Micorservices
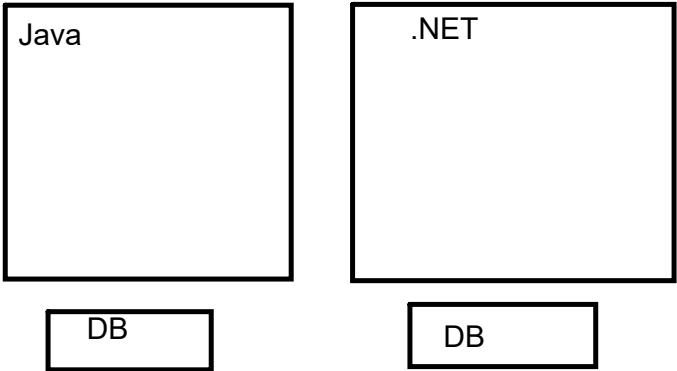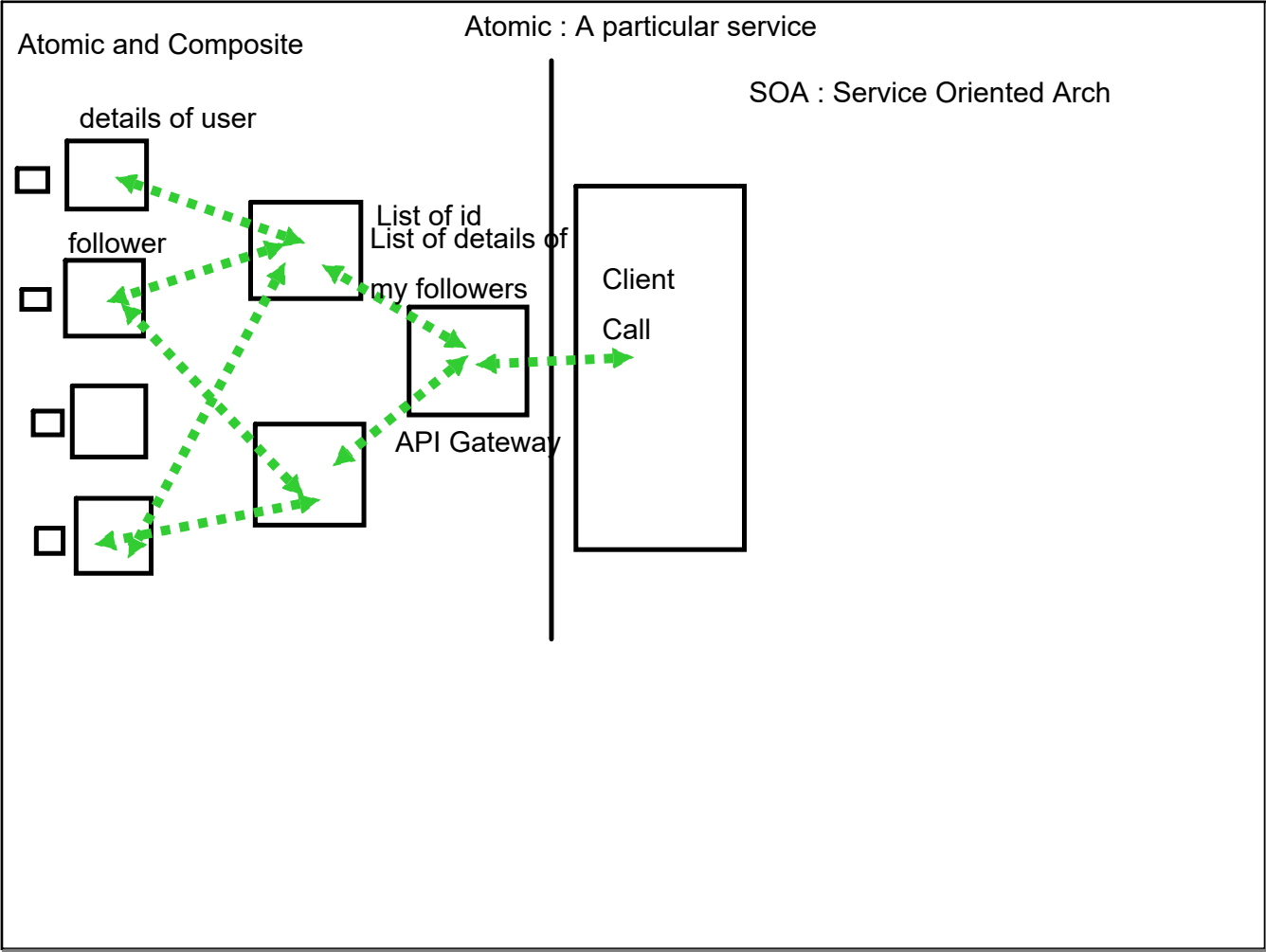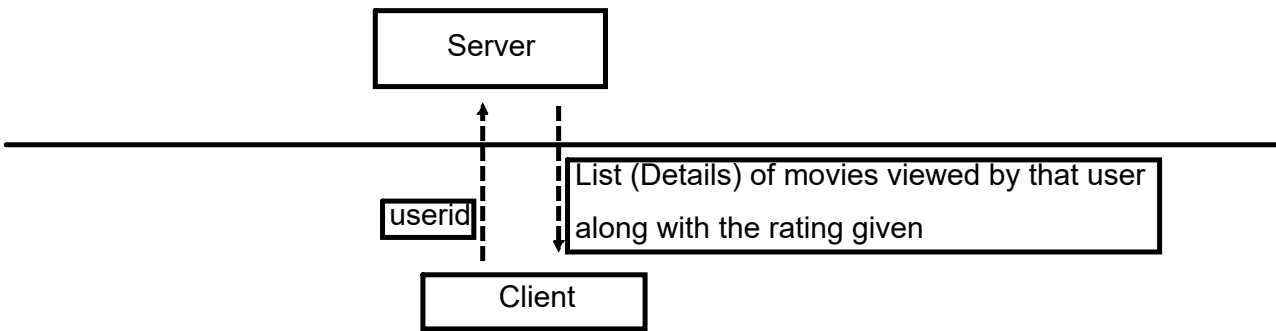
Microservice

Modularity

Use technology

Single MS drop

Dynamic Scale up/Scale Down

Independent/Autonomous

Java

.NET

DB

DB

Micorservices

Atomic and Composite

Atomic : A particular service

SOA : Service Oriented Arch

details of user

List of id
List of details of
my followers

follower

Client

Call

API Gateway

Use Case:

Movie Streaming portal

Server

userid

List (Details) of movies viewed by that user along with the rating given

Client

Micorservices

DB

Atomic MS

Movie-service

7070

Maintain the details

of movies

<movieId>

Composite/Plumbing

Catalog-Service

9090

API Gateway

<userid>

Client

DB

Atomic MS

Rating-Service

6060

Maintain the IDs and ratings of movies

viewed by user

<userid>

List (Details) of movies viewed by that user

along with the rating given

Micorservices

Rating Model

| MovieId | ◄ – – Id of movie viewed by user |
| Rating | ◄ – – Rating given to that movie by that user |

List of RatingModel Object

lombok project

MovieModel

| MovieName |
| Category |

Catalog Service

Model to club the information

movieId

rating

moviename

category

Micorservices

```
┌──────┐        REST-Template        ┌──────┐
│  MS  │──────────────────────────▶ │  MS  │
│      │        Feign-Client         │      │
└──────┘                             └──────┘
```
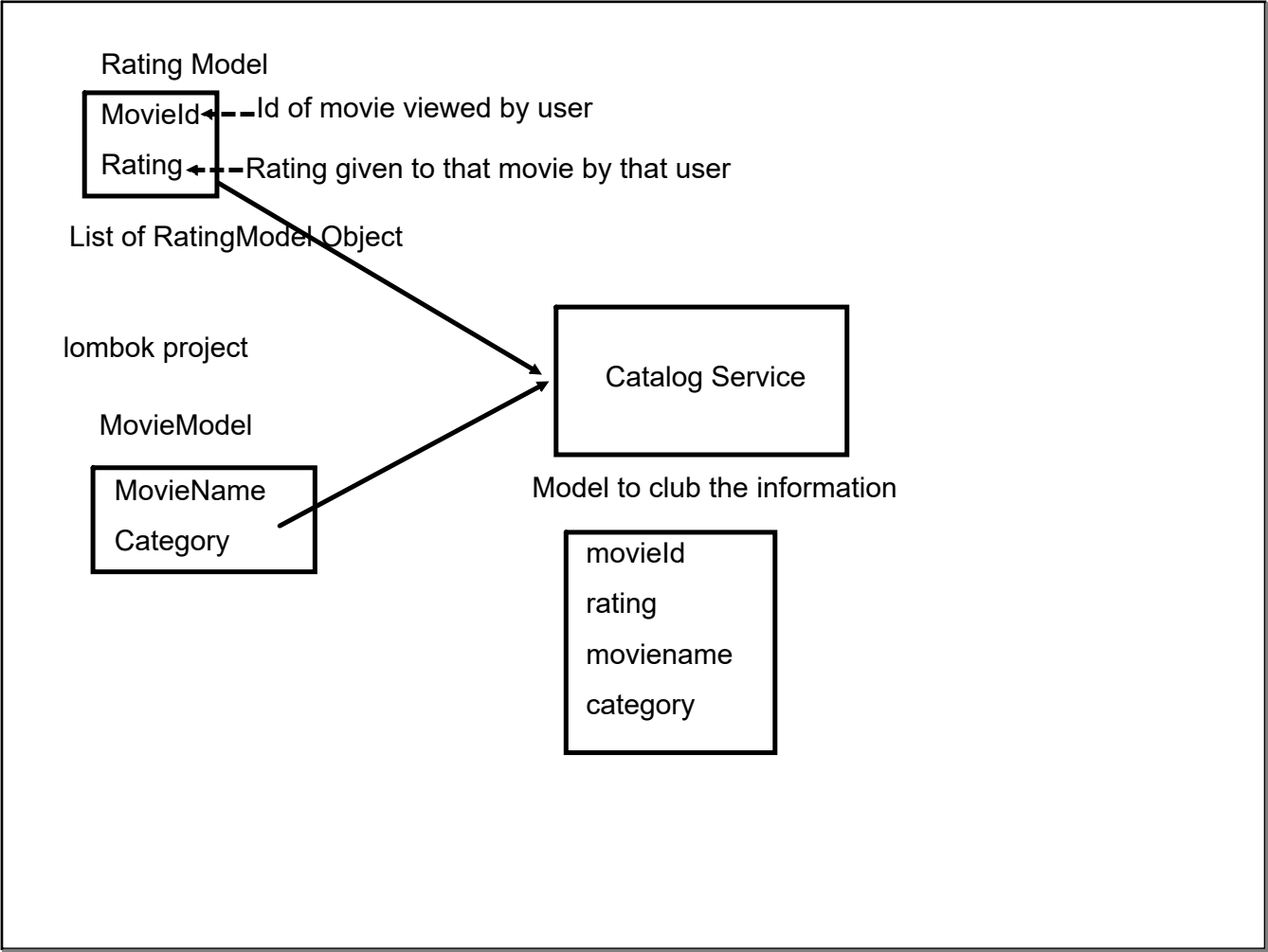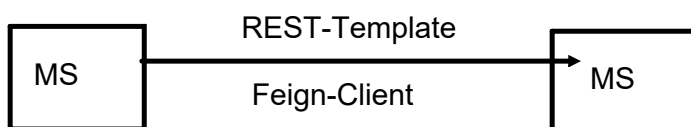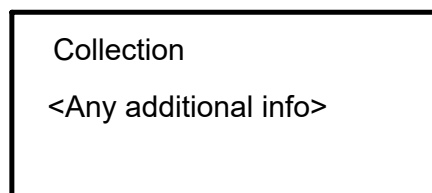
Best Practice:

    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

```
┌──────────────────────────────┐
│  Collection                  │
│                              │
│  <Any additional info>       │
│                              │
│                              │
└──────────────────────────────┘
```

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