LearnMore

Learning Management System

Business Requirement Document

Mohammed Jasrin and Vishal Swami

September 3, 2024

Contents

1 Introduction

1.1 Project Overview

The LearnMore Learning Management System (LMS) is designed to streamline the management of educational content and user interactions within an organization. This platform focuses on Admin-Driven Enrollments, where course assignments are automatically managed based on user roles. Administrators can efficiently oversee users, assign courses, and track progress, while users benefit from a seamless experience with pre-assigned courses and easy access to educational materials.

1.2 Objective

The primary objective of the LearnMore Learning Management System (LMS) is to develop an efficient and user-friendly platform that automates course assignments based on user roles, thereby simplifying administrative tasks and enhancing the learning experience. By integrating automatic enrollment, LearnMore ensures that users are promptly and accurately assigned relevant courses, reducing manual intervention and administrative overhead. The system aims to provide robust tools for managing user profiles, courses, and progress tracking, while incorporating feedback mechanisms to continuously improve the training programs. Additionally, LearnMore is designed to be scalable and adaptable, supporting future growth and additional features while ensuring high performance and usability across both web and mobile interfaces.

1.3 Target Audience

The target audience for the LearnMore Learning Management System (LMS) includes a diverse range of users who benefit from streamlined and effective training management. This encompasses remote teams, project managers, and organizations with a global workforce who require a robust platform for managing and delivering training programs. For remote teams and individual users, LearnMore offers an intuitive interface for accessing assigned courses, tracking progress, and engaging with educational materials. Project managers will find value in the system's tools for overseeing course assignments and monitoring user progress, while organizational leaders will leverage the platform's insights for strategic decision-making and ensuring alignment with business goals. The LMS is tailored to meet the needs of these groups by providing a comprehensive and scalable solution for efficient training management and user engagement.

1.4 Key Features

- Automatic Enrollment: Courses are automatically assigned to users based on their roles, simplifying the enrollment process and ensuring that users receive relevant training without manual intervention.

- Admin Control: Centralized management of user profiles, course assignments, and progress tracking.

- Course Management: Tools for creating, updating, and organizing courses, including multimedia content.

- Progress Tracking: Monitoring and reporting on user progress through course modules and assessments.
- Feedback Integration: Mechanisms for collecting and analyzing user feedback to continuously improve training programs.

2 Business Goals

2.1 Admin Control

Allow administrators to manage course assignments, ensuring users are enrolled in the necessary training programs.

2.2 Tracking and Compliance

Track user progress and ensure compliance with training requirements.

2.3 Increase Accountability

Collect and analyze feedback to improve training programs.

2.4 Feedback Integration

Improve overall productivity by streamlining task management processes, reducing delays caused by time zone differences, and providing timely notifications and updates.

2.5 Efficiency and Scalability

Build a scalable and efficient system to handle multiple users and courses.

2.6 Seamless Integration

Ensure all microservices interact smoothly, with the User microservice managing connections using RestTemplate.

3 Scope

3.1 In Scope

- **Development of five microservices:** User Management, Course Management, Enrollment Management, Progress Tracking, and Feedback Management.
- **Each microservice will have a RESTful API, interact with a SQL database, and use RestTemplate for inter-service communication.**
- **The system will support both web and mobile interfaces.**
- **Integration with an API Gateway for unified access and communication.**
- **WebClient implementation for service-to-service calls where needed.**

3.2 Out of Scope

- **Integration with External Systems:** Integration with third-party systems (e.g., HR systems, external content providers) is not included. Future enhancements could incorporate these integrations.

- **Advanced Analytics:** The system will not include advanced analytics or machine learning features for predictive analysis or automated recommendations. Future versions may integrate these capabilities.

- **Customizable User Interfaces:** Extensive customization of the user interface beyond standard features and themes is not included. Future updates could allow for greater UI customization.

- **Offline Access:** Offline access or capabilities for course materials are not supported. Future releases might offer offline functionality.

- **Legacy System Integration:** Integration with legacy systems or platforms not specified is out of scope. Future enhancements could address compatibility with such systems.

- **Multi-language Support:** Support for multiple languages is not included in the initial release. Future versions may add multi-language capabilities.

- **Third-Party Content Licensing:** Management of third-party educational content or materials is not covered. Future versions might address content licensing and management.

4 Stakeholders

4.1 Business Stakeholders

- **Admin:** Manages users, courses, and enrollments; tracks progress and collects feedback.

- **Users:** Follows the assigned courses, views schedules, interacts with course content, tracks progress, and provides feedback.

5 System Architecture

5.1 Microservices Architecture

- **User Service:** Manages user profiles and course enrollments, with RestTemplate to connect to other microservices.

- **Course Service:** Manages course data and content.

- **Enrollment Service:** Admin assigns courses to users and manages enrollments.

- **Progress Service:** Manages and tracks user progress.

- **Feedback Service:** Manages feedback submission and analysis.
- **API Gateway:** Routes requests to appropriate microservices and handles security.

5.2 Database Design

- **SQL Database:** Each microservice will interact with its own schema within a SQL database.

5.3 Communication

- **RESTful APIs:** Each microservice exposes RESTful APIs for interaction.
- **Inter-Service Communication:** Use RestTemplate for communication between services, with some services using WebClient where necessary.

6 User Interfaces

6.1 Admin Interface

- **Dashboard:** Overview of users, courses, and enrollments.
- **User Management:** Create, update, delete, and view user profiles.
- **Course Management:** Create, update, delete, and view courses; manage course content.
- **Enrollment Management:** Assign and manage course enrollments.
- **Progress Tracking:** View and manage user progress.
- **Feedback Management:** View and analyze feedback from users.

6.2 User Interface

- **View Assigned Courses:** List and view details of courses assigned by the admin.
- **View Schedule:** Display course and module schedules.
- **Interact with Course Content:** Access and engage with course materials (e.g., videos, documents, quizzes).
- **Track Progress:** View progress in assigned courses.
- **Provide Feedback:** Submit feedback on courses and training programs.

7 Implementation Plan

7.1 Development Phases

1. Planning: Define detailed requirements, project timelines, and resource allocation.
2. Design: Create system architecture, database schema, and API specifications.
3. Development: Implement microservices, APIs, and user interfaces.
4. Testing: Conduct unit testing, integration testing, and user acceptance testing.
5. Deployment: Deploy the system to production, provide user training.
6. Maintenance: Monitor performance, address issues, and update as necessary.

7.2 Resources

- **Development Team:** Developers, database administrators, UX/UI designers.
- **Tools:** IDEs, version control systems, testing tools, deployment platforms.
- **Documentation:** Maintain comprehensive documentation for the system architecture, APIs, and user guides.

8 Functional Requirements

8.1 User Microservice

- **CRUD Operations:** Create, Read, Update, Delete user profiles.
- **Enrollment Tracking:** Maintain a list of courses assigned to each user.
- **Role Management:** Support different user roles (Admin, User).
- **Automatic Role-Based Course Assignment:** Automatically assign courses to users based on the role assigned by the admin. Direct integration with the Enrollment microservice using RestTemplate.

8.2 Course Microservice

- **CRUD Operations:** Create, Read, Update, Delete courses.
- **Course Details:** Manage course information, including title, description, and schedule.
- **Course Role Association:** Specify roles during course creation or update, determining user role eligibility for course enrollment.
- **Course Content Management:**
  - Upload and organize course materials (e.g., videos, documents, quizzes).
  - Structure content into modules or lessons.
  - Support for multimedia content.

8.3 Enrollment Microservice

- **CRUD Operations:** Create, Read, Update, Delete enrollments.
- **Admin-Driven Enrollment:** Automatically assign courses to users based on their role and the role-based associations defined in the Course Microservice. The system should update user enrollments accordingly whenever a user's role changes or when new courses are created with specific role associations.

8.4 Progress Microservice

- **CRUD Operations:** Create, Read, Update, Delete progress records.
- **Progress Tracking:**
  - Track completion of course modules and lessons.
  - Monitor quiz scores and assignment submissions.
  - Report on time spent on course content.

8.5 Feedback Microservice

- **CRUD Operations:** Create, Read, Update, Delete feedback records.
- **Feedback Collection:** Collect feedback from users about courses and training programs.

9 Non-Functional Requirements

- **Performance:** System should handle a high number of users and requests efficiently.
- **Scalability:** System must be scalable to accommodate future growth.
- **Security:** Implement strong data protection and access controls.
- **Usability:** User interfaces should be intuitive and easy to navigate.
- **Reliability:** System should be robust with minimal downtime and error handling.
- **Inter-service Communication:** Efficient use of RestTemplate and WebClient for service-to-service communication.

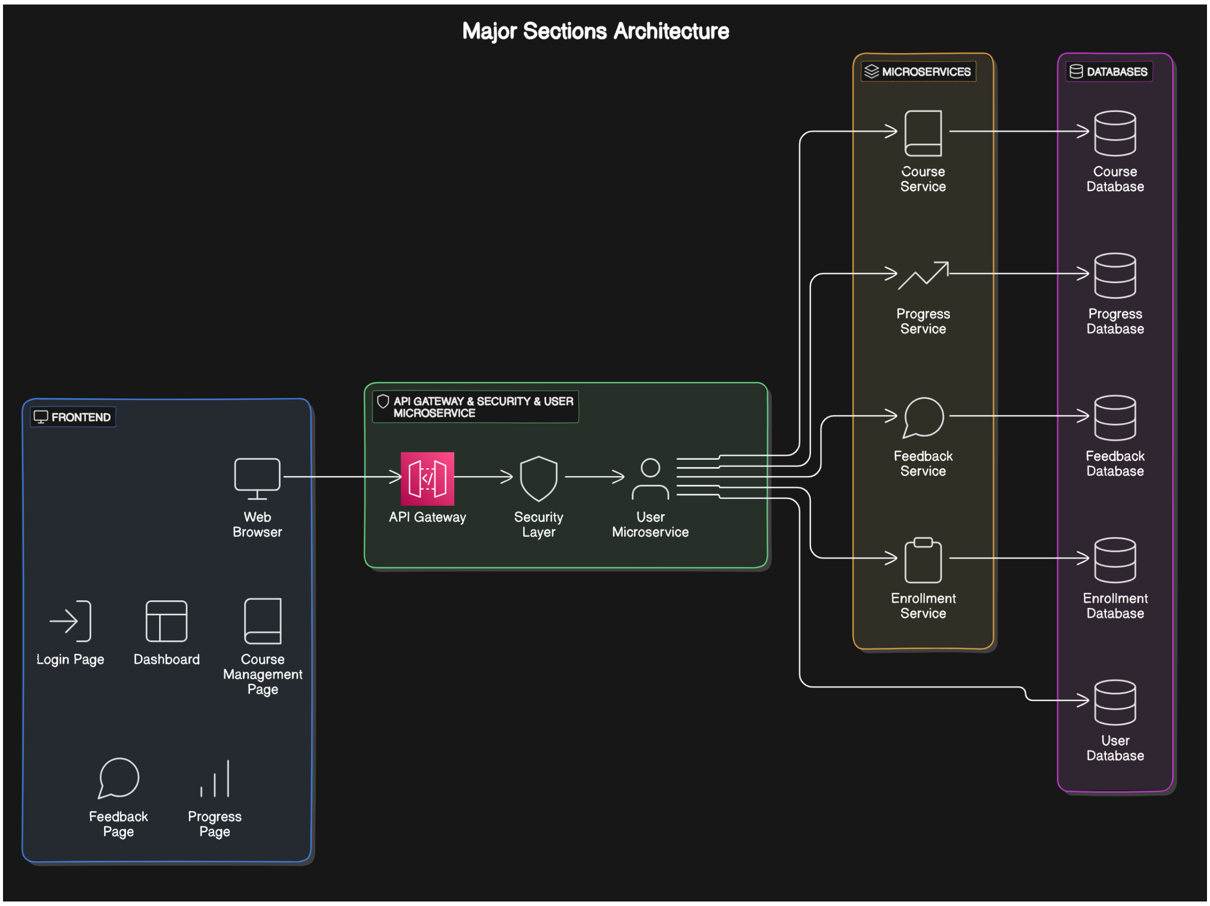

Figure 1: Architecture Diagram

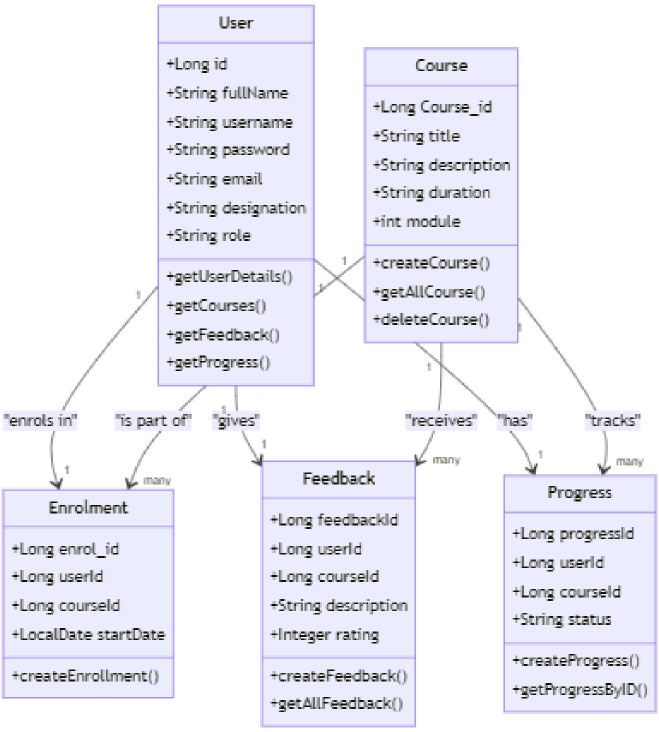

Figure 2: Class Diagram

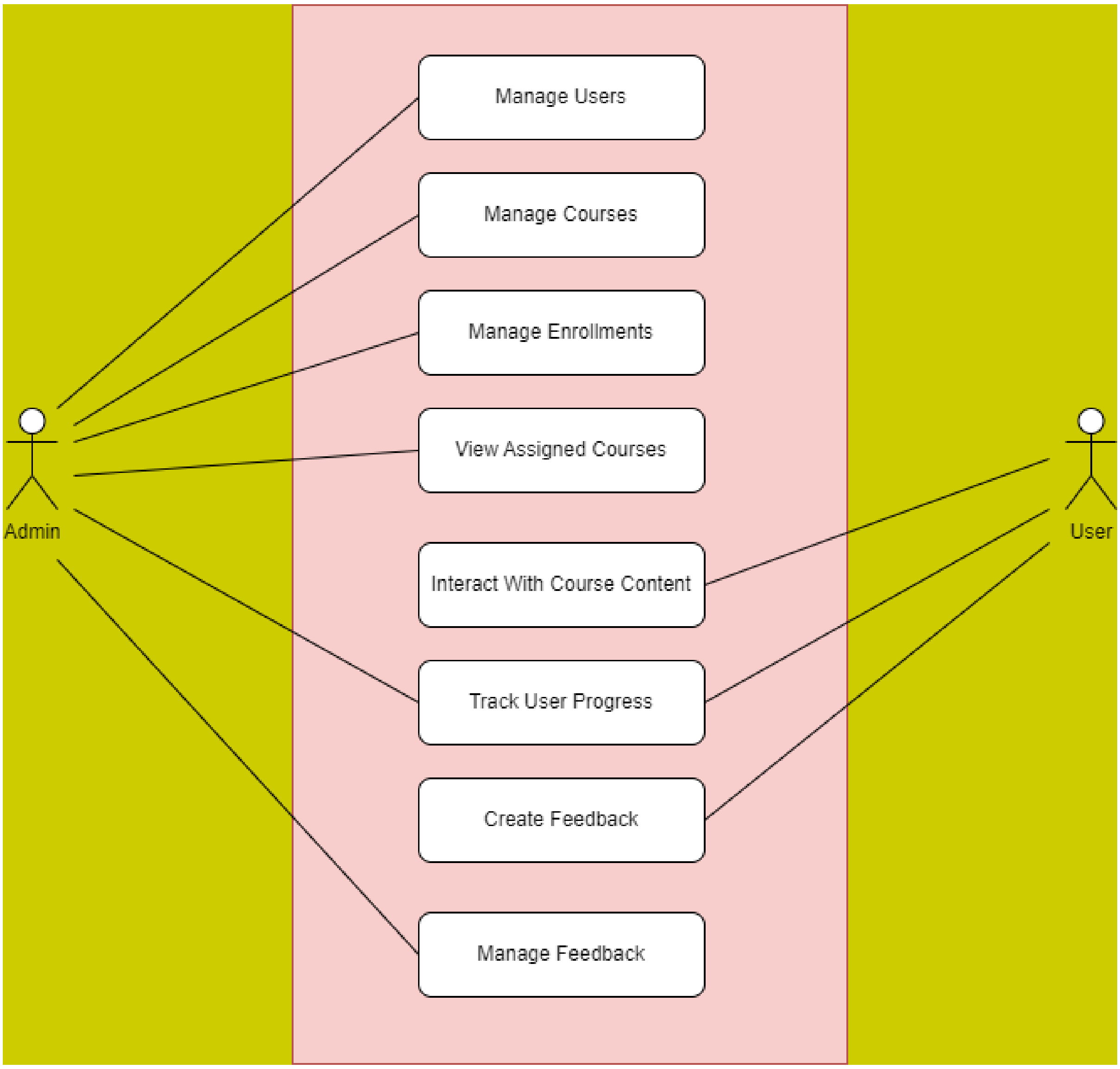

Figure 3: Use Case Diagram

10 Diagrams

10.1 Architecture Diagram

10.2 Class Diagram

10.3 Use Case Diagram

10.4 Sequential Diagram

10.5 Architecture Diagram

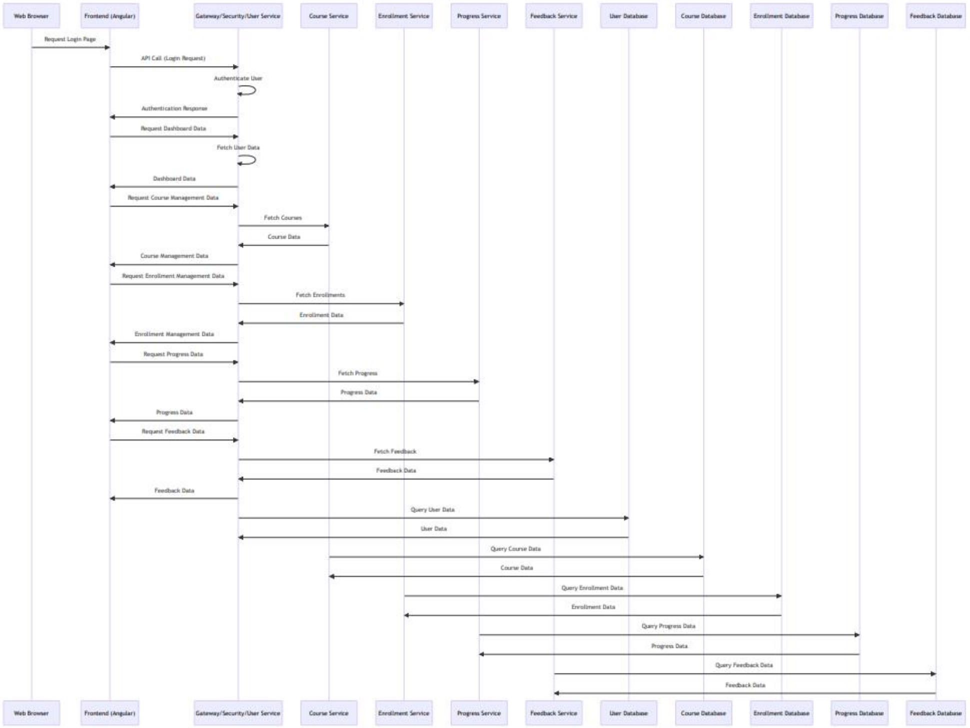

Figure 4: Sequential Diagram

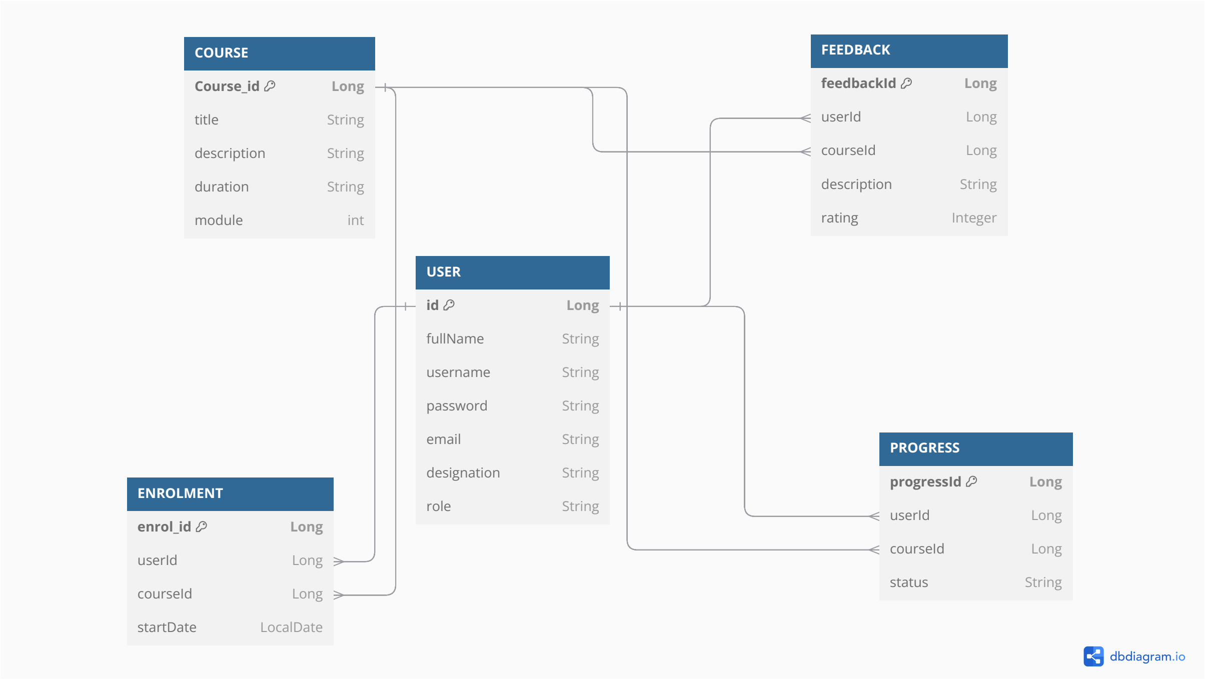

Figure 5: Sequential Diagram