# Course Management System - Project Documentation

## Project Overview

The Course Management System is a comprehensive RESTful web application designed to manage academic course data, student information, and their enrollment interactions. Built with Spring Boot and Java, it provides an enterprise-level architecture suitable for scalable and modular deployments.

The goal of the project is to provide a backend service where administrative personnel can:

- Add, update, retrieve, and delete course offerings.  
- Manage student records including their name and contact details.  
- Handle the enrollment of students in multiple courses, tracking enrollment dates.

The application uses MySQL as its backend database and adheres to RESTful architecture. HTTP methods like GET, POST, PUT, and DELETE are used to interact with the resources in a stateless and predictable manner.

## Technologies Used

- Java 21 (Programming Language)  
- Spring Boot (Backend Framework)  
- Spring Data JPA (ORM for database interaction)  
- MySQL (Relational Database)  
- Maven (Build and Dependency Management)  
- Postman (API Testing Tool)  
- IntelliJ IDEA (Integrated Development Environment)

Key dependencies (pom.xml):  
- spring-boot-starter-web  
- spring-boot-starter-data-jpa  
- mysql-connector-j

## Project Structure and Setup

The structure adheres to a layered architecture separating concerns clearly between model, repository, service, and controller layers.

Directory layout:  
course-management-system/  
├── src/  
│ └── main/  
│ ├── java/com/example/course/  
│ │ ├── controller/  
│ │ ├── service/  
│ │ ├── repository/  
│ │ ├── model/  
│ │ └── CourseManagementSystemApplication.java  
│ └── resources/  
│ └── application.properties  
└── pom.xml

Key steps followed:  
1. Generated Spring Boot project using Spring Initializr.  
2. Defined entity models using @Entity.  
3. Implemented CRUD repositories using JpaRepository.  
4. Developed services and controllers for each domain model.  
5. Configured MySQL credentials and Hibernate in application.properties.

## API Endpoints & HTTP Methods

RESTful endpoints with correct HTTP methods:

GET: Retrieve data – http://localhost:8080/api/courses /api/students/{id}  
POST: Create new data - http://localhost:8080/api/students  
PUT: Update data - http://localhost:8080 /api/courses/1  
DELETE: Remove data - http://localhost:8080/api/enrollments/2

Example usage:  
POST /api/courses  
{  
 "name": "Spring Boot Basics",  
 "description": "Intro to Spring",  
 "duration": "4 weeks",  
 "instructorName": "John Doe"  
}

POST /api/enrollments  
{  
 "studentId": 1,  
 "courseId": 1,  
 "enrolledDate": "2025-04-10"  
}

## Testing the API Using Postman

1. Launch the application from IntelliJ using CourseManagementSystemApplication.java.  
2. Ensure MySQL is running and the database 'course\_db' exists.  
3. Use Postman to test endpoints like:  
 - GET http://localhost:8080/api/courses  
 - POST http://localhost:8080/api/students  
 - POST http://localhost:8080/api/enrollments

## Snapshots:

