High-Level Design Document

High-Level Design Document

> **Project:** Drop **Version:** 1.0 **Date:** 2026-02-23 **Author:** Petter Graff, Senior Enterprise Architect **Status:** Approved **Reviewers:** Alem Bašić (CEO), John (AI Director)

Document History

Version	Date	Author	Changes
0.1	2026-02-21	Standards Architect	Initial draft from source code analysis
1.0	2026-02-23	Petter Graff	Filled from real architecture docs

---

1. Executive Summary

Purpose: Drop is a PSD2 pass-through payment application that enables Norwegian residents (18+) to send money internationally (remittance) and pay merchants via QR code — without Drop ever holding customer funds.

Business Context: Sending money from Norway is expensive and complex. Diaspora communities and internationally connected residents pay high fees through traditional remittance services. Drop removes that friction by operating as a licensed AISP (Account Information Service Provider) and PISP (Payment Initiation Service Provider) under PSD2 / Betalingstjenesteloven — reading balances directly from users' banks and initiating payments on their behalf. ALAI Holding AS builds Drop as a product for the Norwegian market, targeting all residents of Norway and Scandinavia, not just diaspora communities.

Key Outcomes:

- Users send money to 30+ countries at lower fees (0.5% vs. industry 2-5%)

- Merchants accept QR payments without POS hardware — mobile-first
- Drop avoids EMI licensing complexity (350K EUR capital requirement) by adopting the PISP/AISP pass-through model (20-50K EUR capital requirement)
- Strong regulatory compliance: BankID SCA, Sumsub KYC/AML, GDPR, AML (hvitvaskingsloven)

Scope: This document covers the Drop platform — web app (`drop-web`), API server (`drop-api`), and mobile app (`drop-mobile`) — and their integrations with BankID, Open Banking (AISP/PISP), Sumsub KYC, and payment rails. It excludes the Drop landing/marketing site, the Cards feature (feature-flagged, future), and future Vipps Login integration.

---

2. System Context (C4 Level 1)

```
graph TB
    subgraph actors["External Actors"]
        sender["Sender<br/>(Norwegian Resident, 18+)<br/>Sends money abroad via PISP"]
        receiver["Receiver<br/>(30+ countries)<br/>Receives remittance"]
        merchant["Merchant<br/>(Norwegian Business)<br/>Accepts QR payments"]
    end

    subgraph drop_system["Drop Payment System (ALAI Holding AS)"]
        drop["Drop<br/>Next.js 15 + Hono v4 + Expo SDK 54<br/>PSD2 Pass-through App<br/>(AISP + PISP)"]
    end

    subgraph banking["Banking & Open Banking"]
        bankid["BankID Norway<br/>OIDC Identity Provider<br/>Strong Customer Authentication"]
        nordic_banks["Nordic Banks<br/>(DNB, SpareBank1, Nordea)<br/>Berlin Group NextGenPSD2 APIs<br/>AISP: Read balance<br/>PISP: Initiate payment"]
        payment_rails["Payment Rails<br/>SEPA (EEA) / SWIFT (non-EEA)<br/>30+ remittance corridors"]
    end

    subgraph compliance["Compliance & KYC"]
        sumsub["Sumsub<br/>KYC/AML Provider<br/>Document verification<br/>PEP/sanctions screening"]
        finanstilsynet["Finanstilsynet<br/>Norwegian FSA<br/>PISP/AISP registration<br/>Regulatory oversight"]
        okokrim["Okokrim / EFE<br/>Financial Intelligence Unit<br/>STR/SAR filing"]
```

```
    end

    subgraph infrastructure["Infrastructure"]
        aws["AWS App Runner<br/>eu-north-1 (Stockholm)<br/>Container hosting + auto-scaling"]
        cloudflare["Cloudflare<br/>CDN, WAF, DDoS protection<br/>DNS, TLS termination<br/>getdrop.no"]
        sentry["Sentry<br/>Error tracking<br/>Performance monitoring"]
    end

    sender -->|"BankID login, view balance (AISP), send money (PISP), QR payments"| drop
    receiver -.->|"Receives funds via bank transfer"| payment_rails
    merchant -->|"Register business, view dashboard, generate QR code"| drop

    drop -->|"OIDC authorize, ID token verification, age/identity check"| bankid
    drop -->|"AISP: GET /accounts /balances; PISP: POST /payments"| nordic_banks
    drop -->|"PISP payment routing — SEPA for EEA, SWIFT for non-EEA"| payment_rails

    drop -->|"Applicant creation, document upload, webhook results"| sumsub
    drop -.->|"License registration, regulatory reporting"| finanstilsynet
    drop -.->|"STR filing (hvitvaskingsloven)"| okokrim

    drop -->|"Deploy containers, auto-scale"| aws
    drop -->|"DNS routing, TLS, WAF, DDoS protection"| cloudflare
    drop -->|"Error events, performance traces"| sentry

    nordic_banks -->|"Execute transfers"| payment_rails
```

---

3. Container Diagram (C4 Level 2)

```
C4Container
  title Drop — Container Diagram (C4 Level 2)

  Person(user, "End User", "Norwegian resident 18+, authenticated via BankID")
  Person(merchant, "Merchant", "Business owner receiving QR payments")

  System_Boundary(drop, "Drop Platform") {
    Container(web, "drop-web", "Next.js 15, React 19, Tailwind v4", "SSR web app. 10 screens: Login, Onboarding, Dashboard, SendMoney, BankAccounts, TransactionHistory, ScanQR, Profile,
```

```
Notifications, MerchantDashboard. BankID auth via httpOnly cookie.")
    Container(api, "drop-api", "Hono v4, Node.js 22 Alpine", "REST API server. 26+ endpoints under /v1/. BankID OIDC callback, transaction processing, recipient management, merchant registration, GDPR compliance, admin operations.")
    Container(mobile, "drop-mobile", "Expo SDK 54, React Native", "Native iOS/Android app. BankID auth via expo-web-browser deep linking (drop://auth/callback). AsyncStorage for token. 4 tabs: Hjem, Send, QR, Profil.")
    ContainerDb(db, "Database", "SQLite (dev) / PostgreSQL 16 (prod)", "19 tables: users, sessions, transactions, bank_accounts, recipients, merchants, notifications, settings, cards, spending_limits, exchange_rates, audit_log, aml_alerts, str_reports, screening_results, consents, data_access_requests, complaints, rate_limits.")
  }

  System_Ext(bankid, "BankID OIDC", "Norwegian eID provider. OIDC authorize/token/JWKS endpoints. auth.bankid.no (prod).")
  System_Ext(sumsub, "Sumsub", "KYC/AML identity verification. WebSDK (web), React Native SDK (mobile), webhooks for status updates.")
  System_Ext(openbanking, "Open Banking APIs", "Berlin Group NextGenPSD2. AISP (balance reads) and PISP (payment initiation) via Neonomics aggregator (planned).")
  System_Ext(sepa, "SEPA/SWIFT Networks", "International payment rails for remittance settlement to 30+ countries.")

  Rel(user, web, "HTTPS", "Browser — getdrop.no")
  Rel(user, mobile, "HTTPS", "iOS/Android app")
  Rel(merchant, web, "HTTPS", "Merchant dashboard")

  Rel(web, api, "HTTPS REST", "/api/* and /v1/* endpoints, JSON, httpOnly cookie")
  Rel(mobile, api, "HTTPS REST", "/v1/* endpoints, JSON, Bearer token")

  Rel(api, db, "SQL", "Parameterized queries via db.ts dual-driver abstraction")
  Rel(api, bankid, "OIDC", "Authorization code flow, JWKS token verification")
  Rel(api, sumsub, "REST + Webhooks", "Applicant creation, document checks, HMAC-verified webhooks")
  Rel(api, openbanking, "Berlin Group NextGenPSD2", "AISP balance reads, PISP payment initiation with SCA")
  Rel(api, sepa, "ISO 20022 (via banking partner)", "Remittance settlement to 30+ countries")
```

---

4. Component Overview

Component	Responsibility	Technology	Owner Team
drop-web	SSR web application, user onboarding, dashboard, send money, QR scan, merchant dashboard	Next.js 15, React 19, Tailwind v4, shadcn/ui	ALAI — Frontend
drop-api	REST API, BankID OIDC, JWT sessions, payment processing, KYC/GDPR/AML compliance	Hono v4, Node.js 22	ALAI — Backend
drop-mobile	Native iOS/Android, BankID auth, send money, QR scan, transaction history	Expo SDK 54, React Native	ALAI — Mobile
Database	Persistent storage, 19 tables, dual-driver (SQLite/PostgreSQL)	SQLite 3 (dev) / PostgreSQL 16 (prod)	ALAI — Backend
BankID OIDC	Strong Customer Authentication (SCA), Norwegian identity provider	OIDC 1.0	BankID Norge
Sumsub KYC	Document verification, PEP/sanctions screening, AML risk scoring	Sumsub API + SDK	Sumsub
Open Banking (AISP/PISP)	Balance reads from user's bank, payment initiation from user's bank	Berlin Group NextGenPSD2, Neonomics aggregator	ALAI + Neonomics

Component Descriptions

drop-web

Responsibility: Server-side rendered web application serving all 10 core screens. Handles the BankID OIDC redirect initiation, authentication callback (sets httpOnly cookie), and renders the full UI from Login to MerchantDashboard. Acts as BFF (Backend For Frontend) for the Next.js API routes at `/api/auth/*`. **Key Interfaces:** HTTP GET/POST to Next.js API routes (`/api/auth/bankid/*`); Hono API REST calls (`/v1/*`) via fetch with cookie credentials. **Rationale:** Separate from the API to allow independent scaling, enable SSR for SEO on the landing/marketing site, and encapsulate web-specific auth session management (httpOnly cookie).

drop-api

Responsibility: Central REST API serving both web and mobile clients. Owns all business logic: BankID OIDC code exchange, JWT issuance, transaction processing (remittance, QR payment), KYC initiation, GDPR endpoints, merchant management, admin operations, and AML compliance. Applies a 7-step middleware chain on every request. **Key Interfaces:** 26+ endpoints under `/v1/`. External calls to BankID token endpoint, Sumsub API, and Open Banking PISP/AISP. **Rationale:** Single source of business logic truth, consumed by both web (cookie auth) and mobile (Bearer

token auth). Hono v4 chosen for performance on Node.js 22 (see ADR-008).

drop-mobile

Responsibility: Native iOS and Android application. Provides the core payment features: BankID login, dashboard with balance, send money, QR scanner, transaction history, and profile management. **Key Interfaces:** Same Hono API `/v1/*` endpoints as web, using Bearer token (`Authorization: Bearer <jwt>`) instead of cookies. BankID auth via `expo-web-browser` + deep link `drop://auth/callback`. **Rationale:** Separate from drop-web to allow platform-native UX, native push notifications (future), and biometric auth (future).

---

5. Technology Stack

Layer	Technology	Version	Rationale
Frontend Framework	Next.js	15 (App Router)	SSR + RSC for performance; BFF capability for auth cookie management
UI Framework	React	19	Concurrent features, server components
Styling	Tailwind CSS	v4	Utility-first, design token support
UI Components	shadcn/ui (Radix UI)	Latest	Accessible primitives, keyboard nav, unstyled baseline
Mobile Framework	Expo (React Native)	SDK 54	Cross-platform iOS/Android, managed workflow, OTA updates
Backend Language	TypeScript / Node.js	Node 22 LTS	Type safety end-to-end, team expertise, shared types with frontend
Backend Framework	Hono	v4	Ultrafast edge-compatible framework; better performance than Express; native middleware chaining
Primary Database (prod)	PostgreSQL	16	ACID compliance, row-level security, rich indexing, AWS RDS managed
Development Database	SQLite (better-sqlite3)	3.x	Zero-config local dev, WAL mode, dual-driver abstraction switches transparently

Layer	Technology	Version	Rationale
Authentication	BankID OIDC + jose	2.0	Norwegian legal requirement for SCA; jose for JWKS verification
KYC/AML	Sumsub	API v1	Document verification, PEP/sanctions, Norwegian compliance coverage
Open Banking	Berlin Group NextGenPSD2 via Neonomics (planned)	v1.3.12+	PSD2 AISP/PISP; Neonomics aggregator for Nordic bank coverage
Error Tracking	Sentry	SDK v8	Full-stack error capture, session replay, performance tracing
Container Runtime	Docker	24+	Multi-stage build (4 stages), non-root user, Node 22 Alpine
Orchestration	AWS App Runner	-	Auto-scaling, managed TLS, no Kubernetes operational overhead
Edge / CDN	Cloudflare	-	WAF, DDoS protection, CDN for static assets, geo-blocking
Secrets	AWS Secrets Manager	-	JWT_SECRET, BANKID_CLIENT_SECRET, DATABASE_URL, SENTRY_DSN
CI/CD	GitHub Actions (planned)	-	Automated: tsc → lint → vitest → Docker build → ECR push → App Runner deploy

6. Data Flow Overview

6.1 Remittance Payment Flow (Write)

```
flowchart LR
    A([User — Web/Mobile]) -->|"POST /v1/transactions/remittance"| B[Hono API]
    B -->|"1. Verify JWT + session"| C[(PostgreSQL)]
    B -->|"2. Validate: KYC approved, recipient exists, amount 100-50000 NOK"| B
    B -->|"3. Lookup exchange rate"| C
    B -->|"4. Begin atomic transaction"| C
```

```
    C -->|"INSERT transactions status=processing"| C
    C -->|"INSERT audit_log"| C
    C -->|"INSERT notifications"| C
    B -->|"5. Initiate PISP payment"| D[Open Banking API]
    D -->|"SCA redirect URL"| B
    B -->|"6. Return 201 + redirect"| A
    A -->|"7. User completes BankID SCA at bank"| D
    D -->|"8. Webhook: payment confirmed"| B
    B -->|"9. UPDATE transactions status=completed"| C
```

6.2 Balance Read Flow (Read — AISP)

```
flowchart LR
    A([User]) -->|"GET /api/auth/me"| B[Next.js BFF / Hono API]
    B -->|"Verify JWT cookie"| C[(PostgreSQL)]
    C -->|"bank_accounts.balance (cached)"| B
    B -->|"If stale: GET /v1/accounts/{id}/balances"| D[Open Banking AISP]
    D -->|"Live balance"| B
    B -->|"UPDATE bank_accounts SET balance, balance_synced_at"| C
    B -->|"Return {totalBalance, accounts}"| A
```

---

7. Integration Points

7.1 External Integrations

System	Direction	Protocol	Auth	Data Exchanged	SLA/Criticality
BankID OIDC	Outbound	OIDC 1.0 / HTTPS	Client ID + Client Secret (code flow)	ID token (pid, name, birthdate), access token	99.9% / Critical — all auth blocked if down
Sumsub KYC	Outbound + Inbound webhooks	REST HTTPS + Webhooks	API token + HMAC-SHA256	Applicant data, documents, verification results, risk scores	99.5% / High — new registrations blocked

System	Direction	Protocol	Auth	Data Exchanged	SLA/Criticality
Open Banking (Neonomics/ASPSP)	Outbound	Berlin Group NextGenPSD2 / HTTPS	eIDAS QWAC cert + OAuth2	Account lists, balances (AISP); payment initiations, payment status (PISP)	99.5% / Critical — payments blocked if PISP down; AISP degrades to cached balance
SEPA/SWIFT	Outbound via banking partner	ISO 20022	Banking partner credentials	Remittance transfers (amounts, IBANs, reference)	Best-effort / High — delays expected on bank outages
Cloudflare	Inbound (proxied)	DNS + HTTPS	Cloudflare API key	HTTP traffic, TLS, WAF rules	99.99% / Critical — all traffic routed via Cloudflare
AWS Secrets Manager	Outbound	HTTPS	IAM role	JWT_SECRET, BANKID_CLIENT_SECRET, DATABASE_URL, SENTRY_DSN	99.99% / Critical — startup fails if unavailable
Sentry	Outbound	HTTPS (SDK)	DSN token	Error events, stack traces, performance traces	Best-effort / Low — observability only

7.2 Internal Service Integrations

Service	Integration Type	Protocol	Notes
drop-web → drop-api	Synchronous	REST HTTPS	Web auth via httpOnly cookie (`drop_token`); API calls to `/v1/*`
drop-mobile → drop-api	Synchronous	REST HTTPS	Bearer token in `Authorization` header; same `/v1/*` API endpoints
drop-api → PostgreSQL	Synchronous	TCP (SQL)	db.ts dual-driver abstraction; parameterized queries only

---

8. Deployment Overview

```
flowchart TB
    subgraph Internet
```

```
        Users[End Users — Browser + Mobile]
    end

    subgraph Cloudflare["Cloudflare Edge (getdrop.no)"]
        DNS[DNS]
        CDN[CDN — Static Assets /_next/static/*]
        WAF[WAF — OWASP CRS + custom rules]
        DDoS[DDoS Protection L3/L4/L7]
    end

    subgraph AWS["AWS eu-north-1 (Stockholm)"]
        subgraph AppRunner["AWS App Runner (PLANNED)"]
            WebApp[drop-web<br/>Next.js 15 standalone<br/>Node.js 22 Alpine<br/>Port 3000<br/>1-5 instances]
            API[drop-api<br/>Hono v4<br/>Node.js 22 Alpine<br/>Port 3001<br/>1-10 instances]
        end

        subgraph DataTier["Data Tier"]
            RDS[(RDS PostgreSQL 16<br/>db.t3.medium → db.r6g.large<br/>Multi-AZ — prod<br/>100GB gp3, auto-scale to 500GB<br/>30-day backup retention)]
        end

        subgraph Supporting["Supporting"]
            ECR[ECR — Container Registry<br/>Image scanning enabled]
            SM[Secrets Manager<br/>JWT_SECRET / BANKID_CLIENT_SECRET<br/>DATABASE_URL / SENTRY_DSN]
            CW[CloudWatch<br/>Logs + Metrics + Alarms]
        end
    end

    Users --> DNS
    DNS --> CDN
    CDN --> WAF
    WAF --> DDoS
    DDoS --> WebApp
    DDoS --> API
    WebApp --> RDS
    API --> RDS
    AppRunner --> ECR
    AppRunner --> SM
```

```
    AppRunner --> CW
```

Environments

Environment	URL	Purpose	Database	BankID	Scale
Development	http://localhost:3000 + :3001	Local dev via `docker compose up`	SQLite (`./data/drop.db`)	Mock (`BANKID_MOCK=true`)	Single instance
Staging	https://staging.getdrop.no	Pre-release validation, QA, E2E	RDS PostgreSQL (separate)	BankID test environment	1 replica
Production	https://getdrop.no	Live traffic	RDS PostgreSQL Multi-AZ	BankID production	Auto-scaled (1-5 web, 1-10 API)

---

9. Cross-Cutting Concerns

9.1 Authentication & Authorization

- **Strategy:** BankID OIDC (Authorization Code Flow) — email/password removed, returns 410 Gone
- **Identity Provider:** BankID Norge (`auth.bankid.no`) — OIDC 1.0, JWKS-verified ID tokens
- **Authorization Model:** Role-based — `user` (default) vs `merchant` (gated). Middleware `authMiddleware` + `merchantMiddleware` enforce per-endpoint.
- **Token Lifetime:** Web: 24h httpOnly cookie (`drop_token`); Mobile: 7d Bearer JWT in AsyncStorage
- **MFA:** Yes — BankID provides strong two-factor SCA (possession + knowledge/inherence) on every login and payment

9.2 Logging

- **Framework:** Hono native + console.log, captured by Sentry
- **Format:** JSON structured logs where available; request ID propagated via `x-request-id` header
- **Levels:** ERROR/WARN sent to Sentry; INFO to CloudWatch
- **Correlation IDs:** `x-request-id` generated per request (UUID), echoed in response header
- **Retention:** 90 days in CloudWatch
- **PII Handling:** National IDs stored as SHA-256 hash only; raw PII never logged

9.3 Error Handling

- **API Errors:** JSON envelope `{ error: "code", message: "...", details: [...] }`
- **Retry Strategy:** External API calls: exponential backoff [1s, 2s, 4s], max 3 retries
- **Circuit Breaker:** Open Banking API: 3 failures in 60s → 60s cooldown
- **Global Error Handler:** `middleware/error-handler.ts` — catches all unhandled errors, logs to Sentry, returns 500

9.4 Rate Limiting

- **Implementation:** Redis-less; DB-backed `rate_limits` table with SQLite/PostgreSQL dual support
- **Default Limits:** Auth endpoints: 10 req/60s per IP; Transactions: 10 req/60s per IP + 3 per-user; Exchange rates: 120 req/60s
- **Cloudflare WAF:** `/v1/auth/*` → challenge at 20 req/10s; `/v1/transactions/*` → block at 30 req/10s
- **Response:** HTTP 429 (no Retry-After header currently; planned)

9.5 Secrets Management

- **Tool:** AWS Secrets Manager (production); environment variables (development)
- **Rotation:** Manual rotation policy — planned automation via Secrets Manager rotation Lambda
- **Principle:** No secrets in code; `JWT_SECRET` has a dev-only fallback string that triggers a warning

9.6 Feature Flags

- **Tool:** Environment variables read at startup (in-memory, process lifetime)
- **Flags:** `CARDS_ENABLED` (default false), `ADVANCED_ANALYTICS` (default false), `WITHDRAW_ENABLED` (default false)
- **Toggle:** Restart required for flag changes; no runtime toggle UI

---

10. Quality Attributes & Architectural Trade-offs

Quality Attribute	Target	Approach	Trade-off
Availability	99.5% uptime	AWS App Runner multi-instance, Cloudflare 99.99% edge, RDS Multi-AZ	Higher AWS cost vs single-AZ

Quality Attribute	Target	Approach	Trade-off
Performance (p99 latency)	< 200ms API responses	No external cache (SQLite WAL / PostgreSQL handles load), Cloudflare CDN for static assets	No Redis cache — acceptable at current scale
Scalability	1,000 concurrent users (MVP)	App Runner auto-scale: 1-10 API instances, 1-5 web instances; stateless API (JWT)	All-or-nothing scaling (monolith)
Security	OWASP Top 10 compliant	Cloudflare WAF, parameterized SQL, httpOnly cookies, BankID SCA, HMAC webhooks	BankID adds auth flow complexity
Regulatory Compliance	PSD2, GDPR, AML (hvitvaskingsloven)	BankID SCA for payments, Sumsub KYC, 19-table compliance schema, STR filing	Compliance overhead slows feature delivery
Maintainability	Weekly deploys	Monolith-first (ADR-005), vitest test suite, TypeScript strict mode	Module boundary erosion risk without process isolation
Data Consistency	Strong (per transaction)	Atomic DB transactions for all financial operations, idempotency keys on payments	No eventual consistency — simpler but single-DB dependency

11. Key Architectural Decisions

ADR	Decision	Status	Date
ADR-001	Consolidate to single Hono backend (remove dual middleware)	Accepted	2026-02-12
ADR-003	Adopt PSD2 pass-through model — no wallet, no held funds	Accepted	2026-02-12
ADR-004	JWT in httpOnly cookies (web) + Bearer tokens (mobile)	Accepted	2026-02-12
ADR-005	Monolith-first architecture — extract microservices when team/scale demands	Accepted	2026-02-21
ADR-006	Dual-driver DB abstraction: SQLite (dev) / PostgreSQL (prod)	Accepted	2026-02-21

ADR	Decision	Status	Date
ADR-007	BankID as sole identity provider (email/password removed)	Accepted	2026-02-21
ADR-008	Hono v4 as the API framework	Accepted	2026-02-21
ADR-012	AWS App Runner for container hosting	Accepted	2026-02-21

12. Constraints & Assumptions

12.1 Constraints

#	Constraint	Category	Impact
C1	Users must be Norwegian residents (18+) with Norwegian BankID and +47 phone number	Regulatory	Limits market to Norway; no international expansion without separate licensing
C2	Drop must never hold customer funds (PSD2 pass-through model)	Regulatory	PISP/AISP architecture mandatory; wallet model legally excluded
C3	BankID SCA required for every financial operation (PISP payment)	Regulatory / PSD2 RTS	Each payment requires bank SCA redirect — adds UX friction
C4	5-year AML data retention (hvitvaskingsloven)	Regulatory	Compliance tables cannot be purged; storage costs grow over time
C5	GDPR Art. 17 right to erasure — soft delete + 5yr AML retention override	Regulatory	Cannot hard-delete user data if AML records exist
C6	Finanstilsynet PISP/AISP license not yet obtained (Phase 2 blocker)	Regulatory	Live Open Banking API calls not permitted until license or agent arrangement secured
C7	Monolith-first — all containers deploy together	Technical	No independent scaling per module; full deploy required for any change
C8	Budget: AWS Secrets Manager, App Runner, RDS — cost scales with usage	Business	Architecture chosen for low base cost; scales to higher tiers on growth

12.2 Assumptions

#	Assumption	Validation Method	Risk if Wrong
A1	Neonomics or Tink will provide Open Banking aggregator service for Phase 2 Nordic bank connectivity	Contract negotiation in Phase 2	Direct per-bank ASPSP integration required (significantly higher effort)
A2	BankID Norge will approve Drop's OIDC client registration	BankID developer portal application	Must use Vipps Login or alternative OIDC provider
A3	PostgreSQL on RDS handles expected transaction volume without read replicas at MVP	Load testing before production launch	Must add read replicas or implement caching layer
A4	App Runner rolling updates are sufficient (no true blue/green needed at MVP scale)	Monitor during first production deploy	Must implement custom blue/green via ALB traffic shifting

---

13. Risks & Mitigations

Risk	Likelihood	Impact	Score	Mitigation	Contingency
Finanstilsynet license delayed (>12 months)	3	5	15	Use licensed PSP agent arrangement (1-3 months setup) while applying	Demo/mock mode continues; partner with licensed PSP
BankID integration blocked (client not approved)	2	5	10	Apply early; prepare Vipps Login as alternative OIDC (same pid claim)	Vipps Login fallback (same architecture, different OIDC endpoints)
Open Banking ASPSP API unavailability (AISP)	4	2	8	Show cached balance with staleness indicator	Degrade gracefully: display last-known balance
Open Banking ASPSP API unavailability (PISP)	3	5	15	Circuit breaker; notify user to retry	Payment cannot proceed — user notified with ETA

Risk	Likelihood	Impact	Score	Mitigation	Contingency
Single database bottleneck (PostgreSQL)	2	4	8	Connection pooling, read replicas when needed, App Runner horizontal scale	Add read replicas, implement CQRS for transaction reads
Data breach via SQL injection	1	5	5	Parameterized queries (db.ts enforces), WAF, no raw SQL strings in routes	GDPR breach notification within 72h, incident response plan
Sumsub KYC outage (new user registrations blocked)	2	3	6	Retry queue; existing approved users unaffected	Queue new registrations; manual KYC review for priority users

Approval

Role	Name	Date	Signature
Author	Petter Graff	2026-02-23	
Technical Lead	John (AI Director)		
Security Review			
Approver (CEO)	Alem Bašić		

Revision #5
Created 2026-02-23 12:04:54 UTC by John
Updated 2026-05-31 20:03:08 UTC by John