Local Development Setup: Drop — Fintech Payment App

Local Development Setup: Drop — Fintech Payment App

> **Project:** Drop — Remittance + QR Payments **Version:** 1.0 **Date:** 2026-02-23
> **Author:** John (AI Director) **Status:** Approved **Reviewers:** Alem Bašić (CEO)

Document History

Version	Date	Author	Changes
0.1	2026-02-23	John	Initial local dev setup — Next.js 16 + SQLite + Vitest + Playwright

---

1. System Requirements

Software	Required Version	macOS	Linux	Windows (WSL2)
Node.js	`20.x`	`nvm install 20`	`nvm install 20`	Via WSL2 + nvm
npm	`10.x` (with Node 20)	Included	Included	Included
Git	`2.30+`	Pre-installed	`apt install git`	Via WSL2
flyctl	Latest	`brew install flyctl`	fly.io/docs/flyctl/installing/	WSL2
Playwright browsers	Auto-installed	`npx playwright install`	`npx playwright install`	`npx playwright install`

Hardware minimum: 8GB RAM, 10GB free disk space (SQLite DB is small; bcrypt tests are CPU-intensive) **Recommended:** 16GB RAM, SSD

Not supported: Windows without WSL2 (Next.js dev server requires Unix-compatible environment)

---

2. Quick Start (5 Steps)

```
# 1. Navigate to Drop app
cd ~/ALAI/products/Drop/src/drop-app

# 2. Install dependencies
npm install

# 3. Configure environment
cp .env.example .env.local
# Edit .env.local — see Section 5 for required values

# 4. Set up database and seed
npm run db:migrate && npm run db:seed

# 5. Start development server
npm run dev
# Open: http://localhost:3000
```

Total time (fast path): ~5 minutes

If anything fails, read the full setup in Section 3 or check Common Issues.

---

3. Detailed Setup

3.1 Node Version Manager

```
# Install nvm (macOS with Homebrew)
brew install nvm
echo 'export NVM_DIR="$HOME/.nvm"' >> ~/.zshrc
```

```
echo '[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"' >> ~/.zshrc
source ~/.zshrc

# Install Node 20 (Drop requires 20+ for Next.js 16)
nvm install 20
nvm use 20
nvm alias default 20

# Verify
node --version  # v20.x.x
npm --version   # 10.x.x
```

Auto-switching: The project root contains `.nvmrc` with `20`. If your shell is configured, it switches automatically on `cd` into the project.

3.2 Package Manager Setup

```
# Drop uses npm (not yarn or pnpm)
cd ~/ALAI/products/Drop/src/drop-app
npm install

# Verify installation
npm list --depth=0  # Should show direct dependencies
```

Lockfile: `package-lock.json` — committed to git. Run `npm install` (not `npm install --force`) to respect lockfile.

3.3 Database Setup

Drop uses SQLite for development and testing (no Docker required).

```
# Create/migrate database
cd ~/ALAI/products/Drop/src/drop-app
npm run db:migrate
# Creates: ./local.db (SQLite file)

# Verify migration
npm run db:migrate:status
# Should show all migrations applied
```

```
# Seed development data
npm run db:seed
# Inserts: test users, merchants, recipients, exchange rates
```

Verify database is set up:

```
# Check schema compliance
npm run test -- --testPathPattern=db.test.ts
# Should pass: no balance column, no card_number/cvv
```

Note: SQLite file is in `.gitignore`. Each developer has their own local DB.

3.4 Environment Variables

```
# Copy the example file
cp .env.example .env.local
```

Edit `.env.local` with your local values:

Variable	Value for Local Dev	Notes
`DATABASE_URL`	`./local.db`	SQLite file path (relative to app root)
`JWT_SECRET`	Any 32+ char random string	Use `openssl rand -base64 32`
`NEXT_PUBLIC_SERVICE_MODE`	`mock`	ALWAYS `mock` for local dev
`BCRYPT_ROUNDS`	`10`	10 locally (faster); production uses 12
`NEXT_PUBLIC_APP_URL`	`http://localhost:3000`	
`SUMSUB_API_KEY`	`mock-key`	Mocked in dev (no real API calls)
`BAAS_API_KEY`	`mock-key`	Mocked in dev (no real BaaS calls)
`RATE_LIMIT_MAX_AUTH`	`100`	Higher locally to avoid blocking dev flow

Never commit `.env.local`: It's in `.gitignore`. Use `.env.example` for sharing template values.

3.5 Playwright Browser Installation

```
# Install all Playwright browser binaries (required for E2E tests)
npx playwright install

# Or install only Chromium + WebKit (Drop E2E uses these)
```

```
npx playwright install chromium webkit
```

3.6 Seed Data

After `npm run db:seed`, these accounts exist locally:

Role	Email	Password	Notes
Consumer (Amir)	`amir@test.alai.no`	`TestDrop123!`	KYC approved, active
Merchant (Ahmet)	`ahmet@test.alai.no`	`TestDrop123!`	Merchant registered
Pending KYC	`pending@test.alai.no`	`TestDrop123!`	kyc_status = pending
Fresh user	Register via UI	—	Create during testing

---

4. Running the Application

Development Server

```
# Start Next.js dev server with hot reload
npm run dev

# Application available at:
# http://localhost:3000         — Landing page
# http://localhost:3000/login   — Login
# http://localhost:3000/api/health  — Health check (JSON)
```

What starts: Next.js dev server on port 3000 with hot reload for all routes (API + UI).

Useful Dev Commands

```
npm run dev           # Start development server
npm run build         # Build production bundle
npm run start         # Run production build locally
npm run lint          # ESLint
npm run type-check    # TypeScript type checking
npm run db:migrate    # Run pending migrations
npm run db:seed       # Seed test data
npm run db:reset      # Drop + recreate + seed (fresh state)
```

5. Running Tests

All Tests (Recommended)

```
# Run all Vitest tests (unit + integration + performance)
npm run test

# Expected: 40+ tests passing across 11 test files
```

Unit Tests

```
# Run all Vitest tests
npm run test

# Run in watch mode (re-runs on file save)
npm run test:watch

# Run specific test file
npm run test -- auth.test.ts

# Run tests matching pattern
npm run test -- --grep "bcrypt"
```

Integration Tests

```
# Integration tests use real SQLite DB (auto-created in :memory: or temp file)
npm run test -- api-endpoints.test.ts db.test.ts middleware.test.ts
```

E2E Tests (Playwright)

```
# Requires development server running
npm run dev &  # Or start in separate terminal

# Run all Playwright E2E tests
npx playwright test
```

```
# Run specific project
npx playwright test --project=user-flows
npx playwright test --project=full-flows
npx playwright test --project=input-chaos

# Run with browser visible (useful for debugging)
npx playwright test --headed

# Open Playwright UI mode
npx playwright test --ui
```

Performance Benchmarks

```
# Run api-benchmarks.test.ts
npm run test -- api-benchmarks.test.ts

# Expected outputs (NFR targets):
# bcrypt hash: < 1,000ms
# DB SELECT: < 10ms
# DB INSERT: < 20ms
# 50 concurrent rate limit checks: < 2,000ms total
```

Coverage Report

```
npm run test:coverage

# Report generated at: coverage/index.html
open coverage/index.html
```

---

6. Common Issues & Solutions

Problem	Likely Cause	Solution
`Module not found: next`	Dependencies not installed	Run `npm install`
`Database file not found`	Migrations not run	Run `npm run db:migrate`

Problem	Likely Cause	Solution
`JWT_SECRET is required`	Missing env var	Copy `.env.example` → `.env.local`; set `JWT_SECRET`
`Port 3000 already in use`	Another process	`lsof -ti:3000 | xargs kill`
`Node version mismatch`	Wrong Node version active	`nvm use 20`
bcrypt tests very slow (> 2s each)	BCRYPT_ROUNDS=12 in env	Set `BCRYPT_ROUNDS=10` in `.env.local`
Playwright tests fail: "browser not found"	Browsers not installed	`npx playwright install`
`database is locked`	Concurrent writes in SQLite test	Expected under high concurrency; run tests serially: `npm run test -- --pool=forks`
`NEXT_PUBLIC_SERVICE_MODE=live` accidentally set	Wrong env var	Always use `mock` locally; `live` requires real BaaS
TypeScript error on `any`	Strict mode	Add proper type; never use `any`

Still stuck? Ask John (AI Director) via Mission Control or #drop Slack channel.

---

7. Project Structure Reference

```
~/ALAI/products/Drop/src/drop-app/
├── src/
│   ├── app/                    # Next.js App Router
│   │   ├── api/                # 26 API routes
│   │   │   ├── auth/           # register, login, logout, me, otp, pin
│   │   │   ├── transactions/   # remittance, qr-payment, list
│   │   │   ├── rates/          # exchange rates (6 corridors)
│   │   │   ├── merchants/      # register, me
│   │   │   ├── cards/          # list, get (feature-flagged)
│   │   │   └── health/         # health check
│   │   └── (app)/              # Protected app pages
│   ├── components/             # Shared UI components
│   ├── lib/                    # Utilities, helpers, auth, db client
│   └── middleware.ts           # Rate limiting, CSRF, auth middleware
├── db/
│   ├── migrations/             # SQL migration files
│   └── seeds/                  # Seed data
├── __tests__/
│   ├── auth.test.ts            # bcrypt, JWT, password validation
```

```
│   ├── api-endpoints.test.ts    # All 26 API routes
│   ├── db.test.ts               # Schema compliance (no balance, no CVV)
│   ├── middleware.test.ts       # Rate limiting, auth middleware
│   ├── validation.test.ts       # Input validation, XSS, injection
│   ├── transactions.test.ts     # Fee calculations, transaction logic
│   ├── rates.test.ts            # Exchange rates API
│   ├── api-benchmarks.test.ts   # Performance benchmarks
│   ├── feature-flags.test.ts    # Feature flag behavior
│   ├── regression-suite.test.ts # Regression tests
│   ├── sumsub-integration.test.ts # KYC mock integration
│   ├── cards-integration.test.ts  # Cards feature (feature-flagged)
│   └── e2e/
│       ├── user-flows.spec.ts   # Registration, login journeys
│       ├── full-flows.spec.ts   # Remittance, QR payment journeys
│       └── input-chaos.spec.ts  # 20+ edge case inputs
├── .env.example                 # Environment variable template
├── vitest.config.ts             # Vitest configuration
├── playwright.config.ts         # Playwright configuration (3 projects)
├── next.config.ts               # Next.js configuration
└── package.json                 # Scripts and dependencies
```

---

8. IDE Configuration

VS Code Settings (`.vscode/settings.json`)

```
{
  "editor.formatOnSave": true,
  "editor.defaultFormatter": "esbenp.prettier-vscode",
  "editor.codeActionsOnSave": {
    "source.fixAll.eslint": true
  },
  "typescript.tsdk": "node_modules/typescript/lib",
  "typescript.enablePromptUseWorkspaceTsdk": true
}
```

Recommended Extensions

Extension	ID	Purpose
ESLint	`dbaeumer.vscode-eslint`	Inline linting
Prettier	`esbenp.prettier-vscode`	Format on save
Tailwind CSS IntelliSense	`bradlc.vscode-tailwindcss`	Autocomplete for Tailwind
Playwright Test	`ms-playwright.playwright`	Run/debug Playwright tests
Vitest	`vitest.explorer`	Run/debug Vitest tests
TypeScript Error Lens	`usernamehw.errorlens`	Inline TypeScript errors

Related Documents

- Developer Onboarding Guide
- Coding Standards
- Testing Guide

Approval

Role	Name	Date	Signature
Author	John (AI Director)	2026-02-23	Approved (AI)
Tech Lead	John	2026-02-23	Approved
CEO (Alem)	Alem Bašić	TBD	

Revision #5
Created 2026-02-23 12:08:42 UTC by John
Updated 2026-05-31 20:03:38 UTC by John