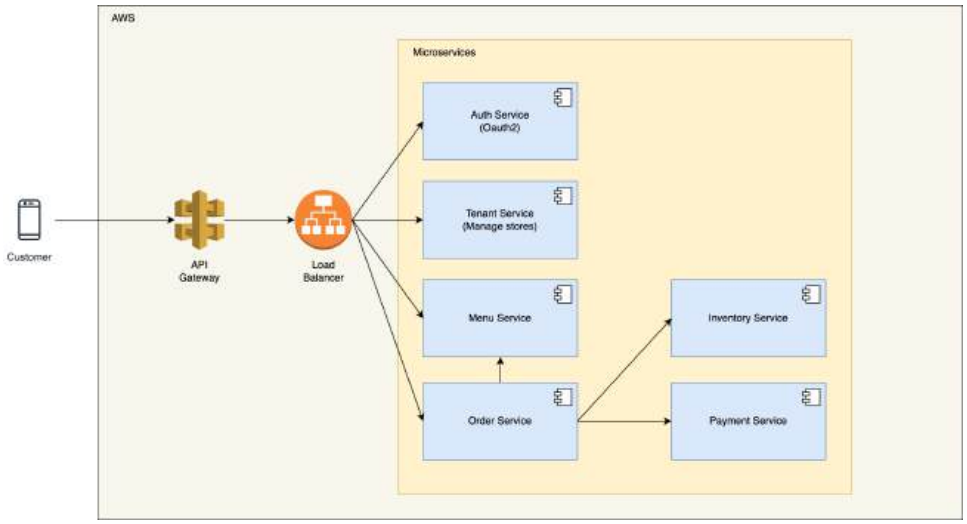
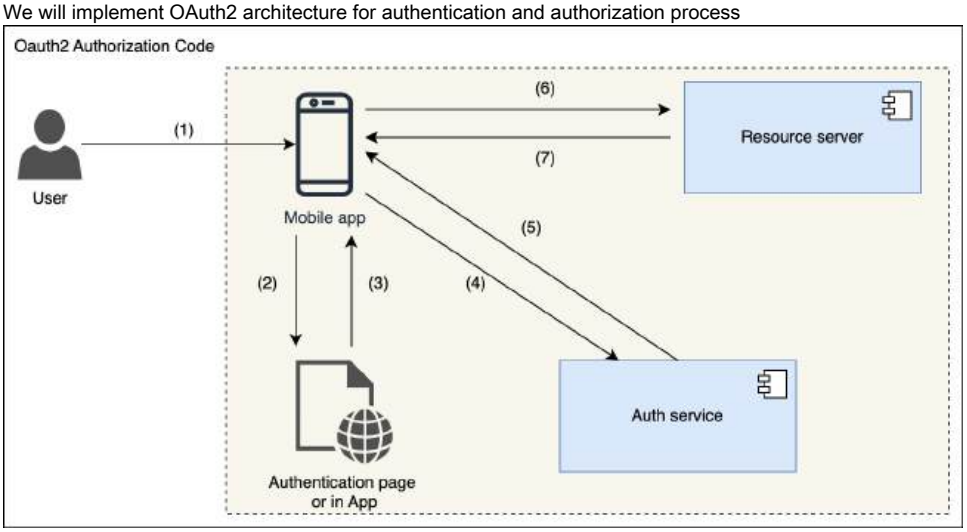
Global Coffee Shop

1. Overview
   This is a global coffee shop system that allows loyal customers to pre-order their coffee before arriving at the store.
   The system is designed to enhance customer convenience by enabling them to select and pay for their beverages in advance through a mobile app or website. Customers can specify their preferred pick-up time, ensuring that their order is ready when they arrive. The system aims to streamline the customer experience, reduce wait times, and provide a seamless and personalized service across various locations worldwide.
2. Microservices Architecture

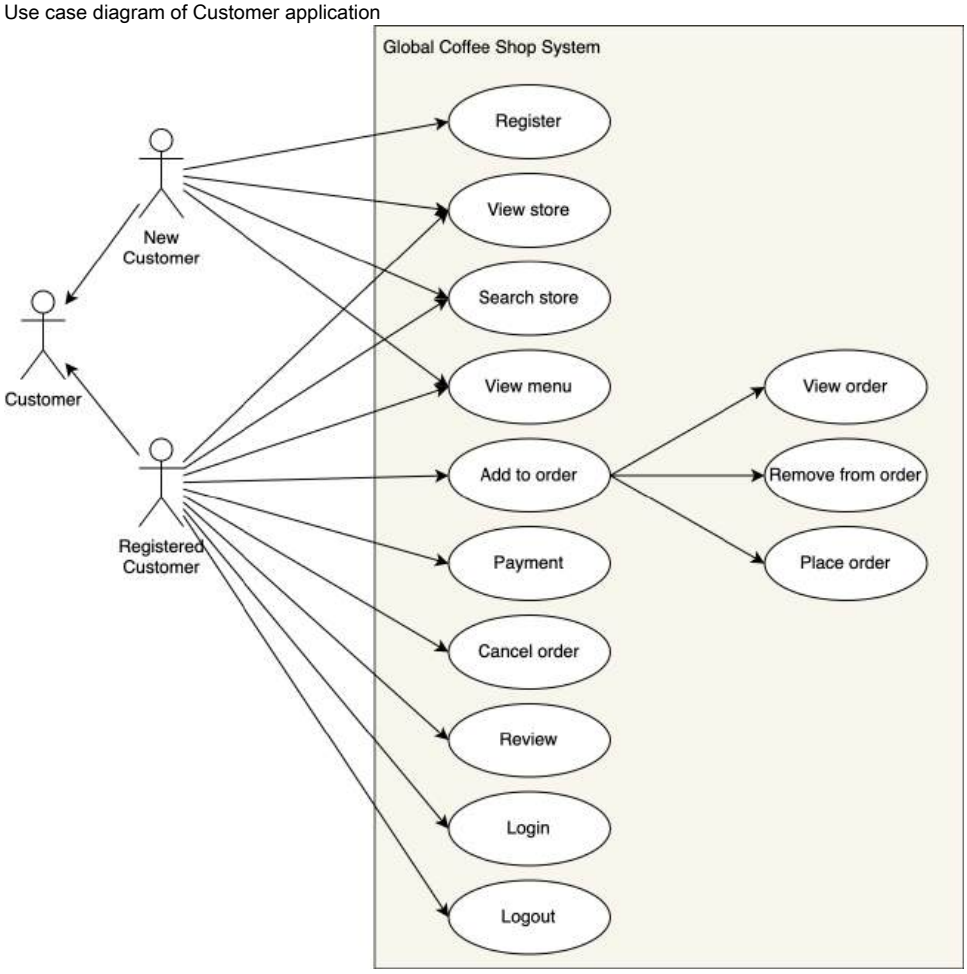


   - Customer: The customer want to order coffee online in this system
   - AWS Api Gateway: Route request, throttling, rate limit, security, logging, custom domain name
   - Load Balancer: Distributes traffic across multiple targets in different availability zones
   - Auth service: Implement OAuth2 architecture, responsible for authentication, authorization in micro-services
   - Tenant service: Implement multi tenancy to register many stores in the world. Manage store information.
   - Menu service: Manage all of the items in the store menu.
   - Order service: Responsible for the coffee ordering process
   - Inventory service: Check whether the item is out of stock
   - Payment service: Payment order
3. Security architecture
   We will implement OAuth2 architecture for authentication and authorization process

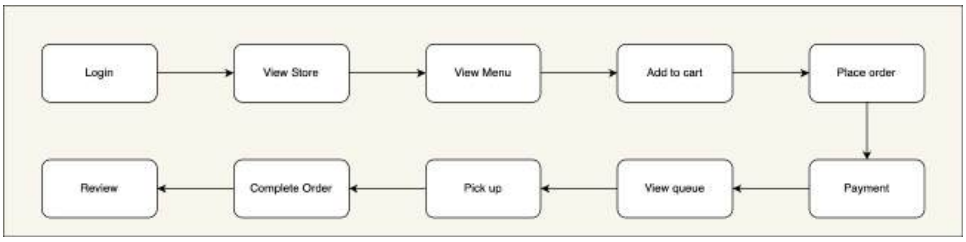


   (1): User visits the application and logs in
   (2): The application redirects the user to authentication page to enter their username and password
   (3): The Auth service validate the username and password, then returns authorization code to the mobile application
   (4): The mobile application exchange authorization code for an access token and a refresh token
   (5): The Auth service responds with access token, its expire time, and the refresh token in the response body
   (6): The mobile application calls the resource server's Rest API, including access token in the request header
   (7): The Resource server validates the access token and responds with the requested date to the mobile application

4. Use case diagram of Customer application



5. Data flow Architecture



6. Database Schema

Database tables:
- Item: Store kind of drinks (Espresso, Americano, Cappucino, ...)
- Menu Item: List items with size and price
- Menu_Component: Common coffee components (Coffee bean, sugar, milk,...)
- Size: Small, Medium, Large, ...
- Order to store every stages of order (PENDING, AWAITING_PAYMENT, IN_QUEUE, PICKED_UP, COMPLETED, CANCELLED, FAILED, REFUNDED), including total price, customer,...
- Order_Item: Store order detail about item, size, quantity, and special instructions
- Queue: define number of queue in a coffee shop
- Order_Queue: Store order in a queue and display estimate ready time, status(PREPARING, READY)
- Review: Customer can review about the order and the coffee shop
- Payment: Store payment stage, payment method

7. Tech stacks

Language/Framework/Library	Version	Description
Java	17	OOP programming language
Spring Boot	3.3.5	Java Framework
Spring Cloud	2023.0.3	Implement communication in microservices
Spring Cloud AWS	3.2.1	Using AWS services (SQS)
Liquibase	4.27.0	Database migration tool
`Terraform`	1.9.5	Init Infrastructure
Postgres	15	Database

8. Testing:
We will import postman collection and change AWS ALB endpoint url

coffee-shop.post..._collection.json