Sistema de Banca por Internet

1. Introducción

1.1. Propósito

Este documento tiene como propósito describir la arquitectura del sistema de banca por internet de BP. Está dirigido a arquitectos de software, desarrolladores, ingenieros de infraestructura y cualquier otro stakeholder involucrado en el desarrollo y mantenimiento del sistema.

1.2. Alcance

El sistema de banca por internet permitirá a los usuarios acceder a su historial de movimientos, realizar transferencias y pagos entre cuentas propias e interbancarias. El sistema se integrará con plataformas internas y externas, cumplirá con normativas de seguridad y ofrecerá alta disponibilidad y tolerancia a fallos.

1.3. Referencias

- OAuth 2.0 Authorization Framework: RFC 6749
- General Data Protection Regulation (GDPR)
- Payment Card Industry Data Security Standard (PCI-DSS)

1.4. Definiciones y Acrónimos

- **SPA**: Single Page Application.
- **OAuth 2.0**: Open standard for access delegation.
- **HA**: High Availability.
- **DR**: Disaster Recovery.
- **API**: Application Programming Interface.

2. Visión General del Sistema

2.1. Contexto del Sistema

El sistema de banca por internet de BP es una solución tecnológica que consta de varios actores y componentes integrados. A continuación, se muestra el diagrama de contexto del sistema:

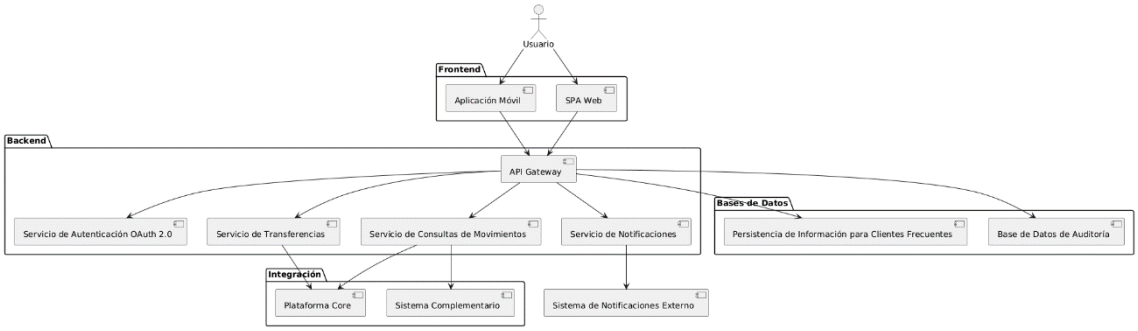

Ilustración 1Diagrama de Contexto

2.2. Principios de Diseño

- **Seguridad:** Protección de los datos y transacciones de los usuarios mediante estándares de la industria.
- **Alta Disponibilidad (HA):** El sistema debe estar disponible en todo momento, con redundancia incorporada.
- **Escalabilidad:** Capacidad para escalar horizontalmente en función de la demanda.
- **Cumplimiento Normativo:** Alineación con regulaciones como GDPR y PCI-DSS

3. Requisitos

3.1. Requisitos Funcionales

3.1.1. Debe tener

- Autenticación y autorización de usuarios mediante OAuth 2.0.
- Integración con la plataforma Core para obtener información básica de clientes, movimientos y productos.
- Integración con el sistema complementario para obtener detalles adicionales de los clientes.
- Mecanismo de notificaciones utilizando al menos dos sistemas externos o propios.
- Registro de todas las acciones del cliente en una base de datos de auditoría.
- Onboarding con reconocimiento facial para nuevos usuarios en la aplicación móvil.
- Funcionalidades de acceso a históricos de movimientos y realización de transferencias y pagos.
- Alta disponibilidad (HA) y tolerancia a fallos.

3.1.2. Debería tener

- Mecanismo de persistencia de información para clientes frecuentes basado en patrones de diseño adecuados.
- Optimización de consultas y servicios para asegurar baja latencia.

- Sistema de monitoreo para detección temprana de fallos y auto-healing.
- Compatibilidad multiplataforma para la aplicación móvil (considerando frameworks como Flutter o React Native).
- Recomendaciones para el flujo de autenticación más seguro y adecuado para OAuth 2.0.

3.1.3. Podría tener

- Funcionalidades adicionales para el futuro como alertas personalizadas, gestión de inversiones, etc.
- Integración con otros sistemas o plataformas financieras externas para ampliar la oferta de servicios.
- Personalización avanzada del interfaz de usuario según las preferencias del cliente.

3.1.4. No tendrá

- Características o servicios que no estén alineados con las normativas actuales de BP o que no sean esenciales para el lanzamiento inicial.

3.2. Requisitos No Funcionales

- **Seguridad**: Implementación de cifrado en tránsito y en reposo, control de acceso basado en roles, y auditoría de seguridad.
- **Rendimiento:** Respuesta rápida (< 2 segundos) para las operaciones comunes como consulta de saldos y movimientos.
- **Disponibilidad**: 99.9% de uptime garantizado mediante el uso de estrategias de alta disponibilidad.
- **Escalabilidad:** Capacidad de escalar horizontalmente para soportar incrementos en la carga del sistema.
- **Mantenibilidad:** Código modular y desacoplado, con un ciclo de vida de desarrollo bien definido y documentado.
- **Cumplimiento:** Asegurar el cumplimiento con GDPR, PCI-DSS y cualquier otra regulación financiera relevante.

4. Arquitectura del Sistema

4.1. Modelo de Aplicación (Contenedor)

El sistema está compuesto por varios contenedores, cada uno con responsabilidades específicas. A continuación, se muestra el diagrama del modelo de contenedor:

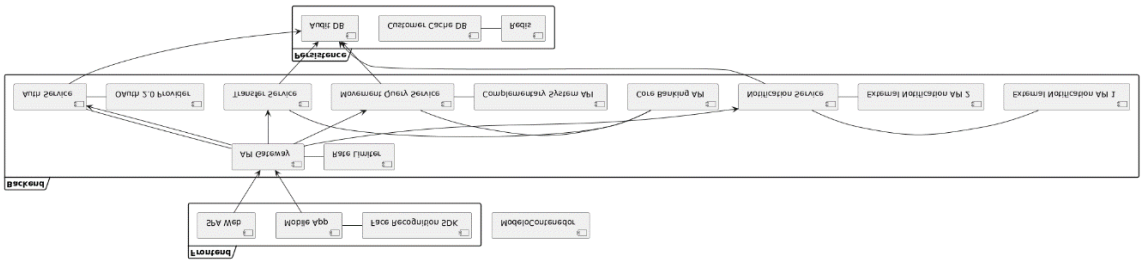

Ilustración 2 Modelo Contenedor

4.2. Modelo de Componentes

Cada contenedor incluye varios componentes que cumplen roles específicos dentro del sistema. A continuación, se describen los componentes más importantes:

- **Face Recognition SDK (Mobile App):** Un SDK integrado en la aplicación móvil para el reconocimiento facial durante el proceso de onboarding.
- **API Gateway:** Punto de entrada único para todas las solicitudes externas. Incluye limitación de tasa para controlar la carga y evitar abusos.
- **Auth Service:** Maneja la autenticación y autorización de usuarios, integrándose con un proveedor OAuth 2.0.
- **Transfer Service:** Gestiona las transacciones financieras entre cuentas, tanto internas como interbancarias.
- **Movement Query Service:** Proporciona acceso al historial de movimientos de las cuentas de los usuarios.
- **Notification Service:** Gestiona el envío de notificaciones a los usuarios a través de múltiples proveedores externos.
- **Audit DB:** Almacena un registro detallado de todas las acciones realizadas por los usuarios.
- **Customer Cache DB:** Base de datos en caché para mejorar el rendimiento en consultas frecuentes.

4.3. Infraestructura y Despliegue

El sistema se desplegará en una plataforma en la nube, utilizando servicios como AWS o Azure para garantizar la escalabilidad, alta disponibilidad y recuperación ante desastres.

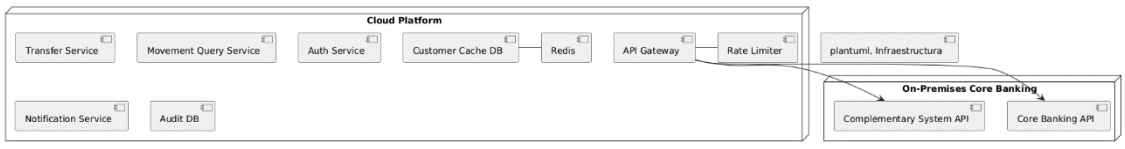

Ilustración 3 Diagrama de Infraestructura

5. Diseño de Seguridad

5.1. Autenticación y Autorización

El sistema utilizará OAuth 2.0 como estándar para la autenticación y autorización de usuarios. A continuación, se describen los aspectos clave del diseño de seguridad para estos procesos:

- Flujo de Autenticación:
  - Se recomienda utilizar el Authorization Code Grant con Proof Key for Code Exchange (PKCE). Este flujo es especialmente adecuado para aplicaciones móviles y SPA, ya que añade una capa de seguridad adicional evitando el uso directo de Client Secrets en el cliente.
  - Durante el proceso de autenticación, el cliente (SPA o aplicación móvil) solicitará un código de autorización que, posteriormente, canjeará por un token de acceso. Este token será utilizado para acceder a recursos protegidos.
- Gestión de Tokens:
  - Acceso: El acceso a los recursos protegidos se gestionará mediante Access Tokens emitidos por el servicio de OAuth 2.0. Estos tokens tendrán una vida útil corta para minimizar el riesgo de uso indebido en caso de compromiso.
  - Refresco de Tokens: Se utilizarán Refresh Tokens para permitir la renovación de los Access Tokens sin necesidad de reautenticar al usuario, mejorando la experiencia de usuario sin comprometer la seguridad.
  - Revocación de Tokens: En caso de detección de un compromiso de seguridad o una solicitud explícita por parte del usuario (e.g., cierre de sesión), los tokens podrán ser revocados inmediatamente.
- Multi-Factor Authentication (MFA):
  - Para mejorar la seguridad del acceso, se puede implementar MFA como una capa adicional. Esto podría incluir el uso de autenticación basada en SMS, aplicaciones de autenticación como Google Authenticator, o biometría (huella digital o reconocimiento facial).

5.2. Protección de Datos

La protección de los datos es crítica en un sistema financiero. A continuación, se describen las medidas de seguridad implementadas para proteger la información sensible:

- Cifrado en Tránsito:
  - Todo el tráfico entre el cliente y el servidor será cifrado utilizando HTTPS/TLS. Esto asegura que los datos no puedan ser interceptados ni alterados por atacantes durante su transmisión.
- Cifrado en Reposo:
  - Los datos sensibles almacenados en las bases de datos, como la información de los clientes y los registros de auditoría, se cifrarán utilizando algoritmos de cifrado robustos, por ejemplo AES-256.

    - Las claves de cifrado serán gestionadas mediante un servicio de gestión de claves (KMS) seguro proporcionado por la plataforma en la nube (AWS KMS, Azure Key Vault).
- Control de Acceso:
    - El acceso a los datos y servicios estará restringido mediante el uso de roles y políticas de control de acceso basadas en principios de privilegio mínimo. Sólo los componentes y usuarios autorizados podrán acceder a la información crítica.
    - El acceso administrativo a los sistemas se realizará mediante conexiones seguras y autenticación multifactor.

5.3. Cumplimiento Normativo

El sistema debe cumplir con varias normativas internacionales y locales para garantizar la seguridad de los datos y las operaciones financieras. A continuación, se describen los aspectos clave del cumplimiento normativo:

- GDPR:
    - Cumplimiento con la Regulación General de Protección de Datos (GDPR) de la UE, que incluye derechos del usuario sobre sus datos, como el acceso, rectificación y eliminación. El sistema incluirá mecanismos para permitir a los usuarios ejercer estos derechos.
    - La retención de datos se gestionará según las políticas de la organización y los requisitos legales, asegurando la eliminación segura de los datos cuando ya no sean necesarios.
- PCI-DSS:
    - Para el manejo de datos de tarjetas de pago, el sistema cumplirá con el estándar de seguridad de datos de la industria de tarjetas de pago (PCI-DSS), asegurando que los datos de tarjetas se manejen de forma segura.
    - Implementación de controles específicos, como el cifrado de datos de tarjeta, segmentación de redes, y monitoreo continuo para detectar y prevenir posibles violaciones de seguridad.
- Auditoría y Registro:
    - Todos los eventos críticos del sistema, incluyendo accesos a datos sensibles, transacciones financieras, y cambios en la configuración del sistema, serán registrados en la Base de Datos de Auditoría.
    - Estos registros estarán disponibles para auditorías internas y externas, asegurando el cumplimiento de las normativas y permitiendo una trazabilidad completa de las actividades del sistema.

5.4. Monitoreo y Respuesta a Incidentes

La detección y respuesta rápida a incidentes de seguridad es fundamental para minimizar el impacto de potenciales amenazas.

- Monitoreo Continuo:

    - Implementación de herramientas de monitoreo y análisis de seguridad, como AWS CloudTrail, Azure Security Center, o Splunk, para vigilar las actividades del sistema en tiempo real.
    - Configuración de alertas automáticas para eventos críticos, como intentos fallidos de acceso, cambios en la configuración de seguridad, o actividad inusual en las transacciones.
- Respuesta a Incidentes:
    - Establecimiento de un plan de respuesta a incidentes que defina los pasos a seguir en caso de un incidente de seguridad, incluyendo la identificación, contención, erradicación, y recuperación.
    - Ejercicios regulares de simulación de incidentes para asegurar que el equipo esté preparado para responder eficazmente.
- Auto-Healing:
    - Configuración de mecanismos de auto-healing para servicios críticos. En caso de detección de fallos o comportamientos anómalos, las instancias afectadas serán automáticamente reiniciadas o reemplazadas por nuevas instancias operativas.

6. Diseño del Frontend

6.1. SPA Web

La aplicación web del sistema de banca por internet será desarrollada como una Single Page Application (SPA). A continuación, se describen los aspectos clave del diseño:

- **Framework:** La SPA se desarrollará utilizando React o Angular. Ambas opciones permiten crear interfaces de usuario dinámicas y responsivas que pueden comunicarse eficientemente con el backend a través de APIs REST.
- **Arquitectura:** La arquitectura de la SPA seguirá el patrón de diseño basado en componentes, lo que permite un desarrollo modular y fácil de mantener. Cada componente de la interfaz de usuario manejará una parte específica de la aplicación, como el inicio de sesión, la visualización de movimientos, o la realización de transferencias.
- **Comunicación con el Backend:** La SPA se comunicará con el API Gateway para todas las operaciones, incluyendo autenticación, consulta de movimientos, y realización de transferencias. Esta comunicación se realizará a través de solicitudes HTTP seguras (HTTPS).
- **Autenticación:** La autenticación de usuarios se gestionará a través del flujo de OAuth 2.0, específicamente el Authorization Code Grant con PKCE. El token de acceso se almacenará en memoria o en un almacenamiento seguro del navegador, como “sessionStorage”, para minimizar el riesgo de exposición.
- **Experiencia de Usuario (UX):** La SPA estará optimizada para ofrecer una experiencia de usuario fluida, con transiciones rápidas y feedback inmediato en las operaciones realizadas por el usuario.

6.2. Aplicación Móvil

La aplicación móvil será desarrollada utilizando un framework multiplataforma que permita un desarrollo eficiente para tanto Android como iOS. Las dos opciones recomendadas son Flutter y React Native.

- Framework:
  - Flutter: Ideal para una experiencia de usuario altamente consistente y un rendimiento nativo.
  - React Native: Excelente para reutilizar código si ya existe un stack basado en React.
- Arquitectura: Similar a la SPA, la aplicación móvil seguirá una arquitectura basada en componentes. Esto facilita la reutilización de código y la gestión del estado de la aplicación, especialmente cuando se maneja el ciclo de vida de la aplicación móvil.
- Integración con SDK de Reconocimiento Facial: Durante el proceso de onboarding, la aplicación móvil integrará un SDK de reconocimiento facial para verificar la identidad del usuario. Este SDK podría ser una solución como FaceTec o Microsoft Azure Face API.
- Comunicación con el Backend: La aplicación móvil se comunicará con el API Gateway de manera similar a la SPA, utilizando HTTPS para todas las solicitudes. También manejará la autenticación mediante OAuth 2.0, siguiendo el flujo recomendado.
- Autenticación Post-Onboarding: Una vez completado el onboarding, los usuarios podrán autenticarse utilizando su huella digital, reconocimiento facial o contraseña, dependiendo de las capacidades del dispositivo y las preferencias del usuario.

7. Implementación

7.1. Pasos de Implementación

La implementación del sistema se llevará a cabo en fases para asegurar un desarrollo controlado y verificable:

Fase 1: Configuración de la Infraestructura

- Configuración inicial de la plataforma en la nube (Azure o AWS).
- Despliegue de las bases de datos necesarias (Audit DB y Customer Cache DB).
- Configuración del API Gateway con políticas de seguridad y balanceo de carga.

Fase 2: Desarrollo de Servicios Backend

- Desarrollo e implementación del Auth Service utilizando OAuth 2.0.
- Desarrollo del Transfer Service con integración a la Core Banking API.

- Desarrollo del Movement Query Service y el Notification Service.
- Pruebas unitarias y de integración de cada servicio.

Fase 3: Desarrollo de las Aplicaciones Frontend

- Desarrollo de la SPA utilizando React o Angular.
- Desarrollo de la aplicación móvil utilizando Flutter o React Native.
- Integración de ambos frontends con el API Gateway.
- Implementación de la funcionalidad de reconocimiento facial en la aplicación móvil.

Fase 4: Pruebas y QA

- Ejecución de pruebas unitarias, de integración, y de carga.
- Realización de pruebas de seguridad, incluyendo pruebas de penetración.
- Validación del cumplimiento normativo.

Fase 5: Despliegue en Producción

- Despliegue de todos los componentes en el entorno de producción.
- Configuración de herramientas de monitoreo y alerta.
- Implementación de estrategias de auto-healing y recuperación ante desastres.

7.2. Integración y Pruebas

- Pruebas Unitarias: Cada componente será probado de manera aislada para asegurar que funcione correctamente por sí solo.
- Pruebas de Integración: Las interacciones entre los componentes serán probadas para asegurar que se comuniquen correctamente y que las funcionalidades combinadas operen como se espera.
- Pruebas de Carga: Se simulará un alto número de usuarios para asegurar que el sistema pueda manejar la carga esperada sin comprometer el rendimiento.
- Pruebas de Seguridad: Se realizarán pruebas de penetración para identificar y corregir posibles vulnerabilidades.

7.3. Despliegue y Monitoreo

- Despliegue: Utilización de pipelines de CI/CD para automatizar el despliegue del código en producción. Esto incluye la configuración de entornos, la ejecución de pruebas automáticas, y la promoción de artefactos entre entornos.
- Monitoreo: Configuración de herramientas de monitoreo (Azure Monitor, AWS CloudWatch) para la observación continua de la salud del sistema. Las métricas clave incluirán la latencia, el uso de recursos, la disponibilidad, y los intentos fallidos de autenticación.

- Auto-Healing: Configuración de políticas de auto-healing en la infraestructura de la nube para reiniciar o reemplazar instancias de servicios en caso de fallos.

8. Resultados y Evaluación

8.1. Monitoreo del Sistema

- KPIs: Monitoreo de métricas clave como el tiempo de respuesta, la disponibilidad del sistema, la tasa de éxito de transacciones, y la frecuencia de fallos.
- Alertas y Respuesta: Configuración de alertas para eventos críticos. Respuesta inmediata en caso de detección de anomalías.

8.2. Revisión Post-Implementación

- Evaluación de Desempeño: Evaluación del sistema en producción para asegurar que cumple con los requisitos funcionales y no funcionales.
- Auditorías de Seguridad: Realización de auditorías periódicas para asegurar el cumplimiento continuo de las normativas de seguridad y protección de datos.