

Aura Baas

1. Overview

Project Name: Aura

I - Objectives : Aura is a backend as a service (**BaaS**) designed to make developing apps easy and secure

II - Purpose: Unlike other BaaS platforms, Aura delivers an unmatched developer experience designed for speed and simplicity.

III - Scope :

What is already in Aura

1. Database
2. Role and Permission Manager
3. Auth
4. File Storage
5. Remote Serverless Functions
6. Function Scheduler
7. Realtime Events
8. Environment Manager (Dev , Prod , Custom ...)
9. Secret Manager
10. AI powered Business Intelligence

What is Coming to Aura

1. Payment (BYOP / Integration with Stripe or Konnect)
2. AI inference
3. Messaging (Email - SMS - Push Notification on Android and IOS)

IV - What is included in Aura

1. Self Hostable Backend (Backend + UI Dashboard)
2. JS / TS SDK that is framework agnostic + CLI
3. Framework specific Extensions (Only React Supported Now)

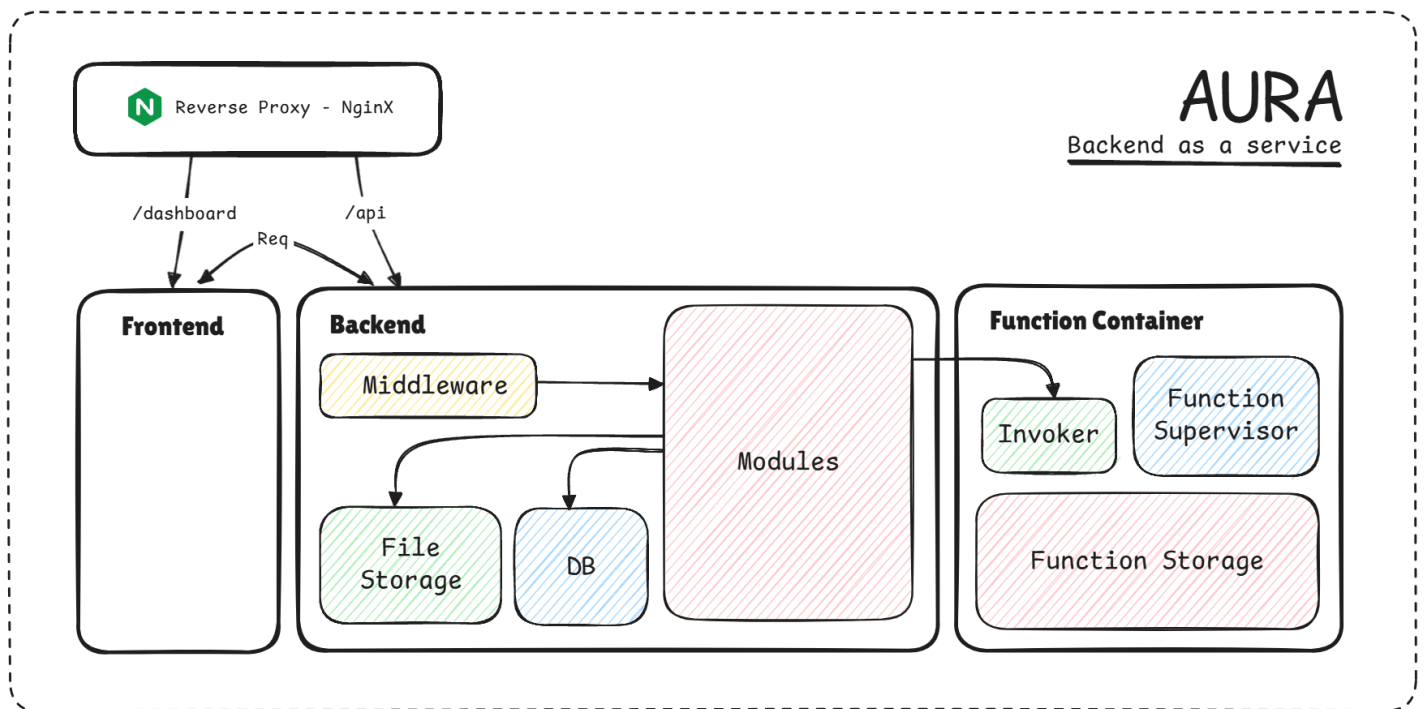
2. System Architecture

I - Tech Stack:

1. Bun (Typescript first Runtime) , High-performance runtime with speed comparable to Go.
2. Hono (HTTP Server Framework) , Built for horizontal scaling and microservices architecture.
3. OptimaDB (Serverless Database) , Offers Memory , Disk and Hybrid Configuration to minimize Query times
4. Next.js (Frontend Framework) , Used in the Admin Dashboard
5. Docker , Containerization for easy deployment and consistent environments.

II - Deployment

Aura is a collection of 3 Docker containers (Backend , Admin Dashboard , Function Container) , Deployed using "docker-compose" , can be self-hosted or managed (Coming Soon)



3. Features

I - OptimaDB (Database Layer)

- Serverless by Design: No servers, no HTTP overhead — query your database directly from code for raw performance.
- Type Safety by Default: Define schemas once and enjoy compile-time safety for queries, migrations, and CRUD operations.

- Storage Modes: Run in Memory, Disk, or Hybrid to balance speed and persistence for your app's needs.
- Auto Migration: Schema migrations are handled automatically, eliminating conflicts and manual scripts.
- Zero Dependencies: Lightweight and self-contained, with no external services required.
- Feature-Rich: Full support for JSON, arrays, relations, and advanced datatypes.
- Modern Developer Experience: Designed for TypeScript with autocompletions, intuitive APIs, and built-in utilities.

II - Authentication

- Email/password, OAuth (Google, GitHub, etc.)
- 2FA / Magic link / OTP
- Role-based access control
- Session and token management with refresh cycle.

III - File Storage

- Secure object storage with signed URLs.
- Protection against most common exploits
- S3 Compatible

IV - Serverless Functions

- Write in TypeScript, deploy instantly.
- Functions can be synchronous , asynchronous or stream responses chunk by chunk
- Functions are containerized and don't have network access unless allowed by the developer
- Functions can import external packages and import each other
- Very little overhead - Max performance

V - Realtime Events

- SSE (Server-Sent Events) based events.
- Sub-second latency for streaming updates.
- Changes are based on signals to no overhead
- Built-in presence system for chat, collaboration, and gaming apps.

VI - Environment & Secret Manager

- Multiple environment support (Dev, Staging, Prod).
- Encrypted secret storage with fine-grained access control.
- CLI integration for easy environment switching.

4. APIs

I - REST API

- CRUD endpoints auto-generated for database collections.
- Secure endpoints with role-based access rules.
- Consistent response format with error handling.

II - GraphQL API (Planned)

- Unified query language across database, files, and auth.
- Optimized resolvers for nested queries.

III - Realtime API

- Subscribe to changes in database, auth, or custom events.
- Low latency with horizontal scaling.

IV - SDKs

- **JS/TS SDK** (Framework agnostic).
- **CLI Tools** for project setup, migration, and deployments.
- Framework-specific extensions (React available, others planned).

V - CLI

- Dev Friendly Syntax (aura push / aura pull / aura init / etc.)
- Collaboration Friendly with snapshots and rollbacks