

## Experiment No. 7

**Aim:** Validating RESTful APIs using Postman.

**Code:**

GET Request:

Select GET Method:

<https://reqres.in/api/users>

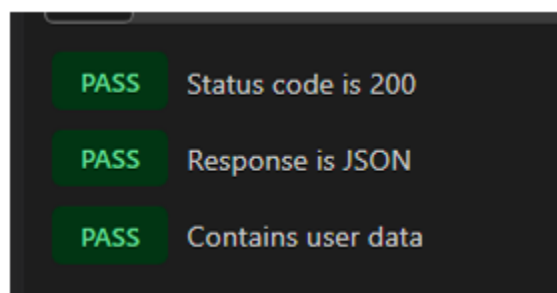
Testing Script

```
Headers (7)  Body  Scripts  Settings
1  pm.test("Status code is 200", function() {
2    pm.response.to.have.status(200);
3  });
4
5  pm.test("Response is JSON", function() {
6    pm.response.to.be.json;
7  });
8
9  pm.test("Contains user data", function() {
10    var jsonData = pm.response.json();
11    pm.expect(jsonData.data.length).to.be.above(0);
12  });
13
```

Add api\_key in Authorization:

Key	api_key
Value	reqres-free-v1
Add To	Query Params

Test result

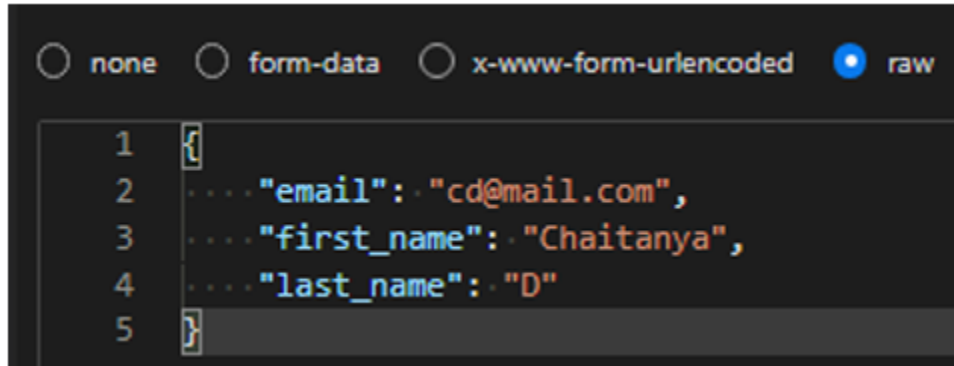


POST Request:

Select POST Method:

<https://reqres.in/api/users>

Body (JSON): Go to Body → raw → JSON → add this:

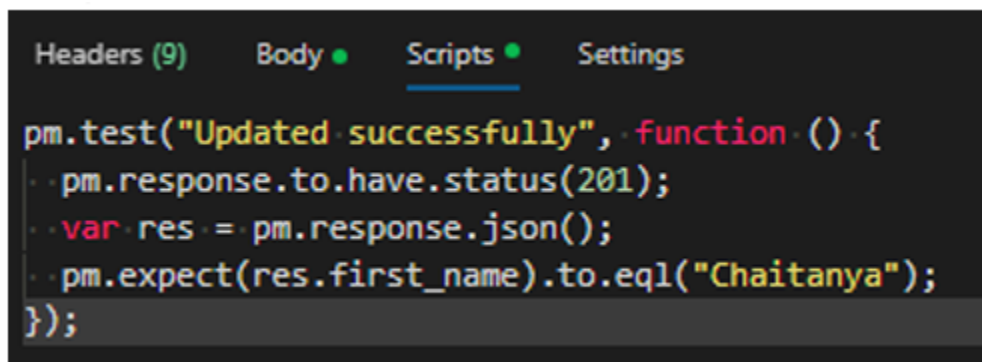


```
1 {  
2   "email": "cd@mail.com",  
3   "first_name": "Chaitanya",  
4   "last_name": "D"  
5 }
```

Add api\_key in Authorization:

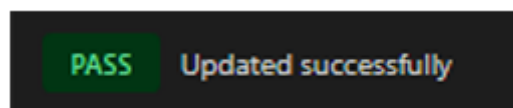
Key	<input type="text" value="api_key"/>
Value	<input type="text" value="reqres-free-v1"/>
Add To	<input type="text" value="Query Params"/>

Testing Scripts: Validate POST Response



```
pm.test("Updated successfully", function () {  
  pm.response.to.have.status(201);  
  var res = pm.response.json();  
  pm.expect(res.first_name).to.eql("Chaitanya");  
});
```

Test Result:



PUT Request:

Select PUT Method

<https://reqres.in/api/users/12>

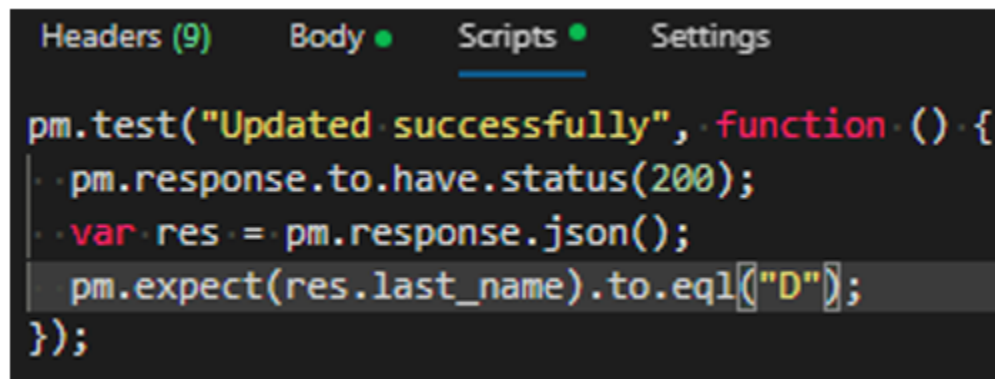
Body(JSON): Body → raw → JSON



A screenshot of a REST client interface showing the 'raw' body tab selected. The body contains a JSON object with three fields: 'email', 'first\_name', and 'last\_name'. The JSON is formatted with line numbers 1 through 5 on the left.

```
1 {  
2   "email": "new@mail.com",  
3   "first_name": "Jeo",  
4   "last_name": "D"  
5 }
```

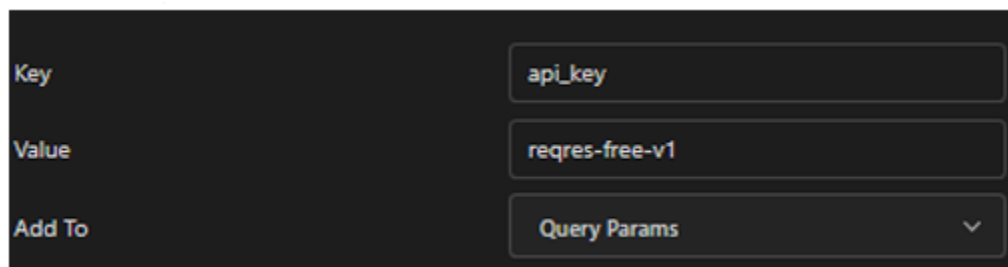
Testing script:



A screenshot of a REST client interface showing the 'Scripts' tab selected. The script is a PM (Postman) test script that checks the response status and the 'last\_name' field of the JSON response.

```
pm.test("Updated successfully", function () {  
  pm.response.to.have.status(200);  
  var res = pm.response.json();  
  pm.expect(res.last_name).to.eql("D");  
});
```

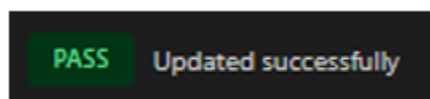
Add api\_key in Authorization:



A screenshot of the 'Authorization' tab in a REST client. It shows three fields: 'Key' with the value 'api\_key', 'Value' with the value 'reqres-free-v1', and 'Add To' with a dropdown menu set to 'Query Params'.

Key	api_key
Value	reqres-free-v1
Add To	Query Params

Test result:



A screenshot of the test result showing a green 'PASS' status and the text 'Updated successfully'.

**PASS** Updated successfully

DELETE Request:

Select DELETE Method:

<https://reqres.in/api/users/1>

Add api\_key in Authorization:

Key	api_key
Value	reqres-free-v1
Add To	Query Params

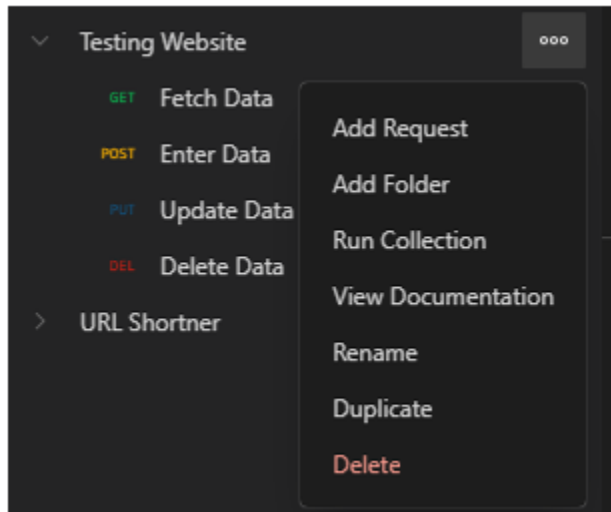
Testing script:

```
Headers (7)  Body  Scripts  Settings
pm.test("Deleted successfully", function () {
  pm.response.to.have.status(204);
});
```

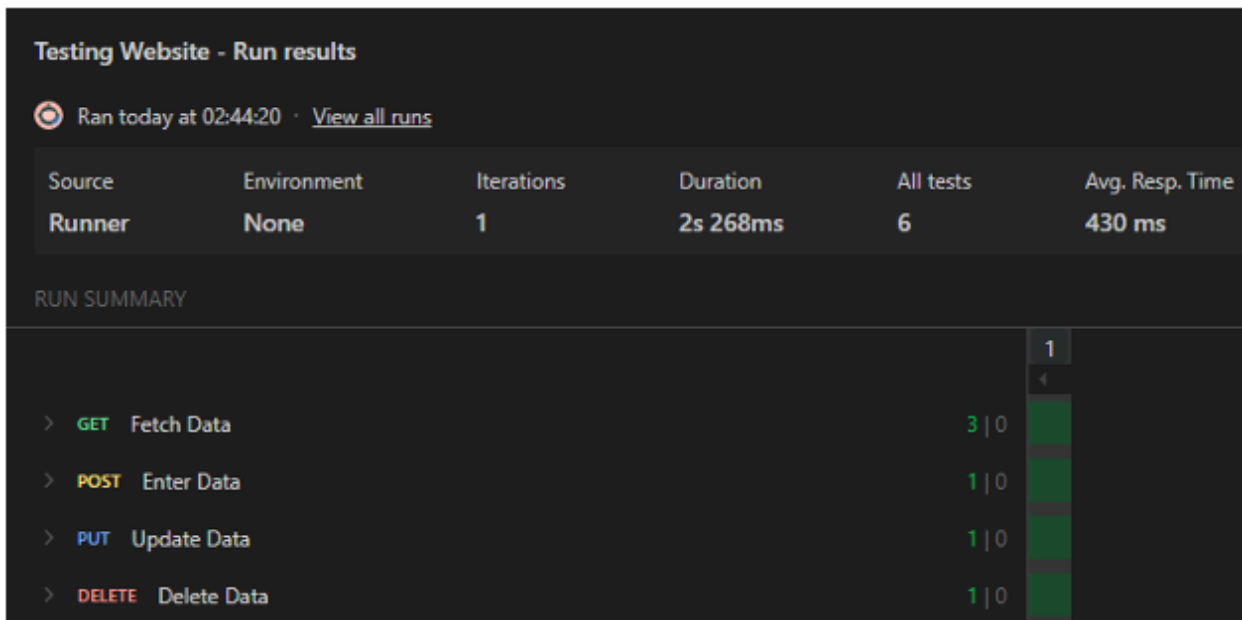
Test Result:

**PASS** Deleted successfully

## Output:



## Run All Tests at Once



## Conclusion:

Postman helps validate RESTful APIs by testing endpoints for correct responses, status codes, and data.

Using test scripts, we can automate checks like email format or record count, ensuring the API works as expected.