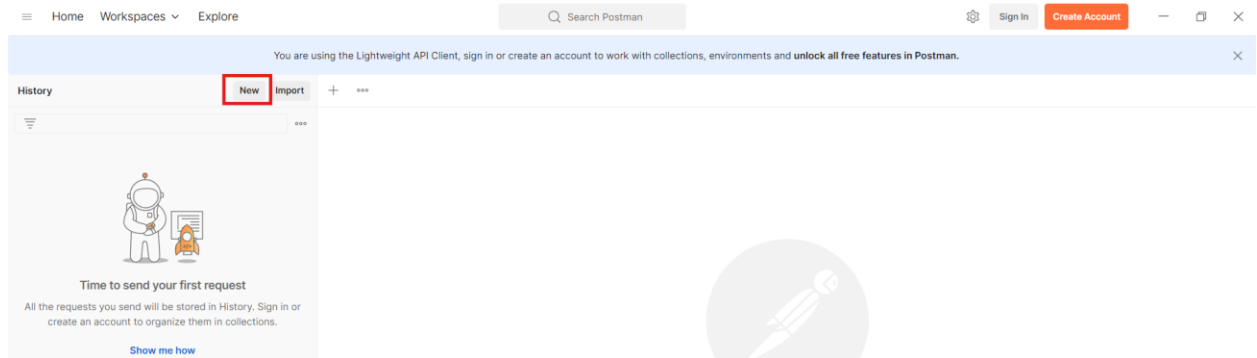
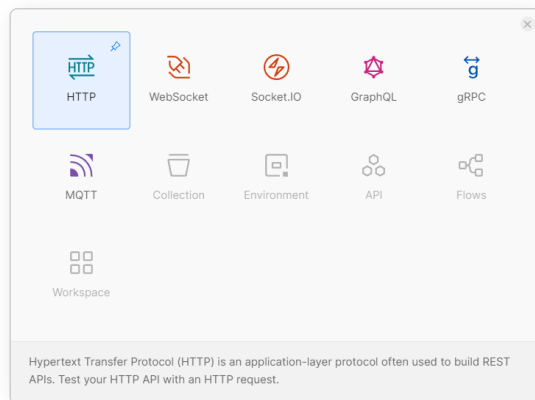


## Demo: API Blackbox Testing with Postman & Postbot AI

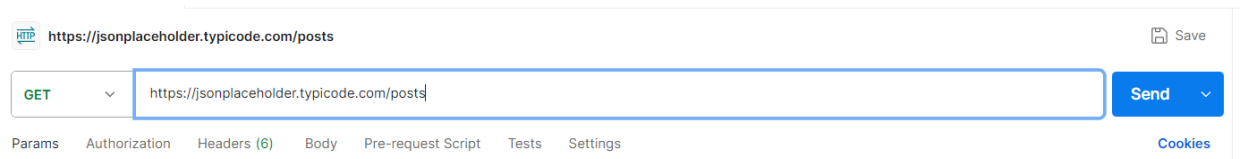
1. Open postman and click on the new button in the top left corner.



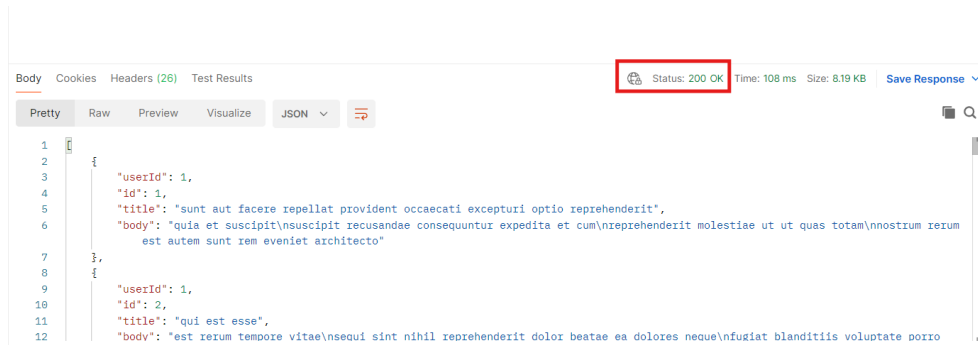
2. Next click on the HTTP request button in the pop up. This option allows us to specify the request type, such as GET or POST, which we'll use to interact with an API.



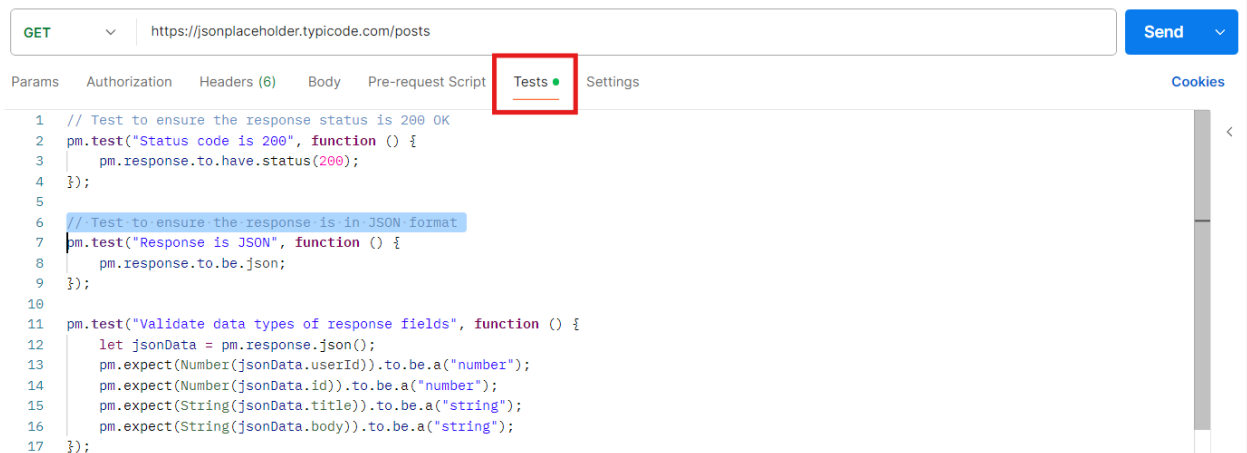
3. In the request URL field, enter **https://jsonplaceholder.typicode.com/posts**. This is a public API endpoint that provides dummy data for testing, so no authentication, headers, or parameters are needed. Set the method to GET. No headers or additional parameters are needed for this basic request. Then click on send.



- This should display the response body in the bottom section of the screen. Validate that the response was successful by checking the status of the response was **200**.



- Writing Tests for the GET Request:** Go to the Test tab located under the URL bar and create test cases to validate the response.



A few example test cases can be :

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

=====

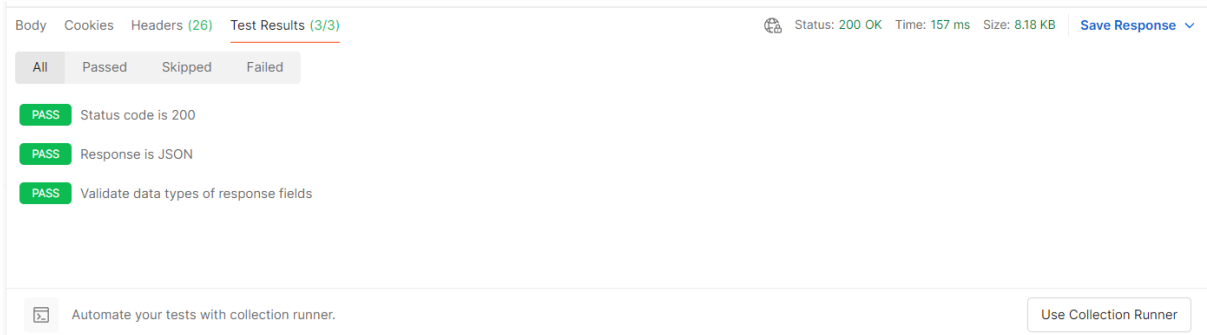
```
// Test to ensure the response is in JSON format
pm.test("Response is JSON", function () {
  pm.response.to.be.json;
```

```
});
```

```
=====
// Confirm that each field in the response body has the
expected data type.
```

```
pm.test("Validate data types of response fields", function
() {
  let jsonData = pm.response.json();
  pm.expect(Number(jsonData.userId)).to.be.a("number");
  pm.expect(Number(jsonData.id)).to.be.a("number");
  pm.expect(String(jsonData.title)).to.be.a("string");
  pm.expect(String(jsonData.body)).to.be.a("string");
});
```

6. Click on the send button again and check if your tests passed in the **Test Results** panel at the bottom of the screen.



Body Cookies Headers (26) Test Results (3/3) Status: 200 OK Time: 157 ms Size: 8.18 KB Save Response

All Passed Skipped Failed

- PASS Status code is 200
- PASS Response is JSON
- PASS Validate data types of response fields

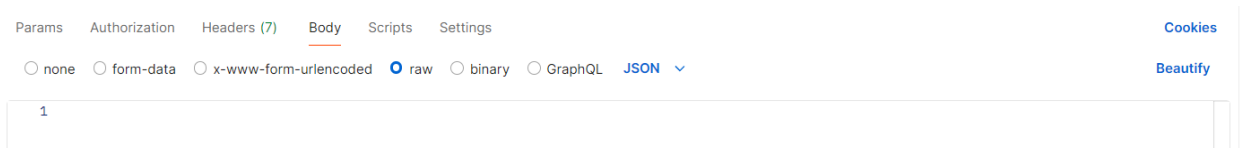
Automate your tests with collection runner. Use Collection Runner

7. **Set up the POST Request:** Next set up a post Request and make sure the url is the same it was previously. <https://jsonplaceholder.typicode.com/posts> , but change the request type.

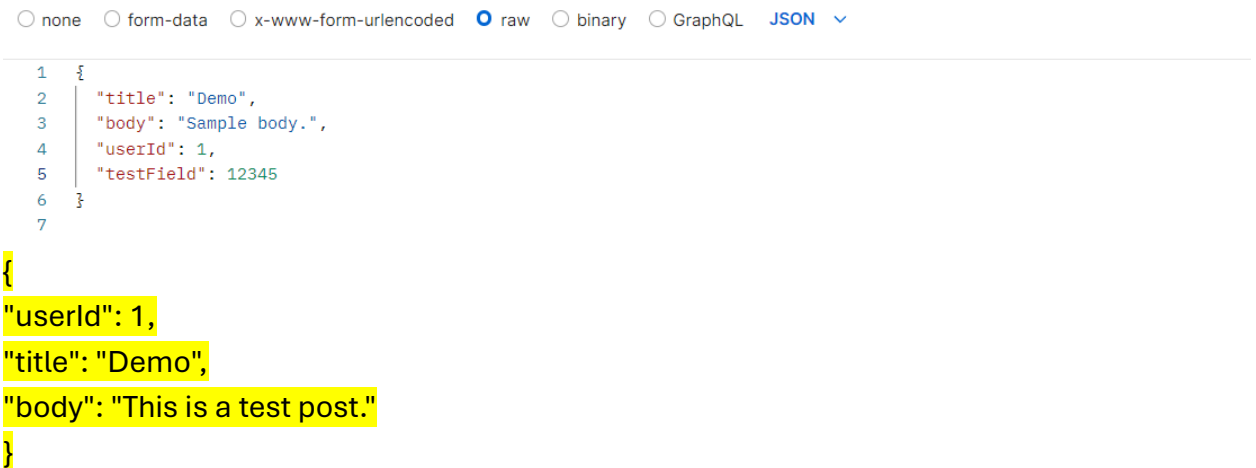


POST https://jsonplaceholder.typicode.com/posts Send

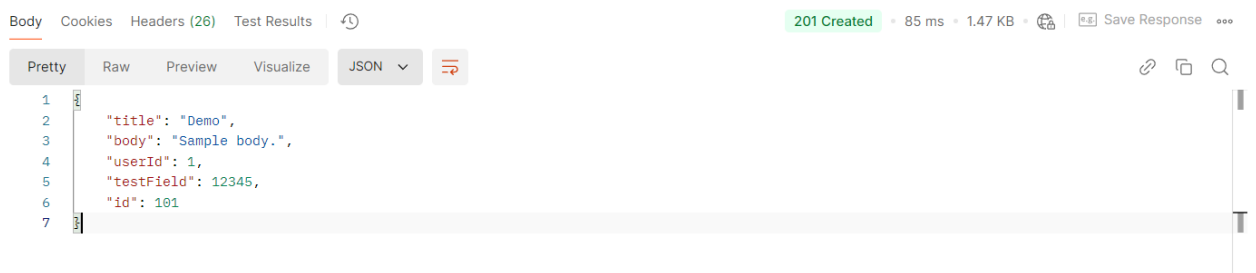
8. Go to the Body tab, select raw, and set the format to **JSON** from the dropdown next to "raw".



9. Add some json data and click on Send. **Note:** the JSON data can be anything.  
Example:



10. You should now see the data you sent in the post request to be displayed in the response body and the response code should be **201 created**.



11. **Writing Tests for the POST Request:** We will now write tests for the post request the same way we did for the get request.  
**Note :** If the Test tab does not show you can go to the 'Scripts' tab and select 'Post-Response'.

POST | https://jsonplaceholder.typicode.com/posts | Send

Params | Authorization | Headers (8) | Body | **Scripts** | Settings

Pre-request

Post-response

```
1 // Test to check if the status code is 201 Created
2 pm.test("Status code is 201", function () {
3     pm.response.to.have.status(201);
4 });
5
6 // Test to confirm the response contains the title you sent
7 pm.test("Response contains title", function () {
8     pm.expect(pm.response.json().title).to.eql("Demo");
9 });
10
```

Cookies

Example tests:

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

```
// Test to confirm the response contains the title you sent
pm.test("Response contains title", function () {
    pm.expect(pm.response.json().title).to.eql("Demo");
});
```

12. **Using Postbot for Automated Test Suggestions:** While in the Tests tab, click on the magic wand icon to invoke Postbot. Review and incorporate suggested tests into your test suite.

The screenshot displays the Postman interface with the 'Scripts' tab selected. The 'Post-response' section contains the following JavaScript code:

```
1 // Test to check if the status code is 201 Created
2 pm.test("Status code is 201", function () {
3   pm.response.to.have.status(201);
4 });
5
6 // Test to confirm the response contains the title you sent
7 pm.test("Response contains title", function () {
8   pm.expect(pm.response.json().title).to.eql("Demo");
9 });
10
```

Below the code editor, the 'Test Results (7/7)' tab is active, showing a list of passed tests:

- PASS** Response time is less than 500ms
- PASS** Response has the required fields
- PASS** Title is a non-empty string
- PASS** Body is a non-empty string
- PASS** UserId is a positive integer

On the right side, a 'Postbot' chat window is open, displaying a message: "I have added tests to check response time, required fields, and their properties. Do these tests look good to you, or is there anything else you'd like to add?". The 'Add more tests' button in the chat is highlighted with a red box.

13. **Save and Run Your Tests:** Save your request. Click the “Send” button again with the tests you’ve written to see the results in the “Test Results” tab below the response area.