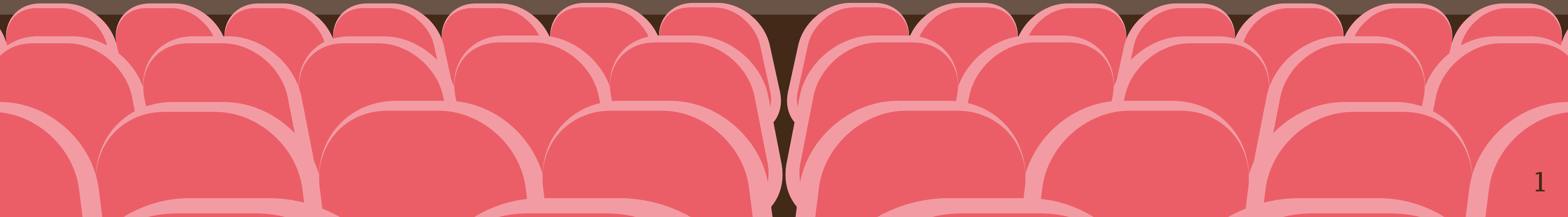
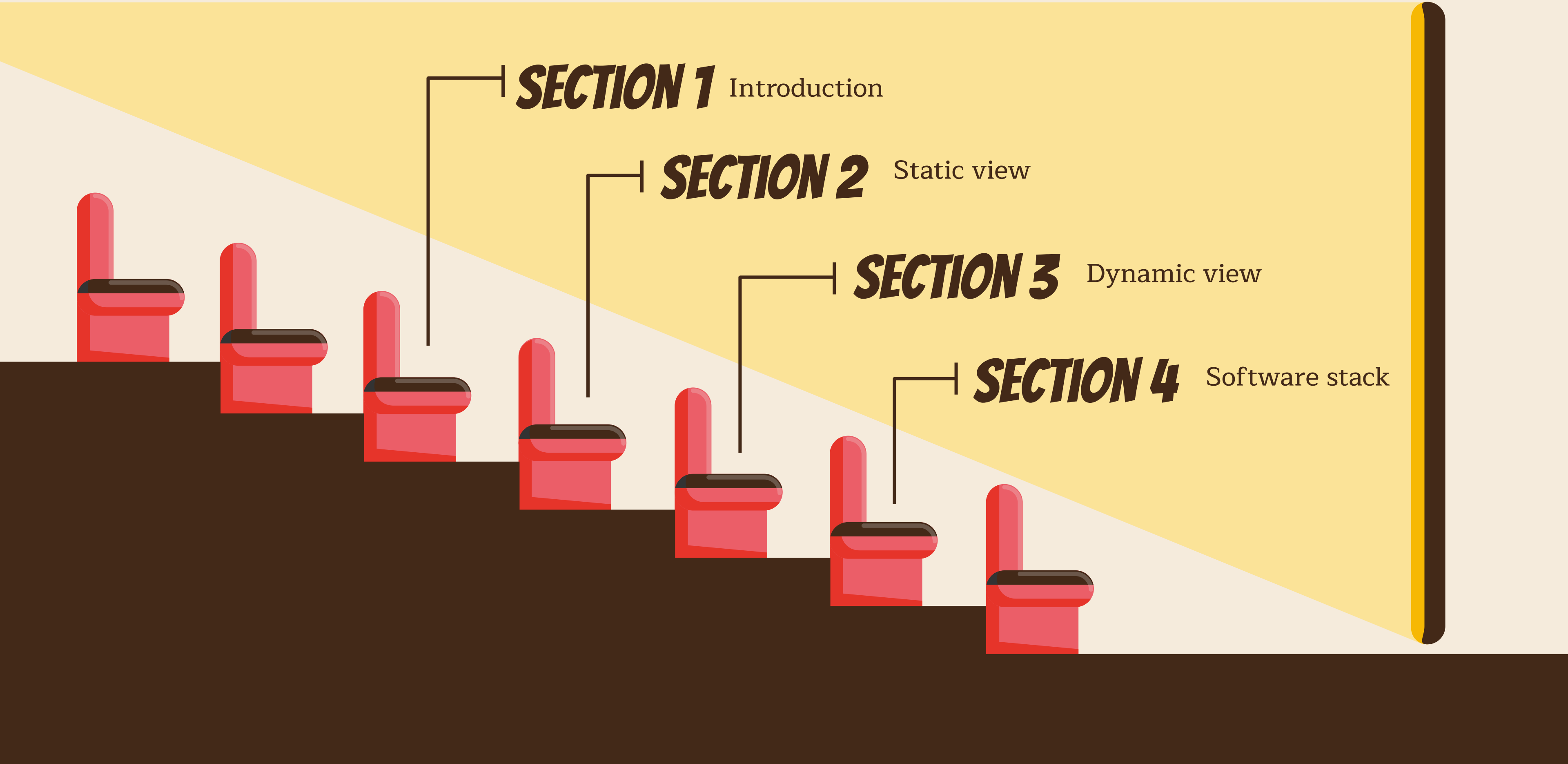
SERVICE-ORIENTED SOFTWARE ENGINEERING

PROJECT: MANGE STORE



CONTENTS OF THIS PRESENTATION



SECTION 1 Introduction

SECTION 2 Static view

SECTION 3 Dynamic view

SECTION 4 Software stack

ABOUT THE PROJECT

01

MangaWorld is designed to provide manga fans with an easy way to explore, buy, and review their favorite manga titles.

Users can enjoy a personalized experience with account management and product reviews, while admins have the capability to manage the manga catalog and user accounts.

The platform focuses on delivering a seamless and enjoyable experience for manga enthusiasts.

FUNCTIONAL REQUIREMENTS

Profile

Allow users to manage their personal information

View Manga

Allow users to browse and search for available manga titles

View Manga Details

Allow users to see detailed information about each manga, including synopsis, author, genre, and release date.

Select Categories

Allow users to choose specific manga categories to explore, such as action, romance, or fantasy.

Write Reviews

Allow users to write and share reviews about manga titles (account creation required

Admin Functions

Allow admins to add new manga titles, manage user accounts, and update the manga catalog.

Purchase Manga

Allow users to select manga volumes, add them to their cart, and complete the payment process.

Receive Confirmation

Send Purchase details to the user's email upon successful payment.

MICROSERVICE-ORIENTED ARCHITECTURE

Loose Coupling

Less dependency on each other.

Service Abstraction

Services hide the logic they encapsulate from the outside world.

MICROSERVICE-ORIENTED ARCHITECTURE

Service Reusability

Logic is divided into services with the intent of maximizing reuse.

Service Autonomy

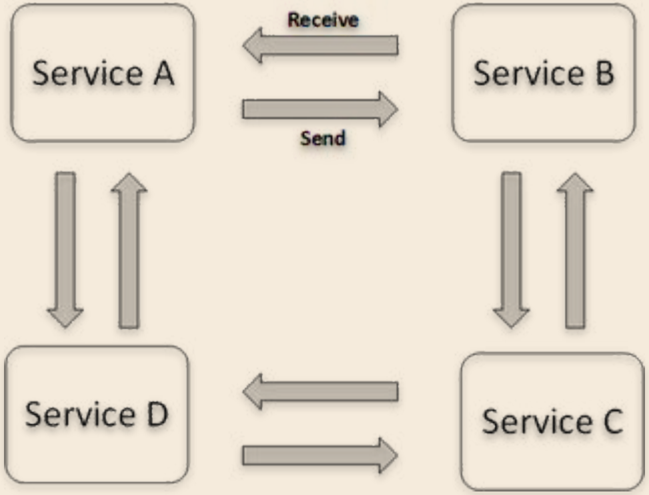
Services should have control over the logic they encapsulate.

ORCHESTRATION VS CHOREOGRAPHY

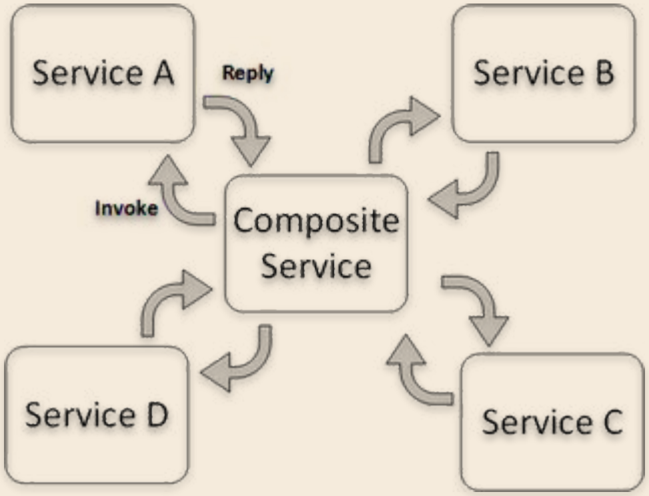
As the previous architecture suggests, the team has decided to opt for a **Choreography** as service composition pattern.

- Decentralized approach
- No Single-point-of-failure

Choreography



Orchestration



STATIC ARCHITECTURE

02

(MICRO)SERVICES

Catalog Service

Manages the catalog of manga products.

Order Service

Processes and manages user orders.

Payment Service

Manages payment transactions and gateways

Account Service

Manages user accounts and authentication, including JWT and CXF for secure communication.

Billing Service

Handles billing and payment processing.

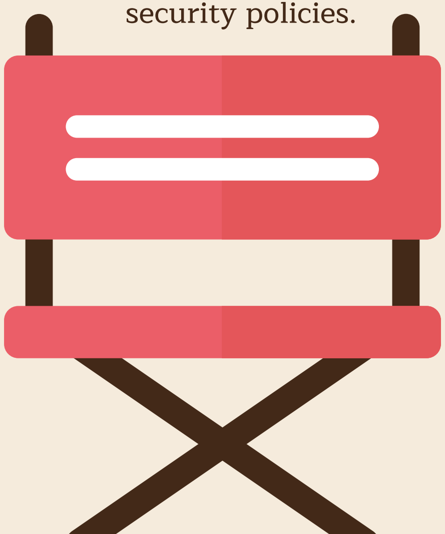
Zipkin Service

Handles distributed tracing and monitoring.

SERVICE PROXY, REGISTRY, LOAD BALANCER

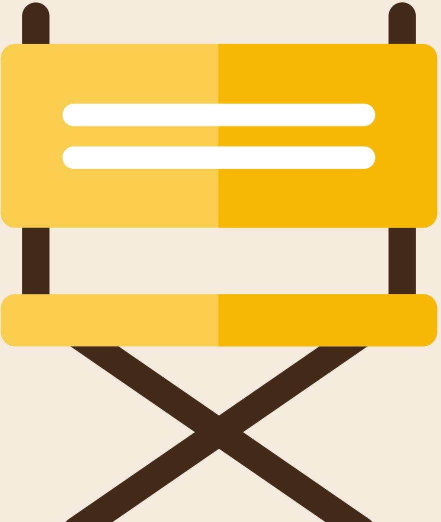
Service Proxy

Usinf NGINX We Ensures traffic is routed to the right destination service or container and to apply security policies.



Service Registry

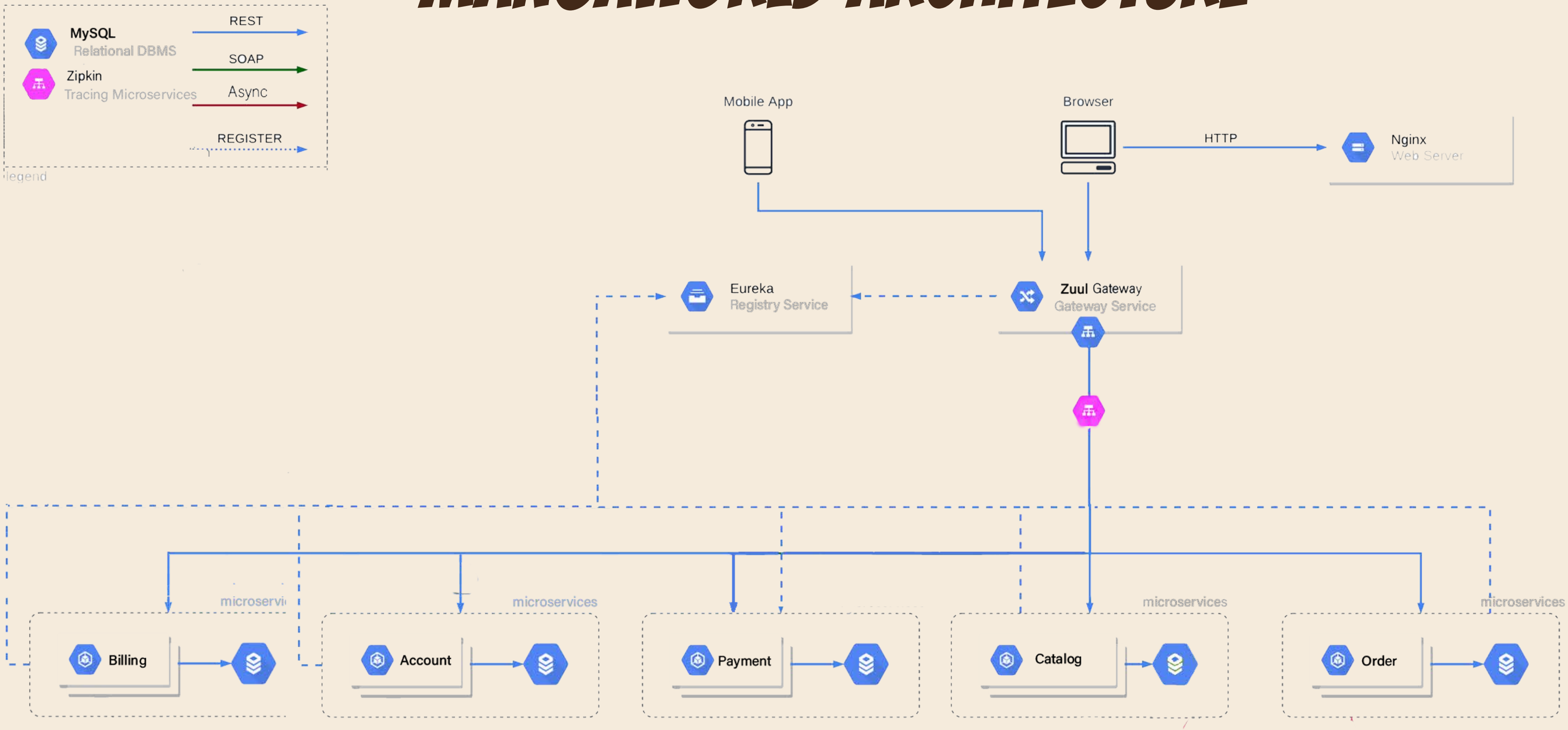
Using Consul and Eureka to Keeps track of all the available microservices in the cluster.



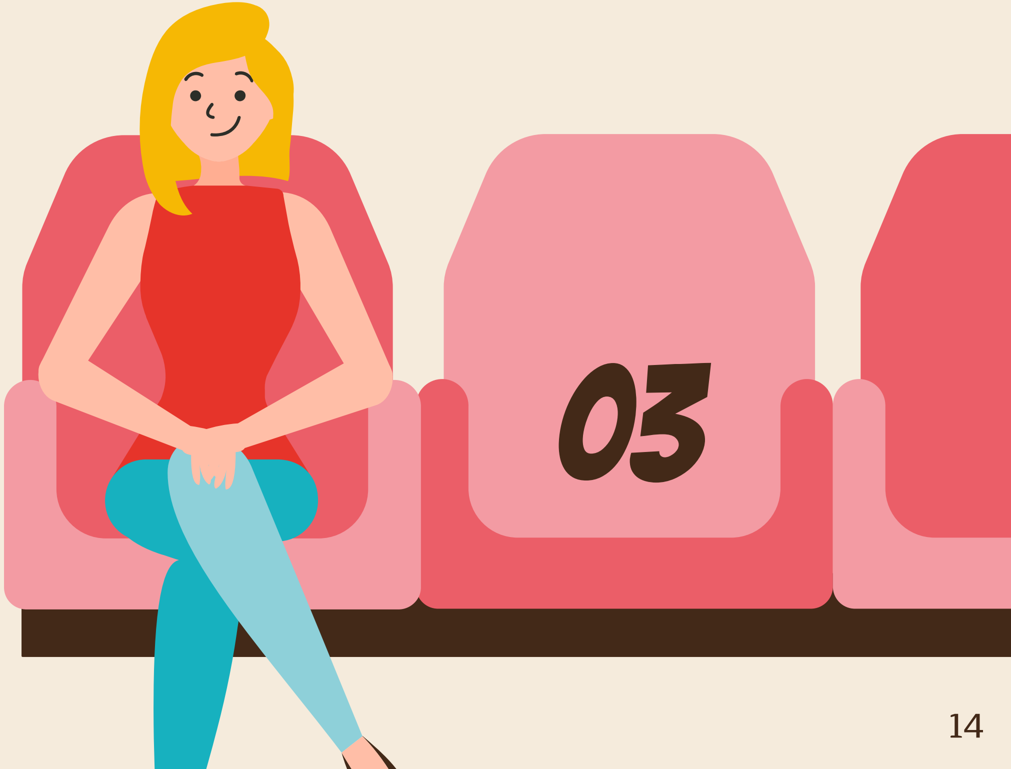
Service Load Balancer

Usinf Zuul we Distributes the working load based on the available instances and one balancing algorithm

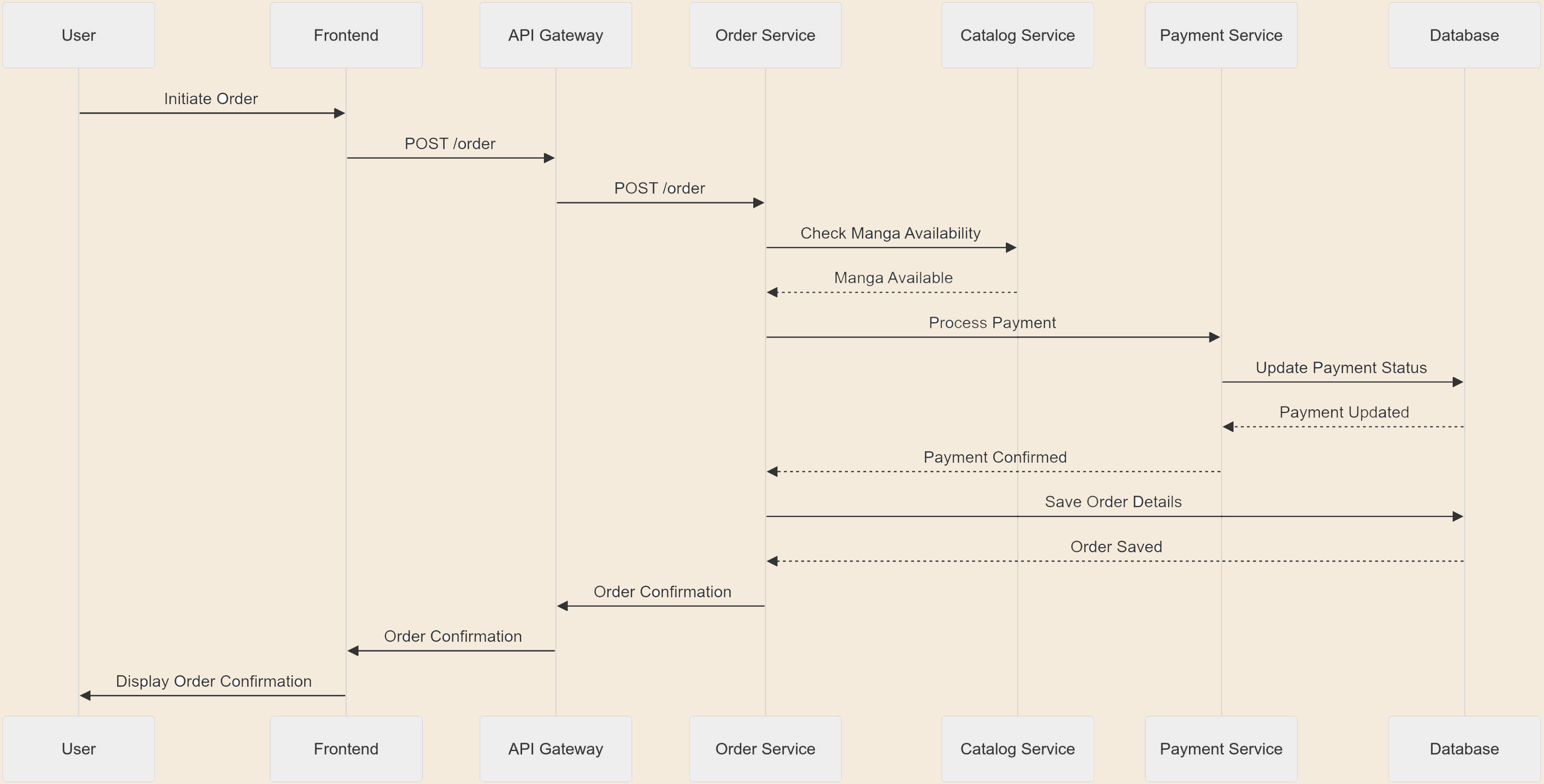
MangaWorld Architecture



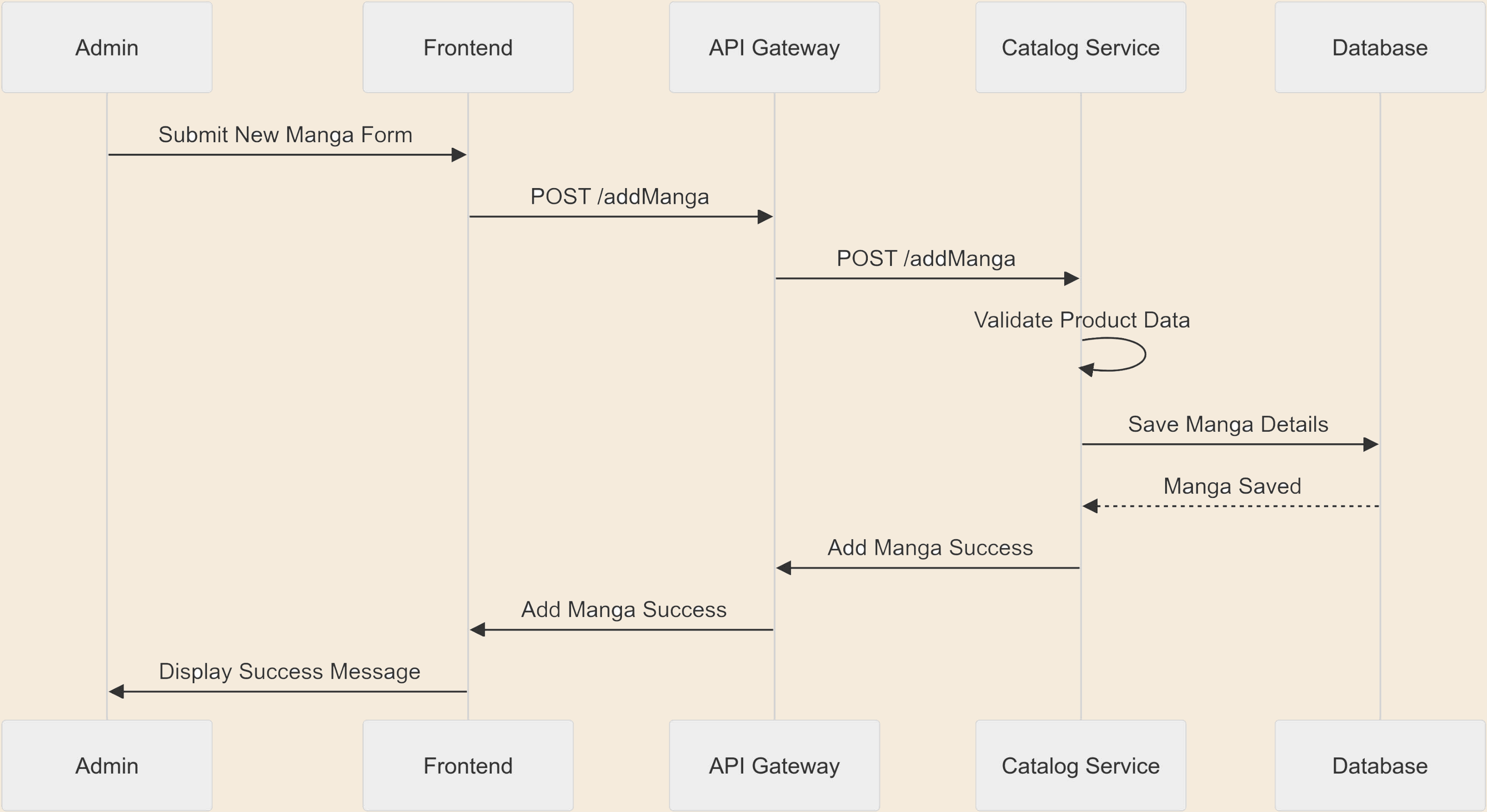
DYNAMIC VIEW



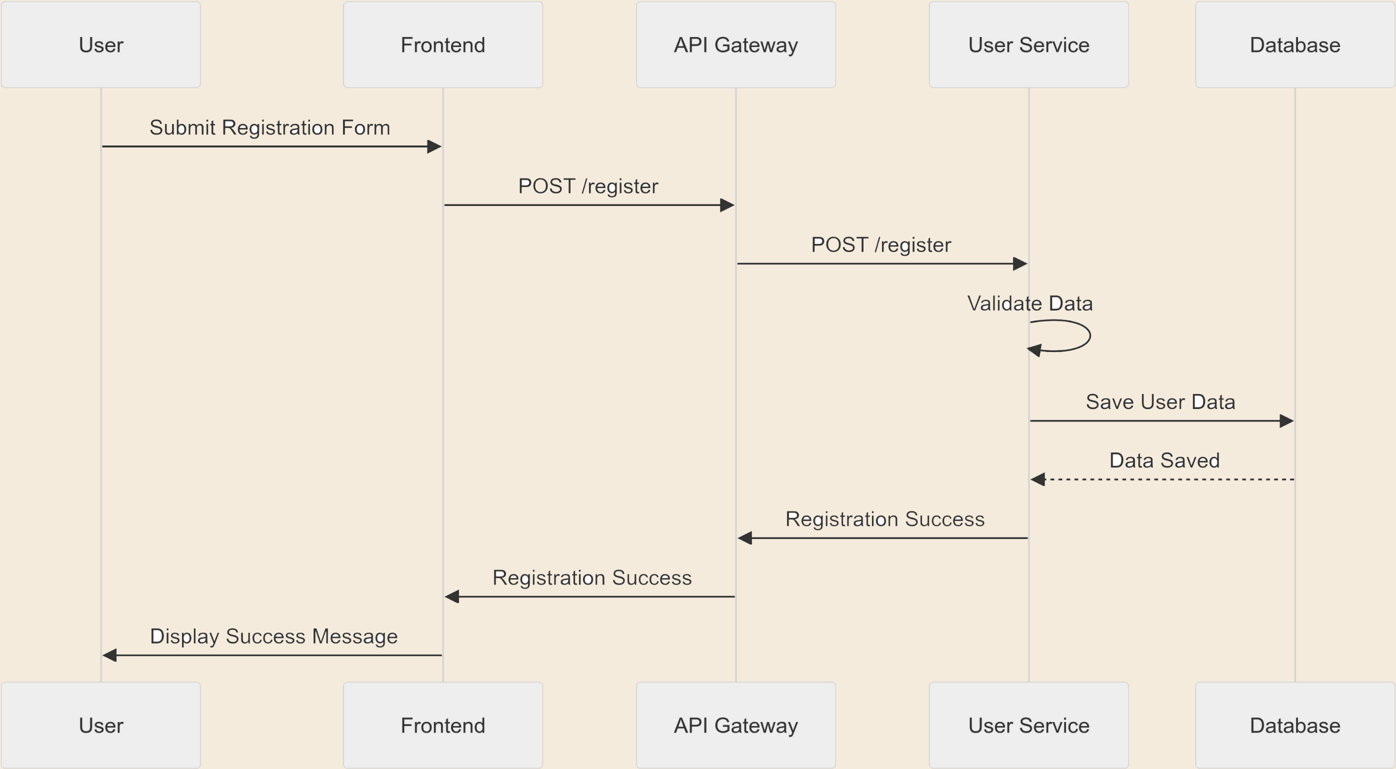
Sequence Diagram - Order Placement



Sequence Diagram – Admin adding a new manga



Sequence Diagram - User Registration



Software Stack

BACKEND

Spring Boot

Framework for Java application

Netflix Stack

Software stack for service-oriented architecture

MySQL

Relational DBMS

NGINX

Reverse Proxy

Consul

Service Registry

Zuul

API Gateway

Zipkin

Tracing Microservices

NETFLIX
OSS

NETFLIX STACK

EUREKA
Service registry

ZUUL
Service router/gateway

FEIGN
Http client

HYSTRIX
Fault tolerance and fallback factory

RIBBON
Client-side load balancer

FRONTEND

React

Js lib for building interactive web ui, focused on the ViewModel layer of the MVVM pattern

Bootstrap

Framework for building responsive, mobile-first sites

Fetch

Javascript API for accessing and manipulating parts of the HTTP pipeline, such as requests and responses

PWA

Allow an user experiences similar to native applications on desktop and mobile devices

Docker

Containerized application

Images repository:

- MySQL
- Zuul
- Consul
- Zipkin
- Nginx

Docker-compose(.yml)

- Multi-container Docker application
- Scale single web service by spawning multiple instances for load-balancing

FOLDERS STRUCTURE, MAVEN AND NPM

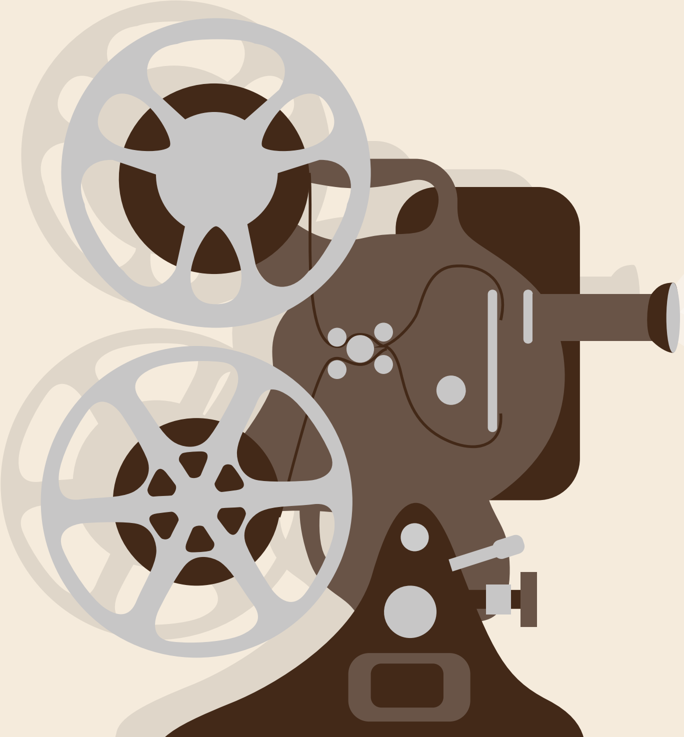
- **Three** layer **hierarchical** folders structure and pom.xml:
  a. Root pom.xml
  b. Microservices pom.xml
  c. Single microservice pom.xml

  notes: *<modules>* and *<parent>* tags

- **Archetype** generation

- **YARN** the frontend package manager

```
├──── MangaStore-Backend-Springboot
│     ├──── data
│     ├──── mangastore-account-service
│     ├──── mangastore-api-gateway-service
│     ├──── mangastore-billing-service
│     ├──── mangastore-catalog-service
│     ├──── mangastore-commons
│     ├──── mangastore-eureka-discovery-service
│     ├──── mangastore-feign
│     ├──── mangastore-order-service
│     ├──── mangastore-payment-service
│     ├──── pom.xml
├──── Mangastore-Frontend-React
│     ├──── build
│     ├──── node_modules
│     ├──── public
│     ├──── src
│     ├──── package.json
│     ├──── README.md
│     └──── yarn.lock
├──── .gitignore
├──── docker-compose.yml
└──── README.md
```



THANKS

Do you have any questions?

GitHub

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik.

Resources

- ✓ spring.io/projects/spring-boot
- ✓ spring.io/projects/spring-cloud
- ✓ netflix.github.io/spring-cloud/spring-cloud.html
- ✓ cxf.apache.org
- ✓ docs.spring.io/spring-security/reference/index.html
- ✓ docker.com
- ✓ reactjs.org
- ✓ redux.js.org
- ✓ getbootstrap.com
- ✓ github.com/Netflix/eureka
- ✓ bootstrap-vue.org
- ✓ https://github.com/elbowz/wyw
- ✓ google.com/search?q=*