Microservices-Based Online Learning Platform

Project Overview

The platform will include core functionalities for course management, user enrollment, content delivery, and progress tracking, each implemented as a separate microservice. This modular design will allow for independent scaling and updates of individual services without affecting the entire platform.

Key Features

Course Management Service: This microservice will handle the creation, updating, and deletion of courses. It will manage course metadata, including titles, descriptions, syllabi, and associated learning materials. It will also provide APIs for other services to access course information.

User Enrollment Service: This service will manage user registration, authentication, and enrollment in courses. It will handle user profiles, track course enrollments, and provide authentication services for other microservices.

Content Delivery Service: This microservice will be responsible for storing and delivering learning content, such as videos, documents, and quizzes. It will provide APIs for accessing and streaming content based on user enrollments. It might integrate with cloud storage services.

Progress Tracking Service: This service will track user progress within courses, including quiz scores, completion of modules, and overall course progress. It will provide APIs for retrieving and updating user progress data.

API Gateway: An API gateway will act as a single entry point for all client requests. It will route requests to the appropriate microservices, handle authentication, and potentially implement rate limiting and other cross-cutting concerns.

Possible Approaches

Web-Based Platform: Develop the frontend using technologies like React.js or Vue.js to create a user-friendly interface for the platform. Microservices can be developed using Node.js (with Express or NestJS) or Python (with Flask or Django). The key is to select libraries and APIs you prefer.

Database: Each microservice can have its own database (e.g., PostgreSQL, MySQL, MongoDB) to ensure data isolation. Consider using a database-per-service pattern.

Again, you can customize your solution per your preferences. You will have a project proposal for that purpose.