**--E-learning Platform**

**📝Overview**

This project is a fully containerized microservices application built using Spring Boot, deployed on AWS EC2, and registered with Eureka Server for service discovery. It consists of five key modules:

**✅ User Module** – Handles authentication and user management

**✅ Course Module** – Manages course-related operations

**✅ Enrollment Module** – Tracks student enrollments

**✅ Assessment & Submission Module** – Handles assessments and submissions

**✅ Notification Module** – Manages logging and email notifications

**🌐 Deployment Architecture**

* EC2 Instance 📡 → Hosted all services on an AWS EC2 instance
* Eureka Server 🔗 → Registered all microservices for seamless service communication
* Docker 🐳 → Each microservice is containerized and deployed using Docker Compose
* Spring Boot REST APIs 🔥 → Services interact using RESTful endpoints

**🛠 Tech Stack**

* Spring Boot – Microservices framework
* Spring Cloud Eureka – Service registry & discovery
* Spring Data JPA – Database integration
* Docker & Docker Compose – Containerized deployment
* Spring Security-To ensure security
* AWS EC2 – Cloud hosting
* Postman – API testing

**👤 User Module:**

**📝 Overview**

A Spring Boot microservice for handling user registration, authentication, password management, and user retrieval.

**⚡ Tech Stack**

✅ Spring Boot – Microservice framework

✅ Spring Security – Authentication & authorization

✅ Spring Data JPA – Database integration

✅ Postman – API testing

**🔗 API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /user/register | 🆕 Register a new user |
| POST | /user/login | 🔑 Authenticate user login |
| POST | /user/forgotpassword | 📧 Send password reset email |
| POST | /user/resetpassword | 🔄 Reset user password |
| GET | /user/userId/{id} | 🔍 Fetch user details by ID |

**🔎 API Testing**

Use Postman or swagger

--------------------------------------------------------------------------------------------------------------

🚀 **Course Module:**

**📝 Overview**

A Spring Boot microservice for managing courses, providing CRUD operations with database integration.

**⚡ Tech Stack**

✅ Spring Boot – Microservice framework

✅ Spring Data JPA – Database interaction

✅ RESTful APIs – Controller-based endpoints

✅ Postman – API testing

**🔗 API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /course/courses | 🆕 Add a new course |
| PUT | /course/courses/{id} | 🔄 Update an existing course |
| DELETE | /course/courses/{id} | ❌ Remove course |
| GET | /course/courses | 📋 Fetch all courses |
| GET | /course/courses/count | 🔢 Get total course count |
| GET | /course/courses/{id} | 🔍 Retrieve course by ID |

**🔎 API Testing**

Use Postman or swagger API testing:

--------------------------------------------------------------------------------------------------------------

**🚀 Enrollment Module:**

**📝 Overview**

A Spring Boot microservice for managing student enrollments, providing CRUD operations and logging service integration.

**⚡ Tech Stack**

✅ Spring Boot – Microservice framework

✅ Spring Data JPA – Database handling

✅ Spring Cloud Feign – Microservice communication

✅ Postman – API testing

**🔗 API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /enrollment/enrollments | 🆕 Add new enrollment |
| GET | /enrollment/enrollments/{id} | 🔍 Get enrollment by ID |
| GET | /enrollment/enrollments | 📋 Fetch all enrollments |
| PUT | /enrollment/enrollments/{id} | 🔄 Update enrollment |
| DELETE | /enrollment/enrollments/{id} | ❌ Remove enrollment |

**🔎 API Testing**

Use **Postman** or swagger

--------------------------------------------------------------------------------------------------------------

**🚀 Assessment Module:**

🚀 **Assessment Service - Spring Boot Microservice**

📝 **Overview**

A RESTful microservice for managing assessments with logging and CRUD operations.

**⚡ Tech Stack**

✅ Spring Boot – Microservices framework

✅ Spring Cloud Feign – Communication

✅ Spring Data JPA – Database handling

✅ Postman – API testing

**🔗 API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /assessments/assessment | Add a new assessment 🆕 |
| PUT | /assessments/assessment/{id} | Update an assessment 🔄 |
| DELETE | /assessments/assessment/{id} | Remove assessment ❌ |
| GET | /assessments/assessment | Get all assessments 📋 |
| GET | /assessments/assessment/count | Count total assessments 🔢 |
| GET | /assessments/assessment/{id} | Get assessment by ID 🔍 |

**🔎 API Testing**

Use Postman

--------------------------------------------------------------------------------------------------------------

**🚀 Submission Module:**

**📌 Overview**

A lightweight REST API for managing submissions with integrated logging and database operations.

**🛠 Tech Stack**

* 🏗 Spring Boot – Microservices framework
* 🔗 Spring Cloud Feign – Logs microservice actions
* 📊 Spring Data JPA – Handles database operations
* 🛡 Spring Security (Future Upgrade) – Authentication and authorization

**📡 API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /submissions/submission | 🆕 Create a new submission |
| PUT | /submissions/submission/{id} | 🔄 Update a submission |
| DELETE | /submissions/submission/{id} | ❌ Remove submission |
| GET | /submissions/submission | 📋 Fetch all submissions |
| GET | /submissions/submission/count | 🔢 Get submission count |
| GET | /submissions/submission/{id} | 🔍 Retrieve submission by ID |

**🔎 API Testing**

Test the service using Postman or swagger

**📧 Notification Module:**

**📝 Overview**

A Spring Boot microservice for sending email notifications, integrating with a mailing service.

**⚡ Tech Stack**

✅ Spring Boot – Microservice framework

✅ Spring Web – REST API setup

✅ Email Service – Handles email sending

**🔗 API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /email/emailService | 📧 Send an email |

**🔎 API Testing**

Use **Postman** or swagger

--------------------------------------------------------------------------------------------------------------

**🚀 Logging Service Module:**

**📝 Overview**

A Spring Boot microservice for centralized logging, allowing different log levels (INFO, WARN, ERROR) based on incoming requests.

**⚡ Tech Stack**

✅ Spring Boot – Microservice framework

✅ SLF4J & LoggerFactory – Log management

✅ RESTful API – Controller-based endpoints

**🔗 API Endpoints**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /logging/post | 📝 Logs messages with specified levels (INFO, WARN, ERROR) |

**🔎 API Testing**

Use **Postman** or swagger