Your API Needs To Operate At The Edge

Unlock hidden performance gains

Adrian Machado, 2025

zuplo

Overview

1. What is “the Edge”?
2. Edge considerations & trade-offs
3. Why deploy APIs to the Edge
4. A quick demo of an Edge API

zuplo

Who am I?

🤠 Howdy - I'm Adrian

- Staff SWE at Zuplo
- Avid writer on all things API related
- Creator of RateMyOpenAPI
- Shoehorn evangelist

zuplo

Who are you?

You will enjoy this presentation if you are:

1. An API eng/dev looking to modernize your API stack
2. An exec (ex. CTO) that oversees API development and wants to improve your API DevX
3. Looking to build an API, but don’t know the best practices and tools to get started with

zuplo

01 What is “the Edge”?

zuplo

The Status Quo

- Traditionally, applications and APIs are deployed to a single region or datacenter
- This can lead to high latency for users on the other side of the world
- CDNs were created to host static content closer to users
- Dynamic content or compute still slow

zuplo

Introducing the Edge

- Your server code (mostly FaaS) is distributed around the world, like a CDN
- Lightweight runtime typically has 0ms coldstarts, ideal for instant responses
- Limited in bundle size, languages, and libraries
- Ex. Cloudflare Workers, Vercel edge

zuplo

02 Edge Trade-Offs

It’s no silver bullet

zuplo

The Edge Runtime

- Currently limited to Javascript (with some WASM support)
- Limited support for NodeJS libraries
- Limits on maximum bundle size
- Many libraries have support for Edge runtimes (ex. Almost any lib that runs in the browser)

zuplo

Where's your Data?

- Your compute may be globally distributed, but your database is often not
- Round-trip from Edge <> DB Location <> Edge can often lead to worse response times
- You can also end up exhausting connections to your Database
- There are some solutions to this…

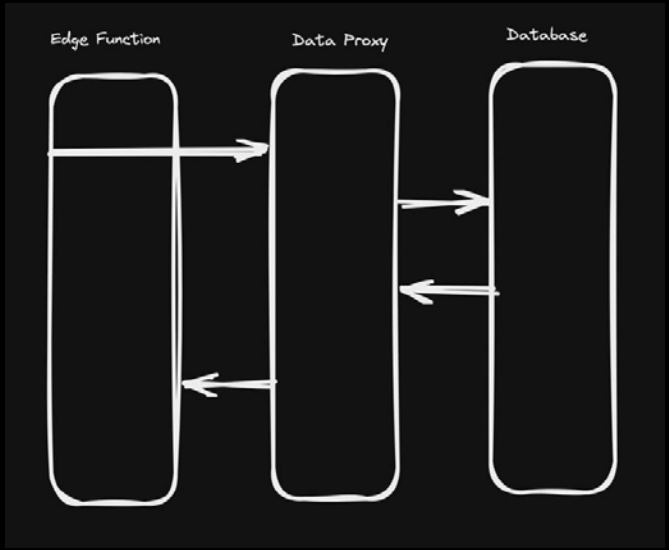
zuplo

Where’s your Data?

zuplo

Database Proxy

- To avoid exhausting connections to your database, you can use a proxy (ex. Prisma Accelerate) to manage a pool that is shared across functions



zuplo

Database Proxy cont’d

- This is a good solution for transitioning existing databases
- Can also introduce caching at this layer so fewer trips to the database are needed

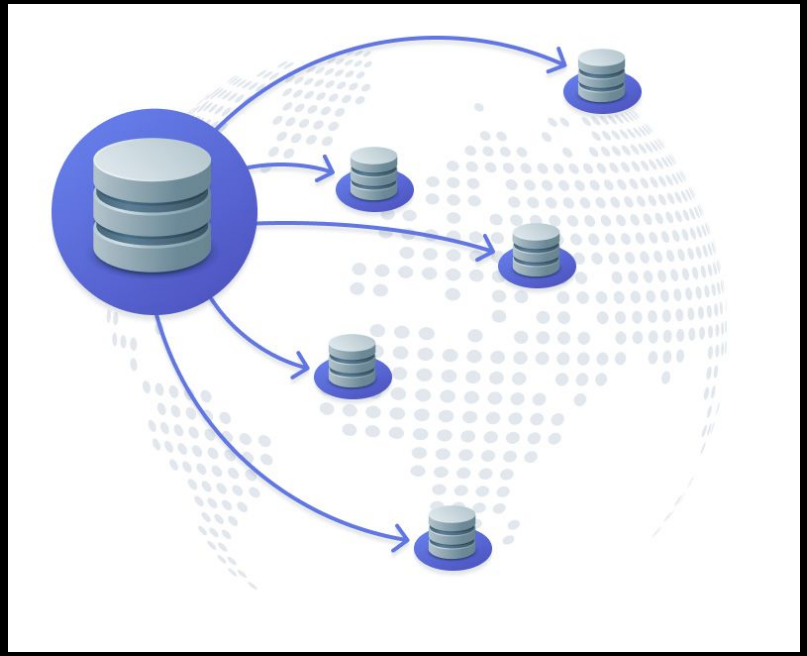
zuplo

Global Datastore

- Create distributed replicas of your database (ex. Cockroach DB) that are closer to your edge functions
- Trade-off with consistency (ex. Stale data with async writes) or latency (ex. Slow reads with sync writes)

zuplo

Global Datastore



zuplo

Data At The Edge

- In some cases, you can have data hosted or cached at the edge
  a. CF R2 edge blob store
  b. CF D1/Upstash for edge SQLite (good for per-user databases and compliance)
  c. KV / Redis / Vercel Edge cache for key-value

zuplo

Data At The Edge

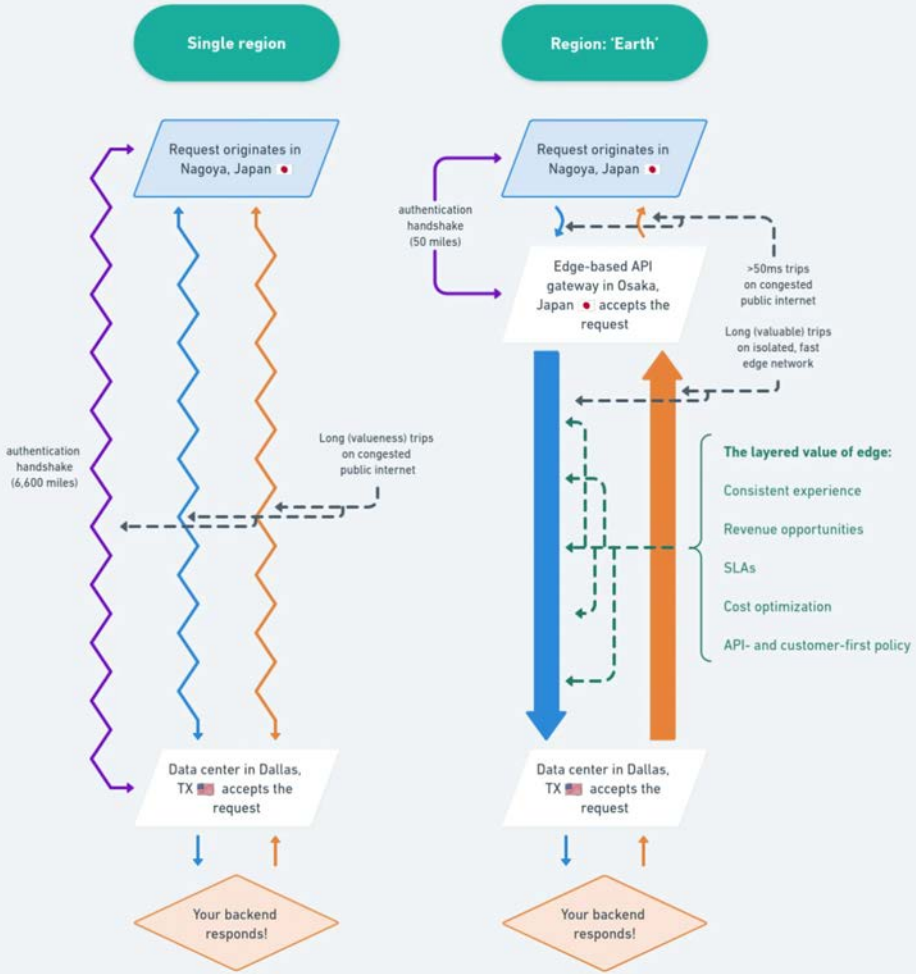
zuplo

03 Why Deploy APIs to the Edge?

Performance Matters

- Your public API users can be anywhere in the world, and slow APIs make for a poor experience - which can mean lost customers
- A lot of work done by APIs is actually filtering out calls - which is primarily compute-based, and can be performed at the edge

zuplo


Single region
Request originates in Nagoya, Japan
authentication handshake (6,600 miles)
Long (valueness) trips on congested public internet
Data center in Dallas, TX accepts the request
Your backend responds!
Region: 'Earth'
Request originates in Nagoya, Japan
authentication handshake (50 miles)
Edge-based API gateway in Osaka, Japan accepts the request
>50ms trips on congested public internet
Long (valuable) trips on isolated, fast edge network
The layered value of edge:
Consistent experience
Revenue opportunities
SLAs
Cost optimization
API- and customer-first policy
Data center in Dallas, TX accepts the request
Your backend responds!

zuplo

Work Done At The Edge

- Authentication / Authorization
- Logging/monitoring
- Request validation
- WAF (ex. Bot/bad actor filtering)
- Rate Limiting
- Caching

zuplo

Ideal Setup

- Run your API gateway at the edge (ex. Zuplo) while your services are hosted close to your data-store (and also in multiple regions)
  a. This gives you freedom over tech stack and library usage
  b. Common compute is offloaded to the edge for quick responses
  c. Gateway dynamically routes to closest region

zuplo

04 Edge API Demo

zuplo

Building an ATM Locator

- ATM locations can be replicated or stored at the Edge - they are location specific and rarely written to
- Performance is important for navigation
- Can benefit from caching since rarely updated

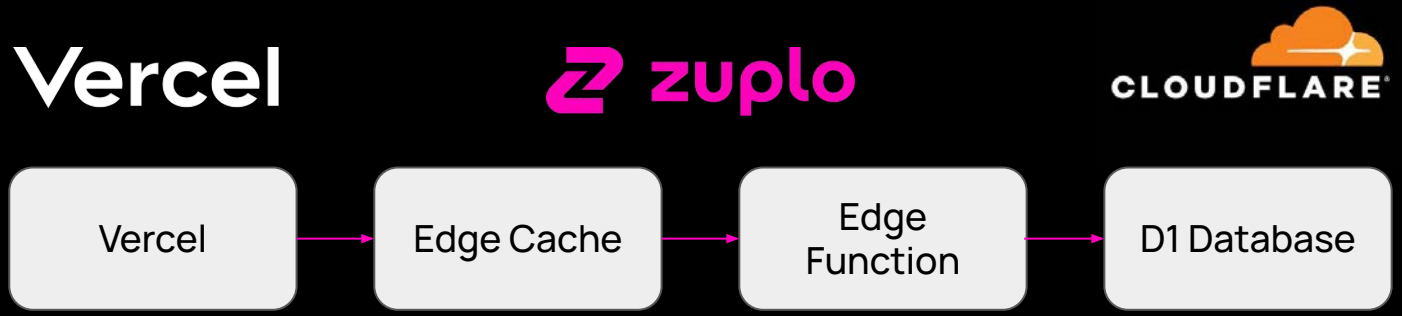
zuplo

Dataset

- Using Capital One's Nessie hackathon API
- Their API performance is already really good - Capital One already uses serverless and edge functions for their APIs and microservices
- They can return 10 results in ~100 ms - lets try and get close to that

zuplo

System Design



zuplo

System Design

- We use Vercel to host a simple NextJS site that renders ATM locations
- Zuplo is an API gateway hosted at the Edge (on the CF network)
- D1 is Cloudflare’s Serverless SQLite DB - ideally we are routed to a nearby instance

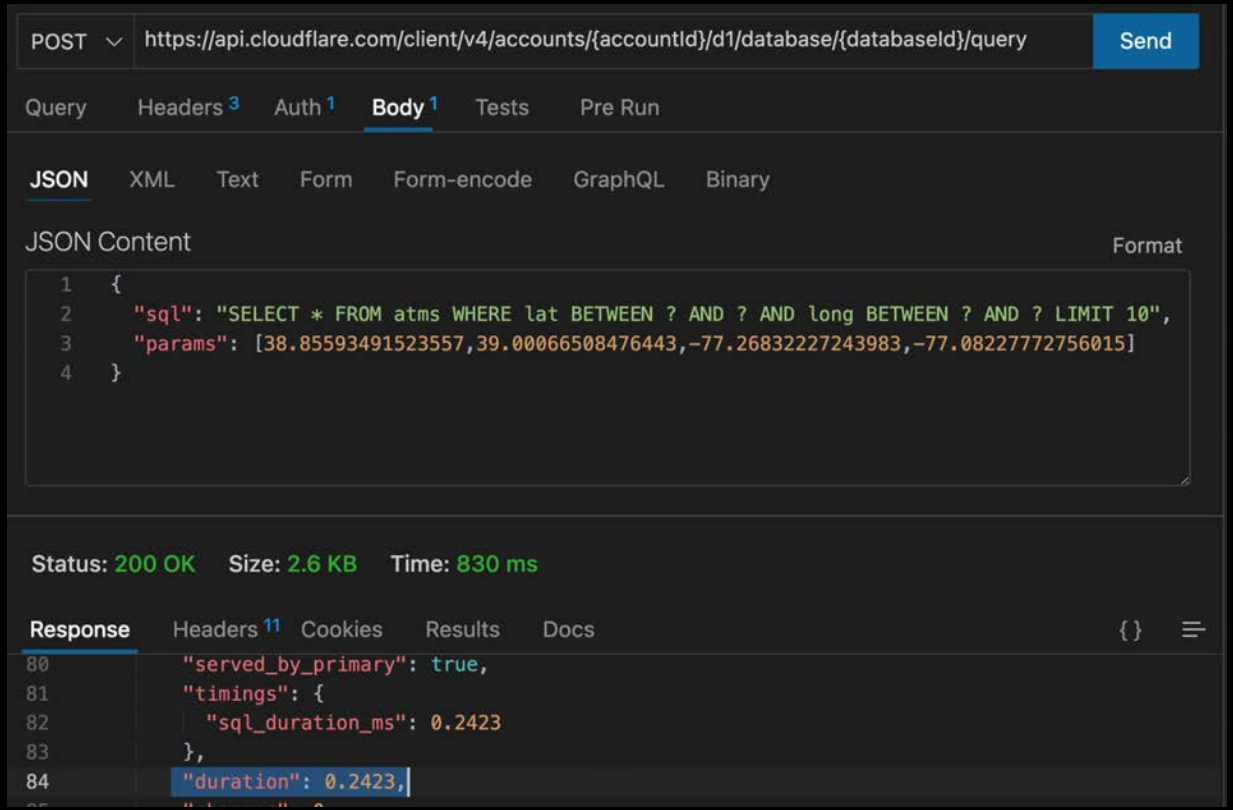
zuplo

Debugging Slowness

- The initial call we make is very slow (1000-2000ms) - this can be due to:
  a. D1 being slow at queries -> can introduce an index
  b. Our D1 instance being far (d1 will only assign a local DB at higher volumes) -> Cache results
  c. API gateway latency -> Move off the edge

zuplo

Debugging Slowness



zuplo

Add Caching

- We can cache based on a prefix of longitude and latitude (ie. regional cache)
- Results are cached at the edge for lowest latency
- Subsequent calls: ~70ms Response time
- Not bad for an entirely free stack

zuplo

zuplo