A PROJECT REPORT ON

STUDENT MANAGEMENT SYSTEM

Submitted by

Ms. Prajkta Iratkar

Ms. Mohini Chavan

Ms. Sakshi Kulkarni

Ms. Vaibhavi Kaware

Ms. Madhavi Varpe

Batch No. 6521

Under the Guidence of

Trainer Mrs. Indrakka Mali Mam

INDEX

SR NO	Topic Name
1	Introduction
2	Objective
3	Software Requirements
4	System overview and snapshots
5	Conclusion

Introduction

The project “Student Management System” is developed using spring boot framework, which mainly focuses on basic operations of student information system. Like Inserting, Deleting, Updating and getting all records of students introduction.

Student Module:

- In the student module the students can perform :
- Fetch all student’s records.
- Fetch student record by student id.
- Fetch student record by student email.
- Fetch student record by student name.
- One student can eligible to apply for many courses.

Course Module:

- In the Course Module :
- We can fetch all the course details as well as student details.
- We can get the student details and delete the record by using course id.

Objectives:

- It provides “better and efficient” service”.
- Faster way to get information about the students.
- Provide facility for proper monitoring and reduce paper work.
- All details will be available on a click

System Overview:

- The Student management system will be automated the traditional system.
- There is no need to use paper and pen.
- Checking a student details is very easy.
- Adding new student record is very easy.
- Deleting or updating a record of a particular student is simple.

Requirements:

Software Requirement:

Database: MySQL

API- Spring Data JPA, spring web, spring security

Tools: Postman, IDE-Spring Tool Suit4

Coding language-Java 1.8

Hardware Requirements:

RAM: 2GB

Processor: 64bit

Memory: 512 MB

Disk Space: 100GB

Spring Tool Suit: STS is an Eclipse-based development environment that is customized for the development of spring applications.

It provides a ready-to-use environment to implement, debug, run and deploy your applications**.**

Postman: Postman is a standalone software testing API (Application Programming Interface) platform to build, test, design, modify, and document APIs. It is a simple Graphic User Interface for sending and viewing HTTP requests and responses.

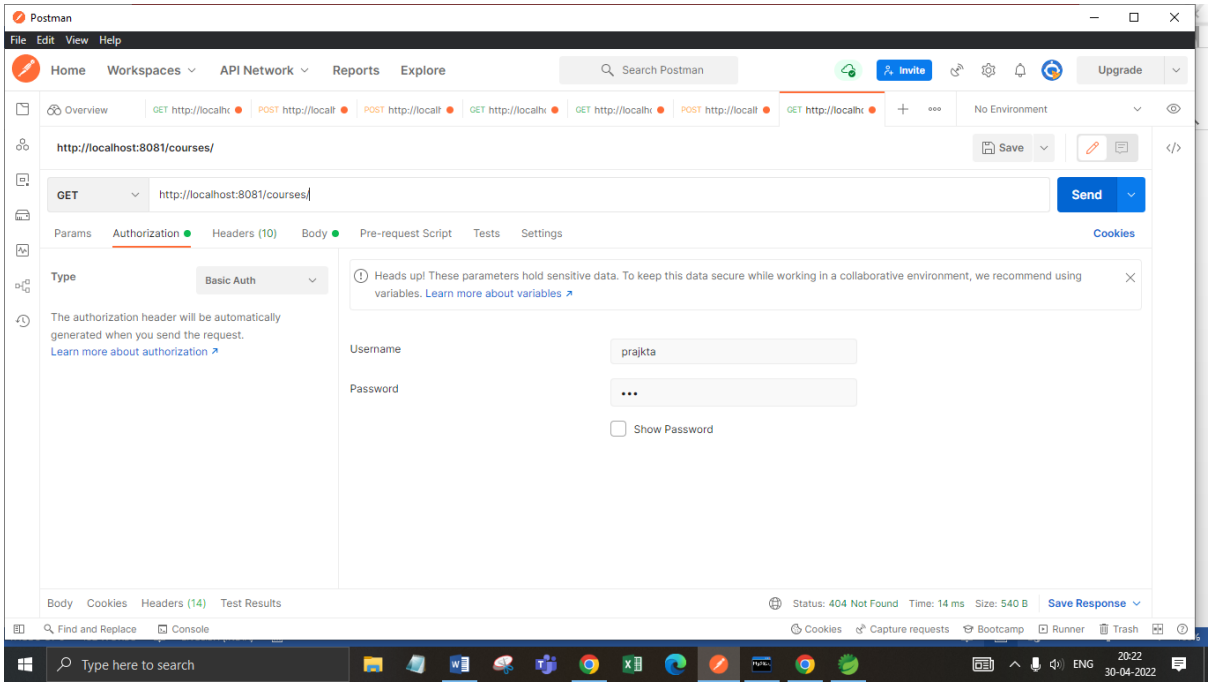
MySQL:

- MySQL is a relational database management system
- MySQL is open-source
- MySQL is free
- MySQL is ideal for both small and large applications
- MySQL is very fast, reliable, scalable, and easy to use
- MySQL is cross-platform

Screenshot

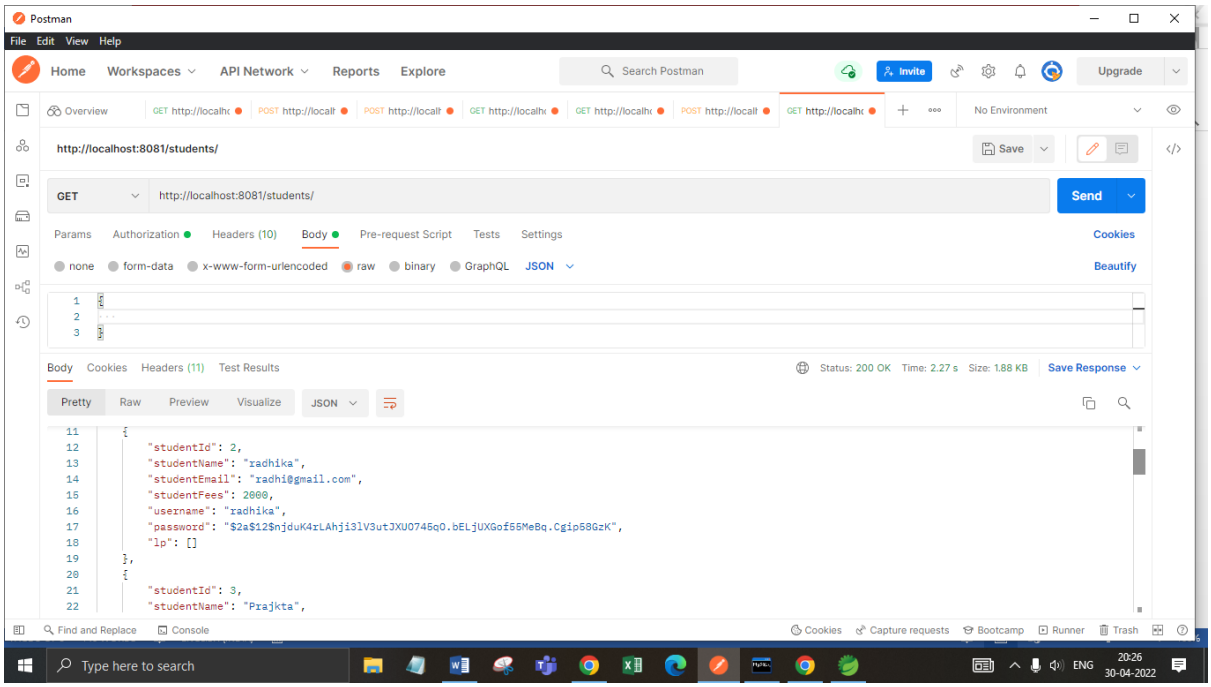
Step 1: Student can login with their username and password.

URL: http://localhost:8081/students/



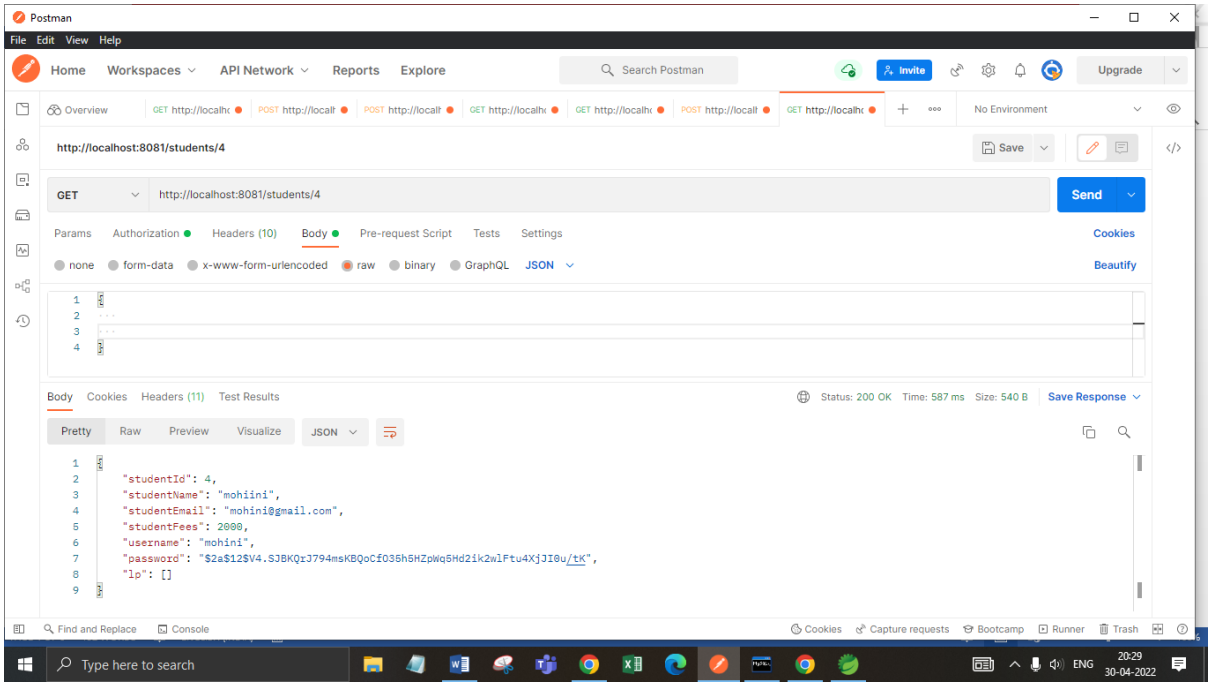
Step 2: Get all students Record.

URL: http://localhost:8081/students/



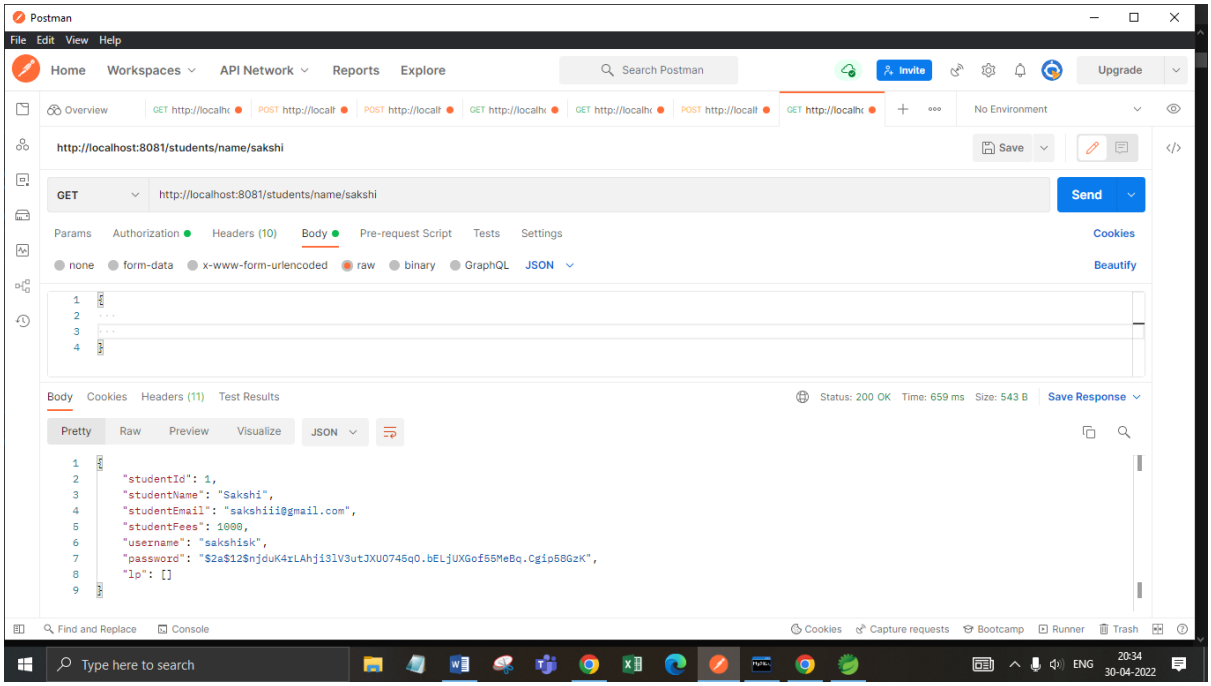
Step 3: Get particular student record by student id.

URL: http://localhost:8081/students/4



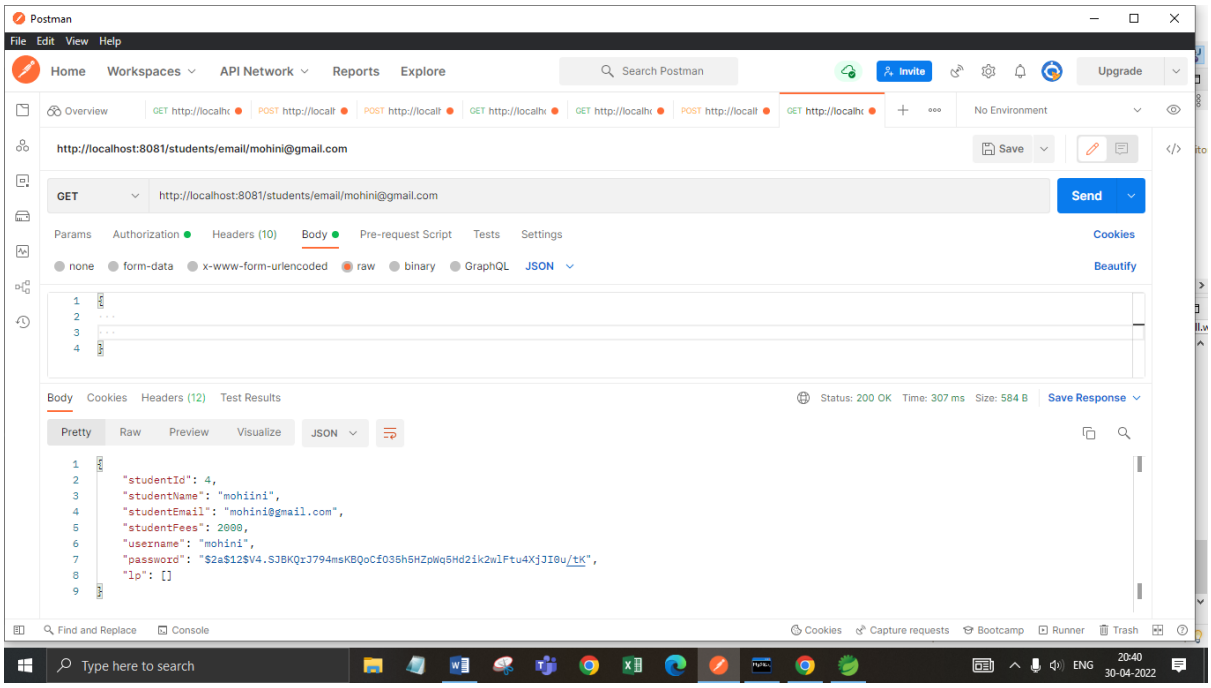
Step 4: Get particular student record by student name.

URL: http://localhost:8081/students/name/sakshi



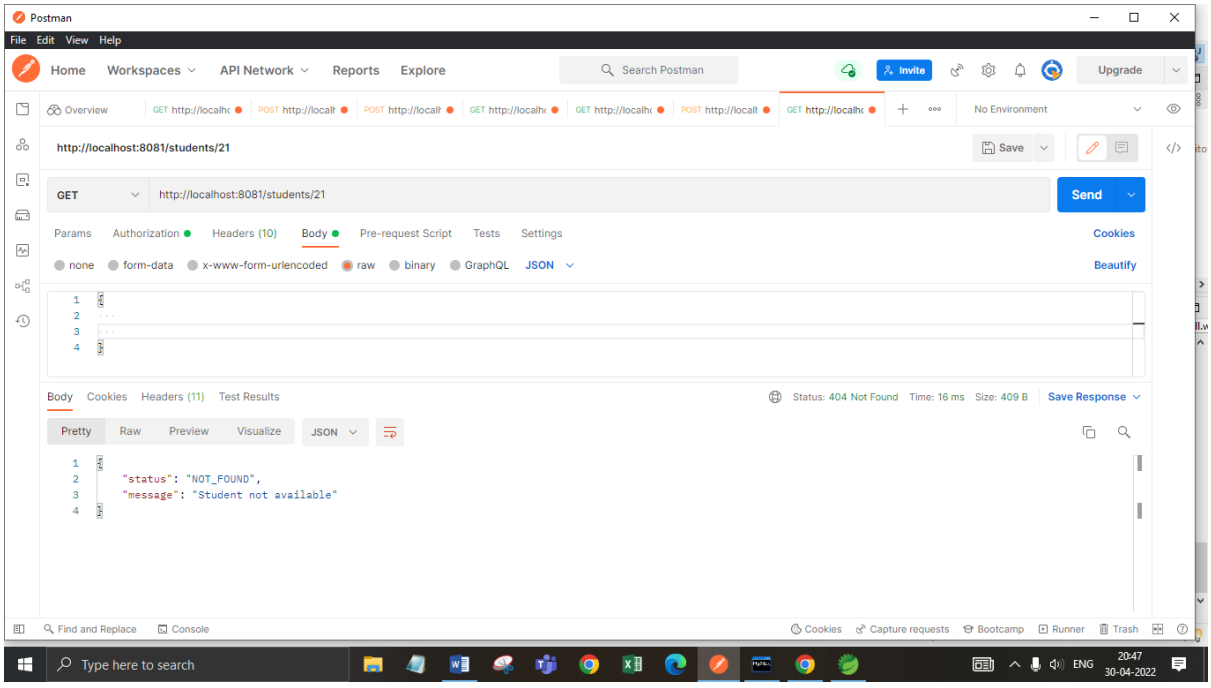
Step 5: Get particular student record by student email.

URL: http://localhost:8081/students/email/mohini@gmail.com



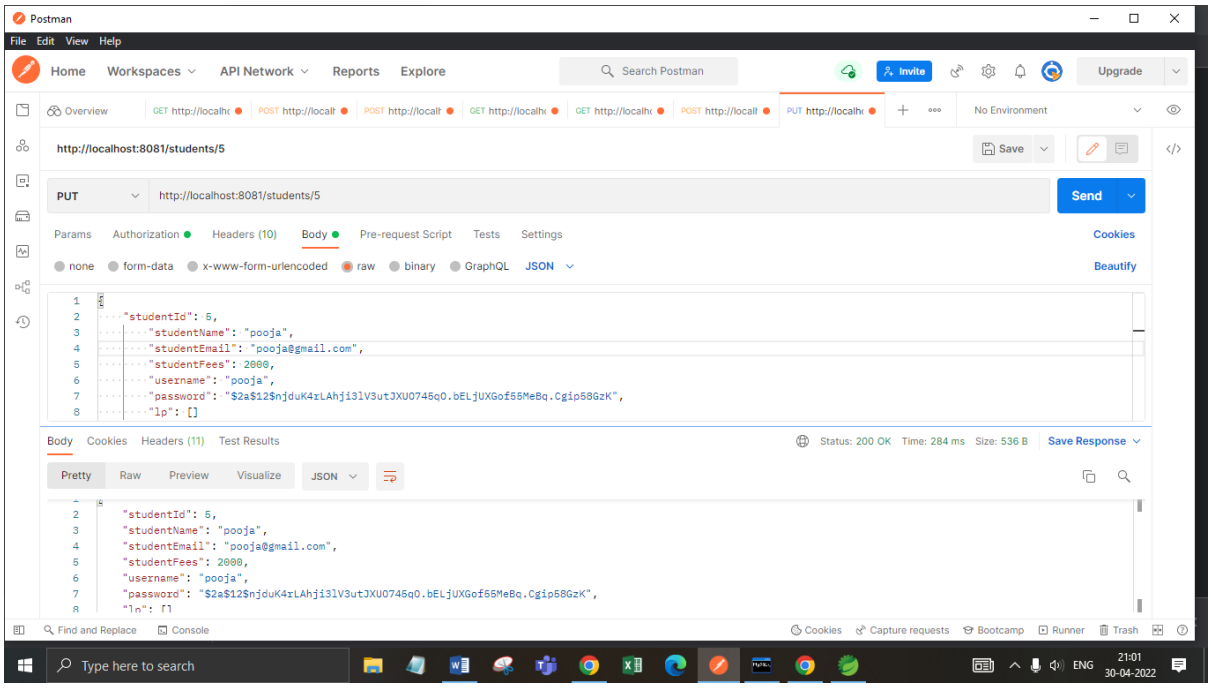
Step 6: If we want to show unavailable student with their id then it shows exception like “student not available”.

URL: http://localhost:8081/students/21



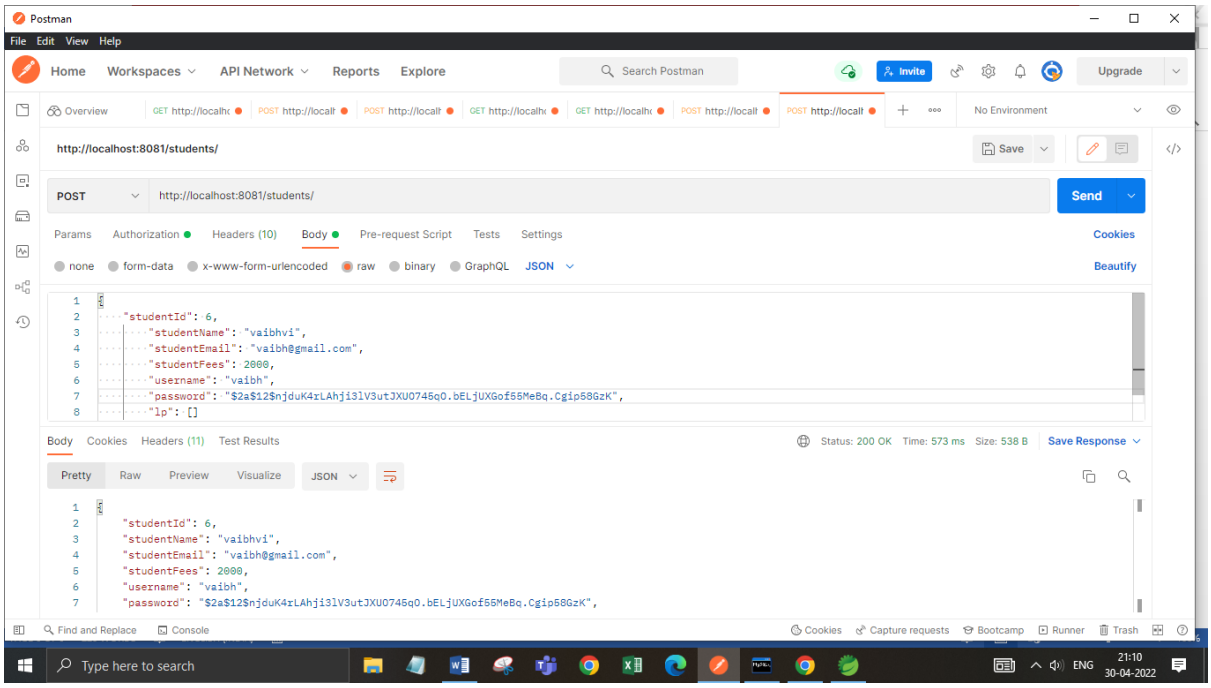
Step 7: We can update student record using put method.

URL: http://localhost:8081/students/5



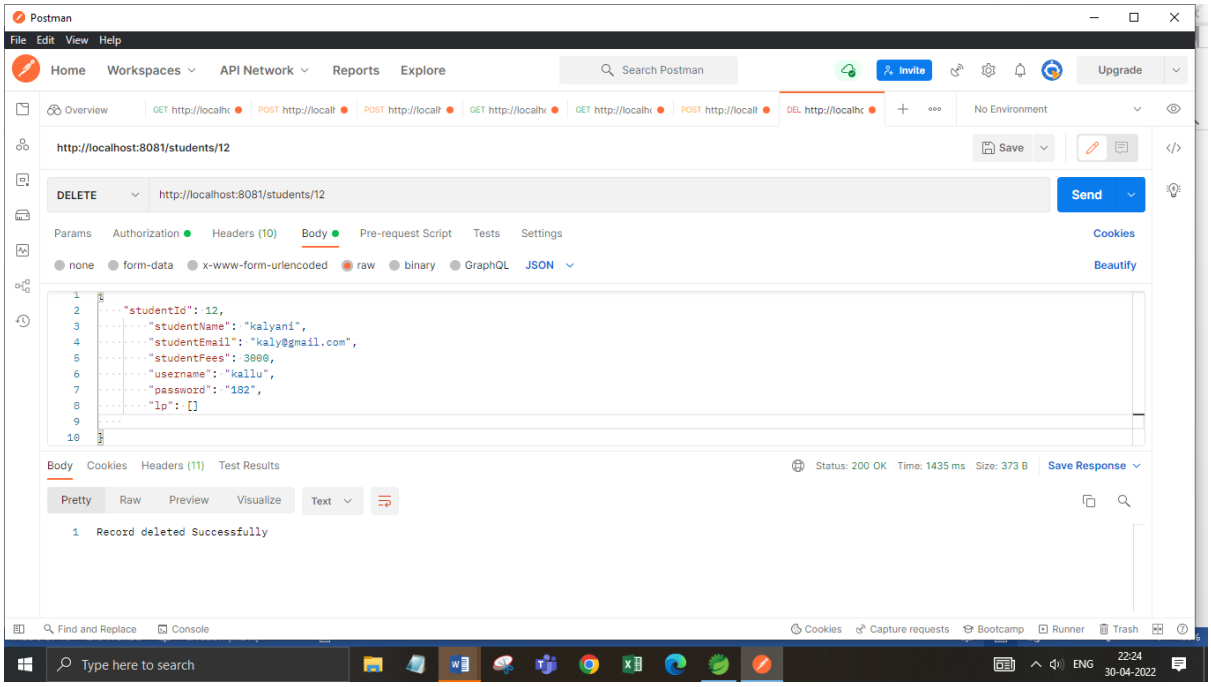
Step 8: We can add student record using post method.

URL: http://localhost:8081/students/



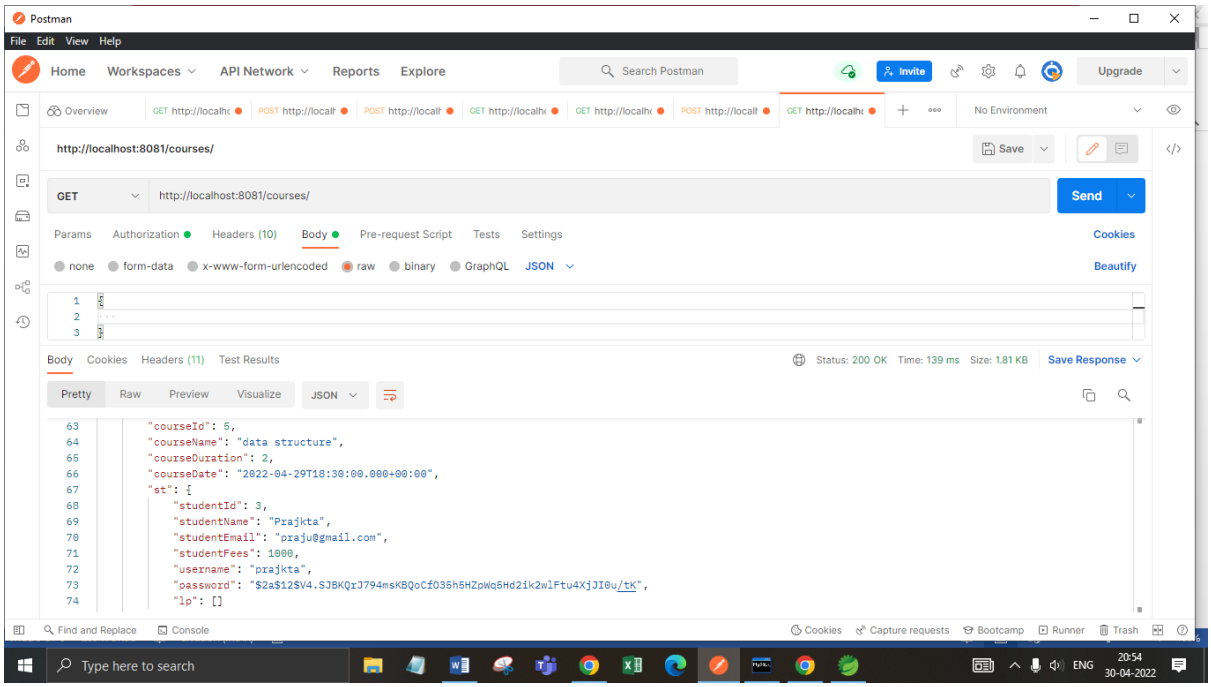
Step 9: We can delete student record using delete method.

URL: http://localhost:8081/students/12



Step 10: Get all courses.

URL: http://localhost:8081/courses/



Annotations:

1.@Service:

We mark beans with @Service to indicate that they're holding the business logic. Besides being used in the service layer, there isn't any other special use for this annotation.

2. @Repository:

@Repository**'**s job is to catch persistence-specific exceptions and re-throw them as one of spring's unified unchecked exceptions**.**

*3. @Autowired***:**

The spring framework enables automatic dependency injection. In other words, by declaring all the bean dependencies in a spring configuration file, Spring container can autowire relationships between collaborating beans. This is called *spring bean autowiring*.

4. @GetMapping:

The *@GetMapping* annotation is a specialized version of @RequestMapping annotation that acts as a shortcut for @RequestMapping(method = RequestMethod.GET).

5. @PostMapping:

The *@PostMapping* is specialized version of @RequestMapping annotation that acts as a shortcut for @RequestMapping(method = RequestMethod.POST).

The @PostMapping annotated methods in the *@Controller* annotated classes handle the HTTP POST requests matched with given URI expression.

6. @Configuration:

Spring @Configuration annotation helps in spring annotation based configuration**.** @Configuration annotation indicates that a class declares one or more @Bean methods and may be processed by the spring container to generate bean definitions and service requests for those beans at runtime.

7. @OneToMany:

A one-to-many relationship between two entities is defined by using the @OneToMany annotation in Spring Data JPA. It declares the mappedBy element to indicate the entity that owns the bidirectional relationship. Usually, the child entity is one that owns the relationship and the parent entity contains the @OneToMany annotation.

8. @generated Value:

Marking a field with the @GeneratedValue annotation specifies that a value will be automatically generated for that field. This is primarily intended for primary key fields but Object DB also supports this annotation for non-key numeric persistent fields as well.

9. @entity:

The @Entity annotation specifies that the class is an entity and is mapped to a database table. The @Table annotation specifies the name of the database table to be used for mapping.

10. @ Exception Handler:

The @ExceptionHandler is an annotation used to handle the specific exceptions and sending the custom responses to the client.

11. @ControllerAdvice:

@ControllerAdvice is a specialization of the @Component annotation which allows to handle exceptions across the whole application in one global handling component. It can be viewed as an interceptor of exceptions thrown by methods annotated with @RequestMapping and similar.

12. @BeanAnnotation:

Spring @Bean annotation tells that a method produces a bean to be managed by the spring container. It is a method-level annotation. During Java configuration (@Configuration), the method is executed and its return value is registered as a bean within a BeanFactory.

Database Table Design:

Student Table

Si NO	Field Name	Data Type
1	ID	Int
2	Name	String
3	Email	String
4	Fees	String
5	Username	String
6	Password	String

Conclusion: Student Management System make students jobs more accessible by giving them an easy place to find and sort information.

THANK YOU!!