TASK1

JAVA REST API EXAMPLE

Student Information System

Using Spring Boot and MongoDB

Name: Nishtha Singh (19BCE0037)

Email: nishtha.singh2019@vitstudent.ac.in

College: VIT VELLORE

About The Project

The project involves building a REST service using Spring Boot and MongoDB, which allows users to manage student information through a set of API endpoints. The service is designed to support various CRUD operations on student data. The student's collection in the MongoDB database contains information about each student's name, student number, email, course list, and GPA.

The project follows a structured approach, beginning with setting up the development environment by installing the required tools such as JDK, IntelliJ, Maven, MongoDB, and Postman. The project uses the Spring Initializer to set up a basic Spring Boot project and add necessary dependencies.

The next step involves defining the model for the student object, creating a repository interface to communicate with the MongoDB database, and implementing a service layer to handle business logic.

The service layer includes methods for retrieving student information by student number or email, fetching all students, adding new students to the system, updating student information, and deleting students

from the system. The project also includes data validation and error handling, caching and performance optimization, security measures such as authentication and authorization, and logging and monitoring.

To test the REST service, the project uses tools such as Postman, which allows users to send HTTP requests to the API endpoints and receive responses. In summary, the project demonstrates how to build a functional REST service using Spring Boot and MongoDB, emphasizing simplicity and a structured approach to implementing the business logic.

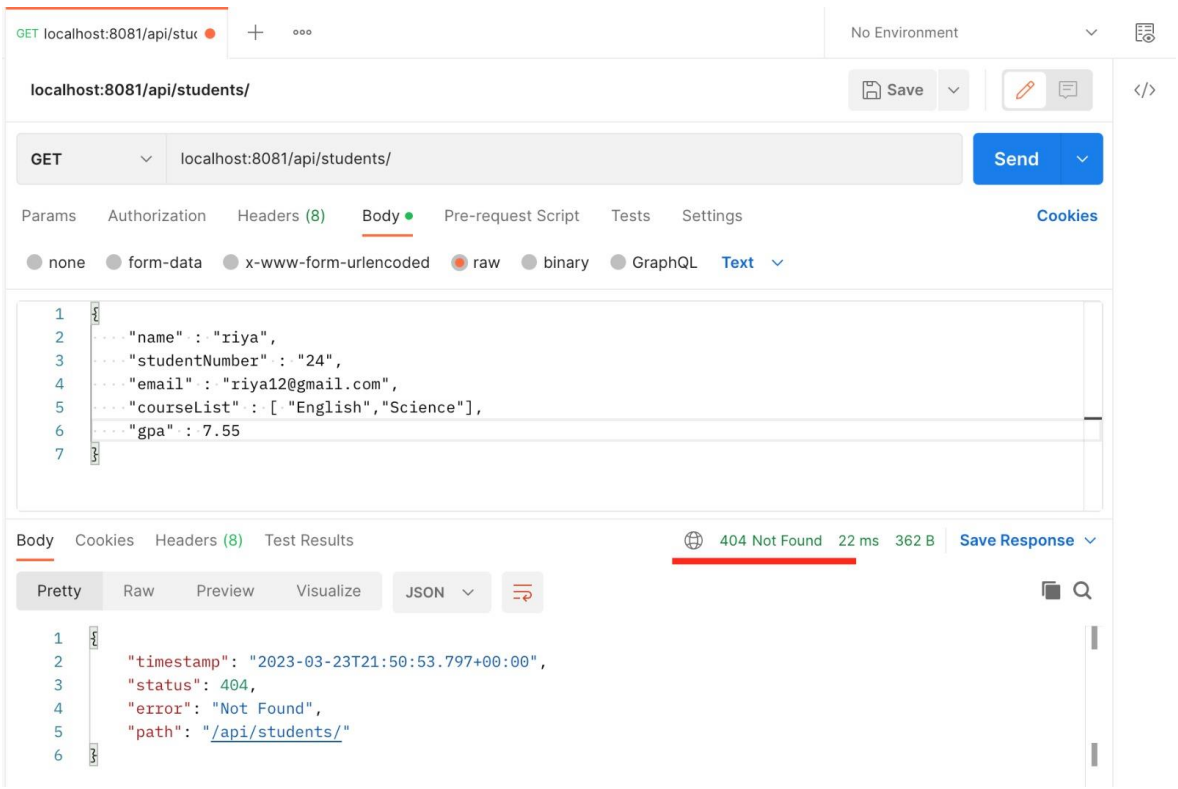
Steps

1. Install JDK 8.0, IntelliJ, Maven, MongoDB, and Postman.
2. Use Spring Initializer to create a new Spring Boot project.
3. Create a model and repository for the "Student" class.
4. Create a StudentService that implements 6 methods for CRUD operations.
5. Now, the student service will be called by student controller. Simply, these methods list students according to some criteria, save a student, update a student and delete a student.
6. Create a REST controller with @GetMapping, @PostMapping, and @DeleteMapping annotations.
7. Next is Database configuration for connecting to Mongo dB database
8. To test the REST service in a Java project, a tool like Postman is used to send HTTP requests and receive responses.

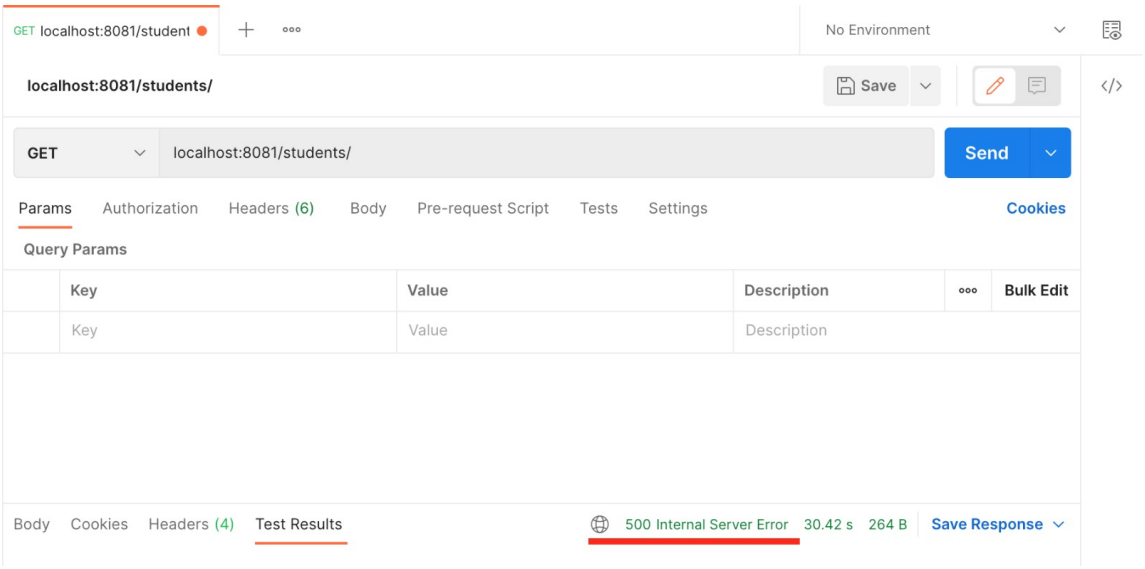
Results

GET REQUESTS

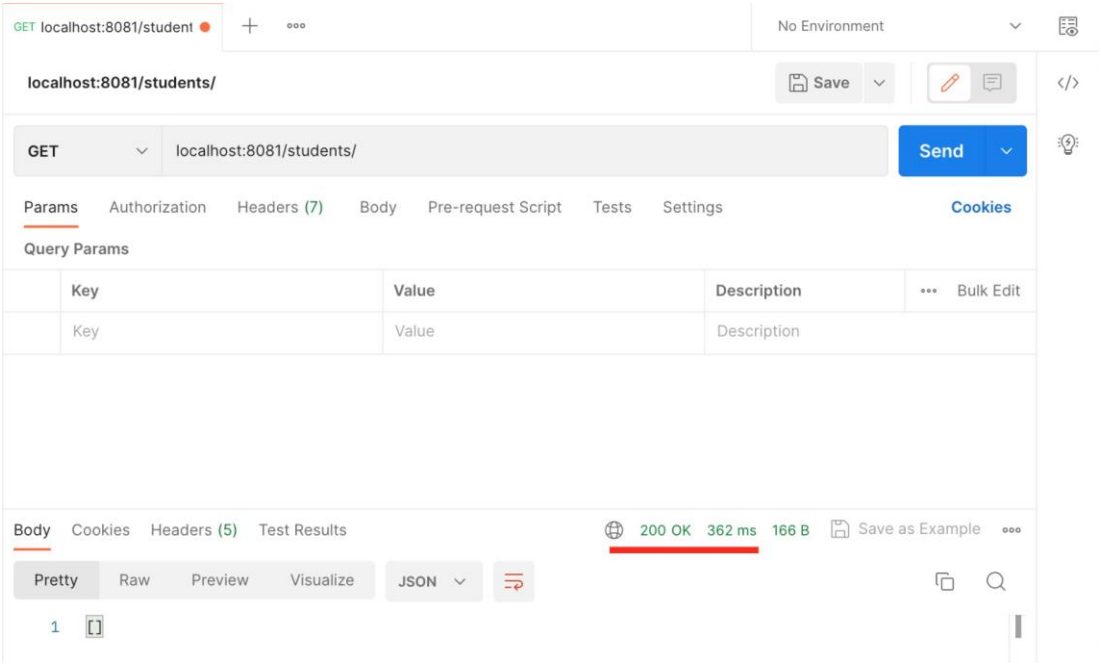
404 – Server Doesn't exist as URL is wrong



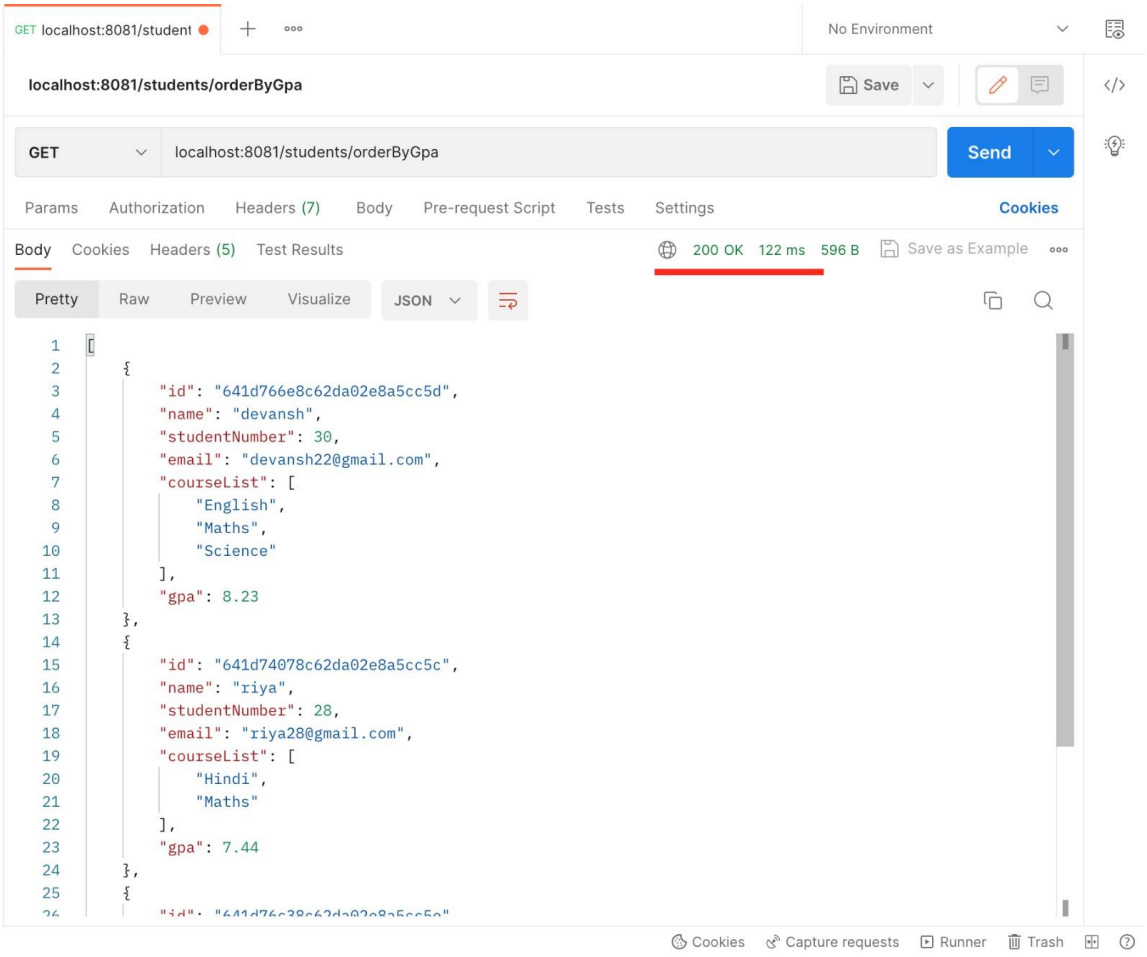
500 – Internal Server Error



200 ok – Empty Student List

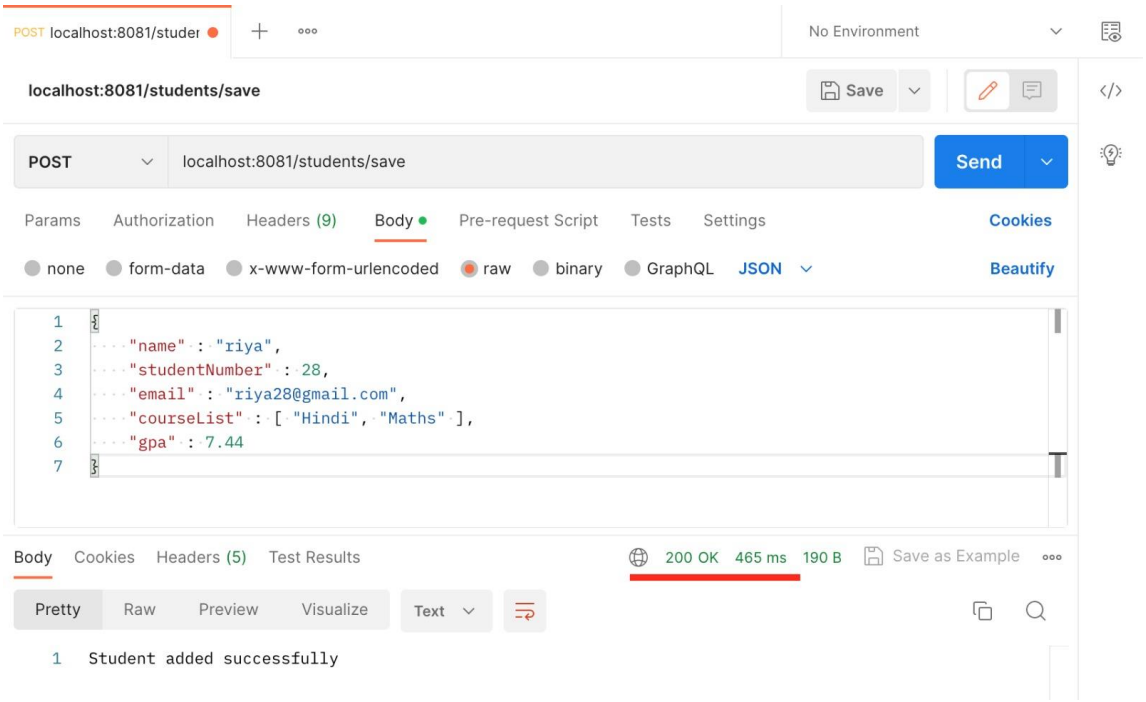


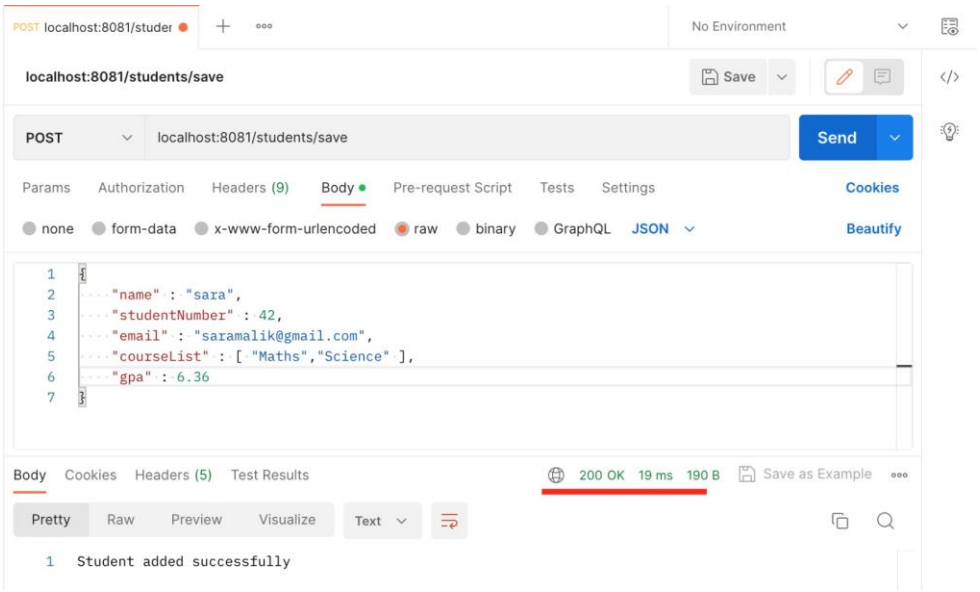
200 ok – Order students by their GPA



POST REQUESTS

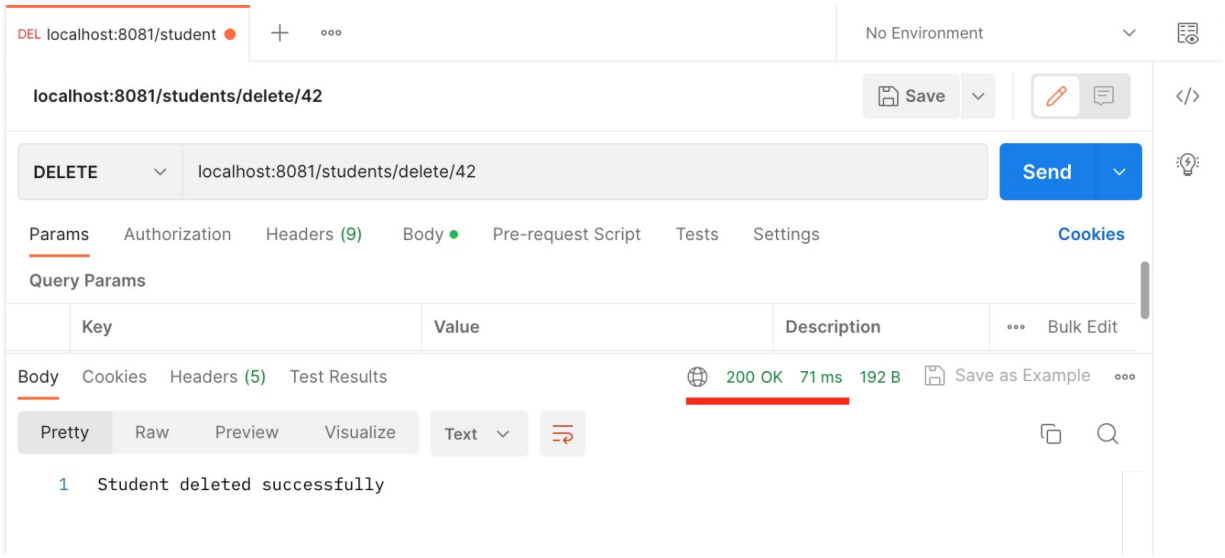
200 ok – Adding a student

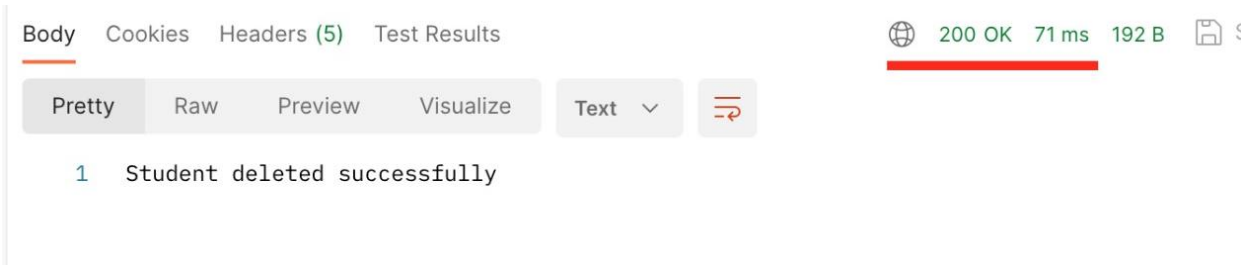




DELETE REQUESTS

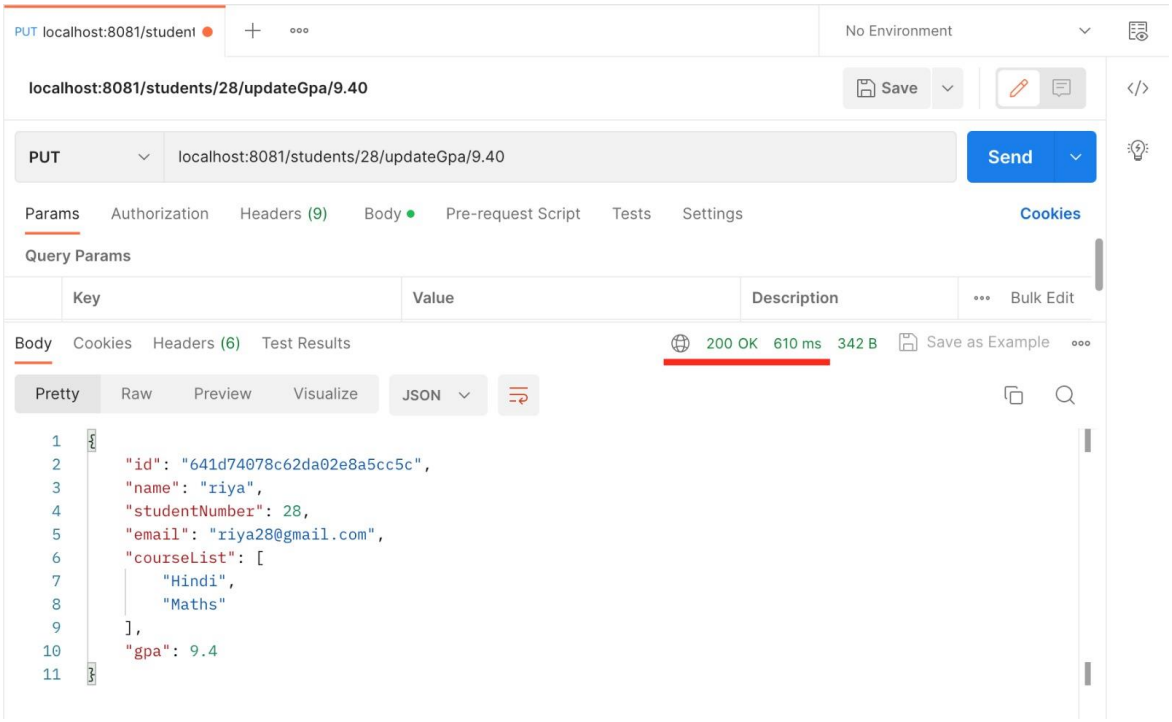
200 ok – Deleting student by their studentNumber



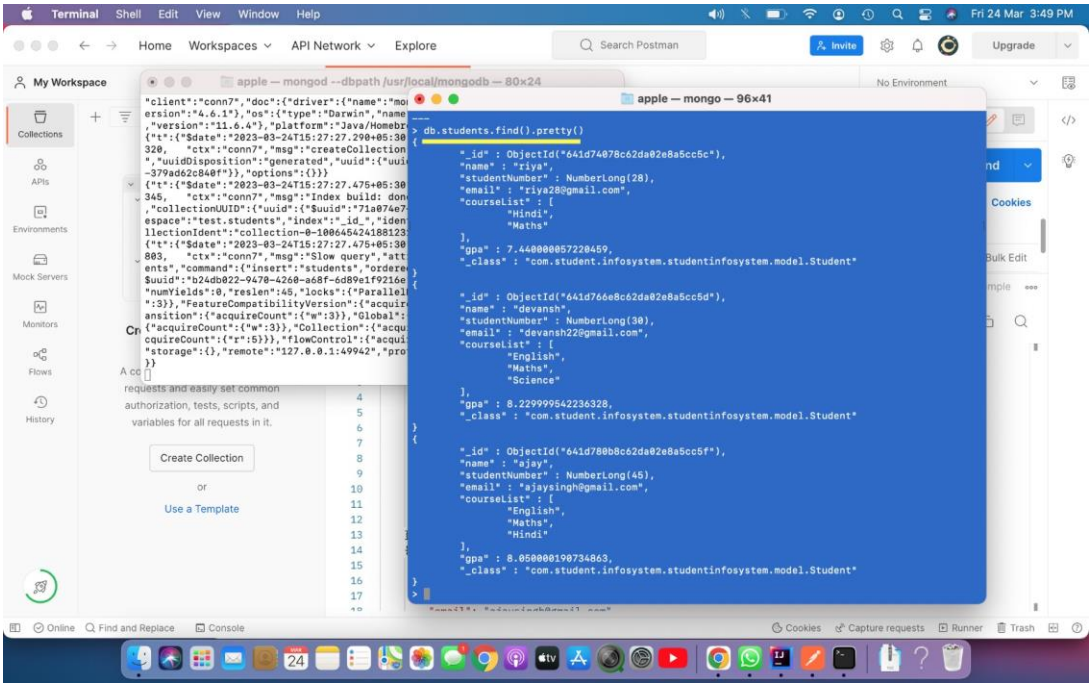


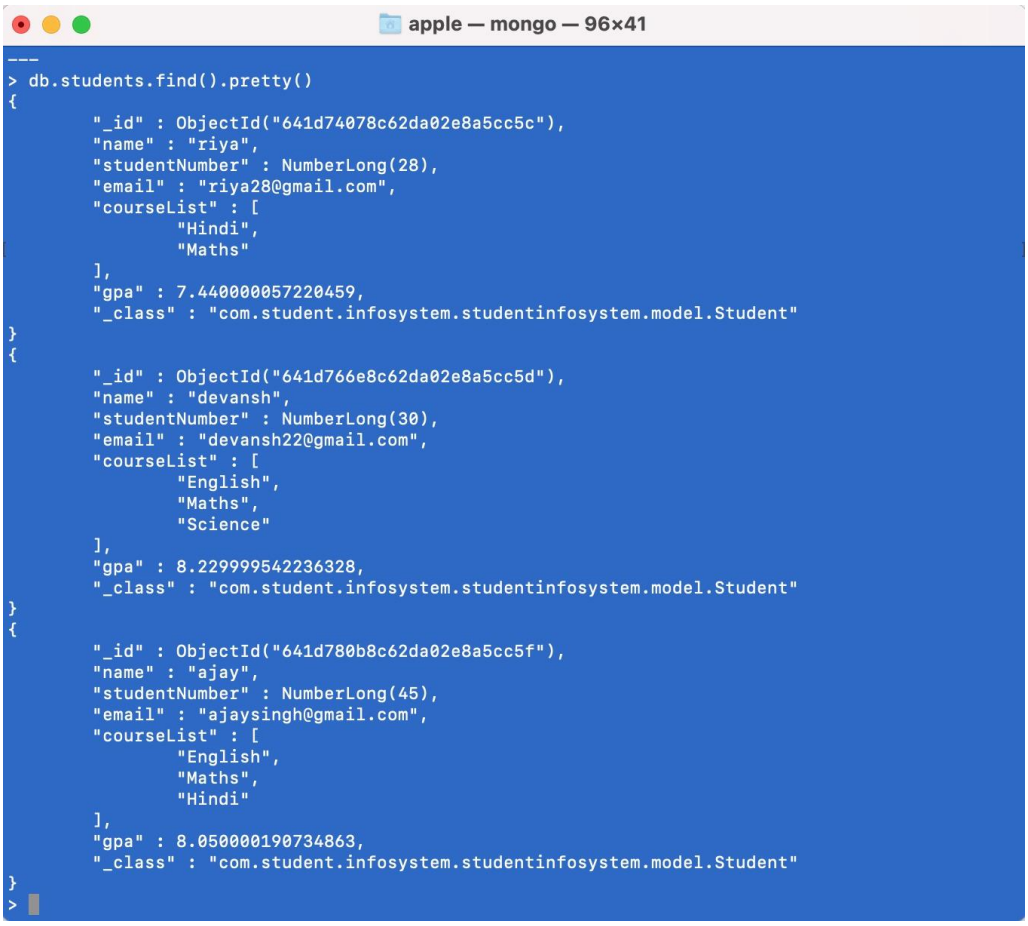
PUT REQUESTS

200 ok – Updating student's GPA by their studentNumber



THE STUDENTS COLLECTION




apple — mongo — 96×41

> db.students.find().pretty()
{
"_id" : ObjectId("641d74078c62da02e8a5cc5c"),
"name" : "riya",
"studentNumber" : NumberLong(28),
"email" : "riya28@gmail.com",
"courseList" : [
"Hindi",
"Maths"
],
"gpa" : 7.440000057220459,
"_class" : "com.student.infosystem.studentinfosystem.model.Student"
}
{
"_id" : ObjectId("641d766e8c62da02e8a5cc5d"),
"name" : "devansh",
"studentNumber" : NumberLong(30),
"email" : "devansh22@gmail.com",
"courseList" : [
"English",
"Maths",
"Science"
],
"gpa" : 8.229999542236328,
"_class" : "com.student.infosystem.studentinfosystem.model.Student"
}
{
"_id" : ObjectId("641d780b8c62da02e8a5cc5f"),
"name" : "ajay",
"studentNumber" : NumberLong(45),
"email" : "ajaysingh@gmail.com",
"courseList" : [
"English",
"Maths",
"Hindi"
],
"gpa" : 8.050000190734863,
"_class" : "com.student.infosystem.studentinfosystem.model.Student"
}
>