

Assignment: Implementing CRUD Operations with Java

//API's created are total 5 in numbers:

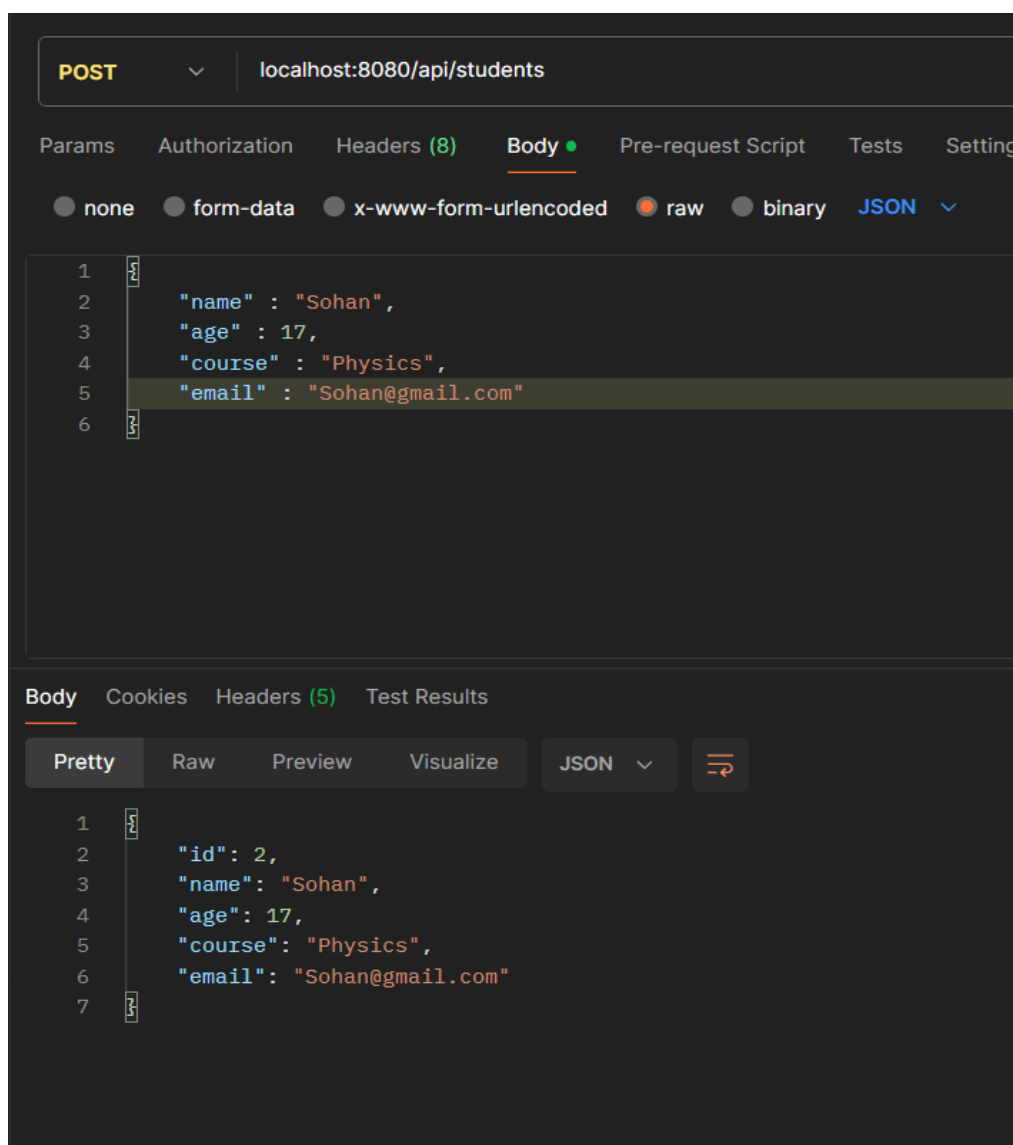
RequestMapping => "/api/students"

Endpoints:

- i. **POST** /api/students - Add a new student.

Example request body:

```
{
  "name": "Sohan",
  "age": 17,
  "course": "Physics",
  "email": "Sohan@gmail.com"
}
```



ii. **GET** /api/students - Retrieve all students.

The screenshot displays a REST client interface with a GET request to `localhost:8080/api/students`. The 'Body' tab is selected, showing a JSON object for a single student: `{ "name": "Sohan", "age": 17, "course": "Physics", "email": "Sohan@gmail.com" }`. Below this, the 'Test Results' section shows the actual response, which is a JSON array of two student objects: `[{ "id": 1, "name": "Rohan", "age": 17, "course": "Maths", "email": "Rohan123@gmail.com" }, { "id": 2, "name": "Sohan", "age": 17, "course": "Physics", "email": "Sohan@gmail.com" }]`.

```
GET localhost:8080/api/students
```

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary **JSON** ▾

```
1 {
2   "name" : "Sohan",
3   "age" : 17,
4   "course" : "Physics",
5   "email" : "Sohan@gmail.com"
6 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize **JSON** ▾

```
1 [
2   {
3     "id": 1,
4     "name": "Rohan",
5     "age": 17,
6     "course": "Maths",
7     "email": "Rohan123@gmail.com"
8   },
9   {
10    "id": 2,
11    "name": "Sohan",
12    "age": 17,
13    "course": "Physics",
14    "email": "Sohan@gmail.com"
15  }
16 ]
```

- iii. **GET** /api/students/{id} - Retrieve a specific student by ID.

The screenshot displays a REST client interface with a GET request to `localhost:8080/api/students/1`. The 'Body' tab is selected, showing a JSON object for a student with ID 1. The response body is also shown in the 'Body' tab, displaying a JSON object for a student with ID 1, name 'Rohan', age 17, course 'Maths', and email 'Rohan123@gmail.com'.

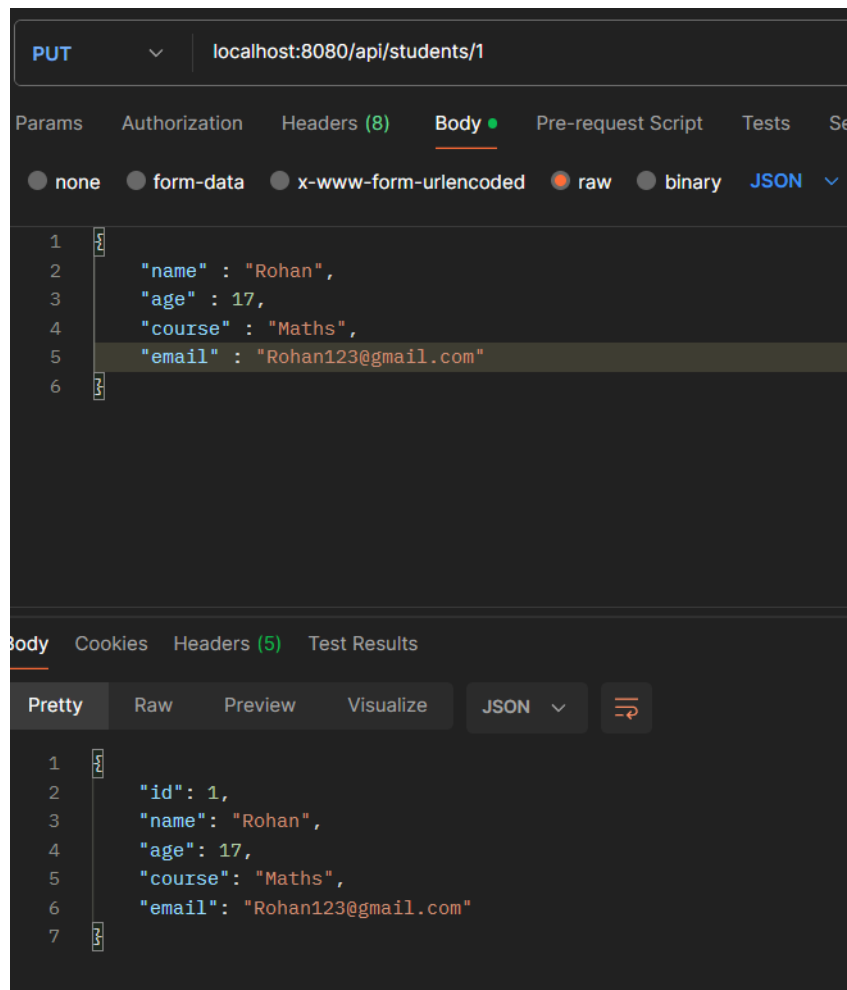
Request:

```
1 {
2   "name" : "Sohan",
3   "age" : 17,
4   "course" : "Physics",
5   "email" : "Sohan@gmail.com"
6 }
```

Response:

```
1 {
2   "id": 1,
3   "name": "Rohan",
4   "age": 17,
5   "course": "Maths",
6   "email": "Rohan123@gmail.com"
7 }
```

- iv. **PUT** /api/students/{id} - Update an existing student's details.



- v. **DELETE** /api/students/{id} - Delete a student by ID.

