POSTMAN

- **What is Postman?**

  Postman is a scalable API testing tool that quickly integrates into CI/CD pipeline. API stands for Application Programming Interface which allows software applications to communicate with each other via API calls.

- **Working with GET Requests**

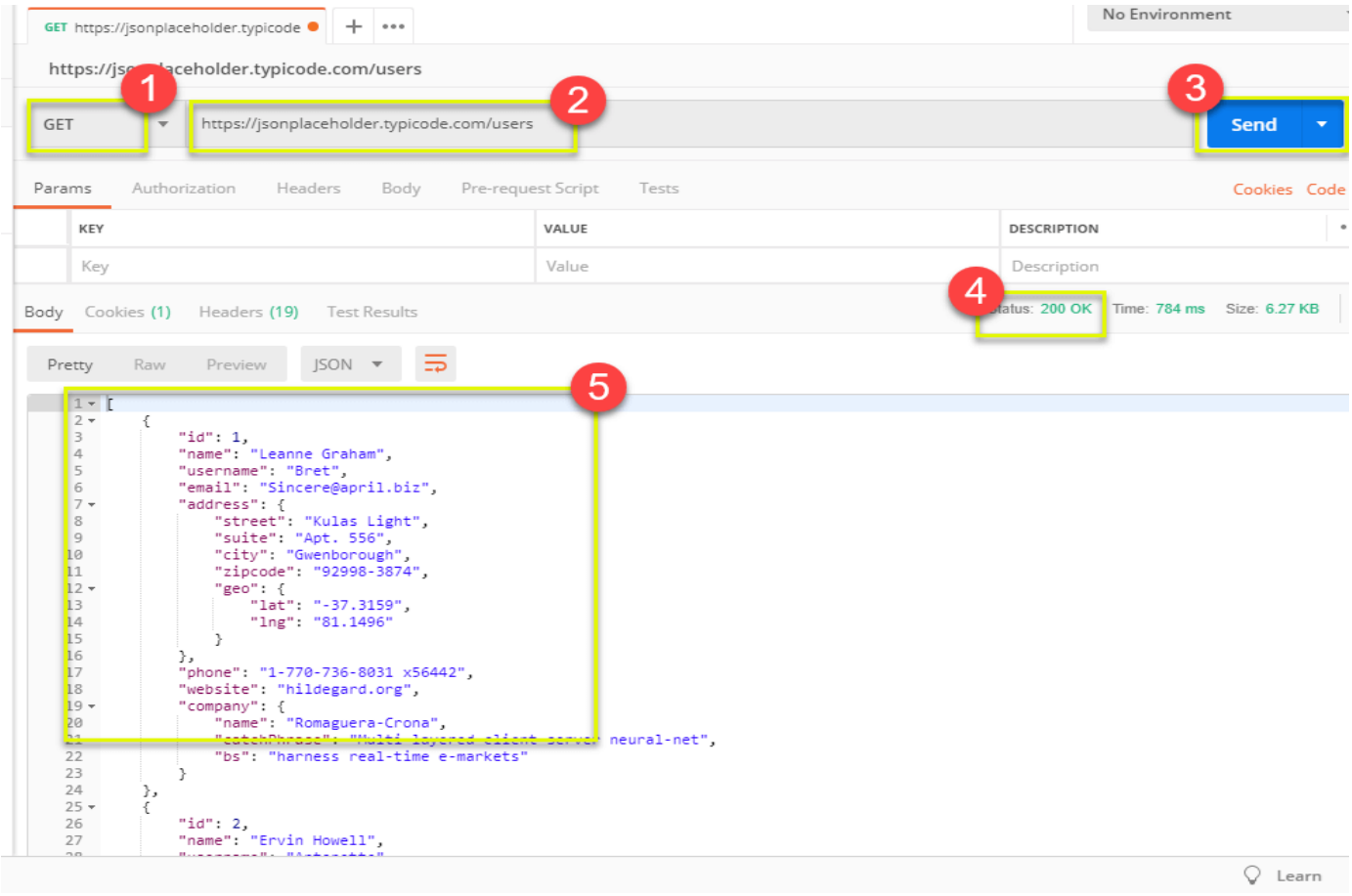
  Get requests are used to retrieve information from the given URL. There will be no changes done to the endpoint.

  We will use the following URL for all examples:

  https://jsonplaceholder.typicode.com/users

  In the workspace

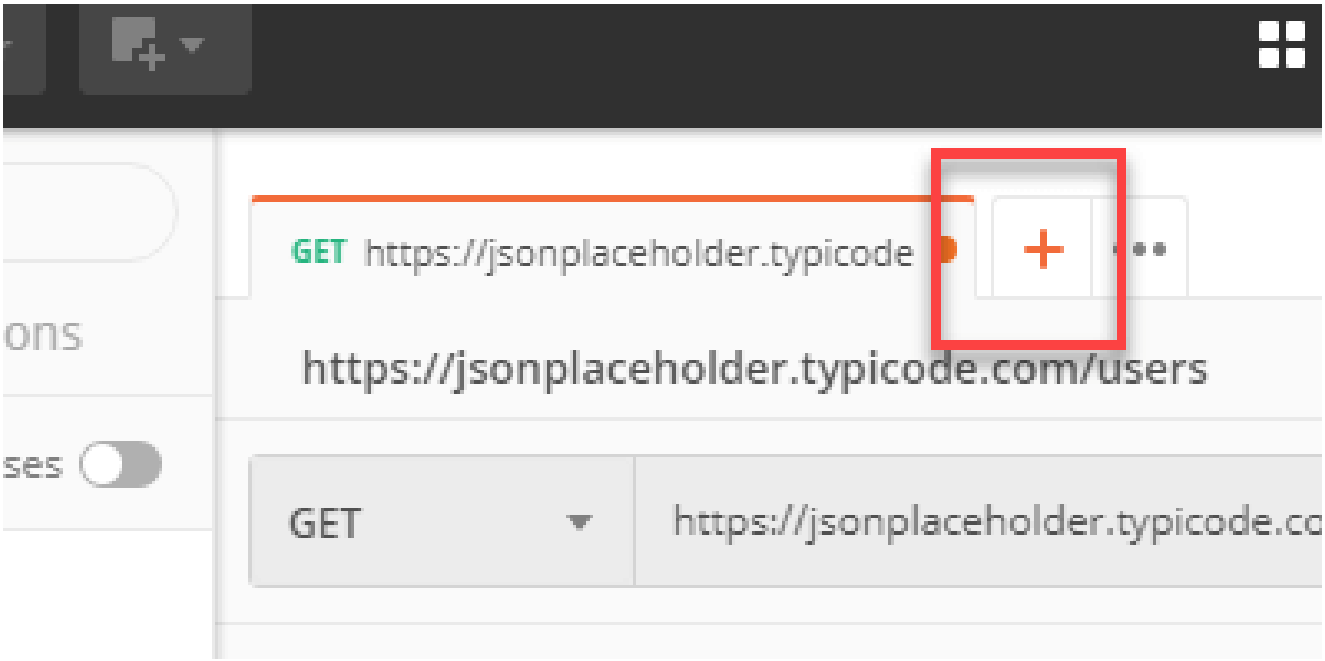
1. Set your HTTP request to GET.
2. In the request URL field, input link
3. Click Send
4. You will see 200 OK Message
5. There should be 10 user results in the body which indicates that your test has run successfully.



- **Working with POST Requests**

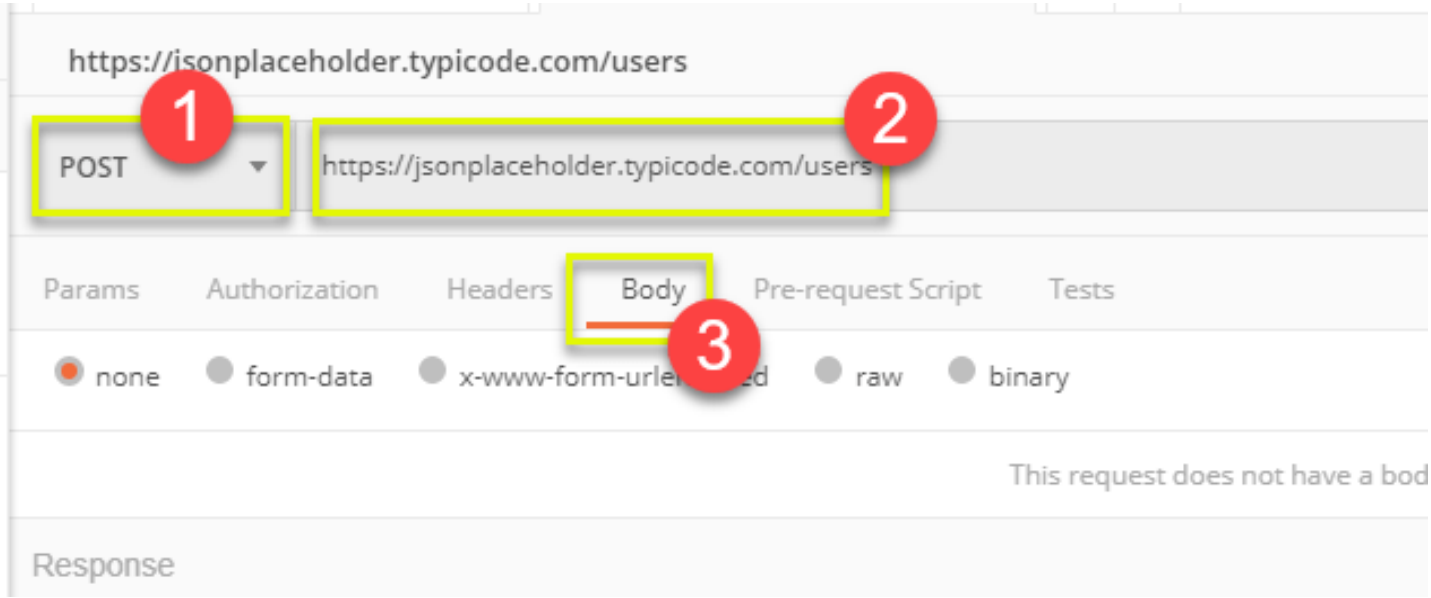
Post requests are different from Get request as there is data manipulation with the user adding data to the endpoint. Using the same data from the previous tutorial in Get request, let's now add our own user.

Step 1) Click a new tab to create a new request.



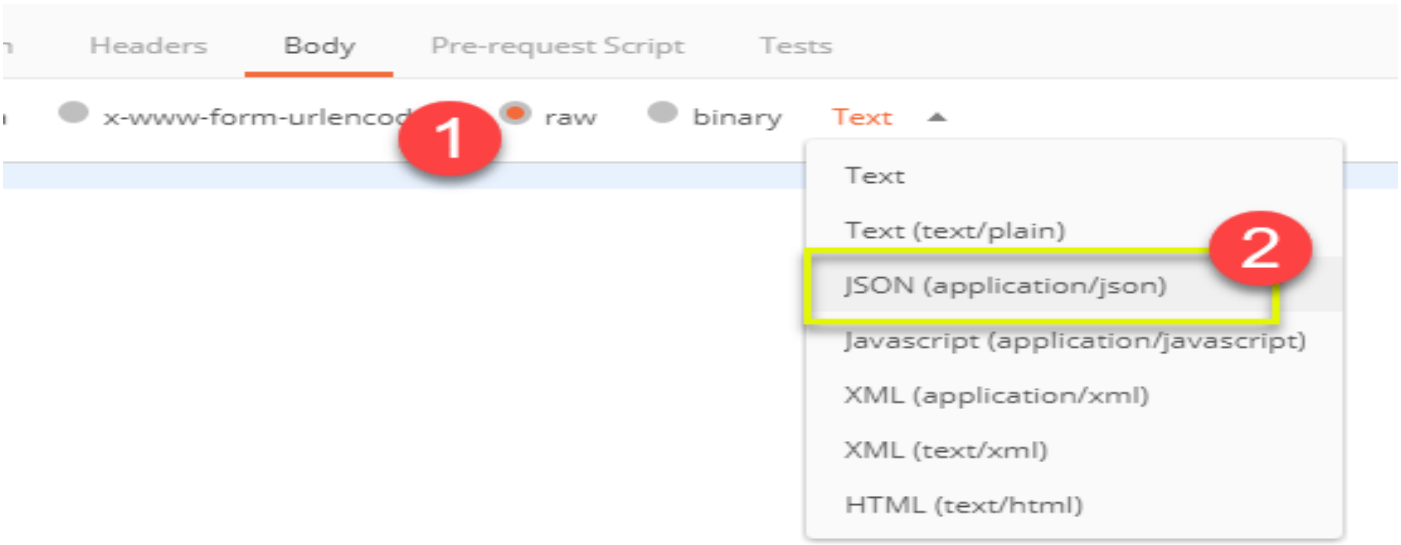
Step 2) In the new tab

1. Set your HTTP request to POST.
2. Input the same link in request url: https://jsonplaceholder.typicode.com/users
3. switch to the Body tab



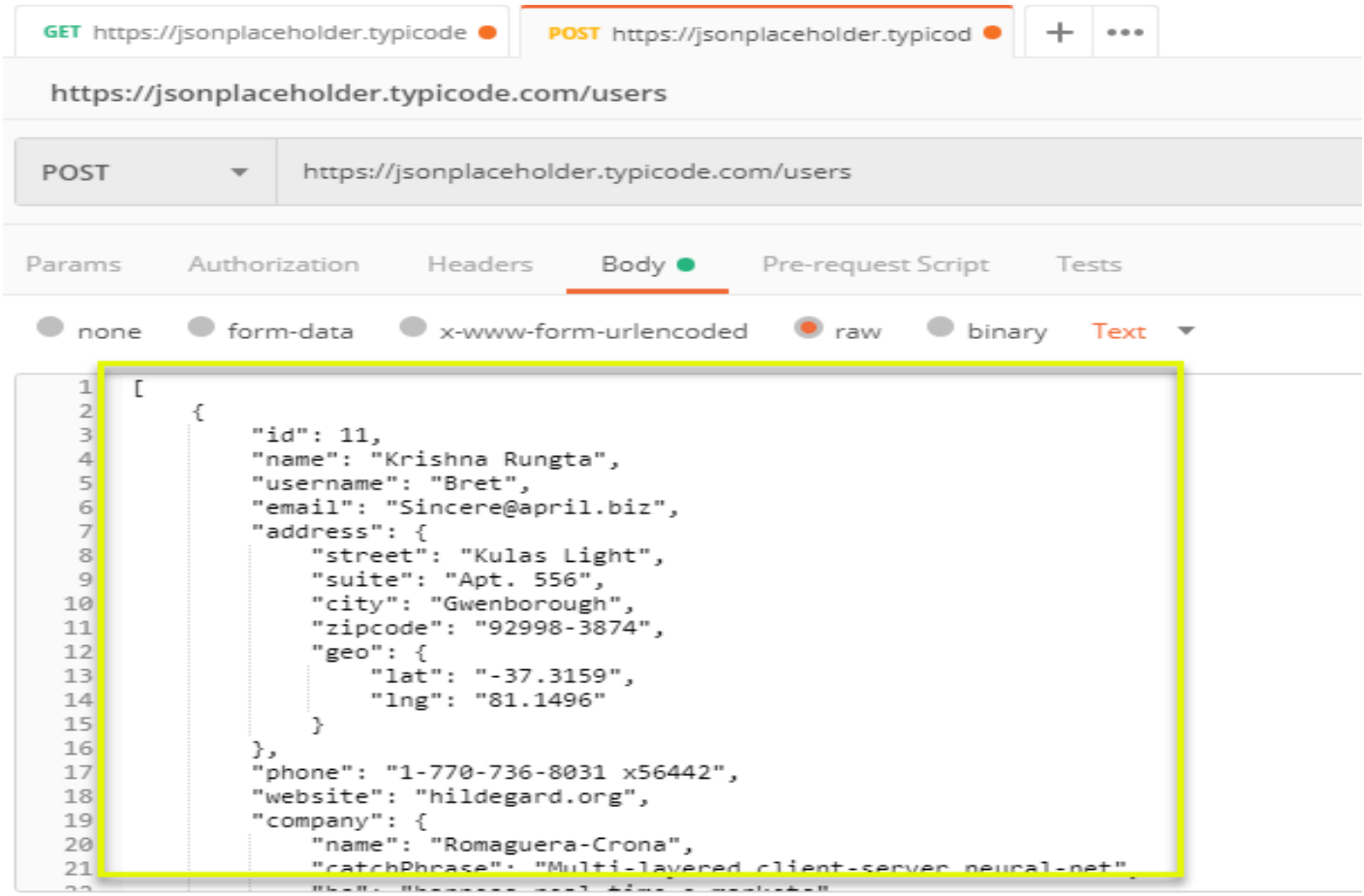
Step 3) In Body,

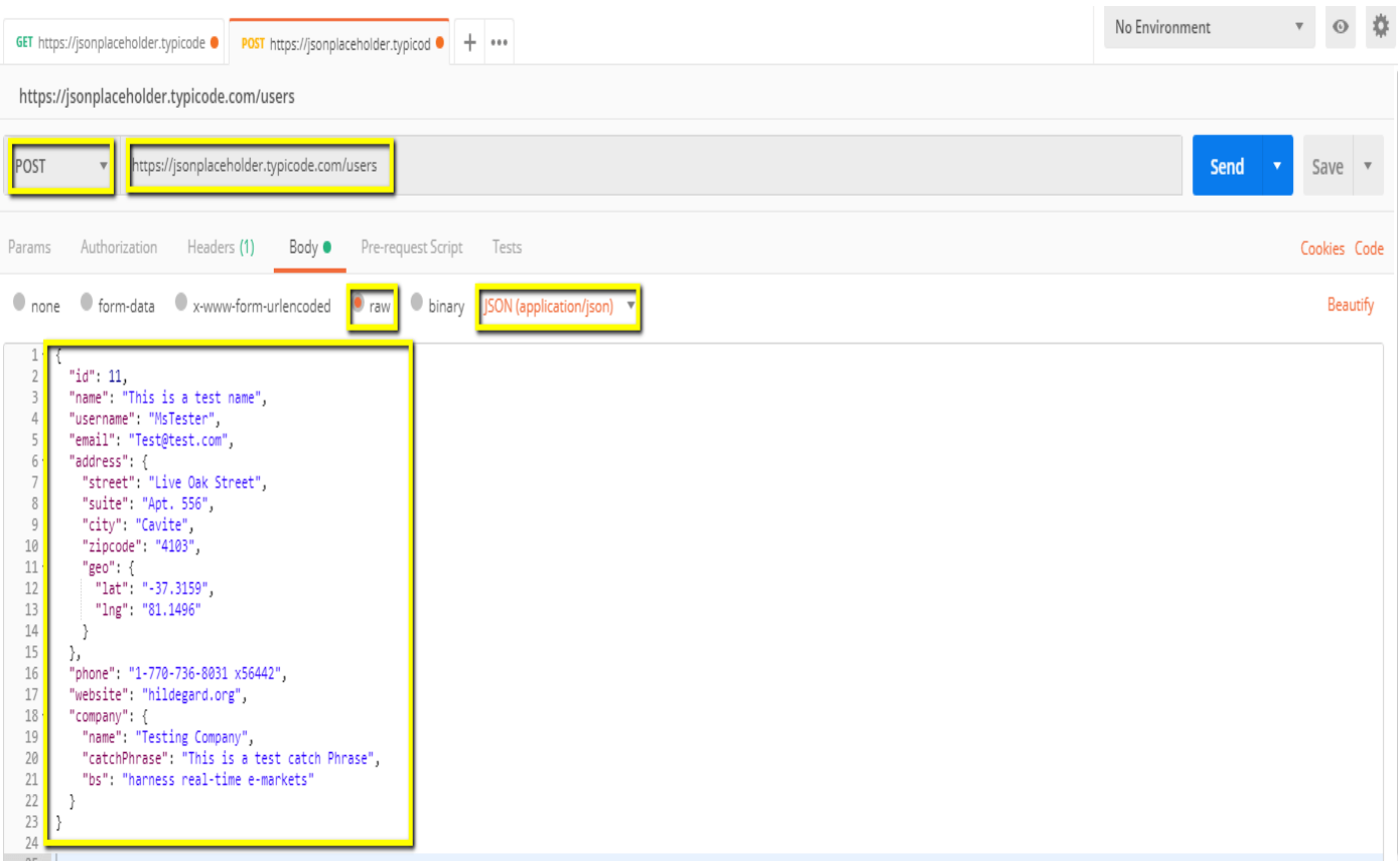
1. Click raw
2. Select JSON



Step 4) Copy and paste just one user result from the previous get request like below. Ensure that the code has been copied correctly with paired curly braces and brackets. Change id to 11 and name to any desired name. You can also change other details like the address.

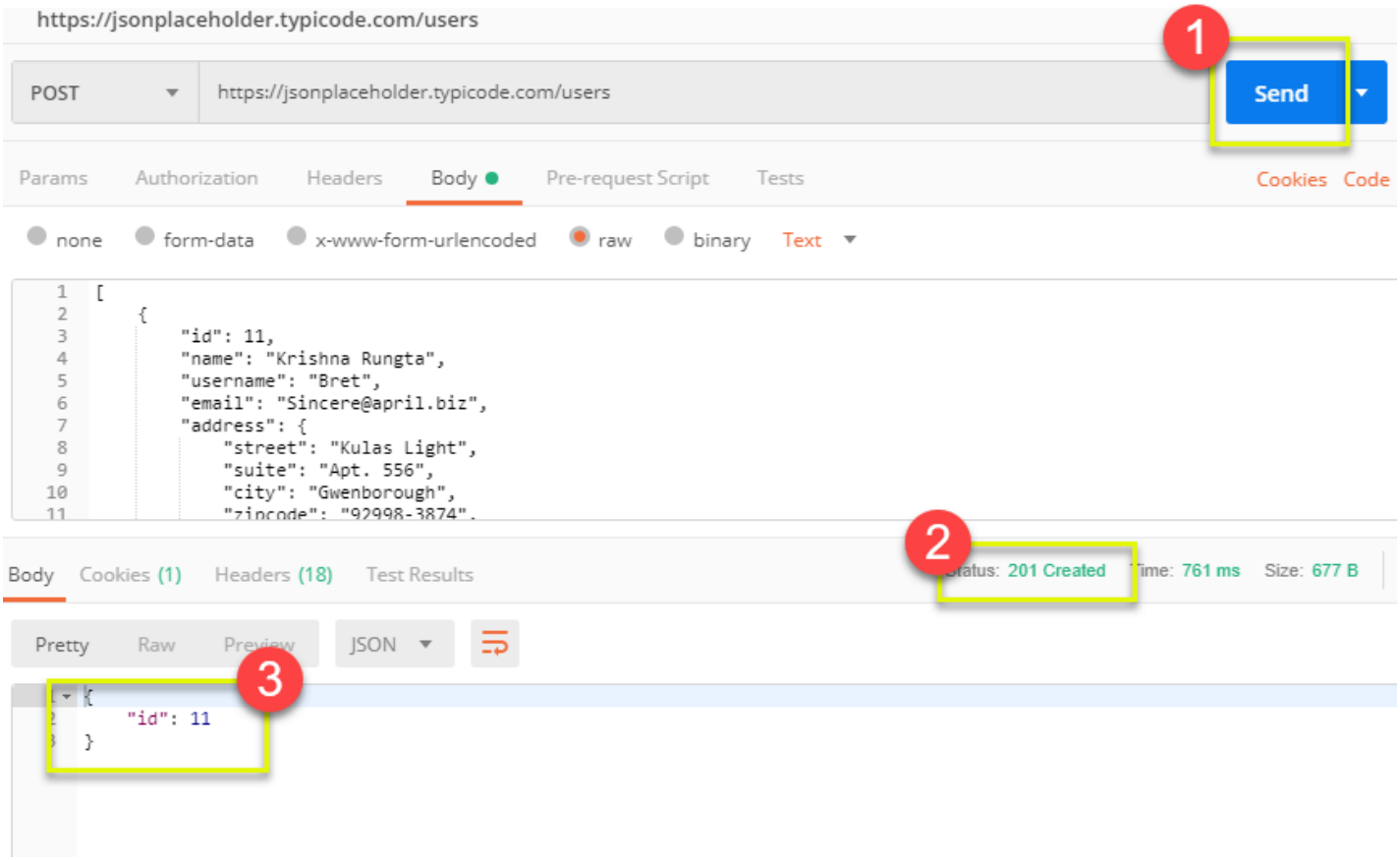

```
[
    {
        "id": 11,
        "name": "Krishna Rungta",
        "username": "Bret",
        "email": "Sincere@april.biz
        ",
        "address": {
            "street": "Kulas Light",
            "suite": "Apt. 556",
            "city": "Gwenborough",
            "zipcode": "92998-3874",
            "geo": {
                "lat": "-37.3159",
                "lng": "81.1496"
            }
        },
        "phone": "1-770-736-8031 x56442",
        "website": "hildegard.org",
        "company": {
            "name": "Romaguera-Crona",
            "catchPhrase": "Multi-layered client-server neural-net",
            "bs": "harness real-time e-markets"
        }
    }
]
```


GET https://jsonplaceholder.typicode
POST https://jsonplaceholder.typicod
https://jsonplaceholder.typicode.com/users
POST
https://jsonplaceholder.typicode.com/users
Params
Authorization
Headers
Body
Pre-request Script
Tests
none
form-data
x-www-form-urlencoded
raw
binary
Text
[
{
"id": 11,
"name": "Krishna Rungta",
"username": "Bret",
"email": "Sincere@april.biz",
"address": {
"street": "Kulas Light",
"suite": "Apt. 556",
"city": "Gwenborough",
"zipcode": "92998-3874",
"geo": {
"lat": "-37.3159",
"lng": "81.1496"
}
},
"phone": "1-770-736-8031 x56442",
"website": "hildegard.org",
"company": {
"name": "Romaguera-Crona",


GET https://jsonplaceholder.typicode
POST https://jsonplaceholder.typicod
No Environment
https://jsonplaceholder.typicode.com/users
POST
https://jsonplaceholder.typicode.com/users
Send
Save
Params
Authorization
Headers (1)
Body
Pre-request Script
Tests
Cookies
Code
none
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
Beautify
{
"id": 11,
"name": "This is a test name",
"username": "MsTester",
"email": "Test@test.com",
"address": {
"street": "Live Oak Street",
"suite": "Apt. 556",
"city": "Cavite",
"zipcode": "4103",
"geo": {
"lat": "-37.3159",
"lng": "81.1496"
}
},
"phone": "1-770-736-8031 x56442",
"website": "hildegard.org",
"company": {
"name": "Testing Company",
"catchPhrase": "This is a test catch Phrase",
"bs": "harness real-time e-markets"
}
}

Step 5) Next,

1. Click Send.
2. Status: 201 Created should be displayed
3. Posted data are showing up in the body.



- **Working with PUT Requests**

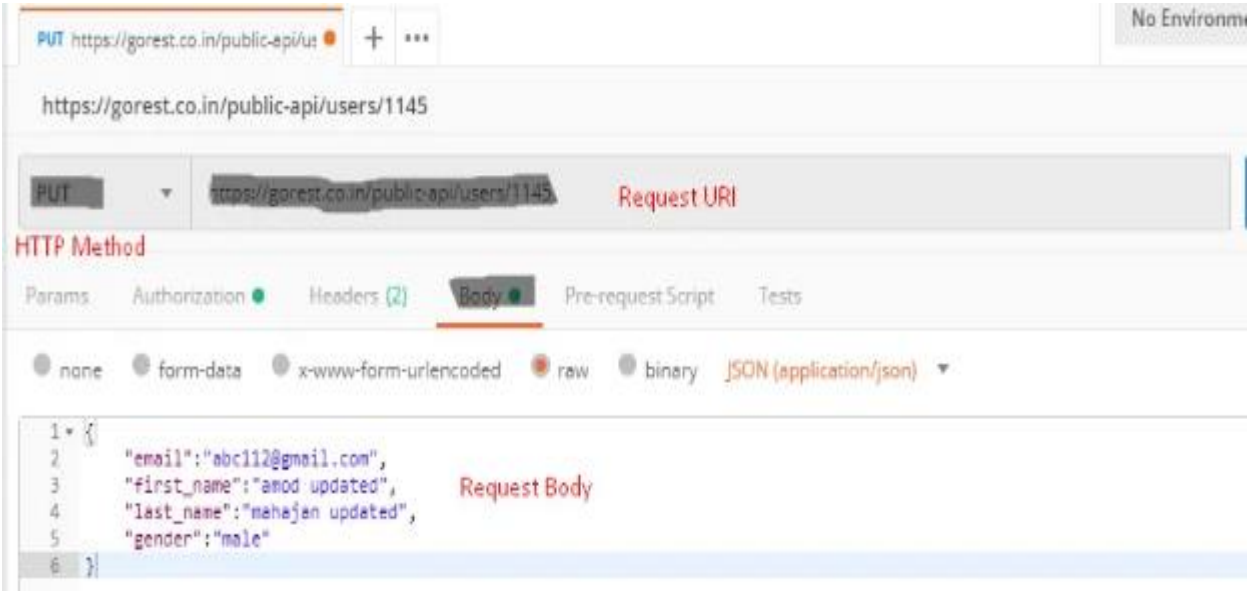
  An HTTP PUT method is used to primarily update the resource information but it also can be used to create a new resource (Depends on API development) if requested resource is not available. If PUT request is made to update resource, it should return 200 (OK) and 204 (No Content) status code. If PUT request is made to create a new resource, it must return a status code 201(Created).

  PUT is not a safe method as it performs data creation and modifications but it is idempotent as if we hit the same request again, it operates on same existing resource. But note here that a PUT request can be made as non-idempotent as well.

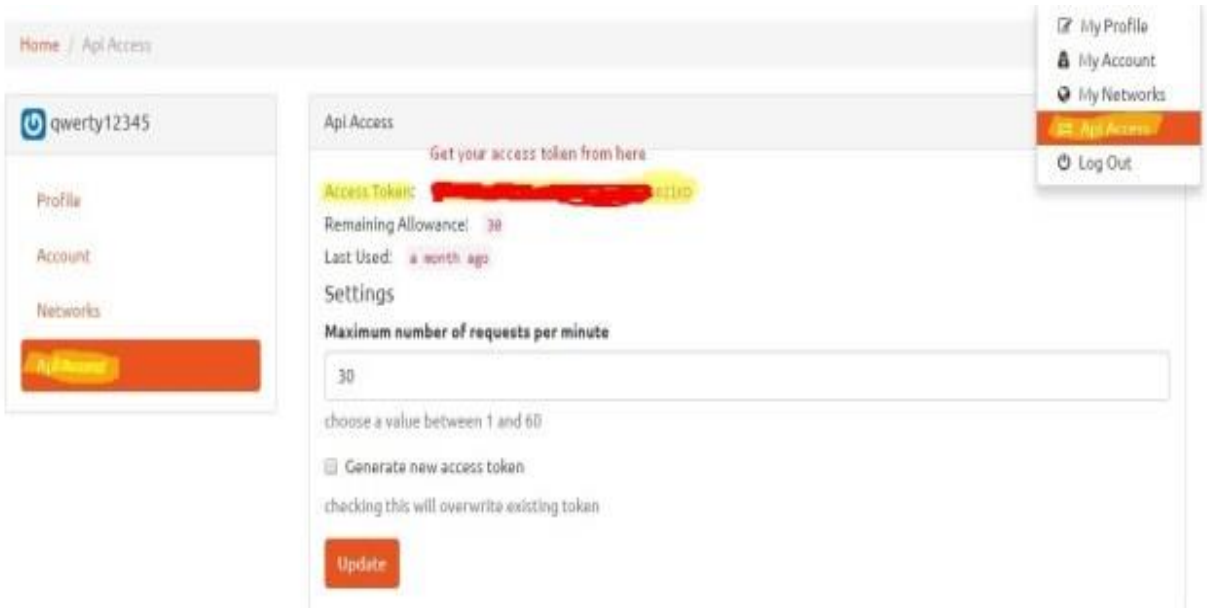
  URI used –https://gorest.co.in/public-api/users/1145

Detailed steps to hit a PUT request in Postman:

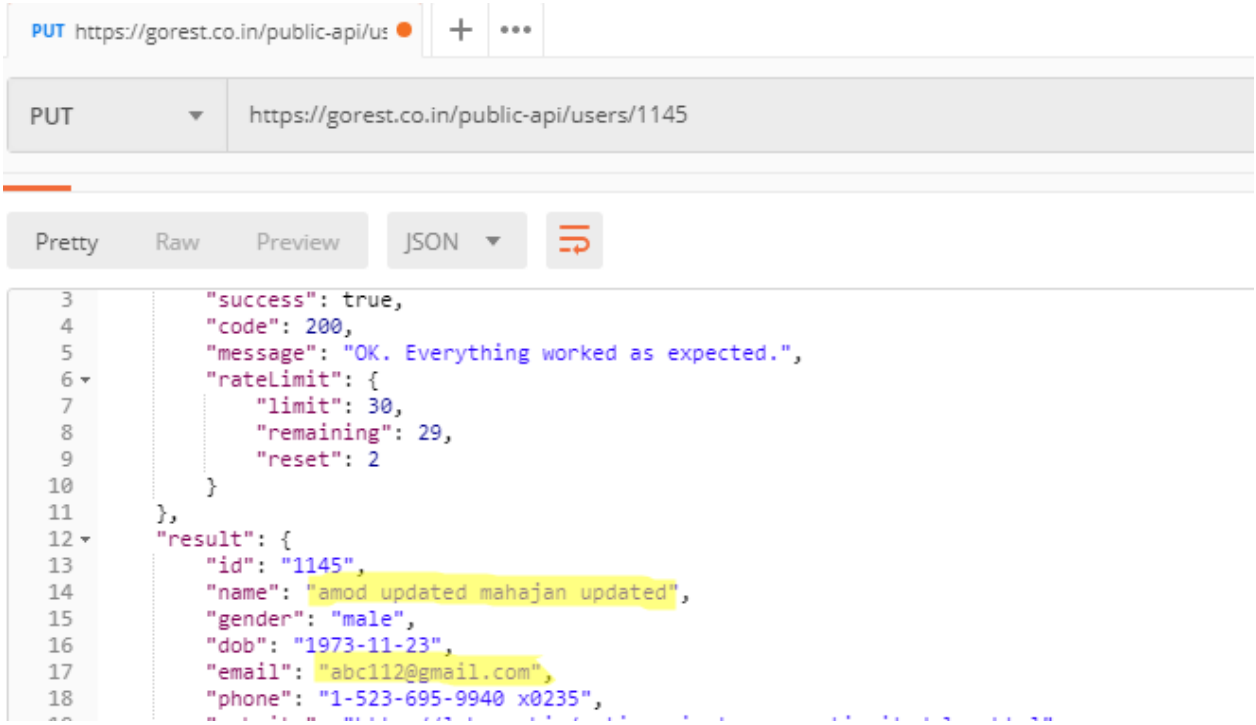
1.**Select the “PUT” in http methods drop down, pass PUT URI in address bar and copy request body under “Body” tab. You can see all these steps in details in previous post.**



2. **To hit PUT request, you need to pass authorisation. You need to pass access token for GoREST APIs. You need to sign up and go to profile to get access token as shown below:**



4. Hit the “Send” button. You can see response in “Body” tab of Response section of Postman. (Bottom part). Response in JSON format and it gives you updated details about user.



- **Sending body data**

You will need to send body data with requests whenever you need to add or update structured data. For example, if you're sending a request to add a new customer to a database, you might include the customer details in JSON. Typically you will use body data with PUT, POST, and PATCH requests.

The **Body** tab in Postman allows you to specify the data you need to send with a request. You can send various different types of body data to suit your API.

If you're sending body data, make sure you have the correct headers selected to indicate the content type your API may need to process the received data correctly.

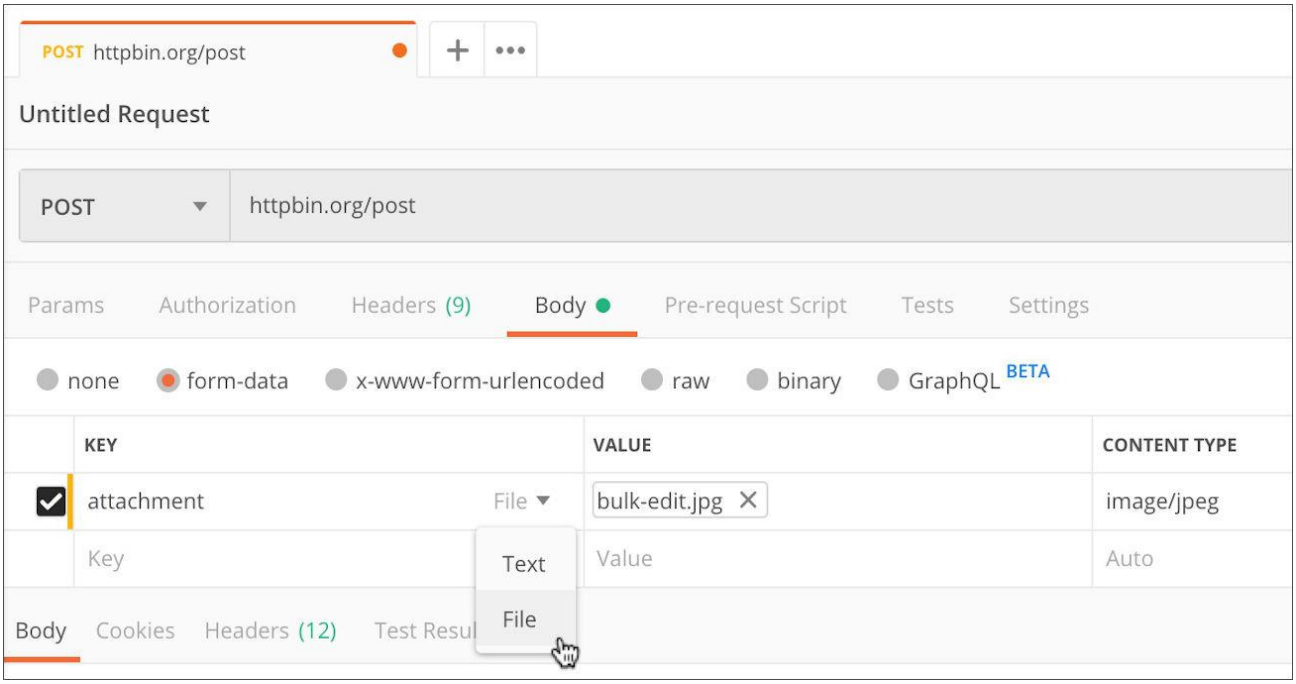
- For form-data and URL encoded body types, Postman will automatically attach the correct Content-Type header.
- If you use raw mode for your body data, Postman will set a header based on the type you select (e.g. text, json).
- If you manually select a Content-Type header, that value will take precedence over what Postman sets.
- Postman does not set any header type for the binary body type.

By default, Postman will select **None**—leave it selected if you do not need to send a body with your request.

Choose the data type you need for your request body—form data, URL-encoded, raw, binary, or GraphQL.

- **Form data**

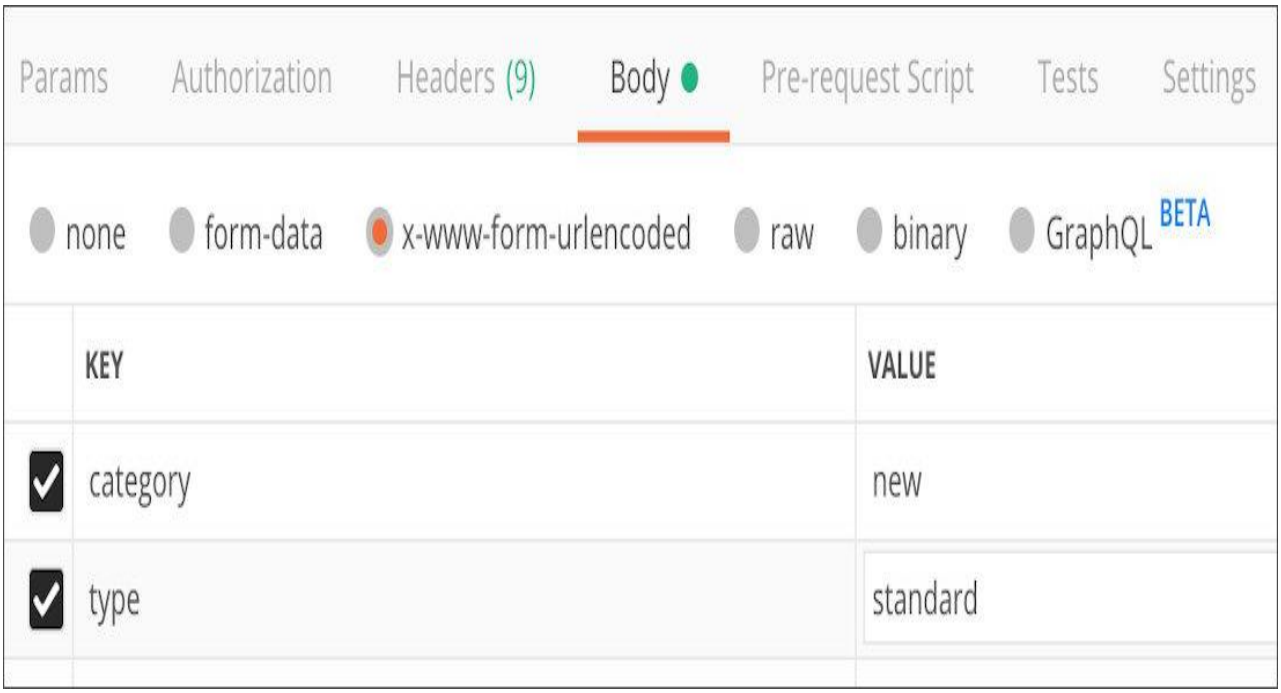
  Website forms often send data to APIs as multipart/form-data. You can replicate this in Postman using the form-data **Body** tab. Form data allows you to send key-value pairs, and specify the content type.



You can attach files using form data. When you repeatedly make API calls that send the same files, Postman will persist your file paths for subsequent use. This also helps you run collections that contain requests requiring file upload. Uploading multiple files each with their own content type is not supported yet.

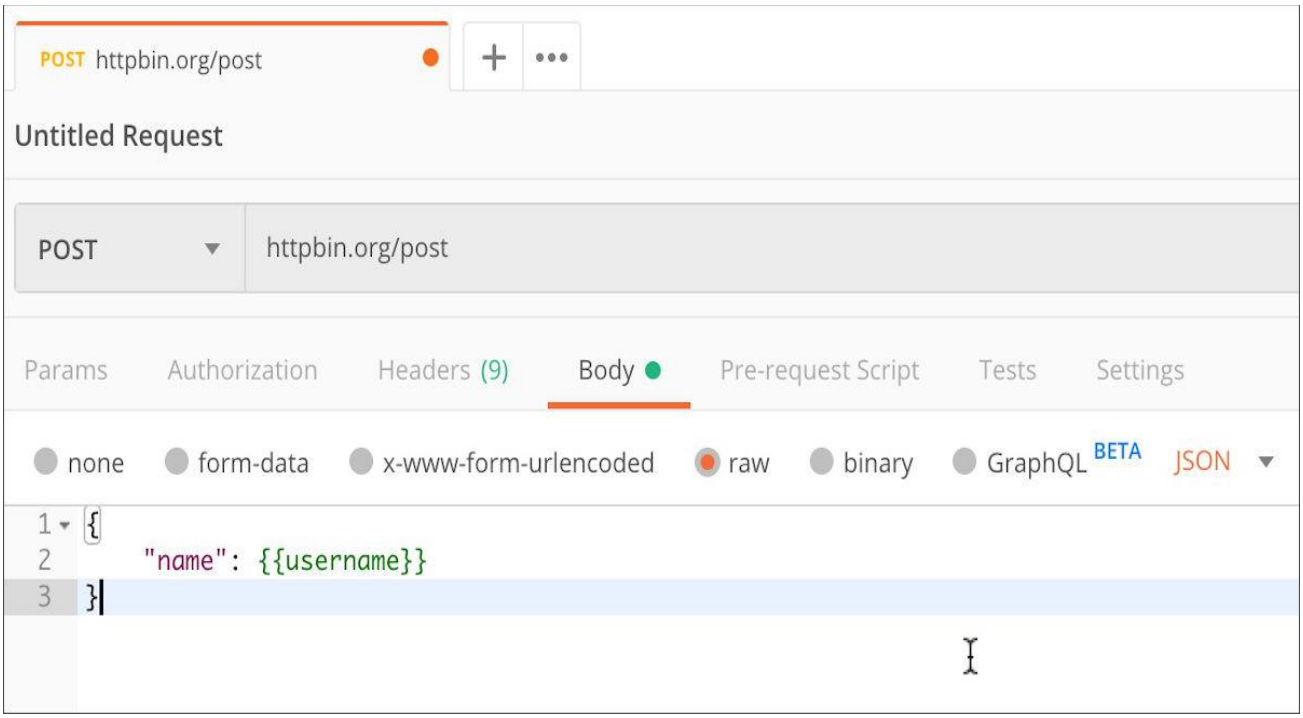
- **URL-encoded**

  URL-encoded data uses the same encoding as URL parameters. If your API requires URL-encoded data, select x-www-form-URL encoded in the **Body** tab of your request. Enter your key-value pairs to send with the request and Postman will encode them before sending.



- **Raw data**

You can use raw body data to send anything you can enter as text. Use the **raw** tab, and the type drop-down list to indicate the format of your data (**Text**, **JavaScript**, **JSON**, **HTML**, or **XML**) and Postman will enable syntax-highlighting as well as appending the relevant headers to your request.



You can set a content type header manually if you need to override the one Postman sends automatically.

You can use variables in your body data and Postman will populate their current values when sending your request.

Select text in the editor and press **CMD/CTRL + B** to beautify your XML/JSON.

- **Binary data**

  You can use **binary** data to send information you can't enter manually in the Postman editor with your request body, such as image, audio, and video files (you can also send text files).