

ANUJA PANDEY

1) Create new collection and map the API's in the same folder

The screenshot shows the Postman interface for a GET request. The request is named "One" and is directed to the URL `http://localhost:8080/student/list?`. The response status is 200 OK, with a time of 37 ms and a size of 18.43 KB. The response body is displayed in JSON format, showing a list of two students:

```
[{"id": 1, "firstName": "Vernon", "lastName": "Harper", "email": "egestas.rhuncus.Proin@massaquisqueporttitor.org", "programme": "Financial Analysis", "courses": ["Accounting", "Statistics"]}, {"id": 2, "firstName": "Murphy", "lastName": "Holmes", "email": "faucibus.orci.luctus@duisac.net", "programme": "Financial Analysis", "courses": ["Accounting", "Statistics"]}]
```

2) Post any student Data and get the created data by Student ID

The screenshot shows the Postman interface for a POST request. The request is named "Two" and is directed to the URL `http://localhost:8080/student`. The request body is in JSON format, representing a new student:

```
{ "id": 101, "firstName": "Anuja", "lastName": "Pandey", "email": "anuja.pandey@tothenew.com", "programme": "Computer Engineer", "courses": ["Computer Science", "Mathematics"] }
```

The response status is 201 Created, with a time of 43 ms and a size of 203 B. The response body is displayed in JSON format, showing a confirmation message:

```
{ "msg": "Student added" }
```

3) Fetch the data only by programme "Financial Analysis"

The screenshot shows the Postman interface with a GET request to `http://localhost:8080/student/list?programme=Financial%20Analysis`. The request is successful, returning a 200 OK status. The response body is displayed in JSON format, showing a list of two students. The first student has the programme "Financial Analysis" and the second student also has the programme "Financial Analysis".

```
1 [
2   {
3     "id": 1,
4     "firstName": "Vernon",
5     "lastName": "Harper",
6     "email": "egestas.rhonus.Proin@massaQuisqueporttitor.org",
7     "programme": "Financial Analysis",
8     "courses": [
9       "Accounting",
10      "Statistics"
11    ]
12  },
13  {
14    "id": 2,
15    "firstName": "Murphy",
16    "lastName": "Holmes",
17    "email": "faucibus.orci.luctus@Outsac.net",
18    "programme": "Financial Analysis",
19    "courses": [
20      "Accounting"
21    ]
22  }
23 ]
```

4) Post student data with existing email Id and check for the error message. Mention the test script.

The screenshot shows the Postman interface with a POST request to `http://localhost:8080/student`. The request body is a JSON object representing a new student with the email "anuja.pandey@tothenew.com". The request fails, returning a 500 Internal Server Error status. The response body is displayed in JSON format, showing an error message: "Email address already exists".

```
1 {
2   "id": 102,
3   "firstName": "Anuj",
4   "lastName": "Pandey",
5   "email": "anuja.pandey@tothenew.com",
6   "programme": "Computer Engineer",
7   "courses": [
8     "Computer Science",
9     "Mathematics"
10  ]
11 }
```

```
1 {
2   "error": "Email address already exists"
3 }
```

5) Post student data with an incorrect format of email Id and check for the error message. Mention the test script.

The screenshot shows a Postman interface for a POST request to `http://localhost:8080/student`. The request body is a JSON object representing a student with an invalid email address. The response is a 500 Internal Server Error with a message: "Please correct the following errors", "fieldErrors": {"email": "not a well-formed email address"}.

```
1 {
2   "id": 102,
3   "firstName": "Anuj",
4   "lastName": "pandey",
5   "email": "anuja.pandeytothenew.com",
6   "programme": "Computer Engineer",
7   "courses": [
8     "Computer Science",
9     "Mathematics"
10  ]
11 }
```

Status: 500 Internal Server Error Time: 17 ms Size: 273 B

```
1 {
2   "error": "Please correct the following errors",
3   "fieldErrors": {
4     "email": "not a well-formed email address"
5   }
6 }
```

6) Fetch the data only by the programme "Computer Science" and "Criminology". Mention the test script

The screenshot shows a Postman interface for a GET request to `http://localhost:8080/student/list?programme=Computer%20Science`. The request has a query parameter `programme=Computer%20Science`. The response is a 200 OK status with a JSON array of two student objects.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> programme	Computer%20Science	
Key	Value	Description

Status: 200 OK Time: 19 ms Size: 2.73 KB

```
1 [
2   {
3     "id": 3,
4     "firstName": "Reece",
5     "lastName": "Jason",
6     "email": "tincidunt.dui@ultricesit.co.uk",
7     "programme": "Computer Science",
8     "courses": [
9       "Calculus",
10      "Algorithms",
11      "Software Development",
12      "Ethics"
13    ]
14  },
15  {
16    "id": 4,
17    "firstName": "Orson",
18    "lastName": "Armando",
19    "email": "nascetur@conguea.com",
20    "programme": "Criminology"
21  }
22 ]
```

GET One POST Two GET Three POST Four POST Five GET Six GET SixOr X + ... No Environment

SixOne Examples (0)

GET http://localhost:8080/student/list?programme=Criminology Send Save

Params Authorization Headers Body Pre-request Script Tests Cookies Code Comments (0)

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> programme	Criminology			
Key	Value	Description		

Body Cookies Headers (3) Test Results (1/1) Status: 200 OK Time: 15 ms Size: 3.04 KB Save Download

Pretty Raw Preview JSON

```
1- [
2- {
3-   "id": 12,
4-   "firstName": "Chadwick",
5-   "lastName": "Lawrence",
6-   "email": "turpts.non-enim@sociisnatoque.com",
7-   "programme": "Criminology",
8-   "courses": [
9-     "Criminal Mind",
10-    "Gender, Crime and Justice ",
11-    "Philosophy of Law "
12-   ],
13- },
14- {
15-   "id": 13,
16-   "firstName": "Zachary",
17-   "lastName": "Vance",
18-   "email": "In.tincidunt.congue@Quisque.com",
19-   "programme": "Criminology",
20- }
```

Learn Build Browse

7) Run all the above cases and generate the report.