SOA Project

This application is developed as an online platform for courses, where users can view a list of available courses and enroll in the ones they want. It is designed for people interested in learning different subjects, allowing them to explore courses and register online.

Users can access the course list and details without being logged in, but they need to sign in to enroll in a course. After enrolling, users receive a confirmation email (simulated as a console message containing the course details).

Application Architecture (Microservices)

The backend consists of six microservices, each with a specific role. There is an authentication service (auth-service), which handles user login and registration, a load balancer using Nginx, a service for retrieving course information (courses-service), a service for managing enrollments (enrollment-service), which allows authenticated users to enroll in a course, a service for generating enrollment confirmations (invoices-service), and a service for simulating email notifications (email-service).

The load balancer handles requests between auth-service, courses-service, and enrollment-service. When a user enrolls in a course, the enrollment-service sends a message to the invoices-service via a Kafka queue. The invoices-service then forwards the confirmation details to the email-service using a RabbitMQ queue to simulate sending an email. Additionally, I also explored the implementation of a FaaS (Function as a Service) using Azure Functions, which returns the number of available courses.

An overview of the app services in the backend can be seen in the diagram below (Figure 1), which illustrates the microservices architecture.

The frontend is a separate microservice that provides pages for login, browsing courses, viewing course details, and enrolling in a course. I attempted to implement a micro frontend architecture consisting of three microservices: a container, a login app, and a courses app.

The application uses Docker for creating the images and deploying the containers. (Figure 2)

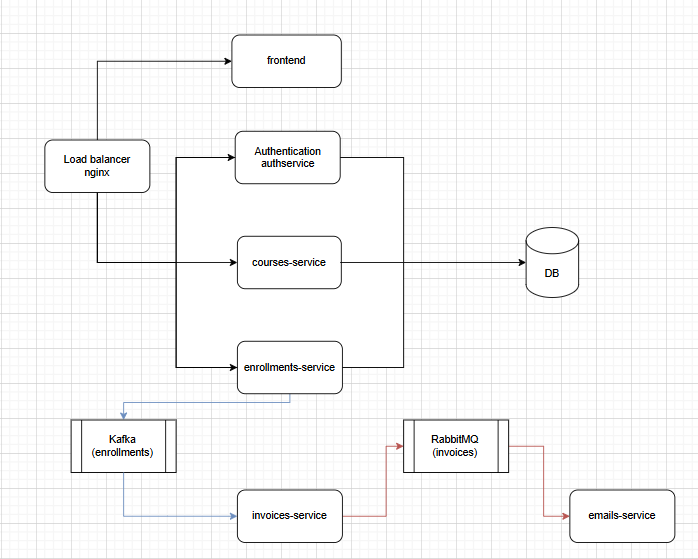


Figure 1. An overview of the app services

A screenshot of a computer

Description automatically generated

Figure 2. Docker

UML Diagram of frontend package:

A screenshot of a computer

Description automatically generated

Figure 3. UML Diagram of frontend package

Context Diagram:

A diagram of a person's face

Description automatically generated

Figure 4. Context Diagram

Container Diagram:

A diagram of a computer program

Description automatically generated

Figure 5. Container Diagram