2.Component List

1)Nginx

* You can upgrade on it without any downtime
* It acts as a robust and cheap load balancer
* It reduces wait time when loading a website

2) Spring Boot fat JAR

* The Jar files contain application dependencies , this simplifies deployment procedures
* It is easy to launch and exit the apps that are using Spring Boot

3) Docker

* It is a container platform that ensures security and stability across applications
* Each container runs independently and will not interfere with other containers in shared environment

4)Redis

* This is cache that helps to reduce database access and hence reduces traffic

5)Prometheus

* Stores and retrieves times based metrics effectively
* No fees
* Active community support
* Freedom of customization

3. Concurrency Model

1)API pods :

Assuming each pod handles ~50 RPS

We need at least 10 pods for 500 RPS

Extra 2-3 pods will be required for buffer and redundancy

2)Load Balancer

It will do Round -robin distribution across the healthy pods and also check health of each pod.

4. Failure Story

If an API pod crashes , the load balancer performs its duty by routing to other healthy pods.

If Redis fails , system will read from database and log a warning.

5. Scaling Story

To ensure smooth load handling during peak times and cost efficiency during idle times we can do following

Horizontal Pod Autoscaler rule

1. If Cpu usage >70% for 2 mins then add a pod
2. If Cpu usage <30% for 5 mins then remove a pod

6. Metrics and Alerts

Key metrics

* API latency
* Redis cache hit ratio
* Pod Cpu usage

Alert example

If P95 latency > 400 ms for 5 mins trigger alert to Slack