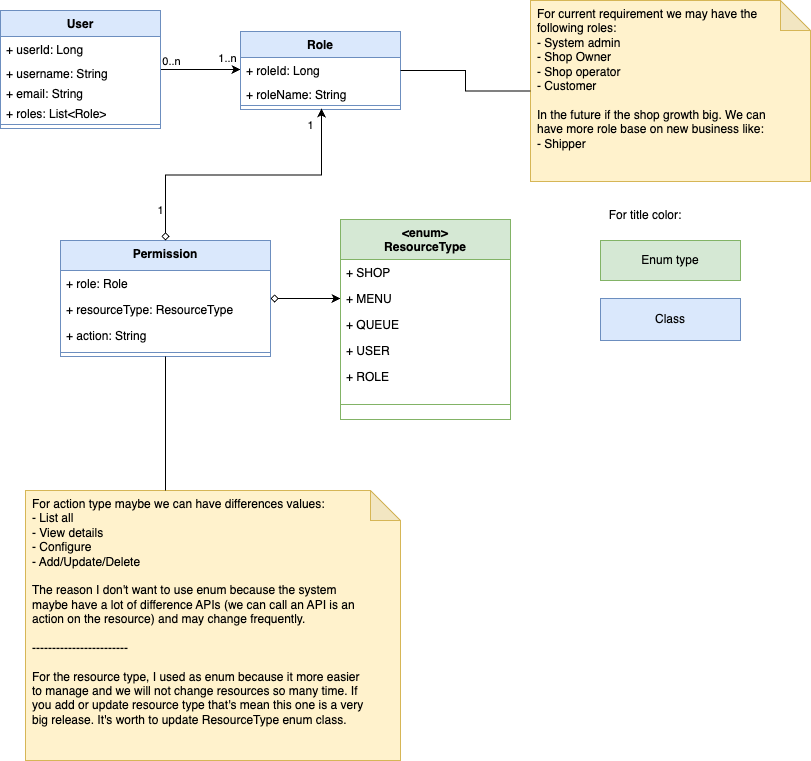
Coffeeshop Solution Design

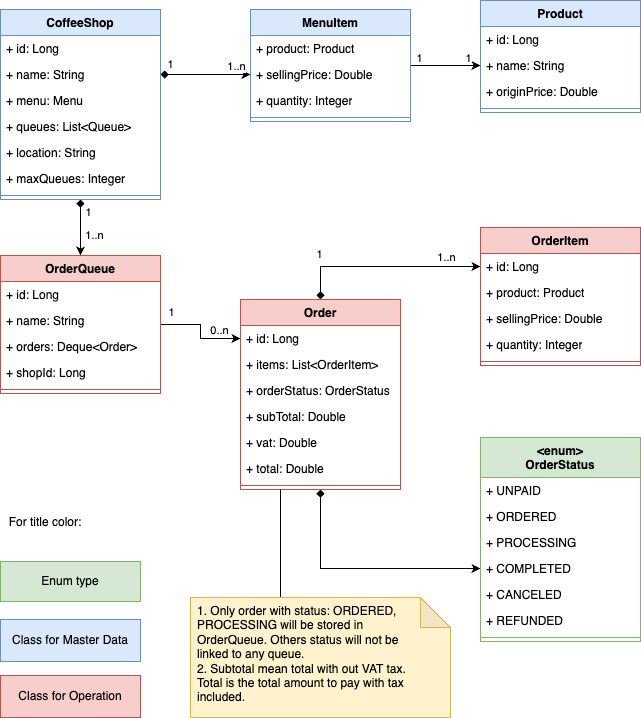
# Class design

This is class design and their “virtual” relationship. Because we’re going to implement it as microservices, so If you want more details on the model for each server I can do it later.

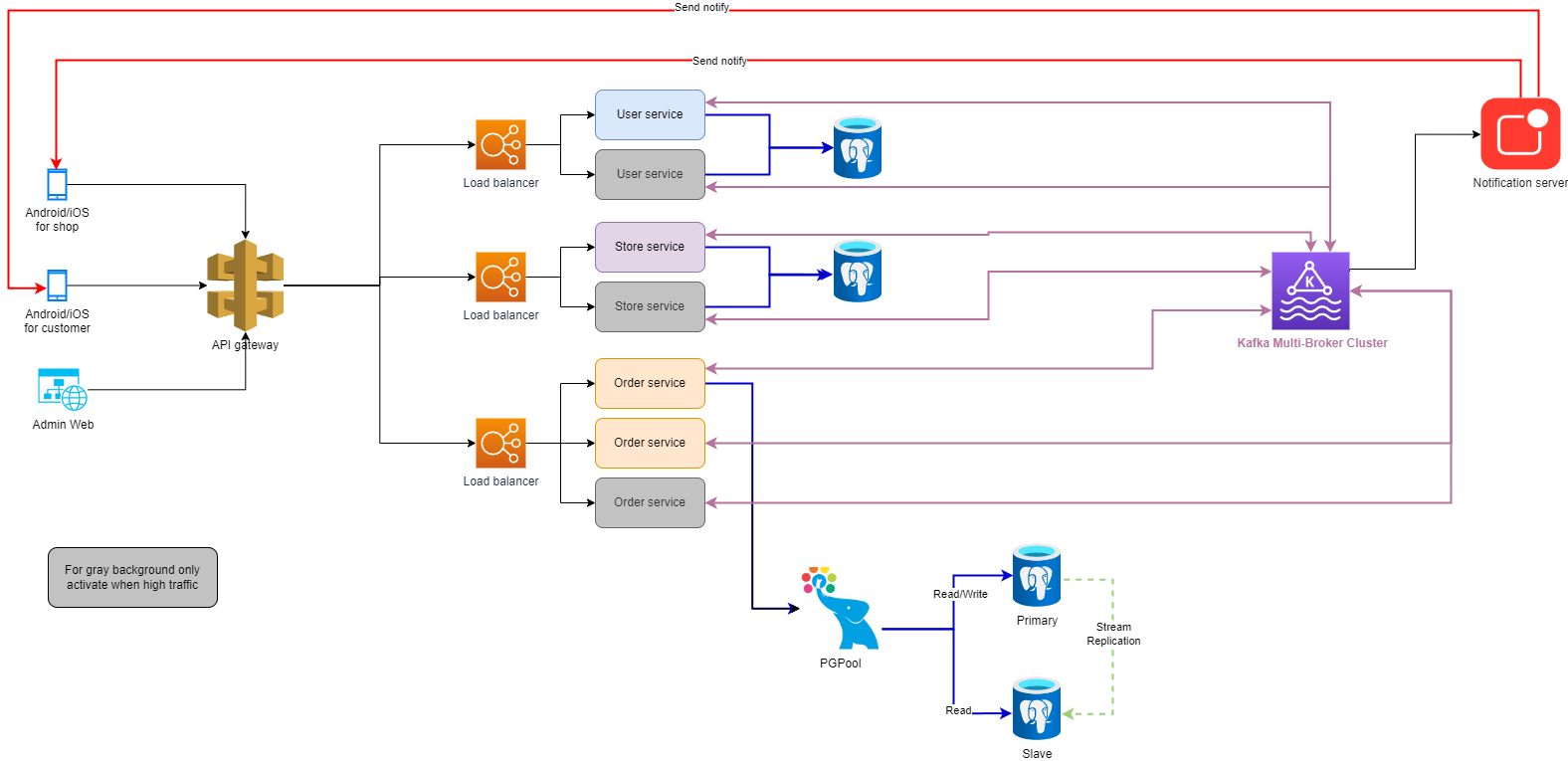
## User management



## The coffee shop



# Component diagram

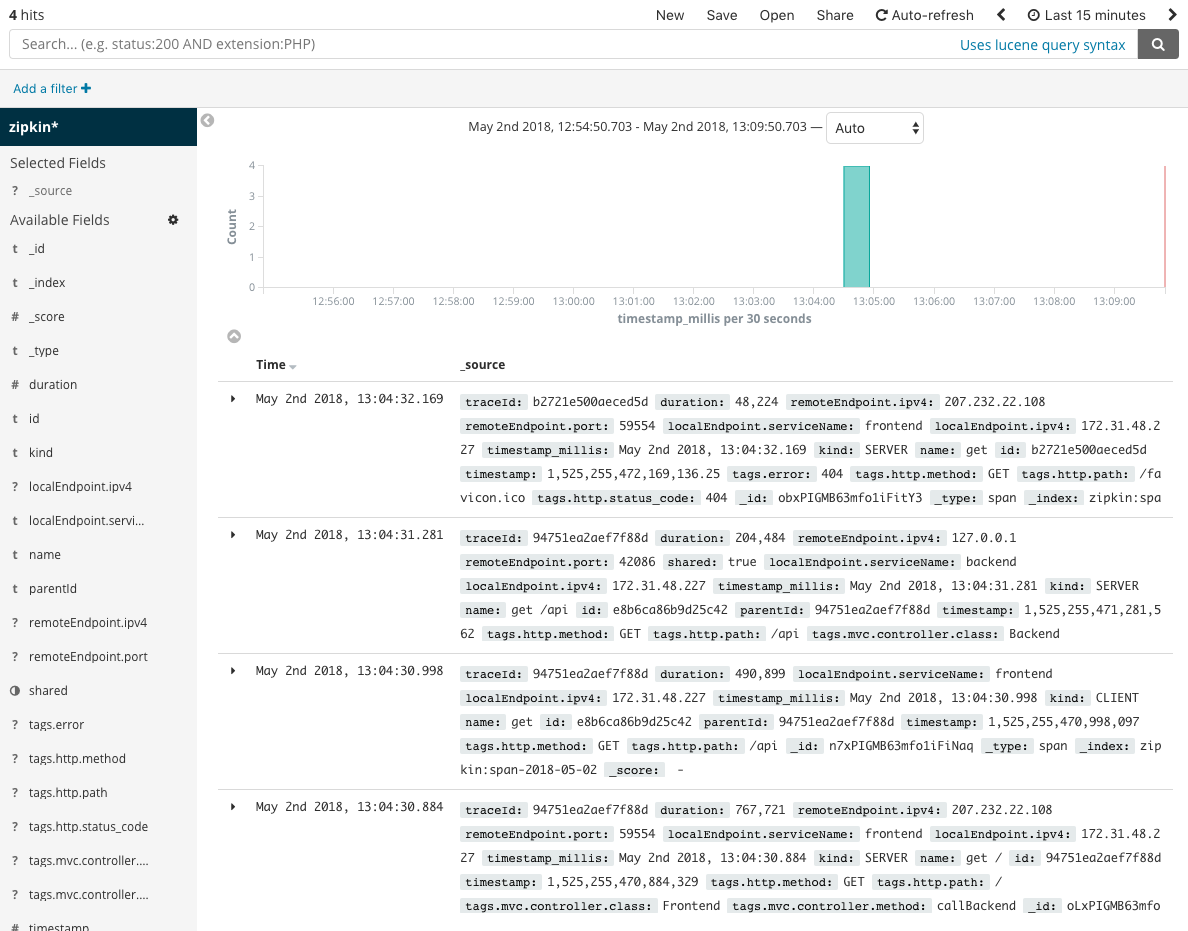


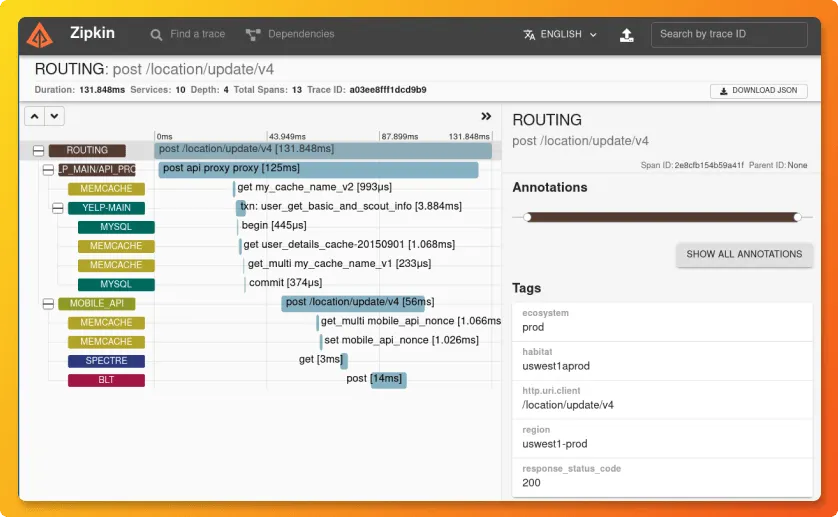
For the gray box, I just want to show that it is possible to scale. In the real world, it’s easy to launch or deactivate a Kubernetes server base on current load of servers.

Component scope:

* User service: manage all information related to user, role permission and authentication.
* Store service: manage store information (CRUD store)
* Order service: handle all requests related to Order like: Place order, Process order, Complete/Cancel order…
* PgPool is helping scale PostgreSQL database for Order service since the most workload will be on order service. So we can separate database into read/write database and read only databases for load balancing.
* Notification server (or maybe service) to manage and send notification for order statuses to user phone (both customer and store operator)

One item that I don’t want to include in the above diagram is logging/tracing/monitor rool. Usually I will choose to gather all logs from all microservice into ELK -> Zipkin to trace microservice performance and issues as sample screenshots below:





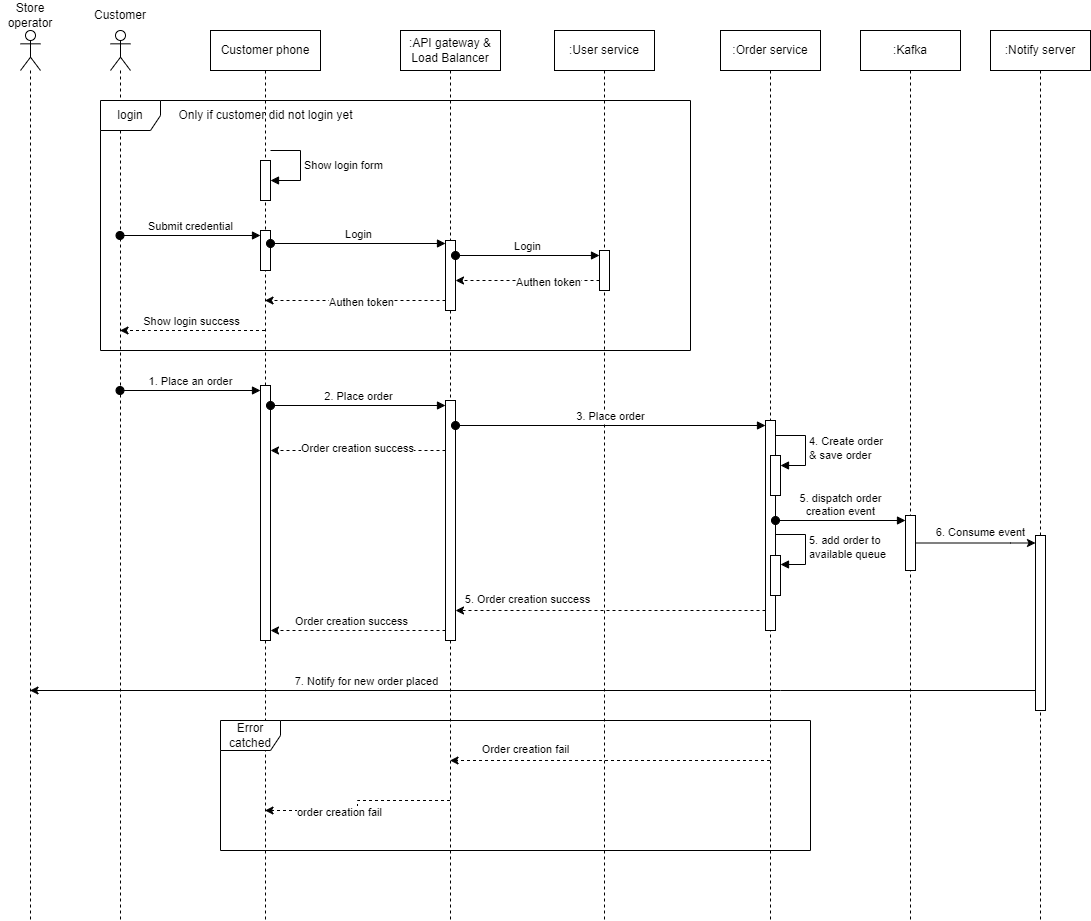
# Security:

For a simple app for coffee shops. Let's use HTTPS for mobile apps and websites to connect with API Gateway. For APIs from services, only accept requests from API gateway and internal services.

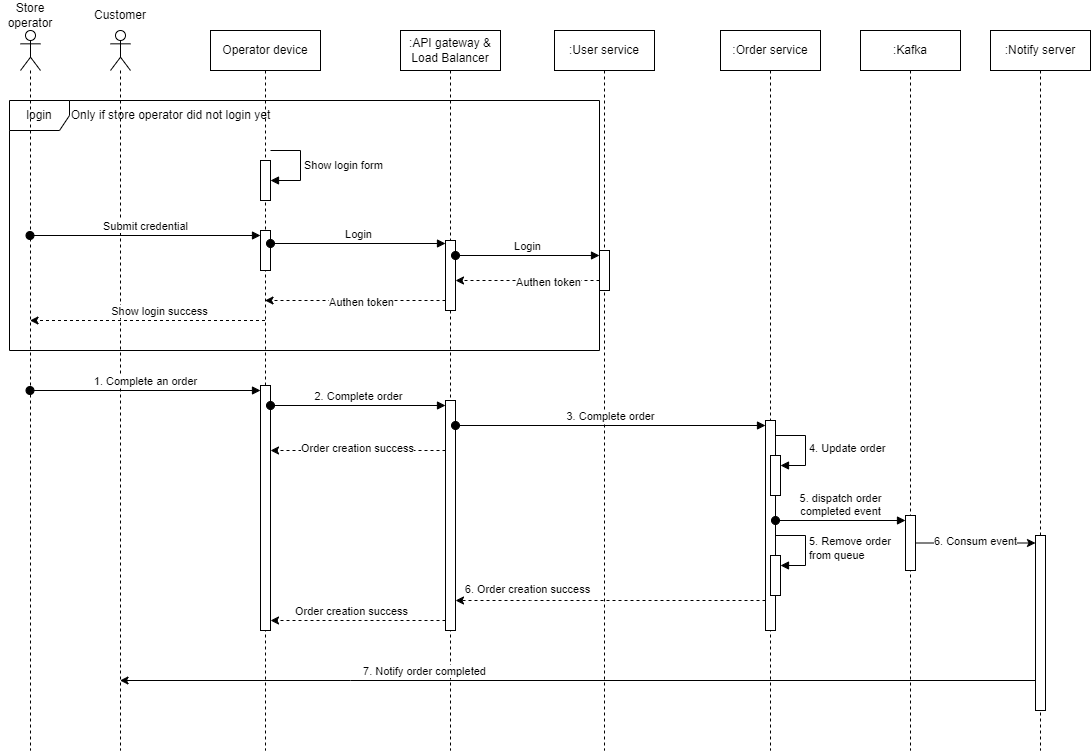
Server will provide JWT token based on ip address + device id for each login. Login JWT token only available for 1 hour and need client call refresh or re-login.

# Workflow

1. Customer making order



1. Store operator complete an order



# Possible API list

List of API that need to implement from services:

| **Service** | **API** |
| --- | --- |
| User service | CRUD for user account |
|  | CRUD for role |
|  | User’s role assignment |
|  | Login |
|  | Reset password |
|  | User list |
|  | Role list |
|  | Permission list |
| Store service | CRUD for store |
|  | Store list |
| Order service | Place order |
|  | Complete order |
|  | Cancel order |
|  | Order list |
|  | Order by status |

# Testing approach

* Android/IOS devices for manual test.
* Base on the budget, If we have a good budget let also build automation testing with **Appium**
* If we have more budget, let's order a mobile cloud testing service to have a variety of device types for testing. We can consider browserstack, katalon…

# Estimation

High level estimation:

| **Component** | **Feature** | **Estimate**  **(Man/Day)** |
| --- | --- | --- |
| User services | CRUD for user account | 2 |
|  | CRUD for role | 2 |
|  | User’s role assignment | 2 |
|  | Login | 1 |
|  | Reset password | 1 |
|  | User list | 1 |
|  | Role list | 1 |
|  | Permission list | 1 |
| Store service | CRUD for user account | 2 |
|  | Store list | 1 |
| Order service | Place order | 3 |
|  | Complete order | 3 |
|  | Cancel order | 2 |
|  | Order list | 1 |
|  | Order by status | 2 |
|  | Queue list by store | 1 |
|  | Process order | 2 |
| Customer Mobile app | Android login page | 3 |
|  | IOS login page | 3 |
|  | Home page design | 2 |
|  | Android home page | 5 |
|  | IOS home page | 5 |
|  | Place order design | 5 |
|  | Android place order | 8 |
|  | IOS place order | 8 |
|  | My orders design | 2 |
|  | Android my orders list | 5 |
|  | IOS my orders list | 5 |
|  | Cancel order design | 2 |
|  | Android cancel order | 3 |
|  | IOS cancel order | 3 |
| Store app | Android login page | 3 |
|  | IOS login page | 3 |
|  | Home page design | 2 |
|  | Android home page | 5 |
|  | IOS home page | 5 |
|  | Processing order design | 2 |
|  | Android processing order | 3 |
|  | IOS processing order | 3 |
|  | Complete order design | 2 |
|  | Android complete order | 3 |
|  | IOS complete order | 3 |
|  | Dev testing | 10.2 |
| Admin Web | User management | 15 |
|  | Role management | 15 |
|  | Permission management | 15 |
|  | Store management | 10 |
| Total development |  | 186.2 |
| Dev test and unit test |  | 55.86 |
| Feature testing |  | 55.86 |
| Infra setup |  | 15 |
| Deployment |  | 5 |
| Total for first release |  | 317.92 |

Noted: not included infra cost in this estimations.