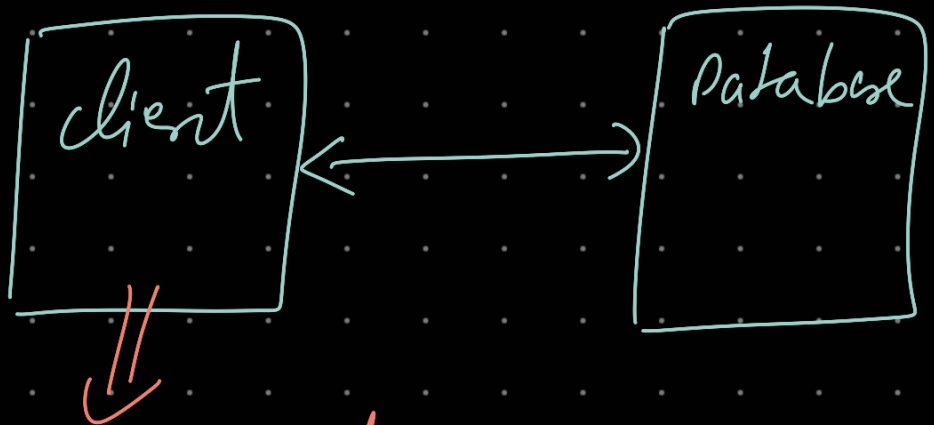
[Microservices]

↳ What, why and when to use

* Simplest way to create a full stack application

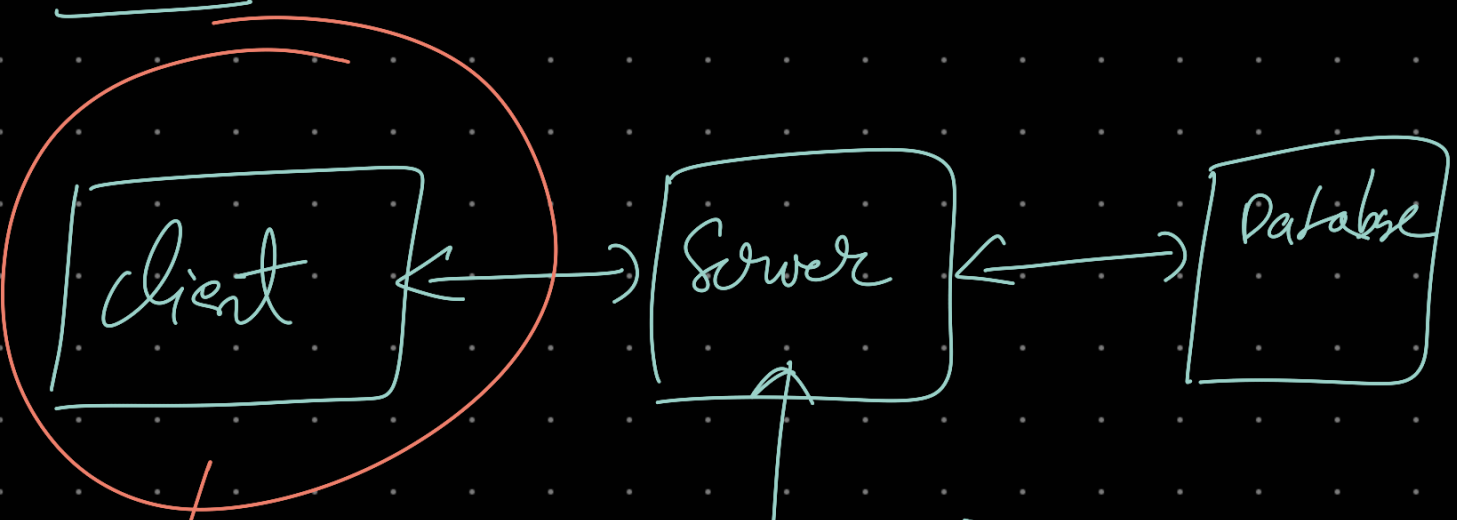
2-tier architecture (very old)



This should know DB url, secrets and how connect and query

↳ Highly risk and not scalable

3 tier architecture



No idea about database

only for frontend

This connect with database

All processing Logic

* More Secure, Robust, Scalable

N tier → Includes more layers

API gateway, Cache etc.

Distributed system vs N-tier

Monolithic

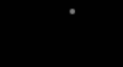
Huge system containing whole code of a system

Eg → Ecommer App

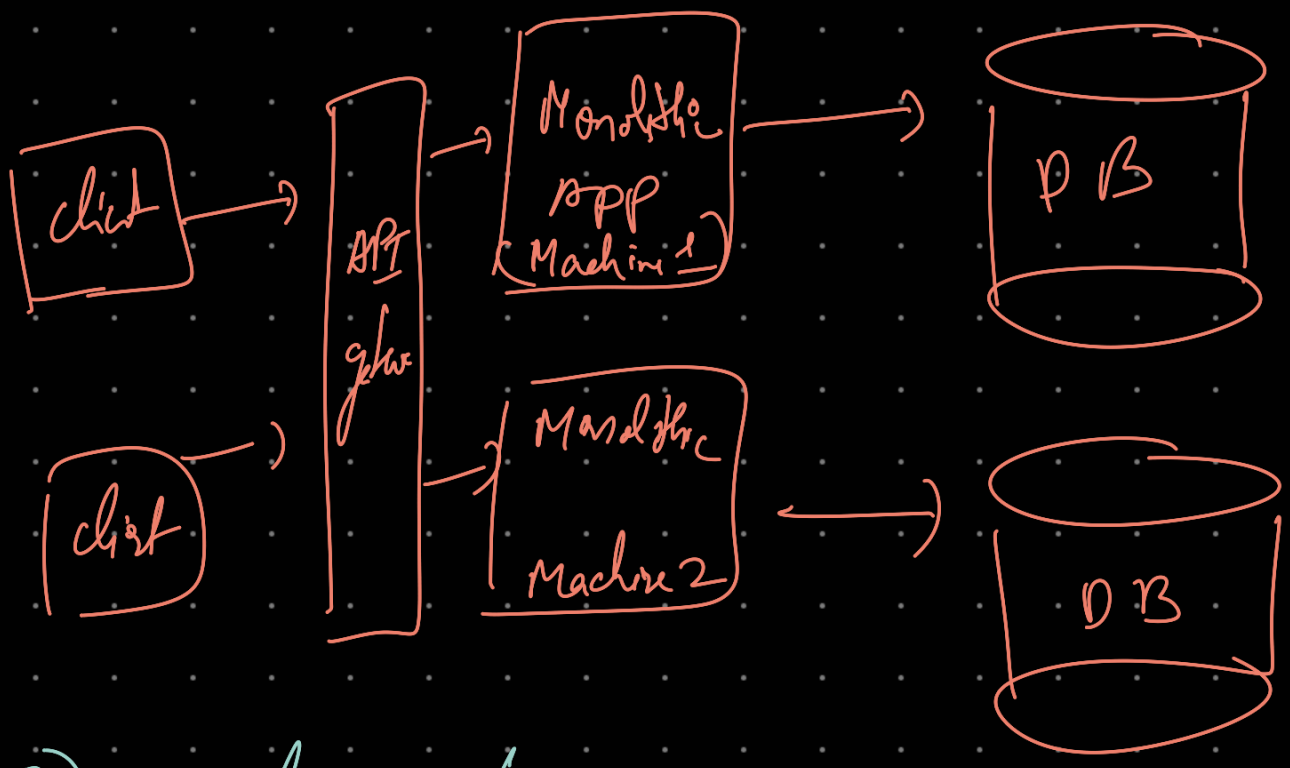
- cart
- order
- payment
- catalog

→ All part of gigantic codebase

Monolithic doesn't mean 1 machine



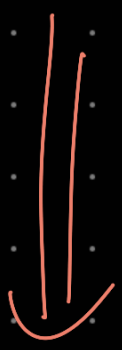
Monolithic Can be vertically and horizontally scale



1. Useful for small team and System
2. 1 repo contain whole code
3. One small change will need deployment of whole code

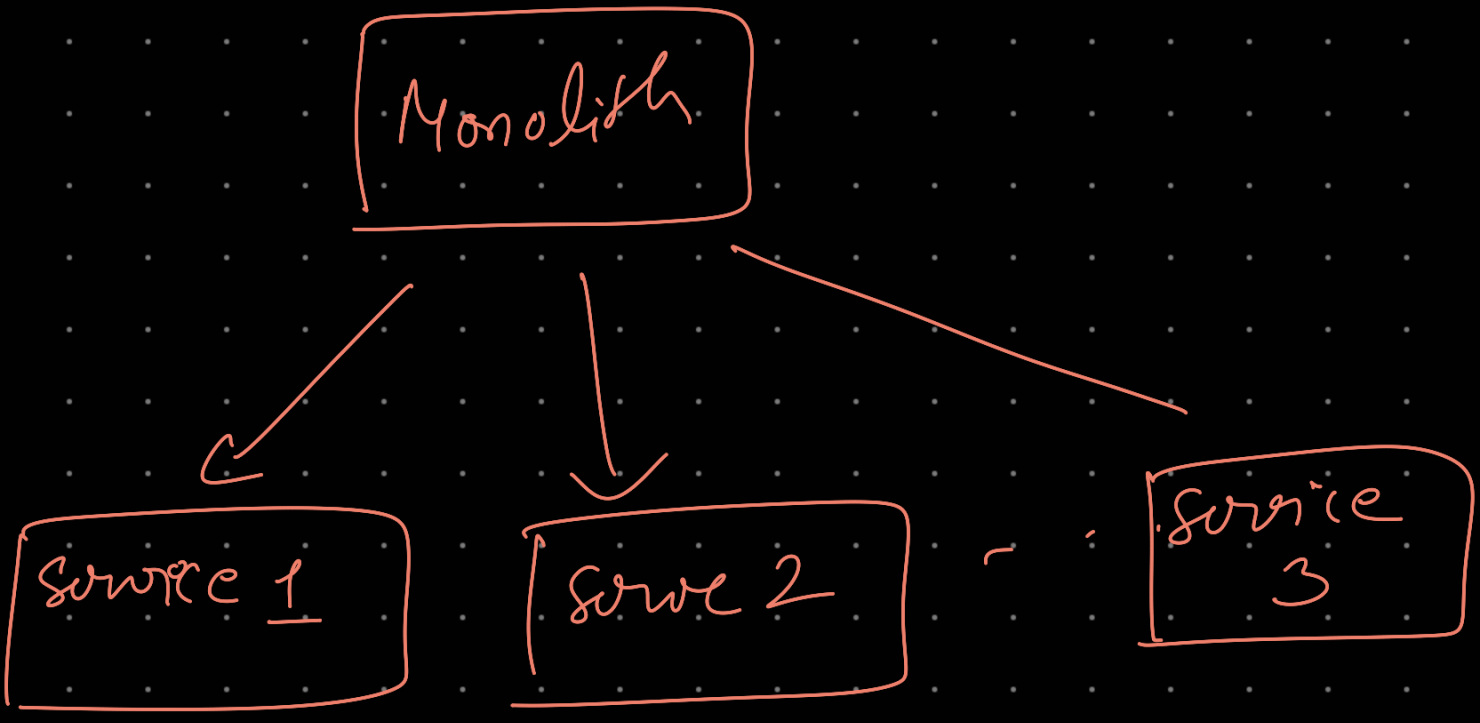
④ Understanding Large code base for beginner will be hard.

⑤ Single point of failure



How to solve this ??

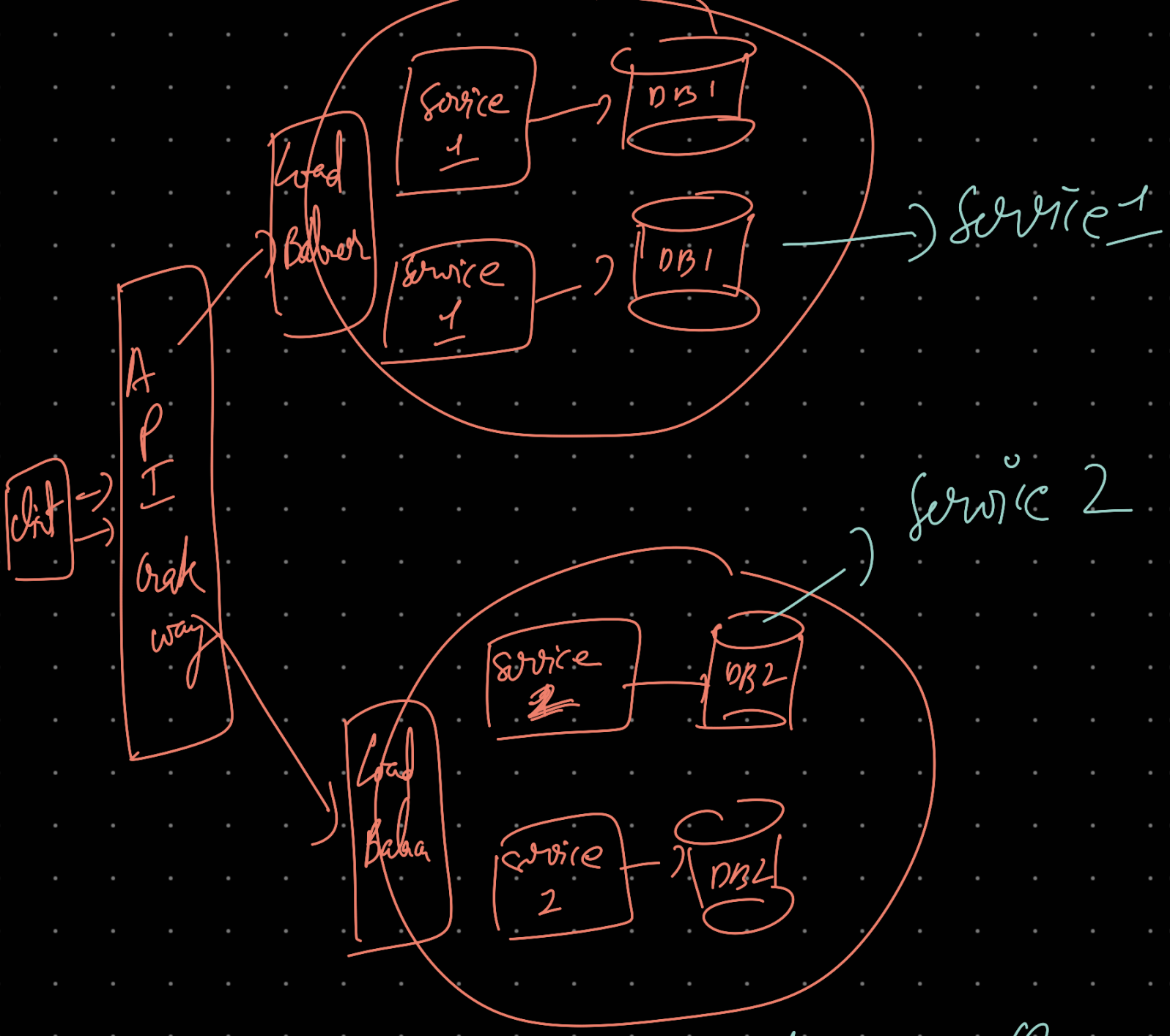
Microservices



* Multiple Repo for each service

Eg → Cart service
Order service
Catalog service

* Each service can be horizontally and vertically scaled
* Each will have its own Database
* For client, it may be abstracted
* No SPOF
* Not easy to design
* Distributed system are hard to design



* Each service is independently scaled, coded and maintained.

Monolith → Cart → Order
(function call)

Microservice → Cart ⟶ Order

↳ Network call

* Each service can be written in any language
  ↳ Polyglot

* Debugging in Microservice architecture is hard

* Complex deployment pipeline

* Move from Monolithic → Microservice where team grows not users

Can be scaled in monolithic