Arquitectura Soluciones en Azure - BP

0. Resumen Ejecutivo

BP Online Banking permite a los clientes **consultar movimientos** y **realizar pagos y transferencias** (propias e interbancarias) con **notificaciones obligatorias**.
La solución es **cloud-native en Azure**, desacoplada por **microservicios** en **Azure Container Apps (ACA)**, con **Azure API Management (APIM)** como puerta de enlace y **Azure Front Door + WAF** en el borde.
Las **lecturas rápidas** se sirven desde **Cosmos DB** (patrón **CQRS**), mientras que la capa **transaccional y de auditoría** reside en **Azure SQL** con **Ledger**.
La autenticación se implementa con **OAuth2/OIDC (Authorization Code + PKCE)** sobre **Microsoft Entra ID / B2C**, incorporando **onboarding biométrico** (liveness).
La **mensajería** con **Azure Service Bus** desacopla procesos críticos, alimenta **notificaciones** (Microsoft Graph + Azure Communication Services/Twilio) y **auditoría**.
Se prioriza **seguridad** (WAF, políticas APIM, Private Endpoints, Key Vault), **alta disponibilidad y DR multirregión**, **observabilidad** (Application Insights, Azure Monitor) y **control de costos** (serverless, autoscale, budgets), cumpliendo normativa de **protección de datos** y estándares de **seguridad financiera**.

1. Requerimientos del ejercicio (Solución propuesta)

- **Movimientos y transferencias**: Microservicios **Movements Query** (read model en Cosmos) y **Transfers** (transaccional en SQL + orquestación tipo *Saga*).
- **Datos cliente desde 2 fuentes**: **Customer Basic Data** compone **Core** y **Detalle**; **cache-aside** con **Redis** para perfiles y beneficiarios frecuentes.
- **Notificaciones** (≥2 canales): **Notifications** integra **Microsoft Graph** (correo) y **ACS/Twilio** (SMS/Push).
- **2 front-ends**: SPA web (**Angular**) y móvil (**Flutter** o **.NET MAUI**).
  Justificación: ambos comparten base de UI y acceden a capacidades nativas; Flutter destaca por rendimiento/consistencia visual; MAUI integra muy bien con .NET si el equipo es .NET-first.
- **Autenticación OAuth2/OIDC**: **Authorization Code + PKCE** para SPA/móvil; **Client Credentials** entre servicios. **MFA** y **Conditional Access**.
- **Onboarding biométrico**: verificación facial con *liveness* (p.ej., **FacePhi**), evidencias en **Blob** y alta automática en **Entra ID B2C**.
- **Auditoría**: **SQL Database (Ledger)** + eventos *append-only* y exportación a **Log Analytics/Sentinel**.
- **Capa de integración**: **APIM** (validación JWT, cuotas, transformaciones, *subscription keys*, mTLS opcional) detrás de **Front Door + WAF**.
- **NFRs**: HA activo-activo, DR, seguridad, monitoreo y *auto-healing*.

2. C1 - Diagrama de Contexto

El C1 muestra a los usuarios Web (SPA) y móvil, el BP Online Banking y los sistemas externos: Core bancario, sistema de detalles, red interbancaria, proveedores de notificación y FacePhi. Las flechas explican qué información fluye: consultas/transferencias desde el front al backend, y llamadas del backend a sistemas externos. El IdP (Entra ID/B2C) autentica; el backend valida el JWT en cada solicitud.

Actores

- Usuario web (SPA)
- Usuario móvil (App)
- Soporte / BP Ops

Sistemas externos

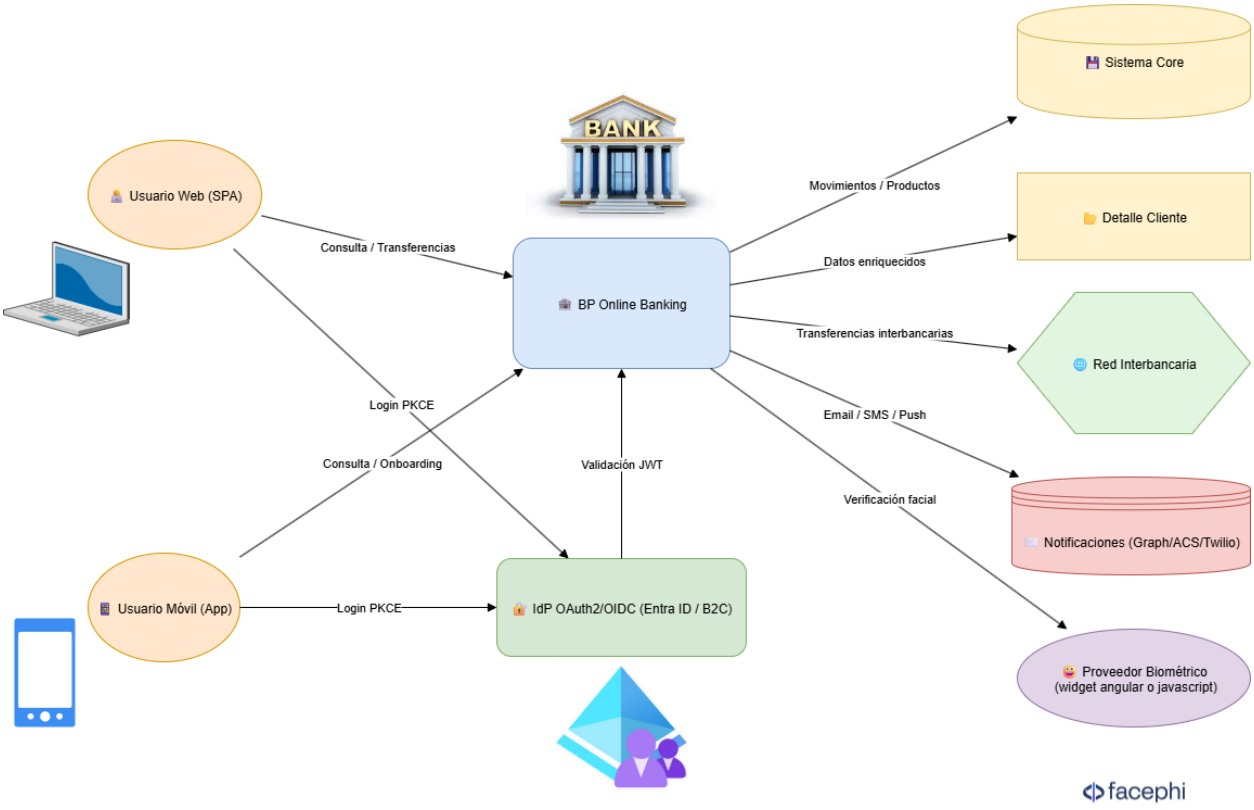
- Core Bancario (productos, saldos, movimientos)
- Sistema de Detalle de Cliente (información enriquecida)
- Red Interbancaria / Switch (transferencias ISO 8583/REST)
- Proveedores de notificaciones (Microsoft Graph, Azure Communication Services, ACS Twilio/SendGrid)
- Proveedor biométrico (Azure AI Vision, Onfido, Jumio, Facephi widget (angular o javascript))
- Identity Provider OAuth2/OIDC (Microsoft Entra ID / Entra ID B2C)

Sistema BP Online Banking

- SPA (Angular)
- Mobile App (Flutter)
- API Gateway (Azure API Management – APIM)
- Microservicios en .NET: Customer Basic Data, Movements Query, Transfers, Notifications, Onboarding Orchestrator, Audit & Compliance, BFF Web/Mobile (opcional) y persistencias (Cosmos, SQL, Blob).
- Bases de datos: Cosmos DB, Azure SQL, Blob Storage
- Mensajería: Azure Service Bus

Flujo

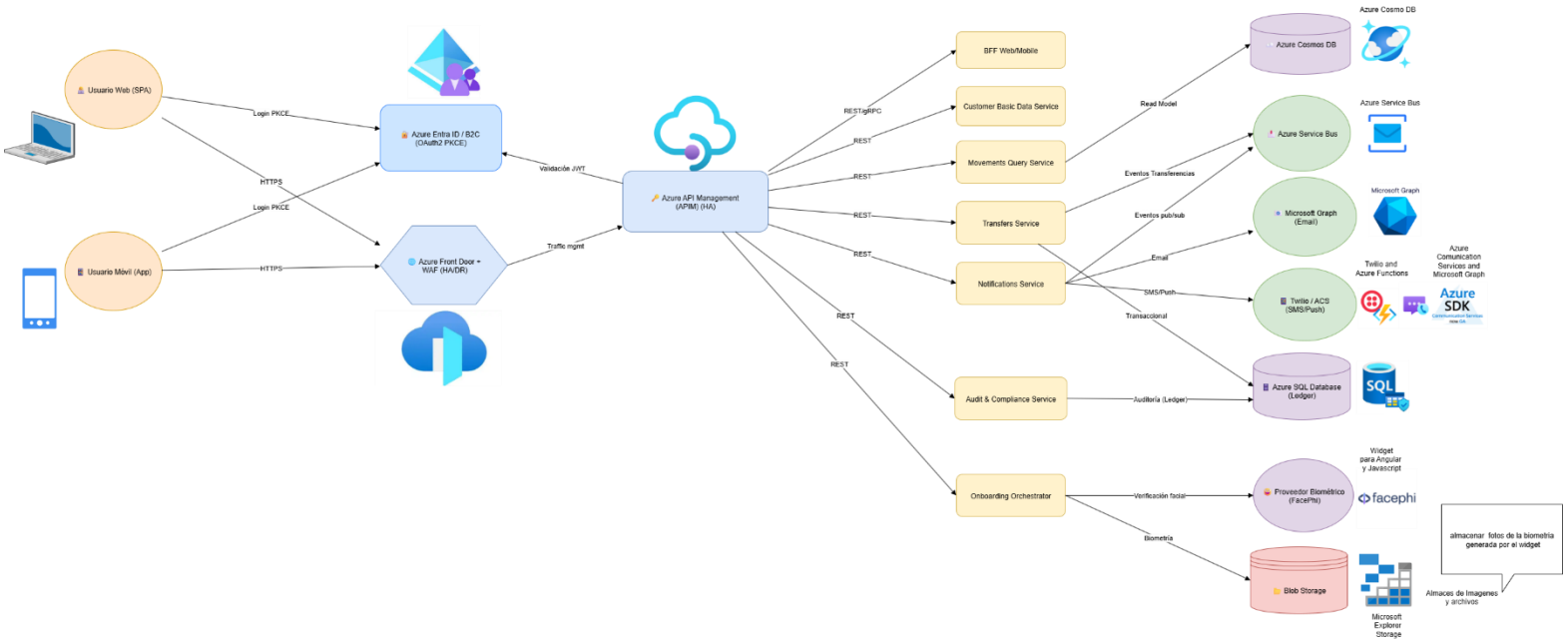
los front-ends se autentican en **B2C**, llaman a **APIM** y este enruta a los servicios; el backend valida **JWT** y se integra con Core, Detalle, Switch, Notificaciones y Biométrico.



3. C2 - Diagrama de Contenedores

El C2 introduce los contenedores: BFF Web/Mobile y microservicios (Customer Data, Movements, Transfers, Notifications, Audit, Onboarding). El tráfico entra por Azure Front Door + WAF, pasa por API Management y llega a los servicios. Persistencias: Cosmos DB (lecturas), Azure SQL (Ledger) y Blob (biometría). Mensajería con Azure Service Bus. Notificación con Graph y ACS/Twilio.

- **Edge**: **Azure Front Door + WAF** (protección OWASP + routing global).
- **API Gateway**: **APIM** (políticas, *subscription keys*, productos, analytics, mTLS).
- **Compute**: **Azure Container Apps** (BFF Web/Mobile y microservicios: Customer, Movements, Transfers, Notifications, Onboarding, Audit).
- **Datos**: **Azure SQL (Ledger)**, **Cosmos DB (Core API)**, **Blob Storage** (evidencias), **Redis** (cache opcional).
- **Mensajería**: **Service Bus** (Topics/Queues).
- **Seguridad y Observabilidad**: **Key Vault, Private Endpoints, Defender for Cloud, Application Insights, Azure Monitor/Sentinel**.



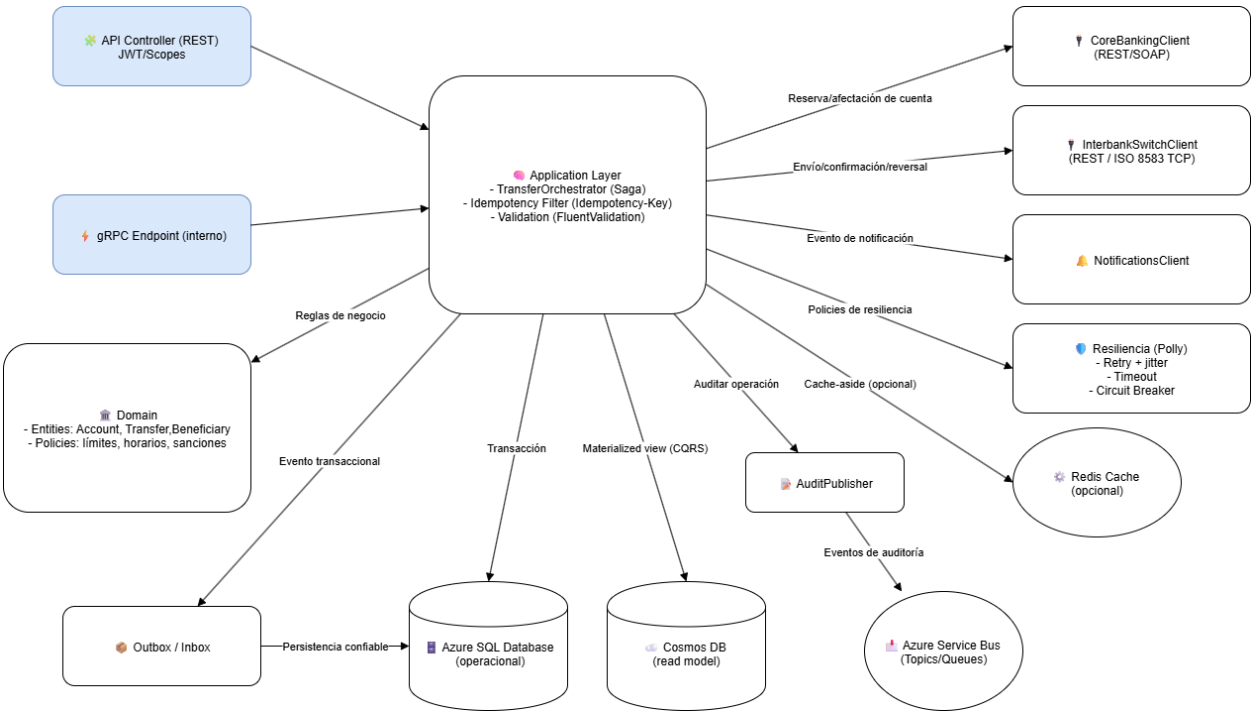
3.1. ¿Dónde se ejecutan los servicios? (Compute)

Componente	**Servicio Azure elegido**	**Por qué (≥2 razones)**	**Alternativas evaluadas**
BFF y microservicios	Azure Container Apps (ACA)	Serverless sin administrar clúster; **autoscale** por HTTP/colas; **blue/green** por *revisions*; **Dapr** y **Managed Identity**.	App Service (simple) / **AKS** (máximo control).
Jobs/Outbox/Projector	**ACA Jobs** o **Azure Functions**	Escala a cero; pago por uso; *bindings* nativos a Service Bus; aislar idempotencia.	WebJobs / contenedor interno.
API Gateway	Azure API Management (Std/Premium)	Políticas (**JWT/claims**, **rate-limit/quota/burst**, **transformaciones**), **productos** y **subscription keys**, **analytics**, **mTLS**. **Ejemplo**: 100 req/min por suscripción y 20 req/s por IP; rechazar sin Ocp-Apim-Subscription-Key; validar aud/iss.	Kong/NGINX. *(DDoS L3/4: Azure DDoS + WAF)*

Borde/Global	Azure Front Door + WAF	Anycast global y protección OWASP; routing sencillo.	Traffic Manager + App Gateway.
Identidad	Entra ID / B2C	OIDC/OAuth2 con PKCE; MFA/CA; políticas personalizadas.	Keycloak / Auth0.
Datos operacionales	Azure SQL (+ Ledger)	ACID y T-SQL; evidencia inmutable para auditoría legal.	SQL MI; PostgreSQL.
Read model	Cosmos DB (Core API)	Baja latencia y RU/s elásticas; TTL y particiones.	ElasticSearch / OpenSearch.
Mensajería	Service Bus (Topics/Queues)	Orden relativo y DLQ; sesiones; resiliencia.	Event Grid; Kafka.
Cache	Azure Cache for Redis	Cache-aside; TTLs; locks para idempotencia.	Sin cache (mayor latencia).
Evidencias	Azure Blob Storage (Hot) C	Costo/GB bajo; versionado y retención legal.	Archivo/relacional (no recomendado).

4. C3 - Diagrama de Componentes (Transfers) y patrones

Capa de Aplicación: *Transfer Orchestrator* (patrón **Saga**), **Idempotency-Key** y validaciones.
Dominio: Entidades Transfer, Account, Beneficiary y **políticas** (límites, horarios, sanciones). Emite **Domain Events**.
Infraestructura: Persistencia en **SQL** (transaccional), **proyecciones** a **Cosmos** (CQRS), **Outbox** confiable hacia **Service Bus** y **AuditPublisher** a **SQL Ledger**. **Redis** acelera lecturas muy frecuentes.
Integraciones: CoreBankingClient (REST/SOAP), InterbankSwitchClient (REST/ISO-8583 TCP), NotificationsClient (Graph/ACS/Twilio).
Resiliencia: **Polly** (retry+jitter, timeout, circuit-breaker), DLQ en Service Bus.



5. Front-ends, autenticación y onboarding

Front-ends

SPA **Angular**; App móvil **Flutter** (alternativa: **.NET MAUI** si el equipo es .NET-first).

Se recomienda Flutter y .NET MAUI como frameworks multiplataforma.

Autenticación

- **Authorization Code + PKCE** (SPA/móvil); **Client Credentials** entre servicios.
- **MFA** y **Conditional Access** (riesgo, IP, dispositivo). MFA/CA endurece el acceso.

Onboarding con biometría

1. Captura rostro + liveness
2. Validación con proveedor biométrico
3. Alta en Entra ID B2C
4. Registro en SQL Ledger y Blob

FacePhi (SDK/widget), liveness, evidencias en Blob con retención;

Ingreso posterior con usuario+clave y biometría local (Face/Touch ID).

Consulta de movimientos

BFF → Movements Query → Cosmos DB (read model)

Transferencias

SPA/App → APIM → Transfers Service → Core → Switch → Notificaciones

Notificaciones

- Email: Graph
- SMS: ACS/Twilio
- Push: Notification Hubs

Auditoría

Eventos en Service Bus → SQL Ledger + Event Hub/Sentinel

6. CI/CD, secretos y configuración

Pipeline (Azure DevOps o GitHub Actions)

1. *Build* → tests → **SAST** → imagen Docker → **Azure Container Registry (ACR)**.
2. **Scan** de imagen (Defender for Cloud).
3. **Infra as Code** (Bicep/Terraform): ACA, APIM, Front Door, DBs, Bus, **Key Vault**, **App Configuration**, **Private Endpoints**.
4. *Deploy* **blue/green** con *revisions* de ACA y aprobaciones (*gates*).
   Secretos en **Key Vault** con **Managed Identity**; **Key Vault references** en los servicios.
   Variables no secretas en **Azure App Configuration** (feature flags).
   Ambientes: dev, test (QA), preprod, prod; **resource groups por dominio** y **tags** (coste/propietario/criticidad. RG por dominio y tags para coste/propietario/criticidad.

7. Patrones y Justificaciones

- ***Microservicios + APIM****: escalado independiente y gobernanza centralizada (políticas/analytics)*.
- ***CQRS + Read Models****: menor latencia en consultas; separar lectura/escritura para aislar picos*.
- ***Service Bus****: desacoplamiento,* retries*,* ***DLQ****, orden relativo por sesión*.
- ***Outbox/Inbox + Idempotency****: consistencia y reentrega segura ante fallos*.
- ***OAuth2/OIDC + PKCE****: clientes públicos seguros (evita* implicit flow *y* code interception*)*.
- ***Container Apps****: serverless, autoscale "por demanda",* ***revisions*** *para* blue/green *con mínimo* ops.

7. Normativa y Seguridad

- ***Datos personales****:* ***GDPR/LOPDP****; catalogación PII;* ***mascaramiento*** *y* ***retención****;* ***Right to Access/Erasure***.

- ***Cifrado**: TLS 1.2+; en reposo AES-256; claves y certificados en **Key Vault**.*
- ***Perímetro**: **Front Door + WAF** (OWASP), cabeceras seguras (HSTS, CSP).*
- ***Red privada**: **Private Endpoints** a SQL/Cosmos/Blob/Bus; APIM Premium con VNet si es necesario.*
- ***Identidad**: **MFA/CA**,* least privilege*, **Managed Identity**.*
- ***Estándares**: **ISO 27001**, **NIST**, **OWASP ASVS**, **PCI DSS** (si hay tarjetas).*
- ***Monitoreo/Auditoría**: **Application Insights + Log Analytics**, trazas **OpenTelemetry**, **Sentinel** para correlación.*

8. Alta Disponibilidad, DR y Monitoreo

- ***HA:** despliegue en **2 regiones** emparejadas**; Front Door** activo-activo**; APIM/ACA/DBs** con zonas.*
- ***DR:** geo-replication **en SQL/Cosmos; backups automáticos;** RPO ≤ 5 min**,** RTO ≤ 30 min**.***
- ***Auto-healing:** **health probes**, autoscale**,** retry con jitter**,** circuit-breaker**, DLQ.***
- ***Monitoreo:** **dashboards** por servicio**,** SLOs **con alertas, cuadernos** Kusto **y** **workbooks.***

9. Costos: cómo estimarlos en la Azure Pricing Calculator

1. **Front Door + WAF**: tráfico de salida (GB/mes) y nº de reglas WAF.
2. **APIM**: **Standard** (sin VNet) o **Premium** (VNet + multi-región).
3. **Container Apps**: vCPU/memoria por app, **réplicas mín/máx** y horas activas.
    i. **Tip**: deja **réplica mínima = 0** donde sea viable para evitar costo "siempre encendido".
4. **ACR**: Basic/Standard según nº de imágenes.
5. **Azure SQL (General Purpose) + Ledger**: vCores y GB.
6. **Cosmos DB (Core API)**: **RU/s autoscale** (pico y base) y GB.
7. **Service Bus**: Standard vs **Premium** (aislamiento y latencia fija).
8. **Blob**: GB y transacciones (evidencias biométricas).
9. **Redis**: memoria y tier (C1–C3) si se usa.
10. **Application Insights + Log Analytics**: GB/mes y retención (30–90 días).
11. **Entra ID B2C**: **MAU** y uso de **MFA**.
12. **ACS/Twilio + Graph**: nº de SMS/Push/Email por mes (costos por país).
    Optimización: budgets y alertas; RU/s máximas en Cosmos; TTL en lecturas; **muestreo** en telemetría; compresión/caché en Front Door; **feature flags** para apagar funcionalidades costosas.

Calcular precios en azure de todos los componentes usados:

https://azure.microsoft.com/es-es/pricing/calculator/

Microsoft Azure Estimate			
Su presupuesto			
Service category	Service type	Custom name	Region
Redes	Azure Front Door		
Web	API Management		East US
Bases de datos	Azure Cosmos DB		East US
Identidad	Microsoft Entra External ID		East US
Contenedores	Azure Container Apps		East US 2
Bases de datos	Azure SQL Database		East US
Integración	Service Bus		East US

Almacenamiento	Storage Accounts		East US
Seguridad	Key Vault		East US
Web	Azure Communication Services		East US

DevOps	Azure Monitor		East US
Seguridad	Microsoft Sentinel		East US
Bases de datos	Azure Cache for Redis		East US
Redes	Azure DNS		
Redes	Azure DDoS Protection		East US

Seguridad	Microsoft Defender for Cloud		East US
Seguridad	Defender External Attack Surface Management		East US
Support			Support
			Licensing Program
			Billing Account
			Billing Profile
			Total
Disclaimer			
All prices shown are in United States – Dollar ($) USD. This is a summary estimate, not a quote. For up to date pricing inform			
This estimate was created at 9/5/2025 6:59:46 PM UTC.			

10. Conclusión y próximos pasos

La arquitectura es **coherente, segura y escalable**; satisface los requerimientos y facilita el cumplimiento normativo.
Siguientes pasos:

1. PoC con tráfico real; 2) elegir *tier* final de **APIM** (Std vs Premium);
2. ajustar **RU/s** y **vCores** con métricas; 4) publicar el PDF en el repositorio.

Repositorio GitHub: https://github.com/stdpacheco/arquitectura-soluciones-en-Nube-Azure-BP

Anexo A — Políticas APIM (ejemplo)

```
<policies>
  <inbound>
    <base/>
    <!-- Límite por IP: 20 req/s -->
    <rate-limit-by-key calls="20" renewal-period="1"
      counter-key="@(context.Request.IpAddress)" />
    <!-- Cuota por suscripción: 6000 req/h -->
    <quota-by-key calls="6000" renewal-period="3600"
      counter-key="@(context.Subscription?.Key)" />
    <!-- Requerir subscription key -->
```

```
    <check-header name="Ocp-Apim-Subscription-Key"
      failed-check-httpcode="401"
      failed-check-error-message="Subscription key required." />
    <!-- Validar JWT -->
    <validate-jwt header-name="Authorization" require-scheme="Bearer"
      failed-validation-httpcode="401">
      <openid-config url="https://login.microsoftonline.com/<TENANT>/v2.0/.well-known/openid-configuration" />
      <audiences><audience>api://online-banking</audience></audiences>
    </validate-jwt>
    <set-header name="X-Correlation-Id" exists-action="override">
      <value>@(context.RequestId)</value>
    </set-header>
  </inbound>
</policies>
```