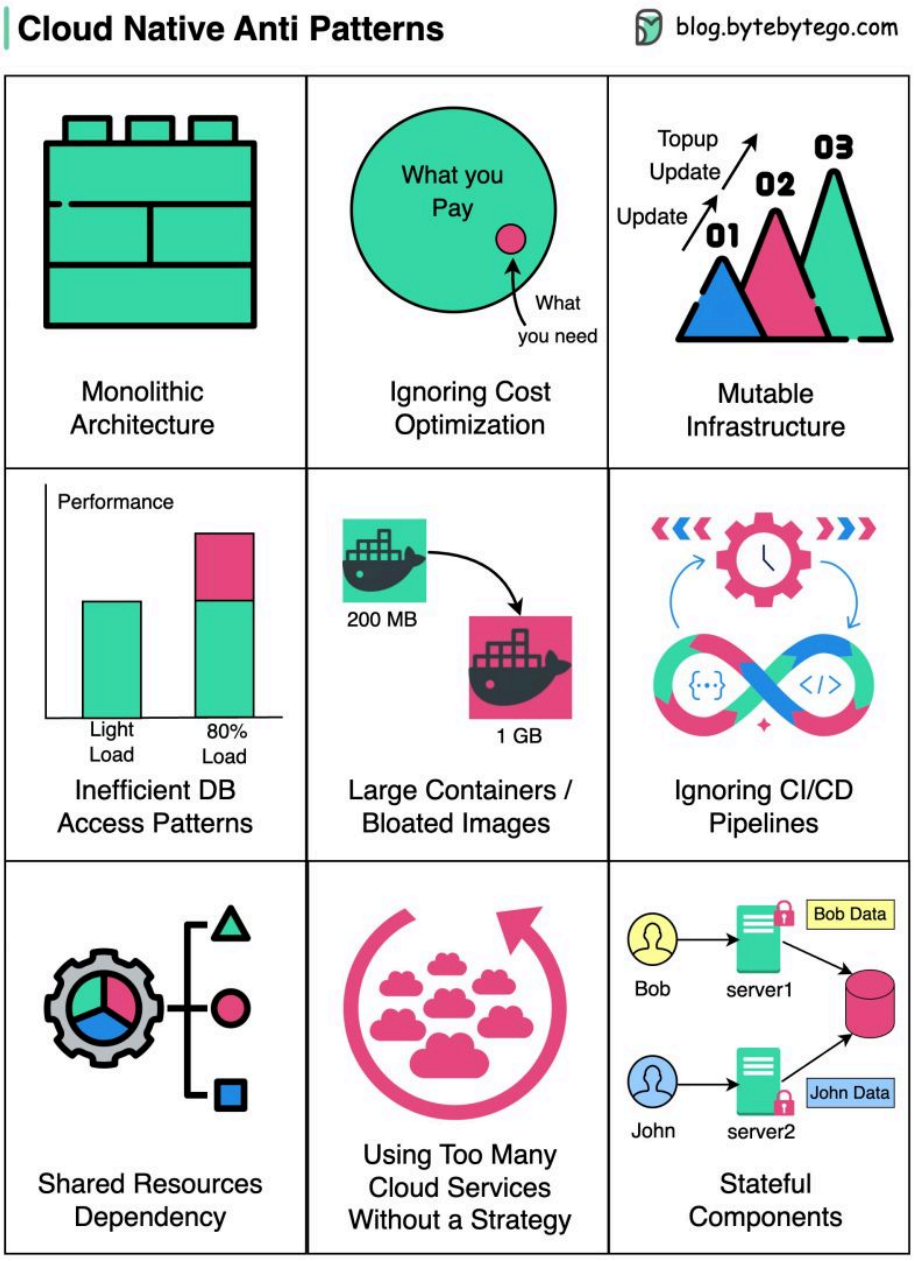
Cloud Native Anti Patterns

By being aware of these anti-patterns and following cloud-native best practices, you can design, build, and operate more robust, scalable, and cost-efficient cloud-native applications.



1. Monolithic Architecture:
   One large, tightly coupled application running on the cloud, hindering scalability and agility

2. Ignoring Cost Optimization:
   Cloud services can be expensive, and not optimizing costs can result in budget overruns

3. Mutable Infrastructure:
- Infrastructure components are to be treated as disposable and are never modified in place
- Failing to embrace this approach can lead to configuration drift, increased maintenance, and decreased reliability

4. Inefficient DB Access Patterns:
   Use of overly complex queries or lacking database indexing, can lead to performance degradation and database bottlenecks

5. Large Containers or Bloated Images:
   Creating large containers or using bloated images can increase deployment times, consume more resources, and slow down application scaling

6. Ignoring CI/CD Pipelines:
   Deployments become manual and error-prone, impeding the speed and frequency of software releases

7. Shared Resources Dependency:
   Applications relying on shared resources like databases can create contention and bottlenecks, affecting overall performance

8. Using Too Many Cloud Services Without a Strategy:
   While cloud providers offer a vast array of services, using too many of them without a clear strategy can create complexity and make it harder to manage the application.

9. Stateful Components:
   Relying on persistent state in applications can introduce complexity, hinder scalability, and limit fault tolerance

Over to you:
What anti-patterns have you faced in your cloud-native journey? How did you conquer them?