RESTFul API Implementation and continuous deployment

TEST 1: Build REST API

Use Java to develop REST API for a class enrollment system for students for a university (You may use frameworks such as Springboot). You can persist data on a database like PostgreSQL/MySql/H2 (preferred) on a docker container or in data structure (in memory).

Requirements:
The users of the system will consist of both school administrators and students. School administrators will create student identities, and students will be able to enroll themselves into classes before each term.

Restraints

- School administrators can create and modify student records but never delete them.
- Each class a fixed credit/unit. Some harder classes might be 4 credits while easier ones could be 2 or 3 credits.
- Each student is only allowed to be enrolled in a maximum of 20 credits for each semester. There is a minimum of 10 credits to be considered full time.

Required APIs:

- API to add new students or modify them
- API to create a new semester
- API to enroll a student into a class for a particular semester
- API to get the list of classes for a particular student for a semester, or the fully history of classes enrolled.
- API to get the list of students enrolled in a class for a particular semester.
- API to drop a student from a class.

Non Functional Considerations
Performance and scalability aspects of your code will also be evaluated. Make sure the data structures that you use are chosen for scale and efficiency. For example, think about which APIs might be called more often than others with what parameters, and make sure those can handle the load efficiently.

Examples of a possible API design:

New student enrolment:

This will add a new record

POST request:

```
{
        “id”:223445,
        “firstName”: “Mike”,
        “lastName”: “Wong”,
        “class”:”3 A”,
```

```
        "nationality": "Singapore"
}
```

Update student record:

This will update an existing record on the basis of unique ID

```
PUT request
{
        "id":223445,
         "class":"3 C"
}
```

Delete student record

This will delete an existing record on the basis of unique ID

```
DELETE request
{
         "id": 223445
}
```

Fetch students record

1) Fetch bulk record: all students in database for that class
   GET Request, param in uri (http://yourdomain/fetchStudents? class= 3 A)

2) Fetch student record by student id
   GET Request, param in uri (http://yourdomain/fetchStudents? Id=223444)

You will be evaluated on

- Code style and reusability
- Unit test cases with code coverage
- Logging and code comments/ documentation
- Proper Readme to build, run and test the code