ENGINEERING DEEP DIVE

Building an Enterprise Dashboard with AI

From Architecture to Production

User Management, Authentication, and RBAC - Live in Production

John Adib | 18 January 2026

What We'll Cover

01
The Challenge
Why we needed a new approach

02
AI-Powered Development
Tools and workflow with Cursor + Claude

03
Architecture Decisions
72 documented decisions in 3 days

04
Tech Stack Deep Dive
Next.js, Hono, Better Auth, Drizzle

05
Authentication System
Passkeys, Magic Links, RBAC

06
Key Takeaways
Lessons learned and what's next

THE CHALLENGE

Enterprise Panel from Scratch

Requirements

- Full user management panel
- Multiple auth methods
- Role-based access control
- Enterprise-grade security

Strong Governance

- Biome with 140+ rules on top of recommended
- Strict TypeScript config
- 72 Decision Records

Traditionally: months

Built in 3 days

What We Built

- Monorepo with apps + packages
- Web app with Next.js 16 + React 19
- Responsive + dark/light mode
- Passkey + Magic Link + Google auth
- RBAC with scopes
- Identity service with Hono
- Deployed to Cloudflare Workers + D1

Live in Production & Tested by Users

Development Workflow

Phase 1: Strategic Decisions (Web Chat)

Me	Claude	ChatGPT	Gemini
Moderator	Web	Web	Left early

Shared ideas between AIs, added my opinion at each step, reached consensus on core decisions

Phase 2: Document

Claude & ChatGPT summarize decisions → Markdown files → Decision Records

Phase 3: Execute

Claude Code | Cursor

Phase 4: Local QA

Manual testing, review code changes, verify functionality before PR

Phase 5: PR Review

CodeRabbit | AI review

Teaching AI Your Standards

.cursorrules

Pure code rules for Cursor IDE

CLAUDE.md

References cursorrules + project context. Enable "Include .claude files" in Cursor settings!

Same Rules Everywhere

Cursor or Claude Code - both follow identical standards. However, sometimes you need to ask nicely to double-check!

Priority #1: Separation of Concerns

- Single responsibility per file
- Clear module boundaries
- Isolated side effects
- Composable functions

Key Rules

- Functional programming only
- No comments in code
- Explicit over implicit
- Small, focused functions

Result: AI generates consistent, maintainable code from day one

Documented Decisions

Why Document?

- Future devs understand "why"
- Prevents re-debating
- AI can reference them
- Faster onboarding

Format: MADR
/docs/decisions/

72 DRs

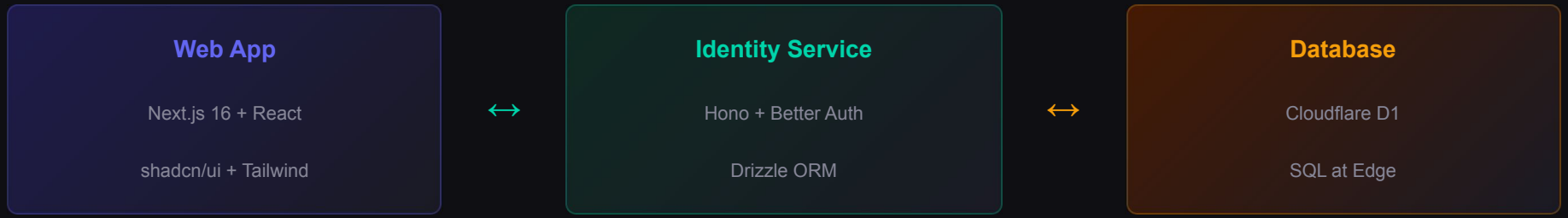
12 Categories

Infrastructure Cloudflare, D1, R2	**Frontend** Next.js, React, shadcn	**Backend** Hono, Auth, Drizzle	**Automation** GitHub Actions, CI
Packages UI, Auth, Logger	**Patterns** Naming, Code style	**Tooling** pnpm, Turborepo	**TypeScript** Strict config, ESM
Code Quality Biome, Lint, Tests	**Formatting** Biome rules	**Services** Identity, Email	**Naming** Conventions

`DR-0001-monorepo.md → DR-0072-zod.md`

TECH STACK

System Architecture



Deployed on	Storage	Monitoring	Reliability
Cloudflare Workers	**Cloudflare R2 & CDN**	**Sentry + Uptimerobot**	**Vitest + Storybook**

TECH STACK

Monorepo with Turborepo

Monorepo

apps/

web/, identity/

packages/

ui/, design-system/, auth/

config/, logger/, utils/

email/, storage/, sdk/

typescript-config/, payment/

pnpm

Fast package manager with workspaces

Turborepo

Caching, parallel builds

Why Monorepo?

- Atomic commits across packages
- Shared code immediately available
- Single CI/CD pipeline

One Command Runs All

pnpm lint + tsc in seconds with Turbo cache

Frontend Framework - Next.js

Why Next.js over Vite + React?

- SSR for speed
- Server Actions for mutations
- React Server Components
- Industry standard

Speed Matters

Server-side rendering + Server Functions = fast initial load + dynamic data without client bloat

Alternatives: Vite + React, TanStack Start, Remix

UI Stack

React 19 | Tailwind CSS | shadcn/ui | Motion

Responsive | Dark/Light Mode

Key Patterns

- Route groups for organization
- Server Actions for mutations
- Parallel routes for modals
- Middleware for auth guards

Backend Framework - Hono

What is Hono?

Hono - means flame🔥 in Japanese - is a small, and ultrafast web framework built on Web Standards. It works on any JavaScript runtime. Express-like API with excellent TypeScript support with about 10M weekly install.

~14KB

Bundle size

Edge

Most famous and designed for

Strong Industry Support

Growing adoption, active community, backed by Cloudflare

Runs Everywhere

Cloudflare Workers | AWS Lambda | GCP Cloud Run | Vercel Edge | Node.js | Bun

Built-in Features

- CORS, JWT, validation middleware
- TypeScript inference
- OpenAPI generation

Alternatives: Elysia, itty-router, Express

TECH STACK - INFRASTRUCTURE

Cloudflare Platform

Why Cloudflare?

- Already used in the Company
- No new vendor
- Generous free tier
- Powers 20% of internet
- Global edge network

Alternatives: GCP/AWS need more setup, access approval, couple hundred $/month vs free on Cloudflare

Workers

Serverless compute at edge. Runs Next.js via OpenNext adapter.

D1 Database

Managed SQLite at edge. Works with Drizzle ORM.

R2 Storage

S3-compatible object storage. Zero egress fees.

CI triggers automatically on push - no build pipeline setup needed

AUTHENTICATION SYSTEM

Better Auth - Self-Hosted Solution

Why Better Auth?

- Best self-hosted auth library
- Excellent TypeScript inference
- Framework-agnostic
- Active maintenance
- Plugin architecture

Rejected alternatives:

Clerk (vendor lock-in, per-user pricing)

Auth.js (maintenance issues)

Lucia (deprecated)

Plugins Enabled

Passkey
WebAuthn/FIDO2

Magic Link
Email login

Google
OAuth2

Admin
User mgmt

Multi-Session
Device mgmt

API Key
For MCP servers

Enterprise Benefits

Self-hosted = data stays in our DB, no per-user costs, no vendor lock-in

AUTHENTICATION SYSTEM

Passkey Authentication

What are Passkeys?

WebAuthn/FIDO2 standard for passwordless authentication. Credentials never leave the device.

How It Works

- User logs in via social/email first time
- Prompted to register passkey
- Passkey stored on device securely
- Future logins use passkey directly

Alternatives rejected: TOTP (UX issues), SMS 2FA (security vulnerabilities)

Supported Platforms

Face ID
Touch ID

Windows
Hello

Android
Biometrics

Security Benefits

- Phishing-resistant
- No shared secrets
- Biometric verification
- Industry standard (FIDO2)

AUTHENTICATION SYSTEM

Magic Link for External Users

What is Magic Link?

One-time email login links. No password needed - click the link, you're in.

Use Case: Suppliers

External users like suppliers access the portal occasionally. No need to manage passwords.

Email sent via:

Resend

How It Works

1. User enters email address
2. Receives one-time link via email
3. Click link to authenticate
4. Link expires after use

No passwords

Simple onboarding

Zero resets

AUTHENTICATION SYSTEM

RBAC (Role-Based Access Control) with Scopes

The Problem

Simple roles aren't enough. Access is fragmented - depends on both WHAT you can do and WHERE you operate.

Our Solution

Combine roles with organizational scopes for flexible, scalable permissions.

Alternative: ABAC (more complex than needed currently)

Three Scopes

- **global** — Access across entire organization
- **hq** — Headquarters operations only
- **fc** — Specific fulfillment center

Example Permissions

admin + global = full access

manager + fc = manage specific FC

viewer + hq = read-only at HQ

Scales as organization grows - add roles and scopes independently

Drizzle ORM - TypeScript-First

Why Drizzle?

- TypeScript-first with type inference
- SQL-like syntax (familiar)
- No code generation step
- Edge runtime compatible
- Supports multiple DBs

0
Code gen

100%
Type safe

TypeScript-First ORM

Define schema in TypeScript, get full type inference. No separate schema files or code generation.

Auto Migration

Drizzle Kit generates and applies migrations automatically. Schema changes just work.

Works With

Cloudflare D1 | PostgreSQL | MySQL | SQLite

Alternative: Prisma (requires codegen, lock-in issues)

AUTOMATION

GitHub Actions Pipeline

Why GitHub Actions?

- Native GitHub integration
- Per-workflow files
- Excellent monorepo support

< 1 min CI runs

Biome is super fast. No tests yet, but Vitest parallel runs ready.

Previous: CircleCI (buggy, slow, poor monorepo support)

Pipeline Features

Build
Turborepo caching, parallel builds

Lint
Biome for formatting + linting

Test
Vitest for unit tests

Deploy
Wrangler to Cloudflare

4 Environments

local | dev | staging | production

USER EXPERIENCE

Animations & Visual Polish

Motion

Smooth animations for login page transitions and interactions.

Page transitions, form animations, micro-interactions

Lottie

Rich vector animations throughout the panel for a polished feel.

Loading states, success indicators, empty states

Gemini Banana

AI-generated video for the login page background.

Dynamic, branded visual for first impressions

Visual polish makes the difference between functional and professional

Also: Storybook + Chromatic for component docs & visual testing

THE JOURNEY

3 Days of Development

Day 1: Foundation

- Monorepo setup with pnpm + Turborepo
- Core package structure
- TypeScript + Biome configs
- Basic Next.js app scaffold
- Initial decision records

Day 2: Auth + Backend

- Identity service with Hono
- Better Auth integration
- Passkey + Magic Link setup
- Drizzle ORM + D1 database
- RBAC with scopes design

Day 3: UI + Polish

- UI package with shadcn
- Design system components
- User management panel
- CI/CD pipeline
- Motion animation for login

72	10+	5	4
Decision records	Packages created	Auth methods	Environments

TAKEAWAYS

Lessons Learned

Always Plan & Document Everything

72 decision records = context for future devs and AI assistants. Worth the upfront investment.

AI Needs Guardrails

.cursorrules + CLAUDE.md + strict Biome rules keep AI code consistent. Without them, quality varies.

Choose Boring Tech

Next.js, Cloudflare, TypeScript - proven at scale. Innovation where it matters, stability where it doesn't.

Self-Host Critical Services

Authentication with Better Auth = no vendor lock-in, no per-user costs, full control.

Edge-First Architecture

Hono + Cloudflare Workers = fast globally with simple ops. The future of backend.

Invest in DX

Storybook, Chromatic, Vitest, strict TypeScript - time spent on tooling pays back exponentially.

SUMMARY

Key Takeaways

3 Days
Development time

72
Decision records

10+
Packages created

Fully Functional Integrated Auth + Enterprise-Ready Dashboard

Next.js 16 | Hono | Better Auth | Drizzle | Cloudflare | Turborepo

AI amplifies developer productivity when given clear standards to follow

Thank You!

These slides were created with Claude Cowork

Presenter

John Adib

linkedin.com/in/MrAdib

Welcome to the new era of building with AI