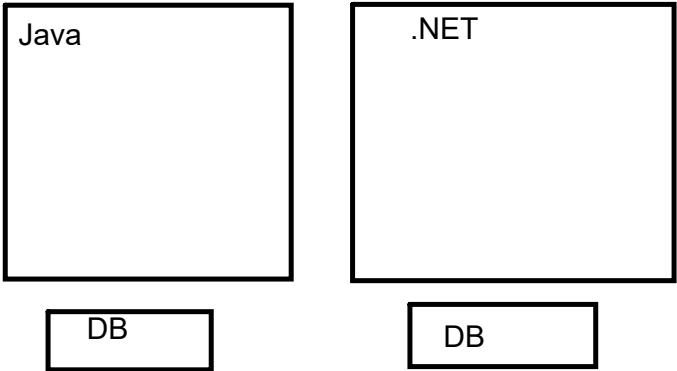
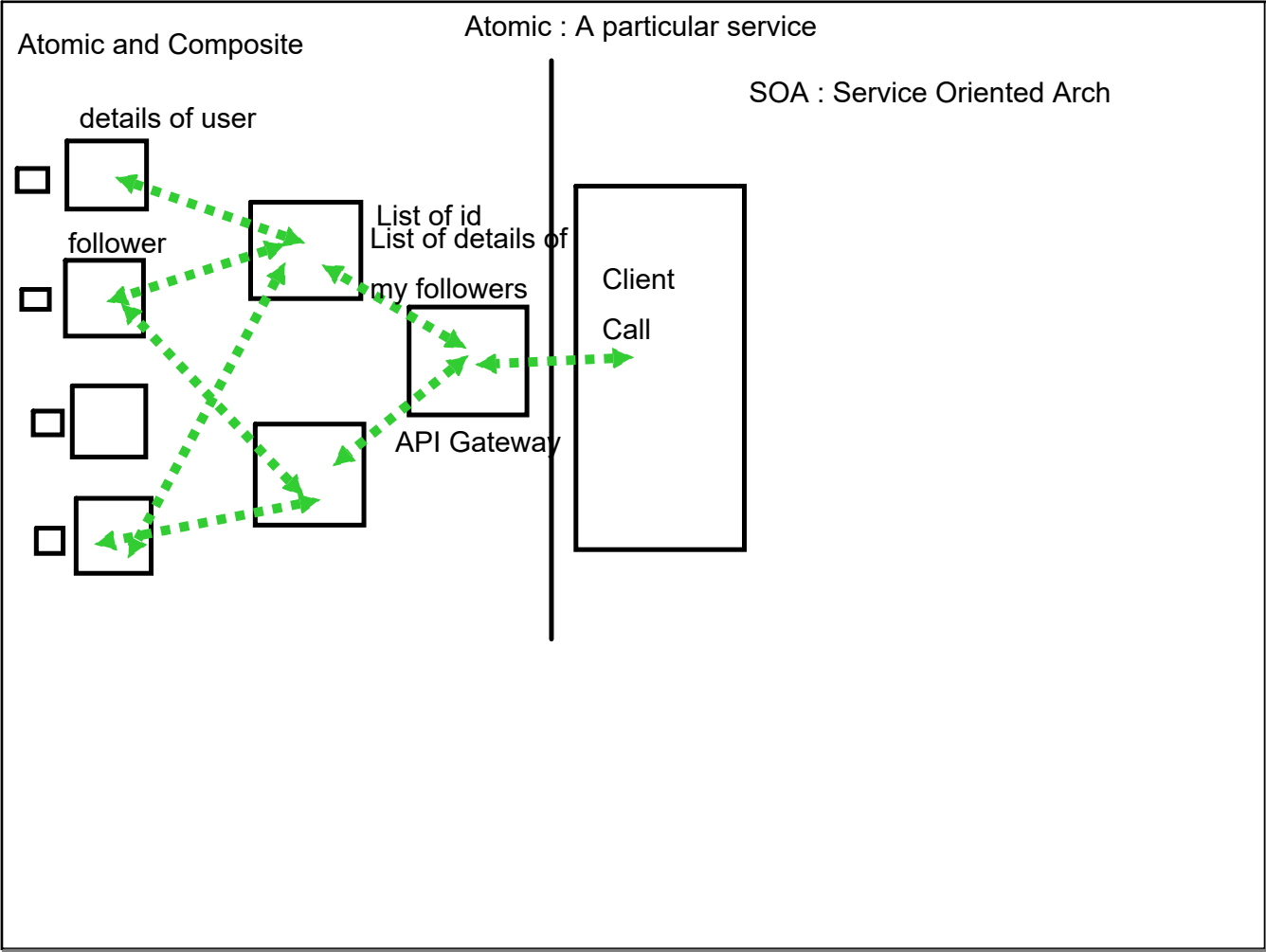


Micorservices

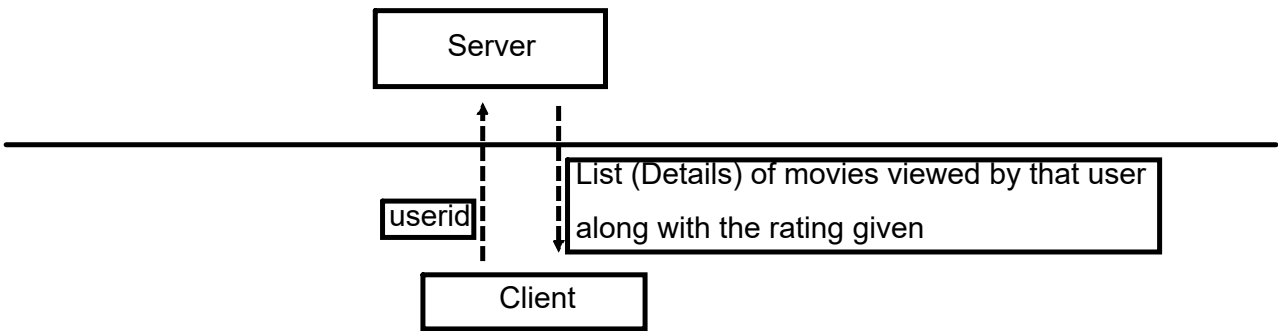
- Microservice
- # Modularity
- # Use technology
- # Single MS drop
- # Dynamic Scale up/Scale Down

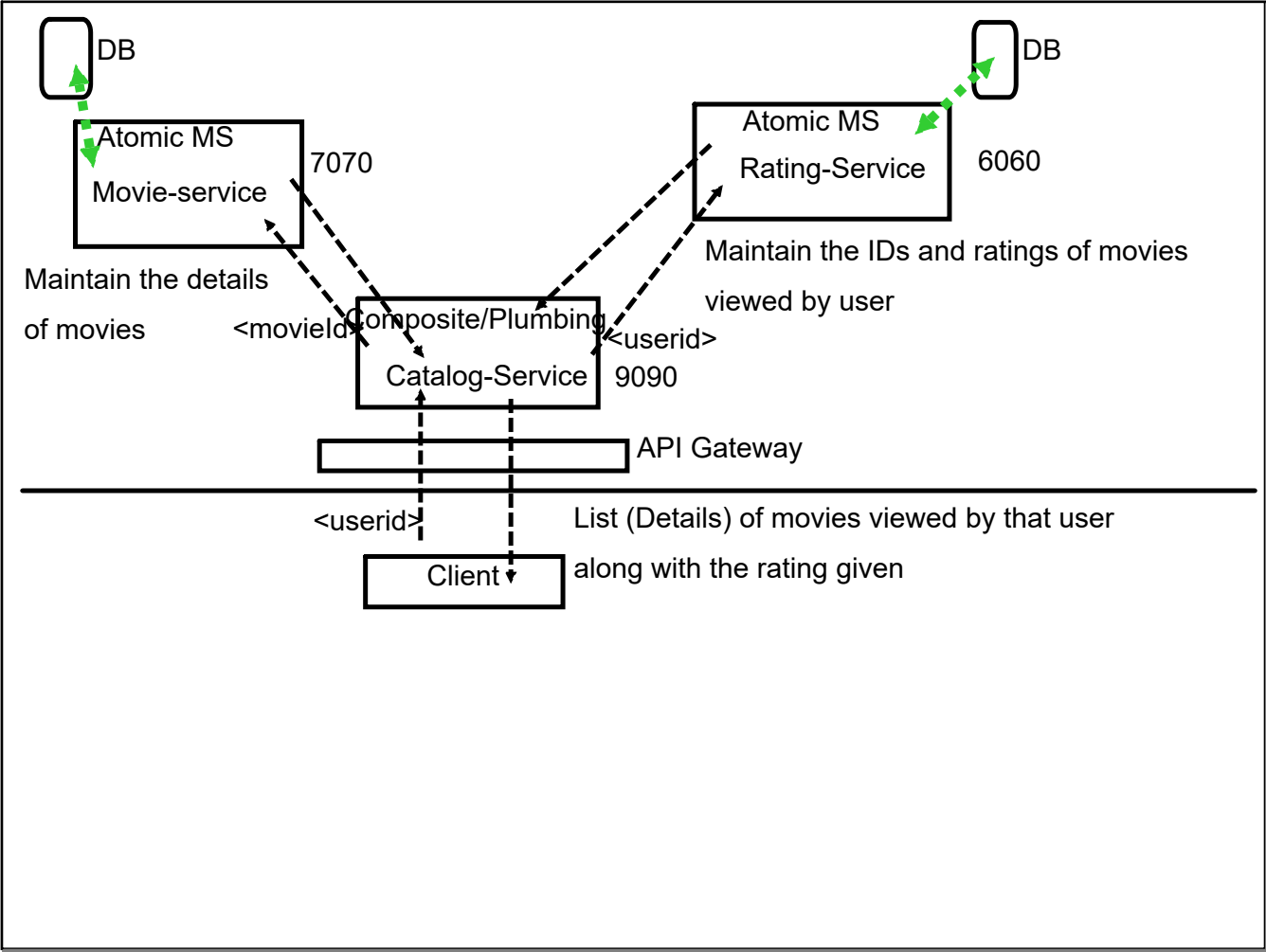
Independent/Autonomous

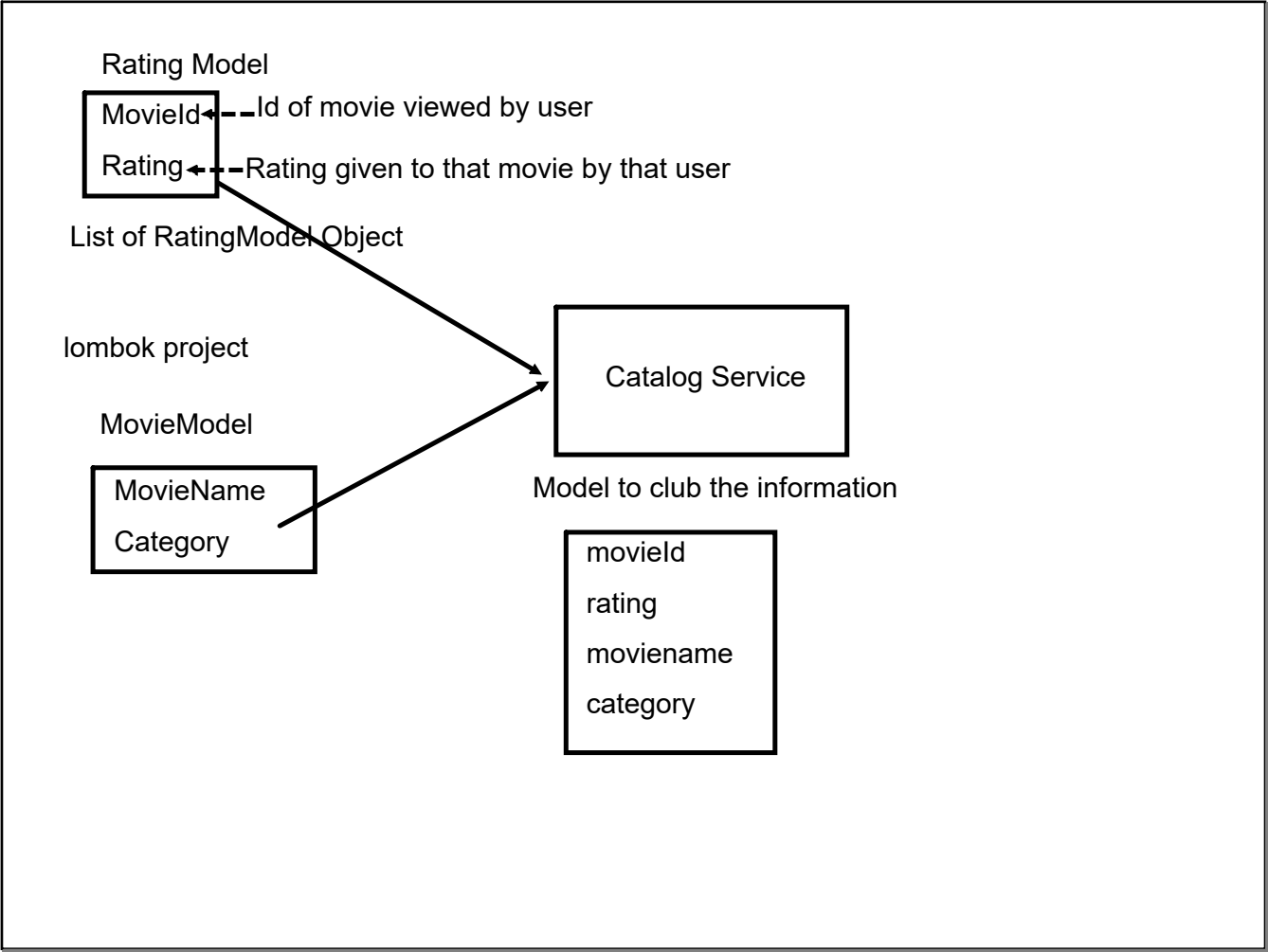


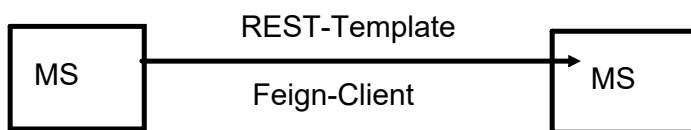


Use Case:  
# Movie Streaming portal









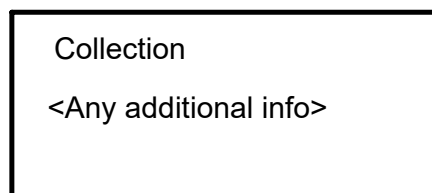
Best Practice:

1. Should never return a list/collection directly

# if list is huge, it creates problem in streaming back to calling service

# it will stop us to update/add any additional info to associate in future

Model class



Naming Server/Discovery Server

=> Eureka Server

1. no need to know the low level of MS (IP/PORT)
2. will auto fetch one of the instances of MS on demand

Feign-Client : add dependency

=> outsource URL management (HTTP management)

=> configurable

=> abstract

=> Add additional cloud tool

Feign Client:

Works like a proxy

# Create interface reflecting the http connection details of other MS (configuration file)

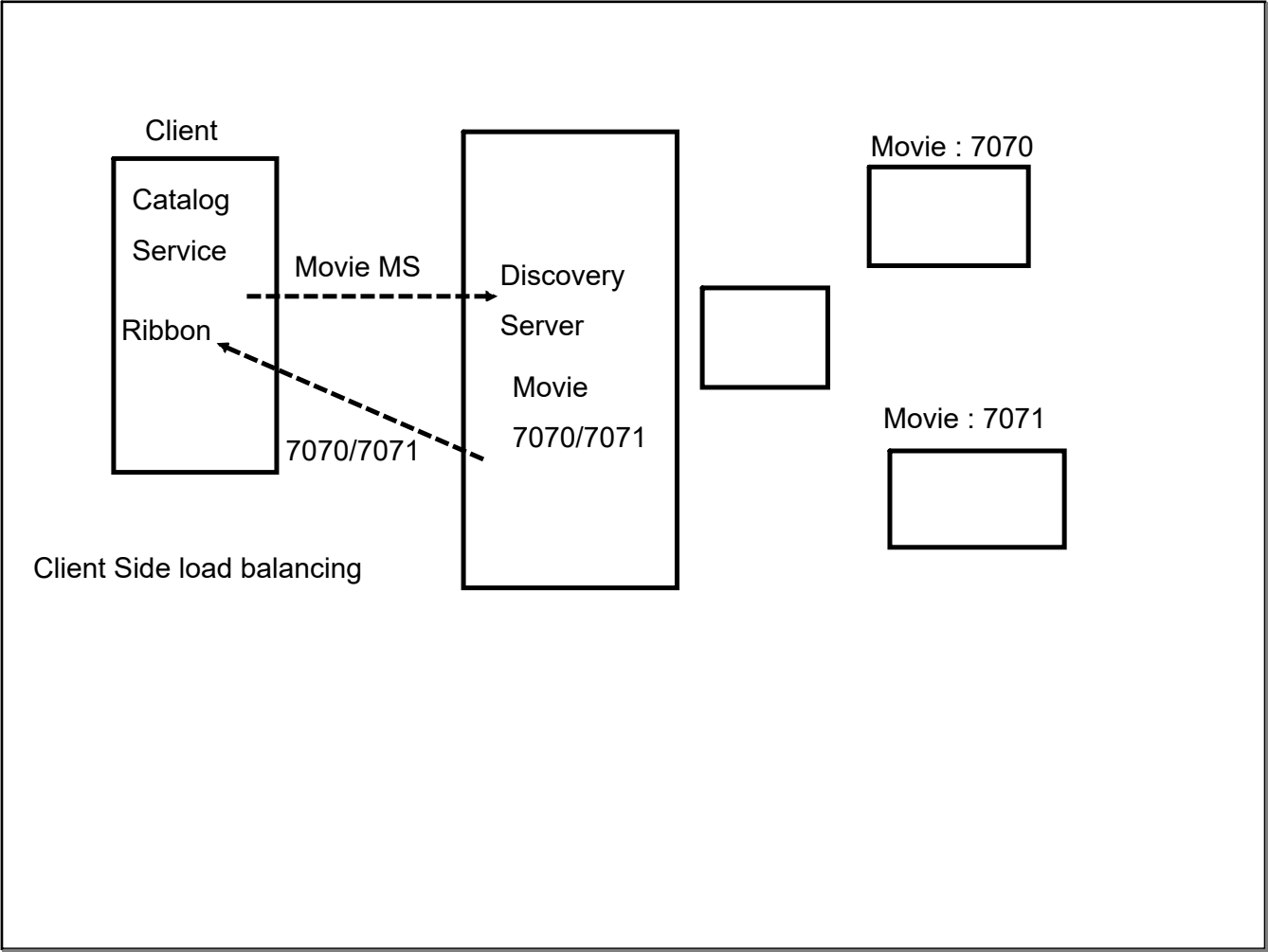
# for every MS separate interface

# Register all proxies with spring boot application

Load Balancing : Ribbon (Client Side load balancer)

# Add dependency





Best Practices

: instead of returning raw data : wrap it in ResponseEntity

