeta y un PIN. Se utiliza un microservicio de transacción que se encarga de realizar las operaciones de retiro, depósito, transferencia y consulta en moneda local o extranjera. Se utiliza un microservicio de notificación que se encarga de enviar mensajes de confirmación, alertas y recibos a los usuarios por correo electrónico o SMS. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al usuario seleccionar la opción de recibir un recibo por correo electrónico, ingresar su dirección de correo electrónico y confirmar la operación fácilmente.

Medición: se mide el tiempo de respuesta del sistema desde que el usuario selecciona la opción de recibir un recibo por correo electrónico hasta que se muestra el mensaje de confirmación. Se mide el nivel de satisfacción del usuario con la interfaz de usuario, la información mostrada y el recibo enviado. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes. Se mide el nivel de disponibilidad del sistema frente a posibles fallas o interrupciones.

Escenario 8: Un usuario quiere recibir una alerta por SMS si su saldo en moneda extranjera baja de un cierto umbral.

Justificación: este escenario muestra una de las funcionalidades de notificación del sistema, que es permitir a los usuarios recibir una alerta por SMS si su saldo en moneda extranjera baja de un cierto umbral que ellos mismos pueden configurar.

Atributos de calidad: usabilidad, rendimiento, seguridad, disponibilidad.

Estímulos: el usuario inserta su tarjeta y digita su PIN en el cajero automático, luego selecciona la opción de configuración en el menú principal, luego selecciona la opción de alertas en el menú de configuración, luego selecciona la opción de alerta por SMS en el menú de alertas, luego ingresa su número de teléfono y el umbral de saldo en moneda extranjera que desea establecer y confirma la operación.

Respuesta: el sistema valida la tarjeta y el PIN del usuario, luego muestra el menú de configuración al usuario, luego muestra el menú de alertas al usuario con las opciones disponibles, luego verifica que el número de teléfono y el umbral de saldo ingresados por el usuario sean válidos, luego guarda la configuración de alerta por SMS del usuario en la base de datos, luego muestra un mensaje de confirmación al usuario en la pantalla del cajero automático. Si en algún momento el saldo del usuario en moneda extranjera baja del umbral establecido, el sistema envía un SMS al usuario con el saldo actual y una advertencia.

Decisiones de arquitectura: se utiliza un microservicio de autenticación que se encarga de validar la identidad de los usuarios mediante el uso de una tarjeta y un PIN. Se utiliza un microservicio de configuración que se encarga de permitir a los usuarios configurar las alertas por SMS. Se utiliza un microservicio de notificación que se encarga de enviar mensajes de confirmación, alertas y recibos a los usuarios por correo electrónico o SMS. Se utiliza un protocolo de comunicación seguro entre los microservicios y la base de datos. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al usuario seleccionar la opción de configuración, la opción de alertas, la opción de alerta por SMS, ingresar su número de teléfono y el umbral de saldo y confirmar la operación fácilmente. Se utiliza un mecanismo de verificación que se encarga de validar el número de teléfono y el umbral de saldo ingresados por el usuario. Se utiliza un mecanismo de envío que se encarga de enviar los SMS a los usuarios mediante un proveedor externo.

Medición: se mide el tiempo de respuesta del sistema desde que el usuario selecciona la opción de configuración hasta que se muestra el mensaje de confirmación. Se mide el tiempo de envío del SMS desde que se detecta el saldo bajo hasta que se recibe el SMS. Se mide el nivel de satisfacción del usuario con la interfaz de usuario, la información mostrada y el SMS enviado. Se mide el nivel de seguridad del sistema frente a posibles ataques o fraudes. Se mide el nivel de disponibilidad del sistema frente a posibles fallas o interrupciones.

Escenario 9: Un administrador quiere monitorear el estado y el rendimiento de los microservicios del sistema.

Justificación: este escenario muestra una de las funcionalidades de observabilidad del sistema, que es permitir a los administradores monitorear el estado y el rendimiento de los microservicios del sistema, así como identificar y solucionar posibles problemas o anomalías.

Atributos de calidad: rendimiento, disponibilidad, mantenibilidad, escalabilidad.

Estímulos: el administrador accede al sistema mediante un navegador web, luego ingresa su usuario y contraseña, luego selecciona la opción de monitoreo en el menú principal, luego visualiza un panel con las métricas, los registros y los rastreos de los microservicios del sistema.

Respuesta: el sistema valida el usuario y la contraseña del administrador, luego muestra el menú principal al administrador, luego muestra el panel de monitoreo al administrador con las métricas, los registros y los rastreos de los microservicios del sistema, que incluyen información sobre el uso de recursos, el tiempo de respuesta, el número de peticiones, el número de errores, el número de instancias, etc. El sistema permite al administrador filtrar, ordenar, agrupar y comparar los datos según diferentes criterios, así como crear alarmas, acciones y reportes personalizados.

Decisiones de arquitectura: se utiliza un microservicio de administración que se encarga de permitir a los administradores acceder al sistema y seleccionar la opción de monitoreo. Se utiliza un servicio externo de monitoreo y observabilidad que se encarga de recopilar, almacenar y analizar las métricas, los registros y los rastreos de los microservicios del sistema. Se utiliza un protocolo de comunicación seguro entre el microservicio de administración y el servicio externo de monitoreo y observabilidad. Se utiliza una interfaz de usuario sencilla e intuitiva que permite al administrador visualizar el panel de monitoreo y manipular los datos fácilmente.

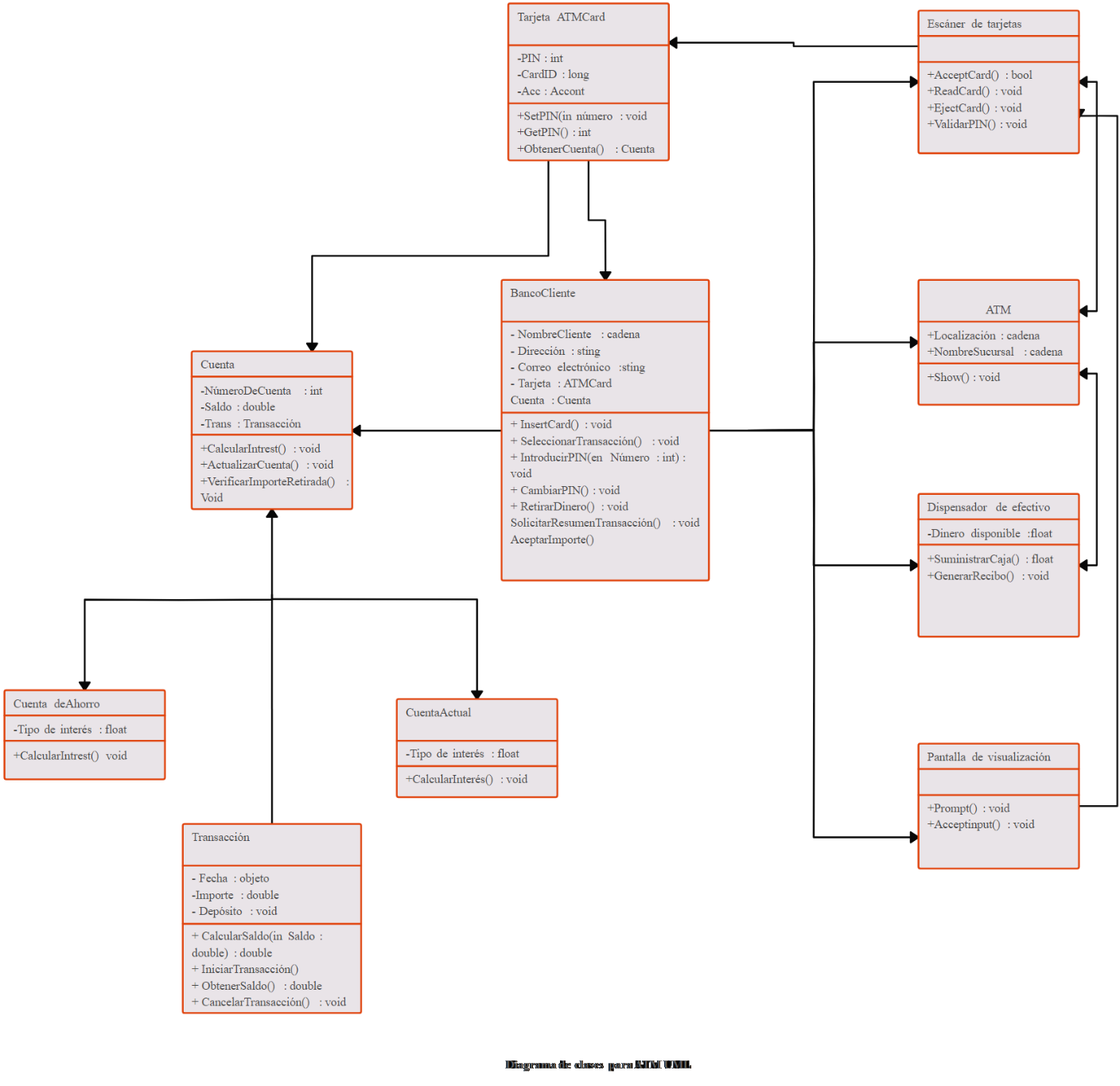
Medición: se mide el tiempo de respuesta del sistema desde que el administrador selecciona la opción de monitoreo hasta que se muestra el panel de monitoreo. Se mide el nivel de satisfacción del administrador con la interfaz de usuario y la información mostrada. Se mide el nivel de rendimiento, disponibilidad, mantenibilidad y escalabilidad de los microservicios del sistema.

# Vista Lógica

Las clases que intervienen en el sistema de cajero automático son las siguientes:

* Clase Cajero: representa el dispositivo físico que interactúa con el cliente y realiza las operaciones solicitadas. Tiene atributos como id, ubicación, inventario, estado, etc. Tiene operaciones como leerTarjeta, validarPIN, mostrarMenu, dispensarEfectivo, etc.
* Clase Tarjeta: representa la tarjeta que el cliente usa para acceder al sistema. Tiene atributos como número, tipo, fecha de vencimiento, etc. Tiene operaciones como validar, bloquear, desbloquear, etc.
* Clase Cliente: representa el usuario del sistema que tiene una o más cuentas asociadas. Tiene atributos como nombre, documento, teléfono, correo, etc. Tiene operaciones como consultarSaldo, retirarDinero, depositarDinero, transferirDinero, etc.
* Clase Cuenta: representa la cuenta bancaria del cliente que tiene un saldo en una moneda. Tiene atributos como número, moneda, saldo, etc. Tiene operaciones como debitar, acreditar, transferir, etc.
* Clase Operación: representa la transacción que el cliente realiza en el cajero. Tiene atributos como id, tipo, fecha, hora, monto, comisión, etc. Tiene operaciones como ejecutar, registrar, notificar, etc.
* Clase TasaDeCambio: representa la tasa de cambio entre dos monedas que se usa para calcular el monto equivalente en una operación. Tiene atributos como monedaOrigen, monedaDestino, valor, fecha, etc. Tiene operaciones como consultar, actualizar, aplicar, etc.
* Clase Operador: representa el empleado del banco que administra y monitorea el sistema. Tiene atributos como usuario, contraseña, rol, etc. Tiene operaciones como acceder, actualizarTasasDeCambio, monitorearCajeros, etc.

La clase Cajero se asocia con la clase Tarjeta, la clase Cliente y la clase Operación. La clase Tarjeta se generaliza en dos subclases: TarjetaDebito y TarjetaCredito. La clase Cliente se asocia con la clase Cuenta. La clase Cuenta se generaliza en dos subclases: CuentaLocal y CuentaExtranjera. La clase Operación se generaliza en cuatro subclases: Consulta, Retiro, Deposito y Transferencia. La clase Transferencia se asocia con la clase TasaDeCambio. La clase Operador se asocia con la clase Cajero y la clase TasaDeCambio.



Dado que el sistema se implementará mediante microservicios en AWS, se pueden identificar clases específicas relacionadas con los microservicios y la infraestructura en la nube. Aquí se presentan algunas clases adicionales que podrían ser relevantes para este contexto:

1. MicroservicioCuentas:
   * Implementa las funcionalidades del microservicio dedicado a la gestión de cuentas. Incluye métodos para la creación de cuentas, consulta de saldos y otras operaciones relacionadas.
2. MicroservicioTransacciones:
   * Implementa las operaciones del microservicio encargado de gestionar las transacciones. Contiene métodos para depósitos, retiros, conversiones de moneda y registro de transacciones.
3. MicroservicioTasasDeCambio:
   * Representa el microservicio que maneja las tasas de cambio entre la moneda local y el dólar estadounidense. Proporciona métodos para consultar y actualizar tasas de cambio.
4. AWSLambda:
   * Clase que encapsula las funciones Lambda de AWS para la ejecución de lógica específica en respuesta a eventos. Cada microservicio puede ser implementado como una o más funciones Lambda.
5. AmazonRDS:
   * Clase que representa la base de datos relacional en Amazon RDS que almacena información sobre cuentas, transacciones y tasas de cambio.
6. AmazonDynamoDB:
   * Clase que representa la base de datos NoSQL en Amazon DynamoDB, que puede utilizarse para almacenar datos de manera escalable y de alto rendimiento.
7. AmazonAPIGateway:
   * Clase que modela el servicio de Amazon API Gateway, que facilita la creación y gestión de APIs para exponer funcionalidades de los microservicios de manera segura y eficiente.
8. AmazonCloudWatch:
   * Clase que encapsula el servicio de monitoreo de AWS para realizar un seguimiento del rendimiento, los registros y las métricas de los microservicios desplegados.
9. AmazonS3:
   * Clase que representa el servicio de almacenamiento en la nube Amazon S3, que puede ser utilizado para almacenar archivos estáticos, como imágenes o documentos relacionados con las transacciones.
10. AmazonIAM:
    * Clase que modela el servicio de administración de identidades y accesos de AWS, para gestionar roles y permisos de acceso de los microservicios a otros servicios de AWS.

Estas clases están orientadas a reflejar la infraestructura y servicios específicos que se utilizarían en AWS para implementar un sistema de cajero automático multimoneda basado en microservicios. La elección de servicios específicos puede depender de los requisitos y restricciones del proyecto. Además, se debe tener en cuenta la seguridad, el rendimiento y la escalabilidad al diseñar la arquitectura y seleccionar los servicios de AWS.

# Vista de desarrollo

El diagrama de componentes UML muestra los componentes de software que conforman el sistema de cajero automático y las interfaces que definen la comunicación entre ellos. Los componentes son unidades modulares e independientes que encapsulan la lógica y los datos necesarios para realizar una función específica. Las interfaces son especificaciones que definen los servicios que ofrece o requiere un componente. Los componentes se conectan mediante interfaces, que pueden ser provistas o requeridas. Los componentes también pueden depender de otros componentes para realizar sus funciones.

En el diagrama de componentes que he creado, se pueden identificar los siguientes componentes e interfaces:

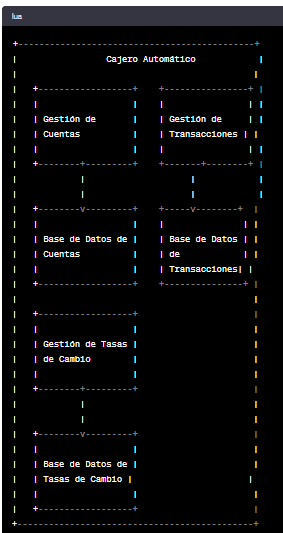
* Componente Cajero: es el componente principal que interactúa con el cliente y realiza las operaciones solicitadas. Provee una interfaz de usuario que permite al cliente seleccionar opciones en el menú y ver información en la pantalla. Requiere una interfaz de autenticación que valida la tarjeta y el PIN del cliente. Requiere una interfaz de transacción que realiza las operaciones de consulta, retiro, depósito y transferencia. Requiere una interfaz de notificación que envía recibos por correo electrónico o SMS al cliente. Requiere una interfaz de hardware que controla el lector de tarjetas, el dispensador de efectivo, el reciclador de billetes y el impresor de recibos.
* Componente Autenticación: es el componente que se encarga de validar la identidad de los clientes mediante el uso de una tarjeta y un PIN. Provee una interfaz de autenticación que recibe la tarjeta y el PIN del cliente y devuelve un resultado de validación. Requiere una interfaz de base de datos que accede a la información de las tarjetas y los clientes almacenada en la base de datos.
* Componente Transacción: es el componente que se encarga de realizar las operaciones de consulta, retiro, depósito y transferencia en moneda local o extranjera. Provee una interfaz de transacción que recibe el tipo y el monto de la operación y devuelve un resultado de ejecución. Requiere una interfaz de base de datos que accede a la información de las cuentas y las operaciones almacenada en la base de datos. Requiere una interfaz de tasas de cambio que consulta y aplica las tasas de cambio entre las dos monedas.
* Componente Notificación: es el componente que se encarga de enviar mensajes de confirmación, alertas y recibos a los clientes por correo electrónico o SMS. Provee una interfaz de notificación que recibe el tipo y el contenido del mensaje y devuelve un resultado de envío. Requiere una interfaz de correo electrónico que envía los mensajes por correo electrónico mediante un proveedor externo. Requiere una interfaz de SMS que envía los mensajes por SMS mediante un proveedor externo.
* Componente Tasas de cambio: es el componente que se encarga de manejar las tasas de cambio entre las dos monedas y de consultarlas por otros componentes. Provee una interfaz de tasas de cambio que recibe la moneda de origen y la moneda de destino y devuelve el valor de la tasa de cambio. Requiere una interfaz de base de datos que accede a la información de las tasas de cambio almacenada en la base de datos. Requiere una interfaz de actualización que permite a los operadores actualizar las tasas de cambio según el mercado.
* Componente Base de datos: es el componente que se encarga de almacenar y gestionar la información del sistema, como las tarjetas, los clientes, las cuentas, las operaciones y las tasas de cambio. Provee una interfaz de base de datos que recibe consultas y actualizaciones y devuelve resultados. Depende de un componente de hardware que provee el almacenamiento físico de los datos.
* Componente Hardware: es el componente que se encarga de proveer y controlar los dispositivos físicos que interactúan con el sistema, como el lector de tarjetas, el dispensador de efectivo, el reciclador de billetes, el impresor de recibos y el almacenamiento de datos. Provee una interfaz de hardware que recibe comandos y devuelve estados. Depende de un componente de sistema operativo que provee el acceso y la gestión de los recursos del sistema.
* Componente Sistema operativo: es el componente que se encarga de proveer el entorno de ejecución y la gestión de los recursos del sistema, como la memoria, el procesador, los dispositivos de entrada y salida, etc. Provee una interfaz de sistema operativo que recibe llamadas al sistema y devuelve resultados.

Espero que esta descripción te ayude a comprender mejor el diagrama de componentes UML. [Si quieres ver más información sobre este tipo de diagrama, puedes consultar estos enlaces1](https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/diagrama-de-componentes/)[2](https://diagramasuml.com/componentes/)[3](https://miro.com/es/diagrama/que-es-diagrama-componentes-uml/). También puedes usar esta herramienta online para crear tu propio diagrama de componentes UML. Espero que esta respuesta te haya sido útil. 😊

---------------

* Gestión de Cuentas:
  + Componente: Microservicio de Gestión de Cuentas
    - Responsabilidades:
      * Crear cuentas en ambas monedas.
      * Mantener registros de saldos en la moneda correspondiente.
      * Permitir a los usuarios consultar su saldo en ambas monedas.
  + Base de Datos: Almacenamiento de Información de Cuentas
    - Almacena la información de las cuentas, incluidos los saldos en ambas monedas.
* Gestión de Transacciones:
  + Componente: Microservicio de Gestión de Transacciones
    - Responsabilidades:
      * Procesar depósitos y retiros en ambas monedas.
      * Realizar conversiones de moneda a una tasa de cambio especificada.
      * Registrar cada transacción en una base de datos para seguimiento y auditoría.
  + Base de Datos: Registro de Transacciones
    - Almacena detalles de cada transacción realizada.
* Gestión de Tasas de Cambio:
  + Componente: Microservicio de Gestión de Tasas de Cambio
    - Responsabilidades:
      * Manejar las tasas de cambio entre las dos monedas.
      * Proporcionar servicios para que otros microservicios consulten las tasas de cambio.
  + Base de Datos: Almacenamiento de Tasas de Cambio
    - Almacena las tasas de cambio actuales entre las dos monedas.
* Interconexiones:
  + El Microservicio de Gestión de Transacciones se comunica con el Microservicio de Gestión de Cuentas para actualizar los saldos después de cada transacción.
  + El Microservicio de Gestión de Transacciones consulta el Microservicio de Tasas de Cambio para realizar conversiones precisas durante las transacciones.

Este diagrama de componentes refleja la modularidad del sistema, donde cada microservicio tiene responsabilidades específicas y se comunica con otros para realizar operaciones más complejas. Además, la base de datos asociada a cada microservicio almacena la información necesaria para cumplir con los requisitos funcionales y garantizar un seguimiento adecuado de las transacciones y cuentas.

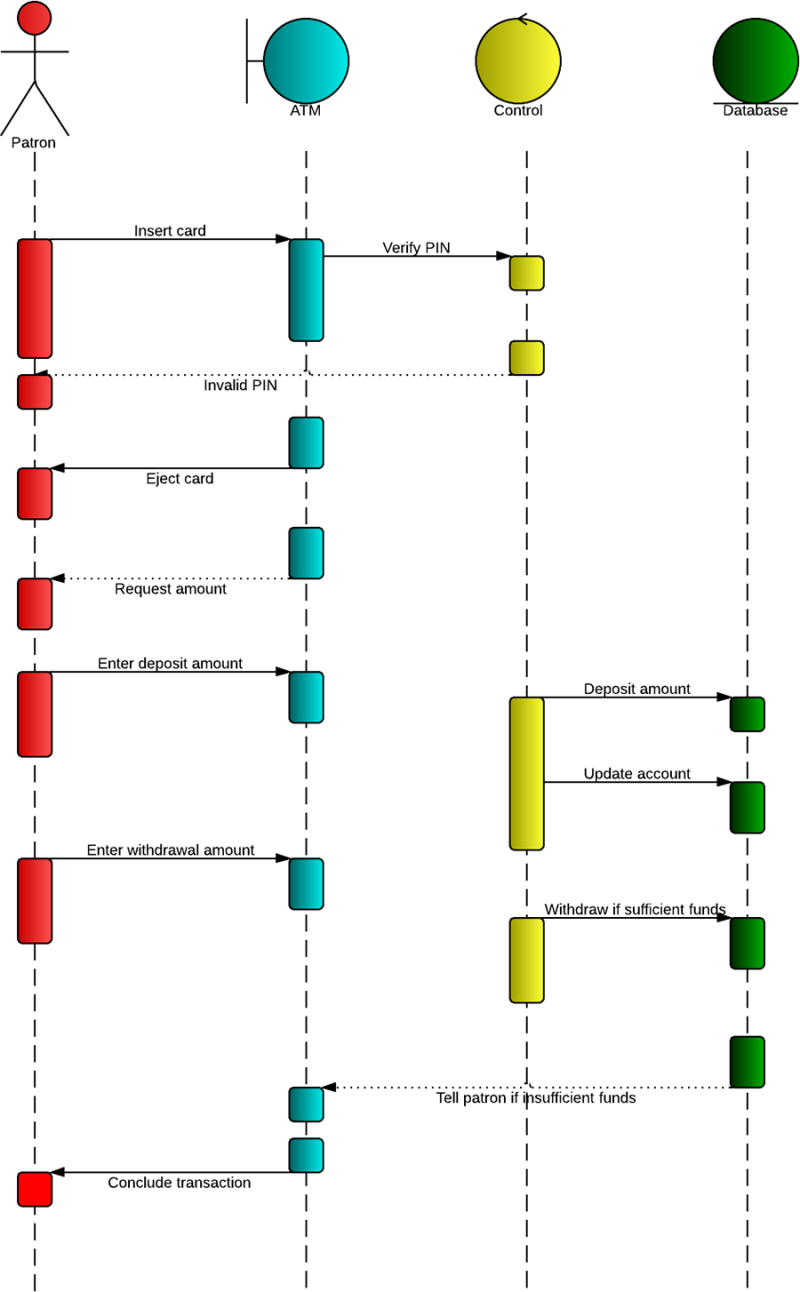


# Vista de proceso

Un diagrama de secuencia es una representación gráfica de cómo interactúan diferentes objetos en un sistema a lo largo del tiempo. Dado que el sistema de cajero automático multimoneda se implementará mediante microservicios en AWS, el diagrama de secuencia se centrará en las interacciones entre los microservicios y otros componentes clave. A continuación, se proporciona una descripción textual del diagrama de secuencia:

* Usuario Consulta Saldo:
  + El usuario inicia una solicitud para consultar el saldo.
  + La interfaz de usuario envía una solicitud al MicroservicioTransacciones.
  + El MicroservicioTransacciones consulta el MicroservicioCuentas para obtener el saldo correspondiente a la cuenta del usuario.
  + El MicroservicioCuentas responde con la información del saldo.
  + El MicroservicioTransacciones devuelve la respuesta a la interfaz de usuario, que presenta el saldo al usuario.
* Usuario Realiza Depósito:
  + El usuario inicia una solicitud para realizar un depósito.
  + La interfaz de usuario envía una solicitud al MicroservicioTransacciones con detalles del depósito.
  + El MicroservicioTransacciones consulta el MicroservicioCuentas para actualizar el saldo de la cuenta del usuario.
  + El MicroservicioCuentas actualiza el saldo y confirma la transacción.
  + El MicroservicioTransacciones registra la transacción en el MicroservicioTransaccionesDB.
  + Se devuelve una confirmación de la transacción a la interfaz de usuario.
* Usuario Realiza Conversión de Moneda:
  + El usuario inicia una solicitud para realizar una conversión de moneda.
  + La interfaz de usuario envía una solicitud al MicroservicioTransacciones con detalles de la conversión.
  + El MicroservicioTransacciones consulta el MicroservicioTasasDeCambio para obtener la tasa de cambio actual.
  + El MicroservicioTransacciones realiza la conversión y actualiza los saldos en ambas monedas en el MicroservicioCuentas.
  + Se registra la transacción en el MicroservicioTransaccionesDB.
  + Se devuelve una confirmación de la conversión a la interfaz de usuario.
* Usuario Realiza Retiro:
  + El usuario inicia una solicitud para realizar un retiro.
  + La interfaz de usuario envía una solicitud al MicroservicioTransacciones con detalles del retiro.
  + El MicroservicioTransacciones consulta el MicroservicioCuentas para verificar el saldo disponible.
  + El MicroservicioCuentas actualiza el saldo y confirma el retiro si hay fondos suficientes.
  + Si la transacción es exitosa, se registra en el MicroservicioTransaccionesDB.
  + Se devuelve una confirmación o mensaje de error a la interfaz de usuario.

Este es solo un ejemplo general de cómo podrían interactuar los microservicios en el sistema de cajero automático multimoneda. El diagrama de secuencia completo contendría más interacciones y detalles específicos de implementación. Además, cada interacción podría involucrar servicios adicionales de AWS, como el enrutamiento a través de Amazon API Gateway o el almacenamiento en bases de datos en Amazon RDS o DynamoDB.



[Sequence diagram: Lucidchart](https://lucid.app/lucidchart/de4e073e-b718-44aa-8e72-7813b095e348/edit?beaconFlowId=06D5129DCD4B2411&page=0_0&invitationId=inv_36697fdb-63f6-4415-bad1-624a93188e55)

# Vista física

El diagrama de despliegue describe cómo los diferentes componentes de un sistema se distribuyen en hardware o en entornos de ejecución, mostrando cómo se interconectan. Dado que estás implementando el sistema de cajero automático multimoneda mediante microservicios en AWS, el diagrama de despliegue se centrará en la representación de estos servicios en la nube de Amazon. Aquí tienes una descripción textual del diagrama de despliegue:

Usuario:

Representa los usuarios finales del sistema, que interactúan con la interfaz de usuario a través de dispositivos como navegadores web o aplicaciones móviles.

Interfaz de Usuario:

La interfaz de usuario está desplegada en los dispositivos de los usuarios y se comunica con los microservicios a través de la API expuesta por Amazon API Gateway.

Amazon API Gateway:

Amazon API Gateway actúa como el punto de entrada para las solicitudes de los usuarios. Expone las APIs que permiten la comunicación entre la interfaz de usuario y los microservicios.

MicroservicioCuentas:

Este microservicio está implementado como una o más funciones Lambda en AWS. Se encarga de la gestión de cuentas, incluida la creación y consulta de saldos.

MicroservicioTransacciones:

Implementado como funciones Lambda, este microservicio procesa las solicitudes relacionadas con las transacciones, como depósitos, retiros y conversiones de moneda.

MicroservicioTasasDeCambio:

Otro microservicio implementado como funciones Lambda, este gestiona las tasas de cambio entre la moneda local y el dólar estadounidense.

Amazon RDS:

Representa la base de datos relacional en Amazon RDS que almacena información sobre cuentas de usuario, transacciones y tasas de cambio.

Amazon DynamoDB:

La base de datos NoSQL de Amazon DynamoDB almacena datos que requieren un acceso rápido y escalabilidad eficiente, como registros de transacciones.

Amazon S3 (Opcional):

Puede utilizarse para almacenar archivos estáticos, como imágenes relacionadas con las transacciones.

Amazon CloudWatch:

Se utiliza para monitorear y gestionar el rendimiento de los microservicios, registrando métricas y permitiendo la configuración de alarmas.

Amazon IAM:

Amazon Identity and Access Management se encarga de la gestión de identidades y accesos, controlando los permisos y roles asociados con los diferentes servicios y componentes.

El diagrama de despliegue completo mostrará las conexiones entre estos componentes y cómo se comunican entre sí. Cabe destacar que la elección de servicios específicos de AWS y su configuración dependerá de los requisitos exactos y las consideraciones de implementación del sistema de cajero automático multimoneda.

Un diagrama de despliegue es un tipo de diagrama UML que muestra la distribución física de los componentes de software en los nodos de hardware que forman parte de un sistema. Los componentes de software son las unidades lógicas que realizan una función específica, como por ejemplo, aplicaciones, servicios, módulos, etc. Los nodos de hardware son los elementos físicos que alojan o ejecutan los componentes de software, como por ejemplo, computadoras, servidores, dispositivos móviles, etc. [Un diagrama de despliegue se utiliza para modelar la arquitectura física de un sistema, es decir, cómo se implementa y se despliega el software en el hardware1](https://diagramasuml.com/despliegue/).

Un diagrama de despliegue se compone de los siguientes elementos:

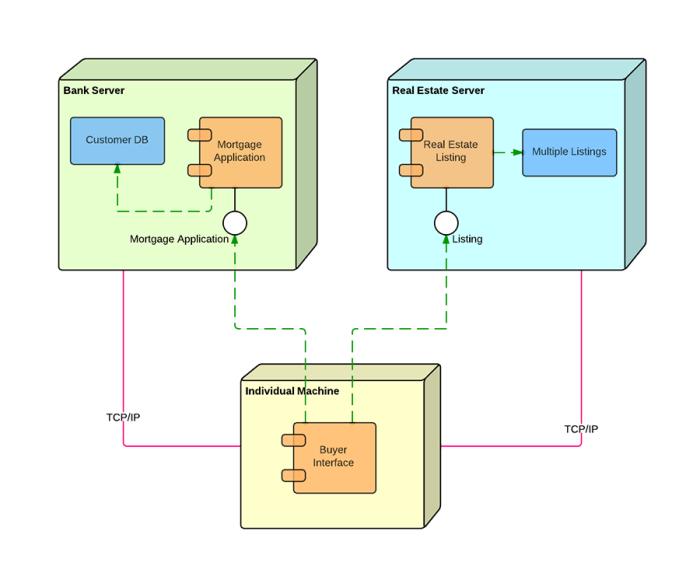
* Nodos: son los elementos que representan los nodos de hardware del sistema. Se dibujan como cubos tridimensionales con el nombre del nodo y opcionalmente el tipo de nodo. Los nodos pueden contener otros nodos o componentes en su interior, indicando que están compuestos o que alojan dichos elementos. Los nodos también pueden tener propiedades o estereotipos que describen sus características o funciones.
* Componentes: son los elementos que representan los componentes de software del sistema. Se dibujan como rectángulos con el nombre del componente y opcionalmente el tipo de componente. Los componentes pueden tener interfaces que definen los servicios que ofrecen o requieren. Los componentes también pueden tener propiedades o estereotipos que describen sus características o funciones.
* Artefactos: son los elementos que representan las unidades físicas de información que se generan o utilizan por los componentes de software, como por ejemplo, archivos, documentos, bases de datos, etc. Se dibujan como rectángulos con una muesca en la esquina superior derecha y el nombre del artefacto. Los artefactos pueden estar asociados o manifestados por los componentes, indicando que son producidos o consumidos por dichos elementos.
* Relaciones: son los elementos que representan las conexiones o dependencias entre los nodos, los componentes y los artefactos. Se dibujan como líneas o flechas con el nombre y el tipo de la relación. Las relaciones pueden ser de varios tipos, como por ejemplo, asociación, comunicación, despliegue, manifestación, dependencia, etc.

Un ejemplo de un diagrama de despliegue para el sistema de cajero automático que sea capaz de manejar operaciones en moneda local y extranjera es el siguiente:

En este ejemplo, se puede observar que el diagrama de despliegue muestra la distribución de los componentes de software en los nodos de hardware que conforman el sistema de cajero automático. El sistema se compone de los siguientes nodos:

* Cajero automático: es el nodo que representa el dispositivo físico que interactúa con el cliente y realiza las operaciones solicitadas. Contiene los componentes de interfaz de usuario, autenticación, transacción y notificación, que se encargan de la lógica de negocio del sistema. También contiene los artefactos de tarjeta, recibo y efectivo, que son las unidades físicas de información que se generan o utilizan por el sistema.
* Servidor web: es el nodo que representa el servidor que aloja la aplicación web que permite a los clientes acceder al sistema desde un navegador. Contiene el componente de interfaz web, que se encarga de la presentación y la interacción con el cliente. También contiene el artefacto de página web, que es la unidad física de información que se genera por el sistema.
* Servidor de aplicaciones: es el nodo que representa el servidor que aloja los servicios web que permiten la comunicación entre los componentes del sistema. Contiene los componentes de autenticación, transacción, notificación y tasas de cambio, que se encargan de la lógica de negocio del sistema. También contiene los artefactos de correo electrónico y SMS, que son las unidades físicas de información que se generan por el sistema.
* Servidor de base de datos: es el nodo que representa el servidor que aloja la base de datos que almacena la información del sistema, como las tarjetas, los clientes, las cuentas, las operaciones y las tasas de cambio. Contiene el componente de base de datos, que se encarga de la gestión y el acceso a los datos. También contiene el artefacto de base de datos, que es la unidad física de información que se utiliza por el sistema.

Las relaciones entre los nodos, los componentes y los artefactos indican cómo se comunican o dependen entre sí. Por ejemplo, el componente de interfaz de usuario se comunica con el componente de autenticación mediante una interfaz de autenticación, el componente de transacción se despliega en el componente de base de datos mediante una interfaz de base de datos, el componente de notificación manifiesta el artefacto de correo electrónico mediante una interfaz de correo electrónico, etc.



# Vista de casos de uso

Ejemplo:



En este ejemplo, se puede observar que el sistema tiene dos actores: el cliente y el operador. El cliente puede realizar cuatro casos de uso: consultar saldo, retirar dinero, depositar dinero y transferir dinero. El operador puede realizar dos casos de uso: actualizar tasas de cambio y monitorear cajeros automáticos. Los casos de uso se organizan en tres paquetes: gestión de cuentas, gestión de transacciones y gestión de tasas de cambio. Los casos de uso de retirar dinero, depositar dinero y transferir dinero incluyen el caso de uso de autenticar cliente, que es un caso de uso común que se realiza antes de cualquier operación. El caso de uso de transferir dinero se extiende con el caso de uso de enviar recibo por correo electrónico, que es un caso de uso opcional que se realiza si el cliente lo solicita. El caso de uso de actualizar tasas de cambio se generaliza con el caso de uso de monitorear cajeros automáticos, que es un caso de uso más abstracto que engloba varias acciones de administración.