

# TEST CASES :

## 1. POST

The screenshot shows the Postman interface with a collection named 'test'. A POST request is made to `http://localhost:8080/users`. The request body is a JSON object:

```
1   "username": "phani",
2   "address": "palakol",
3   "address2": "apstate",
4   "emailId": "hani123@gmail.com",
5   "firstname": "hani",
6   "lastname": "lucky",
7   "phonenumber": 655333658
```

The response status is 200 OK, with a duration of 44 ms and a size of 313 B. The response body is identical to the request body.

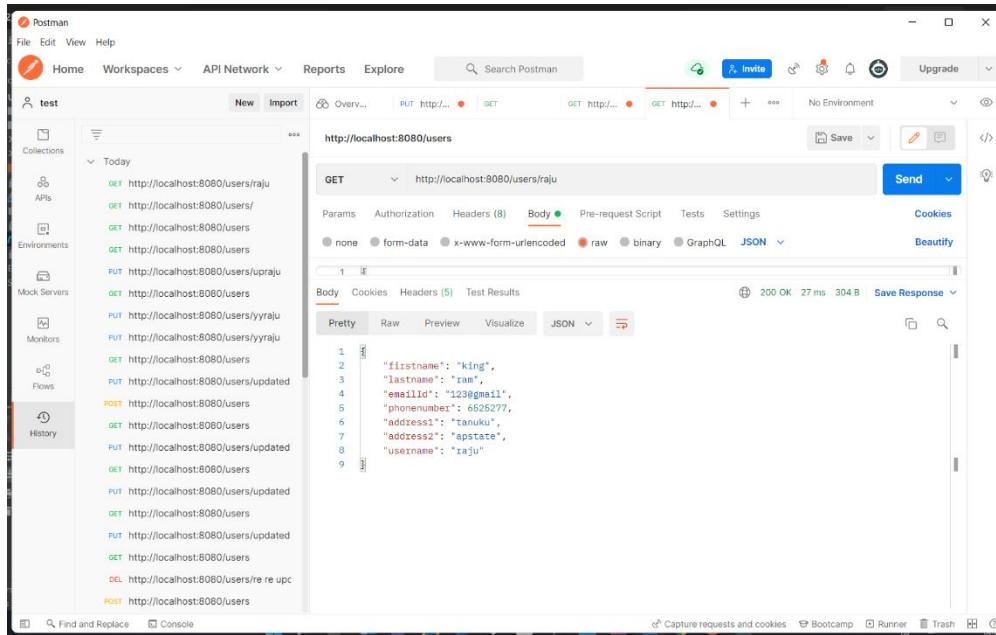
Checking above user added or not by using get method

GET :

The screenshot shows the Postman interface with a collection named 'test'. A GET request is made to `http://localhost:8080/users`. The response status is 200 OK, with a duration of 11 ms and a size of 499 B. The response body is a JSON array containing two user objects:

```
1 [
2   {
3     "firstname": "hani",
4     "lastname": "lucky",
5     "emailId": "hani123@gmail.com",
6     "phonenumber": 655333658,
7     "address1": "palakol",
8     "address2": "apstate",
9     "username": "phani"
10   },
11   {
12     "firstname": "raju",
13     "lastname": "king",
14     "emailId": "updated raju123@gmail.com",
15     "phonenumber": 644533668,
16     "address1": "updated vijayawada",
17     "address2": "updated tsstate",
18     "username": "updated raju"
19   }
20 ]
```

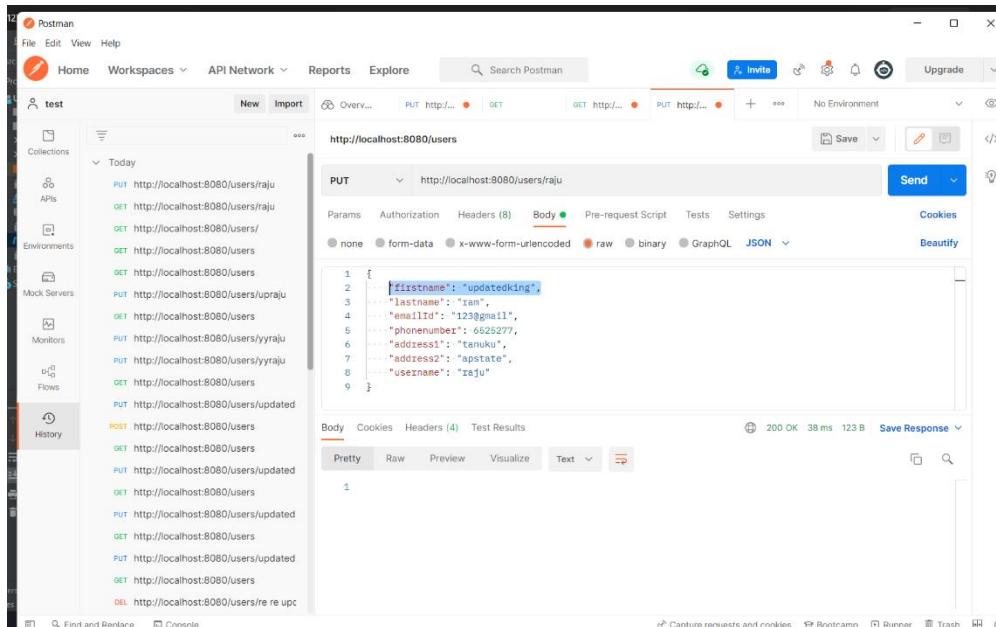
## GET USERDETAILS BY USING USERID:(EXAMPLE : USERID = raju)



The screenshot shows the Postman application interface. A GET request is made to `http://localhost:8080/users/raju`. The response is a 200 OK status with a JSON body containing the following user details:

```
1 {
2   "firstname": "king",
3   "lastname": "ram",
4   "emailId": "123@gmail",
5   "phonenumber": 6525277,
6   "address1": "tanuku",
7   "address2": "apstate",
8   "username": "raju"
9 }
```

## PUT:updating above example raju details

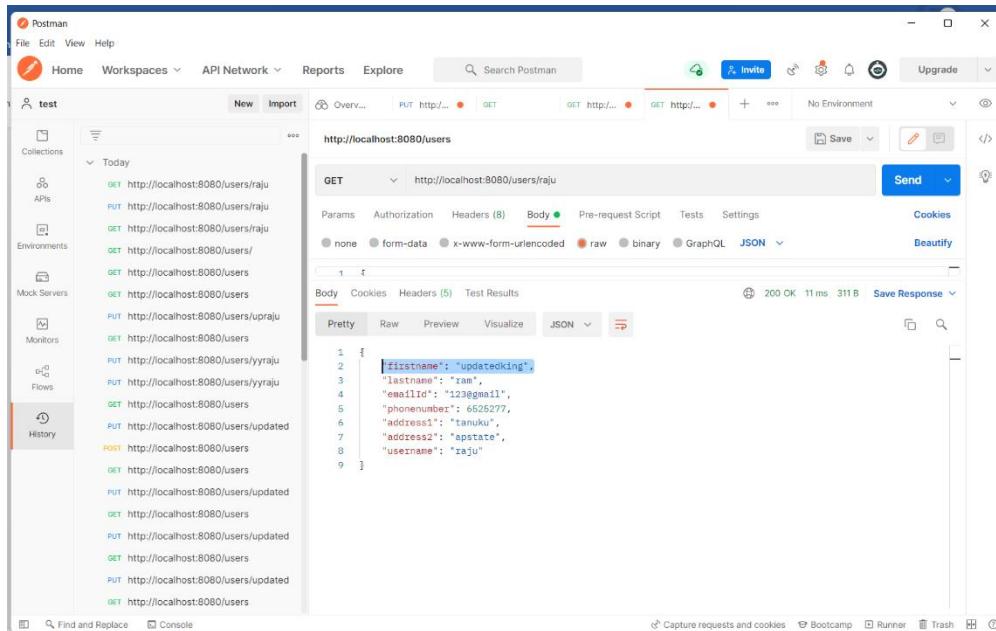


The screenshot shows the Postman application interface. A PUT request is made to `http://localhost:8080/users/raju` with the following JSON body:

```
1 {
2   "firstname": "updatedking",
3   "lastname": "ram",
4   "emailId": "123@gmail",
5   "phonenumber": 6525277,
6   "address1": "tanuku",
7   "address2": "apstate",
8   "username": "raju"
9 }
```

The response is a 200 OK status.

## Verifying updated or not by using get method

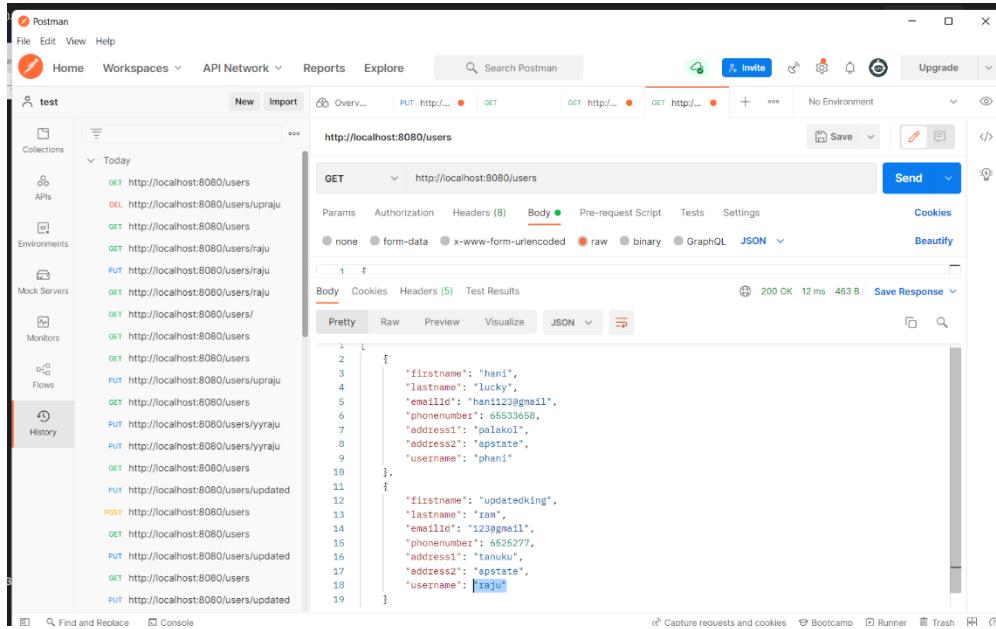


The screenshot shows the Postman interface with a collection named 'test'. A GET request is made to `http://localhost:8080/users/raju`. The response body is displayed in JSON format:

```
1 {
2   "firstname": "updatedKing",
3   "lastname": "ram",
4   "emailId": "123@gmail",
5   "phonenumber": 6526277,
6   "address1": "tanuku",
7   "address2": "apstate",
8   "username": "raju"
9 }
```

**DELETE :**

**Before deleting**

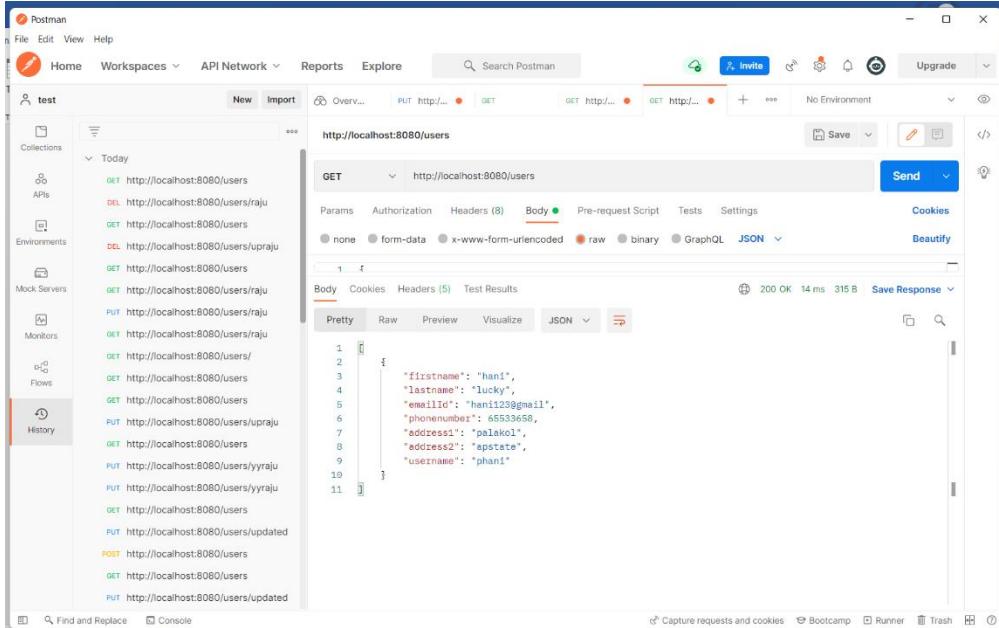


The screenshot shows the Postman interface with a collection named 'test'. A GET request is made to `http://localhost:8080/users`. The response body is displayed in JSON format, showing two user entries:

```
1 [
2   {
3     "firstname": "hani",
4     "lastname": "lucky",
5     "emailId": "hani123@gmail",
6     "phonenumber": 6533656,
7     "address1": "palakol",
8     "address2": "apstate",
9     "username": "pani"
10   },
11   {
12     "firstname": "updatedKing",
13     "lastname": "ram",
14     "emailId": "123@gmail",
15     "phonenumber": 6526277,
16     "address1": "tanuku",
17     "address2": "apstate",
18     "username": "raju"
19   }
]
```

Now deleting user from above users by using userid(example  
userid = raju)

After deleting verify by using get method



The screenshot shows the Postman application interface. On the left, the sidebar displays a collection named 'test' containing various API requests. In the main workspace, a GET request is selected with the URL `http://localhost:8080/users`. The response body is displayed in JSON format:

```
1  {
2   "firstname": "hani",
3   "lastname": "lucky",
4   "emailid": "hani123@gmail",
5   "phonenumber": 65533658,
6   "address1": "pelakol",
7   "address2": "apstate",
8   "username": "phan1"
9 }
10
11 }
```