DEVSU

Ejercicio Práctico para Arquitecto de Soluciones

Miguel Cartagenova

03/09/2025

1. Resumen Ejecutivo

La entidad BP busca implementar un sistema integral de banca en línea que garantice seguridad, disponibilidad y escalabilidad para sus clientes. El alcance incluye consulta de movimientos, transferencias entre cuentas propias y de terceros, y pagos interbancarios, asegurando trazabilidad completa y cumplimiento normativo.

La solución debe soportar la integración de datos desde dos sistemas principales (Core Bancario y sistema complementario), ofrecer notificaciones obligatorias a través de múltiples canales (mínimo dos proveedores) y habilitar un proceso de onboarding digital con validación biométrica facial, orientado principalmente al canal móvil.

La arquitectura propuesta se fundamenta en los siguientes principios:

- Alta disponibilidad y recuperación ante desastres, garantizando continuidad del negocio mediante despliegues en múltiples regiones.
- Seguridad y cumplimiento regulatorio, aplicando estándares internacionales como PCI DSS, SOC2 y la Ley de Protección de Datos Personales, junto con autenticación y autorización basada en OAuth2.0/OpenID Connect.
- Escalabilidad y rendimiento, mediante el uso de microservicios desacoplados, el patrón CQRS para optimizar consultas frecuentes y event-driven architecture para procesos como transferencias y notificaciones.
- Flexibilidad tecnológica, con una SPA en React y una aplicación móvil en Flutter, ambas integradas a través de un API Gateway en Azure que expone servicios de negocio de forma segura y controlada.
- Observabilidad y monitoreo proactivo, mediante Azure Monitor, Application Insights y Grafana, para garantizar el cumplimiento de los SLO definidos en disponibilidad y rendimiento.

La topología multi-región y el uso de Redis/Cosmos con particionado garantizan **baja latencia** para consultas frecuentes y alta concurrencia.

En conjunto, esta propuesta entrega a BP una plataforma moderna y preparada para el crecimiento, que asegura la mejor experiencia para el cliente final y cumple con los más altos estándares de la industria financiera.

2. Decisiones Arquitectónicas (con justificación teórica)

2.1 Frontend (SPA y Mobile)

Opciones evaluadas:

- **SPA (Single Page Application):** Angular vs React.
- **Mobile multiplataforma:** Flutter vs React Native.

Decisión:

- **React** para la aplicación web (SPA), debido a su madurez, amplio ecosistema de librerías y soporte empresarial comprobado.
- **Flutter** para la aplicación móvil, por su rendimiento cercano al nativo, consistencia visual entre plataformas y respaldo directo de Google.

Justificación:

1. **React (SPA)** asegura escalabilidad, modularidad y una comunidad robusta, lo que facilita la incorporación de nuevas funcionalidades de forma ágil.
2. **Flutter (Mobile)** permite mantener una sola base de código para iOS y Android, reduciendo costos de mantenimiento y tiempos de despliegue.
3. El motor gráfico de Flutter (Skia) garantiza interfaces consistentes y de alto rendimiento, algo clave en aplicaciones bancarias donde la experiencia de usuario debe ser fluida y confiable.
4. Ambas tecnologías cuentan con soporte a largo plazo y una alta tasa de adopción en el sector financiero, lo que mitiga riesgos de obsolescencia.

2.2 Seguridad y Autenticación

Patrón propuesto:

OAuth2.0 con OpenID Connect (Authorization Code + PKCE).

Proveedor de identidad:

- **Azure AD B2C** como opción principal, por su gestión en la nube, integración nativa con Azure y cumplimiento normativo en entornos financieros.

- **Keycloak** como alternativa on-premises, en caso de requerir despliegues locales o híbridos.

Flujos de autenticación recomendados:

- **SPA (React):** Authorization Code + PKCE implementado bajo el patrón BFF (Backend-for-Frontend), lo que asegura que los tokens no queden expuestos en el navegador y se gestionen en un servidor intermedio confiable.
- **Aplicación Móvil (Flutter):** Authorization Code + PKCE utilizando AppAuth, con soporte para refresh tokens y rotación automática, garantizando sesiones seguras en dispositivos móviles.

Scopes y control de acceso (RBAC):

- Definición granular de scopes por dominio de negocio:
  - movimientos: read
  - transferencias: write
  - pagos: execute
- Roles diferenciados para clientes finales y operadores internos, aplicando el principio de mínimo privilegio.

Medidas de hardening y seguridad avanzada:

1. Token binding y validación estricta de audiencias y emisores.
2. Rotación automática de claves gestionada con Azure Key Vault.
3. Detección de anomalías e inicios de sesión riesgosos con Microsoft Defender for Identity y aplicación de políticas adaptativas.
4. Soporte a MFA (multi-factor authentication) en operaciones críticas y autenticación biométrica integrada en el flujo de onboarding y acceso móvil.

Justificación:

- Se adopta un estándar de la industria ampliamente soportado, garantizando interoperabilidad con distintos proveedores de identidad y cumplimiento regulatorio (PCI DSS, GDPR, LPDP).
- El uso de PKCE y BFF en la SPA mitiga riesgos como token hijacking o XSS, asegurando que las credenciales nunca estén expuestas en el navegador.
- La rotación de refresh tokens y la gestión segura de claves reducen significativamente la superficie de ataque en dispositivos móviles.

- El diseño de scopes y RBAC aporta trazabilidad, control granular de accesos y facilidad de auditoría en entornos regulados.
- **Importante:** no se implementa la lógica completa de OAuth2.0, sino que se configura el producto de autenticación ya disponible en la organización, ajustándolo con los flujos y controles de seguridad más adecuados al estándar.

2.3 Persistencia y Auditoría

Decisiones:

- Base transaccional (OLTP): Azure SQL Database (consistencia ACID, constraints, índices compuestos y potencial particionado por cliente/fecha).
- Auditoría de acciones: Azure Cosmos DB (API Core/SQL) para alto volumen de escritura y baja latencia, con partición por tenantId/customerId y TTL por política de retención.
- Clientes frecuentes / performance: CQRS + Cache (Azure Cache for Redis) y materialized views (p. ej., movimientos recientes por cliente) para respuestas sub-segundo.
- Integración fiable entre dominios: Outbox Pattern en servicios de dominio (SQL) con publicación a Service Bus/Event Grid para proyección hacia Cosmos DB (auditoría) y actualización de materialized views.
- Captura de cambios opcional: Change Data Capture (CDC) en SQL para sincronizaciones/ETL regulatorias.

Patrones aplicados:

- **CQRS:** separación de escrituras/commands (SQL) de lecturas/queries (Redis + vistas materializadas) para endpoints de alta demanda.
- **Event Sourcing (auditoría):** eventos inmutables de negocio en Cosmos DB (quién, qué, cuándo, desde dónde, correlationId, payload mínimo no sensible) con capacidad de reconstrucción.
- **Outbox + Retries + Idempotencia:** entrega *exactly-once* lógica en proyecciones y notificaciones.

Auditoría + “persistencia para clientes frecuentes”:

- Auditoría: tabla/event store independiente (Cosmos DB o SQL dedicado) con Event Sourcing de acciones del cliente.

- Clientes frecuentes / performance: CQRS + Caché y materialized views (p. ej., movimientos recientes por cliente) para respuestas sub-segundo.
- Trazabilidad: correlación por CorrelationId, inmutabilidad y sello de tiempo; panel de auditoría en App Insights + Log Analytics.

Modelo de datos (alto nivel):

- SQL (OLTP): Clientes, Cuentas, Movimientos (normalizado), Transferencias (estado, sagaId), índices por (customerId, fecha) y stored procedures para operaciones críticas.
- Cosmos DB (Auditoría/Event Store): contenedor AuditEvents con partitionKey=customerId e id=eventId; atributos: type, timestamp, actor, ip/device, correlationId, result.
- Materialized Views (SQL): Movimientos Recientes Por Cliente actualizado por workers que consumen eventos/CDC.
- Redis: claves tipo movs:{customerId}:{cuenta}:{yyyyMM} con expiración controlada.

Seguridad y cumplimiento de datos:

- Cifrado en reposo (TDE en SQL, cifrado en Cosmos) y en tránsito (TLS 1.2+).
- Enmascaramiento dinámico / column-level security para PII (información personal identificable) en SQL; minimización de datos en auditoría (sin PAN/secretos).
- Retención/purgado por política y legal hold cuando aplique; gestión de secretos y claves en Azure Key Vault; acceso mediante Private Endpoints.

Alta disponibilidad / Disaster Recovery y rendimiento:

- SQL: Business Critical/GP con zone redundancy y geo-replicación; índices por patrón de consulta; Query Store y autotuning.
- Cosmos DB: multi-region write si se requiere latencia global y autoscaling de RU/s.
- Redis: clustering con réplica y eviction por LRU.
- Backpressure: colas (Service Bus) para absorber picos; reintentos exponenciales e idempotencia en consumidores.

Observabilidad:

- Trazabilidad end-to-end con correlationId propagado desde el API Gateway.
- App Insights + Log Analytics: métricas de RU/s (Cosmos), vCores/DTU (SQL), latencia P95/P99, fallos de proyección y lag de colas.

Justificación:

1. **Separación de preocupaciones**: OLTP en SQL (consistencia y transacciones) y auditoría/eventos en Cosmos (escrituras baratas, latencia baja) mejora rendimiento y escala.

2. **Escalabilidad horizontal:** Cosmos particionado + CQRS con Redis/materialized views soporta clientes frecuentes y consultas sub-segundo sin recargar el OLTP.
3. **Trazabilidad total:** Event Sourcing + Outbox garantiza reconstrucción y auditoría forense; facilita cumplimiento (PCI/LOPDP).
4. **Resiliencia:** colas + reintentos + idempotencia evitan pérdidas o duplicados ante picos y fallos parciales.

La información auditada se consume también desde la **Capa de Integración**, permitiendo exponer consultas seguras y agregadas a través del API Gateway sin comprometer la fuente transaccional.

2.4 Onboarding Biométrico (mobile-first)

Objetivo:

Habilitar un alta 100% digital y segura del cliente: captura biométrica con detección de vivacidad (liveness), verificación de identidad contra documento oficial, validación regulatoria KYC/AML, y firma electrónica de los contratos/consentimientos con garantías legales equivalentes a la firma manuscrita (QES). Para reducir fraude y asegurar la identidad, el flujo se apoya en Azure Face Liveness + Face Verification y Azure Document Intelligence para extraer/validar datos del documento de identidad.

Tecnología:

- **Biometría y liveness assurance:** Azure AI Face (detección de vivacidad y verificación rostro vs foto del documento).
- **Extracción de datos del documento:** Azure Document Intelligence – ID model (OCR estructurado de cédulas/pasaportes).
- **Firma electrónica y evidencia de identidad: Uanataca** (Prestador de Servicios de Confianza / QTSP) para **QES**, con opciones de **video-identificación** y “one-shot signature” (certificado de un solo uso) cuando aplique; incluye sellado de tiempo y validación/long-term (LTV).
- **Validación de datos frente a cédula digital ecuatoriana**: Integración (vía proveedor autorizado o convenio) con servicios de verificación biométrica/identidad del Registro Civil, contrastando número de cédula y datos extraídos por OCR con fuentes oficiales. *(La disponibilidad/alcance depende de acuerdos regulatorios y del proveedor habilitado.)*

- **Firma de documentos de onboarding con Uanataca:** Emisión/uso de certificados y **firma electrónica acreditada**, con evidencia de video-identificación si se requiere elevación de garantías (equivalencia legal a firma manuscrita bajo el DAS/QES; en Ecuador, conforme a normativa local y acreditaciones vigentes del prestador).

Flujo propuesto (alto nivel):

1. **Pre-checks**: consentimiento de tratamiento de datos, verificación de dispositivo, captura de correlationId.
2. **Captura de documento** (frontal/reverso) + **OCR estructurado** (nombres, número de documento, fecha de expiración) .
3. **Selfie + Liveness** (pasivo/activo) y **comparación rostro vs. documento** (Face Verification).
4. **Cruces KYC/AML** (listas de sanciones, PEP) + validación con fuente oficial (cédula digital/Registro Civil vía proveedor autorizado).
5. **Decisión** (aprobación automática por score o ruta de revisión asistida).
6. **Provisioning de identidad** en **Azure AD B2C** (alta de cuenta, MFA, enrolamiento biométrico del dispositivo o **passkeys**).
7. **Firma electrónica** de contrato/TC/consentimientos con **Uanataca** (QES/one-shot):
    - Se presenta el **PDF**; el usuario firma; se aplica certificado cualificado, **sello de tiempo** y **LTV**.
    - Evidencias (video-ID/huellas criptográficas) quedan asociadas al expediente.
8. **Cierre y auditoría**: se emite evento de **onboarding.success**; se registra en event store (inmutable), con hash del documento firmado; se notifica por los canales configurados.

Arquitectura (componentes):

- **Mobile App (Flutter):** SDK de captura (documento + rostro), guías anti-spoofing, cifrado local de tránsito (TLS).
- **Onboarding API** (microservicio): orquesta el flujo (estados), invoca **Face Liveness/Verify**, **Document Intelligence**, proveedor KYC/AML, y APIs de firma **Uanataca**; publica eventos a **Service Bus** (para auditoría, notificación, analítica).

- **Identity Provider: Azure AD B2C** (OIDC/OAuth2) como fuente de identidad del canal digital.
- **Firma/QES:** integración vía API/rest con **Uanataca** (QES, one-shot, timestamp/LTV).
- **Persistencia:** datos operativos en **SQL**; **auditoría** de eventos y evidencias mínimas en **Cosmos DB**; documentos firmados en repositorio cifrado (hash en auditoría).

Seguridad, privacidad y cumplimiento:

- **Mínima retención** de PII; cifrado **at-rest** (AES-256/TDE) y **in-transit** (TLS 1.2+); secretos en **Key Vault**; endpoints privados.
- **Controles antifraude:** liveness + comparación facial + verificación documental + trazabilidad (correlationId, sello de tiempo).
- **Cumplimiento**: LOPDP (Ecuador), PCI DSS (si aplica a datos de pago); para firma, **QES** (eIDAS) y evidencias de **video-identificación** cuando se requiera mayor nivel de garantía.

KPIs y operación:

- **Tiempos objetivo:** onboarding **≤ 3 min** P95 (red estable), liveness < 5 s, OCR < 3 s, firma < 30 s.
- **Calidad biométrica:** thresholds de match score y liveness score con política de reintentos y fallback a verificación asistida.
- **Observabilidad:** tableros en App Insights (latencia P95, tasa de aprobados, fraude detectado), auditoría consultable por expediente.

2.5 Manejo de Notificaciones obligatorias

Objetivo:

Garantizar el envío confiable, auditable y multicanal (email, SMS, push) de eventos críticos (apertura de cuenta, transferencias, pagos, alertas de seguridad), cumpliendo la normativa de notificación obligatoria y manteniendo continuidad de servicio con proveedores redundantes.

Diseño y componentes:

- **Servicio "Notificaciones" (microservicio dedicado):**
    - Expone un API interno (REST) y consume eventos desde la plataforma (Service Bus/Event Grid).
    - Aplica estrategia de proveedores:
        - **Email:** SendGrid (primario) ↔ **SMTP corporativo** (secundario).
        - **SMS:** Twilio (primario) ↔ Proveedor local/operadora (secundario).
        - **Push: Azure Notification Hubs** (FCM/APNS), con fallback a email/SMS si falla el push.
    - **Plantillas** centralizadas (versionadas) por idioma (dialecto indígena) y tipo de evento.
- **Ruteo y políticas:**
    - **Preferencia del cliente** (opt-ins, canales permitidos) y **tipo de evento** (seguridad → multicanal obligatorio).
    - **Reglas de horario** silencio nocturno para marketing; mensajería permenente para seguridad.
    - **Idempotencia** por notificationId para evitar duplicados.

Patrones de resiliencia y entrega:

- **Outbox Pattern:** los eventos de negocio (transfer_completed, login_risky, etc.) se guardan en Outbox (transaccional con la operación) y luego se publican de forma confiable al bus de servicios.
- **Retries con backoff exponencial** y **dead-letter** para mensajes que no se pueden entregar.

- **Circuit Breaker / Bulkhead:** aísla fallas de un proveedor; conmuta a fallback sin afectar el resto del sistema.
- **Rate limiting** y **throttling** por proveedor para evitar bloqueos.

Auditoría y trazabilidad:

- **Registro inmutable** de cada notificación en **Auditoría** (Cosmos DB / SQL dedicado): notificationId, correlationId, canal, proveedor, payload-minimizado (sin PII sensible), timestamps, estado (enviado/entregado/fallido), código de respuesta del proveedor.
- **Metricas de entrega** (envíos, entregas, rebotes, latencia P95) en App Insights + Log Analytics.
- **Conservación de evidencias** según política (p. ej., 2–5 años para eventos normativos).

Seguridad y cumplimiento:

- Contenido mínimo necesario (privacy by design).
- Cifrado in-transit (TLS 1.2+) y secrets en Key Vault.
- Control de consentimiento (opt-in/opt-out por canal), y etiquetas de datos personales para gobernanza.
- Política de reintentos para mensajes normativos (seguridad/finanzas) con confirmación obligatoria por al menos un canal alterno.

KPIs propuestos:

- **Disponibilidad** del servicio de notificaciones ≥ **99.9%**.
- **Tasa de entrega**: Email ≥ 98%, SMS ≥ 97%, Push ≥ 95% (dependiente de plataformas).
- **Latencia P95**: < 5 s SMS, < 10 s Email, < 3 s Push desde la emisión del evento.
- **Fallback automático** exitoso ≥ 99% cuando falle el proveedor primario.

Justificación:

1. **Cumplimiento normativo**: notificación de operaciones sensibles y eventos de seguridad por **canales redundantes**.
2. **Continuidad y resiliencia**: mínimo dos proveedores por canal con fallback transparente (Circuit Breaker + Retries).
3. **Trazabilidad**: Outbox + Auditoría aseguran evidencia verificable, correlacionada con la transacción (correlationId).
4. **Performance y costo**: uso de Notification Hubs para push a gran escala; SendGrid/SMTP balancea costo/entrega; Twilio garantiza alcance global con posibilidad de proveedor local.

Todas las notificaciones quedan registradas en la capa de **Auditoría** mediante eventos inmutables correlacionados con la transacción original, lo que asegura trazabilidad end-to-end.

2.6 Capa de integración y servicios iniciales

1. API Gateway e Ingreso seguro

Gateway: Azure API Management (APIM) delante de los microservicios.
Front-door/WAF: Azure Application Gateway (WAF) expuesto a internet → **APIM** (modo interno/privado) → microservicios por **Private Endpoints**.

Responsabilidades de APIM:

AuthN/AuthZ perimetral: validación **JWT** (OIDC con Azure AD B2C), verificación de issuer/audience, expiración y scopes.

Throttling / Rate limiting: protección anti-abuso y control de burst por suscripción/cliente.

Transformaciones y normalización: header/claim injection, reescritura de rutas, mapeo de códigos de estado, masking de PII.

Versionado y contratos: v1, v2 por API; políticas de **deprecation** y compatibilidad hacia atrás.

Observabilidad: correlación (CorrelationId), logging hacia **App Insights**/**Log Analytics**; métricas por operación.

Seguridad de transporte: TLS 1.2+, certificados/secretos en **Key Vault**, llamadas salientes por **Private Link**.

Nota: App Gateway (WAF) mitiga OWASP Top-10; APIM aplica políticas finas (validate-jwt, rate-limit, set-backend-service, rewrite-uri).

2. Patrones en la capa de integración

BFF (Backend-for-Frontend) para la **SPA**: sesión segura, tokens en servidor, composición de datos (Core + sistema complementario) sin exponer llaves ni lógica de negocio al browser.

Aggregator / API Composer: endpoints de lectura que unen respuestas del Core bancario y del sistema complementario, con caché Redis y expiraciones cortas para disminuir latencia.

Anti-Corruption Layer (ACL): aislamiento de modelos del Core; DTOs y mapeos que evitan "filtrar" decisiones del sistema legado al dominio nuevo.

Resiliencia: Circuit Breaker, Timeouts, Bulkhead, Retries con backoff, idempotency keys para operaciones de escritura.

Mensajería y consistencia: Outbox en servicios de dominio + Service Bus/Event Grid para eventos de negocio (notificaciones, auditoría, proyecciones CQRS).

3. Servicios iniciales

a) Servicio Básicos de Cliente:

Responsabilidad: Unificar datos del **Core** y del **sistema complementario** (perfil, estado KYC, límites, preferencias).

Lecturas: GET /clientes/{id} (compuesto).

Caché: Redis por clienteId (TTL 30–120s) para minimizar llamadas repetidas.

Seguridad: scope clientes:read.

Observabilidad: CorrelationId desde APIM, métricas de latencia y tasa de aciertos de caché.

Errores: degradación elegante, si el complementario falla, devolver Core + warnings.

b) Servicio Movimientos (CQRS + Cache):

Responsabilidad: consultas de movimientos **rápidas** para "clientes frecuentes".

Lecturas (Query model):

GET /cuentas/{id}/movimientos?desde&hasta&limit (optimizado sobre **vista materializada** + **Redis**).

Escrituras (Command model): registro/ingestión vía dominio contable (SQL ACID).

CQRS: proyección desde Outbox/CDC a **vista materializada** (SQL) y calentamiento de **Redis**.

Seguridad: scope movimientos:read.

SLO: P95 < **300 ms** (cache hit), < **800 ms** (cache miss).

c) Servicio Transferencias (Saga/Orquestación):

Responsabilidad: orquestar transferencias intra/interbancarias con consistencia y compensaciones.

Flujo (saga):

Validación (saldo, límites, AML)

Reserva de fondos

Ejecución (Core/clearing)

Confirmación

Notificación (evento transfer.completed)

Idempotencia: Idempotency-Key por solicitud; reintentos seguros.

Eventos: publica transfer.started/transfer.completed/transfer.failed a **Service Bus** (alimenta **Notificaciones** y **Auditoría**).

Seguridad: scopes transferencias:write, MFA para montos/operaciones sensibles.

SLO: P95 < **2 s** intra; interbancario sujeto a clearing, con estados intermedios.

Extensiones siguientes: **Pagos** (servicio dedicado, similar a Transferencias) y **Catálogos** (monedas, bancos, motivos, límites) con **Redis** para baja latencia.

4. Contratos, versionado y gobernanza

- **Contratos OpenAPI** publicados en **APIM**; linting pre-merge.
- **Versionado** por ruta/cabecera; **política de deprecación** y guía de migración.
- **Schema Registry** (p. ej., JSON Schema compartido) para eventos y DTOs.
- **Backwards compatibility** obligatoria en lecturas; *breaking changes* encapsulados detrás del BFF cuando sea posible.

5. Idempotencia, errores y cumplimiento:

- **Idempotencia** en POST/PUT (clave por cliente+hash payload), respuestas 200/409 coherentes.
- **Errores normalizados** (traceId, correlationId, code, detail, hint).
- **PII mínima** en respuestas y logs (masking); **retención** y **derecho al olvido** en línea con LPDP/GDPR.

6. Observabilidad end-to-end:

- **Trazas distribuidas** (APIM → BFF/Aggregator → microservicios) con CorrelationId.
- **Tableros**: latencia P95 por endpoint, tasa de fallos, % cache hit, colas en Service Bus, eventos de notificación/auditoría.
- **Alertas**: saturación de caché, circuit breaker abierto, fallos de proveedor (Twilio/SendGrid), RU/s Cosmos, DTU/vCore SQL.

Justificación:

- **Separación clara de responsabilidades:** el **Aggregator** y el **BFF** simplifican el consumo desde web/móvil y desacoplan del Core y del complementario.
- **Rendimiento y experiencia: CQRS + Redis** entrega lecturas sub-segundo para movimientos y paneles del cliente.
- **Consistencia y resiliencia: Saga** con reservas/compensaciones evita estados inválidos en transferencias; **Outbox + mensajería** garantiza entrega y auditoría.
- **Seguridad/compliance:** validación JWT en APIM, WAF, PII mínima, MFA en operaciones críticas, auditoría inmutable de eventos.

Todos estos servicios y patrones de integración se soportan sobre una **infraestructura en la nube** diseñada para ofrecer alta disponibilidad, seguridad y eficiencia operativa, detallada en el siguiente apartado.

2.7 Infraestructura y Nube

Justificación:

La definición de la infraestructura en la nube debe responder a tres pilares estratégicos del negocio bancario: seguridad, resiliencia y eficiencia operativa.

En un entorno altamente regulado, donde la disponibilidad del servicio y el cumplimiento normativo son críticos, la arquitectura en la nube permite alinear los requerimientos de alta disponibilidad, continuidad del negocio y auditoría, con la flexibilidad de escalar recursos bajo demanda y optimizar costos.

Proponemos adoptar Azure como plataforma principal por su madurez en servicios financieros, su ecosistema de seguridad y cumplimiento certificado, y la integración nativa de herramientas de observabilidad y DevSecOps.

La elección de cada componente, desde gateways, cómputo, datos y mensajería, hasta monitoreo e infraestructura como código, responde a la necesidad de construir un sistema resiliente, auditable y preparado para el crecimiento, minimizando riesgos y garantizando trazabilidad end-to-end que refuerza lo revisado en los puntos anteriores.

Topología y principios:

- **Dos regiones (activo-activo)** con zonas de disponibilidad: continuidad ante fallas regionales y mantenimientos; RTO/RPO ajustables por dominio (OLTP vs auditoría).
- **VNET única con subredes por dominio** (edge, integración, datos, gestión) y Private Endpoints a PaaS: reduce superficie de ataque, evita exposición pública.
- **Zero-Trust**: WAF perimetral, validación JWT en el gateway, mínimos privilegios, secretos en Key Vault, segmentación de red.

Capa de entrada / Seguridad perimetral:

QUÉ:

- **Azure Application Gateway (WAF)** al frente (OWASP Top-10, TLS, mTLS opcional).
- **Azure API Management (APIM)** detrás del WAF (modo interno) como **gateway de APIs**.

POR QUÉ:

- WAF mitiga ataques comunes y descarga TLS.

- APIM aporta **validate-jwt**, **rate limiting**, **versionado**, **transformaciones**, **cuotas** y **observabilidad** sin re-desplegar los servicios.

Cómputo / servicios de aplicaciones:

QUÉ:

- **App Service** para BFF/aggregators y microservicios livianos (autoscaling + slots).
- **AKS** (opcional/por etapa) para cargas **state-less** que requieran **autoscaling agresivo**, sidecars, o **blue/green** a gran escala.

POR QUÉ:

- **App Service** minimiza operación (PaaS), ideal para un MVP robusto.
- **AKS** se activa cuando la complejidad/escala lo amerita (tráfico pico, cost-per-request optimizado, políticas finas con Ingress/sidecars).

Integración y mensajería:

QUÉ:

- **Azure Service Bus** para orquestaciones (Saga), **Outbox**, reintentos, **dead-letter**.
- **Event Grid** para eventos de dominio ligeros/reativos (fan-out).

POR QUÉ:

- **Service Bus** garantiza entrega confiable, orden y idempotencia (clave en Transferencias/Notificaciones).
- **Event Grid** simplifica fan-out/automatización sin acoplar productores y consumidores.

Datos operativos, cache y auditoría:

QUÉ:

- **Azure SQL Database** (OLTP ACID) para dominios transaccionales.
- **Azure Cache for Redis** para clientes frecuentes (lecturas sub-segundo).
- **Cosmos DB** (Core SQL) como event store/auditoría de acciones y trazas inmutables.

- **Azure Storage** (Blob) para documentos firmados y adjuntos.

POR QUÉ:

- SQL ofrece consistencia fuerte y integridad referencial.
- Redis evita cargar el OLTP y baja la latencia de “Movimientos”.
- Cosmos DB escala escrituras/auditoría con particionado y baja latencia global.
- Blob es económico, con WORM/immutability y SAS para compartición controlada.

Identidad, secretos y cumplimiento:

QUÉ:

- **Azure AD B2C** (o **Keycloak** on-prem) como IdP.
- **Azure Key Vault** para llaves, certificados y secretos.
- **Defender for Cloud** + **Microsoft Purview** (opcional) para gobierno y postura.
- **Azure Policy / Blueprints** para **landing zone** segura (guardrails).

POR QUÉ:

- IdP gestionado: **OIDC/OAuth2** estandarizado, MFA, políticas adaptativas.
- Key Vault centraliza y rota secretos (KMS), requisito de **PCI/LOPDP**.
- Defender y Purview elevan visibilidad, clasificación de datos y cumplimiento.
- Azure Policies evita que nos salemos: cifrado forzado, endpoints privados, etiquetas, RBAC.

Observabilidad y confiabilidad:

QUÉ:

- **Azure Monitor + Application Insights + Log Analytics** (telemetría unificada).
- **Dashboards** (Grafana/Power BI) para KPIs (latencia P95, tasa de entrega, RU/s/DTU, backlog de colas).
- **Alertas** por SLO (errores, circuit breaker, RU/s límite, DTU, TTL de caché).

POR QUÉ:

- **Trazas distribuidas** con CorrelationId de extremo a extremo (APIM→BFF→servicios).

- **Detección temprana** de degradaciones y **forense** para auditoría/reguladores.

Entrega continua e Infraestructura como Código (IAC):

QUÉ:

- **Azure DevOps/GitHub Actions** con pipelines CI/CD (build, test, SAST/DAST, IaC).
- **Terraform/Ansible** para IaC y config idempotente (APIM, App Service, Cosmos, SQL, Redis, VNets).
- **Slots** y **feature flags** (App Configuration) para despliegues seguros.

POR QUÉ:

- Reproducibilidad, **auditoría de cambios**, y **rollback** confiable.
- Menor tiempo al mercado con controles de calidad y **seguridad shift-left**.

Alta Disponibilidad /Disaster Recovery y costos

QUÉ:

- **Geo-replicación** (SQL, Cosmos), **zone redundancy**, **backup automático** y pruebas de restauración.
- **Autoscaling** (reglas por CPU/RPS/colas), **reservas**/Savings Plans donde aplique.

POR QUÉ:

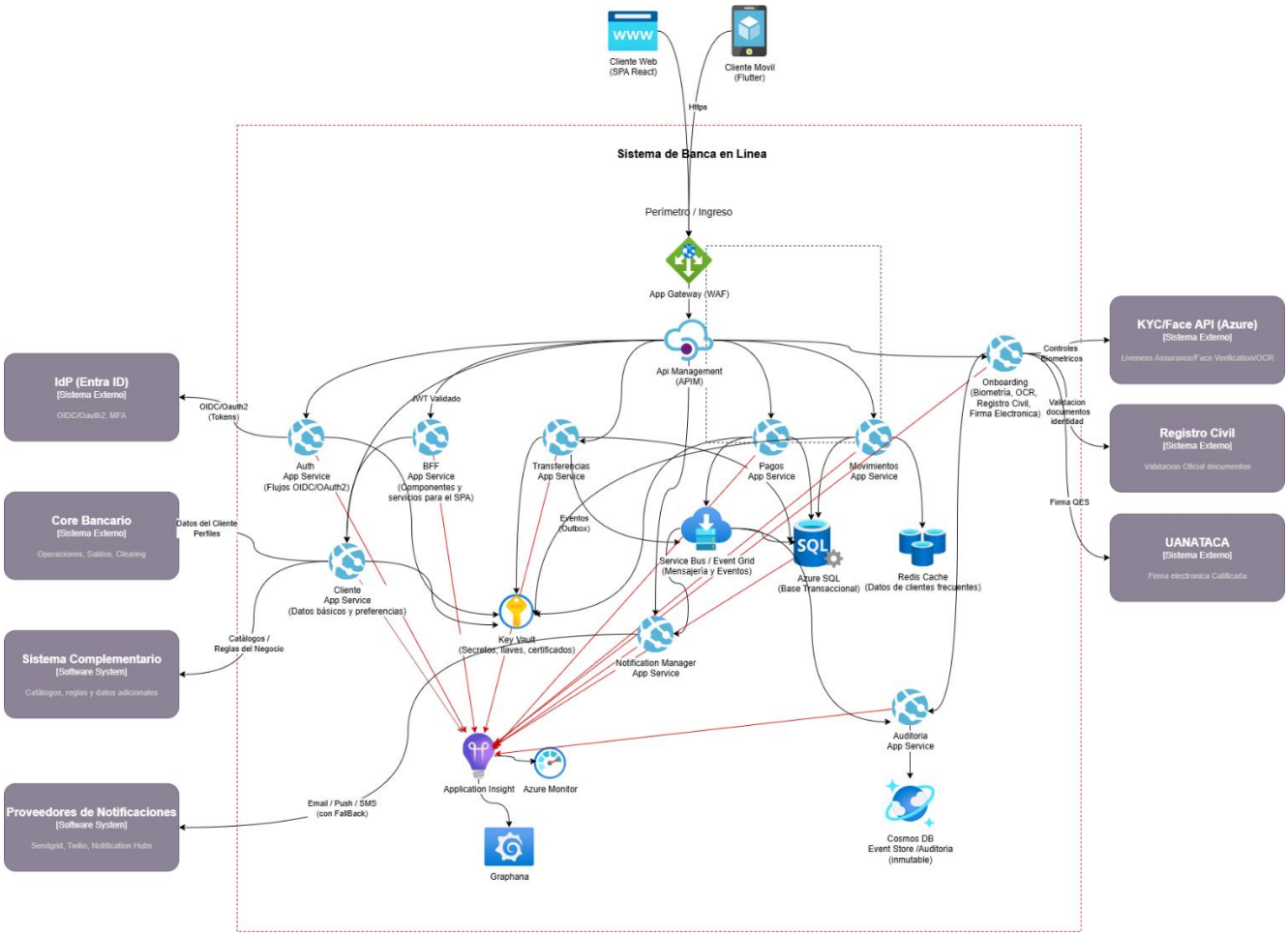
- Asegura **continuidad** con RTO/RPO alineados al negocio.
- **Cost-to-serve** optimizado: escalar cuando hay demanda, ahorrar en valle.

Decisiones técnicas resumidas:

- **Gateway**: App Gateway (WAF) + **APIM** → seguridad, versionado, throttling.
- **Compute**: App Service hoy; **AKS** cuando la escala/patrón lo exijan.
- **Mensajería**: Service Bus (fiabilidad) + Event Grid (fan-out).
- **Datos**: SQL (OLTP), **Redis** (cache), **Cosmos** (auditoría/eventos), **Blob** (documentos).

- **Seguridad**: B2C/Keycloak, **Key Vault**, Private Endpoints, Defender, Policies.
- **Ops**: Monitor/App Insights, Grafana/Power BI, alertas por SLO.
- **DevOps/IaC**: Azure DevOps/GitHub + Terraform/Ansible, slots/flags.
- **HA/DR/Costos**: multi-región, backups, autoscaling, reservas.

Diagrama infraestructura:



3. Diagramas C4

Introducción:

Para complementar las decisiones arquitectónicas descritas en los capítulos anteriores, se emplea el modelo C4 (Contexto, Contenedores, Componentes y Código) como estándar visual para representar la arquitectura propuesta.

El enfoque C4 permite mostrar la solución en distintos niveles de detalle:

- Contexto (C1): la visión global del sistema y cómo interactúa con actores y sistemas externos.
- Contenedores (C2): los principales bloques de software, aplicaciones y servicios de infraestructura que conforman la solución.
- Componentes (C3): la descomposición interna de microservicios clave, resaltando sus responsabilidades y colaboraciones.

Estos diagramas ayudan a comunicar la arquitectura de forma clara y consistente a perfiles técnicos y de negocio, facilitando la comprensión de los límites del sistema, sus dependencias y la trazabilidad entre requerimientos y decisiones tecnológicas.

3.1 C1 – Contexto - Sistema de Banca en Línea

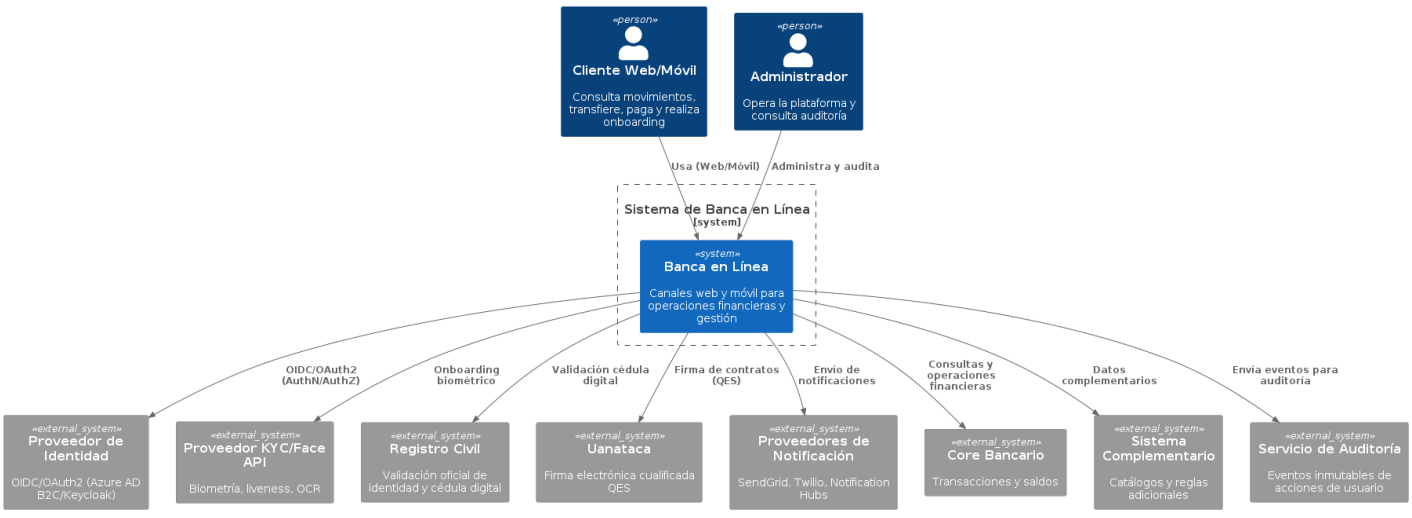
Descripción:

En el nivel de contexto se presenta una visión global del sistema de banca en línea y cómo interactúa con los diferentes actores y sistemas externos. Los clientes web y móviles acceden a los servicios para consultar movimientos, realizar transferencias, efectuar pagos y completar procesos de onboarding digital. Los administradores operan la plataforma y consultan registros de auditoría para control y cumplimiento.

El sistema se integra con un proveedor de identidad (Azure AD B2C o Keycloak) que gestiona la autenticación y autorización bajo estándares OAuth2.0/OpenID Connect. Durante el onboarding, se conecta con servicios de KYC/biometría (Azure Face API, OCR) y con el Registro Civil para la validación oficial de la cédula digital, además de apoyarse en Uanataca para la firma electrónica cualificada de documentos.

Asimismo, la solución interactúa con el Core Bancario y un sistema complementario, que proveen datos financieros y catálogos de referencia. Finalmente, los proveedores de notificación (SendGrid, Twilio, Notification Hubs) garantizan la entrega de alertas y mensajes obligatorios a los clientes.

Este nivel muestra el ecosistema completo y define los límites de la solución, asegurando que todos los flujos críticos de seguridad, negocio y cumplimiento estén claramente identificados desde el inicio.



3.2 C2 – Contenedores

Descripción:

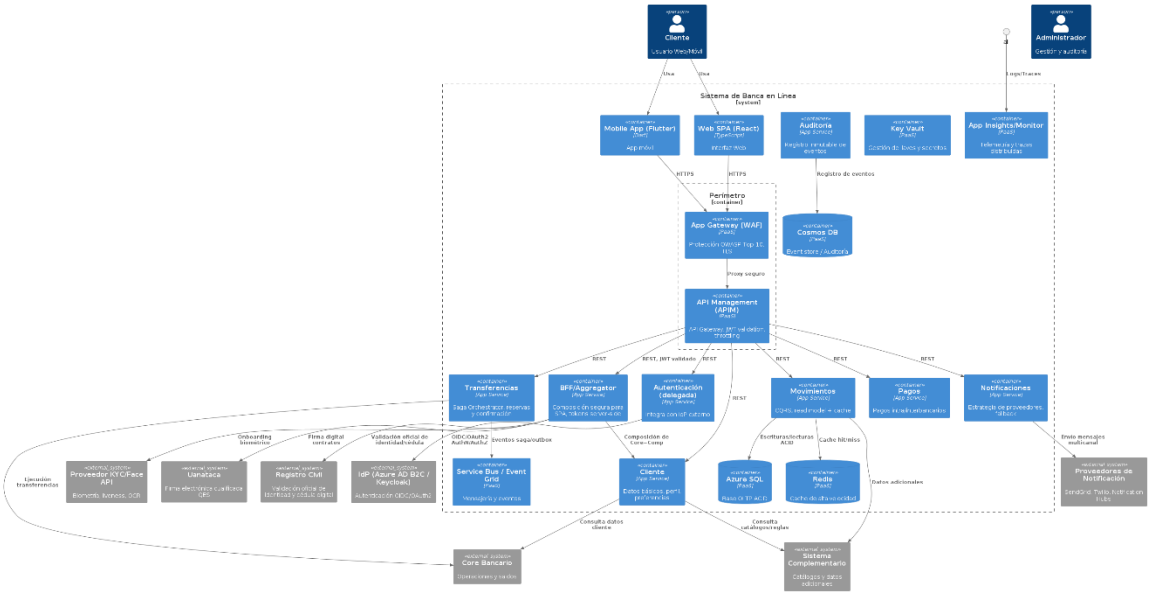
En el nivel de contenedores se describen los principales bloques de software y servicios que conforman el sistema, así como las dependencias externas críticas. La arquitectura está organizada en torno a un **API Gateway (Azure API Management)** que expone de manera segura los microservicios a los canales web y móvil.

Los **microservicios internos** se dividen por dominio de negocio: Autenticación (delegada al IdP), Cliente, Movimientos (con CQRS y Redis), Transferencias (implementando patrón Saga), Pagos, Notificaciones y Auditoría. Estos se apoyan en **servicios de datos** (Azure SQL, Cosmos DB, Redis) y **plataformas de mensajería/eventos** (Service Bus/Event Grid).

La **integración bancaria** se realiza a través de conectores hacia el **Core Bancario** y al **Sistema Complementario**, que son las fuentes primarias de información financiera y catálogos. Asimismo, el sistema se conecta con **proveedores externos**: el **IdP (Azure AD B2C/Keycloak)** para la autenticación, servicios de **KYC/Face API** para el onboarding biométrico, **Uanataca** para firma digital cualificada de documentos, y **proveedores de notificación** (SendGrid, Twilio, Notification Hubs) para el envío multicanal de alertas obligatorias.

Además de los microservicios internos y proveedores globales, la arquitectura contempla la integración con el Registro Civil como sistema externo autorizado para la validación de identidad y cédula digital del cliente. Este flujo ocurre durante el proceso de onboarding biométrico: tras la captura de datos y verificación facial, el sistema consulta al Registro Civil para contrastar la información oficial del ciudadano.
De esta manera, la solución no solo garantiza seguridad técnica, sino también cumplimiento normativo local, asegurando que la identidad validada sea legalmente reconocida.

Esta distribución de responsabilidades garantiza seguridad, trazabilidad y escalabilidad, permitiendo que cada microservicio evolucione de manera independiente y asegurando el cumplimiento normativo.



3.3 Diagrama de Componentes

Descripción:

En el nivel de **componentes** se detalla la estructura interna de los microservicios clave. El objetivo es mostrar cómo cada servicio encapsula responsabilidades específicas y colabora con sistemas externos, siguiendo principios de **resiliencia, trazabilidad e independencia de dominio**.

- **Servicio de Transferencias**: implementa el patrón **Saga**, orquestando validación, reservas, ejecución en el Core Bancario, confirmación y notificación. Incluye un **Outbox Publisher** para garantizar consistencia y un **Compensation Handler** para revertir transacciones fallidas.

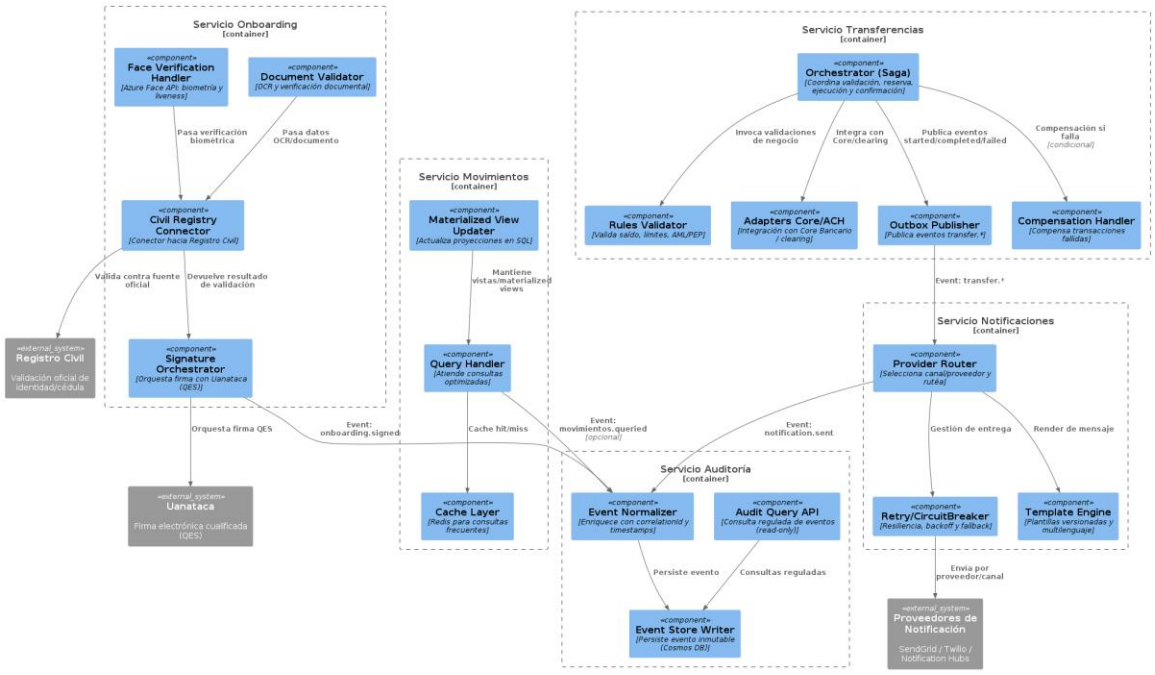
- **Servicio de Movimientos**: aplica **CQRS**, separando comandos y consultas. Un **Query Handler** atiende lecturas optimizadas, apoyado en un **Cache Layer** (Redis) y un **Materialized View Updater** que mantiene proyecciones actualizadas en SQL.

- **Servicio de Notificaciones**: implementa un **Provider Router** que selecciona el canal y proveedor más adecuado (SendGrid, Twilio, Notification Hubs), con soporte de **Template Engine** para personalización y **Retry/Circuit Breaker** para resiliencia.

- **Servicio de Onboarding**: orquesta el flujo de validación biométrica y documental. Incluye un **Face Verification Handler** (Azure Face API), un **Document Validator**

(OCR/Document Intelligence), un **Civil Registry Connector** (Registro Civil), y un **Signature Orchestrator** para la firma digital con Uanataca.

- **Servicio de Auditoría**: cuenta con un **Event Normalizer** (enriquecimiento de eventos con correlationId), un **Event Store Writer** (persistencia inmutable en Cosmos DB) y un **Audit Query API** para consultas reguladas.



4. Alta Disponibilidad, Monitoreo y Cumplimiento

Introducción:

La solidez de una arquitectura bancaria no se mide únicamente por sus capacidades funcionales, sino también por su capacidad de resistir fallos, garantizar continuidad operativa, ofrecer visibilidad en tiempo real y cumplir con los marcos regulatorios aplicables. En este capítulo se describen los mecanismos de alta disponibilidad (HA), los enfoques de monitoreo y observabilidad end-to-end, y las prácticas de cumplimiento normativo y seguridad que refuerzan las decisiones arquitectónicas planteadas en los capítulos anteriores. El objetivo es asegurar que la plataforma no solo entregue valor al cliente, sino que lo haga con confiabilidad, trazabilidad y conformidad legal en todo momento.

Alta Disponibilidad (HA) y Recuperación ante Desastres (DR):

- **Multi-región activo-activo en Azure**, con despliegue en al menos dos regiones geográficamente separadas, apoyado en **Availability Zones** como base de resiliencia (en línea con el diseño de **Infraestructura y Nube – 2.7**).
- **Geo-replicación en bases de datos críticas**:
    - Azure SQL (2.3 Persistencia) en modo **Business Critical** con failover groups.
    - Cosmos DB (2.3 Auditoría/Event Store) con **multi-region writes** para baja latencia y replicación automática.
- **Redis en clúster con réplicas** (2.3 Movimientos frecuentes), con failover automático.
- **Service Bus** con geo-disaster recovery namespace (2.6 Integración).
- **Backups automáticos y pruebas periódicas de restauración**, garantizando RPO ≤ 15 min y RTO ≤ 1 hora en servicios críticos como **Transferencias** (2.6) y **Onboarding** (2.4).
- App Service: **Auto-Heal** (reinicio basado en reglas) y **Health Checks**.
- APIM/WAF: **health probes** y **host name affinity** para detectar y aislar fallos de backend.

Monitoreo y Observabilidad:

- **Application Insights + Azure Monitor** integrados en todos los microservicios (2.7 Infraestructura).
- **Log Analytics** centralizado con dashboards de KPIs:
    - Latencia P95/P99 en APIs expuestas vía **APIM** (2.6 Capa de Integración).
    - Tasa de cache hit en **Redis** (2.3 Persistencia).
    - Consumo de RU/s en **Cosmos DB** (2.3 Auditoría).
    - Backlog en colas de **Service Bus/Event Grid** (2.6).
- **Grafana/Power BI** para vistas ejecutivas y de negocio.
- **Alertas proactivas** basadas en SLO definidos (ej. transferencias ≤ 2s intra-banco, como se definió en 2.6).
- **CorrelationId propagado** desde **API Gateway** hasta Auditoría (2.3 y Diagramas C4-C3), asegurando trazabilidad end-to-end.
- **Integración ITSM** para notificación automática de incidentes (Teams, Slack, PagerDuty).

Cumplimiento y Seguridad:

- **Normativas financieras**: cumplimiento con **PCI DSS**, **SOC2** y **LOPDP** (mencionadas en 2.2 Seguridad y 2.7 Infraestructura).
- **Zero-Trust** aplicado a todo el stack:
    - Autenticación y autorización centralizadas en el **IdP (Azure AD B2C/Keycloak, 2.2)**.
    - Acceso a secretos únicamente vía **Key Vault** (2.7).
    - Endpoints privados y segmentación de red definidos en **Infraestructura (2.7)**.
- **Auditoría inmutable**: eventos registrados en **Cosmos DB** con TTL, correlación y sellado de tiempo (2.3 Persistencia y 2.4 Onboarding).
- **MFA obligatorio** en accesos administrativos y operaciones sensibles (ej. transferencias, 2.2 Seguridad).

- **Defender for Cloud** y **Microsoft Purview** para fortalecer postura de seguridad y gobernanza de datos (2.7).
- **Validaciones periódicas de cumplimiento**: escaneos, hardening y IaC con Terraform/Ansible (2.7), alineados a auditorías regulatorias.

KPIs Operativos y SLO/SLA por Dominio:

Para garantizar observabilidad y cumplimiento de los acuerdos de servicio, se definen objetivos de nivel de servicio (SLOs) y acuerdos de nivel de servicio (SLAs) por dominio crítico:

Dominio / Servicio	Indicador Clave	SLO (objetivo interno)	SLA (externo al cliente)
Movimientos	Latencia de consulta (cache hit)	P95 < 300 ms	< 500 ms
	Latencia de consulta (cache miss)	P95 < 800 ms	< 1 s
Transferencias	Tiempo de confirmación intra-banco	P95 < 2 s	< 5 s
	Tasa de éxito de transacciones	≥ 99.95%	≥ 99.9%
Pagos	Procesamiento interbancario	Depende clearing externo	SLA compartido
Notificaciones	Tasa de entrega email/SMS/push	≥ 98%	≥ 97%
	Latencia de envío	P95 < 5 s SMS, < 10 s Email, < 3 s Push	99% dentro de SLO
Onboarding	Tiempo promedio de validación biométrica	< 5 s	< 8 s
	Porcentaje de validaciones exitosas	≥ 99%	≥ 98.5%
Auditoría	Persistencia de evento	< 200 ms	Garantía de inmutabilidad
Infraestructura	Disponibilidad global	≥ 99.95%	≥ 99.9%

Justificación:

Este capítulo se conecta directamente con las decisiones previas: la resiliencia en bases de datos y colas respalda la **separación OLTP/Auditoría (2.3)**, las alertas basadas en KPIs garantizan los tiempos definidos en la **Capa de Integración (2.6)**, y la trazabilidad soportada en **CorrelationId (2.3 y C4-C3)** asegura cumplimiento de auditoría. De esta forma, la arquitectura no solo es técnica y funcionalmente robusta, sino también **viable frente a reguladores y auditores**, garantizando continuidad, seguridad y confianza.

5. Conclusiones

La arquitectura propuesta consolida los principios de **seguridad, resiliencia y cumplimiento** requeridos por un sistema bancario moderno, alineando cada decisión tecnológica con los objetivos de negocio y regulatorios.

En particular, la solución:

- **Garantiza escalabilidad y resiliencia** gracias al diseño desacoplado en microservicios, soportado en patrones de CQRS, Saga y Event Sourcing.
- **Cumple con requerimientos normativos** al integrar auditoría inmutable, notificaciones obligatorias multicanal y medidas de seguridad alineadas con PCI DSS, SOC2 y la LOPDP.
- **Ofrece una experiencia moderna y consistente** al cliente mediante una SPA en React y una aplicación móvil multiplataforma en Flutter, soportadas por procesos de onboarding biométrico y firma digital.
- **Facilita la operación y el monitoreo proactivo** mediante integración nativa con los servicios de observabilidad y seguridad de Azure, asegurando trazabilidad end-to-end con correlationId y dashboards centralizados.

En conjunto, esta propuesta entrega a la entidad BP una **plataforma digital robusta, segura y preparada para el crecimiento**, capaz de soportar nuevas funcionalidades y de adaptarse a la evolución de los requerimientos regulatorios y de negocio.

6. Recomendaciones Finales y Próximos Pasos

La arquitectura planteada no solo cumple con los requisitos técnicos y regulatorios, sino que además deja sentadas las bases para la evolución futura del ecosistema digital. Para reforzar la estrategia, se sugieren los siguientes pasos:

1. **Piloto controlado:** implementar un MVP con alcance limitado (ej. consulta de movimientos y onboarding biométrico) para validar experiencia de usuario, latencia y resiliencia.
2. **Pruebas de cumplimiento y seguridad:** ejecutar auditorías tempranas de PCI DSS y LOPDP sobre el MVP, incorporando auto-healing y hardening desde IaC.
3. **Optimización de costos:** establecer monitoreo de cost-to-serve por dominio y ajustar escalado automático/reservas en función de la demanda real.
4. **Gobernanza y cultura DevSecOps:** consolidar pipelines CI/CD con validaciones de seguridad, pruebas automáticas y métricas de calidad como parte del ciclo de vida.
5. **Evolución funcional:** extender progresivamente a pagos interbancarios, catálogos dinámicos y analítica avanzada sobre datos de auditoría para prevención de fraude.