Reviews & Reports

Architecture reviews, team reports, validation

- Petter Graff Architecture Review
- Architecture Validation Report

Petter Graff Architecture Review

Drop Fintech Platform — Architecture Review

Reviewer: Petter Graff (Software Architect, CTO Pratexo) **Review Date:** 2026-02-22 **Subject:** Drop — Norwegian fintech payment application (remittance + QR payments) **Model:** PSD2 pass-through (AISP/PISP) — Drop never holds customer funds

> **NOTE (2026-03-03):** This review was conducted against the pre-ADR-014 codebase. SQLite/dual-driver findings are historical — resolved by ADR-014 (PostgreSQL-only + Drizzle ORM).

---

1. Overall Assessment

Grade: C+ (6.5/10)

Drop has a **solid conceptual foundation** for a PSD2-regulated fintech application. The architectural documentation is comprehensive, the compliance framework is thoughtfully designed, and the core technical decisions (monolith-first, dual-database abstraction, BankID-only auth) are appropriate for an MVP. However, **critical production-readiness gaps** exist across security, data integrity, and operational resilience that would prevent regulatory approval and create significant business risk.

What I see here: An engineering team that understands fintech compliance requirements at a *documentation level* but has not yet built the operational muscle memory to implement them correctly. The Vault Squad analysis is accurate — you have 17 CRITICAL findings that must be fixed before processing a single real transaction.

This is **not a failing grade** — it's a realistic early-stage fintech build. But you're not production-ready, and pretending otherwise would be dangerous.

---

2. What's Done Well

2.1 Regulatory Awareness

- **19-table schema** includes 7 dedicated compliance tables (`audit_log`, `aml_alerts`, `str_reports`, `screening_results`, `consents`, `data_access_requests`, `complaints`) — most fintechs bolt these on later
- **Pass-through PSD2 model** correctly avoids e-money license complexity — Drop positions as PISP/AISP only
- **BankID-only authentication** is the right call for Norwegian fintech (SCA by design, no password management)
- **Dual-database abstraction** (`db.ts`) with SQL compatibility translation is clean engineering — SQLite for dev speed, PostgreSQL for production reliability

2.2 Security Fundamentals

- **Parameterized SQL throughout** — zero SQL injection vulnerabilities
- **bcrypt with 12 rounds** for password hashing
- **Idempotency keys** on transactions with unique index — double-charge protection exists
- **Structured audit logging** with IP, user-agent, request-id correlation

2.3 AML/Compliance Framework

The transaction monitoring module (`transaction-monitor.ts`) has 5 rule types:

- Structuring detection (multiple small txns avoiding thresholds)
- Velocity checks (>5 txns/hour)
- High-amount flagging (>25K NOK)
- High-risk corridor detection (FATF grey-list countries)
- Unusual pattern analysis (3x user average)

Problem: It's never called from any API route (Vault Squad finding BE-C2). Dead code doesn't count.

2.4 Documentation Quality

C4 diagrams, ADRs, and HLD documents are better than 80% of fintech startups. Architecture is understandable, decisions documented with rationale, trade-offs acknowledged.

---

3. Critical Improvements (Production Blockers)

3.1 Authentication Catastrophe

Finding	Impact	Source
JWT secret defaults to hardcoded `"dev-secret-change-in-production"`	Anyone can forge tokens if env var missing	`auth.ts:8`
No OIDC `state` validation	CSRF on authentication flow	`bankid.ts:80-108`
No OIDC `nonce` validation	ID token replay attacks	`bankid.ts:93-101`
Cookie missing `Secure` flag	Tokens sent over HTTP in plaintext	`auth.ts:206`

Risk: Complete account takeover possible.

Fix:

```
// Startup validation
if (process.env.NODE_ENV === 'production' &&
    process.env.JWT_SECRET === 'dev-secret-change-in-production') {
  throw new Error('FATAL: Production requires real JWT_SECRET');
}

// BankID callback — validate state + nonce
const storedState = cookies.get('bankid_state');
const storedNonce = cookies.get('bankid_nonce');
if (state !== storedState) throw new Error('Invalid state');
if (idToken.nonce !== storedNonce) throw new Error('Invalid nonce');

// Cookie flags
cookies.set('drop_token', token, {
  httpOnly: true,
  secure: true,
  sameSite: 'strict',
  maxAge: 1800  // 30min, not 7 days
});
```

3.2 PISP Call Inside Database Transaction

External HTTP call (30s timeout) while holding database write lock. Under load → deadlocks, data loss, double-charging.

Fix: Two-phase commit:

```
// Phase 1: Create pending transaction (fast)
const txId = await createPendingTransaction(...);

// Phase 2: Initiate payment OUTSIDE transaction
try {
  const result = await initiateRemittance(...);
  await updateTransactionStatus(txId, result.status);
} catch (error) {
  await updateTransactionStatus(txId, 'failed');
}
```

3.3 TEXT Timestamps Everywhere

All 30+ timestamp columns use TEXT storing ISO 8601 strings. No timezone awareness, no native date math, ISO 20022 non-compliant.

Fix: Migrate to TIMESTAMPTZ in PostgreSQL migration.

3.4 No Database Connection Management

PostgreSQL pool has no timeout configuration. One slow query exhausts entire pool → cascading failure.

Fix:

```
const pool = new Pool({
  connectionString: process.env.DATABASE_URL,
  max: 20,
  connectionTimeoutMillis: 5000,
  idleTimeoutMillis: 30000,
  statement_timeout: 30000,
  query_timeout: 30000,
});
```

3.5 Zero Transaction Monitoring

AML monitoring code exists but is never called from any API route. Hvitvaskingsloven §7 requires real-time monitoring.

Fix: Wire `checkTransaction()` before PISP initiation. Block critical alerts.

3.6 Missing Open Banking Consent Lifecycle

No `ob_consents` table. Can't enforce 90-day consent renewal, access frequency limits, or consent revocation.

Fix: Create `ob_consents` table with consent tracking, expiry, access counting.

---

4. Strategic Improvements (Scale)

4.1 Network Topology & Failure Modes

No documented failure scenarios or retry strategies. Need failure domain mapping for BankID, PISP, PostgreSQL.

4.2 No Double-Entry Bookkeeping

Single-entry transaction model. Finanstilsynet will ask for general ledger proving credits = debits.

Fix: Add `transaction_legs` table with debit/credit entries per transaction.

4.3 Missing Composite Indexes

High-frequency queries will full-table-scan at scale.

Fix: Add composite indexes: `(user_id, created_at DESC)` on transactions, audit_log, notifications.

4.4 No Table Partitioning Strategy

`audit_log` will reach 90M rows in 5 years. Plan partitioning by month.

---

5. Architecture Smell Flags

5.1 Rate Limiting Writes to PostgreSQL

Every request writes to PostgreSQL for rate limiting. Should use in-memory (Map/Redis).

5.2 Fire-and-Forget Audit Logging

Critical actions (login, payment, consent) use async audit — should be synchronous for regulatory compliance.

5.3 No Circuit Breaker for External APIs

If PISP API goes down, every request waits 30s then fails. Need circuit breaker (fail fast after 5 failures, retry after 60s).

---

6. Questions Before Sign-Off

Security & Auth

1. JWT key rotation process?
2. BankID downtime fallback?
3. Session hijacking detection (2 IPs simultaneously)?
4. Direct database access logging?

Data Integrity

5. DST transition handling with TEXT timestamps?
6. Exchange rate locking between quote and execution?
7. Transaction status if PISP call times out?
8. Client-generated idempotency key security?

Compliance

9. How to enforce 90-day AISP consent expiry?
10. Who reviews AML alerts and how fast?
11. STR filing integration with Økokrim?
12. GDPR data export time for user request?

Operations

13. Database backup RPO/RTO?
14. Zero-downtime schema migration strategy?
15. PagerDuty alert rules?
16. AWS eu-north-1 failover plan?

---

7. Priority Roadmap

Period	Focus	Priority
Month 1-2	Security hardening (auth, CSRF, PISP fix, DB timeouts, AML wiring)	CRITICAL
Month 3-4	Data integrity (timestamps, ob_consents, double-entry, indexes)	HIGH
Month 5-6	Operational resilience (circuit breakers, Multi-AZ, DR runbook)	HIGH
Month 7+	Scale preparation (Redis rate limit, read replicas, monitoring)	MEDIUM

Final Verdict

> "Fintech is not about having fancy architecture diagrams — it's about operational discipline. Every timeout needs a value. Every transaction needs an audit log. Every external call needs a circuit breaker. These aren't optional extras — they're the difference between a production system and a demo. You're 60% there. The remaining 40% is where most fintechs fail."

— Petter Graff, CTO Pratexo

Architecture Validation Report

Architecture Docs Validation Report

Date: 2026-02-21 **Scope:** All 41 architecture docs in `docs/architecture/` **Method:** 3 critic agents (code, consistency, completeness) + 1 validator agent **Source of truth:** Actual source code in `src/drop-api/`, `src/drop-app/`, `src/drop-mobile/`

Summary

Metric	Count
Total findings submitted	104
Confirmed (real issues)	**70**
Rejected (false positives)	34
Docs affected	34 of 41

Severity Breakdown

Severity	Count	Description
HIGH	21	Factual errors, security gaps, regulatory compliance issues
MEDIUM	37	Incorrect details, cross-doc inconsistencies, missing coverage
LOW	12	Minor naming issues, broken cross-refs, cosmetic

Root Cause Clusters

Many findings share a common root cause. Fixing the root cause resolves multiple findings at once:

#	Root Cause	Findings	Fix
1	**Next.js version: 16 to 15**	CODE-001 thru CODE-004, CODE-030, CODE-042 (6)	Global find-replace "Next.js 16" to "Next.js 15"
2	**JWT lifetime: 24h/7d to 7d uniform**	CODE-011, CODE-012, CODE-013, CONS-001, CONS-002, CONS-003 (6)	Replace all "24h (web) / 7d (mobile)" with "7d (all clients)"
3	**SameSite: strict to Lax**	CODE-009, CODE-010, CODE-028, CONS-004 (4)	Replace "sameSite=strict" to "sameSite=Lax"
4	**middleware.ts to middleware/*.ts**	CODE-015, CODE-025, CODE-026 (3)	Fix all "middleware.ts" references to actual file paths
5	**Deprecated 410 endpoints still documented**	CODE-045, COMP-002, CODE-031, CONS-014 (4)	Remove/update email+password flows, document 410s
6	**Amount units: NOK to ore in DB**	CODE-008, CONS-019, COMP-020 (3+)	Clarify DB stores ore, API converts to/from NOK
7	**Deployment: aspirational vs actual**	CODE-005, CODE-024, CONS-007, CONS-008 (4)	Mark AWS/blue-green as planned; document Docker Compose as current
8	**Demo mode undocumented**	CODE-019, CODE-020, CODE-034, CODE-047, COMP-001 (5)	Document demo-login flow and SEED_DEMO override
9	**Mobile storage: SecureStore to AsyncStorage**	CODE-014, CONS-005 (2)	Fix to AsyncStorage (expo-secure-store is devDep)
10	**Data classification scheme mismatch**	CONS-006 (1)	Unify to one taxonomy across security + data architecture

---

HIGH Severity Findings (21)

H1. Next.js Version Wrong (6 docs)

IDs: CODE-001, CODE-002, CODE-003, CODE-004, CODE-030, CODE-042 **Docs:** component-overview, system-context, container-diagram, deployment-architecture, ADR-011 **Claim:** "Next.js 16" **Reality:** `next: ^15.5.12` (src/drop-app/package.json:24) **Fix:** Replace all "Next.js 16" with "Next.js 15"

H2. Deployment Architecture Aspirational (2 docs)

IDs: CODE-005, CODE-024 **Doc:** deployment-architecture.md **Claim:** AWS App Runner, ECR, RDS, Secrets Manager, Fly.io staging **Reality:** Only docker-compose.yml and docker-compose.production.yml exist. No fly.toml, no apprunner.yaml, no CI/CD. **Fix:** Mark deployment targets as planned. Document current Docker Compose setup.

H3. Dockerfile 4 Stages, Not 3

ID: CODE-006 **Doc:** deployment-architecture.md **Claim:** 3-stage build (deps, builder, runner) **Reality:** 4 stages: deps, test, builder, runner. Test stage is mandatory quality gate. **Evidence:** src/drop-app/Dockerfile:15 (test), :48 (builder), :81 (runner) **Fix:** Document 4-stage pipeline. Highlight mandatory test gate.

H4. Build Tools in Production Image

ID: CODE-007 **Doc:** deployment-architecture.md **Claim:** "No build tools in final image" **Reality:** Runner stage installs python3, make, g++ (Dockerfile:82) **Fix:** Update doc AND flag as security concern.

H5. Amount Units: NOK vs ore

ID: CODE-008 **Doc:** flow-qr-payment.md **Claim:** Amount in response is NOK (e.g., amount: 129) **Reality:** DB stores ore. `nokToOre()` at transactions.ts:13. Seed data: 4500000 ore = 45000 NOK. **Fix:** Document dual representation: API accepts/returns NOK, DB stores ore.

H6. JWT Lifetime Wrong (5 docs)

IDs: CODE-011, CODE-012, CODE-013, CONS-001, CONS-002, CONS-003 **Docs:** flow-login-authentication, data-architecture, ADR-004, ADR-007 **Claim:** "24h (web), 7d (mobile)" **Reality:** `signToken(payload, expiresIn='7d')` -- uniform 7d. No platform branching. (auth.ts:42) **Fix:** Replace all "24h web / 7d mobile" with "7d (all clients)"

H7. Data Classification Scheme Mismatch

ID: CONS-006 **Docs:** data-architecture.md vs security-architecture.md **Claim:** data-arch uses CRITICAL/HIGH/MEDIUM/LOW; security uses CRITICAL/RESTRICTED/CONFIDENTIAL/INTERNAL/PUBLIC **Fix:** Adopt one taxonomy across both docs.

H8. Registration Flow is BankID-Only

ID: CONS-014 **Docs:** component-overview.md, flow-registration-onboarding.md **Claim:** "4 steps (info, OTP, PIN, success)" **Reality:** /register returns 410. Users auto-created on first BankID login (bankid.ts:findOrCreateUser). **Fix:** Update to BankID auto-registration.

H9. Demo Mode Architecture Undocumented

ID: COMP-001 **Evidence:** auth.ts:131-158 (demo-login), mode.ts (isDemoMode), payments.ts (demo branching) **Fix:** Create flow-demo-mode.md or add section to existing auth docs.

H10. Token Refresh + 410 Endpoints Undocumented

ID: COMP-002 **Evidence:** auth.ts:201-210 (POST /refresh), auth.ts:109-128 (three 410 endpoints) **Fix:** Add refresh flow and deprecated endpoints table.

H11. GDPR Endpoints Undocumented

ID: COMP-003 **Evidence:** user.ts:132-360 -- POST /objection, /rectification, /restriction **Claim in docs:** "Manual process" (data-lifecycle.md) **Reality:** Fully implemented API endpoints with audit logging. **Fix:** Update data-lifecycle.md. Create GDPR rights matrix.

H12. Withdrawal/Angrerett Flow Undocumented

IDs: COMP-004, COMP-005 **Evidence:** withdrawal.ts creates withdrawal_requests table at runtime. **Fix:** Create flow-withdrawal.md. Add table to schema docs.

H13. Error Handling Chain Undocumented

ID: COMP-007 **Evidence:** error-handler.ts, Sentry (captureError), alerts. app.ts:50 mounts globalErrorHandler. **Fix:** Add observability section to docs.

H14. Merchant = Admin (Security Gap)

ID: COMP-013 **Evidence:** admin.ts:14-16 -- `isAdmin(role) { return role === 'merchant' }` **Impact:** Any merchant can access audit logs, screening, and file STRs. **Fix:** Document RBAC model. Flag for security review.

H15. Transaction Reconciliation Missing

ID: COMP-024 **Evidence:** QR payments INSERT as 'completed' immediately (transactions.ts:459). Remittances as 'processing' with no webhook/polling. **Fix:** Document the gap. Note production needs async reconciliation.

MEDIUM Severity Findings (37)

ID	File	Issue	Fix
CODE-009	security-architecture.md	SameSite=strict, actual Lax	Change to Lax
CODE-010	flow-login-authentication.md	Cookie: 24h strict, actual 7d Lax	Fix both values
CODE-014	security-architecture.md	SecureStore, actual AsyncStorage	Fix storage name
CODE-015	security-architecture.md	middleware.ts, actual middleware/auth.ts + rate-limit.ts	Fix file paths
CODE-016	container-diagram.md	merchantMiddleware "extends auth", actually independent	Fix description
CODE-018	deployment-architecture.md	JWT_SECRET "cwd hash", actual static string	Fix fallback desc
CODE-019	flow-login-authentication.md	Demo email: amir@example.com, actual demo@example.test	Fix email
CODE-020	flow-login-authentication.md	Demo calls POST /login, actual POST /demo-login (no creds)	Rewrite demo section
CODE-021	container-diagram.md	Rate limit missing per-user limit (3 req)	Add dual limits
CODE-023	ADR-012 vs deployment-arch	Blue/green contradiction	Make consistent
CODE-025	database-design.md	middleware.ts:11, actual middleware/rate-limit.ts:11	Fix path
CODE-026	data-architecture.md	Rate limit cleanup "every check", actual every 100 checks	Fix frequency + path
CODE-027	data-architecture.md	"6 tables" but lists 7	Fix count
CODE-028	Backend LLD correct, security-arch wrong on SameSite	Fix security-arch	
CODE-038	flow-qr-payment.md	Merchant ID regex doesn't match seed data	Relax format spec

ID	File	Issue	Fix
CODE-043	flow-login-auth.md	Cross-refs list deprecated endpoints	Update to current
CONS-004	Frontend vs backend LLD	SameSite contradiction	Fix frontend LLD
CONS-005	Security vs backend LLD	SecureStore vs AsyncStorage	Fix security-arch
CONS-007	ADR-012 vs deployment-arch	Health check: /api/health vs /v1/health	Standardize
CONS-008	ADR-012 vs deployment-arch	Blue/green contradiction (dup of CODE-023)	Make consistent
CONS-009	ADR-008 vs container-diagram	Rate limiting "in-memory", actual DB-backed	Fix ADR-008
CONS-011	data-architecture.md	"6 tables" header, 7 listed (dup of CODE-027)	Fix count
CONS-025	Bank linking vs open banking	AISP consent: 180d, should be 90d per PSD2	Use 90d (regulatory)
CONS-019	flow-qr-payment.md	Fee "charged to merchant", user pays total	Clarify fee model
COMP-006	DATABASE-SCHEMA.md	Missing otp_codes table reference	Document or remove dead code
COMP-008	All docs	Feature flags: 8 flags, no mapping doc	Create feature-flags.md
COMP-009	security-architecture.md	AML rules: 5 specific rules undocumented	Add detection rules
COMP-010	flow-remittance.md	Missing /summary and /:id/receipt endpoints	Add to flow
COMP-012	All docs	Middleware chain undocumented	Add request lifecycle
COMP-014	flow-qr-payment.md	HMAC verification is optional, not mandatory	Clarify + security note
COMP-016	All docs	Data retention cron undocumented	Add to data-lifecycle
COMP-020	flow-remittance.md	PSD2 disclosure endpoint undocumented	Add /disclosure to sequence
CODE-033	deployment-architecture.md	SERVICE_MODE "mock", actual "demo"	Fix value
CODE-034	flow-login-auth-backend.md	Demo mode env: NEXT_PUBLIC..., actual DROP_MODE	Fix env var
CODE-045	flow-login-auth.md vs backend	Frontend describes email/password as active	Update frontend LLD

LOW Severity Findings (12)

ID	File	Issue
CODE-030	ADR-011	Next.js 16 to 15
CODE-031	component-overview.md	Registration 4-step, actual BankID-only
CODE-032	migration-strategy.md	Cross-ref path: ../lld/ should be ../hld/
CODE-035	audit-architecture.md	idx_audit_log_timestamp index doesn't exist
CODE-036	security-architecture.md	bcrypt file ref (actually correct)
CODE-040	container-diagram.md	"Offline-capable" contradicts component-overview "No offline"
CODE-041	database-design.md	idx_tx_idempotency breaks naming convention
CONS-010	component-overview vs system-context	Duplicate doc ID "HLD-001"

Recommended Fix Priority

Batch 1: Global Find-Replace (resolves ~20 findings)

1. "Next.js 16" to "Next.js 15" (all docs)
2. "24h (web)" / "7d (mobile)" to "7d (all clients)" (all docs)
3. "sameSite=strict" to "sameSite=Lax" (all docs)
4. "middleware.ts" to correct paths (all docs)

Batch 2: Section Rewrites (resolves ~15 findings)

1. Deployment architecture: mark AWS as planned, document Docker Compose current state
2. Auth flow: remove email/password, document BankID-only + demo-login
3. QR payment: fix amount units, fee model, HMAC optionality
4. Registration: BankID auto-creation, not 4-step

Batch 3: Missing Documentation (resolves ~15 findings)

1. Demo mode architecture (auth.ts, mode.ts, payments.ts branching)
2. GDPR endpoints (user.ts: objection, rectification, restriction)
3. Withdrawal flow (withdrawal.ts)
4. Feature flag mapping (feature-flags.ts: 8 flags)
5. Error handling chain (error-handler.ts, Sentry, alerts)
6. AML monitoring rules and thresholds
7. Data retention cron job
8. PSD2 disclosure endpoint
9. Middleware request lifecycle

Batch 4: Cross-Doc Consistency (resolves ~10 findings)

1. Unify data classification taxonomy
2. Fix ADR contradictions (blue/green, rate limiting, health check paths)
3. Fix AISP consent period (180d to 90d per PSD2)
4. Assign unique document IDs

Non-Doc Issues Found (for engineering backlog)

These are code issues discovered during documentation review:

#	Issue	File	Severity
1	Build tools (python3, make, g++) in production Docker image	Dockerfile:82	HIGH
2	merchant role = admin role (any merchant can access audit/screening/STR)	admin.ts:14	HIGH
3	SEED_DEMO=true can enable demo data in production	db.ts:241	HIGH
4	HMAC verification on QR payments is optional	transactions.ts:400	MEDIUM

#	Issue	File	Severity
5	withdrawal_requests table created at runtime, not in schema	withdrawal.ts:34	MEDIUM
6	No async reconciliation for transactions	transactions.ts	MEDIUM
7	Dead otp_codes cleanup code	cron.ts:84	LOW
8	idx_tx_idempotency breaks naming convention	db.ts:190	LOW

Generated by drop-critics team: code-critic, consistency-critic, completeness-critic, validator