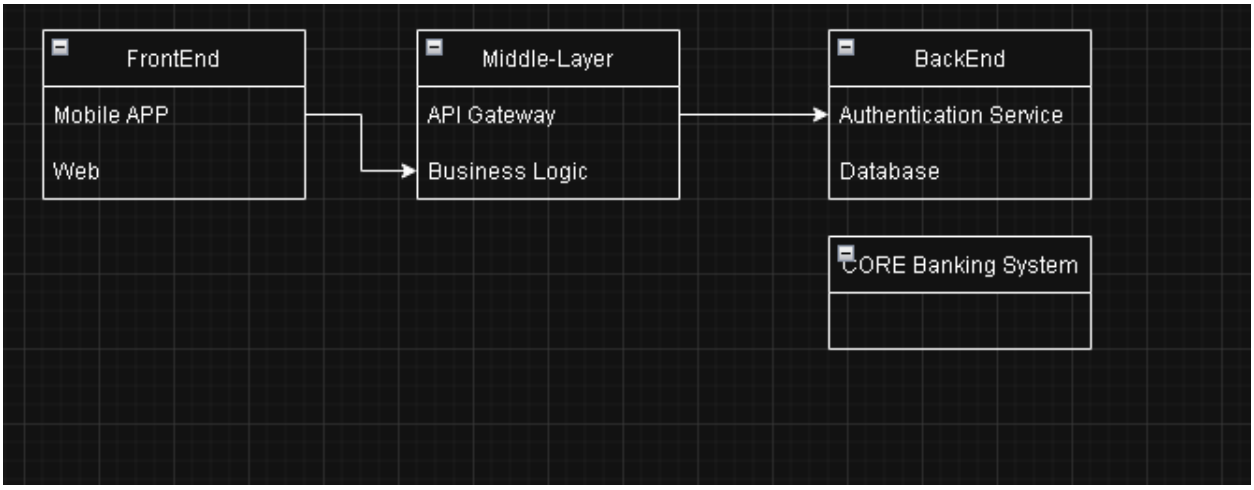
Luis Alberto Garcia Diaz

Eval. Tec. Jefe de Desarrollo/Líder técnico

1. 

    a. Autenticación y Autorización
        i. OAuth 2.0 + JWT para delegar acceso entre front-end y back-end de forma tokenizada
        ii. Tokens firmados con claves asimétricas (RSA/ECDSA) y con expiración corta
        iii. Validación de scopes y roles en cada capa intermedia
    b. Encriptación de Datos en Tránsito
        i. TLS 1.3 obligatorio en todos los canales (app ↔ API ↔ core bancario)
    c. Intercambio Seguro entre Microservicios
        i. Comunicación basada en gRPC con TLS o APIs internas con MTLS (mutua autenticación)
        ii. Uso de certificados rotativos.
    d. Firewalls Lógicos por Capa
        i. Cada capa ubicada en una zona segura aislada
        ii. Reglas de comunicación estrictas: solo los puertos y protocolos necesarios
    e. Control de Acceso Interno Granular
        i. Políticas RBAC o ABAC en servicios backend y bases de datos
        ii. Validación cruzada entre el token del usuario y el sistema intermedio antes de ejecutar operaciones
    f. Auditoría y Trazabilidad
        i. Todos los llamados entre capas deben ser auditables, con requestId y userId propagados
        ii. Logs firmados y protegidos contra alteraciones
    g. Protección contra Ataques Comunes
        i. Validación estricta en el front-end y en el middleware (anti-inyección, anti-XSS)
        ii. Rate limiting y detección de anomalías para prevenir ataques de fuerza bruta
    h. Capa de Observabilidad con Seguridad
        i. Dashboards que visualicen únicamente metadatos necesarios
        ii. Acceso a monitoreo restringido por perfil técnico y separado del entorno productivo

2. Tecnologias:
    a. Front:
        i. Flutter o MAUI, Desarrollo unificado para ambas plataformas
        ii. React: madurez y alto rendimiento
        iii. TypeScript: Tipado estricto
        iv. Redux: Escalabilidad en los datos
    b. Back:
        i. .net 8: Seguridad y escalibidad
        ii. gRPC:Intercomunicacion
        iii. IdentityServer: Oauth
        iv. SQL Server: base de datos
        v. Redis: Cache y sesiones
        vi. Vault: Gestion de credenciales
        vii. Docker+ kubernets: Orquestacion.
    c. Core Bancario:
        i. Kafka para desacoplar flujos
        ii. Conexión vía VPN segura o circuitos dedicados
    d. DevOps
        i. GitHub Actions
        ii. Terraform
        iii. GitHub: Code Repository.
        iv. Elastic: logs
3. Seguridad:
    a. Autenticación y Autorización
        i. Implementación de OAuth 2.0 con JWT, validando scopes, roles, y origen de la solicitud.
        ii. Tokens firmados con RSA y rotación periódica de claves.
        iii. Login adaptativo (según dispositivo, ubicación o comportamiento).
        iv. Integración con MFA y biometría en la app móvil.
    b. Protección de Datos en Tránsito y en Reposo
        i. TLS 1.3 obligatorio en todas las capas: comunicación interna, móvil-backend, y con el Core bancario.
        ii. mTLS para microservicios sensibles que requieren autenticación mutua.
        iii. Cifrado de datos en reposo usando AES-256, con claves gestionadas por Vault o KMS empresarial.
        iv. Separación lógica de datos sensibles (credenciales, transacciones) y datos operativos.
    c. Gestión de Sesiones y Tokenización
        i. Tokens stateless con expiración corta y refresco seguro.
        ii. Detección y revocación de sesiones comprometidas.
        iii. Tokenización de identificadores sensibles (como cuenta CLABE) antes de almacenarlos o transmitirlos.
        iv. Persistencia segura en Redis con TTL dinámico según tipo de sesión.
    d. Buenas Prácticas OWASP contra Ataques Comunes

        i. Validación de entradas desde front-end hasta middleware: anti-inyección, anti-XSS, anti-CSRF.
        ii. Rate limiting, CAPTCHA y detección de anomalías en tiempo real.
        iii. Uso de Content Security Policy (CSP) para navegación web.
        iv. Escaneo automático de dependencias y paquetes para vulnerabilidades conocidas.
        v. Sandbox de componentes expuestos o con ejecución remota.
4. CD/CI/Logs
    a. Integración Continua (CI)
        i. Repositorio centralizado (GitHub) con ramas protegidas (main, release, hotfix)
        ii. Validación automática por PR (Pull Request):
        iii. Linter y pruebas de sintaxis (ESLint)
        iv. Unit tests con JUnit
        v. Análisis de vulnerabilidades (GitHub)
        vi. Versionado semántico automático (major. minor. patch) con etiquetas y changelogs
        vii. Construcción de artefactos: imágenes Docker versionadas, empaquetado de apps móviles (.apk / .ipa)
    b. Despliegue Continuo (CD)
        i. Entornos separados: Dev, QA, Staging, Producción (con pipelines independientes y controles por ambiente)
        ii. Despliegue progresivo:
            1. Blue/Green deployments o Canary releases para minimizar riesgos
        iii. Rollback automático si hay fallos en health checks
        iv. Orquestación con Kubernetes y Helm Charts para mantener consistencia entre entornos
        v. Cifrado y gestión de secretos:
        vi. Azure Key Vault para inyectar credenciales vía sidecar o config maps
    c. Auditoría y visibilidad
        i. Logs centralizados con Elastic
        ii. Trazabilidad con OpenTelemetry, propagando requestId y userId entre servicios
    d. Automatización y Gobernanza
        i. GitOps: infraestructura como código con Terraform, validada y auditada por commit
        ii. Pipelines como código (azure-pipelines.yml) con historial completo
        iii. Validación de firmas de commit y autorización por roles para el despliegue a producción
        iv. Monitoreo de eficiencia del pipeline: tiempo promedio por paso, frecuencia de errores, tasa de rollback
5. RoadMap
    a. Fases:
        i. Fundamentos:(Mes 1- mes 2)
        ii. Servicios Claves (mes3-mes 4
        iii. Optimización (mes 5, mes 6)

b. Roles: (nombre|Rol|Complemento)
    i. Front-End | Flutter, React | UX, pruebas adaptativas |
    ii. Back-End | .NET , gRPC | Seguridad, integración core |
    iii. DevOps | CI/CD, Helm, Vault, Kubernetes | SRE, alertamiento |
    iv. Seguridad | Criptografía, protección OWASP | Sesiones, MFA, monitoreo |
    v. QA | Automatización, stress testing | Validación continua |
    vi. Product | Priorización, roadmap | Comunicación con stakeholders |

Fase	Sprint	Objetivo Clave	Roles
1	1	Setup de CI/CD, gestión de secretos, diseño de arquitectura	DevOps
	2	Módulo de autenticación, mTLS entre servicios, base de datos	Back-end, Seguridad, QA
	3	Primer prototipo UI/UX móvil, login biométrico	Front-end,QA
2	4	Módulos: consulta de saldo, movimientos, notificaciones	Back-end, Front-end, QA
	5	Integración controlada con Core bancario (Kafka o adaptadores)	DevOps, Front-end, Back-End
	6	Implementación de alertas, logs distribuidos, monitoreo	Back-end, DevOps, product
3	7	Pruebas de estrés, hardening, ajustes de rendimiento	QA, Seguridad, DevOps
	8	Orquestación completa en Kubernetes, habilitación multicanal	DevOps, Front-end
	9	Publicación controlada, feedback de usuarios, governance	Product Owner, QA

Extras:

- Normativas Aplicables:
    - La arquitectura propuesta se fundamenta en estándares internacionales como ISO 27001, PCI DSS y prácticas OWASP, asegurando que cada capa del sistema cumpla con requisitos de confidencialidad, integridad y disponibilidad, de acuerdo con la normatividad mexicana y global.

Norma	Aplicación	Observaciones
ISO/IEC 27001	Gestión de la seguridad de la información	Requiere controles organizativos, físicos y técnicos, ideal para justificar trazabilidad, controles de acceso y logs
ISO/IEC 27002	Prácticas de seguridad complementarias	Aplica directamente en componentes como hardening de servidores, políticas de contraseña y segregación de redes
PCI DSS	Protección de datos financieros	Especialmente relevante si el sistema procesa transacciones con tarjetas (no se menciona en el contexto, pero se mencionó en la reunión

		previa), exige cifrado, tokenización y auditoría
OWASP ASVS / Top 10	Desarrollo seguro de aplicaciones	Alinear validaciones anti-XSS, CSRF, inyección y otros controles a esta guía. Se acepta como estándar global en desarrollo seguro
Ley de Protección de Datos Personales (LGPDPPSO)	México	Para reforzar el componente legal de privacidad, pero vale destacar su implicación en trazabilidad y derechos ARCO
Normatividad CNBV (México)	Requisitos de seguridad bancaria	Ideal para justificar separación de ambientes, monitoreo continuo, autenticación fuerte y resiliencia ante fallos

- 🔧 **Marcos DevOps y CI/CD**

Norma/Guía	Aplicación	Comentario
NIST SP 800-218 (SSDF)	Desarrollo seguro integrado	Propone prácticas para CI/CD y gestión de vulnerabilidades. Puedes justificar escaneo de código y validación por commit
CNCF Best Practices	Entornos Kubernetes y contenedores	Puedes alinear tu orquestación con Helm, gestión de secretos con Vault
GitOps Principles (por Weaveworks)	Gobernanza y trazabilidad	Refuerza que toda infraestructura se gestiona como código, auditada y versionada