UNIVERSITY OF SRI JAYEWARDENEPURA

Faculty of Technology

Department of Information and Communication Technology

ITS 4243 - Microservices and Cloud Computing

Assignment 01

Name: E.D.N. JAYAMAL

Index NO: ICT/21/857

1. **What is Spring Boot and why is it used?**

Spring Boot is a Java framework. It is designed to make the development of Spring based applications easier and faster. Normally, Spring based projects requires lots of configurations but Spring Boot reduces these configurations because it gives auto configuration, starter packages and an embedded server (like Tomcat).

Benefits of using Spring Boot

- It makes setup very simple
- Reduces boilerplate code
- We can run the application with just one command (run)
- It has built in tools to handle security, database, web APIs and etc.

2. **Explain the difference between Spring Framework and Spring Boot.**

Spring Framework	Spring Boot
This is the core framework for Java enterprise applications.	Built on the top of Spring Framework.
Needs a lot of XML/Java based configurations.	Has auto configurations. Therefore, less manual work.
No built-in server. (need to deploy manually)	Has built-in servers. (Tomcat/Jetty)
Have lots of manual work.	Uses Starter Dependencies to reduce complexity.
	Focus on rapid development.

3. **What is Inversion of Control (IoC) and Dependency Injection (DI)?**

IoC means, we give the control for creating objects to the Spring framework instead of creating it manually. IoC is the core of Spring framework. It creates and manages objects. Normally in Java we create the objects so we have the control of the object but in in IoC the Spring creates, manages the objects and give the object to the place where we need it.

DI is used by Spring to give (inject) the required object into a required class. As in Java, we do not need to write manually the object in a class. Using "new" keyword. When the IoC controls the object creation, the DI injects the objects when those are needed.

4. **What is the purpose of application.properties / application.yml?**

application.properties / application.yml files store the configuration settings for Spring Boot projects. These files help to change the behavior of the application without changing the Java code. We need these files because every application need settings such as,

- Port the server run on
- How to connect to the database

- Logging levels
- Security settings
- Email configurations
- Custom variables used in the project

Instead of writing the above settings in the same code, it can write in these files.Spring Boot supports above given two formats which are application.properties and application.yml. application.properties format uses key=value format and application.yml uses a cleaner YAML format.

application.properties – key=value format

```
server.port=8080
```

application.yml – YAML format

```
server:
    port: 8080
```

5. Explain what a REST API is and list HTTP methods used.

REST APIs are used in applications to communicate with each other over the internet using HTTP. It usually sends data in JSON format. In simple terms, REST API is like a messenger between the client and the server (Spring Boot). When the client asks for data, the server sends it back in JSON format.

Benefits

- Easy to understand
- Work with almost any frontend
- Use simple HTTP
- Return useful data in JSON
- Make backend and frontend separate

HTTP methods used in REST APIs

HTTP method	Meaning	Use
GET	Fetch data	Reading data (get all students)
POST	Send data to server	Creating new records (add a student)
PUT	Replace existing data	Updating entire record (update full student details)
PATCH	Update part of the data	Updating only one field (update email only)
DELETE	Remove data	Deleting a record (delete a student)

6. **What is Spring Data JPA? What is an Entity and a Repository?**

Sprint Data in JPA is a Spring module which helps to work with databases easily. Normally we write long SQL queries manually when a database is used. But with Spring DATA JPA, we do not need to write most SQL queries manually because it provides built-in methods such as,

- save() → to save data
- findAll() → to get all data
- findById → to get one record
- deleteById() → to delete a record

So, Spring DATA JPA reduce SQL codes, work with databases faster and automatically map Java objects to database tables.

An **Entity** a Java class which is mapped to a table in the database. Each object becomes a row in the table.

Ex:

```
public class Student {
        private Long id;
        private String name;
}
```

Student class → student table → The Student class is the entity when it becomes a database table.

A **Repository** is an interface which is used to perform database operations such as save, update, delete and read. It communicates with the entities and database internally. We don't need to write any implementation and Spring generates everything automatically.

7. **What is the difference between @Component, @Service, @Repository, @Controller, @RestController?**

	What it does	When it is used
@Component	Marks a class as a Spring bean (generic component)	When the class does not fit to service, repository or controller categories
@Service	Indicate that the lass includes business logic and also helps with readability and organization	Use for classes that handle processing, calculations and business rules.
@Repository	Handles database operations and converts database	Use when working with database tables through JPA

	exceptions into Spring exceptions.	(Java Persistence API)/Hibernate
@Controller	Handles HTTP request and returns view (HTML pages)	Use for traditional web applications where you return JSP (JavaServer Page)/HTML
@RestController	Handles HTTP request and returns JSON/XML instead of views	Use when creating REST APIs for frontend/mobile apps.

8. What is @Autowired? When should we avoid it?

@Autowired is a Spring annotation which automatically injects the required object into a class. Instead of using "new" keyword for creating objects manually, Spring does it using @Autowired.

When to avoid

The following is how @autowired injects dependencies into a class.

```
userServerice userService;
```

- It is hard to test
- Not good for immutability
- Cases issues when refactoring

Instead of @Autowired method, constructor injection method is used.

Ex:

```
public UserController(UserService service) {
        this.service = service;
}
```

9. Explain how Exception Handling works in Spring Boot (@ControllerAdvice).

Exception handling in Spring Boot is when something goes wrong in the application such as missing record, invalid input or server error an exception is thrown. If we cannot handle it properly, the application may crash and user sees a messy error in the page. Spring boot give a way to handle this exception, so we can return a friendly error message and proper HTTP status codes.

@ControllerAdvice is a special Spring annotation used to handle exceptions for all controllers in one place. It helps to centralize exception handling instead of writing try-catch blocks in every controller.

- Catch error (Ex: Record not found)
- Return a proper message
- Send correct HTTP status (400, 404, 500)

- Avoid showing technical error messages to the user

10. What is the role of Maven/Gradle in a Spring Boot project?

Maven and Gradle are build automation tools. These tools help to manage and build Java projects automatically. Instead of doing things such as compiling, downloading, libraries, packaging and many more, these tools can do it automatically.

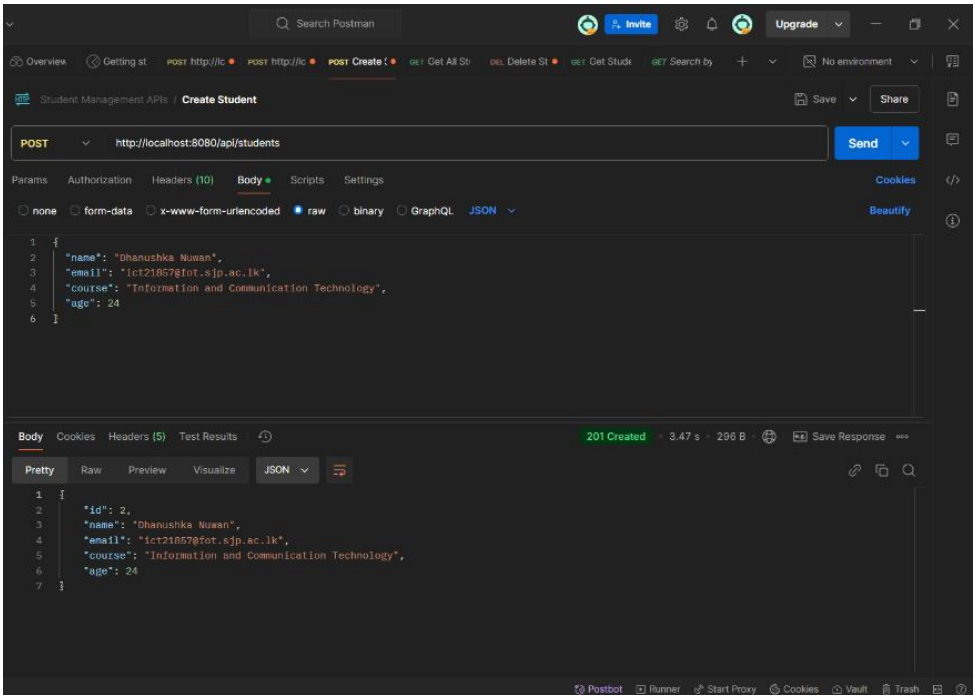
Maven and Gradle,

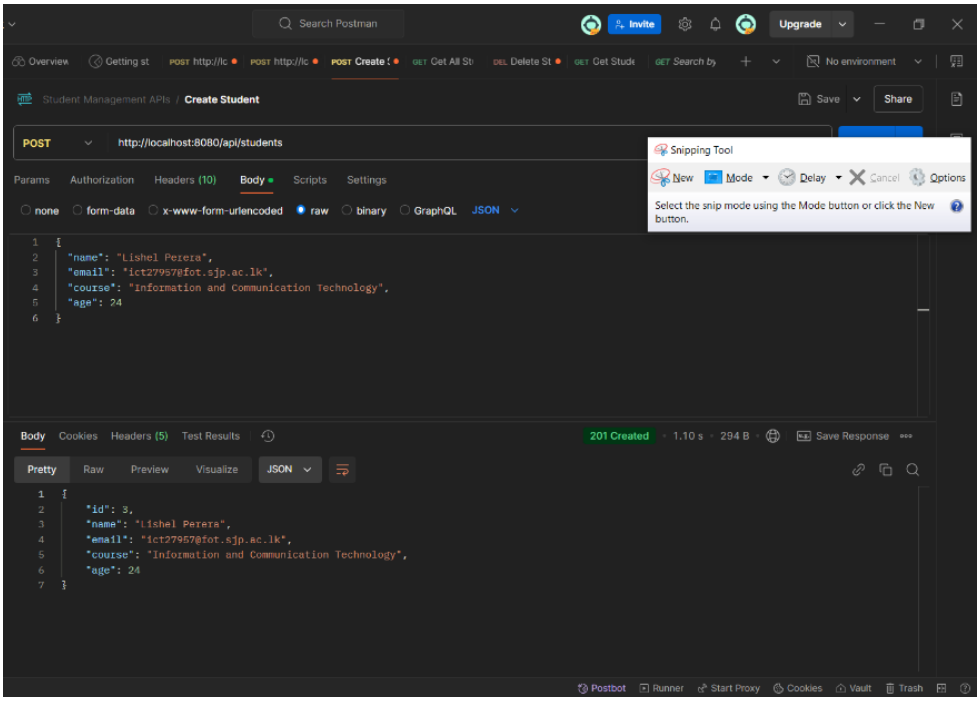
- Manage dependencies
- Build and compile the project
- Automate tasks
- Keep the project organized

Screenshot(s) of Postman Testing

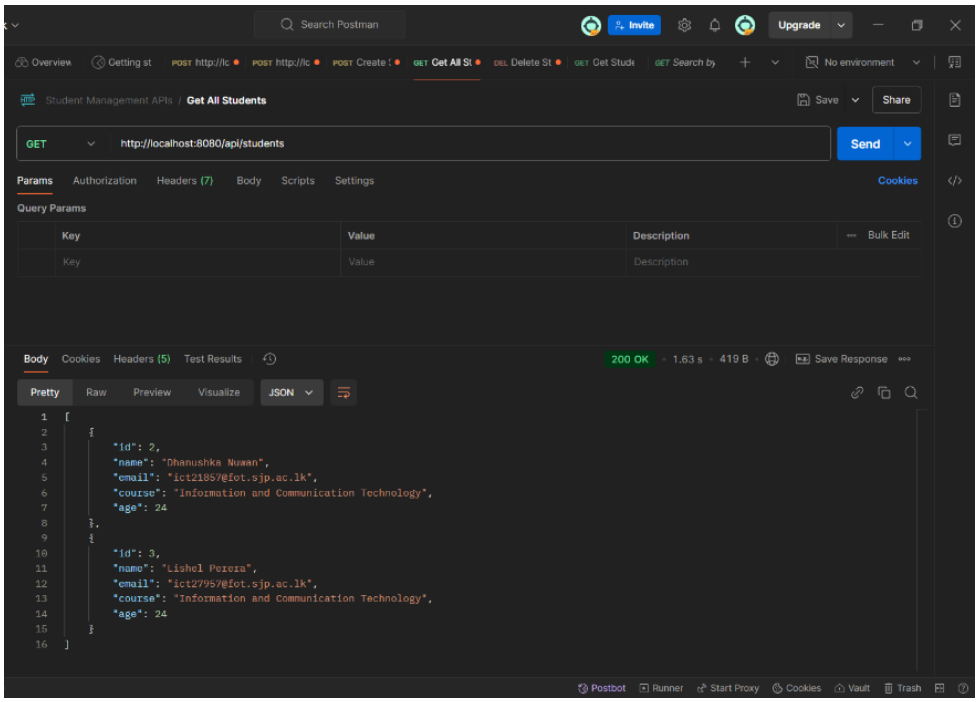
Link to Source Code - [dhanushnjay/Students-API]

Create a student

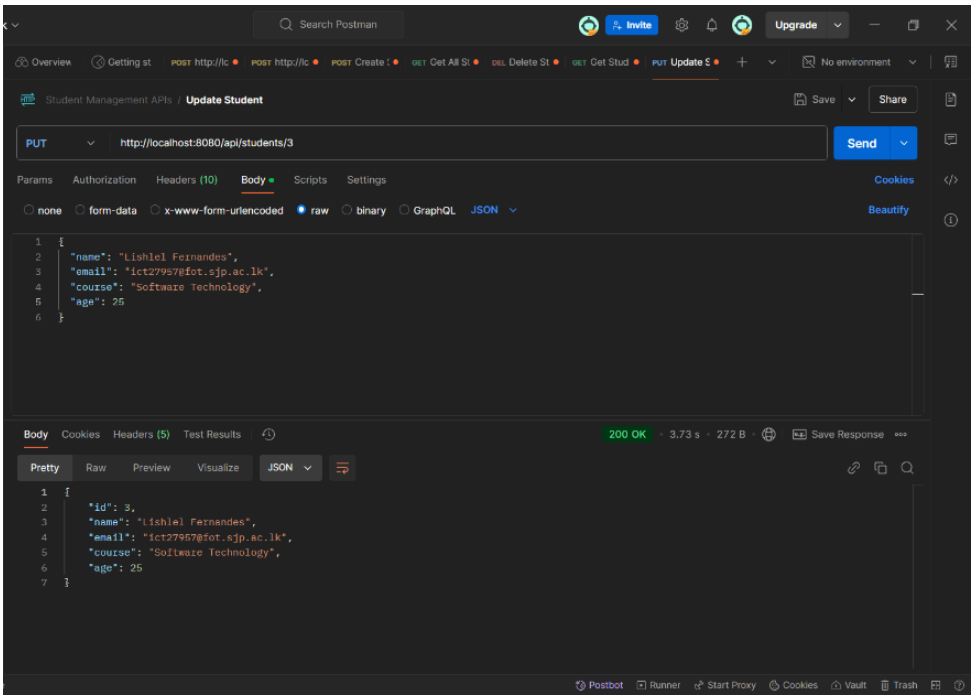




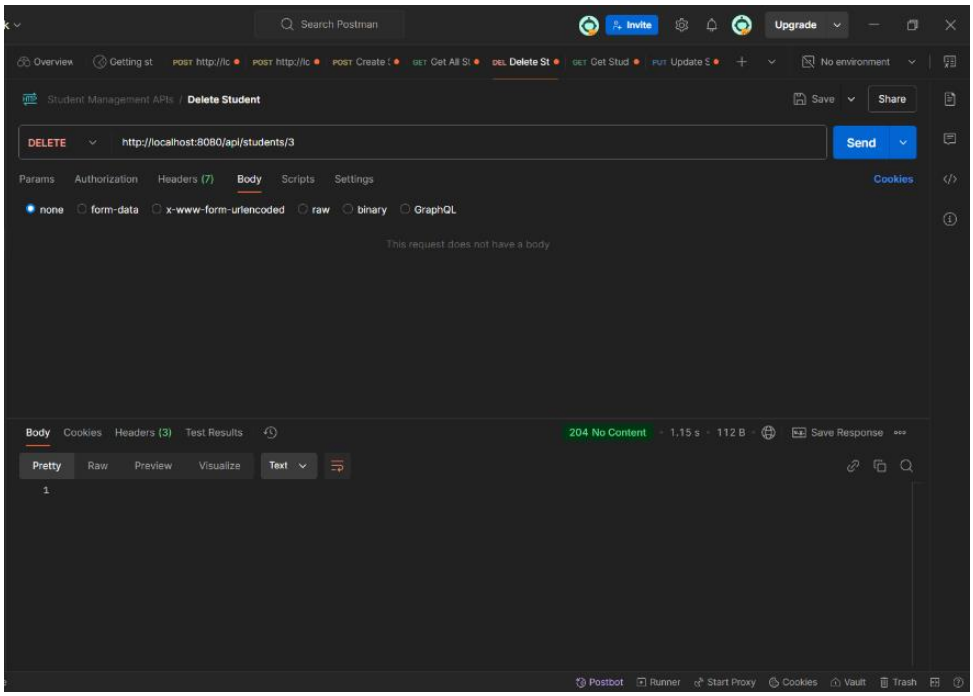
Get all students



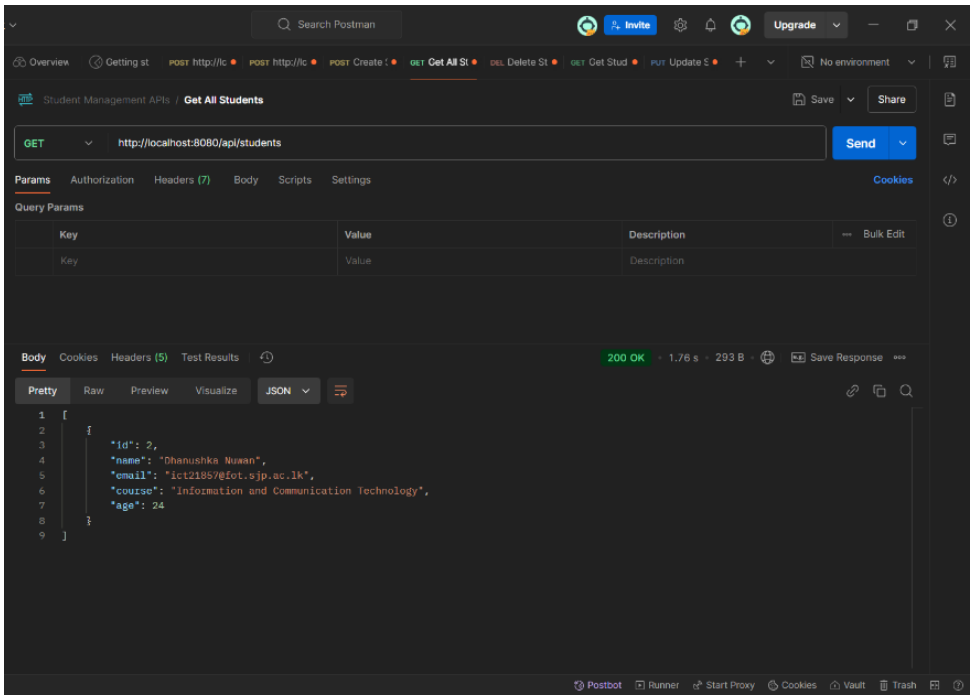
Update student



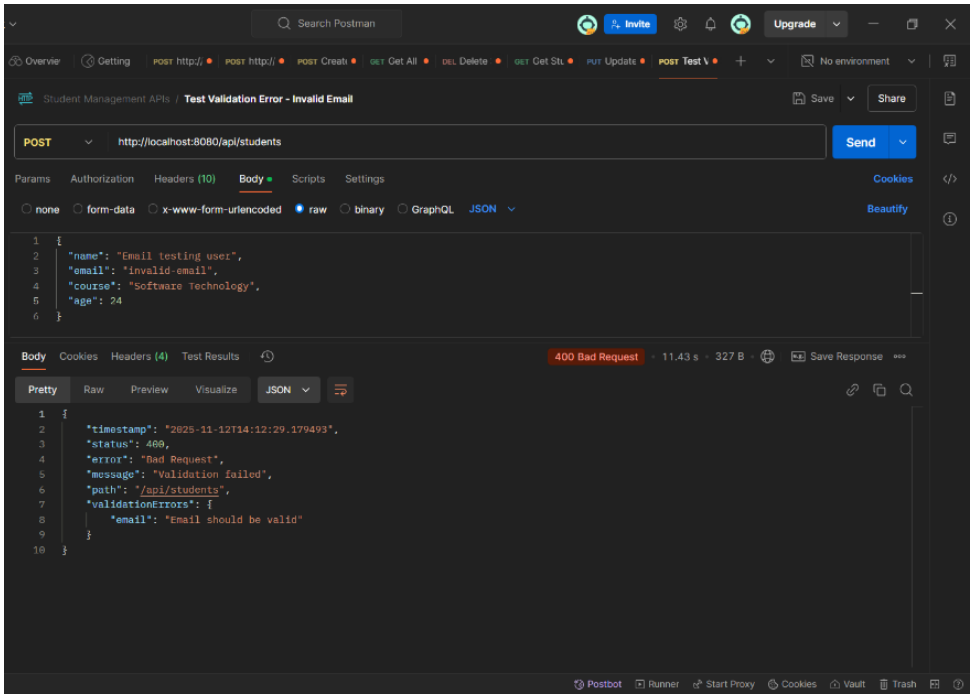
Delete student



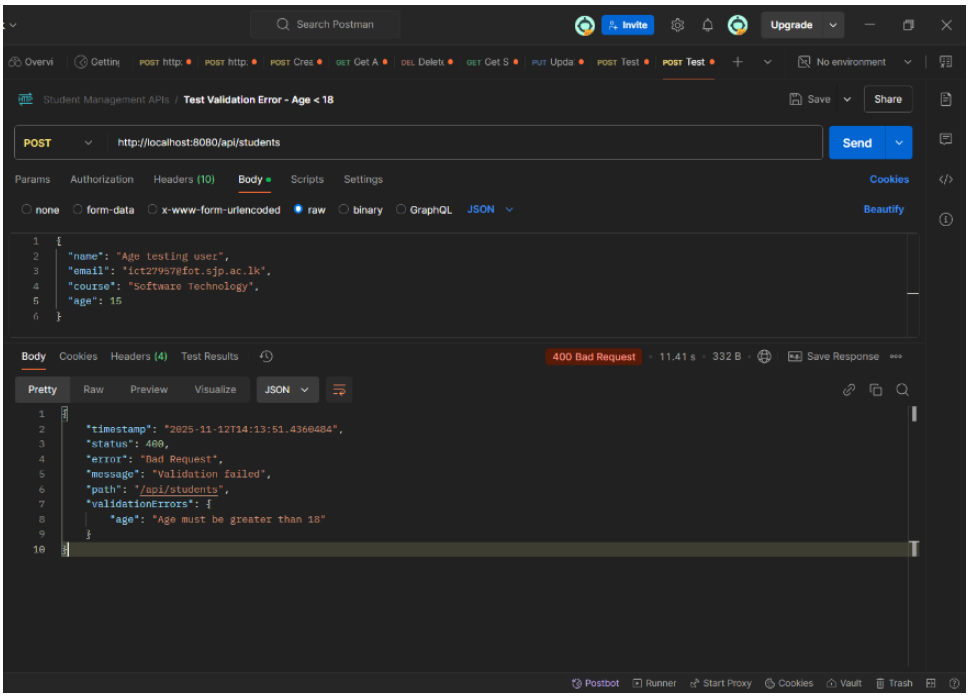
Get all students after deleting



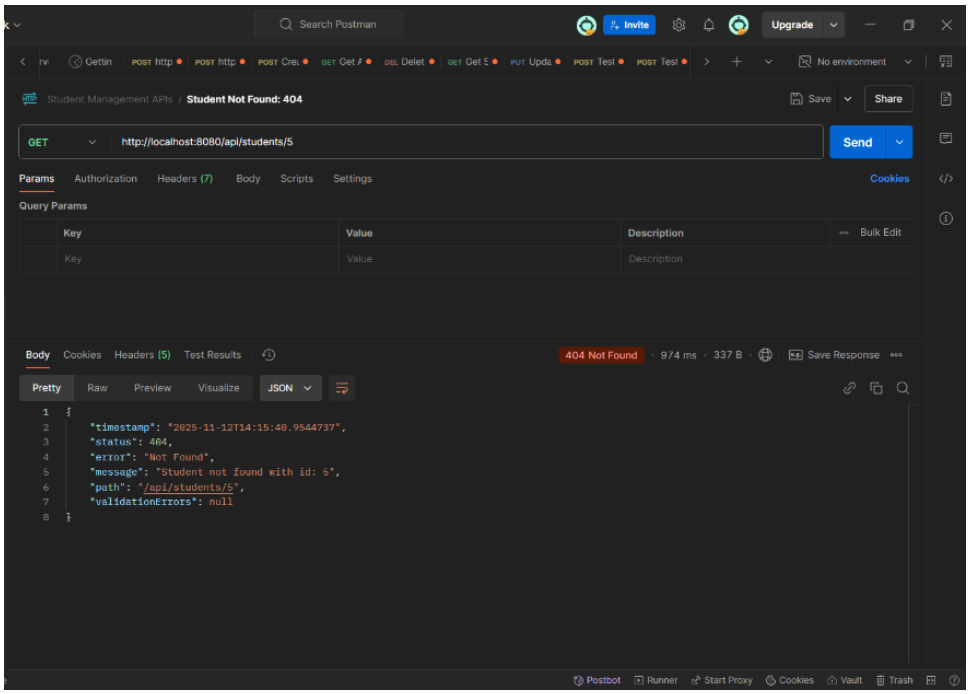
Email Validation Error



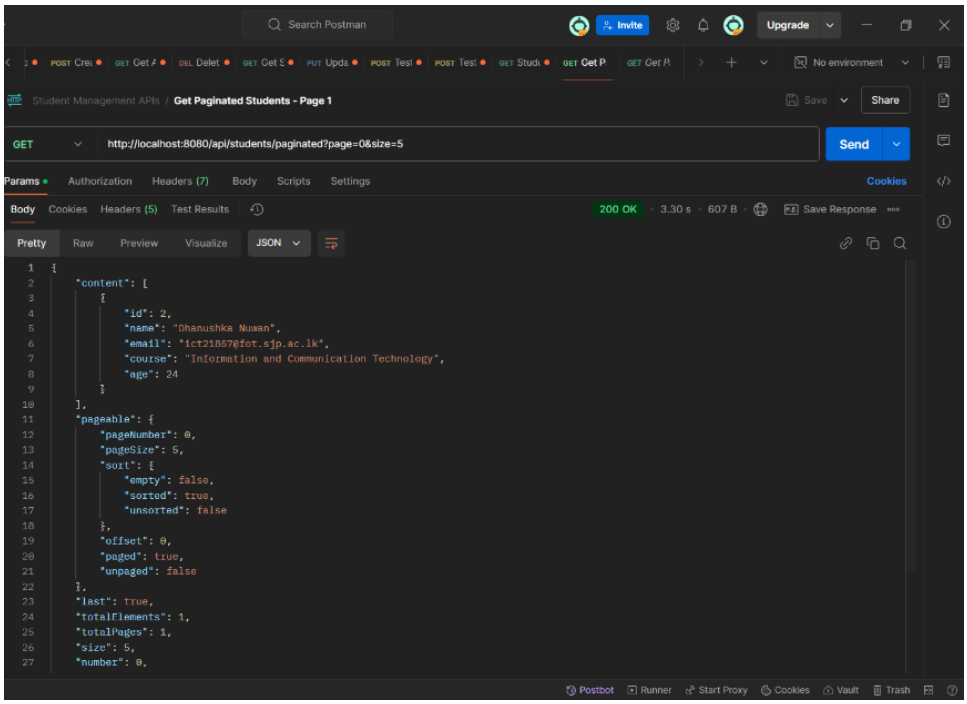
Age Validation Error



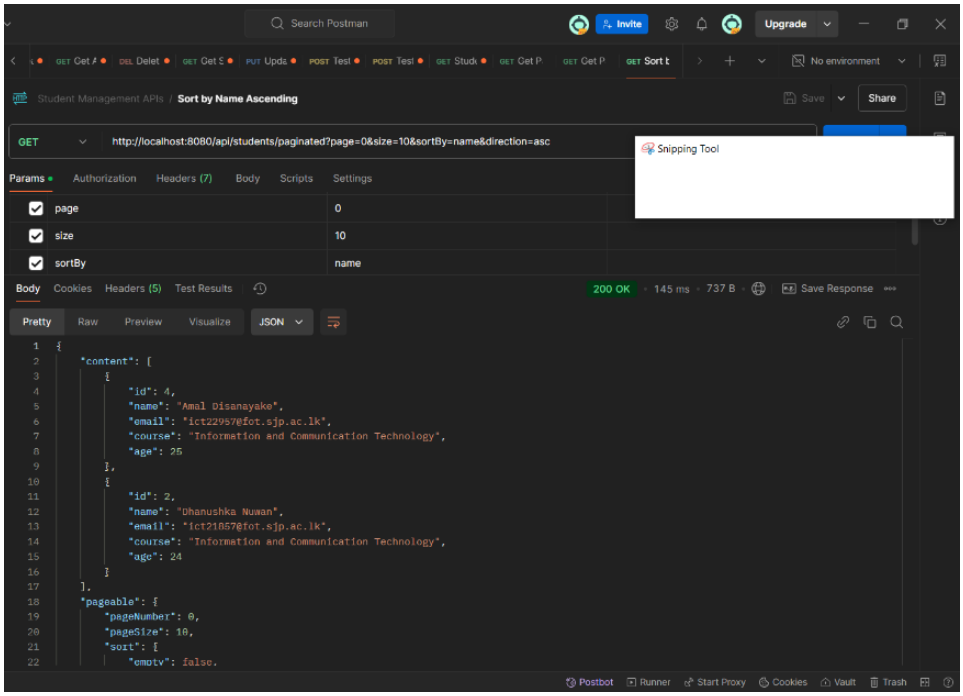
Student not Found: 404



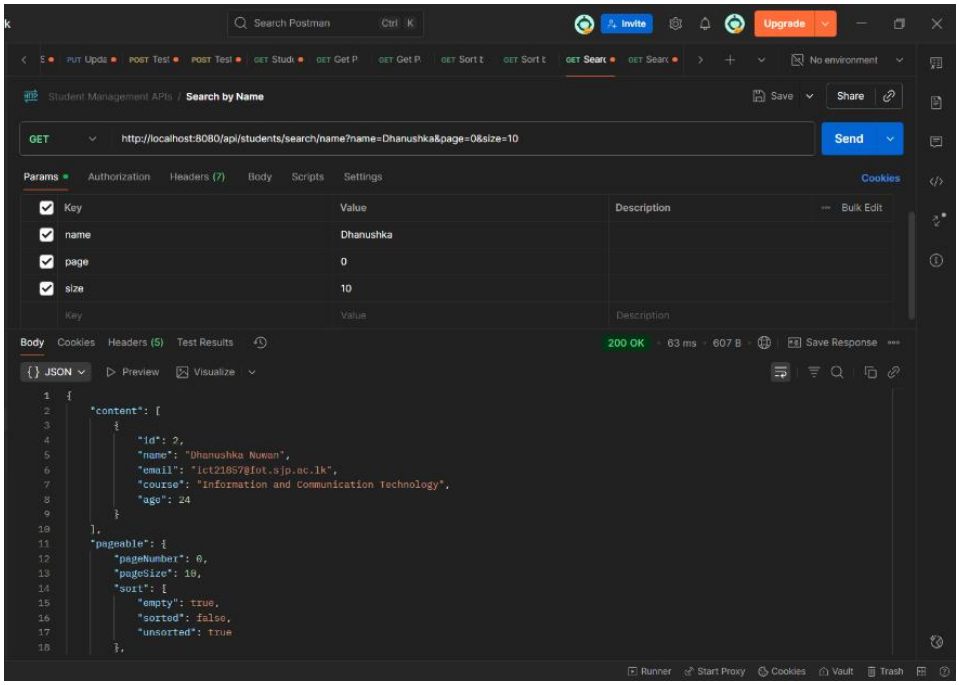
Get Paginated Students



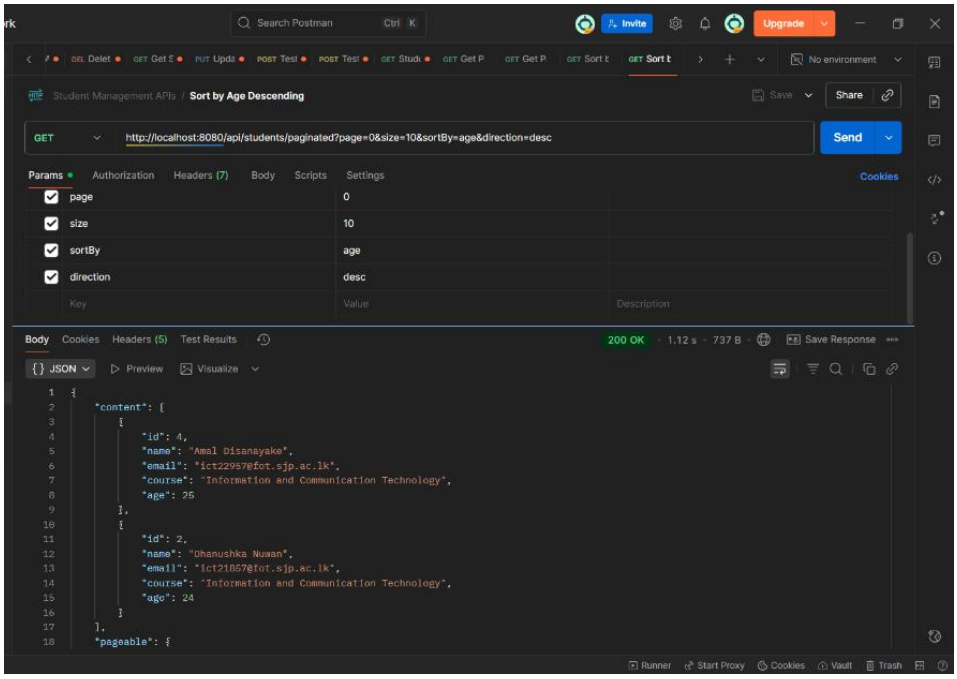
Sort Students by Name (Alphabetical Order)



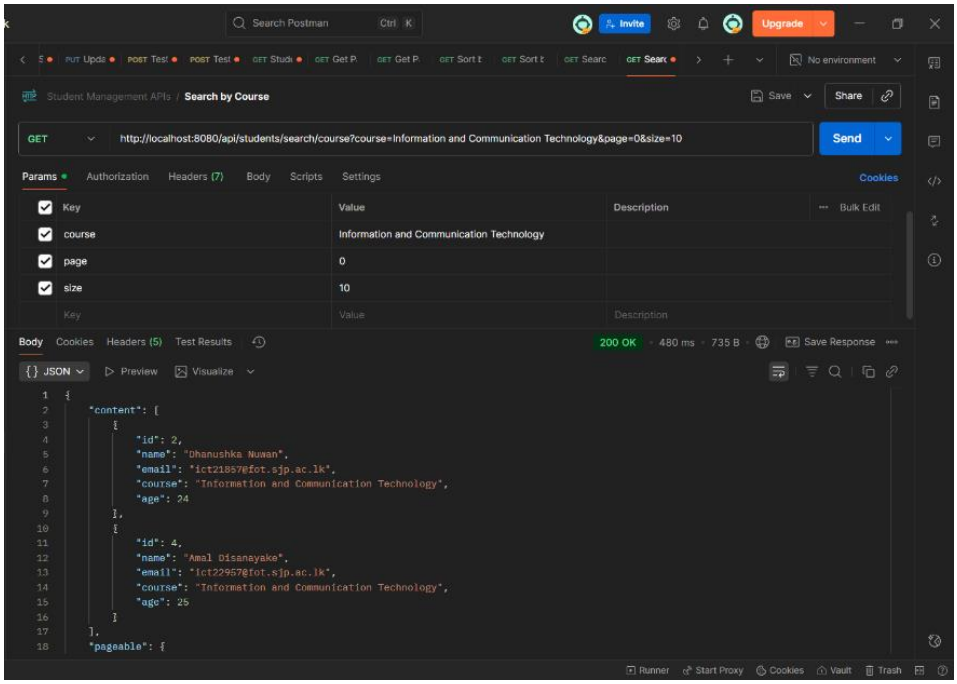
Sort Student by Name



Sort Students by Age (Descending Order)



Sort Students by Course



Sort Students by Keywords