Ejercicio Practico: Arquitecto de Software DevSu LLC.

Nombre: Joseph Iván Gallardo Pincay

Descripción:

Usted ha sido contratado por una entidad llamada BP como arquitecto de soluciones para diseñar un sistema de banca por internet, en este sistema los usuarios podrán acceder al histórico de sus movimientos, realizar transferencias y pagos entre cuentas propias e interbancarias.

Toda la información referente al cliente se tomará de 2 sistemas, una plataforma Core que contiene información básica de cliente, movimientos, productos y un sistema independiente que complementa la información del cliente cuando los datos se requieren en detalle.

Debido a que la norma exige que los usuarios sean notificados sobre los movimientos realizados, el sistema utilizará sistemas externos o propios de envío de notificaciones, mínimo 2.

Este sistema contará con 2 aplicaciones en el Front, una SPA y una Aplicación móvil desarrollada en un Framework multiplataforma. (Mencione 2 opciones y justifique el porqué de su elección).

Ambas aplicaciones autenticarán a los usuarios mediante un servicio que usa el estándar Oauth2.0, para el cual no requiere implementar toda la lógica, ya que la compañía cuenta con un producto que puede ser configurado para este fin; sin embargo, debe dar recomendaciones sobre cuál es el mejor flujo de autenticación que se debería usar según el estándar.

Tenga en cuenta que el sistema de Onboarding para nuevos clientes en la aplicación móvil usa reconocimiento facial, por tanto, su arquitectura deberá considerarlo como parte del flujo de autorización y autenticación, a partir del Onboarding el nuevo usuario podrá ingresar al sistema mediante usuario y clave, huella o algún otro método especifique alguno de los anteriores dentro de su arquitectura, también puede recomendar herramientas de industria que realicen estas
tareas y robustezca su aplicación.

El sistema utiliza una base de datos de auditoría que registra todas las acciones del cliente y cuenta con un mecanismo de persistencia de información para clientes frecuentes, para este caso proponga una alternativa basada en patrones de diseño que relacione los componentes que deberían interactuar para conseguir el objetivo.

Para obtener los datos del cliente el sistema pasa por una capa de integración compuesta por un api Gateway y consume los servicios necesarios de acuerdo con el tipo de transacción, inicialmente usted cuenta con 3 servicios principales, consulta de datos básicos, consulta de movimientos y transferencias que realiza llamados a servicios externos dependiendo del tipo, si considera que debería agregar más servicios para mejorar el rendimiento de su arquitectura o agregar más servicios para mejorar la repuesta de información a sus clientes, es libre de hacerlo.

Consideraciones

Para este reto, mencione aquellos elementos normativos que considere importantes a tener en cuenta para entidades financieras, Ejemplo ley de datos personales, seguridad etc.

Garantice en su arquitectura, alta disponibilidad (HA), tolerancia a fallos, recuperación ante desastres (DR), Seguridad y Monitoreo, Excelencia operativa y auto-healing.

Si lo considera necesario, su arquitectura puede contener elementos de infraestructura en nube, Azure u AWS, garantice baja latencia, cuenta con presupuesto para esto.

En lo posible plantee una arquitectura desacoplada con elementos y reusables y cohesionados para otros componentes que puedan adicionarse en el futuro.

El modelo debe ser desarrollado bajo modelo c4 (Modelo de Contexto, Modelo de aplicación o contenedor y componentes), describa hasta el modelo de componentes, la infraestructura la puede modelar como usted lo considere usando la herramienta de su preferencia.

1. Background

La entidad BP requiere un sistema de banca por internet que permita a los usuarios gestionar sus cuentas bancarias, realizar transferencias y consultar su historial de movimientos. El sistema deberá interactuar con una plataforma Core y un sistema de información complementaria, además de cumplir con normativas financieras y de seguridad aplicables.

2. Requerimientos

Requisitos de Producto (usando el método MoSCoW)

- **Must Have (Obligatorios)**
    - Acceso de los usuarios al historial de movimientos.
    - Funcionalidad de transferencia y pago entre cuentas propias e interbancarias.
    - Integración con una plataforma Core para datos básicos del cliente y movimientos.
    - Sistema de notificaciones obligatorio para informar a los usuarios de movimientos realizados.
    - Autenticación y autorización seguras mediante OAuth2.0.
    - Inclusión de un flujo de Onboarding con reconocimiento facial en la app móvil.
    - Cumplimiento con regulaciones de seguridad y privacidad de datos financieros.
    - Alta disponibilidad (HA), tolerancia a fallos y mecanismos de recuperación ante desastres (DR).
    - Monitoreo y seguridad con auto-healing y excelencia operativa.
- **Should Have (Deseables)**
    - Persistencia de información para clientes frecuentes con almacenamiento de alta velocidad.
    - Infraestructura en la nube para optimizar baja latencia y escalabilidad.
- **Could Have (Opcionales)**
    - Implementación de servicios adicionales para optimizar rendimiento y tiempos de respuesta en el consumo de información de cliente.

3. Elementos Normativos Clave

3.1. Ley de protección de datos personales de Ecuador.

La Ley Orgánica de Protección de Datos Personales de Ecuador, promulgada en mayo de 2021, tiene como objetivo garantizar el derecho de los ciudadanos a la protección de sus datos personales. Esta ley

regula, prevé y desarrolla principios, derechos, obligaciones y mecanismos de tutela para asegurar que los datos personales sean tratados de manera adecuada.

¿Quién debe cumplir la Ley de Protección de Datos?

Toda **empresa o entidad, sea pública o privada, domiciliada en Ecuador**, que tenga y/o maneje datos personales debe cumplir esta normativa. Las empresas domiciliadas en otro país también deberán cumplirla siempre que los dueños de los datos estén residiendo en Ecuador y/o por las condiciones que mencionamos anteriormente.

3.2. Normativa PCI-DSS (Payment Card Industry Data Security Standard)

La normativa PCI-DSS (Payment Card Industry Data Security Standard) es un conjunto de estándares de seguridad diseñados para proteger la información de las tarjetas de pago durante su procesamiento, almacenamiento y transmisión. Fue desarrollada por el Consejo de Normas de Seguridad de la Industria de Tarjetas de Pago (PCI SSC), que incluye a las principales marcas de tarjetas como Visa, MasterCard, American Express, Discover y JCB.

La normativa PCI-DSS es esencial para cualquier entidad que maneje datos de tarjetas de pago, ya que ayuda a reducir el riesgo de fraude y proteger la información sensible de los clientes

¿Es aplicable PCI DSS al proyecto de BP?

- **No aplica directamente**, ya que el alcance del sistema descrito no incluye almacenamiento, transmisión o procesamiento de datos de tarjetas.
    - Las funcionalidades mencionadas (consultar movimientos, transferencias y pagos) pueden operar sin manejar datos de tarjetas si estas transacciones se procesan a través de otros sistemas (por ejemplo, el sistema Core Bancarios o gateways de pago externos).
- **Responsabilidad en integración**:
    - Si BP interactúa con terceros que manejan pagos mediante tarjetas (por ejemplo, integraciones con procesadores de pago o redes de tarjetas como Visa o MasterCard), estos terceros deben cumplir con PCI DSS. BP debe asegurarse de seleccionar proveedores certificados.

- **Monitoreo del sistema Core**:
    - Si el sistema Core Bancario maneja datos de tarjetas (lo que no está especificado en el enunciado), BP deberá garantizar que este sistema esté en cumplimiento con PCI DSS, aunque el sistema de banca por internet no se vea afectado directamente.

3.3. Norma ISO-IEC 27001

La norma ISO 27001 es un estándar internacional para la gestión de la seguridad de la información. Su objetivo principal es proteger la confidencialidad, integridad y disponibilidad de la información dentro de una organización.

Puntos Clave de la Norma ISO 27001

1. **Sistema de Gestión de Seguridad de la Información (SGSI)**:
    - Establece un marco para gestionar la seguridad de la información mediante políticas, procedimientos y controles.
2. **Evaluación de Riesgos**:
    - Identifica y evalúa los riesgos de seguridad de la información para implementar medidas de mitigación adecuadas.
3. **Confidencialidad, Integridad y Disponibilidad**:
    - Asegura que la información solo sea accesible a personas autorizadas (confidencialidad), que no sea alterada de manera no autorizada (integridad) y que esté disponible cuando se necesite (disponibilidad).
4. **Mejora Continua**:
    - Utiliza el ciclo PDCA (Planificar, Hacer, Verificar, Actuar) para la mejora continua del SGSI.
5. **Cumplimiento Legal y Contractual**:
    - Asegura que la organización cumpla con todas las leyes y regulaciones aplicables relacionadas con la seguridad de la información.
6. **Auditorías Internas y Externas**:
    - Realiza auditorías periódicas para verificar la efectividad del SGSI y asegurar el cumplimiento con la norma.

La implementación de la norma ISO 27001 ayuda a la empresa BP a gestionar de manera efectiva los riesgos de seguridad de la información y a proteger sus activos de información de amenazas potenciales.

3.4. Norma ISO 22301 – Sistema de Gestión de Continuidad de Negocio.

La norma ISO 22301 es un estándar internacional que establece los requisitos para un sistema de gestión de la continuidad del negocio (SGCN). Su objetivo principal es ayudar a las organizaciones a prevenir, prepararse, responder y recuperarse de incidentes inesperados que puedan interrumpir sus operaciones. Si bien no es requerido por ninguna ley del Ecuador, es un marco normativo que da la pauta para garantizar la continuidad operativa del sistema de banca por internet, ya que representa un servicio crítico para los clientes.

El sistema de banca BP depende de:

- **Alta disponibilidad (HA):** Configuración Multi-AZ y respaldo de infraestructura en AWS.
- **Tolerancia a fallos (FT):** Uso de réplicas y monitoreo activo para garantizar la continuidad.
- **Recuperación ante desastres (DR):** Snapshots automáticos y failover entre zonas/instancias.

Estas características están alineadas con los principios de ISO 22301, ya que permiten:

- **Evaluar riesgos e interrupciones potenciales**: Por ejemplo, fallos en zonas de disponibilidad o ataques cibernéticos.
- **Definir planes de continuidad específicos** para el sistema:
    - Ejemplo: Mantener el acceso al sistema Core y garantizar que los clientes puedan operar en línea incluso ante fallos.
- **Probar y evaluar periódicamente** los planes de recuperación y continuidad.

3.5. Grupo de acción financiera Internacional (GAFI).

Las recomendaciones del Grupo de Acción Financiera Internacional (GAFI) son un conjunto de estándares internacionales diseñados para

combatir el lavado de dinero, la financiación del terrorismo y otras amenazas relacionadas con la integridad del sistema financiero.

Recomendaciones del GAFI para Entidades Financieras

1. **Políticas y Coordinación AML/CFT**:
    - Implementar políticas y procedimientos para prevenir el lavado de dinero y la financiación del terrorismo (AML/CFT).
    - Coordinar esfuerzos a nivel nacional e internacional para combatir estas actividades ilícitas.
2. **Debida Diligencia del Cliente (CDD)**:
    - Realizar procedimientos de identificación y verificación de clientes.
    - Monitorear las transacciones para detectar actividades sospechosas.
3. **Registro y Reporte de Transacciones**:
    - Mantener registros detallados de las transacciones y reportar aquellas que sean sospechosas a las autoridades competentes.
4. **Transparencia y Beneficiarios Finales**:
    - Asegurar la transparencia en la propiedad y el control de las entidades legales.
    - Identificar y verificar a los beneficiarios finales de las cuentas y transacciones.
5. **Controles Internos y Auditoría**:
    - Establecer controles internos robustos y realizar auditorías periódicas para asegurar el cumplimiento de las políticas AML/CFT.
6. **Capacitación y Concienciación**:
    - Capacitar al personal sobre las políticas y procedimientos AML/CFT.
    - Fomentar una cultura de cumplimiento dentro de la organización.
7. **Cooperación Internacional**:

- Colaborar con otras jurisdicciones y organismos internacionales para compartir información y coordinar esfuerzos en la lucha contra el lavado de dinero y la financiación del terrorismo.

Estas recomendaciones ayudan a las entidades financieras como BP a fortalecer sus sistemas de prevención y detección de actividades ilícitas, protegiendo así la integridad del sistema financiero global.

4. Detalles de Arquitectura.

4.1. Alta Disponibilidad (HA)

1. **Zonas de Disponibilidad (AZs)**:
   - Configuración en múltiples zonas de disponibilidad dentro de la nube seleccionada (AWS) para replicar servicios y datos.
   - Asegura que el sistema continúe operando incluso si una zona falla.
2. **Balanceo de Carga**:
   - Utilizar un **Application Load Balancer (ALB)** para distribuir las solicitudes entre instancias de microservicios en diferentes zonas.
   - Permite mantener la disponibilidad durante picos de tráfico.
3. **Bases de Datos de Alta Disponibilidad**:
   - Configurar la base de datos de auditoría en modo multi-AZ o con replicación activa/activa para garantizar el acceso continuo.
   - Tecnologías sugeridas:
     - **Amazon RDS con Multi-AZ**.

4.2. Tolerancia a Fallos

1. **Despliegue en Contenedores**:
   - Utilizar Kubernetes (EKS en AWS) para gestionar los microservicios con múltiples réplicas.
   - Configurar políticas de reinicio automático en caso de fallo de un contenedor.

2. **Redundancia en Servicios Críticos**:
    - Configurar instancias redundantes para todos los servicios críticos, como el API Gateway y los microservicios principales.
    - Ejemplo: Si un servicio de transferencia falla, su réplica puede continuar atendiendo solicitudes.
3. **Fallback en Notificaciones**:
    - Si un canal (como email) falla, implementar un fallback para enviar la notificación por SMS o push.

4.3. Recuperación Ante Desastres (DR)

1. **Copia de Seguridad y Replicación**:
    - Configurar replicación continua de bases de datos y backups automatizados hacia una región secundaria.
    - Usar herramientas como **AWS Backup**.
2. **Pruebas de DR**:
    - Implementar simulaciones periódicas para probar los tiempos de recuperación (RTO < 30 minutos, RPO ≈ 0).
3. **Plan de DR Multi-Región**:
    - Desplegar una arquitectura multi-región para recuperación ante desastres, redirigiendo el tráfico hacia la región secundaria mediante un **DNS Failover** (Route 53 en AWS).

4.4. Seguridad

1. **Protección de Datos**:
    - **Encriptación en tránsito**: TLS 1.2/1.3 para todas las conexiones.
    - **Encriptación en reposo**: Cifrar bases de datos y backups con claves gestionadas en KMS (AWS Key Management Service).
2. **Autenticación Segura**:
    - Uso de OAuth2.0 con Authorization Code Flow y PKCE.
    - Integración de MFA y biometría para usuarios móviles.

3. **Control de Acceso**:
    - Implementar políticas de IAM (Identity Access Management) para restringir el acceso a los recursos.
    - Configurar roles mínimos necesarios para microservicios y desarrolladores.
4. **Protección Contra Ataques**:
    - Configuración de firewalls para proteger los servicios de contenedores.
    - Uso de **WAF (Web Application Firewall)** para prevenir ataques como inyección SQL y XSS.

4.5. Monitoreo

1. **Supervisión en Tiempo Real**:
    - Utilizar herramientas como **Amazon CloudWatch** para rastrear métricas clave (CPU, memoria, tiempos de respuesta, tráfico).
    - Configurar dashboards para la visibilidad de eventos y alertas.
2. **Registros Centralizados**:
    - Consolidar logs de API Gateway, microservicios y bases de datos en un sistema como **ElasticSearch**.
3. **Alertas Proactivas**:
    - Configurar alertas automáticas para anomalías de rendimiento, uso excesivo de recursos o fallos de servicio.

4.6. Excelencia Operativa

1. **Integración y Despliegue Continuo (CI/CD)**:
    - Implementar pipelines de CI/CD utilizando herramientas como **AWS CodePipeline**.
    - Automatizar pruebas y despliegues para reducir riesgos en producción.
2. **Estandarización de Infraestructura**:

- Utilizar herramientas de Infraestructura como Código (IaC), como **Terraform** o **AWS CloudFormation**, para asegurar consistencia en los despliegues.

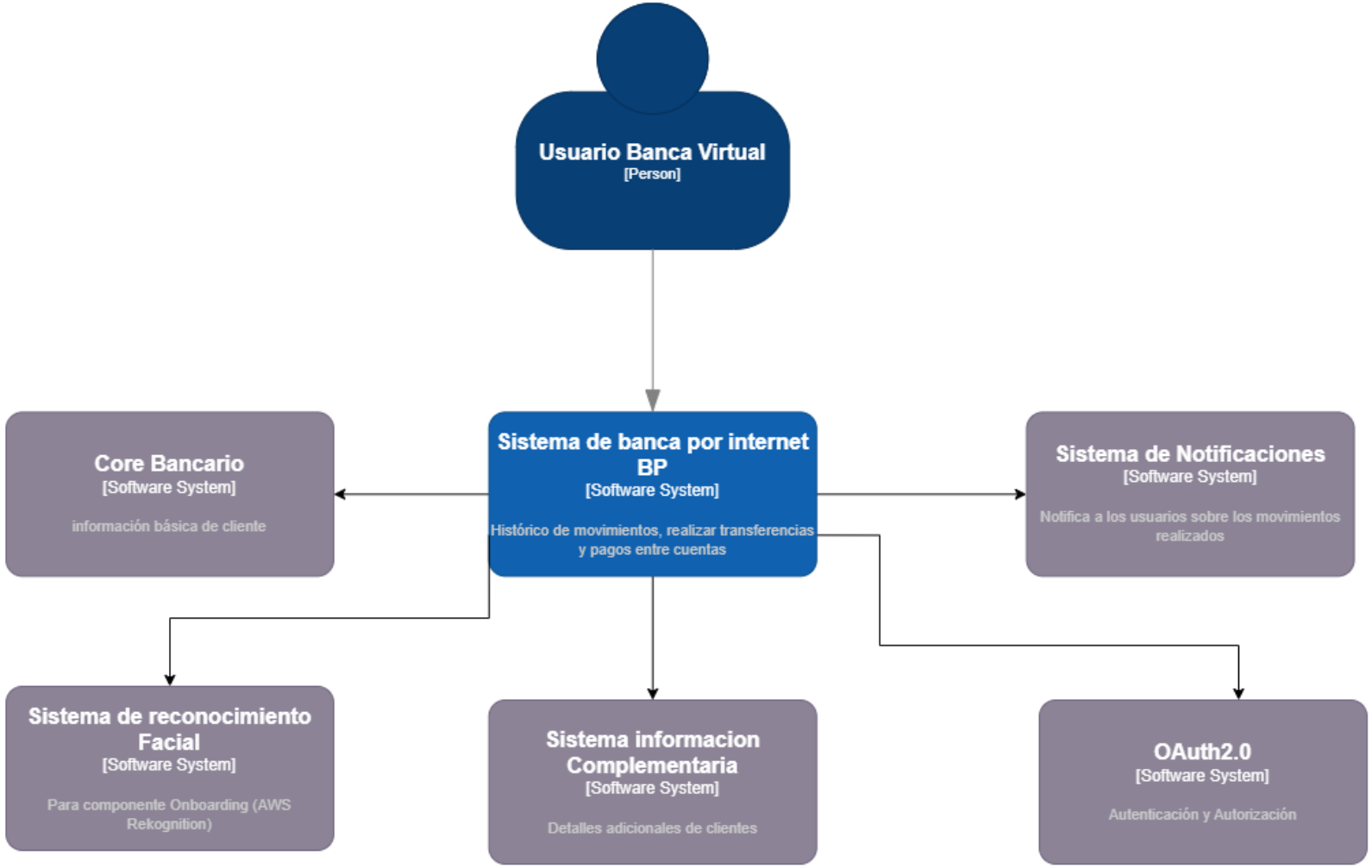
3. **Pruebas Automatizadas**:
   - Configurar pruebas de carga, seguridad y regresión en cada etapa del despliegue.

4.7. Auto-Healing

1. **Gestión Autónoma de Contenedores**:
   - Configurar Kubernetes para reiniciar contenedores fallidos automáticamente.
   - Habilitar políticas de reintento en aplicaciones y microservicios para manejar errores transitorios.
2. **Autoscaling**:
   - Escalar automáticamente las instancias de microservicios según la carga, utilizando **Horizontal Pod Autoscaler** en Kubernetes.
3. **Fallback Inteligente**:
   - Implementar lógica de fallback para redirigir las solicitudes a instancias disponibles en caso de fallos.

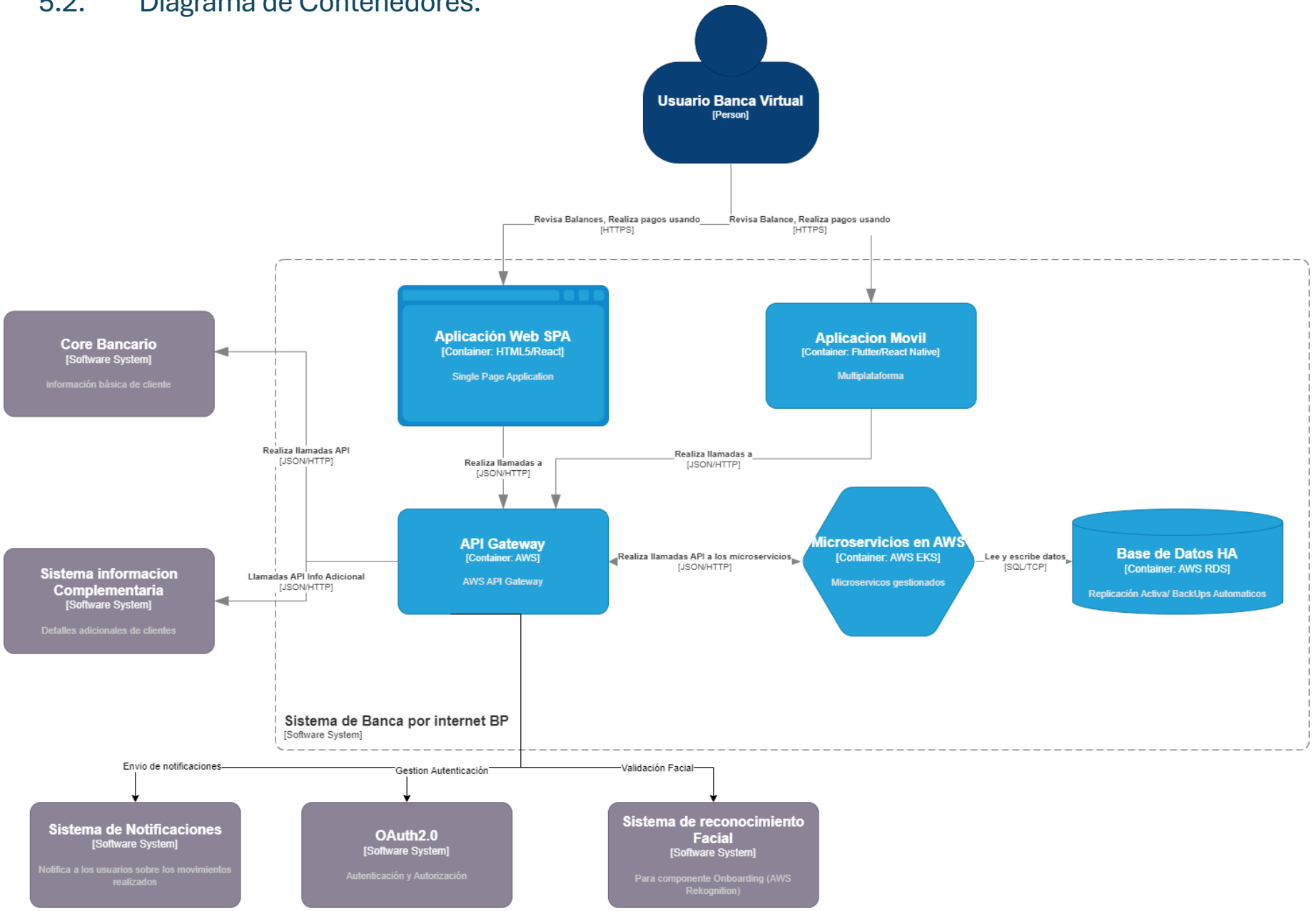
5. Modelo C4 (Contexto, Contenedor, Componentes)

5.1. Diagrama de Contexto.

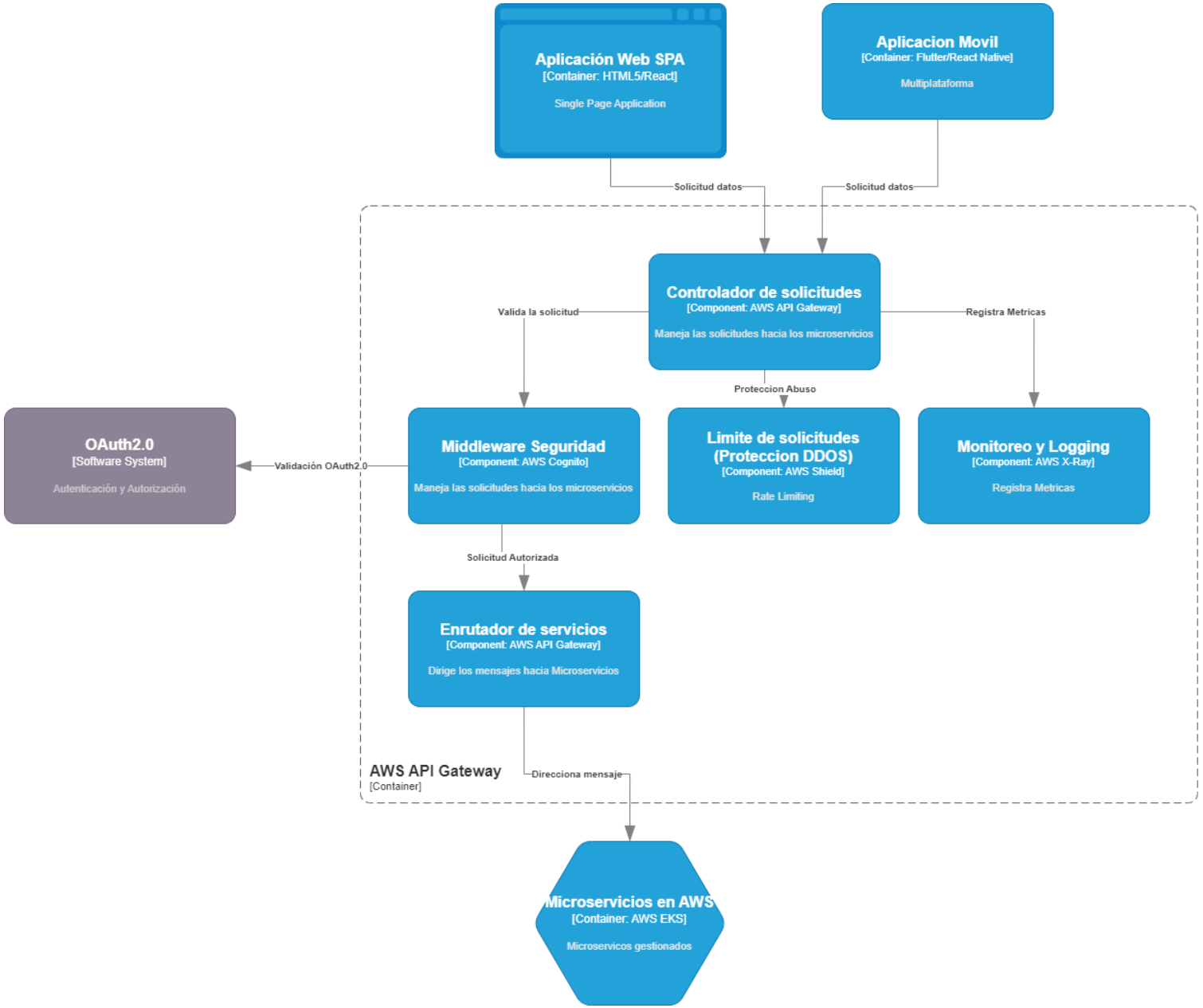


[System Context] Sistema de banca por Internet BP
Diagrama de Contexto BP

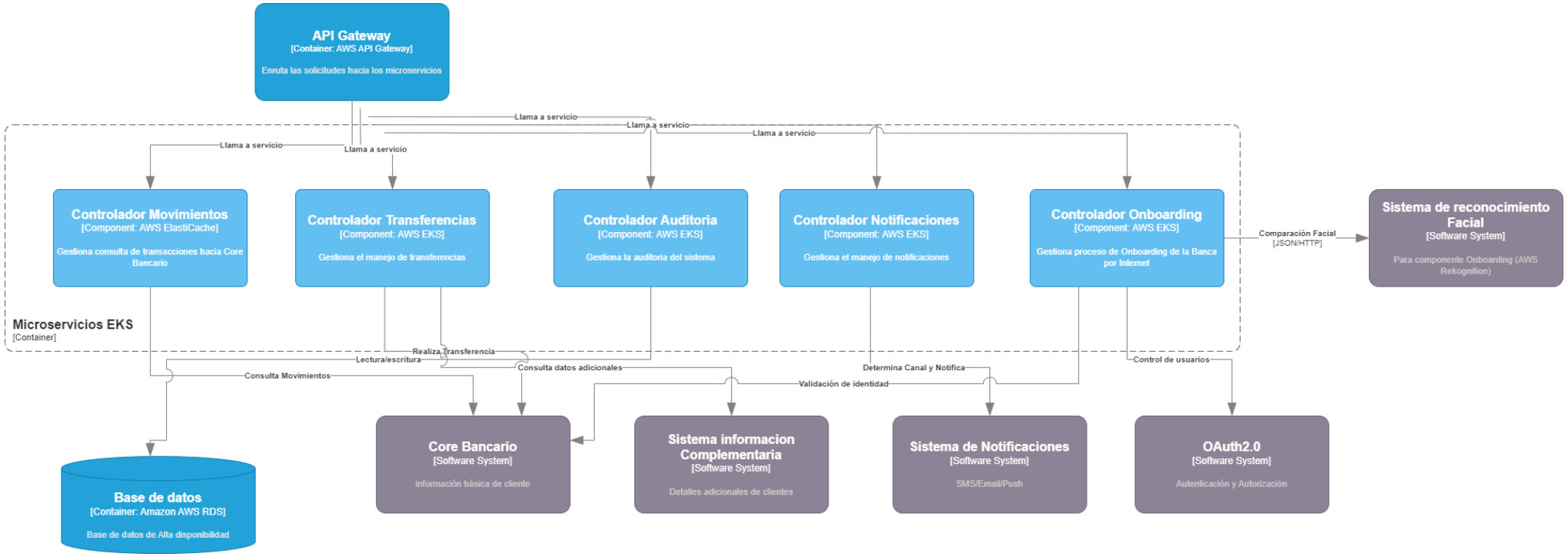
5.2. Diagrama de Contenedores.



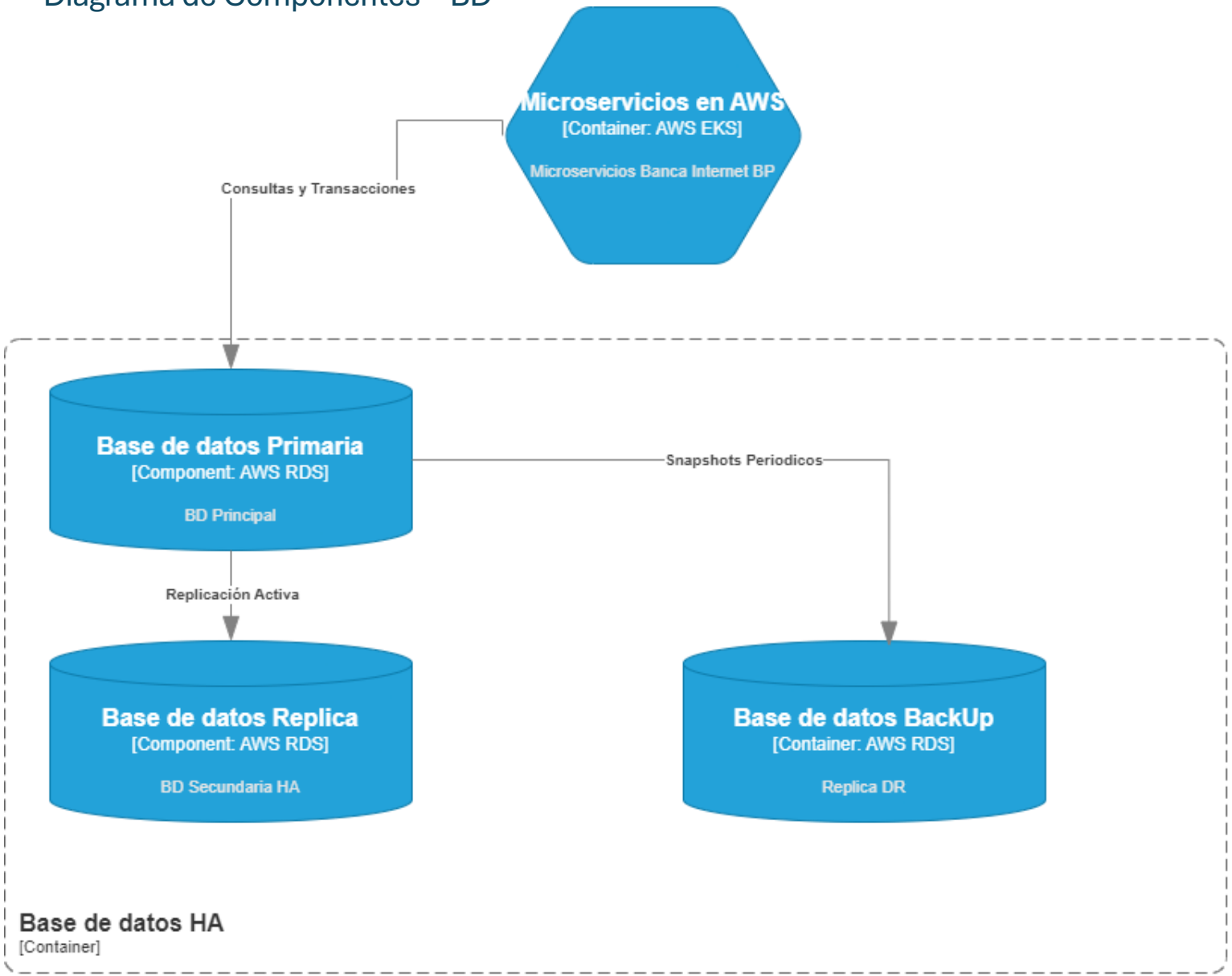
5.3. Diagrama de Componentes – API Gateway



5.4. Diagrama de componentes – Microservicios en AWS



5.5. Diagrama de Componentes – BD



6. Consideraciones

6.1. Desarrollos Fron-End

Para las aplicaciones Front-End, proponemos una **SPA (Single Page Application)** para la web y una **aplicación móvil multiplataforma** desarrollada con frameworks modernos. A continuación, se presentan las opciones seleccionadas con su justificación.

Opciones para la SPA

1. **React**:
    - **Justificación**:
        - Amplio soporte de la comunidad y ecosistema robusto de herramientas.
        - Ideal para aplicaciones interactivas y en tiempo real como banca por internet.
        - Fácil integración con bibliotecas como **Redux** para el manejo de estado.
        - Excelente desempeño en combinación con **AWS Amplify** y distribución vía **Amazon CloudFront**.

Opciones para la Aplicación Móvil Multiplataforma

1. **Flutter**:
    - **Justificación**:
        - Renderizado nativo de alta calidad para Android e iOS, ofreciendo una experiencia consistente para el usuario.
        - Herramientas integradas para manejar animaciones y interfaces complejas, necesarias en una app bancaria.
        - Código único para múltiples plataformas, lo que reduce costos y tiempo de desarrollo.
        - Soporte para integración con AWS Amplify y servicios de autenticación como Amazon Cognito.
2. **React Native**:
    - **Justificación**:
        - Comunidad madura con acceso a una amplia gama de bibliotecas de terceros.

- Reutilización de código entre aplicaciones web y móviles, especialmente útil si se utiliza React para la SPA.
- Compatible con herramientas nativas para optimizar el desempeño (por ejemplo, interacciones específicas de plataforma).

6.2. Flujo de Onboarding.

Para integrar el **reconocimiento facial** en el flujo de **Onboarding** y contemplar múltiples métodos de autenticación, propongo un diseño que incluye herramientas de la industria y un flujo adaptado para garantizar la seguridad y experiencia de usuario. Esto se implementará como parte de la aplicación móvil y el backend.

Flujo Propuesto para el Onboarding y Autenticación

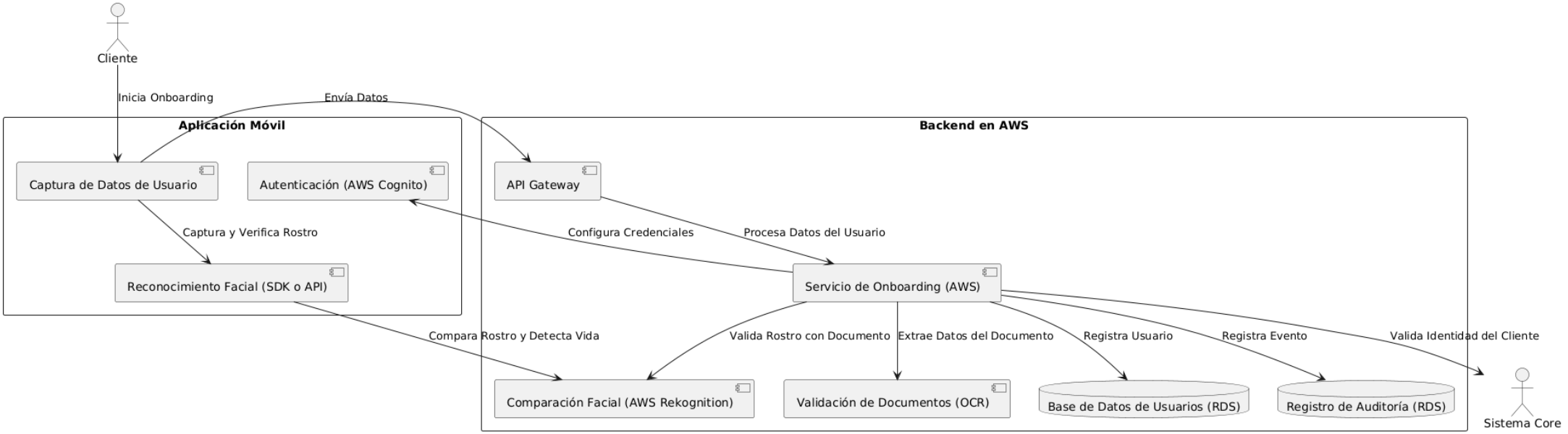
1. **Registro Inicial**:
    - El nuevo cliente descarga la aplicación móvil y selecciona la opción de registro.
    - El sistema solicita información básica (nombre, correo, número de documento).
2. **Verificación de Identidad con Reconocimiento Facial**:
    - El usuario captura una foto o un video.
    - El sistema compara el rostro con el documento de identidad proporcionado (OCR + biometría facial).
    - Se valida que el usuario esté presente (detección de vida) para prevenir fraudes.
3. **Creación de Credenciales**:
    - Una vez verificada la identidad, se solicita crear un nombre de usuario y contraseña.
    - Opcionalmente, se habilita la autenticación biométrica (huella o reconocimiento facial) para futuros accesos.
4. **Autenticación en el Futuro**:
    - Métodos disponibles:
        - **Usuario y Contraseña**: Método tradicional.

- **Reconocimiento Facial**: Utilizando el hardware nativo del dispositivo (Face ID, Android BiometricPrompt).
- **Huella Dactilar**: Integración con sensores biométricos del dispositivo.
- **MFA (Multi-Factor Authentication)**: Mensajes SMS o códigos generados por aplicaciones como Authy o Google Authenticator.

Herramientas Recomendadas para Reconocimiento Facial

1. **AWS Rekognition**:
    - Servicio gestionado de Amazon que ofrece:
        - Detección facial.
        - Comparación de caras con imágenes existentes (para validar contra documentos).
        - Detección de vida (liveness detection).
    - Integración nativa con otros servicios AWS como S3 y Lambda.

Flujo de Onboarding Explicado:



1. **Cliente**:
    - Inicia el proceso desde la aplicación móvil, capturando sus datos y rostro.
2. **Aplicación Móvil**:
    - Utiliza un SDK o API de reconocimiento facial para capturar y verificar la biometría.
    - Envía los datos al backend mediante el **API Gateway**.
3. **Backend en AWS**:
    - **AWS Rekognition** valida el rostro y detecta vida.
    - **OCRService** extrae información del documento de identidad.

- Se valida la identidad con el **Sistema Core** y se registra al cliente en la **Base de Datos de Usuarios (RDS)**.
- **AWS Cognito** crea las credenciales para autenticación futura.
- Se registra el evento en la **Base de Datos de Auditoría**.

6.3. Componentes y Costos Estimados

6.3.1. Front-End

- **SPA en Amazon S3 + CloudFront**:
  - Almacenamiento: $0.023/GB (S3 Standard) para 50 GB de datos (~$1.15/mes).
  - Transferencia de datos: $0.085/GB (suponiendo 500 GB/mes ~ $42.50/mes).
  - **Costo estimado:** ~$45/mes.
- **Aplicación Móvil Multiplataforma**:
  - Sin costos directos, ya que se distribuye desde tiendas como App Store/Google Play.

6.3.2. Back-End

- **Amazon API Gateway**:
  - $3.50 por millón de llamadas + $0.09/GB transferido.
  - Suponiendo 50 millones de llamadas y 200 GB transferidos:
    - Llamadas: $175/mes.
    - Transferencia: $18/mes.
  - **Costo estimado:** ~$193/mes.
- **Amazon EKS (Microservicios)**:
  - Nodo Kubernetes (EC2 m5.large): $0.096/hora (~$69.12/mes por nodo).
  - Supongamos un clúster con 4 nodos: ~$276/mes.
  - Costo adicional para balanceador de carga (Elastic Load Balancer): ~$30/mes.
  - **Costo estimado:** ~$306/mes.

6.3.3. Base de Datos

- **Amazon RDS (Multi-AZ, PostgreSQL)**:
  - Nodo db.m6g.large: $0.115/hora (~$82.80/mes por instancia).
  - Replica Multi-AZ: ~$82.80/mes.
  - Almacenamiento: $0.115/GB al mes para 100 GB (~$11.50/mes).
  - **Costo estimado:** ~$177/mes.
- **Amazon ElastiCache (Redis Multi-AZ)**:
  - Nodo cache.m6g.large: $0.108/hora (~$77.76/mes por nodo).
  - Replica Multi-AZ: ~$77.76/mes.
  - **Costo estimado:** ~$155/mes.

6.3.4. Servicios Gestionados

- **Amazon SNS**:
  - $0.50 por millón de notificaciones + costos por canal (SMS, email, push).
  - Supongamos 10 millones de notificaciones, principalmente push (bajo costo):
    - Costo base: $5/mes.
  - **Costo estimado:** ~$5/mes.
- **AWS Rekognition**:
  - $1 por 1,000 imágenes procesadas.
  - Suponiendo 100,000 imágenes/mes:
    - **Costo estimado:** ~$100/mes.
- **AWS Cognito**:
  - Gratis para los primeros 50,000 MAUs (Monthly Active Users).
  - Supongamos 100,000 MAUs:
    - 50,000 usuarios extra: $0.0055/usuario (~$275/mes).
  - **Costo estimado:** ~$275/mes.

6.3.5. Monitoreo y Recuperación

- **AWS CloudWatch**:
  - $0.30/metric/month y $0.50/GB logs (suponiendo 500 métricas y 100 GB logs).
  - **Costo estimado:** ~$215/mes.
- **AWS Backup**:
  - Snapshots de 100 GB: $0.05/GB (~$5/mes).
  - **Costo estimado:** ~$5/mes.

6.3.6. Resumen de Costos Mensuales Estimados

Componente	Costo Estimado Mensual
Front-End	$45
API Gateway	$193
Amazon EKS	$306
RDS Multi-AZ	$177
ElastiCache	$155
SNS	$5
Rekognition	$100
Cognito	$275
CloudWatch + Backup	$220
Total Aproximado	**$1,476/mes**

6.3.7. Optimización de Costos

6.3.7.1. Front-End

- Usar **AWS S3 Intelligent-Tiering** para almacenamiento menos accesado ($0.0125/GB).
- Habilitar compresión en CloudFront para reducir transferencia de datos.

6.3.7.2. API Gateway

- Implementar WebSocket API Gateway para reducir costos de solicitudes frecuentes.
- Usar **Direct Lambda Integration** para bypass del Gateway en casos no críticos.

6.3.7.3. Amazon EKS

- Escalar nodos dinámicamente con Kubernetes Autoscaler.
- Usar instancias **Spot** para cargas no críticas (ahorro de hasta 70%).

6.3.7.4. Base de Datos

- Configurar **RDS Aurora Serverless** para escalar automáticamente según la demanda.
- Usar Redis en modo "On-Demand" y escalar según la frecuencia de uso.

6.3.7.5. Servicios Gestionados

- **Cognito**: Limitar inactividad de cuentas para reducir el conteo de usuarios activos.
- **SNS**: Priorizar push notifications (costos menores) sobre SMS.

6.3.7.6. Monitoreo

- Habilitar retención limitada de logs históricos en **CloudWatch** y exportar logs a **S3** para almacenamiento más económico.