University of Central Punjab
Faculty of Information Technology

# Automated Software Testing

# Fall 2024

## Assignment 4: API Testing with Postman

| CLO # | Course Learning Outcome Statement | Taxonomy Level |
|---|---|---|
| CLO 3 | **Analyze** API functionality, security, and performance to identify testing needs and develop effective API testing strategies using industry-standard tools and techniques | C4 |

**Deadline: 30-January 2025, on portal.**

- **Instructions**
  - Use the title page from this document.
  - Organize each task with headings.
  - Submit a single **Word document** containing:
    - Details for each task (screenshot, HTTP Method, Endpoint, Header, Body, Pre-response Script, Post-response Script).
  - Each group (max 4 members) or individual should make one combined submission, with all member names listed in the table below.
  - ***LATE PENALTY 25% PER DAY. Plagiarism has zero tolerance***

| Name: | Reg. #: |
|---|---|
| MUHAMMAD MAZHAR REHAN | L1F21BSSE0183 |
| RANA MUHAMMAD ZAIN BIN IMRAN | L1F21BSSE0539 |
| MUHAMMAD UMAN | L1F21BSSE0611 |
| USAMA BIN NASEER | L1F21BSSE0540 |

## Objective:

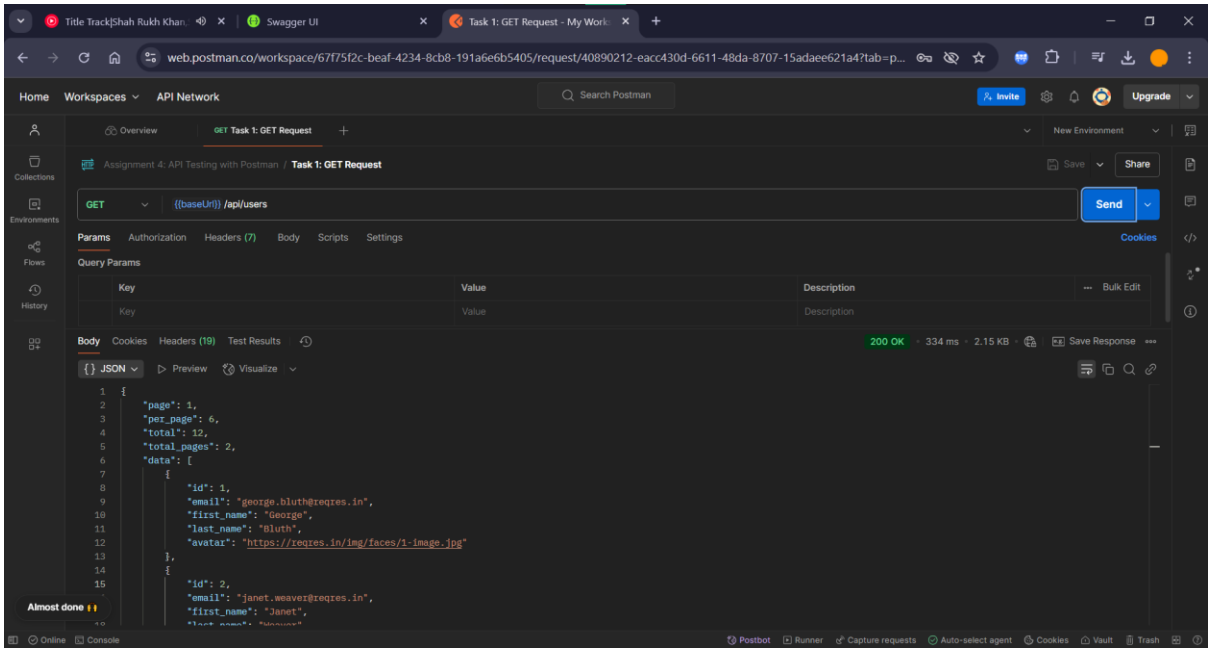
The objective of this assignment is to help students understand and practice the basics of API testing using Postman. The tasks are designed to test their ability to send requests, understand responses, and validate APIs.

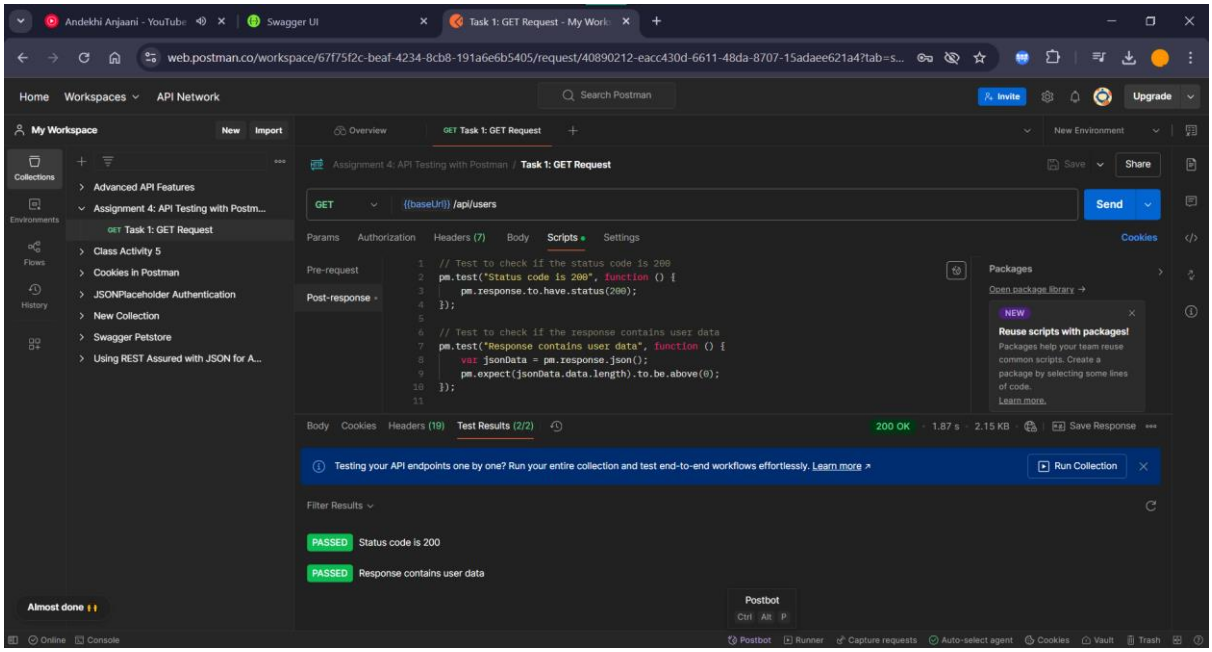
## Task Details:

API Requests
Use the following public API: https://reqres.in.
Complete the following tasks and provide the requested details for each API request:

### Task 1: GET Request

- Endpoint: `/api/users`
- Take a screenshot of the execution.
- Share the following details in the Word document:
    1. Screenshot of the execution






2. HTTP Method

   GET

3. Endpoint along with base URL.

   **Base URL**: https://reqres.in
   **Endpoint**: `/api/users`
   The complete URL for the request is:
   https://reqres.in/api/users

4. Header

   For this **GET** request, **no headers** are required.

5. Body (if applicable, leave it blank if no body was sent)

   Since **GET** requests typically do not include a body, this section can be left **blank**.

6. Pre-response Script (add which is applicable)

   For this simple GET request, we did not need any **pre-response script**. However, if necessary, I could set an environment variable like so:

   ```
   pm.environment.set("lastRequest", "GET /api/users");
   ```

7. Post-response Script (add which is applicable)

```
// Test to check if the status code is 200
pm.test("Status code is 200", function () {
```

```
    pm.response.to.have.status(200);
});

// Test to check if the response contains user data
pm.test("Response contains user data", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.data.length).to.be.above(0);
});
```

These tests will confirm that the API responds with a status code of 200 and that the response includes user data.

**Task 2: POST Request**

- Endpoint: `/api/users`
- Include the JSON body in the request.
- Take a screenshot of the execution.
- Share the following details in the Word document:
    1. Screenshot of the execution
    2. HTTP Method

       POST

    3. Endpoint along with base URL.

       https://reqres.in/api/users

    4. Header

       Content-Type: application/json

    5. Body (include the JSON body sent)

```
{
  "name": "Mazhar Rehan",
  "job": "Software Engineer"
}
```

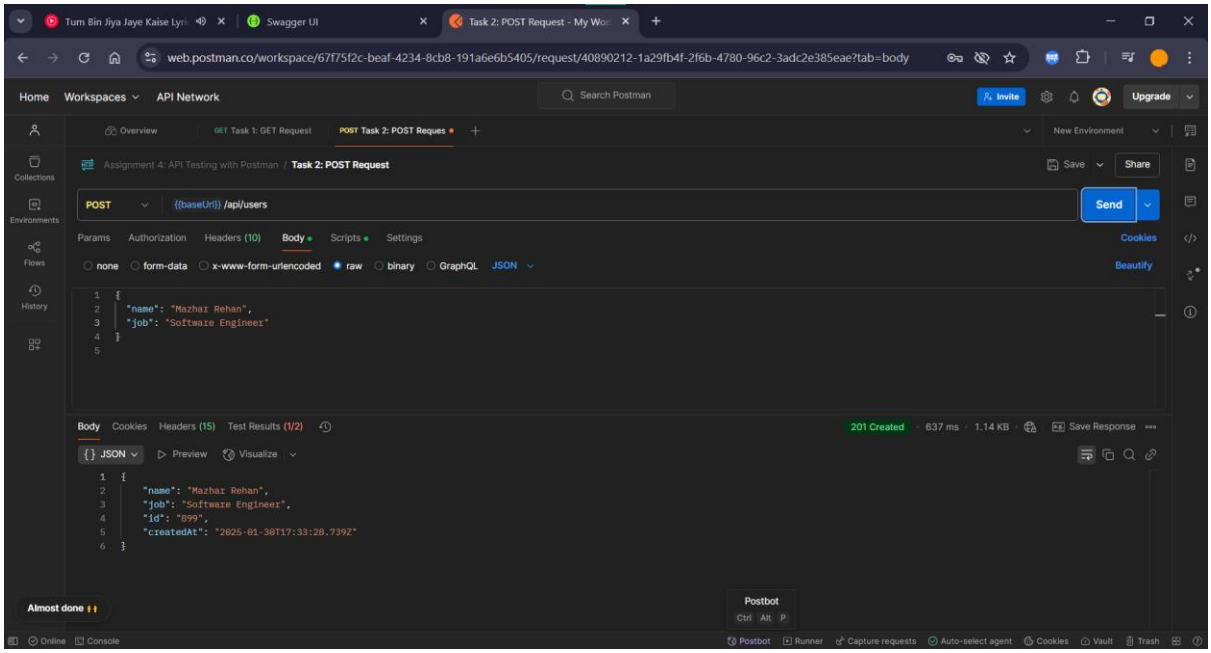
    6. Pre-response Script (add which is applicable)

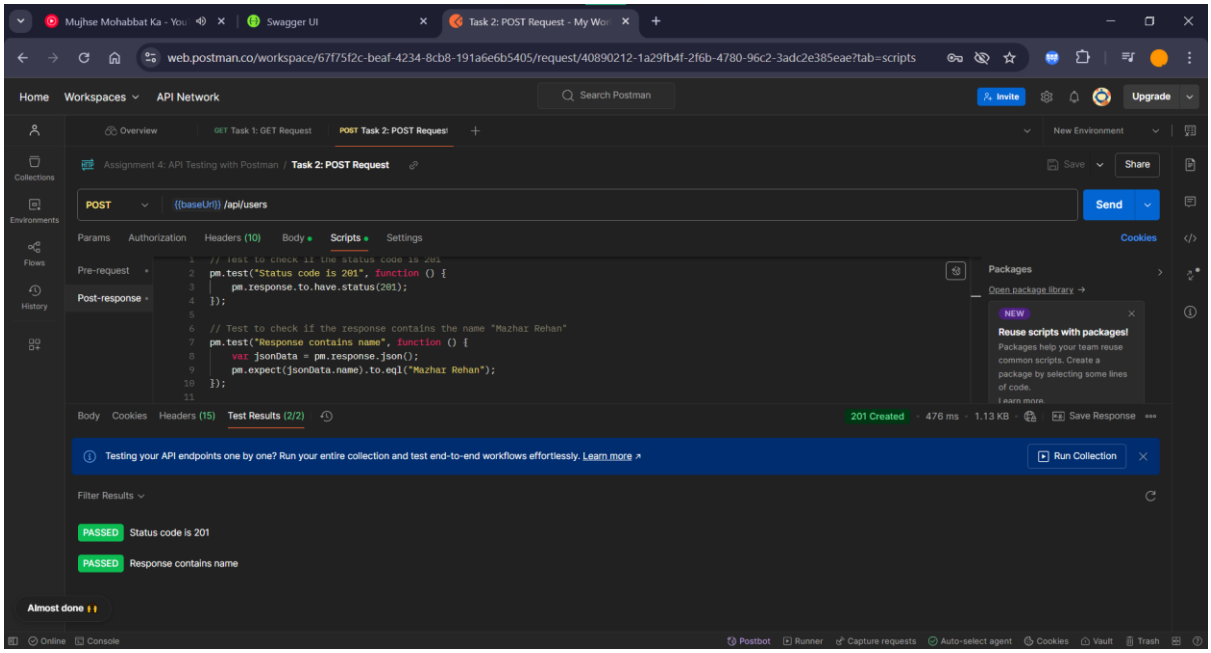
       We did not need any **pre-response script**. However, if necessary, we could set an environment variable like so:

```
// Set a dynamic user ID
pm.environment.set("userId", Math.floor(Math.random() * 1000));
```

    7. Post-response Script (add which is applicable)

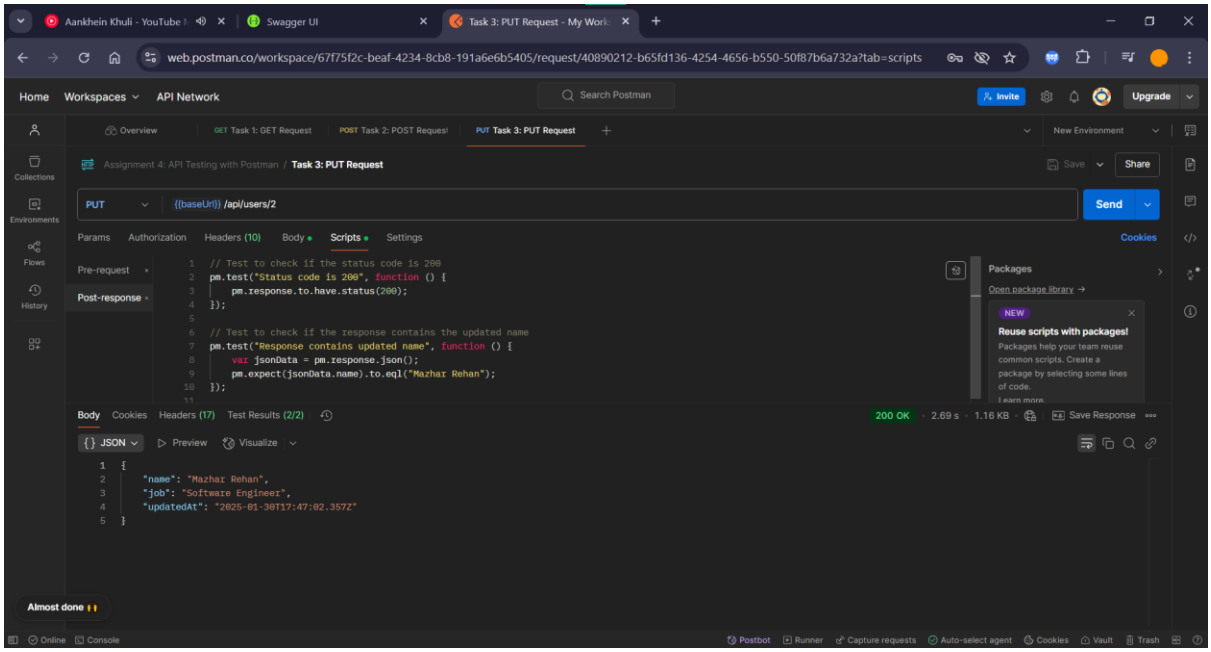
```
// Test to check if the status code is 201
pm.test("Status code is 201", function () {
    pm.response.to.have.status(201);
});

// Test to check if the response contains the name "Mazhar
Rehan"
pm.test("Response contains name", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.name).to.eql("Mazhar Rehan");
});
```

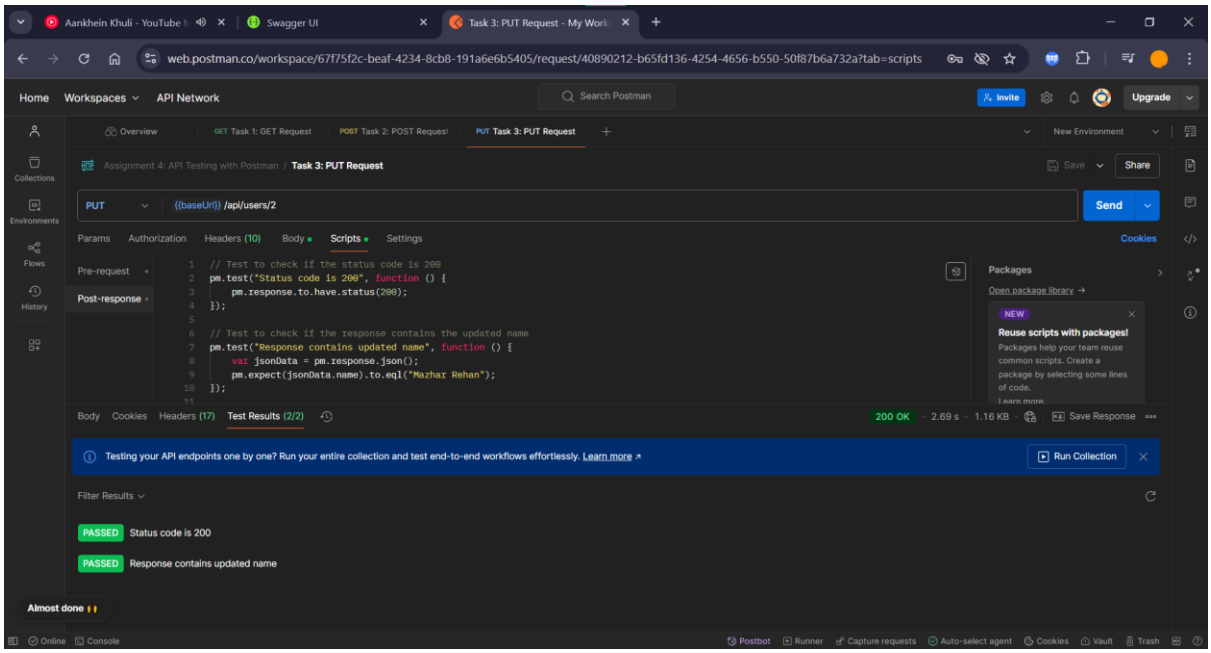

Task 2: POST Request - My Wor
web.postman.co/workspace/67f75f2c-beaf-4234-8cb8-191a6e6b5405/request/40890212-1a29fb4f-2f6b-4780-96c2-3adc2e385eae?tab=body
Home
Workspaces
API Network
Search Postman
Invite
Upgrade
Assignment 4: API Testing with Postman / Task 2: POST Request
Save
Share
POST
{{baseUrl}} /api/users
Send
Params
Authorization
Headers (10)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"name": "Mazhar Rehan",
"job": "Software Engineer"
}
Body
Cookies
Headers (15)
Test Results (1/2)
201 Created
637 ms
1.14 KB
Save Response
JSON
Preview
Visualize
{
"name": "Mazhar Rehan",
"job": "Software Engineer",
"id": "899",
"createdAt": "2025-01-30T17:33:28.739Z"
}
Almost done
Postbot
Ctrl Alt P


Swagger UI
Task 2: POST Request - My Wor
web.postman.co/workspace/67f75f2c-beaf-4234-8cb8-191a6e6b5405/request/40890212-1a29fb4f-2f6b-4780-96c2-3adc2e385eae?tab=scripts
Home
Workspaces
API Network
Search Postman
Invite
Upgrade
Assignment 4: API Testing with Postman / Task 2: POST Request
Save
Share
POST
{{baseUrl}} /api/users
Send
Params
Authorization
Headers (10)
Body
Scripts
Settings
Cookies
Pre-request
Post-response
pm.test("Status code is 201", function () {
pm.response.to.have.status(201);
});
// Test to check if the response contains the name "Mazhar Rehan"
pm.test("Response contains name", function () {
var jsonData = pm.response.json();
pm.expect(jsonData.name).to.eql("Mazhar Rehan");
});
Packages
Open package library
NEW
Reuse scripts with packages!
Packages help your team reuse common scripts. Create a package by selecting some lines of code.
Learn more
Body
Cookies
Headers (15)
Test Results (2/2)
201 Created
476 ms
1.13 KB
Save Response
Testing your API endpoints one by one? Run your entire collection and test end-to-end workflows effortlessly. Learn more
Run Collection
Filter Results
PASSED
Status code is 201
PASSED
Response contains name
Almost done

## Task 3: PUT Request

- Endpoint: `/api/users/{id}`
- Update a user's data.
- Take a screenshot of the execution.
- Share the following details in the Word document:
    1. Screenshot of the execution




2. HTTP Method

   PUT

3. Endpoint along with base URL.

   https://reqres.in/api/users/2

   here `id=2`

4. Header

   Content-Type: application/json

5. Body (include the JSON body sent)

```
{
  "name": "Mazhar Rehan",
  "job": "Software Engineer"
}
```

6. Pre-response Script (add which is applicable)

   We did not need any **pre-response script**. However, if necessary, we could set an environment variable like so:

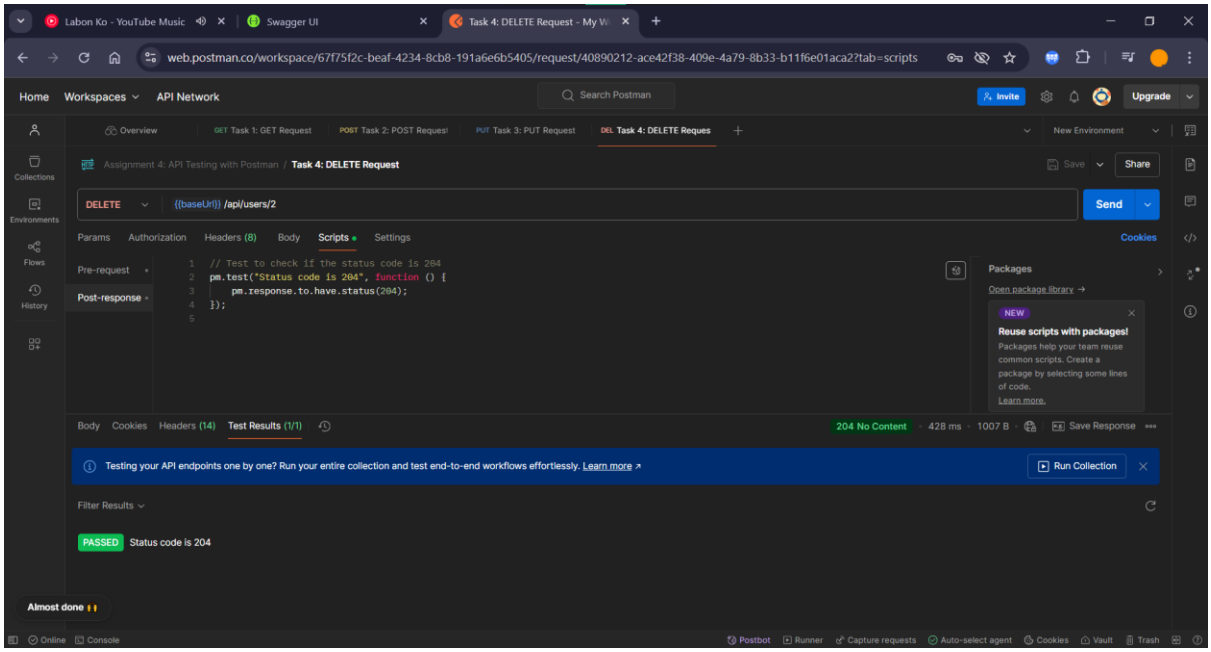
```
// Set a dynamic user ID
pm.environment.set("userId", Math.floor(Math.random() * 1000));
```

7. Post-response Script (add which is applicable)

```
// Test to check if the status code is 200
pm.test("Status code is 200", function () {
    pm.response.to.have.status(200);
});

// Test to check if the response contains the updated name
pm.test("Response contains updated name", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.name).to.eql("Mazhar Rehan");
});
```

## Task 4: DELETE Request

- Endpoint: `/api/users/{id}`
- Delete a user.
- Take a screenshot of the execution.
- Share the following details in the Word document:
    1. Screenshot of the execution




2. HTTP Method

   DELETE

3. Endpoint along with base URL.

   https://reqres.in/api/users/{id}

   here `id=2`

4. Header

   Content-Type: application/json

5. Body (if applicable, leave it blank if no body was sent)

   Blank (since **DELETE** requests typically do not include a body)

6. Pre-response Script (add which is applicable)

   We did not need any pre-response script**.** However, if necessary, we could set an environment variable like so:

```
// Set a dynamic user ID
pm.environment.set("userId", Math.floor(Math.random() * 1000));
```

7. Post-response Script (add which is applicable)

```
// Test to check if the status code is 204
pm.test("Status code is 204", function () {
    pm.response.to.have.status(204);
});
```

---