
elasticpath
SANTA'S
TECH
STACK

elasticpath

Situation

Requirements

elasticpath

Requirement	eCommerce
Track naughty & nice behaviors.	User Profiles
Allow people to send in a list of toys.	Wishlists
Have a list possible of toys and gifts	Product Information Management (PIM)
Keep track of how many gifts have been built by the elves	Inventory
Keep track of deliveries to ensure everyone gets either a gift or coal	Carts / Orders
Make sure when using the system people experience the joy of the holidays.	Unique front-ends (website, mobile, etc.)

Large Scale System

elasticpath

- 8 Billion users
- Yearly activity, but surge in data collection during December
- All fulfillment happens in a single night. (requires high-throughput)
- Users across the world. (global distributed system)
- Every possible language. (multi-language)
- Security Compliance (GDPR, ADPPA, COPPA, etc.)
- Full accessibility (a11y)

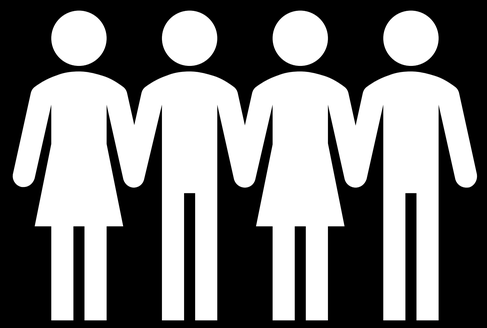
Role of Architect

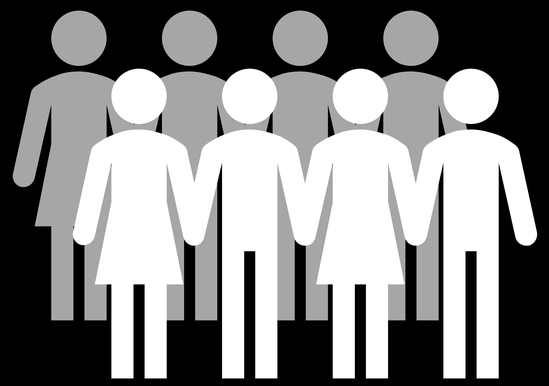
elasticpath

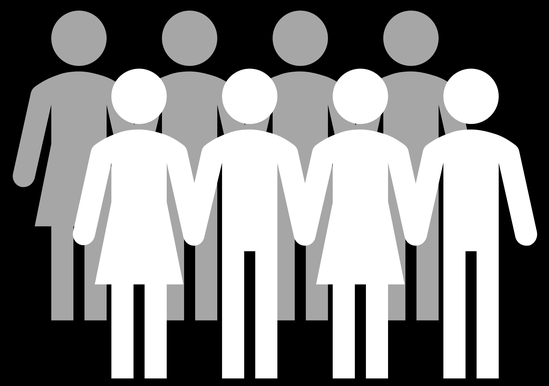
- Have a dedicated architect.
- Architecture decisions will impact the entire project and development.
- Choosing the wrong design can create lock-in, an inability to scale, or project failure.
- Be sure the architect is involved at the beginning of the process with Business Owners and Product Managers.

elasticpath

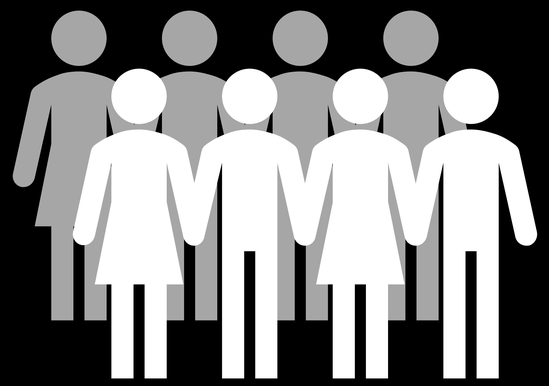
Scaling

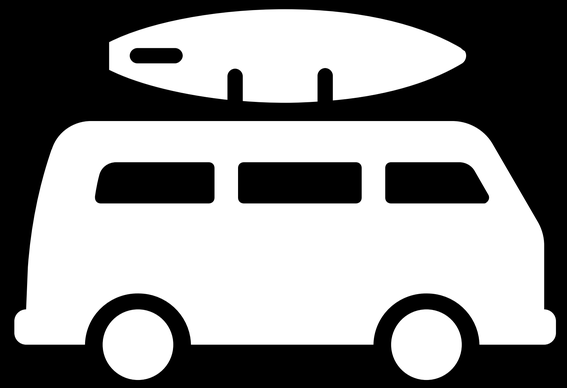


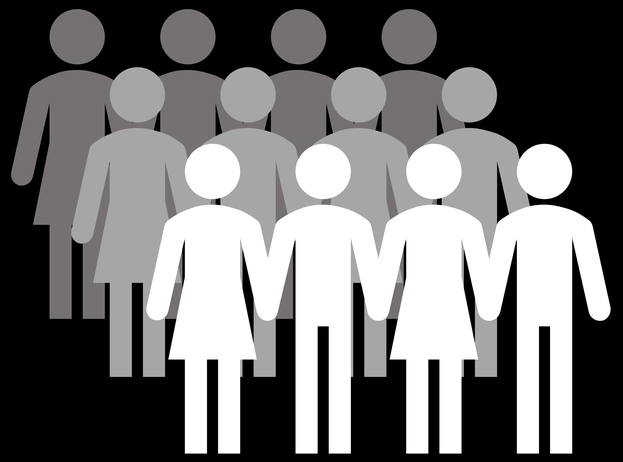


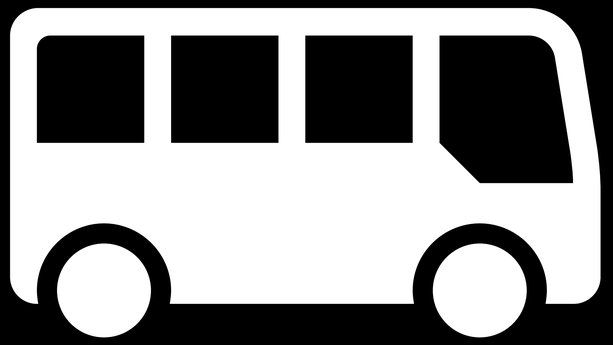


2 Trips



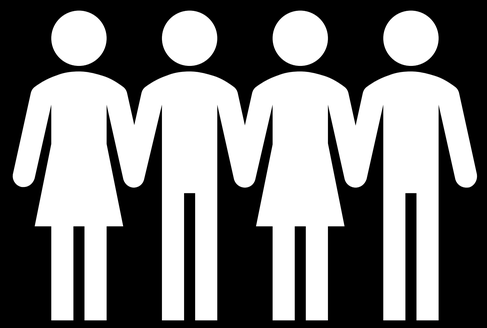


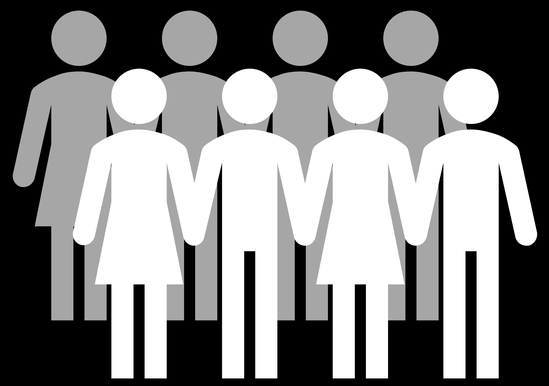


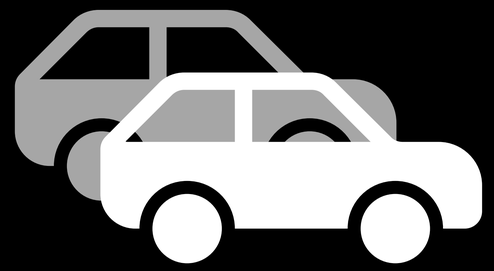


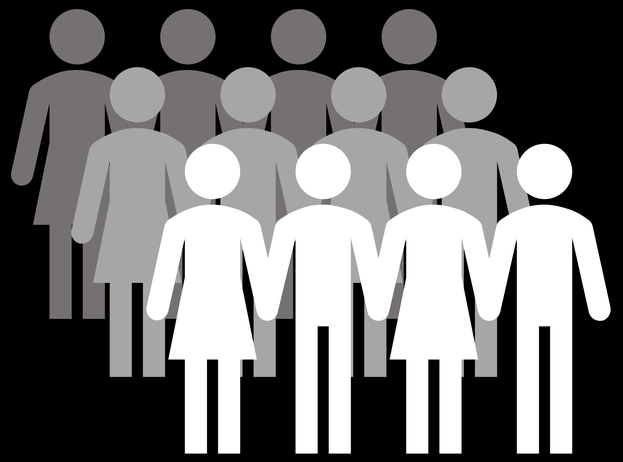
Vertical Scaling

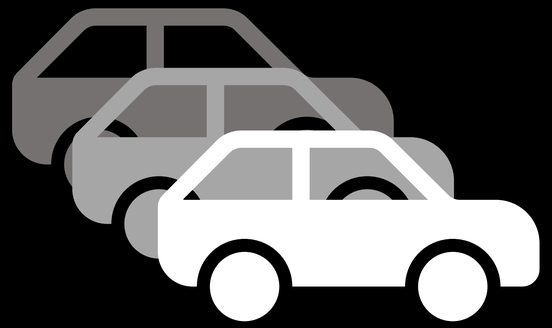
- Easiest and most common way to scale.
- Adding better hardware or for the cloud increasing capacity by adding more processing power.
- Increase storage, CPU, memory, and other resources.
- May be the best option for specific tasks (i.e. CPU bound calculations)
- Even with the cloud, vertical scaling has clear limits.











Horizontal Scaling

elasticpath

- Increased complexity and costs.
- Ability to auto-scale by adding additional servers, containers, etc.
- Can be layered on to the existing system, less downtime.
- Built-in high availability and fault tolerance.
- May be the best option for some tasks (i.e. Network Traffic)

Just use the cloud.

elasticpath

It's not as simple as adopting cloud specific options:

- Cloud offerings often create lock-in.
- We may need redundancy between clouds.
- Data residency laws may prohibit certain cloud options.

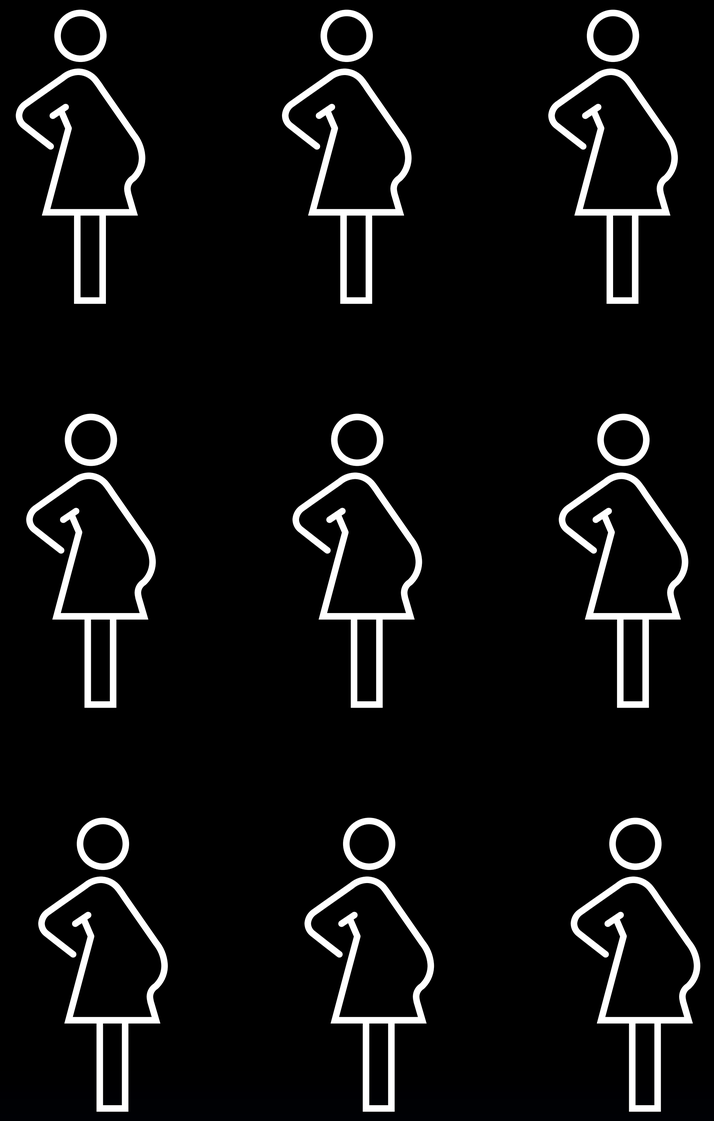
Instead, we should focus on using industry standard tools. These gravitate towards open-source.

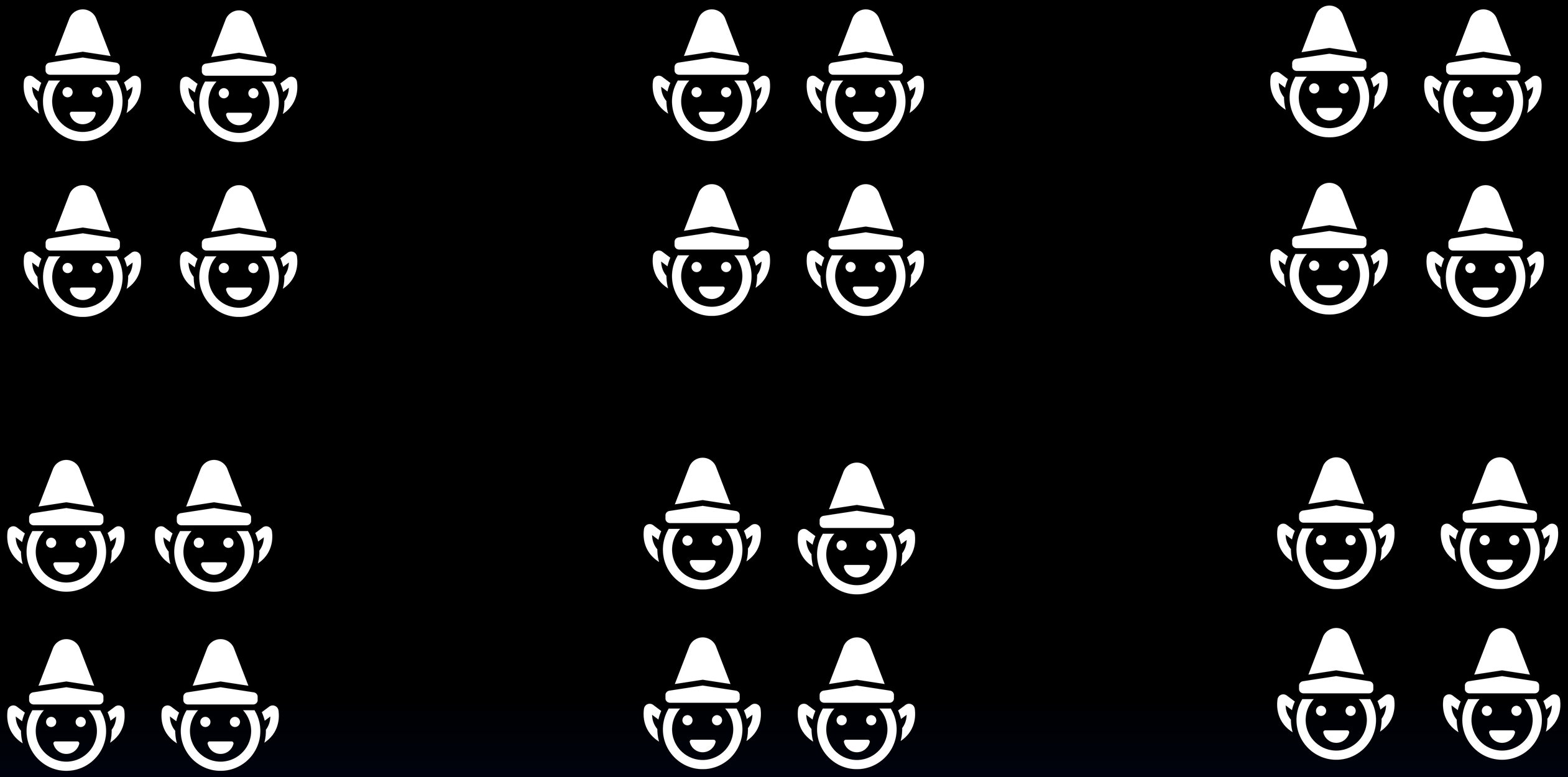
Partitioning

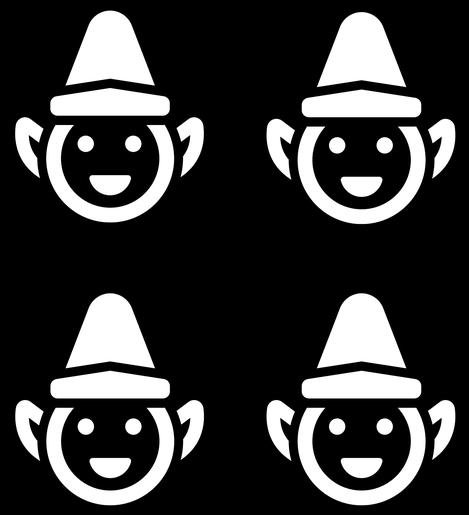
- A load balancer can be leveraged to send traffic accordingly.
- For databases either sharding or partitioning must be used.
- Consider logical partitions based on business rules:
  - Data residency laws
  - Dedicated databases for specific functions

elasticpath

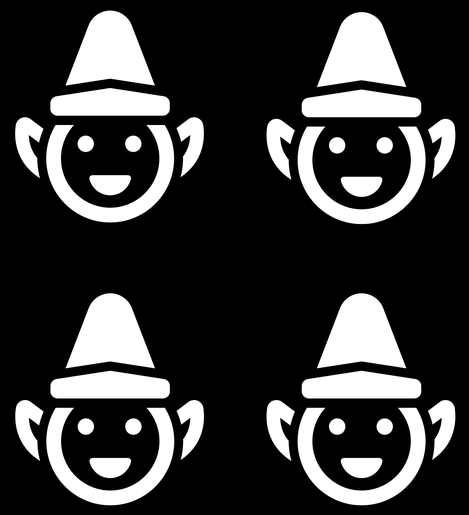
Breaking things up.







- User Profile
- Wish List
- Product Information
- Search
- Order Information
- Inventory
- Etc.



Each team:

- Full ownership & autonomy
- You build it, you run it
- Focused innovation

elasticpath

Microservices

A microservices architecture consists of a collection of small, autonomous services. Each service is self-contained and should implement a single business capability.

Characteristics of a Microservice

elasticpath

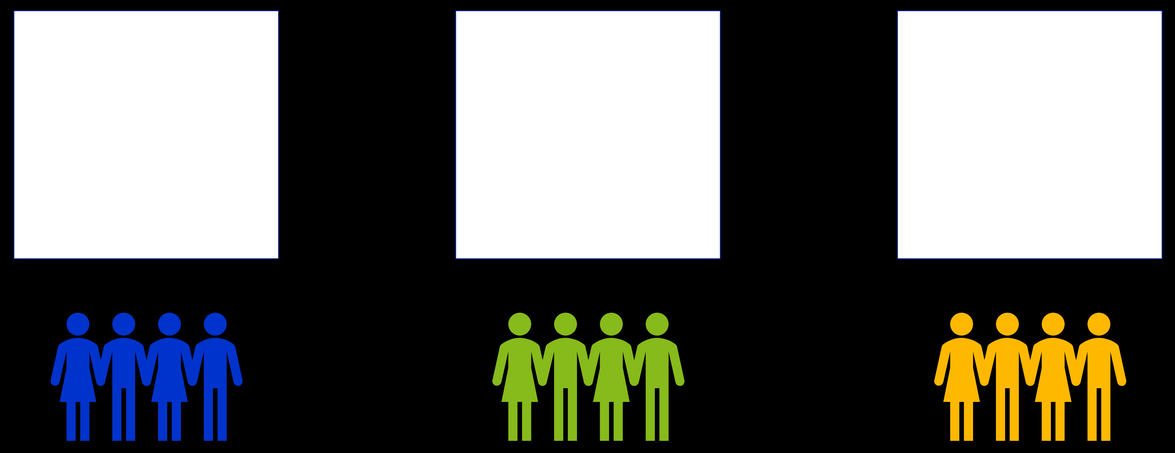
- Services are small, independent, and loosely coupled.
- Each service is a separate codebase.
- Services can be deployed independently
- Services are responsible for persisting their own data or external state.
- Services communicate with each other by using well-defined APIs
- Services don't need to share the same technology stack, libraries, or frameworks.
- Owned by a small team.

elasticpath

Microservice Antipatterns

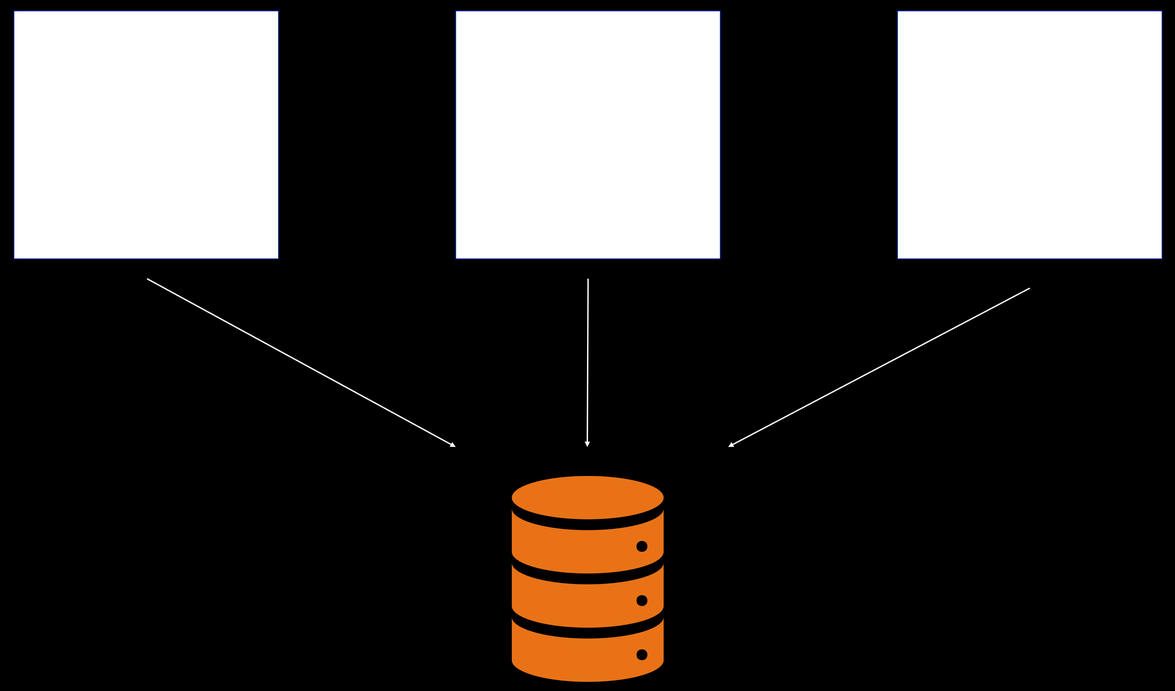
Assuming it's just tech

elasticpath



Team structure is equally important.

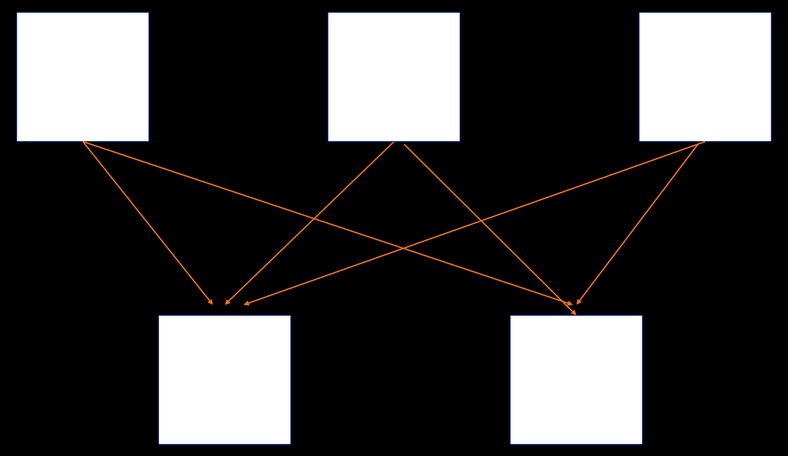
Sharing a database



This creates a single dependency

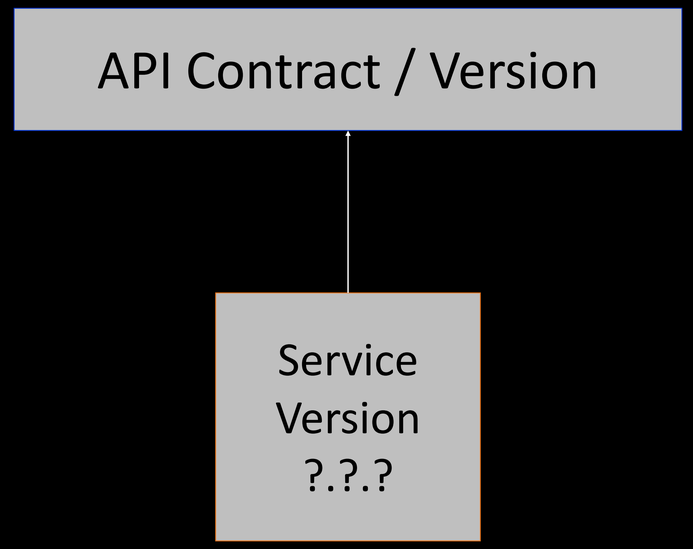
Layered Service Architecture

elasticpath



Be sure teams can work and deploy independently of other teams.

Lack of Internal Versioning



Each microservice should have versioning.

Lack of Automation

elasticpath

DevOps, CI/CD, Monitoring, Container Orchestration become more important.

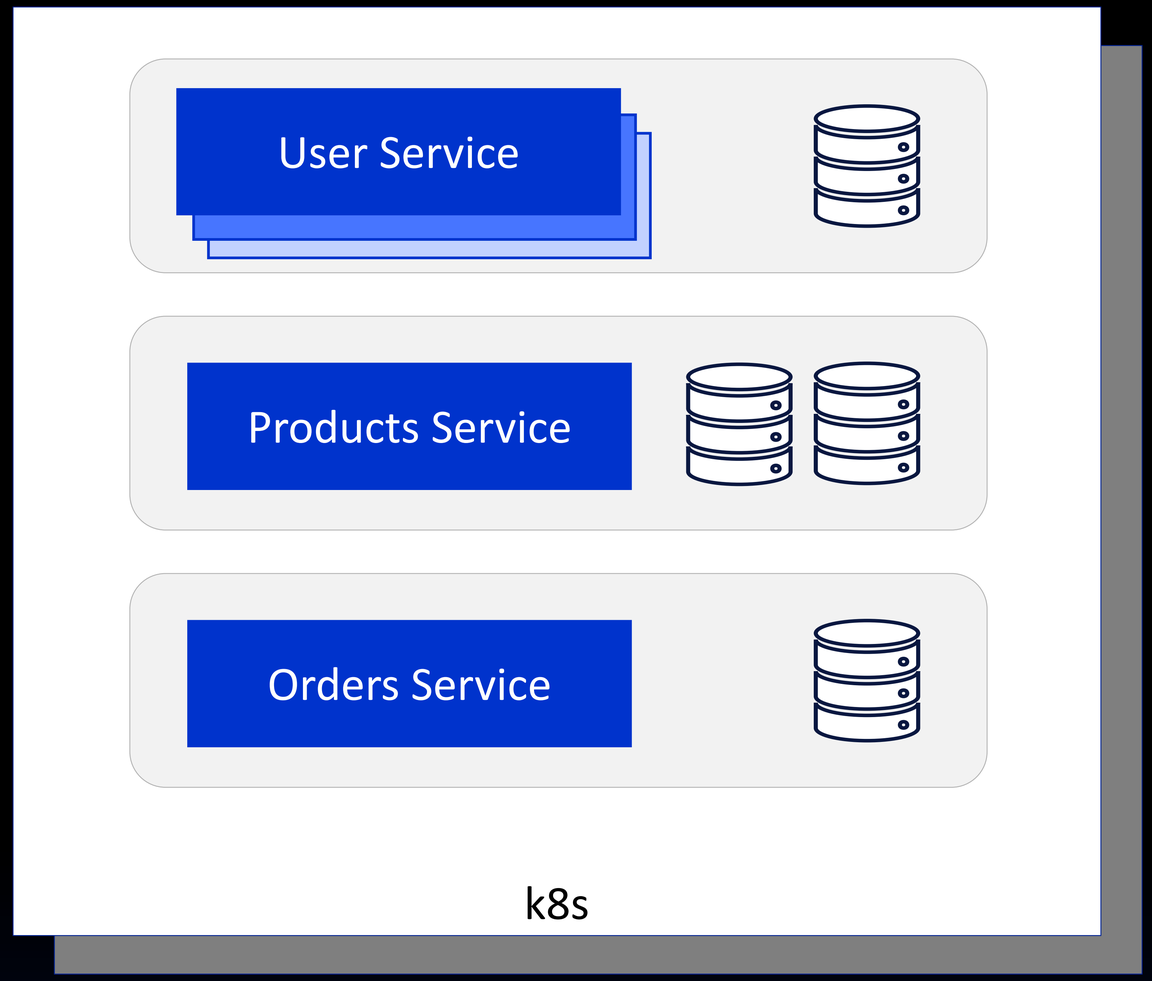
Technology

elasticpath

Services

elasticpath

CLIENT



elasticpath

Frontend / Client

Traditional Server-Rendered Website

vs

Single-Page Application (SPA)

Image Service (DAM)

$325.00 IN STOCK

Inventory Service

Multi-Vibe

1 Add to Cart

Description

Abstract geometric lampshade and matching base creates a crisp modern-looking piece of art, rather than a typical boring lamp. Designed especially for those who love to stand out and shine.

Product Service

Crown
$375.00

Green Mod
$500.00

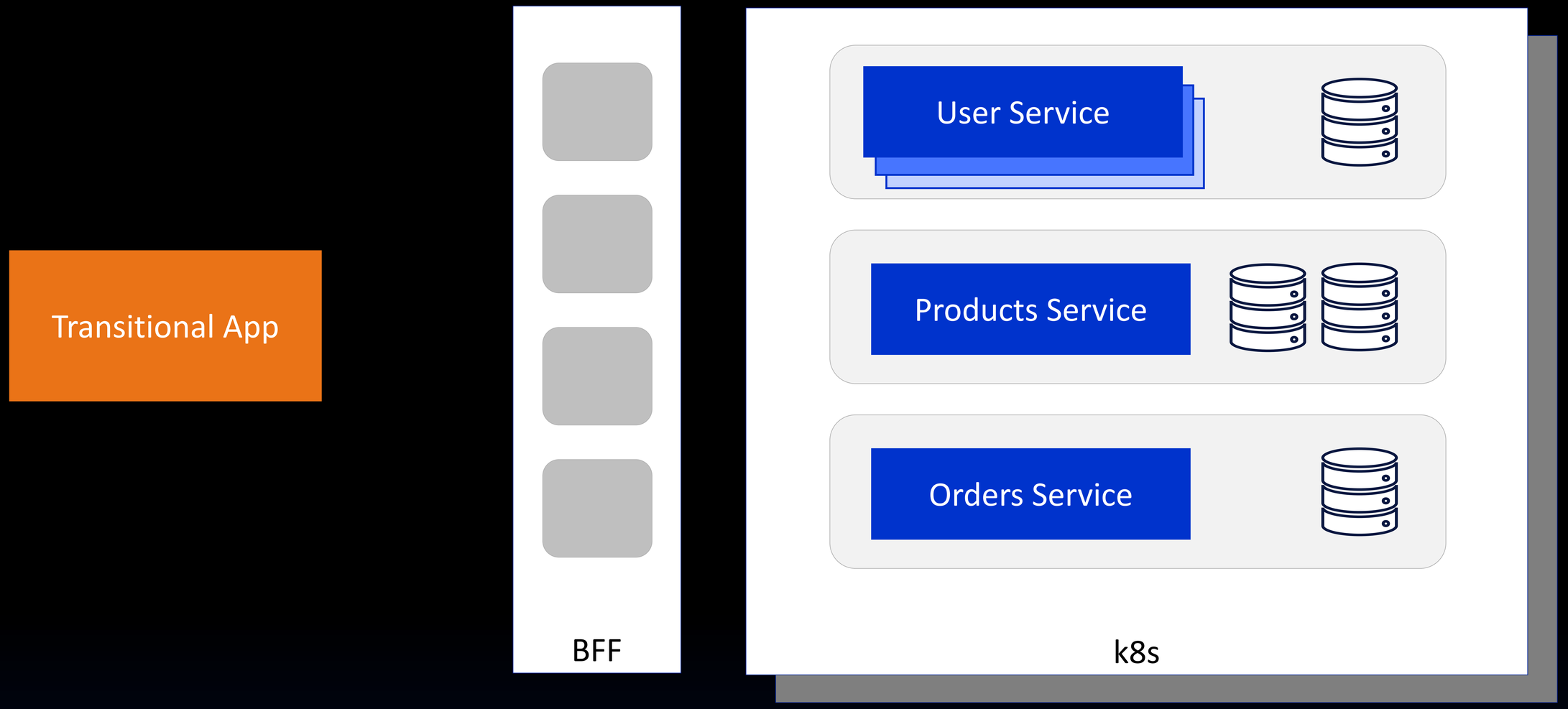
Jewellery
$500.00

Little Grey
$475.00

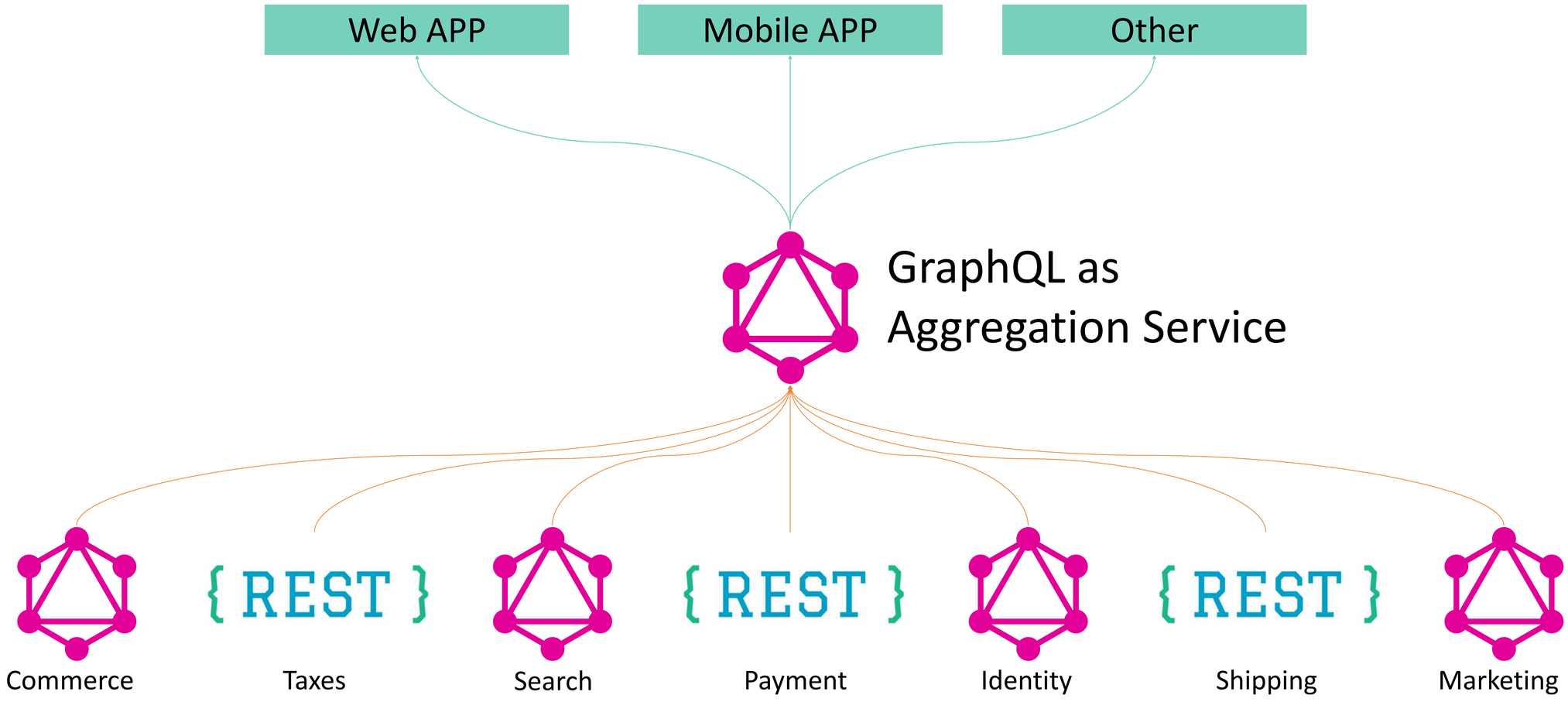
Multi-Vibe
$325.00

Orb
$450.00

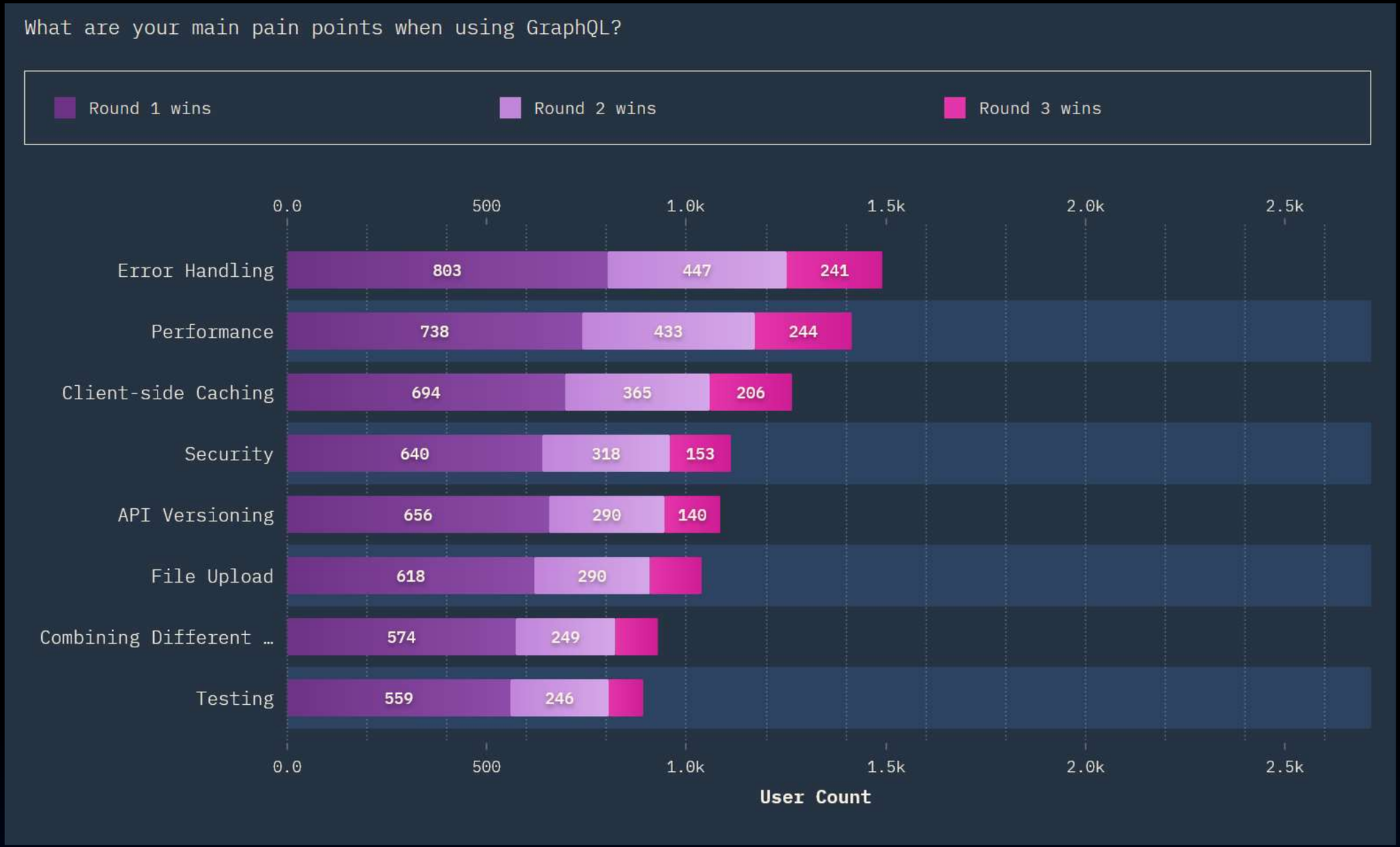
Backend For Frontend



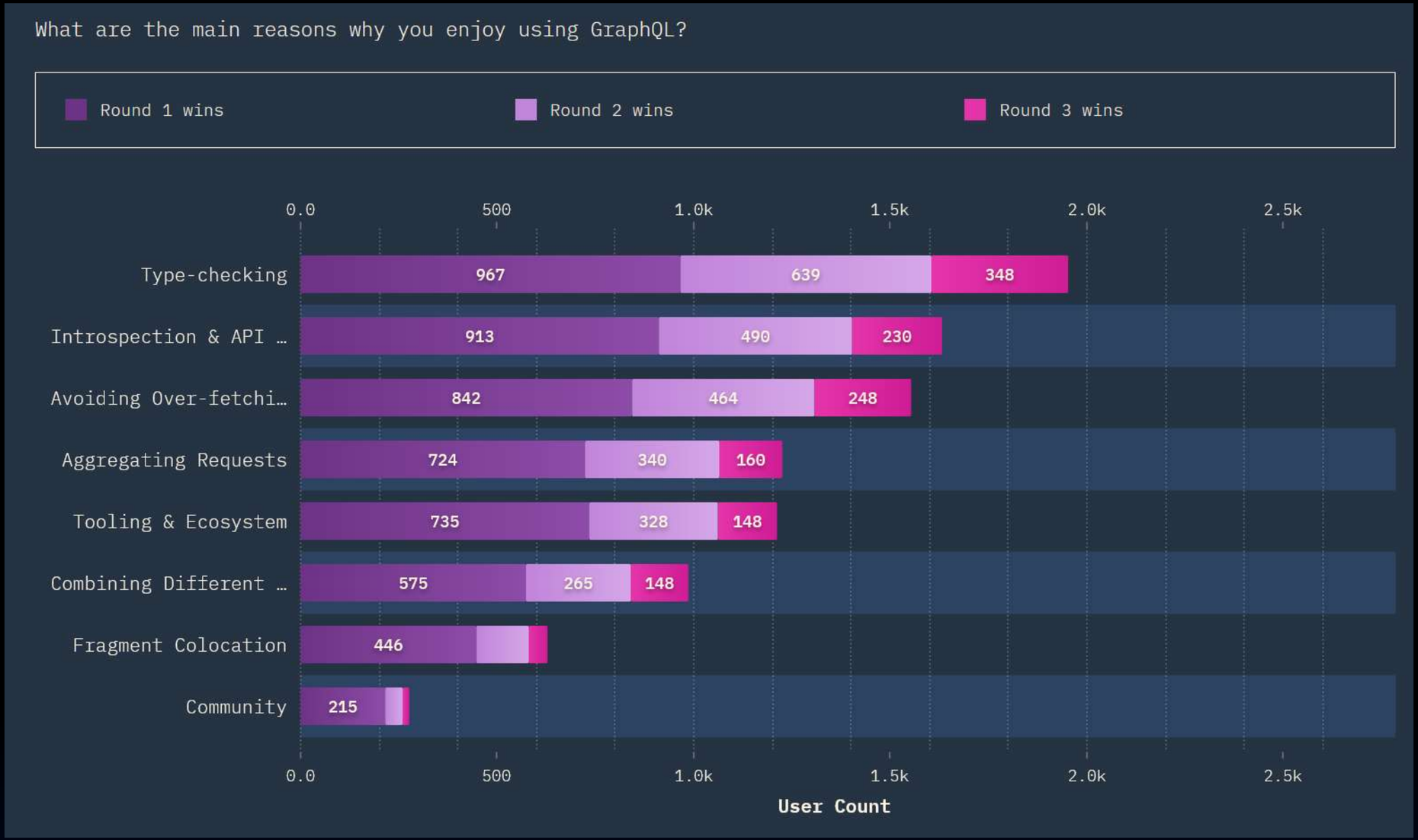
GRAPHQL AS AN AGGREGATION SERVICE



GraphQL Pain Points



GraphQL Benefits



Enter Open API Spec (aka Swagger)

elasticpath

- Writing the spec first ensures proper API design and an API-first approach.
- Specification provides a document similar to introspection, outlining the REST interface.
- Creates better tooling by generating Docs, SDKs, and more.
- Provides better versioning by tracking changes across all services.

Services

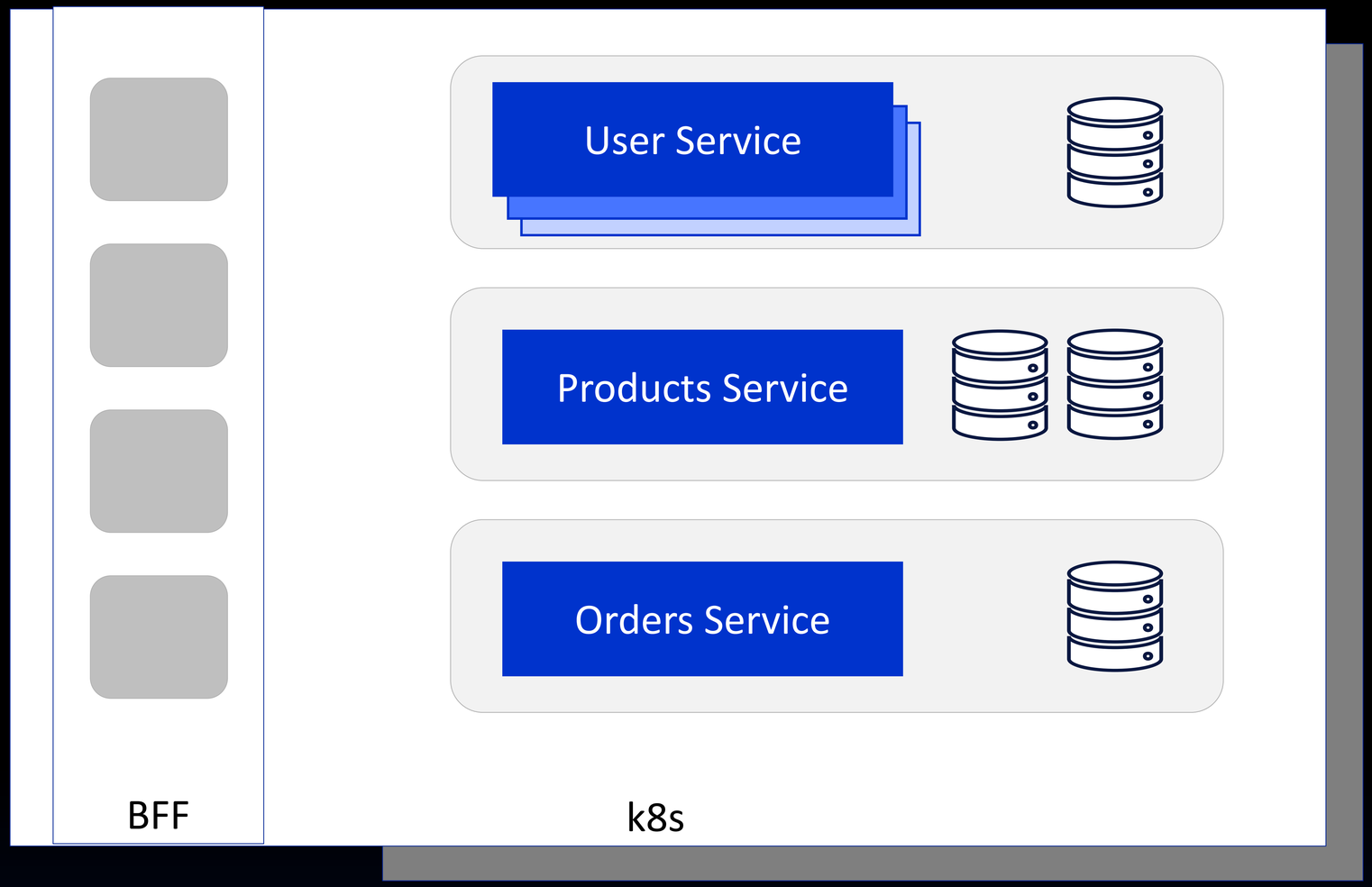
elasticpath

Transitional App

Services

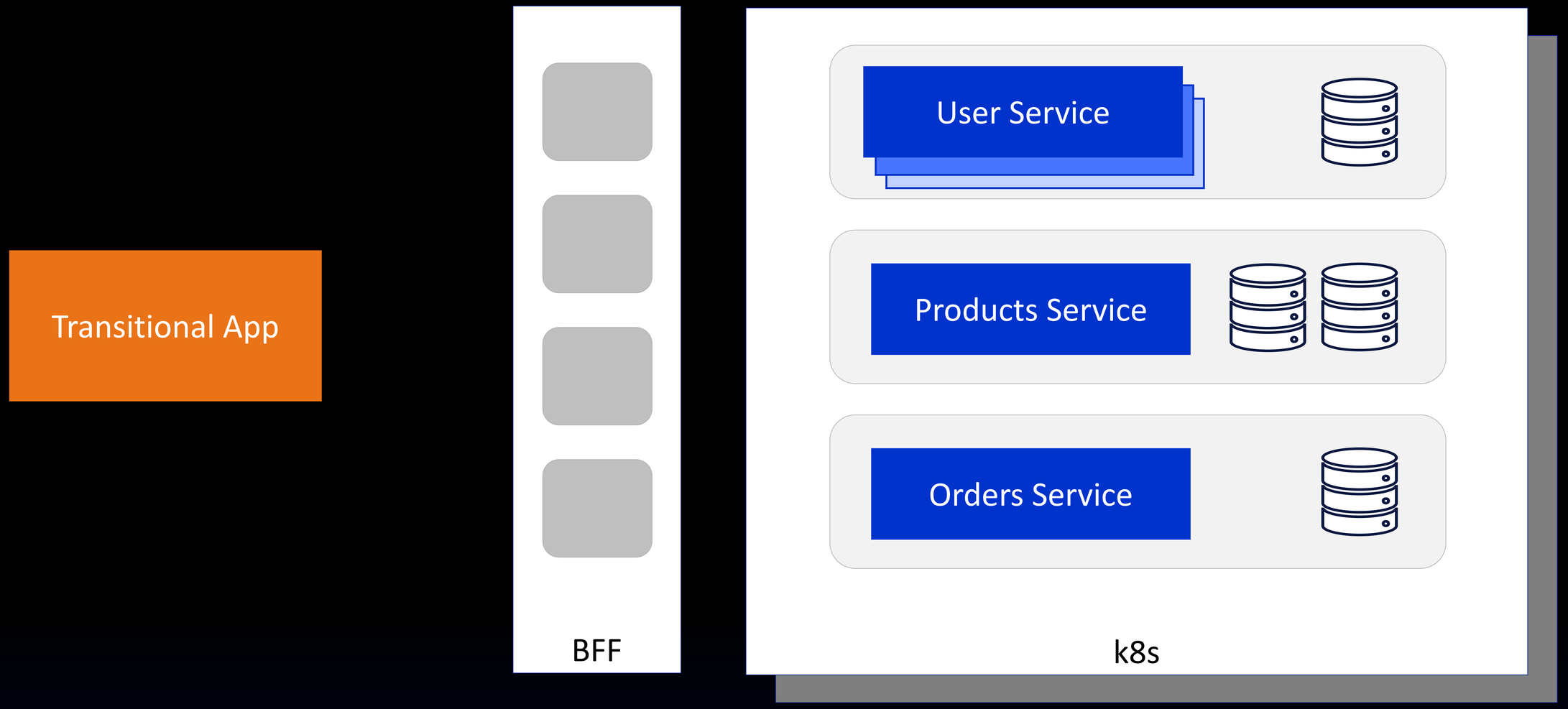
elasticpath

Transitional App



Services

elasticpath



Services


Transitional App
BFF
User Service
Products Service
Orders Service
k8s

API Routes

▾ **Examples**

- Basic API Routes
- API Routes with GraphQL
- API Routes with REST
- API Routes with CORS

API routes provide a solution to build your **API** with Next.js.

Any file inside the folder `pages/api` is mapped to `/api/*` and will be treated as an API endpoint instead of a `page`. They are server-side only bundles and won't increase your

The serverMiddleware property

Define server-side middleware.

- Type: `Array`
  - Items: `String` or `Object` or `Function`

Nuxt internally creates a connect instance that you can add your own custom middleware to. This allows us to register additional routes (typically `/api` routes) **without need for an external server**.

This allows you to create a client API/server API pattern using Nuxt alone. This means that from the browser (for example, within a Vue component) you can make a request to a route in your server middleware.

One benefit of this pattern is that the server middleware exists on the server (like most middleware), not on the client. This means that you can handle environment variables and secrets in the server middleware, without exposing that information to the user.

Because connect itself is a middleware, registered middleware will work with both `nuxt start` and also when used as a middleware with programmatic usages like express-template . Nuxt Modules can also provide `serverMiddleware` using this.addServerMiddleware()

Table of Contents

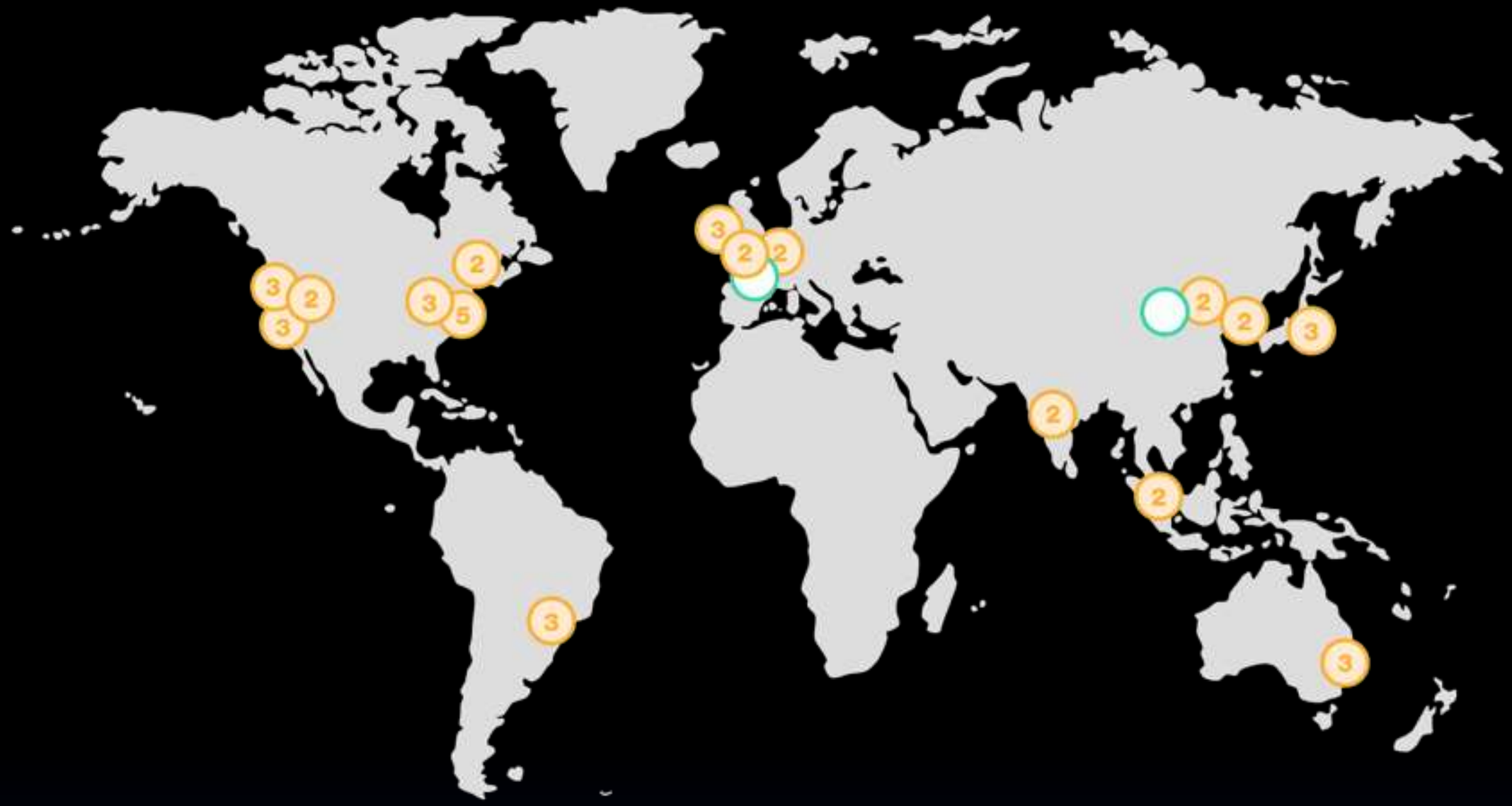
+server

As well as pages, you can define routes with a `+server.js` file (sometimes referred to as an 'API route' or an 'endpoint'), which gives you full control over the response. Your `+server.js` file (or `+server.ts`) exports functions corresponding to HTTP verbs like `GET`, `POST`, `PATCH`, `PUT` and `DELETE` that take a `RequestEvent` argument and return a `Response` object.

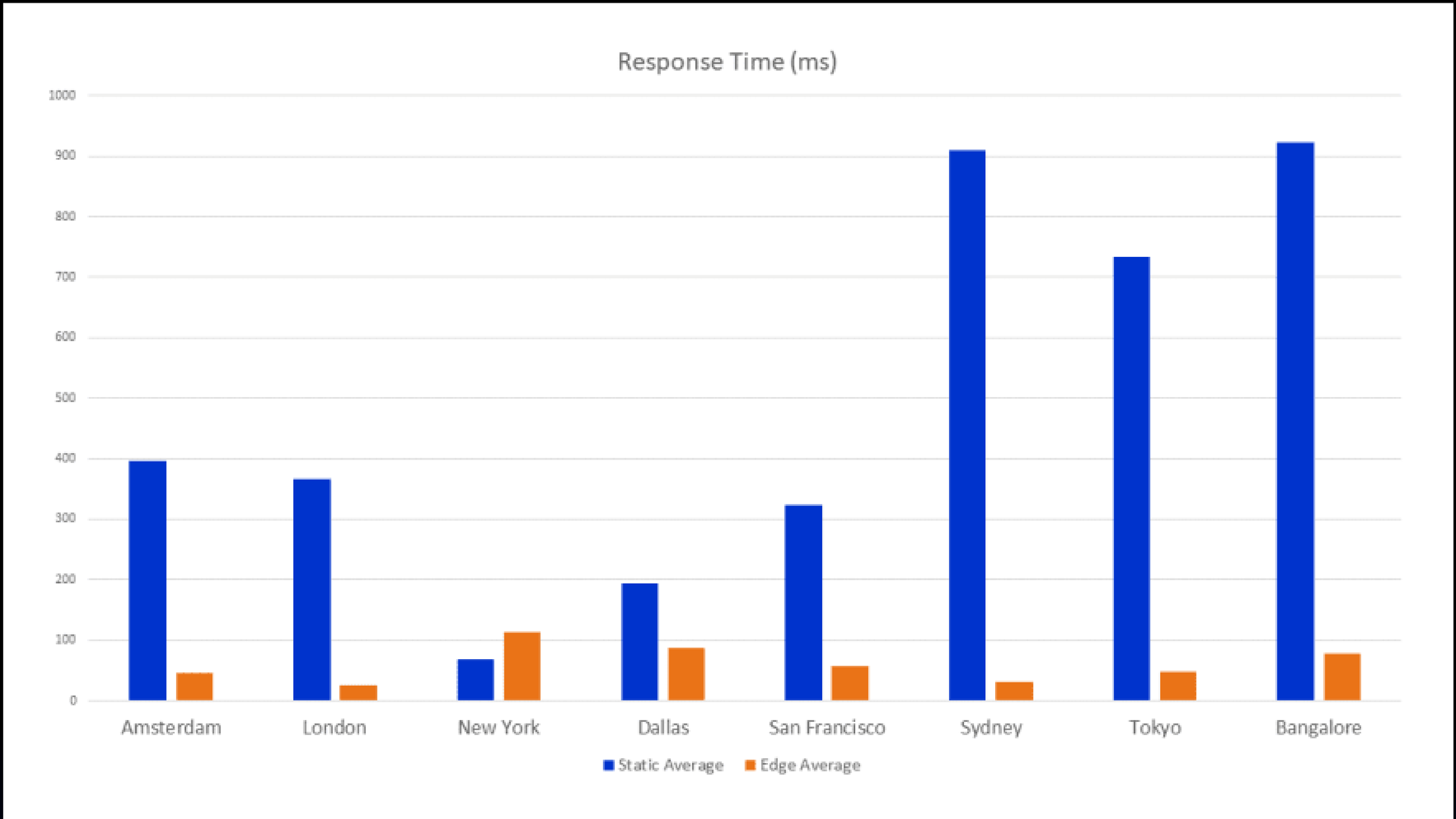
For example we could create an `/api/random-number` route with a `GET` handler:

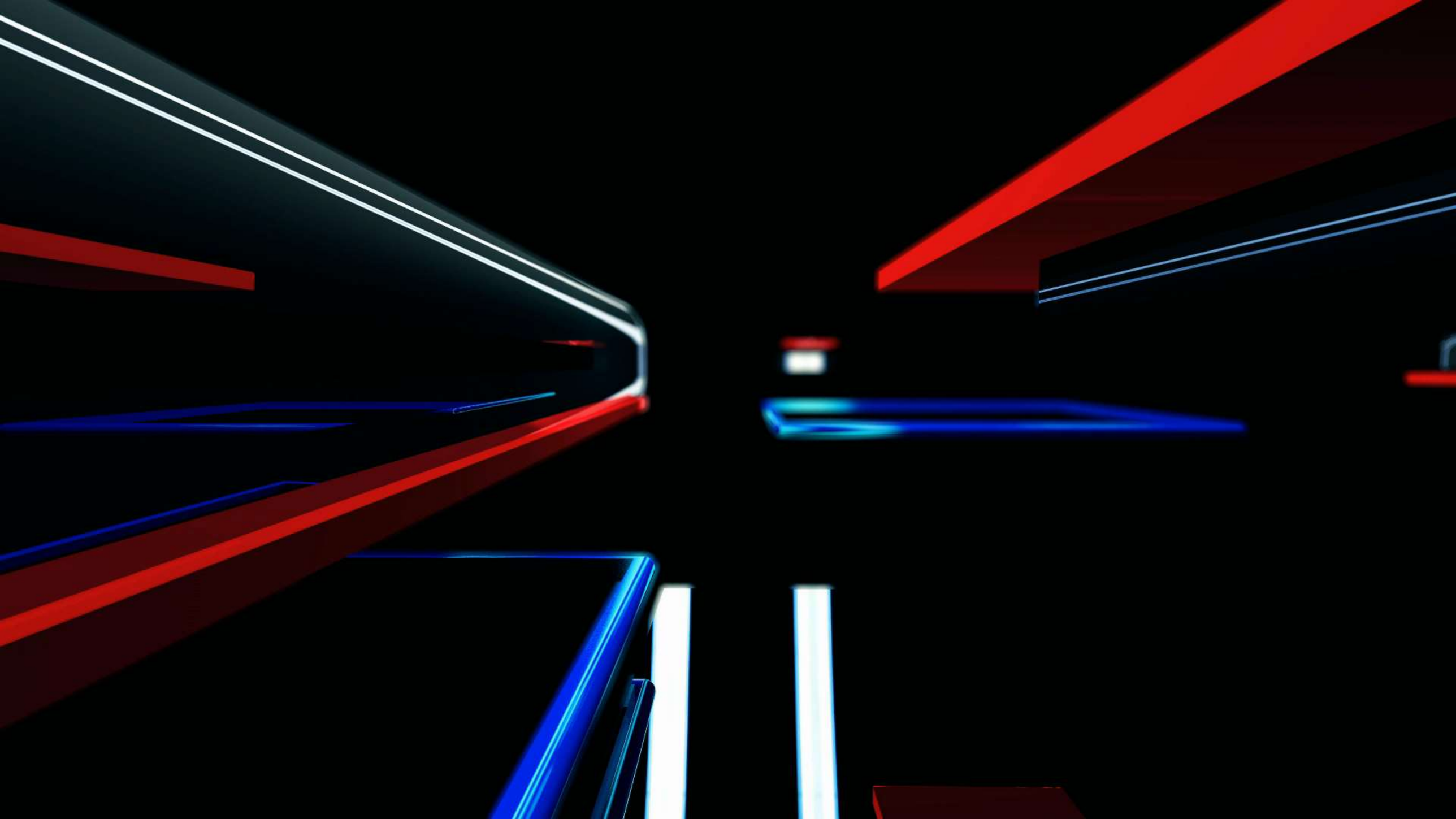
src/routes/api/random-number/+server.js

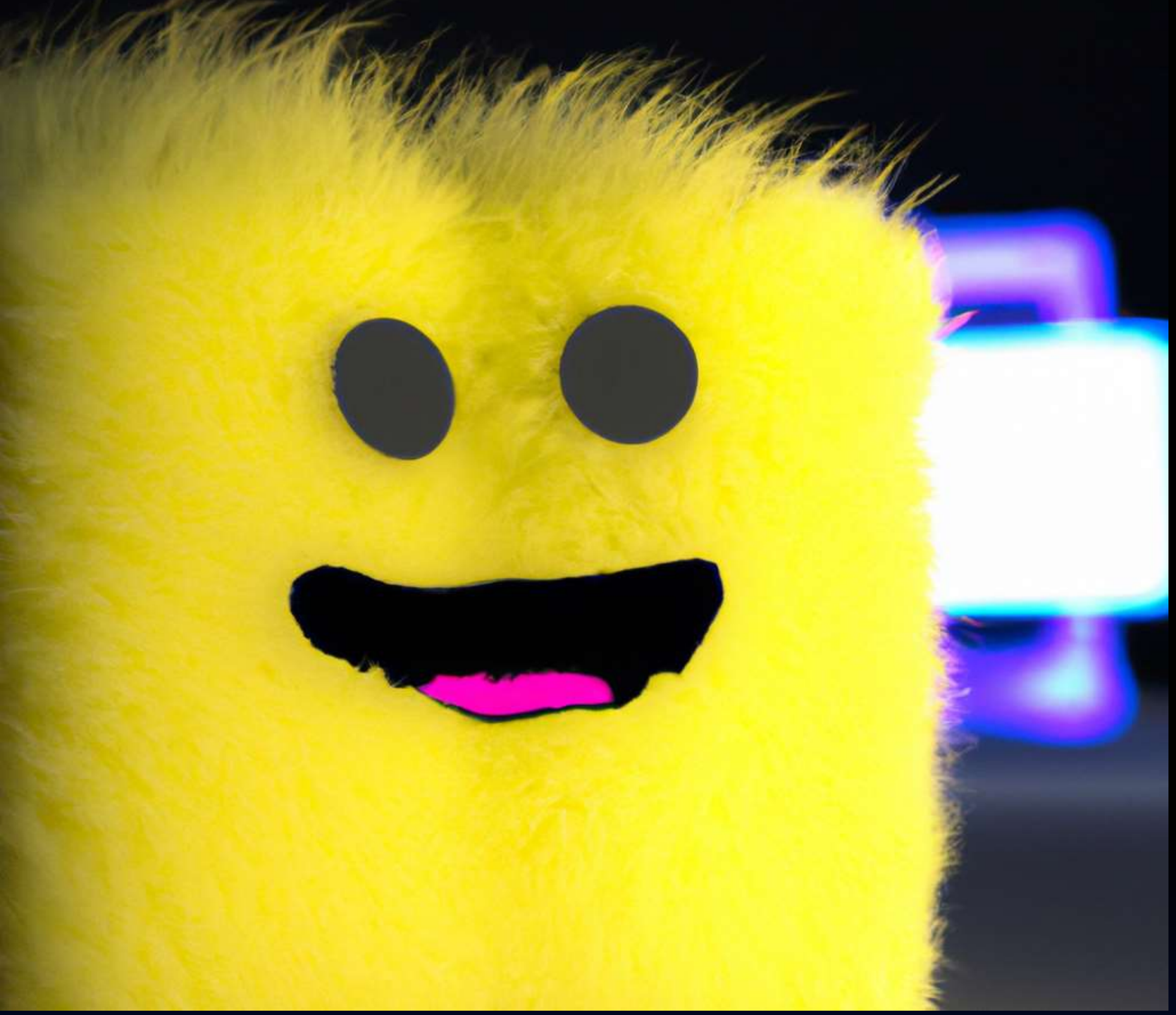
```
import { error } from '@sveltejs/kit';

/** @type {import('./$types').RequestHandler} */
export function GET({ url }) {
  const min = Number(url.searchParams.get('min') ?? '0');
  const max = Number(url.searchParams.get('max') ?? '1');

  const d = max - min;

  if (isNaN(d) || d < 0) {
    throw error(400, 'min and max must be numbers, and min must be less than ma
  }

  const random = min + Math.random() * d;
```

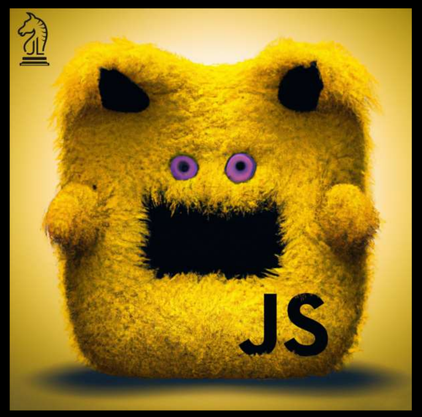


AWS Edge Locations


Response Time (ms)
1000
900
800
700
600
500
400
300
200
100
0
Amsterdam
London
New York
Dallas
San Francisco
Sydney
Tokyo
Bangalore
Static Average
Edge Average

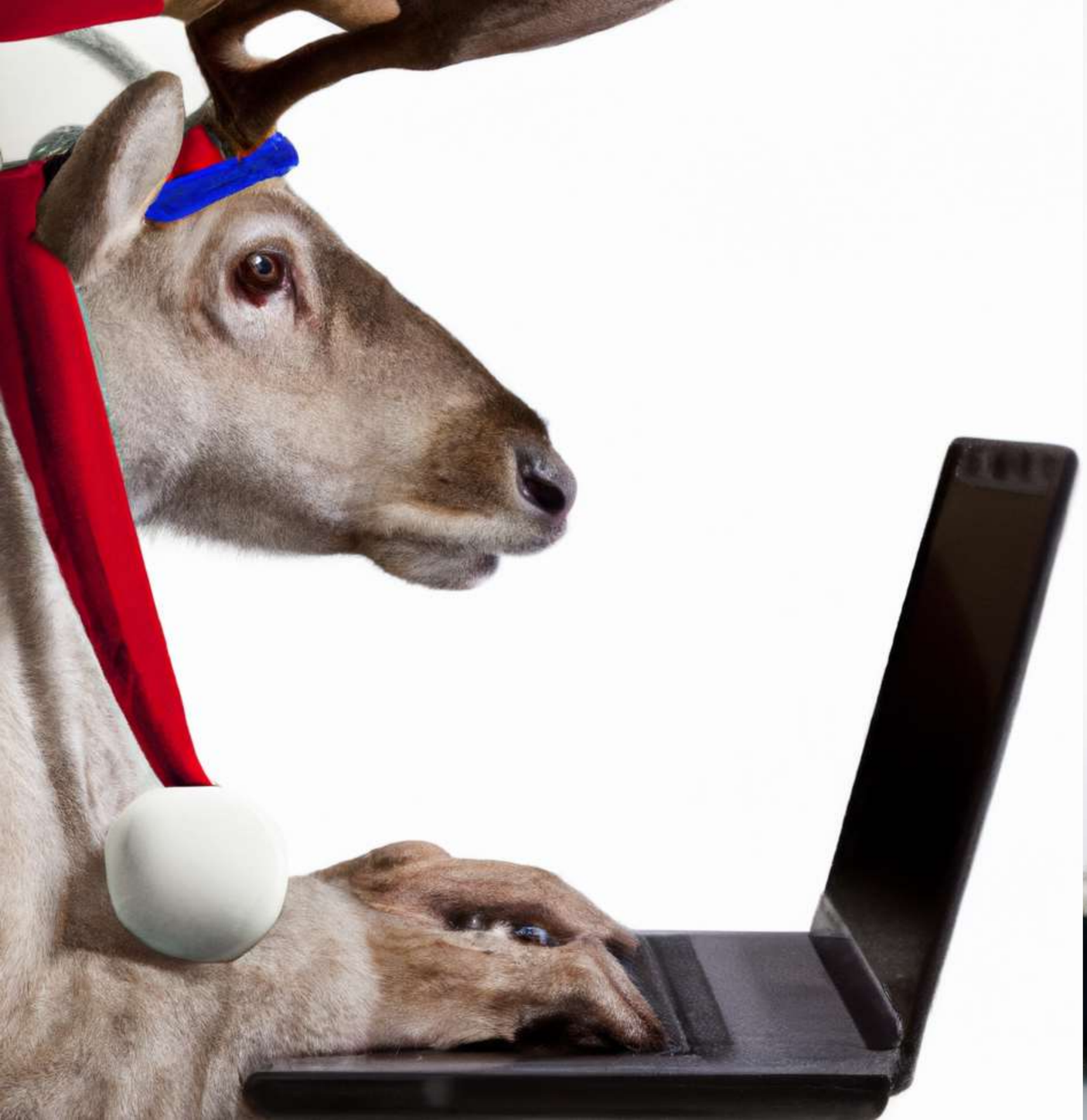





JS


JS


JS

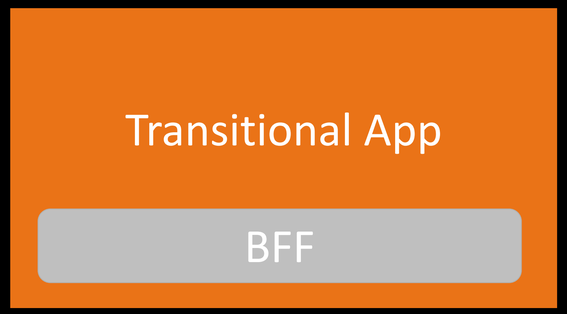


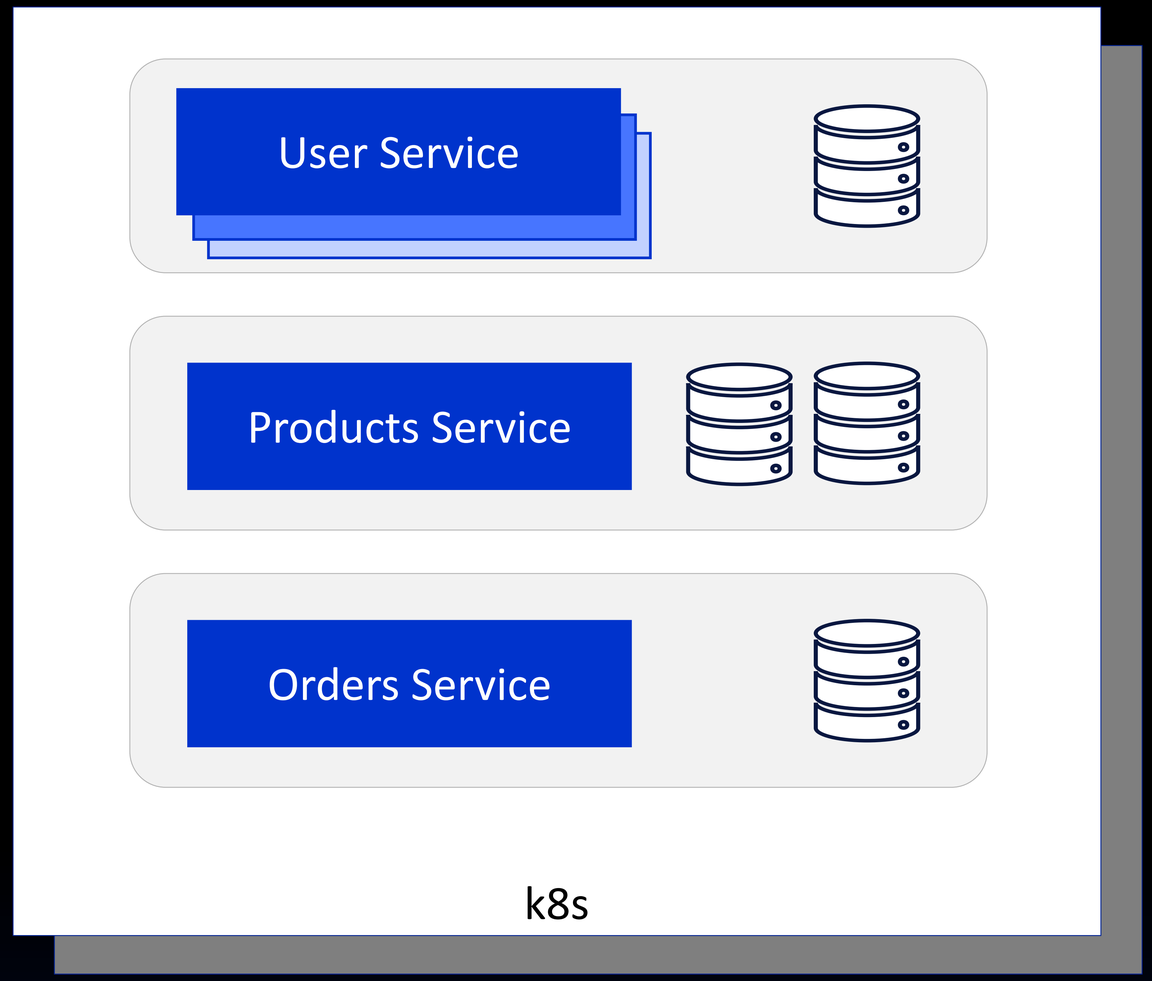
elasticpath

API SaaS

Services

elasticpath





Services

elasticpath

Transitional App

BFF

strapi

builder.io

elasticsearch

algolia

elasticpath

elasticpath

Thank you!

CONNECT

James.Luterek@elasticpath.com

In/jamesluterek

/jluterek

/jamesluterek