**Student Management API Documentation**

**Table of Contents**

1. [Project Overview](#project-overview)
2. [Prerequisites](#prerequisites)
3. Code Base
4. [Project Setup](#project-setup)
5. [Configuration](#configuration)
6. [Entity](#entity)
7. [Repository](#repository)
8. [Service](#service)
9. [Controller](#controller)
10. [API Endpoints](#api-endpoints)
11. [Running the Application](#running-the-application)
12. [Testing the API](#testing-the-api)

Project Overview

**This project implements a RESTful API for managing students using Spring Boot, Hibernate/JPA, and MySQL. The API allows you to create, read, update, and retrieve student information, including their date of birth and age.**

**Prerequisites**

* Java Development Kit (JDK) 17
* Maven for dependency management
* MySQL Database (installed and running)
* An IDE (e.g., IntelliJ IDEA, Eclipse)

**Codebase**

****

**Project Setup**

1. ***Create the Project: Use Spring Initializr to generate a Spring Boot project with the necessary dependencies.***
2. ***Extract the Project: Open the extracted project folder in your preferred IDE.***

**Maven Dependencies**

***pom.xml includes the following dependencies:***

***<dependencies>***

***<dependency>***

***<groupId>org.springframework.boot</groupId>***

***<artifactId>spring-boot-starter-web</artifactId>***

***</dependency>***

***<dependency>***

***<groupId>org.springframework.boot</groupId>***

***<artifactId>spring-boot-starter-data-jpa</artifactId>***

***</dependency>***

***<dependency>***

***<groupId>mysql</groupId>***

***<artifactId>mysql-connector-java</artifactId>***

***<scope>runtime</scope>***

***</dependency>***

***<dependency>***

***<groupId>org.projectlombok</groupId>***

***<artifactId>lombok</artifactId>***

***<scope>provided</scope>***

***</dependency>***

***<dependency>***

***<groupId>org.springframework.boot</groupId>***

***<artifactId>spring-boot-starter-test</artifactId>***

***<scope>test</scope>***

***</dependency>***

***</dependencies>***

**Configuration**

**Database Configuration**

***Update src/main/resources/application.properties to configure MySQL:***

***properties***

***spring.datasource.url=jdbc:mysql://localhost:3306/student\_db***

***spring.datasource.username=username***

***spring.datasource.password=your\_db\_password***

***spring.jpa.hibernate.ddl-auto=update***

***spring.jpa.show-sql=true***

API Endpoints

| HTTP Method | Endpoint | Description |
| --- | --- | --- |
| **POST** | **/students/create** | **Create a new student** |
| **GET** | **/students/all** | **Retrieve all students** |
| **GET** | **/students/age-range** | **Get students between 18 and 25 years old** |
| **PUT** | **/students/update-age/{id}** | **Update age of the student by ID** |

Request/Response Format

* Create Student:
  + **Request Body**:

**{**

**"name": "John Doe",**

**"dayOfBirth": 15,**

**"monthOfBirth": 5,**

**"yearOfBirth": 1998**

**}**

* + **Response**:

**{**

**"id": 1,**

**"name": "John Doe",**

**"dayOfBirth": 15,**

**"monthOfBirth": 5,**

**"yearOfBirth": 1998,**

**"age": 26**

**}**

* Get All Students:
  + **Response**:

**{**

**"id": 1,**

**"name": "John Doe",**

**"dayOfBirth": 15,**

**"monthOfBirth": 5,**

**"yearOfBirth": 1998,**

**"age": 26**

**}**

* Get Students Between Ages(18 and 25):
  + **Response**:

**{**

**"id": 1,**

**"name": "John Doe",**

**"dayOfBirth": 15,**

**"monthOfBirth": 5,**

**"yearOfBirth": 1998,**

**"age": 26**

**}**

* Update Student Age:
  + **Response**:

**{**

**"id": 1,**

**"name": "John Doe",**

**"dayOfBirth": 15,**

**"monthOfBirth": 5,**

**"yearOfBirth": 1998,**

**"age": 26**

**}**

Running the Application

1. Make sure MySQL is running and the database student\_db is created.
2. Run the Spring Boot application from your IDE or use the command line:

mvn spring-boot:run

1. The application will start on http://localhost:8080.

Testing the API

You can use **Postman** or **curl** to test the API endpoints.

1. **Create a Student**:  
   Use a POST request to http://localhost:8080/students/create with the request body.
2. **Get All Students**:  
   Use a GET request to http://localhost:8080/students/all.
3. **Get Students Between Ages**:  
   Use a GET request to http://localhost:8080/students/age-range.
4. **Update Student Age**:  
   Use a PUT request to http://localhost:8080/students/update-age/{id}.