Feature Baseline

Source: `~/ALAI/products/Plock/docs/demo-readiness/02-feature-baseline.md`

PLOCK — Feature Baseline

Date: 2026-03-14
Purpose: Establish a first canonical feature baseline for PLOCK using real repository evidence and authoritative documentation. This is the operational feature map that future QA, backlog slicing, implementation planning, and demo prep should build on.

---

1. Method

This baseline is based on:

- actual repository structure
- backend route/service modules
- frontend micro-frontend structure
- database migrations
- runbooks, security, and integration docs
- PRD / GTM / regulatory context where relevant

Every feature is tagged with a maturity label:

- **exists** — strongly supported by code + docs
- **partial** — real evidence exists, but depth/completeness is unclear
- **planned** — documented as intended direction, but implementation evidence is weak or phase-dependent

---

2. Feature Groups

1. Platform / architecture
2. Backend domain features
3. Frontend / UX features
4. Database / tenant isolation

5. Integrations
6. AI features
7. Testing / QA
8. Ops / deployment / observability
9. Security
10. Business / GTM / support
11. Known contradictions and gaps

---

3. Platform / Architecture

F-PLAT-001 — Kotlin/Ktor backend platform

Status: exists
Evidence: `backend/build.gradle.kts`, `Application.kt`, Ktor plugins and route modules

PLOCK uses a Kotlin/Ktor backend, not an Express/Node backend.

F-PLAT-002 — React micro-frontend frontend platform

Status: exists
Evidence: `frontend/`, `frontend/shell/package.json`, MFE folders, Vite setup

PLOCK frontend is a React 19 + TypeScript + Vite Module Federation system with multiple MFEs.

F-PLAT-003 — Monorepo with mixed backend/frontend tooling

Status: exists
Evidence: root `package.json`, `backend/`, `frontend/`, workspaces, Turborepo scripts

PLOCK uses:

- Gradle for backend
- npm/Turborepo workspace structure for frontend packages and MFEs

F-PLAT-004 — Railway-oriented deployment model

Status: partial-to-exists
Evidence: `infrastructure/README.md`, Terraform files, Railway env references

The deployment model is strongly Railway-oriented, although production readiness should still be independently verified.

---

4. Backend Domain Features

F-BE-001 — Authentication

Status: exists
Evidence: `AuthRoutes.kt`, `Authentication.kt`, JWT config in `application.yaml`

F-BE-002 — Warehouse management

Status: exists
Evidence: `WarehouseRoutes.kt`, warehouse references in API/docs, migrations

F-BE-003 — Location/bin management

Status: exists
Evidence: `LocationRoutes.kt`, `LocationService.kt`, locations schema

F-BE-004 — Product management

Status: exists
Evidence: `ProductRoutes.kt`, `products` table, DTOs/services

F-BE-005 — Inventory management

Status: exists
Evidence: `InventoryRoutes.kt`, `InventoryService.kt`, inventory schema, tests

F-BE-006 — Order management

Status: exists
Evidence: `OrderRoutes.kt`, `OrderService.kt`, order schema

F-BE-007 — Picking workflow

Status: exists
Evidence: `PickingRoutes.kt`, `PickingService.kt`, `mfe-picking`, tests

F-BE-008 — Receiving workflow

Status: exists
Evidence: `ReceivingRoutes.kt`, `ReceivingService.kt`, receiving tables/migrations, discrepancy test

F-BE-009 — Shipment management

Status: exists
Evidence: `ShipmentRoutes.kt`, `ShipmentService.kt`, shipment schema, tests

F-BE-010 — Carrier management

Status: exists
Evidence: `CarrierRoutes.kt`, `CarrierService.kt`, `carriers` schema

F-BE-011 — Carrier integration endpoints

Status: exists
Evidence: `CarrierIntegrationRoutes.kt`, integration clients and services

F-BE-012 — Cycle counts

Status: exists
Evidence: `CycleCountRoutes.kt`, `CycleCountService.kt`, tests, schema

F-BE-013 — Stock movements / audit trail for inventory changes

Status: exists
Evidence: `StockMovementRoutes.kt`, `stock_movements` table, schema

F-BE-014 — Returns workflow

Status: exists
Evidence: `ReturnRoutes.kt`, `ReturnService.kt`, return schema

F-BE-015 — Supplier management

Status: exists
Evidence: `SupplierRoutes.kt`, `SupplierService.kt`, supplier schema

F-BE-016 — Zone management

Status: exists
Evidence: `ZoneRoutes.kt`, `ZoneService.kt`, zones schema

F-BE-017 — User and role management

Status: exists
Evidence: `UserRoutes.kt`, `RoleRoutes.kt`, `RbacService.kt`

F-BE-018 — Dashboard endpoints

Status: exists
Evidence: `DashboardRoutes.kt`

F-BE-019 — Barcode support

Status: exists
Evidence: `BarcodeRoutes.kt`, `BarcodeService.kt`, ZXing dependencies

F-BE-020 — Audit endpoints

Status: exists
Evidence: `AuditRoutes.kt`, `AuditService.kt`, audit schema

F-BE-021 — SSE / real-time update infrastructure

Status: exists
Evidence: `SseRoutes.kt`, `SseService.kt`, Ktor SSE dependency

F-BE-022 — Webhooks

Status: exists
Evidence: `WebhookRoutes.kt`, `WebhookService.kt`, webhook tables

F-BE-023 — Offline/mobile sync support

Status: partial
Evidence: `OfflineSyncRoutes.kt`, `OfflineSyncService.kt`, `V7__mobile_pwa_support.sql`

F-BE-024 — Health endpoints

Status: exists
Evidence: `HealthRoutes.kt`, runbooks

---

5. Frontend / UX Features

F-FE-001 — Shell application

Status: exists
Evidence: `frontend/shell`

F-FE-002 — Inventory MFE

Status: exists
Evidence: `frontend/mfe-inventory`

F-FE-003 — Orders MFE

Status: exists
Evidence: `frontend/mfe-orders`

F-FE-004 — Picking MFE

Status: exists
Evidence: `frontend/mfe-picking`

F-FE-005 — Settings MFE

Status: exists
Evidence: `frontend/mfe-settings`

F-FE-006 — AI MFE

Status: partial-to-exists
Evidence: `frontend/mfe-ai`, AI strategy docs

F-FE-007 — Shared shell + MFE architecture

Status: exists
Evidence: frontend repo structure, module federation dependency, blueprint/docs

---

6. Database / Tenant Isolation Features

F-DB-001 — PostgreSQL primary database

Status: exists
Evidence: build config, env docs, migrations, runbooks

F-DB-002 — Flyway-managed schema

Status: exists
Evidence: Flyway dependency, migrations, ADR references

F-DB-003 — RLS tenant isolation by `warehouse_id`

Status: exists
Evidence: `RLS-AUDIT.md`, V6 migration, TenantPlugin docs

F-DB-004 — Warehouse-scoped data model

Status: exists
Evidence: RLS docs, warehouse-based route and domain model

F-DB-005 — Encrypted Fortnox token storage

Status: exists
Evidence: `FortnoxTokens.kt`, `CryptoService.kt`, docs

F-DB-006 — Mobile/offline support tables

Status: partial
Evidence: `V7__mobile_pwa_support.sql`

---

7. Integration Features

F-INT-001 — Fortnox OAuth

Status: exists
Evidence: `FortnoxOAuthRoutes.kt`, secrets docs

F-INT-002 — Fortnox sync flow

Status: exists
Evidence: `FortnoxSyncRoutes.kt`, `FortnoxSyncService.kt`, runbook incident section

F-INT-003 — PostNord integration

Status: exists
Evidence: PostNord client/service files, docs, secrets config

F-INT-004 — DHL integration

Status: exists
Evidence: DHL client/service files, docs, secrets config

F-INT-005 — Instabee integration

Status: exists
Evidence: Instabee client/service files, docs, secrets config

F-INT-006 — Integration sync logging

Status: exists
Evidence: `integration_sync_log` schema, runbook references

---

8. AI Features

F-AI-001 — AI Chat positioning and product surface

Status: partial
Evidence: `AI-STRATEGY.md`, PRD, `mfe-ai`

F-AI-002 — Smart Picking / route optimization

Status: partial
Evidence: PRD, AI strategy, runbook mentions

F-AI-003 — Natural language warehouse querying

Status: planned-to-partial
Evidence: PRD + AI strategy

---

9. Testing / QA Features

F-QA-001 — Backend automated tests

Status: exists
Evidence: `backend/src/test/kotlin/*`

F-QA-002 — E2E tests

Status: exists
Evidence: `e2e/app.spec.ts`, `e2e/inventory.spec.ts`

F-QA-003 — Performance testing structure

Status: partial
Evidence: `tests/performance/`, `infrastructure/k6/`

F-QA-004 — Regression test structure

Status: partial
Evidence: `tests/regression/`

---

10. Ops / Deployment / Observability Features

F-OPS-001 — Local docker-based dev environment

Status: exists
Evidence: `docker-compose.yml`, runbooks

F-OPS-002 — Deployment scripts / Terraform infra

Status: exists
Evidence: `infrastructure/README.md`, `main.tf`, scripts

F-OPS-003 — Runbooks for incident handling

Status: exists
Evidence: `RUNBOOK.md`, `docs/RUNBOOK.md`

F-OPS-004 — Sentry-based monitoring

Status: partial-to-exists
Evidence: backend and frontend Sentry references, runbooks, secrets docs

---

11. Security Features

F-SEC-001 — JWT auth

Status: exists
Evidence: auth plugin, `application.yaml`, docs

F-SEC-002 — BCrypt password hashing

Status: exists
Evidence: backend dependencies, security docs

F-SEC-003 — AES-256-GCM token encryption

Status: exists
Evidence: `CryptoService.kt`, Fortnox token docs

F-SEC-004 — PostgreSQL RLS enforcement

Status: exists
Evidence: RLS migration and audit docs

F-SEC-005 — Role-based access control

Status: exists
Evidence: role/user routes and services

F-SEC-006 — Secret management model

Status: exists but needs cleanup
Evidence: `docs/SECRETS-MANAGEMENT.md`, infra docs

---

12. Business / GTM / Commercial Features

F-BIZ-001 — Pricing model

Status: exists in docs
Evidence: `README.md`, `PRD.md`, `GTM-STRATEGY.md`

F-BIZ-002 — Fortnox ecosystem GTM strategy

Status: exists in docs
Evidence: `GTM-STRATEGY.md`

F-BIZ-003 — Swedish SMB WMS positioning

Status: exists in docs
Evidence: README, PRD, GTM

F-BIZ-004 — 3PL commercial direction

Status: planned
Evidence: PRD, GTM, regulatory references

F-BIZ-005 — Regulatory/compliance framing

Status: exists in docs
Evidence: `REGULATORY.md`

---

13. Known Contradictions / Risks

RISK-001 — Stale architecture document

`docs/ARCHITECTURE.md` conflicts with actual repo reality and must not be used as canonical architecture input.

RISK-002 — API spec likely drifted

`docs/API-SPEC.md` appears to reflect an org-centric model and may not be fully synchronized with current Ktor routing and warehouse-based tenancy.

RISK-003 — Runbook drift

There are conflicting operational instructions around:

- npm vs pnpm
- startup flow
- JDK/runtime expectations

RISK-004 — Secrets hygiene problem

`infrastructure/README.md` contains secret-like Terraform environment variable examples that should be reviewed immediately.

RISK-005 — Missing canonical demo-readiness package

The intended documentation package under `docs/demo-readiness/` does not exist yet.

---

14. Summary by Maturity

Exists

- Kotlin/Ktor backend
- React MFE frontend
- Flyway + PostgreSQL
- RLS tenant isolation
- receiving / inventory / orders / picking / shipments
- carrier integrations
- RBAC
- dashboard/backend operational domains
- barcode
- audit/webhooks/SSE
- backend tests
- e2e presence

- pricing/GTM/business framing

Partial

- AI execution depth
- smart picking maturity verification
- mobile/offline maturity
- shipment monitoring UX depth
- reorder alerting maturity
- ops/monitoring completeness
- performance/regression readiness
- API spec accuracy

Planned

- 3PL billing and mature 3PL workflows
- polished demo-readiness package
- full support/onboarding package
- fully canonical feature and QA documentation set

---

15. Working Rule

Future planning and QA work must use this feature baseline together with:

- canonical source-of-truth map
- user stories baseline

Nothing should be marked “implemented” solely because it appears in a stale document or in an agent-generated summary without artifact proof.

Revision #3
Created 2026-03-14 12:03:43 UTC by John
Updated 2026-05-31 20:05:17 UTC by John