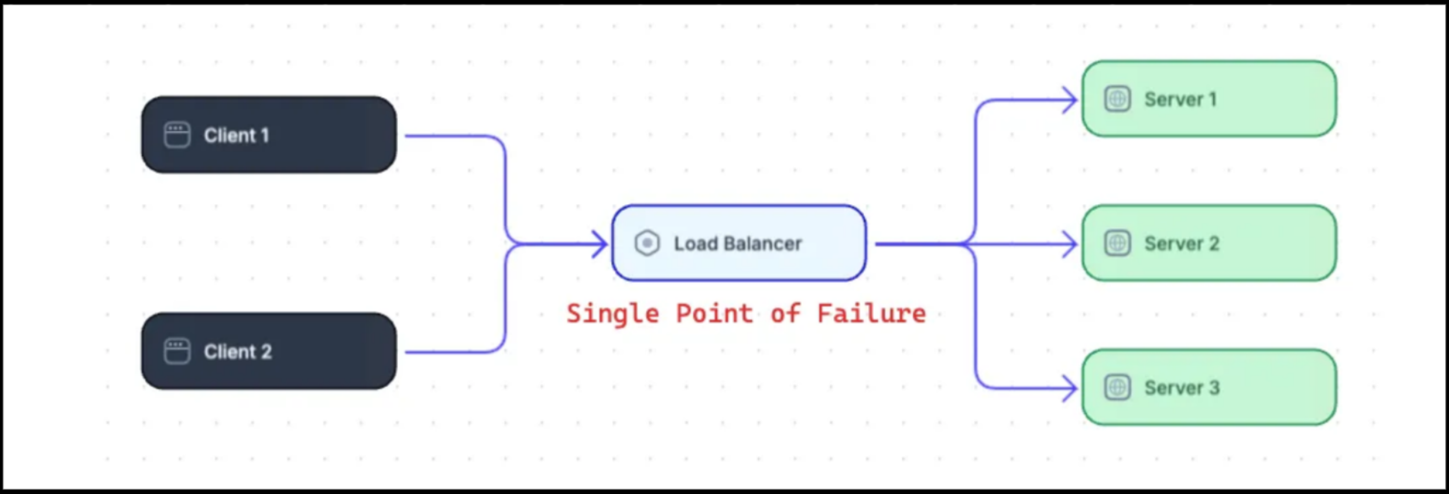
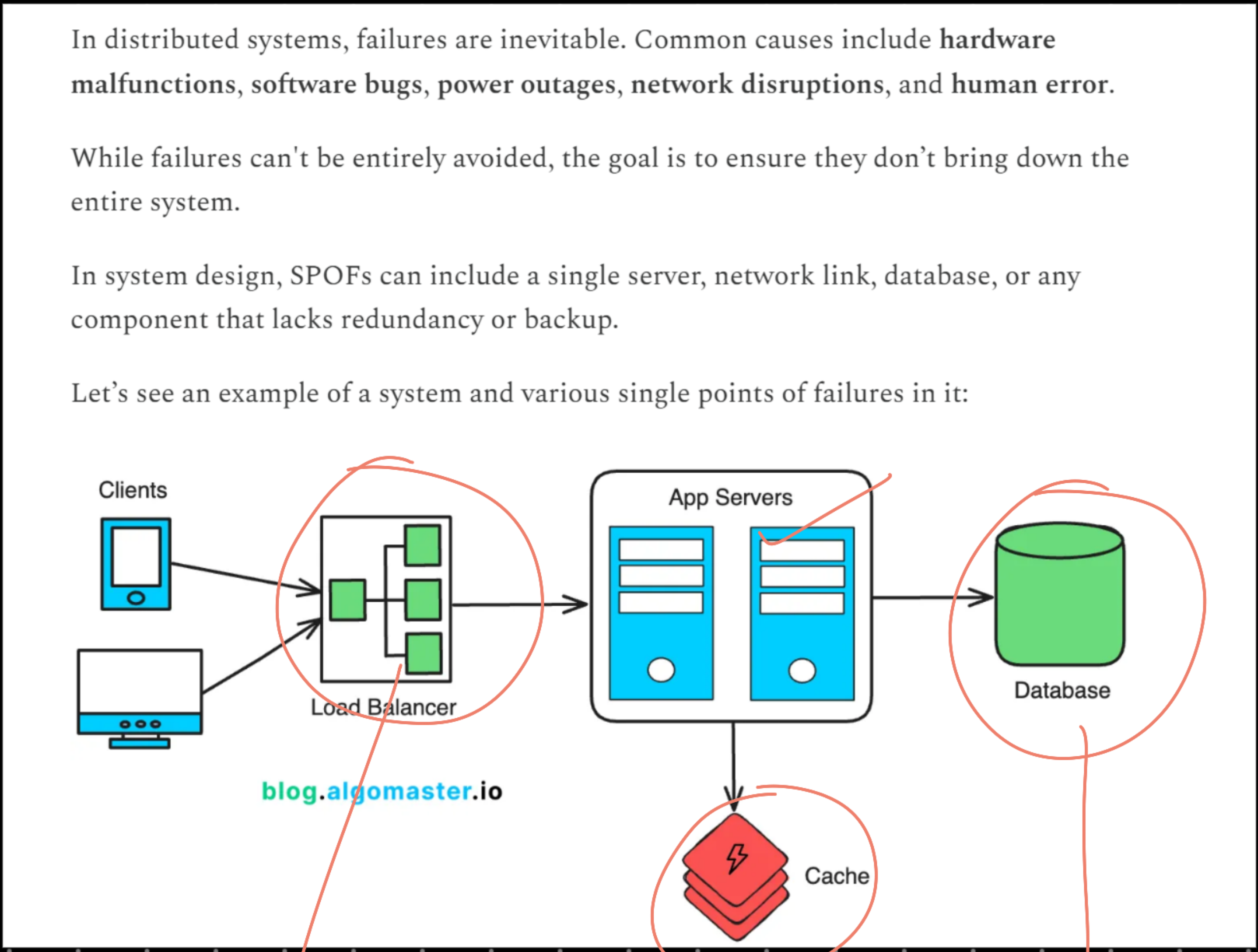
[SPOF]

↳ Single point of failure

↳ Component of system that can bring down whole system

DB loss User loss Revenue loss





In distributed systems, failures are inevitable. Common causes include **hardware malfunctions**, **software bugs**, **power outages**, **network disruptions**, and **human error**.

While failures can't be entirely avoided, the goal is to ensure they don't bring down the entire system.

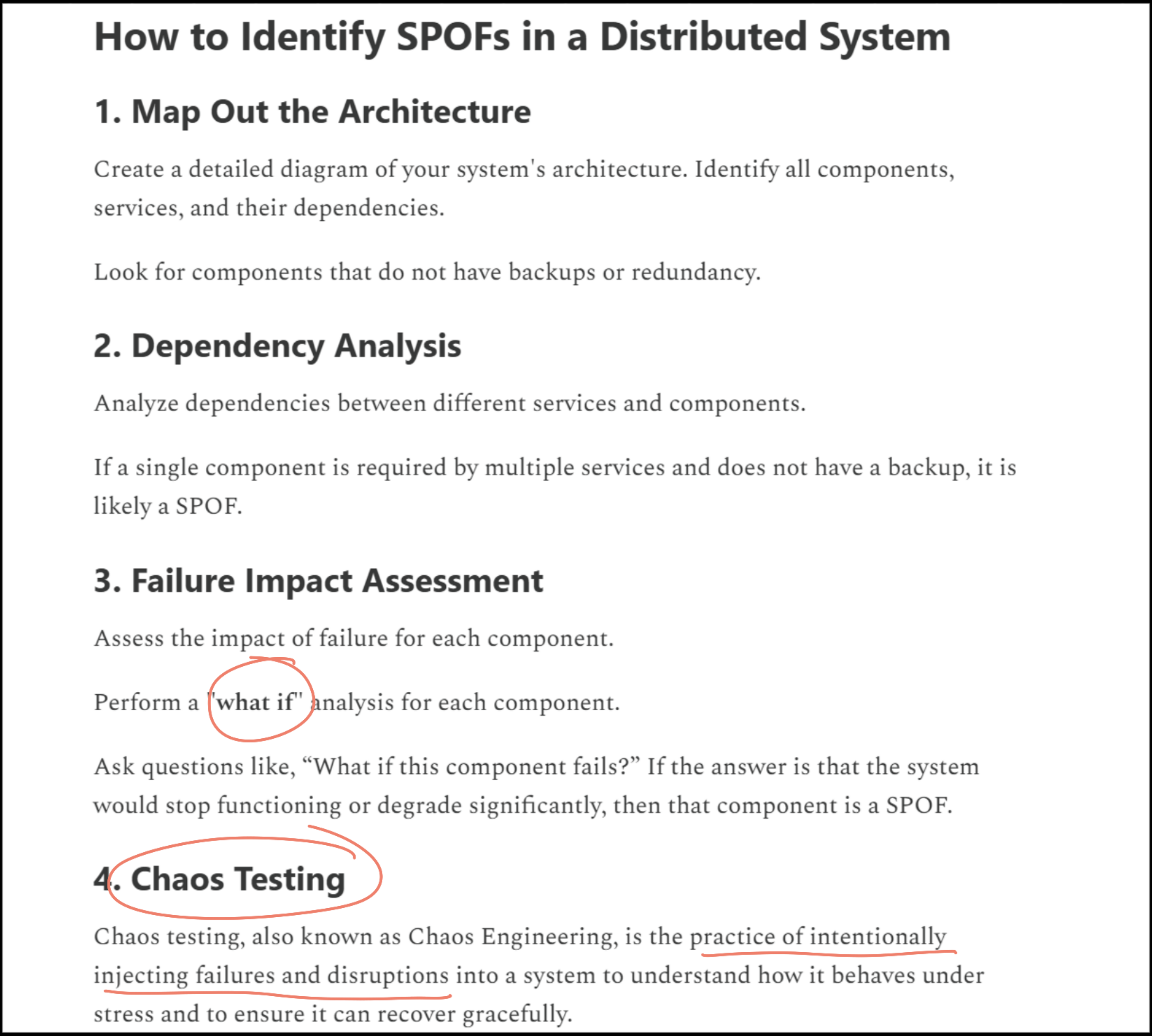
In system design, SPOFs can include a single server, network link, database, or any component that lacks redundancy or backup.

Let's see an example of a system and various single points of failures in it:

SPOF

It effect request processing time, as all query goes to DB

SPOF



How to Identify SPOFs in a Distributed System

1. Map Out the Architecture

Create a detailed diagram of your system's architecture. Identify all components, services, and their dependencies.

Look for components that do not have backups or redundancy.

2. Dependency Analysis

Analyze dependencies between different services and components.

If a single component is required by multiple services and does not have a backup, it is likely a SPOF.

3. Failure Impact Assessment

Assess the impact of failure for each component.

Perform a "**what if**" analysis for each component.

Ask questions like, “What if this component fails?” If the answer is that the system would stop functioning or degrade significantly, then that component is a SPOF.

4. Chaos Testing

Chaos testing, also known as Chaos Engineering, is the practice of intentionally injecting failures and disruptions into a system to understand how it behaves under stress and to ensure it can recover gracefully.

⇓

How to avoid ?

⇓

① Redundancy → Multiple component

② Load Balancing → distribute even load

③ Data Replication
- → Synchronous
- → Asynchronous

④ Geographic Distribution

↳ CDN, Multi-region cloud Deployments

⑤ Graceful Handling of failures

⑥ Monitoring, Alerting

↳ PagerDuty, Coralogix, Grafana

Fault tolerance → System Ability to handle errors