Reto

Arquitectura de Soluciones.

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)

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.

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, es libre de hacerlo.

**Consideraciones**.

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

* Ley de Protección de Datos personales N°29733
* SBS Resolución N° 504- 2021 – Enrolamiento Artículo 18: contar con al menos 2 factores de autenticación que valide la identidad del usuario:
  + Algo que sólo el usuario conoce: pregunta personalizada por el propio usuario
  + Algo que sólo el usuario posee: mensaje de texto al celular registrado del cliente o email de este.
  + Algo que el usuario es: Autenticación biométrica.

Garantice alta disponibilidad (**HA**) y tolerancia a fallos, (**DR)**, Seguridad y Monitoreo.

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

El modelo debe ser desarrollado bajo **c4**(Diagrama de Contexto, Contenedores y Componentes), describa hasta el modelo de componentes.

**Diagrama

Descripción generada automáticamente**

Tabla de decisiones de arquitectura

|  |  |  |
| --- | --- | --- |
| Componentes | Alternativas | Comentarios |
| Front – aplicación web | * Angular * React | Angular y React son dos de los frameworks más populares para el desarrollo de aplicaciones web. |
| Front – Aplicación Móvil | * Flutter * React native | Flutter y React Native son dos de los frameworks más populares para el desarrollo de aplicaciones móviles multiplataforma. Ambos frameworks tienen sus propias ventajas y desventajas, y la elección entre ellos depende de las necesidades específicas del proyecto. |
| Proveedor de identidad | * **Entra ID (Azure AD) – Oauth2.0** | **Recomendaciones:**  Para autenticación se utilizará OpenID Connect con la concesión de token: “Autorization\_code” |
| Proveedor de identidad biométrico | * **Facephi** | **Recomendaciones:**  Para las autenticaciones se subcontrataráfacephi, utilizando sus SDK dentro de las APIS internas para la autenticación. |
| Almacén de eventos (persistencia de información) | * **Apache Kafka (streaming)** * **Apache Cassandra** | **Patrón de diseño: patrón de enrutamiento dinámico**  En términos generales, el objetivo es detectar cualquier modificación en la transacción que se envíe al bus de datos (Kafka) y almacenar la información de auditoría en Cassandra, que actúa como la base de datos histórica de la transacción. De esta manera, se puede realizar un seguimiento de los cambios en la transacción y mantener un registro histórico de las mismas. |
| Arquitectura zonal | * **Azure** | En resumen, se utilizarán dos zonas geográficas para replicar la arquitectura de tal manera que se garantice la tolerancia a fallos. |
| Monitoreo de los servicios | * **Apache Kafka (streaming)** * **Jagger** * **ELK** | Al usar el bus de evento con Kafka, este se encarga de recolectar los logs de todos los micros y separarlos para realizar la ingesta en ELK para los logs y métricas, en Jagger para trazabilidad |
| Sistema de notificación | * **Send pulse Service** * **Amazon Simple Email Service** | Se escoge los siguientes proveedores de notificaciones para los clientes:  **SendPulse**: Multicanal, segmentación avanzada. **Amazon SES**: Escalabilidad AWS, integración fácil, costo por uso. Elección basada en necesidades específicas y requisitos. |
| Servicios bancarios | * **Servicios core** * **Servicios independientes** * **Servicios de notificación** * **Servicios de monitoreo** * **Servicios de autorización** | La comunicación entre los microservicios que se tienen en la plataforma bancaria.   * Rest - JSON. |

**Simbología usada**

