API Spec Verification Pass

Source: `~/ALAI/products/Plock/docs/demo-readiness/06-api-spec-verification-pass.md`

PLOCK — API Spec Verification Pass

Date: 2026-03-14
Reviewer: John
Scope: Narrow manual verification of `docs/API-SPEC.md` against the real Ktor route surface under `backend/src/main/kotlin/no/alai/plock/`.

---

1. Verdict

Canonical verdict: FAIL as a canonical API source
Reference verdict: PASS WITH NOTES as API-intent/reference material

Why

`docs/API-SPEC.md` contains useful product intent, but it does not currently match the real backend route surface closely enough to be treated as canonical.

The biggest problems are:

- org-centric model appears where repo reality is warehouse-centric
- several documented endpoint groups do not exist in the current Ktor app
- several real Ktor route groups are missing or materially different in the spec
- multiple path shapes differ even when the domain concept is roughly aligned

---

2. Verification Sources

Primary sources used for this pass:

- `backend/src/main/kotlin/no/alai/plock/plugins/Routing.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/AuthRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/WarehouseRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/LocationRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/ProductRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/InventoryRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/OrderRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/PickingRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/ReceivingRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/CarrierRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/CarrierIntegrationRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/ShipmentRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/RoleRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/UserRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/DashboardRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/routes/SseRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/integrations/fortnox/FortnoxOAuthRoutes.kt`
- `backend/src/main/kotlin/no/alai/plock/integrations/fortnox/FortnoxSyncRoutes.kt`

---

3. Real Route Surface Snapshot

`Routing.kt` mounts these main groups under `/api/v1` (plus unauthenticated Fortnox OAuth + health/auth):

- `/auth` and `/auth/me`
- `/products`
- `/users`
- `/dashboard`
- `/warehouses`
- `/locations`
- `/inventory`
- `/orders`
- `/picking`
- `/zones`
- `/suppliers`
- `/receiving`
- `/carriers`
- `/shipments`
- `/roles`
- `/audit`
- `/stock-movements`
- `/cycle-counts`
- `/returns`

- `/webhooks`
- `/events*`
- `/barcodes`
- `/integrations/fortnox/*`

This is already enough to show that the live API surface is broader in some places and materially different in others than `docs/API-SPEC.md` suggests.

---

4. Sections That Are Roughly Aligned

These areas are directionally aligned, even when path or payload details differ:

4.1 Auth

`docs/API-SPEC.md` documents auth/login concepts, and real routes do include:

- `POST /auth/register`
- `POST /auth/login`
- `GET /auth/me`

4.2 Warehouses / products / users / dashboard

These domains exist in the real backend and in the API spec at a high level.

4.3 Carrier + shipment integrations

The spec correctly signals that carrier and Fortnox integration domains exist, although the actual route design is different.

4.4 Real-time events

The spec includes SSE/event streaming, and the real backend does expose:

- `GET /events/stream`
- `GET /events` (legacy alias)
- `GET /events/connections`
- `POST /events/test`

---

5. Major Drift / Mismatch Areas

5.1 Tenancy model drift

Spec problem: org-centric model
Repo reality: warehouse-centric model

Examples in `docs/API-SPEC.md`:

- `Organizations`
- `organization_name`
- `slug`
- organization object inside auth responses
- `/organizations/me`

Repo reality:

- real tenant model is `warehouse_id`
- auth/user DTOs expose `warehouseId`
- no `organization` route group exists in the Ktor app

Impact: High. This is not a cosmetic difference; it changes the domain model.

5.2 Auth flow drift

Spec says

- `POST /auth/register` creates organization + admin user
- `POST /auth/refresh` exists
- `POST /auth/logout` exists

Actual routes

- `POST /auth/register`
- `POST /auth/login`
- `GET /auth/me`

Mismatch

Real auth routes do **not** currently expose documented refresh/logout endpoints, and register/login response shapes differ from the org-centric API spec.

Verdict: partial alignment only

5.3 Warehouses and locations path drift

Spec says

- `/warehouses/:warehouseId/zones`
- `/warehouses/:warehouseId/bins`
- nested warehouse-scoped bin endpoints

Actual routes

- `/warehouses`
- `/warehouses/{id}`
- `/locations`
- `/locations/{id}`
- `/zones`
- `/zones/{id}`
- `/put-away/suggest`

Mismatch

The real backend models zones and locations as first-class route groups, not nested warehouse bin endpoints as described in the spec.

5.4 Inventory route drift

Spec says

- `GET /inventory`
- `POST /inventory/adjustments`
- `GET /inventory/transactions`
- `POST /inventory/cycle-count`
- `GET /inventory/anomalies`

Actual routes

- `GET /inventory`
- `GET /inventory/balance`
- `GET /inventory/summary`
- `GET /inventory/search`
- `POST /inventory`
- `POST /inventory/recalculate`
- `GET /inventory/{id}`
- `POST /inventory/{id}/adjust`
- `DELETE /inventory/{id}`

Mismatch

The real system exposes different operational endpoints and splits stock adjustment and audit trail differently. Inventory transaction history is represented more directly via `/stock-movements`, not the spec's `/inventory/transactions` route.

5.5 Receiving / inbound drift

Spec says

- `/inbound/purchase-orders`
- `/inbound/purchase-orders/:poId/receive`

Actual routes

- `/receiving`
- `/receiving/{id}`
- `/receiving/{id}/receive-item`
- `/receiving/{id}/complete`
- `/receiving/{id}/cancel`
- `/receiving/{id}/process`
- `/receiving/{id}/discrepancies`
- `/receiving/discrepancies/{discrepancyId}/resolve`

Mismatch

The real backend has a receiving-order workflow, not the purchase-order route structure described in the spec.

5.6 Orders / outbound drift

Spec says

- `/outbound/orders`
- `/outbound/orders/release`
- `/outbound/orders/:orderId/status`

Actual routes

- `/orders`
- `/orders/{id}`
- `/orders/{id}/allocate`
- `/orders/{id}/deallocate`
- `/orders/{id}/ship`

Mismatch

The domain overlaps, but path structure and supported actions differ materially.

5.7 Picking drift

Spec says

- `/picking/routes`
- `/picking/routes/:routeId/confirm-pick`
- `/picking/routes/:routeId/assign`

Actual routes

- `/picking/waves`
- `/picking/waves/{id}`
- `/picking/waves/{id}/start`
- `/picking/waves/{id}/pick-item`
- `/picking/{pickListId}/items/{itemId}/confirm`

Mismatch

The live system is pick-wave/pick-item oriented, not route-assignment oriented in the way the spec describes.

5.8 Carrier / shipment drift

Spec says

- `/carriers/labels/generate`
- `/carriers/labels/:trackingNumber/status`
- `/carriers/services`
- `/carriers/labels/:trackingNumber/void`

Actual routes include

- carrier CRUD under `/carriers`
- shipment CRUD and lifecycle under `/shipments`
- provider-specific endpoints under `/carriers/postnord/*`, `/carriers/dhl/*`, `/carriers/instabee/*`
- generic integration shipment routes under `/shipments` in `CarrierIntegrationRoutes.kt`

Mismatch

The spec compresses real shipping/carrier behavior into a simplified label-oriented interface that does not match the current route surface.

5.9 Dashboard / reports drift

Spec says

- `/dashboard/stats`
- `/reports/inventory-health`
- `/reports/performance`
- `/reports/export`

Actual routes

- `/dashboard/stats`
- `/inventory/low-stock`

Mismatch

Only `dashboard/stats` is clearly aligned. The `/reports/*` route group is not present in the current Ktor app.

5.10 AI route drift

Spec says

- `/ai/chat`
- `/ai/actions/confirm`
- `/ai/chat/history`
- `/ai/pick-route/optimize`
- `/ai/pick-route/job/:jobId`

Actual routes

- no `/ai/*` route group was found in the current backend route surface

Mismatch

This is major product-intent content, not current API reality.

5.11 Integrations drift

Spec says

- `GET /integrations`
- `GET /integrations/fortnox/connect`
- `GET /integrations/fortnox/callback`
- `POST /integrations/fortnox/sync`

- `DELETE /integrations/:provider`
- webhooks under integrations

Actual routes include

- `GET /integrations/fortnox/auth`
- `GET /integrations/fortnox/callback`
- `GET /integrations/fortnox/status`
- `POST /integrations/fortnox/sync/articles`
- `POST /integrations/fortnox/sync/orders`
- `POST /integrations/fortnox/sync/stock`
- `POST /integrations/fortnox/sync/products`
- `POST /integrations/fortnox/confirm-shipment/{orderId}`

Mismatch

The real Fortnox integration surface is more concrete and warehouse-scoped than the generic integration control plane described in the spec.

5.12 Users / RBAC drift

Spec says

- `POST /users/invite`
- `PATCH /users/:userId`
- `GET /users/me`
- mostly flat user-role assumptions

Actual routes include

- `/users`
- `/users/{id}` with `GET`, `PUT`, `DELETE`
- `/roles`
- `/roles/assign`
- `/roles/user/{userId}`
- `/roles/user/{userId}/permissions`
- `GET /auth/me`

Mismatch

RBAC is present, but it is organized differently and more explicitly in the real backend than in the spec.

---

6. Important Real Route Groups Missing from the Spec

These live route groups exist in the backend but are not represented well, or at all, in `docs/API-SPEC.md`:

- `/audit`
- `/stock-movements`
- `/cycle-counts`
- `/returns`
- `/webhooks`
- `/barcodes`
- `/suppliers`
- `/zones`
- `/put-away/suggest`
- `/events/connections`
- `/events/test`
- `/health` and `/health/ready`

---

7. Practical Conclusion

Use `docs/API-SPEC.md` only for:

- product/API intent
- domain discovery hints
- identifying candidate workflows to verify against code

Do **not** use it for:

- canonical route inventory
- API QA signoff
- implementation planning without code check
- tenant-model assumptions

---

8. Recommended Next Narrow Step

Do **not** rewrite the full API spec yet.

Instead, use this sequence:

1. keep the current warning on `docs/API-SPEC.md`
2. use this verification note as the current interpretation layer
3. later perform a targeted API-spec correction pass section-by-section, starting with:
   - tenancy/auth
   - inventory
   - receiving/orders/picking
   - carriers/integrations

---

9. Decision

`docs/API-SPEC.md` is now a **verified non-canonical reference**.

Until corrected, the canonical API truth for PLOCK remains:

- the actual Ktor route files, and
- the canonical demo-readiness baseline package.

Revision #3
Created 2026-03-14 12:03:45 UTC by John
Updated 2026-05-31 20:05:21 UTC by John