ADR-008: Hono API Framework

ADR-008: Hono v4 for Mobile API

Status: Accepted **Date:** 2026-02-21 **Deciders:** John (AI Director) **Category:** Backend

---

Context

Drop has two client platforms with different API needs:

Platform	Auth Pattern	Token Storage	API Style	Deployment
Web (Next.js)	Cookie-based JWT via BFF	httpOnly cookie	Next.js API Routes (collocated)	Vercel / App Runner
Mobile (Expo)	Bearer token	AsyncStorage	REST API (separate process)	App Runner

Next.js API Routes work well for the web BFF pattern (server-side rendering + API in one deployment), but mobile needs a lightweight, standalone REST API with Bearer token authentication and mobile-specific concerns (deep link callbacks, longer token lifetimes).

Frameworks considered for the mobile API:

Framework	Performance	TypeScript	Edge Compatible	Bundle Size	Ecosystem
Hono v4	Excellent (minimal overhead)	First-class	Yes (Workers, Deno, Bun)	~14KB	Growing fast
Express 5	Good (mature)	Requires @types	No (Node-only)	~200KB	Massive
Fastify 5	Excellent (schema validation)	Good (built-in types)	No (Node-only)	~300KB	Large
Elysia	Excellent (Bun-native)	First-class	Bun only	~20KB	Small

Hono was selected for its TypeScript-first design, minimal overhead, and edge compatibility. The mobile API runs as a separate Hono server on App Runner alongside the Next.js web app.

Decision

Use Hono v4 for the mobile REST API. Keep Next.js API Routes for the web BFF.

```
graph TB
    subgraph clients["Client Platforms"]
        web["Web Browser<br/>(Next.js SSR + CSR)"]
        mobile["Mobile App<br/>(Expo SDK 54)"]
    end

    subgraph backend["Backend Services"]
        nextjs_api["Next.js BFF<br/>API Routes (/api/*)<br/>Cookie auth, SSR"]
        hono_api["Hono v4 API<br/>REST (/v1/*)<br/>Bearer auth, mobile-optimized"]
    end

    subgraph shared["Shared Layer"]
        db["Database Access (db.ts)"]
        bankid["BankID OIDC (bankid.ts)"]
        validation["Validation (validation.ts)"]
    end

    web --> nextjs_api
    mobile --> hono_api
    nextjs_api --> db
    nextjs_api --> bankid
    hono_api --> db
    hono_api --> bankid
    nextjs_api --> validation
    hono_api --> validation
```

Both APIs share the same database access layer, BankID integration, and validation utilities. The difference is in auth pattern and deployment:

Aspect	Next.js API Routes	Hono v4 API
Base path	`/api/`	`/v1/`
Auth	Cookie JWT (httpOnly)	Bearer token (Authorization header)

Aspect	Next.js API Routes	Hono v4 API
Token lifetime	7 days	7 days
BankID callback	HTTP redirect to `/dashboard`	JSON response with token
Rate limiting	SQLite-backed (persistent)	Database-backed (SQLite `rate_limits` table, persistent)
Deployment	Vercel or App Runner	App Runner (standalone)

Consequences

Positive

- Mobile API is lightweight and fast (Hono ~14KB, minimal middleware overhead)
- TypeScript-first with excellent type inference for request/response
- Edge-compatible runtime means future flexibility (Cloudflare Workers, Deno Deploy)
- Clear separation between web BFF (cookie auth) and mobile API (Bearer auth)
- Shared business logic prevents code duplication

Negative

- Two API servers to maintain (Next.js + Hono)
- Two deployment targets on App Runner
- Shared library updates must be tested against both frameworks
- Smaller ecosystem compared to Express (fewer middleware packages)

Risks

- **Diverging behavior:** Same endpoint implemented twice may behave differently. Mitigation: shared database access layer and validation utilities ensure consistent business logic.
- **Hono ecosystem maturity:** Hono is newer than Express/Fastify. Mitigation: Hono v4 is stable and backed by Cloudflare; core routing and middleware are well-tested.

References

- Container Diagram (C4 Level 2) -- Shows both API containers
- Authentication System -- Web vs mobile auth flows
- API Reference -- Next.js API endpoints
- ADR-005: Monolith First -- Overall architecture approach

- Hono v4 documentation: hono.dev

Revision #5
Created 2026-02-21 05:59:00 UTC by John
Updated 2026-05-23 10:56:46 UTC by John