Project Documentation

Project charters, plans, pipeline reviews, incident reports

- Production Plan
- Project Charter
- Risk Register
- ZiCA Business Case
- Workflow
- Incident Report
- Hallucination Analysis (Feb 2026)

Production Plan

Drop — Production Plan

Datum: 21. februar 2026 **Status:** Odobren (Alem) **Cilj:** Go-Live sa beta korisnicima

---

1. Arhitektura — Bez banke

Drop ne treba banku kao partnera. Neonomics (Bergen, Norveška) ima PSD2 licencu i konekcije sa svim norveškim bankama.

```
Korisnik → Drop app → Neonomics API → Korisnikova banka (DNB, SpareBank 1, Nordea...)
                ↓
          BankID (autentifikacija)
```

- **Drop** = frontend + biznis logika + merchant mreža
- **Neonomics** = PSD2 licenca + Open Banking konekcije (PISP/AISP)
- **BankID** = autentifikacija (OIDC)
- **Korisnikova banka** = izvršava plaćanje, čuva novac

Drop nikad ne drži novac. Sve ide direktno sa bankovnog računa korisnika.

---

2. Revenue Model

2.1 QR pla?anja (merchant fee)

Stavka	Vrijednost
Fee	**1.0%** od iznosa
Ko plaća	Merchant (ne kupac)
Primjer	Kupac plati 500 NOK → Merchant primi 495 NOK → Drop zaradi 5 NOK
Konkurencija	Vipps 1.75%, Nets 2.0-2.75% → **Drop je najjeftiniji**

2.2 Remittance (transfer fee)

Stavka	Vrijednost
Fee	**0.5%** od iznosa (min 10 NOK, max 500 NOK)
Ko plaća	Pošiljalac
Primjer	Pošalji 5.000 NOK u Srbiju → Fee 25 NOK → Drop zaradi 25 NOK
Konkurencija	Western Union 5-10%, Wise 0.7-1.5% → **Drop je najjeftiniji**

2.3 FX Spread (faza 2)

Stavka	Vrijednost
Markup	**0.1-0.3%** na mid-market rate
Ko plaća	Ugrađeno u exchange rate (transparentno)
Primjer	Mid-market 1 NOK = 10.17 RSD → Drop rate = 10.14 RSD → Spread profit

2.4 Revenue projekcija

Metrika	Mjesec 6	Mjesec 12	Mjesec 24
Aktivni korisnici	500	5.000	25.000
QR transakcije/mj	2.000	30.000	200.000
Prosječna QR vrijednost	200 NOK	250 NOK	300 NOK
Remittance transakcije/mj	200	3.000	15.000
Prosječni remittance iznos	3.000 NOK	3.500 NOK	4.000 NOK
QR prihod	4.000 NOK	75.000 NOK	600.000 NOK
Remittance prihod	3.000 NOK	52.500 NOK	300.000 NOK
Ukupno mjesečno	**7.000 NOK**	**127.500 NOK**	**900.000 NOK**

2.5 Troškovi

Stavka	Mjesečno
AWS infrastruktura	~1.000 NOK (€80-150)
Neonomics (per-transaction)	~0.50-5 NOK po transakciji

Stavka	Mjesečno
BankID	Uključeno u Neonomics
Cloudflare	Free
Apple Developer	99 USD/godišnje
Google Play	25 USD jednokratno
Ukupni fiksni troškovi	**~1.500 NOK/mj**

2.6 Break-even

Sa fiksnim troškovima od ~1.500 NOK/mj:

- Treba ~150 QR transakcija od 200 NOK (1% × 200 × 150 = 3.000 NOK)
- Ili ~100 remittance transakcija od 3.000 NOK (0.5% × 3000 × 100 = 1.500 NOK)
- **Break-even: ~250 transakcija mjesečno** — realistično u prvih 3 mjeseca

---

3. AWS Infrastruktura

Što imamo (spremno)

☑ Terraform moduli (App Runner, RDS, ECR, Secrets Manager, Cloudflare)

☑ Dockerfajlovi za app + API (multi-stage, test gate)

☑ GitHub Actions CI/CD (auto-deploy)

☑ Health check endpointi

☑ PostgreSQL 16 in all environments (ADR-014, 2026-03-03 — supersedes dual-driver approach)

☑ 19 tabela dizajnirano, migracije spremne

☑ DPIA, security architecture, compliance tables

☑ BankID OIDC implementacija (sandbox radi)

Mjese?ni AWS trošak: ~€80-150

Servis	Namjena	Cijena
App Runner (×2)	Web + API	~€10-20/mj
RDS PostgreSQL 16	Baza podataka	~€30-50/mj

Servis	Namjena	Cijena
ECR	Docker registry	~€1/mj
Secrets Manager	Ključevi i tajne	~€2/mj
CloudWatch	Logovi + alarmi	~€5-10/mj
Cloudflare	CDN + WAF + DNS	Free
Route 53	DNS backup	~€1/mj

Region: `eu-north-1` (Stockholm) — GDPR compliant, najbliže Norveškoj.

---

4. Kriti?ni put do Go-Live

Faza 1: Neonomics + Infra (2-4 sedmice)

#	Task	Vlasnik	Effort
1	Neonomics ugovor + API pristup	Alem	1-2 sedmice
2	`terraform apply` — podigni AWS	John	1 dan
3	Neonomics API integracija (PISP + AISP)	John	2-3 sedmice
4	BankID sandbox → Neonomics BankID	John	2 dana
5	Staging deploy + E2E test	John	2 dana

Faza 2: Beta (2-4 sedmice)

#	Task	Vlasnik	Effort
6	10 beta korisnika (prijatelji/porodica)	Alem	1 sedmica
7	5 beta merchantov (lokalne radnje)	Alem	2 sedmice
8	Bug fixing iz beta feedbacka	John	ongoing
9	App Store submit (iOS + Android)	John + Alem	1 sedmica

Faza 3: Launch (2-4 sedmice)

#	Task	Vlasnik	Effort
10	Production deploy	John	1 dan
11	App Store approve + publish	Apple/Google	1-2 sedmice
12	Prvih 100 korisnika	Alem (marketing)	ongoing
13	FX rate API integracija (Wise/CurrencyCloud)	John	1 dan

Timeline

```
Sedmica 1-2:  Neonomics ugovor + terraform apply
Sedmica 3-4:  Neonomics API integracija
Sedmica 5-6:  Beta (10 korisnika + 5 merchantov)
Sedmica 7-8:  App Store submit + production
Sedmica 9+:   Launch + growth
```

Realistični Go-Live: ~8 sedmica od Neonomics ugovora.

---

5. Zašto ne treba banka

Pitanje	Odgovor
Ko daje pristup bankama?	**Neonomics** (PSD2 licenca, konekcije sa svim NO bankama)
Ko drži korisnikov novac?	**Korisnikova banka** (Drop nikad ne drži novac)
Ko autentificira korisnika?	**BankID** (via Neonomics ili direktno)
Ko izvršava plaćanje?	**Korisnikova banka** (Drop samo inicira via PISP)
Ko ima PSD2 licencu?	**Neonomics** (Drop radi kao njihov agent)
Šta Drop radi?	**Frontend + biznis logika + merchant mreža + UX**

Drop je "samo" app — ali app koji je jeftiniji, brži i jednostavniji od konkurencije. Slično kao što Uber ne posjeduje aute, Drop ne posjeduje banke.

---

6. Konkurentska prednost

Faktor	Drop	Vipps	Western Union	Wise
QR merchant fee	**1.0%**	1.75%	N/A	N/A
Remittance fee	**0.5%**	N/A	5-10%	0.7-1.5%
BankID login	Da	Da	Ne	Ne
Instant QR	Da	Da	Ne	Ne
Remittance	Da	Ne	Da	Da
Oboje u jednoj app	**Da**	Ne	Ne	Ne

Jedini koji nudi I QR plaćanja I remittance u jednoj app — po najnižim cijenama.

---

7. Rizici

Rizik	Vjerovatnoća	Uticaj	Mitigacija
Neonomics odbije/kasni	Nizak	Visok	Alternativa: Enable Banking, Yapily
Regulatorne promjene	Nizak	Srednji	PSD3 je evolucija PSD2, ne revolucija
Merchant adoption spor	Srednji	Srednji	Fokus na niche (diaspora radnje) pa širi
Vipps reagira snižavanjem cijena	Srednji	Nizak	Vipps nema remittance — naša prednost ostaje
Cyber napad	Nizak	Visok	WAF, rate limiting, BankID SCA, audit logs

---

Dokument kreirao: John (AI Director) Odobrio: Alem Bašić (CEO)

Project Charter

Project Charter: [PROJECT NAME]

Date: YYYY-MM-DD **Status:** Draft | In Review | Approved **Project Manager:** [Name] **Tech Lead:** [Name] **Client:** [Name / Company]

---

1. Project Overview

Project Name: [Name] **Client:** [Company Name] **Contact:** [Name, email] **Package:** [Package name and price] **Timeline:** [X weeks from contract signing] **Expected Start:** [Date] **Expected Completion:** [Date]

2. Problem Statement

[What problem does this project solve? 2-3 sentences.]

3. Project Objectives

1. [Objective 1]
2. [Objective 2]
3. [Objective 3]

4. Scope

In Scope

- ☐ [Deliverable 1]
- ☐ [Deliverable 2]

☐ [Deliverable 3]

Out of Scope

- [Item 1]
- [Item 2]

5. Deliverables

#	Deliverable	Description	Due
1			Week X
2			Week X
3			Week X

6. Timeline & Milestones

Milestone	Date	Gate
Kick-off		Charter approved
Design Complete		Design review passed
MVP / Alpha		Core features working
Beta		All features, testing
Launch		UAT signed off

7. Team & RACI

Role	Person/Agent	R/A/C/I
CEO	Alem	A (approvals)
AI Director	John	A (delivery)
Project Manager	[Agent]	R (coordination)
Tech Lead	[Agent]	R (architecture)
Scrum Master	[Agent]	R (process)
Developer	[Agent]	R (implementation)

Role	Person/Agent	R/A/C/I
Designer	[Agent/Human]	R (design)
QA	[Agent]	R (testing)

8. Budget

Item	Amount	Notes
Total	[NOK]	
Deposit (50%)	[NOK]	Upon contract signing
Final (50%)	[NOK]	Upon delivery

9. Risk Register

#	Risk	Probability	Impact	Mitigation
1		L/M/H	L/M/H	
2		L/M/H	L/M/H	
3		L/M/H	L/M/H	

10. Success Criteria

- ☐ [Criterion 1 — measurable]
- ☐ [Criterion 2 — measurable]
- ☐ [Criterion 3 — measurable]

11. Communication Plan

What	How	Frequency	Who
Status update	Standup file	Daily	PM → John
Sprint review	Meeting notes	Bi-weekly	Team → Client
Decisions	Decision record	As needed	John
Escalations	Direct session	Immediate	John → Alem

12. Approvals

Role	Name	Date	Signature
AI Director	John		☐
CEO	Alem		☐
Client			☐

Risk Register

Risk Register: [PROJECT NAME]

Last Updated: YYYY-MM-DD **Owner:** Project Manager **Review Frequency:** Weekly (sprint planning)

Risk Matrix

	Low Impact	Medium Impact	High Impact
High Prob	Medium	High	Critical
Medium Prob	Low	Medium	High
Low Prob	Low	Low	Medium

Active Risks

ID	Risk Description	Category	Probability	Impact	Score	Mitigation Strategy	Owner	Status	Date Identified
R-001		Technical / Resource / Client / External	L/M/H	L/M/H	L/M/H/C			Open / Mitigating / Closed	
R-002									

Risk Categories

- **Technical** — Technology limitations, integration issues, performance

- **Resource** — Availability, skill gaps, capacity
- **Client** — Decision delays, requirement changes, availability
- **External** — Third-party dependencies, regulatory, market
- **Financial** — Budget overrun, payment delays
- **Timeline** — Deadline pressure, estimation errors

Risk Response Strategies

- **Avoid** — Eliminate the threat by changing plan
- **Mitigate** — Reduce probability or impact
- **Transfer** — Shift to third party (insurance, outsource)
- **Accept** — Acknowledge and monitor (for low-impact risks)

Closed Risks

ID	Risk	Resolution	Date Closed

Review Log

Date	Reviewer	Changes

ZiCA Business Case

> **DEPRECATED (2026-02-14):** This is version 1.0 of the business case, written under the original "Zica" name. It has been superseded by zica-business-case-v2.md which reflects the rebrand to **Drop** and the updated product model. Retained for historical reference only.

Zica — Business Case

Date: 2026-02-08 **Version:** 1.0 **Compiled by:** John (AI Director) **Sources:** 6 AI agents (nicksaraev, product, legal, finance, marketer, security)

Executive Summary

Zica je fintech payment aplikacija — rebrand FontelePay. Ovaj dokument pokriva kompletan biznis model, product strategiju, compliance, financijski model i go-to-market plan. Sve generirano od strane AI agent tima i konsolidirano od Johna.

---

1. Business Model (Agent: nicksaraev)

Revenue Streams

Stream	Model	Procjena
Transaction fee	0.5-1% po transakciji	Primarni prihod
Premium subscription	~100 NOK/mj	Više kartica, napredna analitika, priority support
Freemium base	Besplatno	Osnovne funkcije — privlači korisnike

Target Market

- **Primarni:** B2C — krajnji korisnici (mladi profesionalci, studenti, SME vlasnici)
- **Sekundarni:** B2B — white-label za banke/startupe koji trebaju payment infrastrukturu

Go-To-Market

1. Build MVP sa core funkcijama
2. Soft launch — zatvorena beta sa 50-100 korisnika
3. Iterate bazirano na feedbacku
4. Javni launch sa marketing kampanjom
5. Partner integracije (banke, e-commerce)

---

2. Product Strategy (Agent: product)

Unique Value Proposition

"Sigurna, lokalna i prilagodljiva fintech app dizajnirana za nordijsko tržište — fokus na SME i mlade profesionalce kojima Vipps ne nudi dovoljno, a Revolut je previše generic."

Differentiators

1. **Lokalna podrška** — norveški jezik, norveški support, norveška firma
2. **SME fokus** — invoice tracking, business accounts, spending kategorije
3. **Sigurnost** — JWT httpOnly, encrypted data, GDPR-first dizajn

User Persone

Persona	Opis	Pain Point
Marko (SME vlasnik)	Mala trgovina, treba pregled finansija	Teško prati transakcije i troškove
Ana (finansijski menadžer)	Srednje preduzeće, treba business tools	Nedostaju personalizirana rješenja
Ivan (mladi profesionalac)	Želi sigurnu app za osobne finansije	Strah od nesigurnih aplikacija

Roadmap

Verzija	Features	Timeline
v1 (MVP)	Auth, accounts, transactions, cards, history	5 sedmica

Verzija	Features	Timeline
v2	Transfer novca, plaćanje računa, spending analitika	+4 sedmice
v3	Business accounts, invoice integration, partner API	+6 sedmica

Product-Market Fit signali

- Aktivni korisnici > 100 u prvih 30 dana
- Retention > 40% nakon 30 dana
- NPS > 30
- Korisnici izvršavaju > 3 transakcije sedmično

---

3. Legal & Compliance (Agent: legal)

Licence (Finanstilsynet, Norveška)

Licenca	Potrebna?	Cijena	Trajanje
E-money licence	Da (za čuvanje sredstava)	~50,000-100,000 NOK godišnje	6-12 mj za dobijanje
Payment institution	Da (za procesiranje)	~30,000-80,000 NOK godišnje	6-12 mj
BaaS alternativa	**DA — PREPORUČENO**	Ušteđuje licence	Odmah

BaaS Partner opcija (PREPORU?ENO za start)

- **Swan** — EU banking licence, IBAN accounts, SEPA transfers
- **Stripe Issuing** — kartice (virtualne + fizičke)
- **Sumsub** — KYC/AML verifikacija

Koristimo BaaS partnere → ne trebamo vlastitu licencu za start. Vlastita licenca je goal za Year 2+.

Compliance checklist

- ☐ PSD2 — transparentnost, SCA (Strong Customer Authentication)
- ☐ AML/KYC — identity verification, transaction monitoring, suspicious activity reporting

- ☐ GDPR — data minimization, consent, right to erasure, DPO assignment
- ☐ NE koristiti "bank" u marketingu bez licence

---

4. Financial Model (Agent: finance)

Startup Costs (realni za AI-first kompaniju)

Stavka	Tradicionalno	ALAI (AI-first)	Ušteda
Development	500,000 NOK	~5,000 NOK (compute)	99%
Legal/licence	50,000 NOK	50,000 NOK	0%
Marketing (launch)	100,000 NOK	100,000 NOK	0%
Infrastruktura	20,000 NOK	5,000 NOK/god	75%
Osoblje (3 osobe × 12mj)	1,800,000 NOK	0 NOK	100%
UKUPNO	**2,470,000 NOK**	**~160,000 NOK**	**93%**

Monthly Costs (post-launch)

Stavka	Iznos
BaaS partneri (Swan/Stripe)	~5,000-15,000 NOK
Hosting (Vercel Pro)	~1,000 NOK
Claude Code	~1,100 NOK
Monitoring/tools	~500 NOK
Marketing (ongoing)	~40,000-60,000 NOK
Legal (ongoing compliance)	~5,000 NOK
Mjesečni burn	**~55,000-85,000 NOK**

Revenue Projection

Period	Korisnici	MRR (NOK)	Kumulativno
Mj 1-3	50-200	5,000-15,000	Soft launch
Mj 4-6	500-1,000	30,000-60,000	Growing
Mj 7-12	2,000-5,000	100,000-250,000	Scaling

Period	Korisnici	MRR (NOK)	Kumulativno
God 2	10,000-20,000	500,000-1,000,000	Established
God 3	30,000+	1,500,000+	Mature

Break-Even

Scenarij	Break-even
Optimistički (brz rast)	Mjesec 6-8
Realistički (umjeren rast)	Mjesec 10-14
Pesimistički (spor rast)	Mjesec 18-24

LTV:CAC

Metrika	Vrijednost
CAC (customer acquisition cost)	~200 NOK
LTV (lifetime value, 24 mj)	~2,400 NOK (100 NOK/mj × 24)
LTV:CAC ratio	**12:1** (odličan, target je >3:1)

Funding

Opcija	Prednost	Nedostatak
Bootstrap	Puni control, nema dilucije	Sporiji rast
Angel investor	500K-2M NOK, brži rast	Dilucija 10-20%
Innovasjon Norge	Grants, nema dilucije	Spor proces, papirologije

Preporuka: Bootstrap MVP + apply Innovasjon Norge za marketing budget.

---

5. Marketing Strategy (Agent: marketer)

Brand Positioning

"Zica nije još jedna payment app — Zica je tvoj finansijski partner. Lokalna, sigurna, napravljena za nordijsko tržište."

Launch Plan: Soft Launch

1. **Mjesec 1-2:** Closed beta (50-100 korisnika, invite-only)
2. **Mjesec 3:** Public beta sa referral programom
3. **Mjesec 4-6:** Paid acquisition + content marketing

Kanali

Kanal	Budget	Expected CAC
Instagram/TikTok (organic)	0	Low
Google Ads	20,000 NOK/mj	~200 NOK
Content/SEO	10,000 NOK/mj	Dugoročno najjeftinije
Partnerships (e-commerce)	Per deal	Medium
Referral program	~50 NOK/referral	Najjeftinije

Marketing budget Year 1: 500,000-750,000 NOK

KPIs

KPI	Target
CAC	< 200 NOK
Conversion (signup → active)	> 20%
Retention (6 mj)	> 70%
NPS	> 30

---

6. Risk Matrix (Consolidated)

Rizik	Severity	Vjerovatnoća	Mitigacija
Regulatorne prepreke	HIGH	Medium	BaaS partneri pokrivaju licence
Vipps dominacija u Norveškoj	HIGH	High	Fokus na niše (SME, business tools)
Security breach	CRITICAL	Low	Threat model + security agent review

Rizik	Severity	Vjerovatnoća	Mitigacija
Spor user rast	MEDIUM	Medium	Referral program + content marketing
BaaS partner promijena uslova	MEDIUM	Low	Multi-provider strategija
Cash flow gap (pre break-even)	HIGH	Medium	Innovasjon Norge grant

---

7. Decision: GO / NO-GO

Za GO:

- Startup cost samo ~160K NOK (93% manje od tradicionalnog)
- LTV:CAC 12:1 (odličan)
- Break-even 10-14 mjeseci (realistično)
- BaaS partneri eliminišu licence barijeru za start
- Pipeline validacija — sistem radi

Za NO-GO:

- Vipps je dominantan u Norveškoj
- Marketing budget (500-750K NOK) je realan trošak koji ne pokriva AI
- Compliance je ongoing obaveza
- Alem je jedini human — bottleneck za biznis odluke

Preporuka

UVJETNI GO — Build MVP kao portfolio + beta test. Ako Product-Market Fit signali budu pozitivni (100+ korisnika, >40% retention), tada full commit sa marketing budgetom.

Agents koji su doprinijeli ovom dokumentu

Agent	Kompanija	Doprinos
nicksaraev	ALAI	Business model, pricing, GTM

Agent	Kompanija	Doprinos
product	ALAI Product	Product strategy, persone, roadmap
legal	ALAI Legal	Compliance, licence, regulativa
finance	ALAI Finance	Finansijski model, projections
marketer	ALAI Marketing Team	Marketing strategy, kanali, KPIs
security	ALAI Security	Threat model, hardening
dev	ALAI	Architecture, tech decisions

7 od 15 agenata aktivirano. Svi dali output.

Compiled: 2026-02-08 by John (AI Director) Awaiting: Alem (CEO) GO/NO-GO decision

Workflow

Drop — Development Workflow

Overview

Drop development follows the GOTCHA framework workflow: **Boot → Mission Control → Agent Spawn → Build → Validate → Done**

---

1. Boot (`boot.sh`)

Every session starts with:

```
bash ~/system/boot.sh
```

This verifies all 6 GOTCHA layers:

- **Goals** — specs and rules loaded
- **Orchestration** — John (AI Director) ready
- **Tools** — task.sh, mc.js, hivemind accessible
- **Context** — domain knowledge available
- **Hard prompts** — instruction templates loaded
- **Args** — behavior config applied

Boot also reads HiveMind intel.

---

2. Mission Control — Task Management

Create a task

```
node ~/system/tools/mc.js add "Feature: Send Money" --desc "Wire /send to /api/transactions"
--priority H --owner john
```

Start working

```
node ~/system/tools/mc.js start <id>
```

This creates `/tmp/mc-active-task` — required by enforcer hooks to allow Write/Edit operations.

Complete

```
node ~/system/tools/mc.js done <id> "Wired /send to API. Tests passing."
```

Other commands

```
mc.js list                    # All open tasks
mc.js list --owner john       # My tasks
mc.js pause <id>              # Pause (blocks Write/Edit)
mc.js resume <id>             # Resume paused task
mc.js block <id> "reason"     # Block with reason
mc.js show <id>               # Full details
mc.js active                  # Who's working on what
```

---

3. Agent Spawn — Builder/Validator Teams

For implementation tasks, John spawns Claude subagents:

Builder Agent

- **Role:** Implements ONE task
- **Tools:** Read, Write, Edit, Bash, Glob, Grep
- **Model:** Sonnet (never Opus for agents)
- **Config:** `~/.claude/agents/builder.md`

Spawn pattern:

```
Task tool → subagent_type: "general-purpose"
Prompt: "Implement feature X. Read the code first. Follow CLAUDE.md rules."
```

Validator Agent

- **Role:** Verifies ONE task (READ-ONLY)
- **Tools:** Read, Bash, Glob, Grep (no Write/Edit)
- **Model:** Sonnet or Haiku
- **Config:** `~/.claude/agents/validator.md`

Spawn pattern:

```
Task tool → subagent_type: "general-purpose"
Prompt: "Validate feature X. Check code quality, tests, no regressions."
```

Model Budget

Model	When
Opus	Alem session, planning — NEVER for agents
Sonnet	Builders, validators — default for agents
Haiku	Trivial tasks — file search, lint, git

---

4. Development Flow — Feature Lifecycle

```
[Pending] → mc.js start → [In Progress] → Build → Test → mc.js done → [Done]
                                ↓
                          spawn builder
                                ↓
                          spawn validator
                                ↓
                           HiveMind post
```

Feature Tracking

Drop uses a built-in feature tracking system (`src/lib/features.ts`) with 25 features across categories:

Category	Features	Status
Authentication	Registration, PIN Login, Logout, Biometric	3/4 passing
KYC	Identity Verification	1/1 passing
Banking	IBAN, Balance, Send, Receive, History, Top-up	5/6 passing
Cards	Virtual, Freeze/Unfreeze, Transactions, Physical	4/4 passing

Check feature status:

```
import { getFeatureStats, printFeatureReport } from '@/lib/features'
printFeatureReport() // ASCII status report
```

---

5. Architecture Quick Reference

Stack

- **Framework:** Next.js 16.1.6 (App Router)
- **Runtime:** React 19.2.3
- **Database:** PostgreSQL 16 (all environments) via Drizzle ORM
- **Auth:** JWT (7-day expiry) + SHA-256 PIN hashing
- **Styling:** Tailwind CSS 4

API Endpoints

Endpoint	Method	Purpose
`/api/auth/register`	POST	Register (phone + PIN)
`/api/auth/login`	POST	Login
`/api/account`	GET	Account details + balance
`/api/cards`	GET/POST	List/create cards
`/api/cards/[id]`	GET/PATCH	Card details/freeze
`/api/transactions`	GET/POST	List/send money

Endpoint	Method	Purpose
`/api/transactions/simulate`	POST	Simulate incoming (demo)
`/api/health`	GET	Health check

Service Mocks (dev mode)

Service	Provider	Mock File
BaaS	Swan	`src/lib/services/mock-swan.ts`
Cards	Stripe Issuing	`src/lib/services/mock-stripe.ts`
KYC	Sumsub	`src/lib/services/mock-sumsub.ts`

Mode controlled by `NEXT_PUBLIC_SERVICE_MODE` env var (default: "mock").

Database Schema

4 tables: `users`, `accounts`, `cards`, `transactions` Schema defined in `src/lib/db.ts`

State Management

Global context via `AppProvider` (`src/lib/context/AppContext.tsx`):

- Auth: register, login, logout
- KYC: start, submit, check
- Banking: getBalance, sendMoney, refreshTransactions
- Cards: create, freeze, getDetails
- Demo: simulateIncoming, simulatePurchase

---

6. Local Development

```
cd ~/ALAI/products/Drop/src/drop-app
npm install
npm run dev          # Dev server on localhost:3000
npm run build        # Production build
npm run test         # Jest tests (25 tests, 100% passing)
npm run lint         # ESLint
```

Test Results (Feb 7, 2026)

- 25 tests total — 100% passed
- Integration: 9/9
- Edge cases: 16/16
- Execution time: 21.2s

---

7. Deployment

- **Platform:** AWS Amplify (Next.js optimized)
- **Region:** eu-central-1 (Frankfurt)
- **Build:** Turbopack with standalone output
- **Details:** See `DEPLOYMENT.md`

---

8. Open Tasks (Drop)

Task	Priority	Owner	Description
#191	HIGH	john	Wire /send page to /api/transactions/remittance
#192	HIGH	john	Wire /scan page to /api/transactions/qr-payment
#193	HIGH	john	Wire /merchant page to real APIs
#180	MED	john	E2E test: full remittance flow
#182	MED	john	Update design and layout
#196	MED	john	Document merchant, recipients, rates feature
#198	LOW	john	Delete mock-data.ts and orphaned components

---

9. Anti-Hallucination Rules

From `~/system/rules/agent-anti-hallucination.md`:

- **TBD > Hallucination** — say "I don't know" rather than guess
- **Cross-file check** — read schema before writing code
- **No phantom deps** — only import what exists in package.json
- **Placeholder = fatalError()** — never leave silent stubs

Incident Report

Incident Report: [INC-XXX] [Short Title]

Date: YYYY-MM-DD **Severity:** P1 Critical | P2 High | P3 Medium | P4 Low **Status:** Investigating | Mitigating | Resolved | Post-Mortem Complete **Owner:** [Name/Agent] **Duration:** [Start time] — [End time] ([X hours])

---

1. Summary

[1-2 sentence description of what happened and the impact]

2. Timeline

Time	Event
HH:MM	Incident detected — [how]
HH:MM	Investigation started
HH:MM	Root cause identified
HH:MM	Fix deployed
HH:MM	Incident resolved
HH:MM	Monitoring confirmed stable

3. Impact

- **Users affected:** [Number/percentage]
- **Services affected:** [List]
- **Data loss:** [Yes/No — details]
- **Duration:** [X hours/minutes]
- **Financial impact:** [If applicable]

4. Root Cause

[What actually caused the incident. Be specific — not "human error" but "configuration file had incorrect database connection string because of merge conflict in PR #123"]

5. Resolution

[What was done to fix the issue]

1. [Step 1]
2. [Step 2]
3. [Step 3]

6. What Went Well

- [Thing that worked during incident response]
- [Thing that helped reduce impact]

7. What Went Wrong

- [Thing that contributed to the incident]
- [Thing that slowed resolution]

8. Action Items

#	Action	Owner	Due Date	Status
1	[Preventive action]			☐
2	[Process improvement]			☐
3	[Monitoring improvement]			☐

9. Lessons Learned

[Key takeaways that should inform future work]

10. Approvals

Role	Name	Date	Reviewed
Tech Lead			☐
John			☐

Hallucination Analysis (Feb 2026)

Drop — Hallucination Analiza

Datum: 2026-02-09 **Analyst:** John (AI Director) **Kontekst:** Kod nastao kao demo (nije prošao factory workflow)

Executive Summary

Backend je solidan (22 API rute, sve realne). Frontend ima 3 stranice sa mock podacima umjesto pravih API poziva. Dokumentacija zaostaje ~2 faze za kodom. Phantom integracije (Wise, Swan, Thunes, Sumsub, Stripe) navedene u docs ali nemaju ni liniju koda.

Ukupna tačnost dokumentacije: ~65%

Kriti?ni problemi

1. Tri stranice koriste lažne podatke

Stranica	Problem	Potreban fix
`/send`	Klik "Bekreft og send" samo mijenja UI step, NE poziva API	Wire `POST /api/transactions/remittance`
`/scan`	Koristi `mockMerchant` iz mock-data.ts, nema QR payment	Wire `POST /api/transactions/qr-payment`
`/merchant`	Komplet dashboard je hardcoded mock data	Wire `GET /api/merchants/dashboard` + `/api/merchants/transactions`

Impakt: Korisnik misli da je poslao novac, ali ništa se ne dešava u bazi.

2. Phantom integracije — 0 koda

Integracija	Claim u docs	Stvarnost
Wise API	"Mock → real later"	0 referenci u kodu
Swan BaaS	"Mock → real later"	0 referenci u kodu
Thunes	"Remittance provider"	0 referenci u kodu
Sumsub KYC	"Mock identity verification"	0 referenci u kodu
Stripe Issuing	"Mock card management"	0 referenci u kodu

3. Nula testova

- 0 test fajlova (izvan node_modules)
- 0 test skripti u package.json
- Test plan je prazan template
- Pipeline kaže "Validator assigned" — nikad pokrenut

Srednje ozbiljni problemi

4. Pipeline status zaostaje 2 faze

- PIPELINE.md kaže: "Design (Phase 3)"
- Stvarnost: Implementacija 70%+ gotova (20 API ruta, 10 stranica)

5. ~40% koda nedokumentirano

Ove feature nemaju ni riječ u arhitekturi ili brief-u:

- Merchant sistem (register, dashboard, QR, transactions)
- Recipients CRUD API
- Exchange rates API
- Health check endpoint
- /onboarding, /scan, /topup stranice

6. Legal compliance violation

- Odluka (ADR): "Ne koristiti riječ 'banking' bez licence"
- Stvarnost: UI tekst sadrži "bank account" na više mjesta

7. localStorage — phantom feature

- Project brief navodi "localStorage persistence" kao Must Have
- Stvarnost: 0 referenci na localStorage u kodu

Šta radi ispravno

Dependencies — ?ISTO

- 0 phantom zavisnosti
- `npm run build` prolazi
- `npx tsc --noEmit` — 0 TypeScript grešaka
- Svih 12 runtime deps postoji u node_modules

API rute — 22/22 REALNE

Svaka ruta ima:

- Prave DB operacije (ne stubovi)
- Error handling sa HTTP status kodovima
- Input validacija
- Auth/authorization provjere
- Rate limiting na javnim endpointima
- Transaction atomicity za multi-step operacije

Auth — 100% implementiran

- JWT via jose library
- httpOnly cookies (XSS zaštita)
- signToken/verifyToken/requireAuth middleware

SQLite baza — ispravna

- 6 tabela (users, recipients, merchants, transactions, exchange_rates, cards)
- Seed data funkcionalan
- Parameterized SQL (injection zaštita)

Frontend — 7/10 stranica spojeno na API

- dashboard, login, onboarding, cards, history, topup — REAL
- send (djelimično — rates i recipients sa API, ali submit je mock)

- scan, merchant — MOCK

Brojke

Kategorija	Rezultat
API rute	22/22 real (100%)
Dependencies	0 phantom, build OK
TypeScript errors	0
Stranice	7/10 real, 3/10 mock
Komponente	7 korištenih, 7 orphaned
Testovi	0
Ext. integracije	0/5 implementirano
Dokumentacija	~65% tačna

Orphaned komponente (instalirane, nikad korištene)

- alert, avatar, select, separator, sheet, skeleton, sonner

Fajlovi analizirani

- 20 API route fajlova u `src/app/api/`
- 10 page.tsx fajlova u `src/app/`
- 14 komponenti u `src/components/`
- 5 lib fajlova u `src/lib/`
- package.json, PIPELINE.md, architecture-document.md, project-brief.md, project.json, CLAUDE.md