EP23: How to choose the right database? Also...

ALEX XU
SEP 10, 2022

In this newsletter, we'll cover the following topics:

- What does API gateway do
- REST vs. GraphQL
- Choose the right database
- Bloom filters (Youtube video)
- Troubleshooting Kubernetes deployments

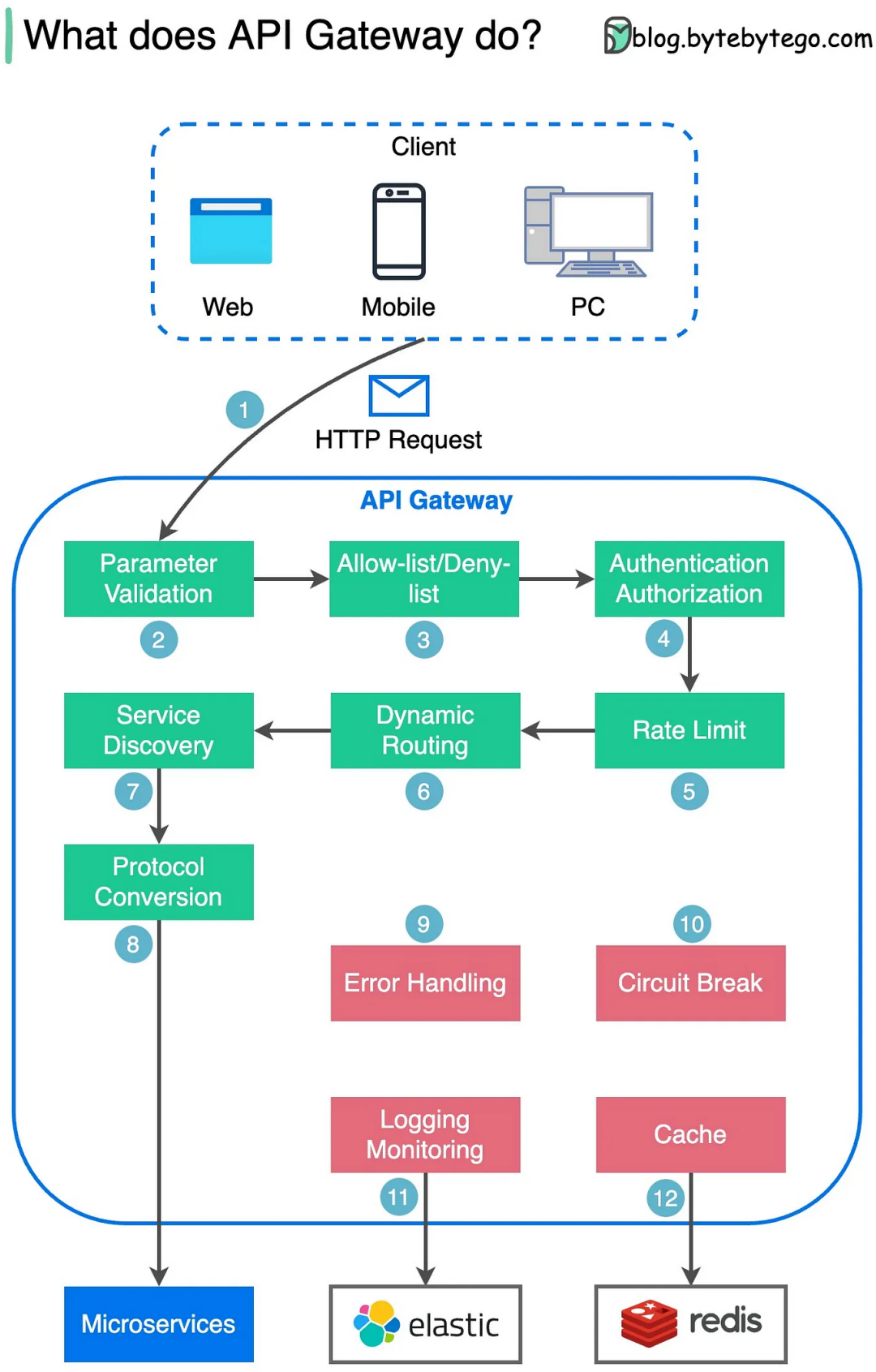
Learn 8 Data Modeling Patterns in Redis (sponsored)

Data modeling can be challenging because of various business needs. That's why we wrote a comprehensive e-book that goes through 8 different scenarios and shows how to model them in Redis along with code snippets. In it, you will learn:

- The Embedded Pattern (1-to-1, 1-to-many, and many-to-many)
- The Partial Embed Pattern (1-to-1, 1-to-many, and many-to-many)
- The Aggregate Pattern
- The Polymorphic Pattern
- The Bucket Pattern
- The Revision Pattern
- The Tree and Graph Pattern
- The Schema Version Pattern

What does API gateway do?

The diagram below shows the detail.



Step 1 - The client sends an HTTP request to the API gateway.

Step 2 - The API gateway parses and validates the attributes in the HTTP request.

Step 3 - The API gateway performs allow-list/deny-list checks.

Step 4 - The API gateway talks to an identity provider for authentication and authorization.

Step 5 - The rate limiting rules are applied to the request. If it is over the limit, the request is rejected.

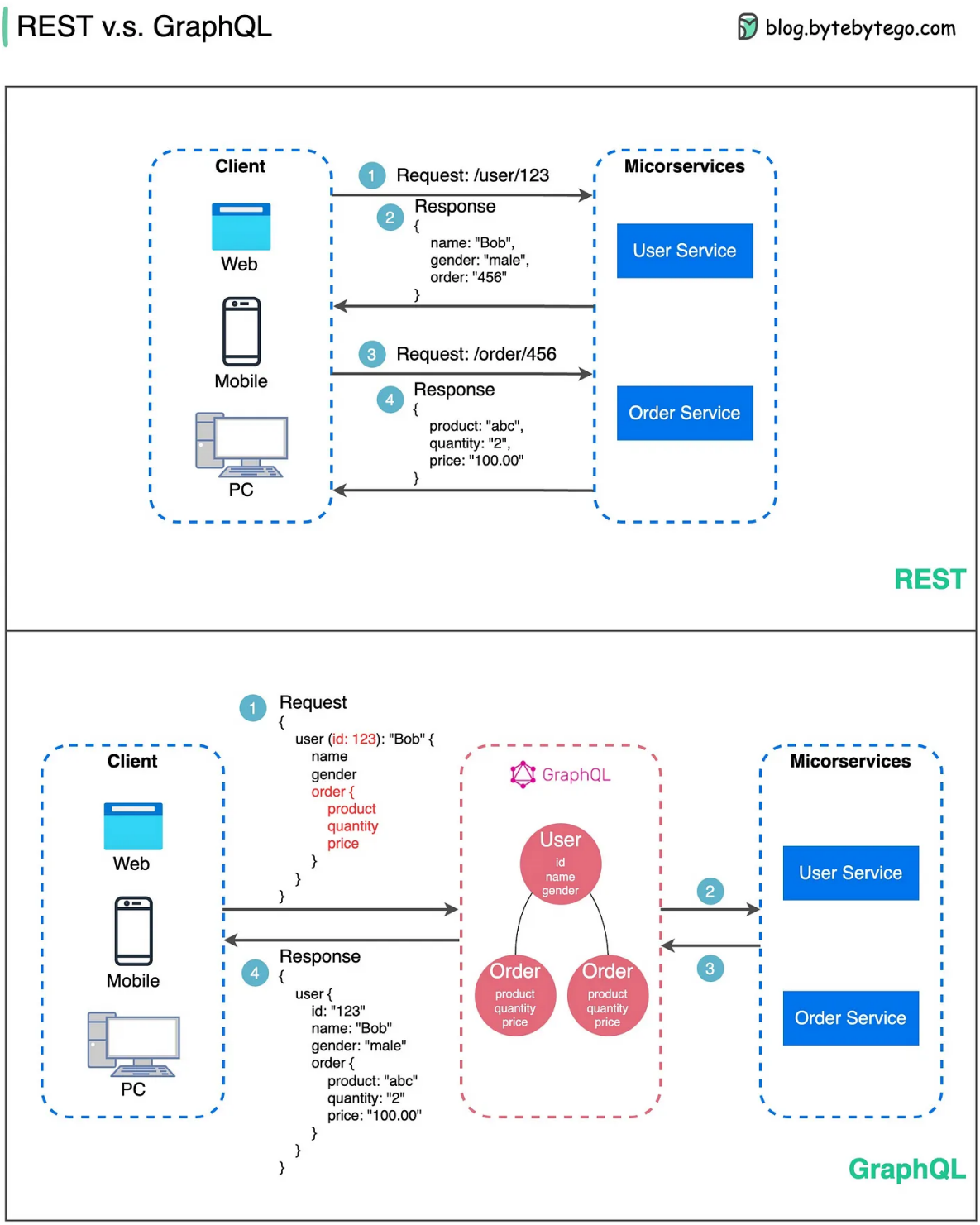
Steps 6 and 7 - Now that the request has passed basic checks, the API gateway finds the relevant service to route to by path matching.

Step 8 - The API gateway transforms the request into the appropriate protocol and sends it to backend microservices.

Steps 9-12: The API gateway can handle errors properly, and deals with faults if the error takes a longer time to recover (circuit break). It can also leverage ELK (Elastic-Logstash-Kibana) stack for logging and monitoring. We sometimes cache data in the API gateway.

What is GraphQL? Is it a replacement for the REST API?

The diagram below shows the quick comparison between REST and GraphQL.



- GraphQL is a query language for APIs developed by Meta. It provides a complete description of the data in the API and gives clients the power to ask for exactly what they need.
- GraphQL servers sit in between the client and the backend services.
- GraphQL can aggregate multiple REST requests into one query. GraphQL server organizes the resources in a graph.

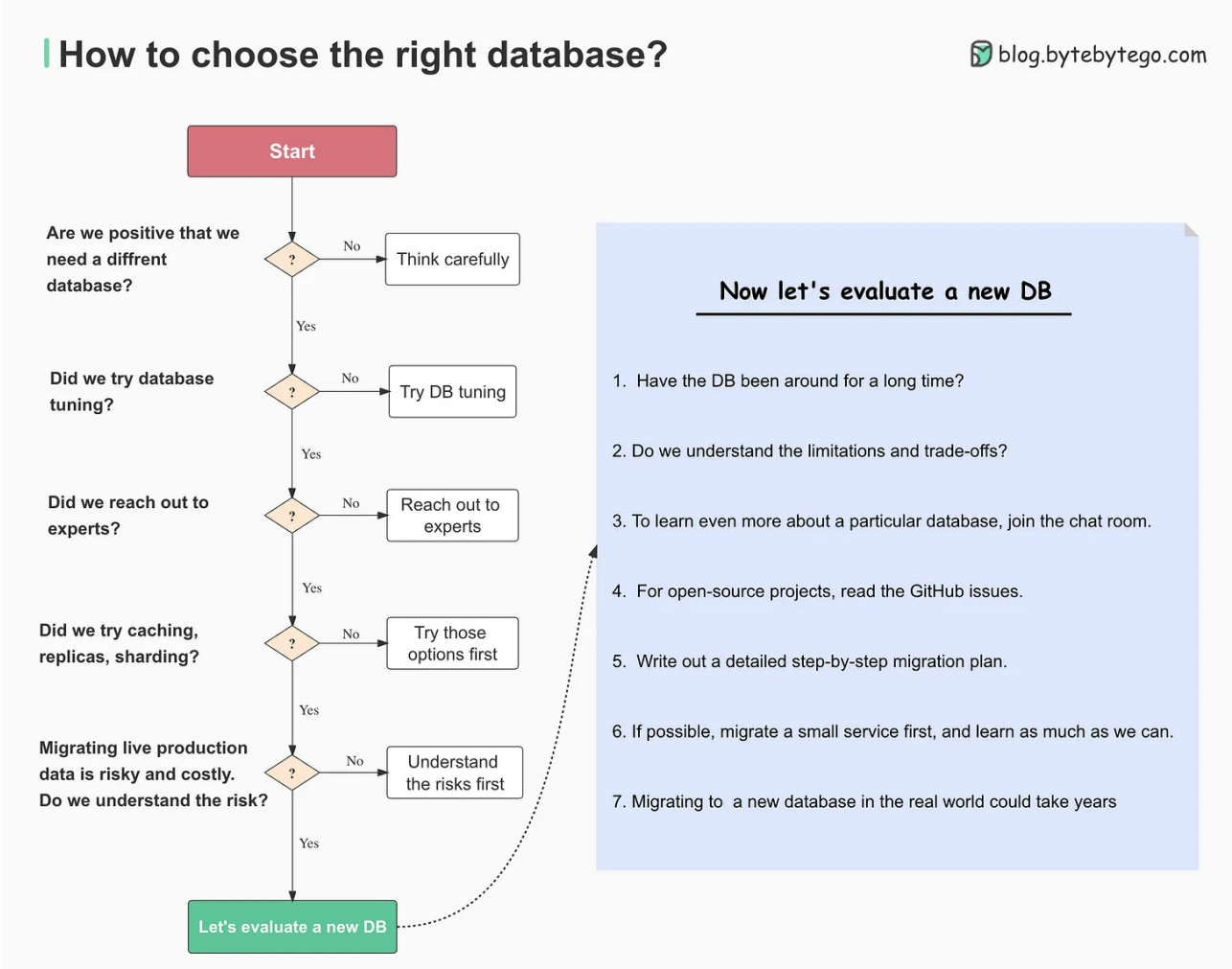
- GraphQL supports queries, mutations (applying data modifications to resources), and subscriptions (receiving notifications on schema modifications).

How to choose the right database

Choosing the right database is often the most important decision we'll ever make.

We are talking about a database for a real growing business, where a bad choice would lead to extended downtime, customer impact, and even data loss.

This take is probably a bit controversial.



First, are we positive that we need a different database?

Is the existing database breaking at the seams? Maybe the p95 latency is through the roof. Maybe the working set is overflowing the available memory, and even the most

basic requests need to go to the disk.

Whatever the issues are, make sure they are not easily solvable.

Let's read the database manual of our current database system. There could be a configuration knob or two that we can tweak to give us a bit more breathing room.

Can we put a cache in front of it, and give us a few more months of runway?

Can we add read replicas to shed some read load?

Can we shard the database, or partition the data in some way?

The bottom line is this: Migrating live production data is risky and costly. We better be damn sure that there is no way to keep using the current database.

We have exhausted all avenues for the current database.

How do we go about choosing the next one?

We developers are naturally drawn to the new and shiny, like moths to flame. When it comes to databases, though, boring is good.

We should prefer the ones that have been around for a long time, and have been battle tested.

Software engineering at scale is about tradeoffs. When it comes to databases, it is even more true.

Instead of reading the shiny brochures, go read the manual. There is usually a page called "Limits". That page is a gem.

Learn as much as possible about the candidate now. The investment is relatively small at this juncture.

Once we narrow down the database options, what's next?

Create a realistic test bench for the candidates using our data, with our real-world access patterns.

During benchmarking, pay attention to the outliers. Measure P99 of everything. The average is not meaningful.

After everything checks out, plan the migration carefully. Write out a detailed step-by-step migration plan.

Picking the right database is not glamorous, and there is a lot of hard work involved. Migrating to a new database in the real world could take years at a high scale.

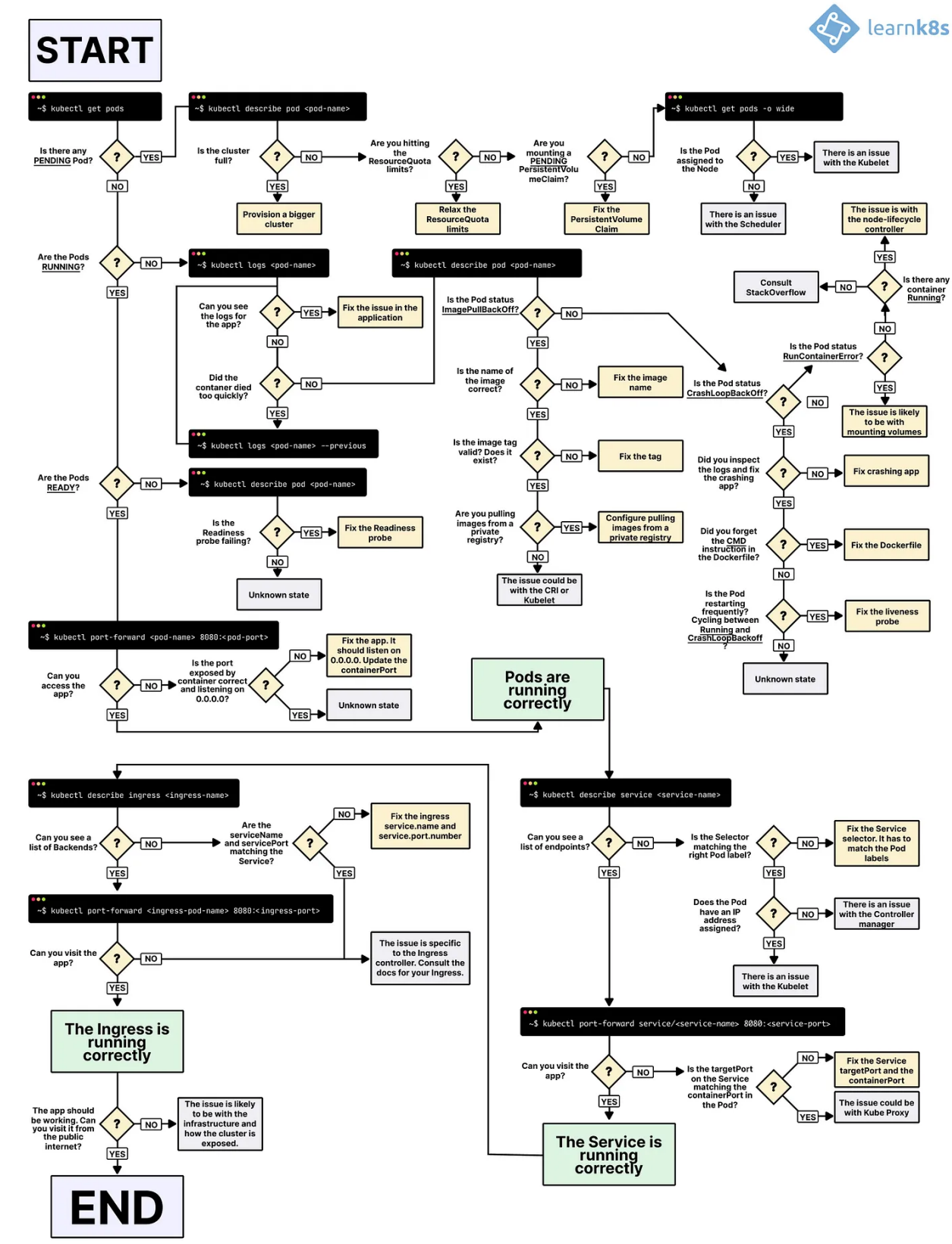
Good luck.

Bloom Filters

Algorithms You Should Know For System Design #2.

Kubernetes

A visual guide on troubleshooting Kubernetes deployments by learnk8s on Twitter.



Troubleshooting flowchart https://learnk8s.io/troubleshooting-deployments learnk8s

Thanks for making it this far!

If you want to learn more about System Design, check out our books:

[Paperback edition]

[Digital edition]

148 Likes

2 Comments

Write a comment...

Guillaume Writes Arranged ROMs (Rational Opinion... Sep 11, 2022

The question can also be: "do you need a replacement database, or an additional one along with your current database, to optimize for some use cases?". In my experience, the broader the range of use cases, the more a system can benefit from purpose-based databases.

LIKE (3) REPLY SHARE

Pat Ben Writes hacks.vc ✊ Sep 19, 2022

Just a quick question. Your diagrams and transitions/animations in your videos are so compelling. I wonder what tools (whiteboard, diagramming software, etc.) you use for this?

LIKE REPLY SHARE