

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH



BÁO CÁO BÀI TẬP LỚN KIỂM THỬ PHẦN MỀM
ĐỀ TÀI: KIỂM THỬ WEBSITE OpenWeather

GVHD: Nguyễn Thị Ngọc Thanh
SINH VIÊN THỰC HIỆN: Phạm Nguyễn Khả Tú
MÃ SỐ SINH VIÊN : 215151050529
KHOA : Công Nghệ Công Tin
LỚP: DH21IT02

TPHCM, 05/2024

Mục Lục

I. Môi trường kiểm thử.....	1
II. Test API sử dụng Postman	1
1. Chuẩn bị:.....	1
2. Triển khai viết các TestCase.....	10
2.1. Chức năng 1: Kiểm thử chức năng lấy tất cả Repos trong GitHub của user tạo	10
2.2. Chức năng 2: Kiểm thử chức năng tạo một Repos mới.....	13
2.3. Chức năng 3: Kiểm thử chức năng sửa tên Repos.....	16
2.4. Chức năng 4: Kiểm thử chức năng xóa một Repos	21

Danh mục hình ảnh

Hình IV.1 . Đăng nhập Posman.....	1
Hình IV.2 .Tạo workspace	2
Hình IV.3 .Tạo collection	3
Hình IV.4 . Đăng nhập Github	4
Hình IV.5 . Tạo Token	4
Hình IV.6 . Tạo Token	5
Hình IV.7 . Tạo Token	5
Hình IV.8 . Đặt tên cho token	6
Hình IV.9 . Cấp quyền cho token.....	6
Hình IV.10 . Cấp quyền cho token.....	7
Hình IV.11 . Cấp quyền cho token.....	8
Hình IV.12 . Tạo token	8
Hình IV.13 . Tạo token thành công.....	8
Hình IV.14 . Tạo biến môi trường cần thiết	9
Hình IV.15 . Biến môi trường cần thiết.....	9
Hình IV.16 . Add request	10
Hình IV.17 . Nhập Url API và Phương thức	11
Hình IV.18 . Cấp token đã tạo.....	11
Hình IV.19 . Add Request.....	14
Hình IV.20 . Cấp token đã được tạo trước	14
Hình IV.21 . Điền thông tin cho phương thức Post	15
Hình IV.22 . Send thành công.....	15
Hình IV.23 . Viết testcase	16
Hình IV.24 . Add request phương thức Patch	17
Hình IV.25 . Cấp token đã tạo.....	17
Hình IV.26 . Nhập Repo muốn sửa tên	18
Hình IV.27 . Update lại tên mới cho Repo	19
Hình IV.28 . TestCase API kiểm tra sửa tên repo	20
Hình IV.29 . Kết quả testcase API	20
Hình IV.30 . Add request phương thức Delete.....	21
Hình IV.31 . Cấp token đã tạo trước đó	21
Hình IV.32 . Chọn Repo muốn delete	22
Hình IV.33 . Send kiểm tra kết quả delete thành công	22
Hình IV.34 . Testcase API chức năng Delete	23
Hình IV.35 . Kết quả chạy các testcase API.....	24

Hình IV.36 .Kết quả chạy các testcase API.....	24
Hình IV.37 .Kết quả chạy các testcase API.....	25
Hình IV.38 .Kết quả chạy các testcase API.....	26

I. Môi trường kiểm thử

PC (Intel(R) Core(TM) i5-1035G1 CPU ,Ram: 8GB)

OS (Windows 11, Version: 22H2)

Browser (Chrome -Version 124.0.6367.201)

IDE (Visual Studio Code 2022)

Tạo Unit Test Project (Framework: .NET Framework 4.7.2)

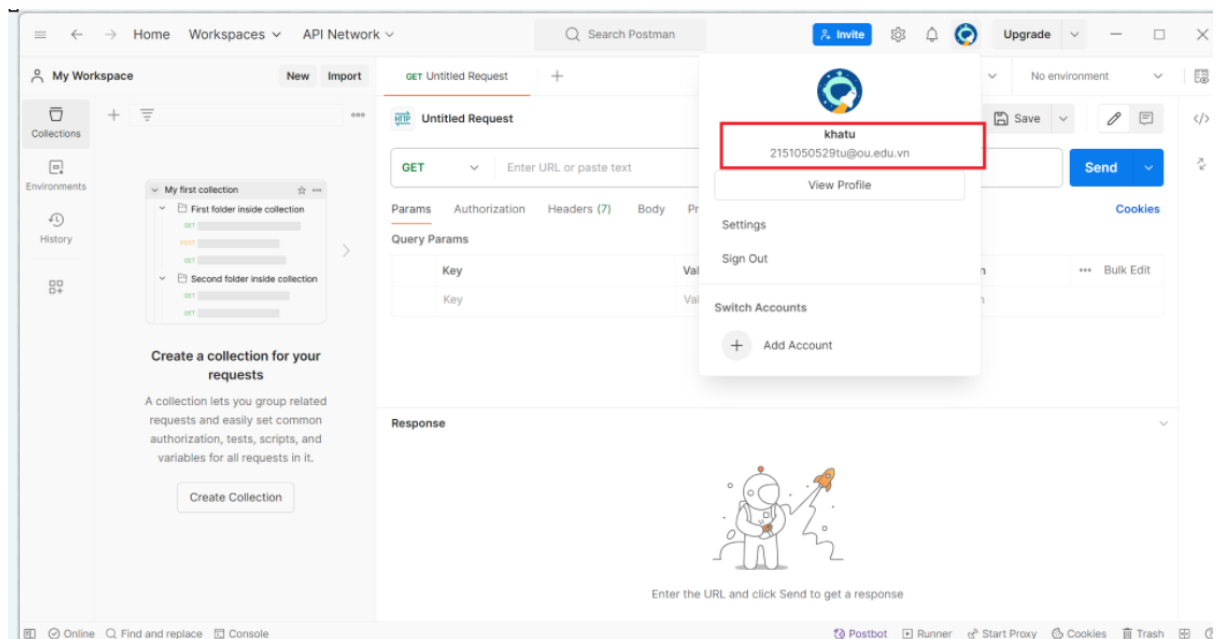
Cài đặt thư viện NuGet cho IDE:

- Selenium.WebDriver – Version: 4.20.0
- Selenium.WebDriver.ChromeDriver – Version: 124.0.6367.6000

II. Test API sử dụng Postman

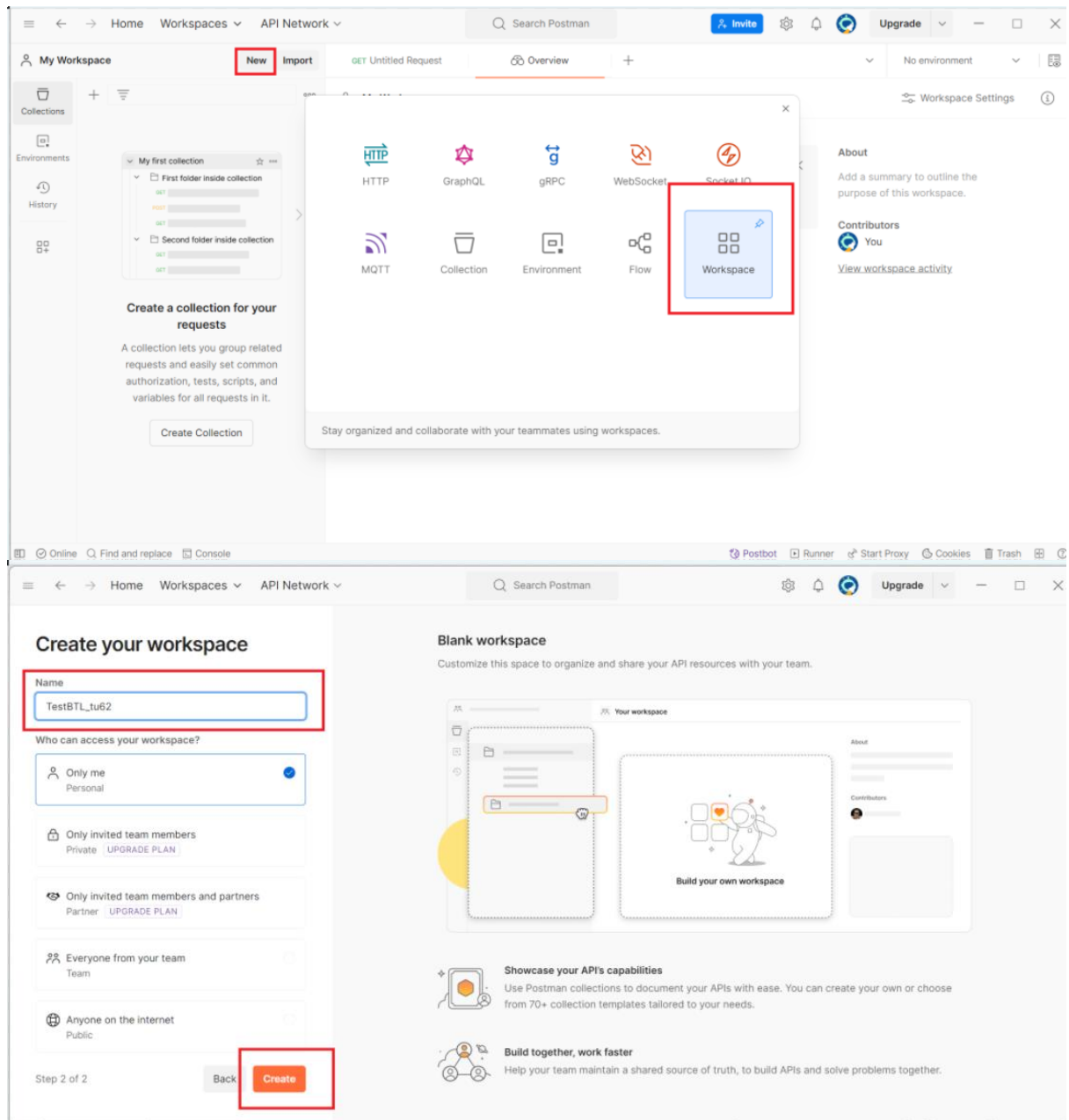
1. Chuẩn bị:

Bước 1: Đăng nhập vào postman



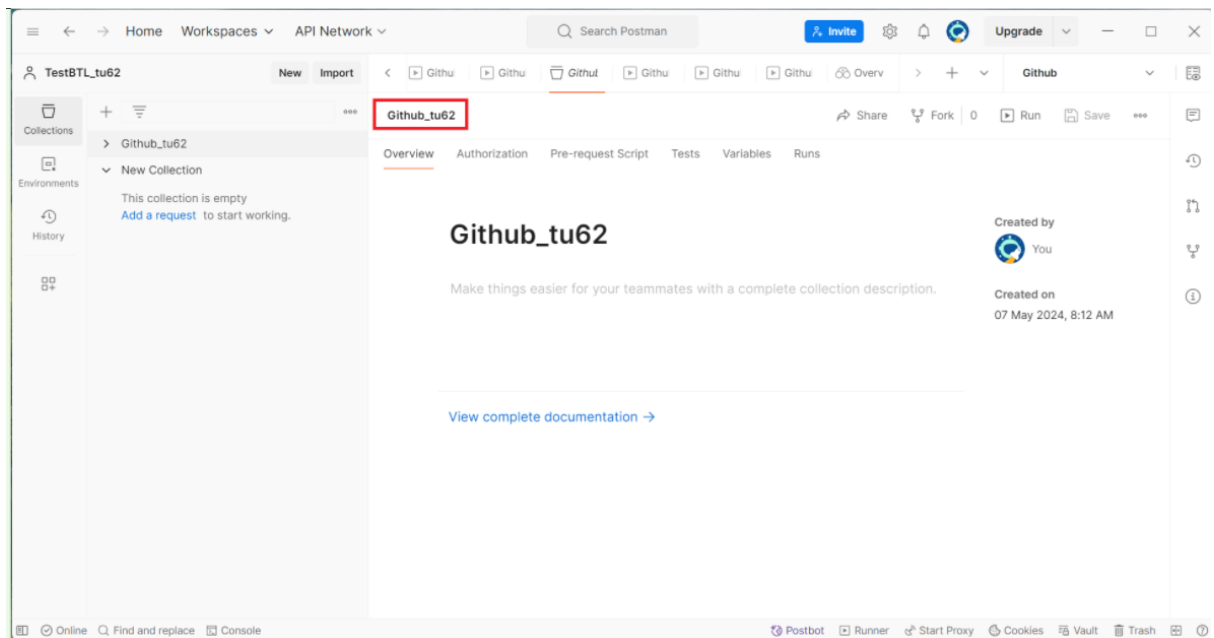
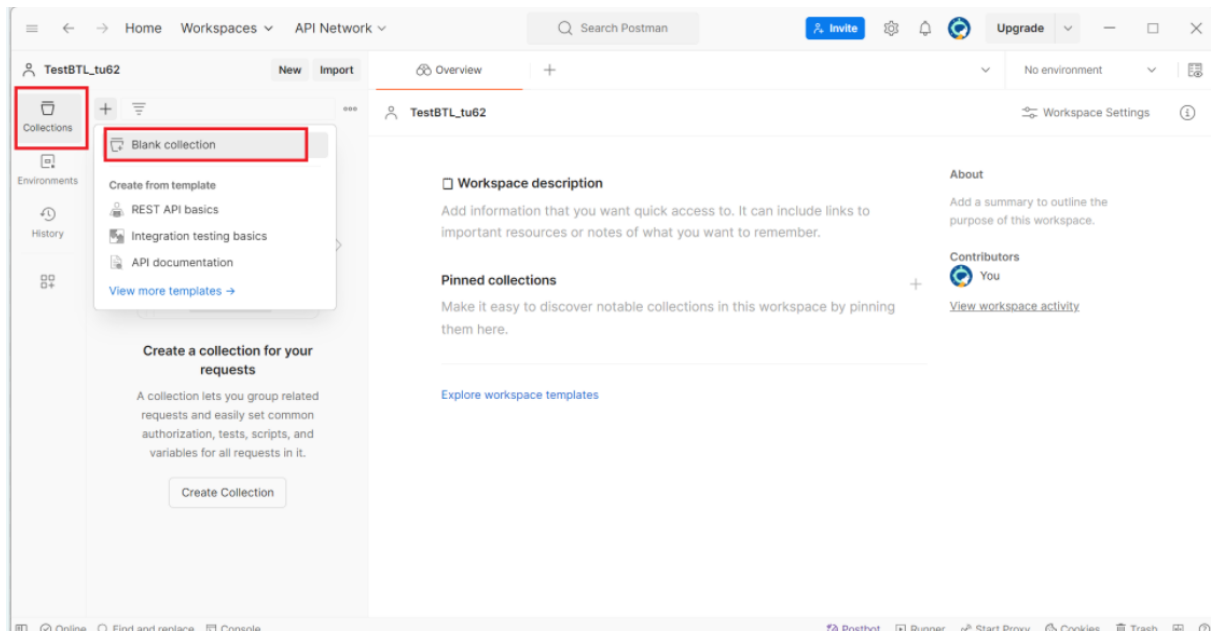
Hình II.1. Đăng nhập Posman

Bước 2: Tạo một workspace mới



Hình II.2.Tạo workspace

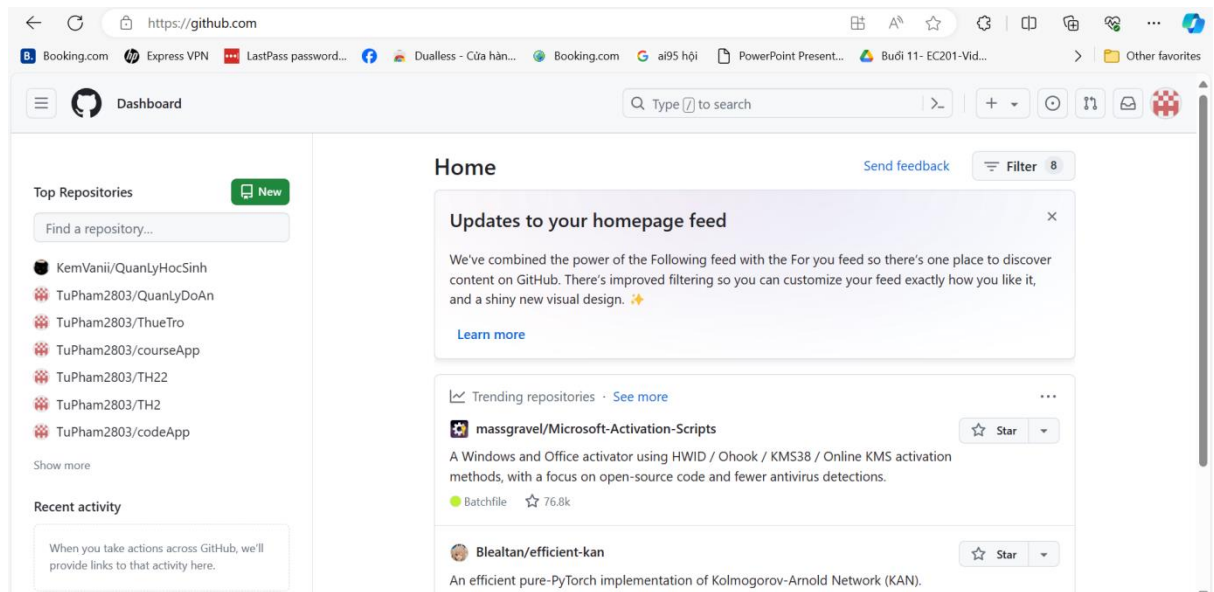
Bước 3: Tạo một collection mới



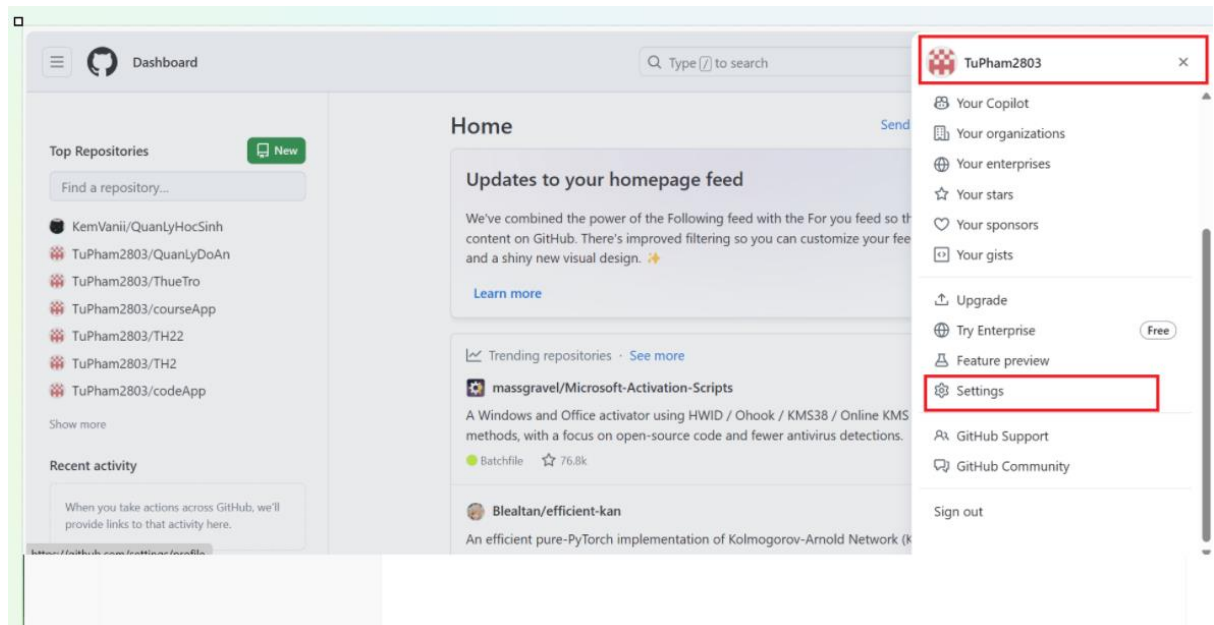
Hình II.3. Tạo collection

Bước 4: Tạo xác thực token trên Github

- Đăng nhập trang Github: [GitHub](https://github.com)

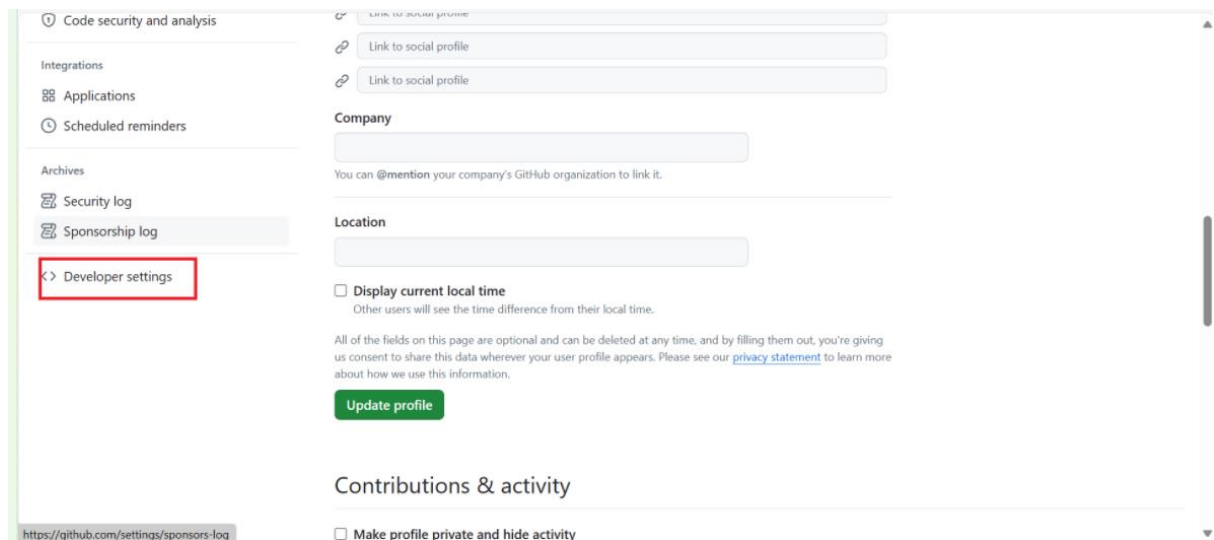


Hình II.4. Đăng nhập Github

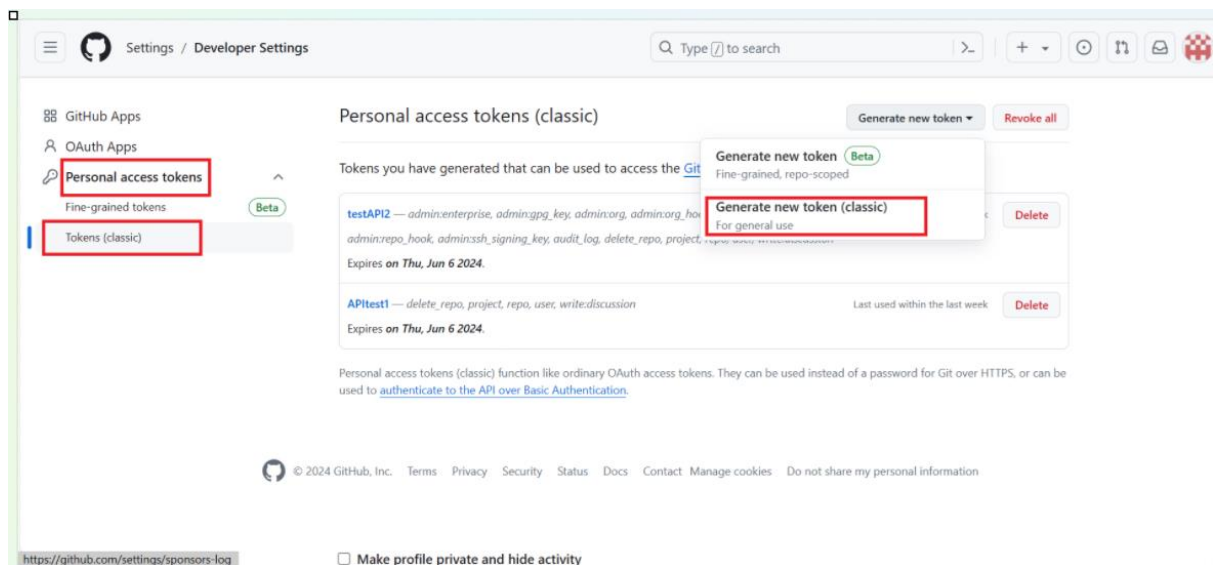


Hình II.5. Tạo Token

- Vào Setting -> Developer Settings -> Personal access tokens -> General new token(classic)

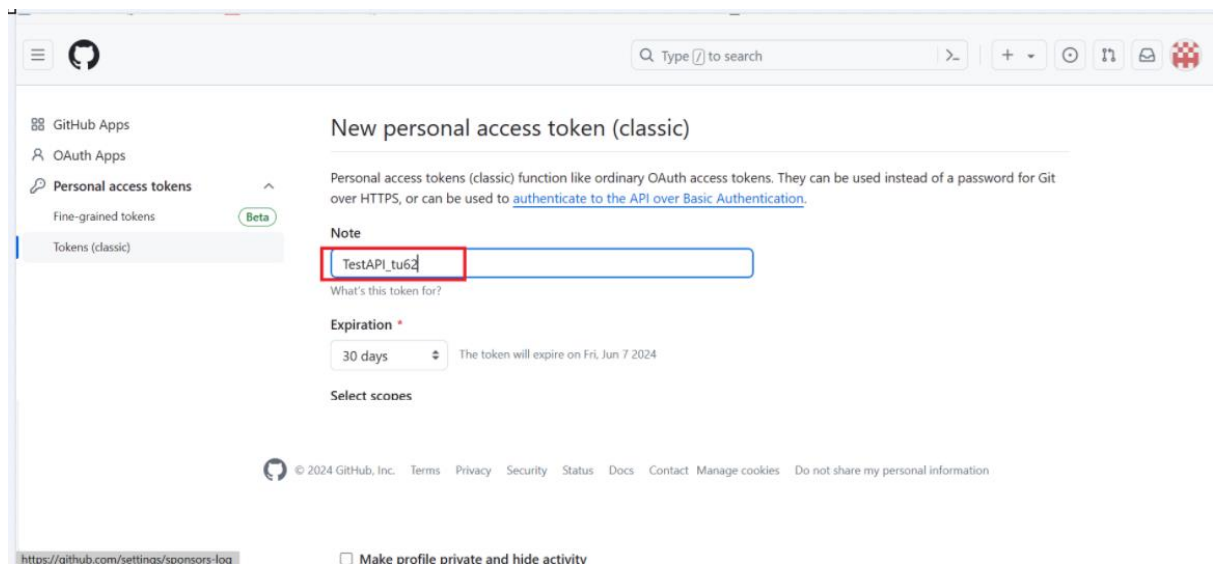


Hình II.6. Tạo Token



Hình II.7. Tạo Token

- Đặt tên và cấp quyền cho token



Hình II.8. Đặt tên cho token

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input checked="" type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input checked="" type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects
<input checked="" type="checkbox"/> manage_runners:org	Manage org runners and runner groups
<input checked="" type="checkbox"/> admin:public_key	Full control of user public keys
<input checked="" type="checkbox"/> write:public_key	Write user public keys
<input checked="" type="checkbox"/> read:public_key	Read user public keys

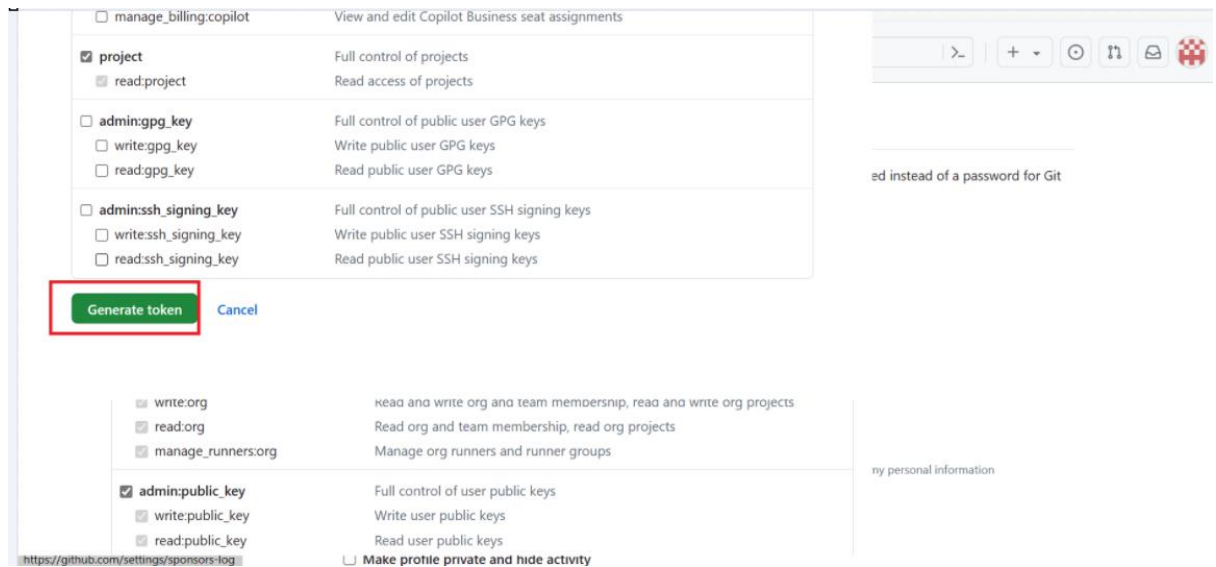
Hình II.9. Cấp quyền cho token

<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input checked="" type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects
<input checked="" type="checkbox"/> manage_runners:org	Manage org runners and runner groups
<input checked="" type="checkbox"/> admin:public_key	Full control of user public keys
<input checked="" type="checkbox"/> write:public_key	Write user public keys
<input checked="" type="checkbox"/> read:public_key	Read user public keys
<input checked="" type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input checked="" type="checkbox"/> write:repo_hook	Write repository hooks
<input checked="" type="checkbox"/> read:repo_hook	Read repository hooks
<input type="checkbox"/> admin:org_hook	Full control of organization hooks
<input type="checkbox"/> gist	Create gists
<input type="checkbox"/> notifications	Access notifications
<input checked="" type="checkbox"/> user	Update ALL user data
<input checked="" type="checkbox"/> read:user	Read ALL user profile data
<input checked="" type="checkbox"/> user:email	Access user email addresses (read-only)
<input checked="" type="checkbox"/> user:follow	Follow and unfollow users

Hình II.10. Cấp quyền cho token

<input type="checkbox"/> manage_billing:enterprise	Read and write enterprise billing data
<input type="checkbox"/> read:enterprise	Read enterprise profile data
<input type="checkbox"/> audit_log	Full control of audit log
<input type="checkbox"/> read:audit_log	Read access of audit log
<input type="checkbox"/> codespace	Full control of codespaces
<input type="checkbox"/> codespace:secrets	Ability to create, read, update, and delete codespace secrets
<input type="checkbox"/> copilot	Full control of GitHub Copilot settings and seat assignments
<input type="checkbox"/> manage_billing:copilot	View and edit Copilot Business seat assignments
<input checked="" type="checkbox"/> project	Full control of projects
<input checked="" type="checkbox"/> read:project	Read access of projects
<input type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input type="checkbox"/> write:gpg_key	Write public user GPG keys
<input type="checkbox"/> read:gpg_key	Read public user GPG keys
<input type="checkbox"/> admin:ssh_signing_key	Full control of public user SSH signing keys
<input type="checkbox"/> write:ssh_signing_key	Write public user SSH signing keys
<input type="checkbox"/> read:ssh_signing_key	Read public user SSH signing keys

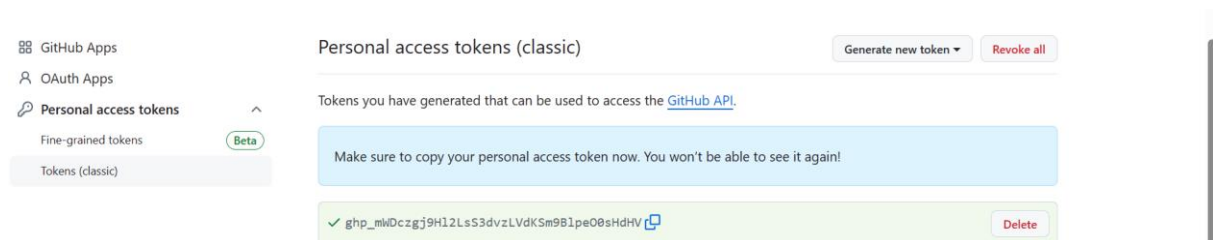
Hình II.11. Cấp quyền cho token



Hình II.12. Tạo token

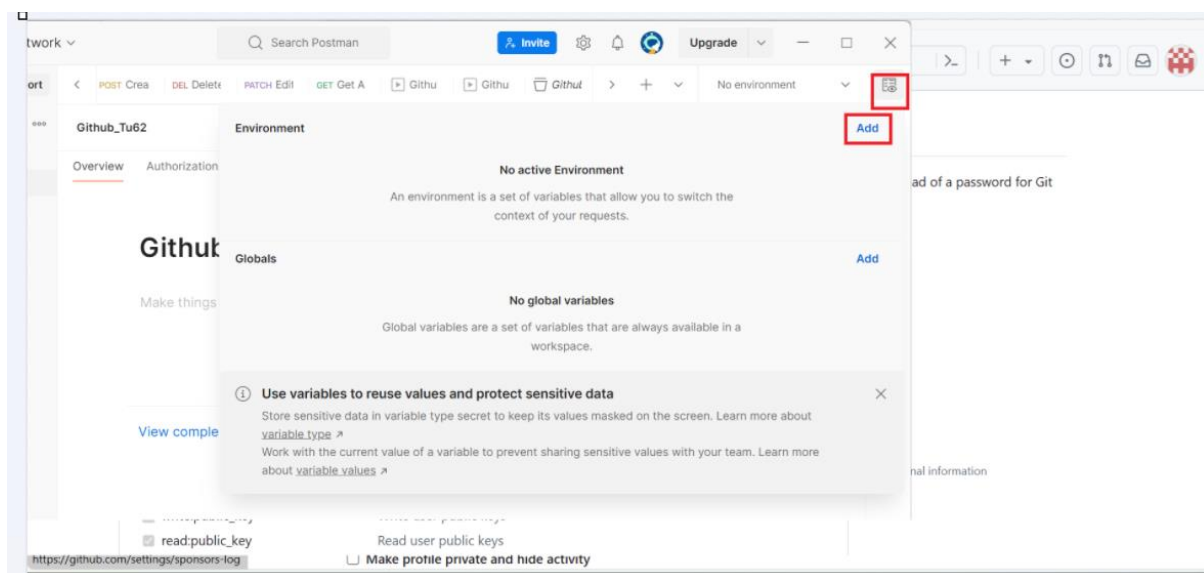
Tạo token thành công:

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

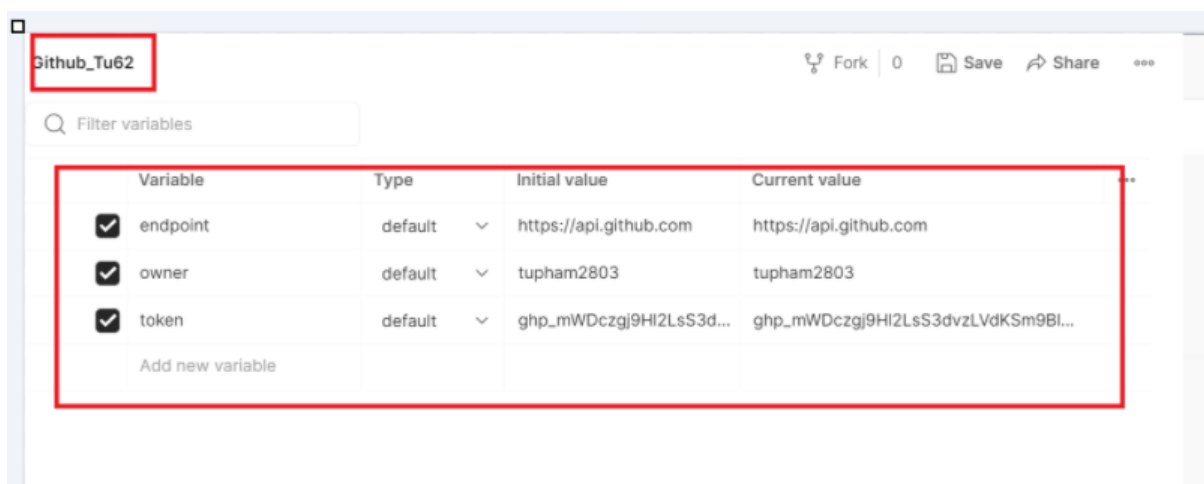


Hình II.13. Tạo token thành công

Bước 5: Tạo các biến môi trường cần thiết



Hình II.14. Tạo biến môi trường cần thiết



Hình II.15. Biến môi trường cần thiết

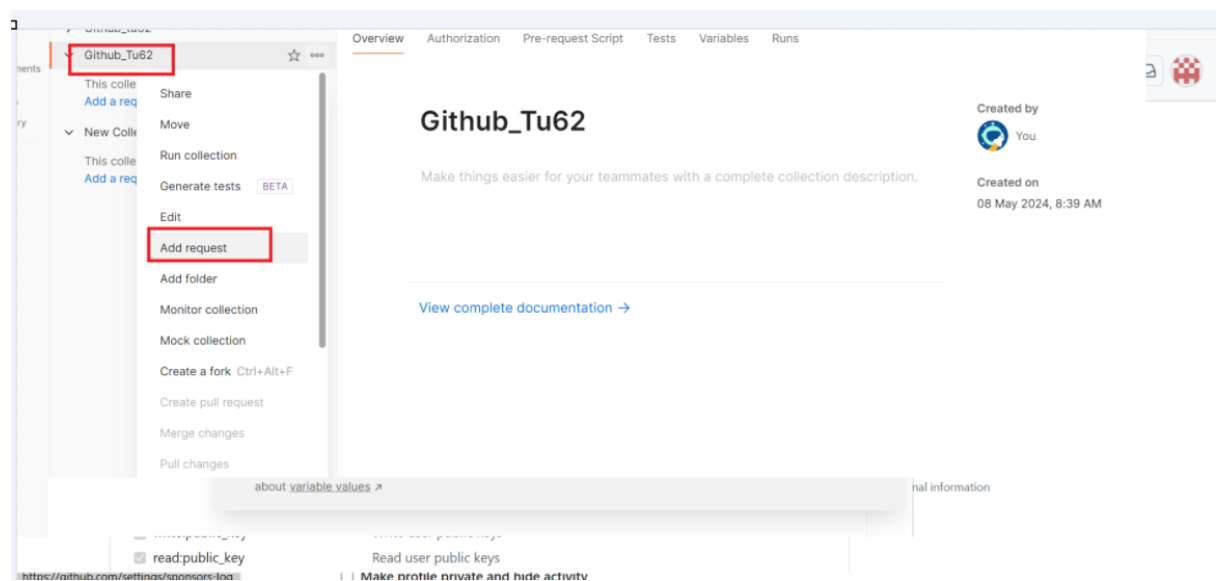
2. Triển khai viết các TestCase

2.1. Chức năng 1: Kiểm thử chức năng lấy tất cả Repos trong GitHub của user tạo

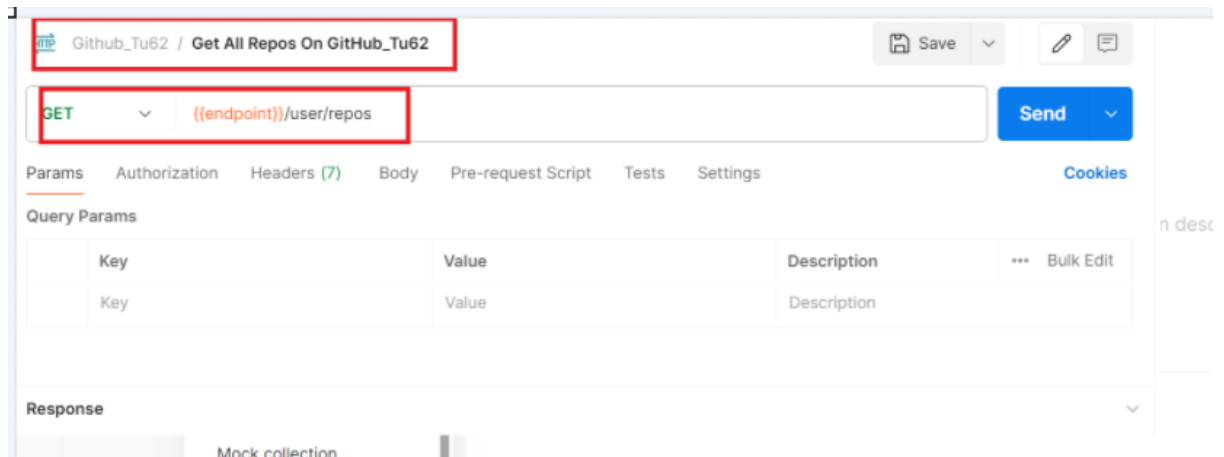
- Mô tả chức năng: Chức năng này cho phép lấy tất cả các repo mà một người dùng đã tạo trên GitHub.

Bước 1: Bước 1: Lấy API: <https://api.github.com/user/repos>

Bước 2: Bước 2: Add Request mới -> Đặt tên cho Request, chỉnh phương thức và thêm API vào

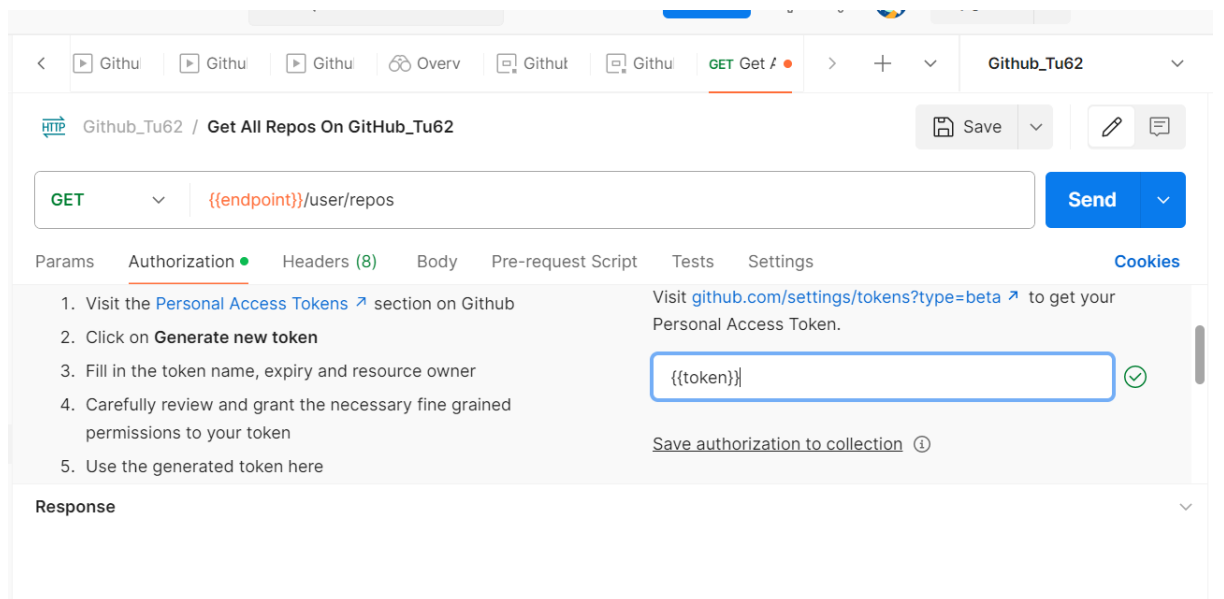


Hình II.16. Add request



Hình II.17. Nhập Url API và Phương thức

Bước 3: Cấp token đã tạo được



Hình II.18. Cấp token đã tạo

Bước 4: Viết các testcase

TC1.1-Tu62-Response status code is 200

: Kiểm tra Response status code trả về =200

```
pm.test("TC1.1-Tu62-Response status code is 200", function () {
  pm.response.to.have.status(200);
});
```

Params Authorization ● Headers (8) Body Pre-request Script Tests ● Settings

```
1 pm.test("TC1.1-Tu62-Response status code is 200", function () {
2   | pm.response.to.have.status(200);
3   });
```

TC1.2-TU62-Response type is JSON

Kiểm tra dữ liệu trả về đúng định dạng JSON

// Test to check if the response type is JSON

```
pm.test("TC1.2-TU62-Response type is JSON", function () {
  pm.expect(pm.response).to.have.header('Content-Type',
  'application/json; charset=utf-8');
});
```

[HTTP](#) Github_Tu62 / Get All Repos On GitHub_Tu62

GET {{endpoint}}/user/repos

Params Authorization ● Headers (8) Body Pre-request Script Tests ● Settings

```
3   });
4   // Test to check if the response type is JSON
5   pm.test("TC1.2-TU62-Response type is JSON", function () {
6     pm.expect(pm.response).to.have.header('Content-Type', 'application/
7       json; charset=utf-8');
7   });
```

TC1.3-Tu62-The number of returned repos is equal to 17

Kiểm tra số lượng Repo trả về có bằng 17

// Test to check if the number of returned repos is equal to 17

```
pm.test("TC1.3-Tu62-The number of returned repos is equal to 17",
function () {
  const responseData = pm.response.json();
  pm.expect(responseData.length).to.equal(17);
});
```



```
GET {{endpoint}}/user/repos

Params  Authorization ● Headers (8) Body Pre-request Script Tests ● Settings

7  });
8  // Test to check if the number of returned repos is equal to 17
9  pm.test("TC1.3-Tu62-The number of returned repos is equal to 17",
    function () {
10     const responseData = pm.response.json();
11     pm.expect(responseData.length).to.equal(17);
12 });
```

Bước 5: Kiểm tra chạy thử các testcase

Body Cookies Headers (28) Test Results (3/3) [Get an environment variable](#) 200 OK 870 ms 84.79 KB Save as example

All Passed Skipped Failed

- PASS** TC1.1-Tu62-Response status code is 200
- PASS** TC1.2-TU62-Response type is JSON
- PASS** TC1.3-Tu62-The number of returned repos is equal to 17

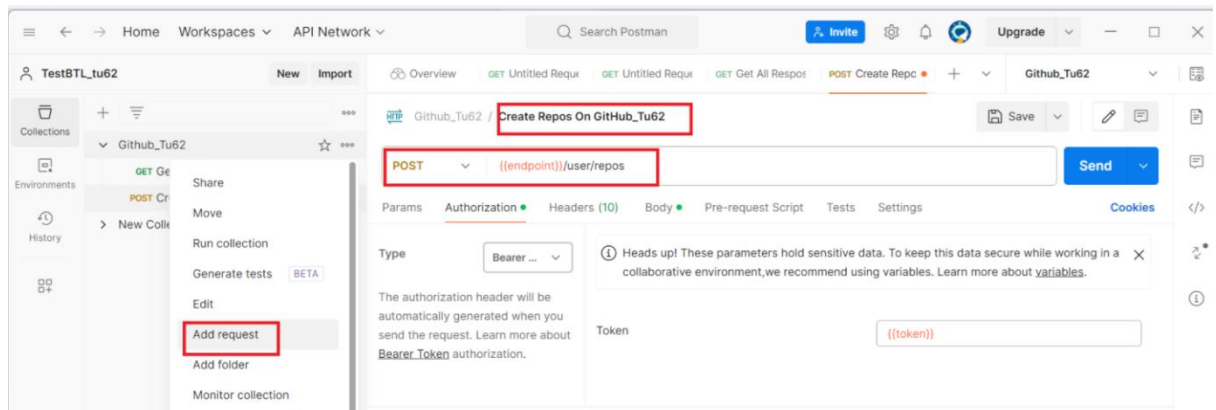
2.2. Chức năng 2: Kiểm thử chức năng tạo một Repos mới

Mô tả chức năng: Chức năng tạo repo mới trên GitHub cho phép người dùng nhập tên, mô tả, chọn quyền riêng tư, thêm README, .gitignore, hoặc license, sau đó nhấn "Create repository" để tạo repo mới.

Bước 1: Lấy API :

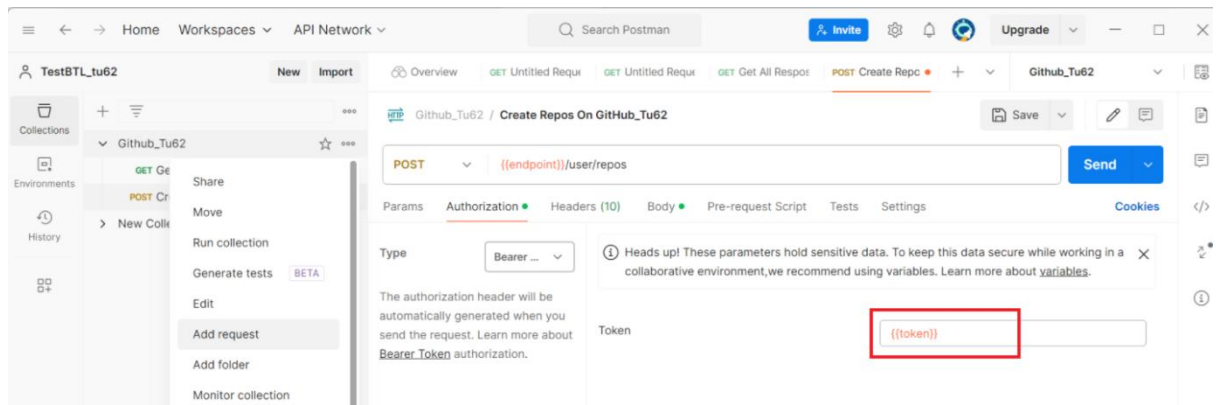
<https://api.github.com/repos/{owner}/{repo}>

Bước 2: Add Request mới -> Đặt tên cho Request, chỉnh phương thức và thêm API vào



Hình II.19. Add Request

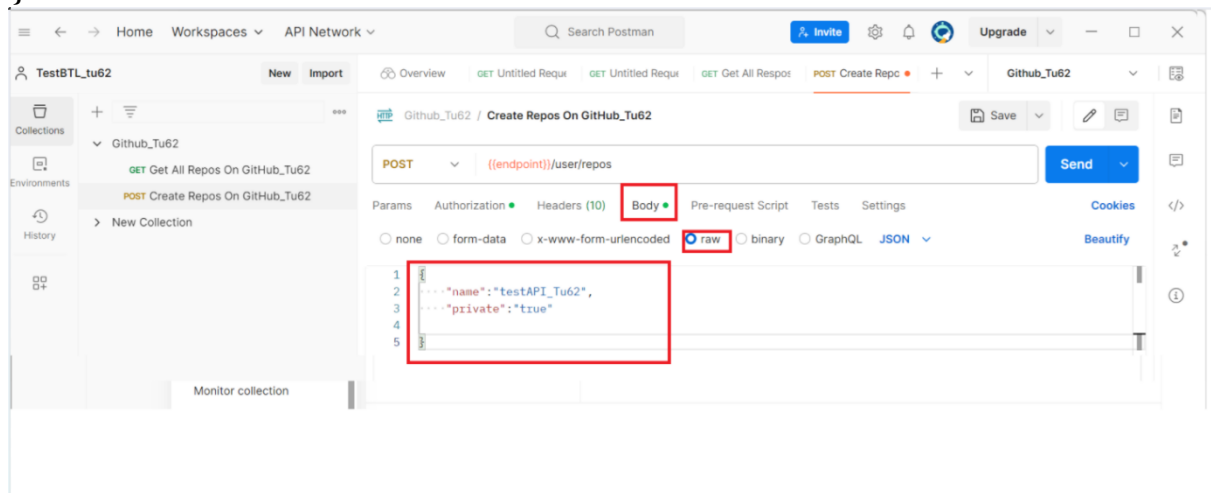
Bước 3: Bước 3: Cấp token đã tạo được



Hình II.20. Cấp token đã được tạo trước

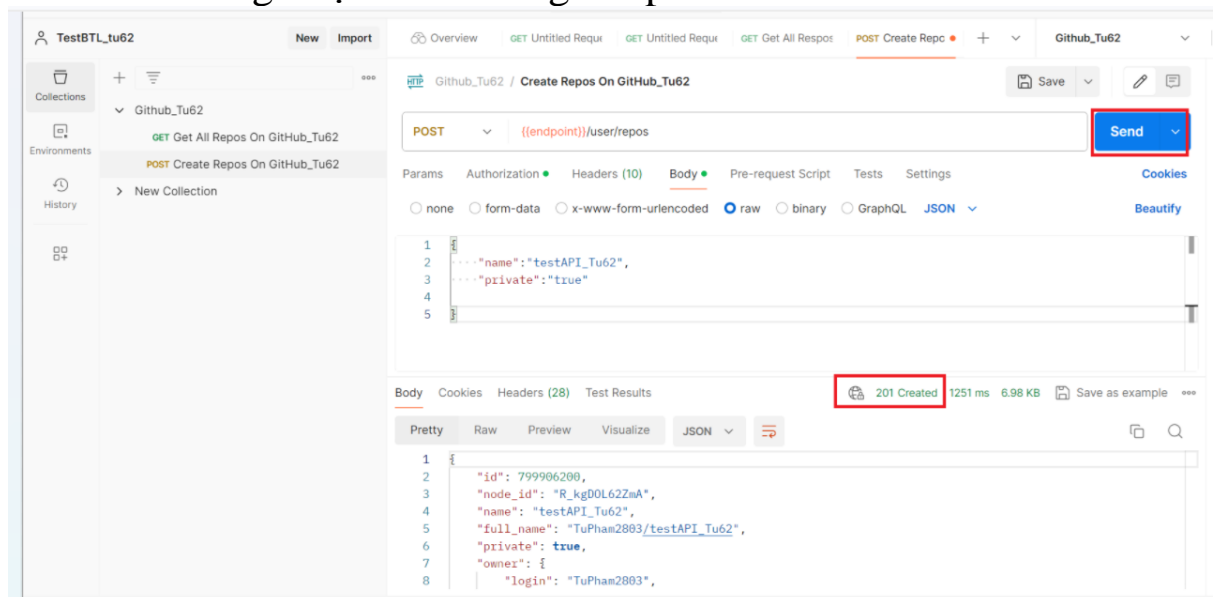
Bước 4: Bước 4: Điền các thông tin để tạo repos

```
{  
  "name": "testAPI_Tu62",  
  "private": true  
}
```



Hình II.21. Điền thông tin cho phương thức Post

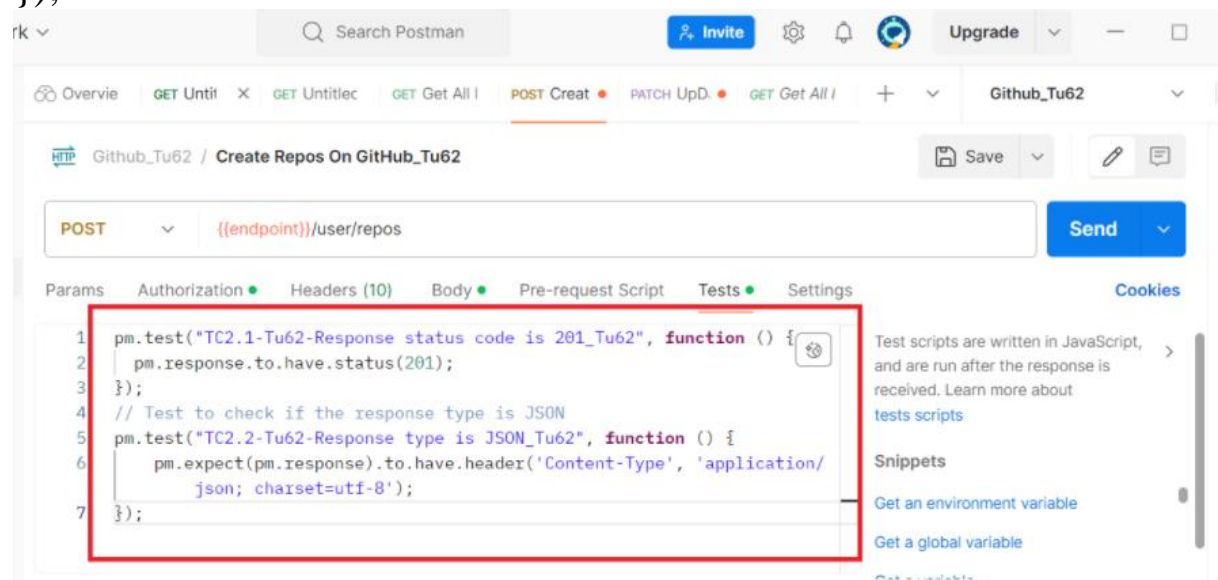
Bước 5: Send gửi tạo thành công 1 repos



Hình II.22. Send thành công

Bước 6: Viết các testcase kiểm tra chức năng create Repos

```
pm.test("TC2.1-Tu62-Response status code is 201_Tu62", function () {
  pm.response.to.have.status(201);
});
// Test to check if the response type is JSON
pm.test("TC2.2-Tu62-Response type is JSON_Tu62", function () {
  pm.expect(pm.response).to.have.header('Content-Type',
  'application/json; charset=utf-8');
});
```



Hình II.23. Viết testcase

Bước 6: Send chạy thử testcase

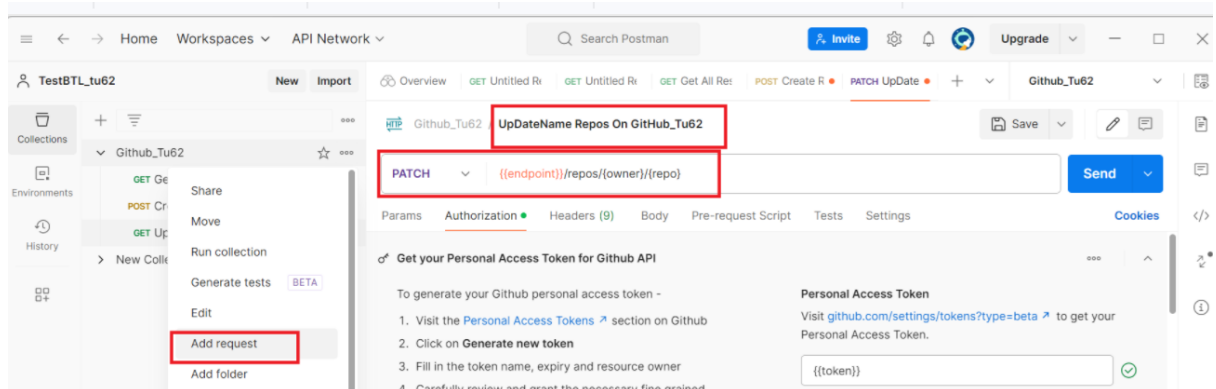
2.3. Chức năng 3: Kiểm thử chức năng sửa tên Repos

Mô tả Chức năng sửa tên repo trên GitHub: cho phép người dùng vào tab "Settings" của repo, thay đổi tên trong mục "Repository name", và nhấn "Rename" để lưu tên mới.

Bước 1: Lấy url API

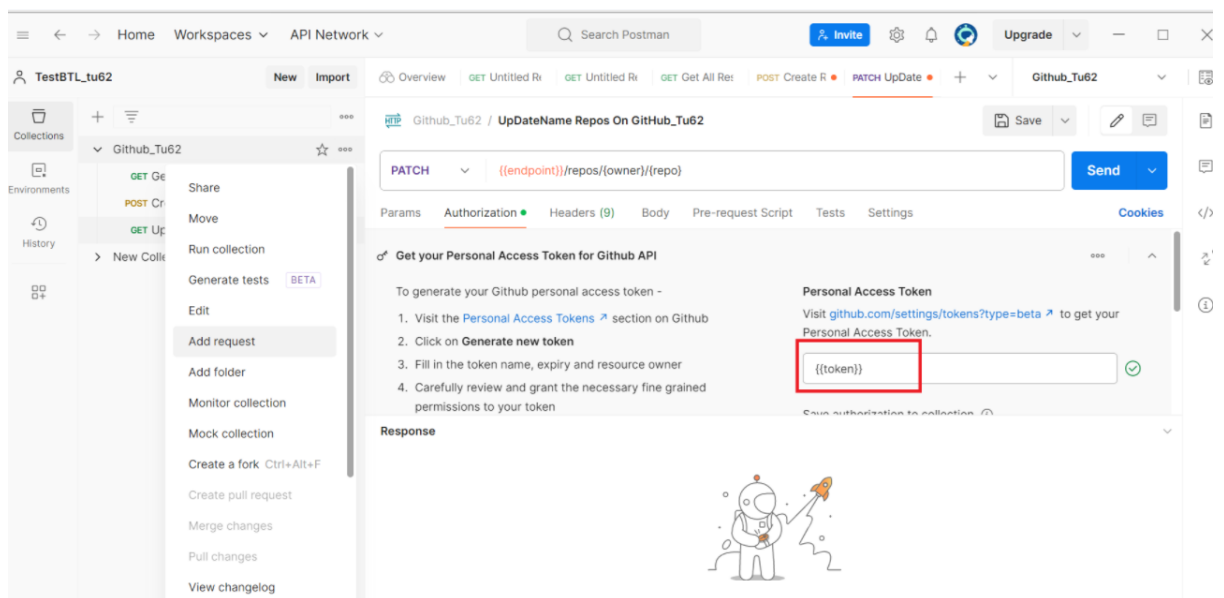
`https://api.github.com/repos/{owner}/{repo}`

Bước 2: Add Request mới -> Đặt tên cho Request, chỉnh phương thức và thêm API vào



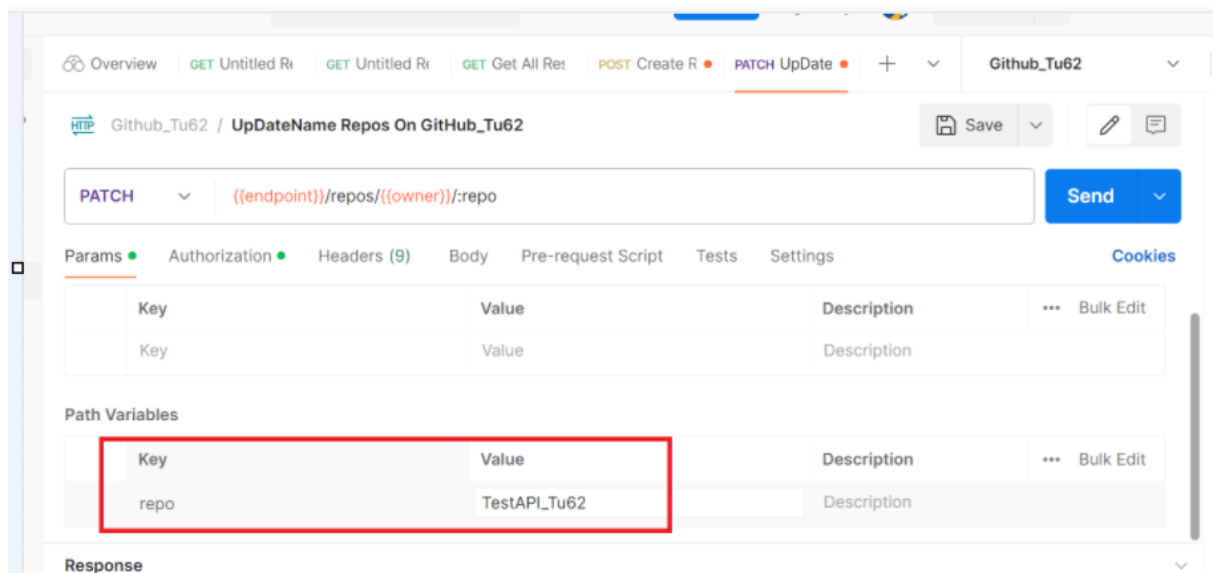
Hình II.24. Add request phương thức Patch

Bước 3: Cấp token đã tạo trước đó



Hình II.25. Cấp token đã tạo

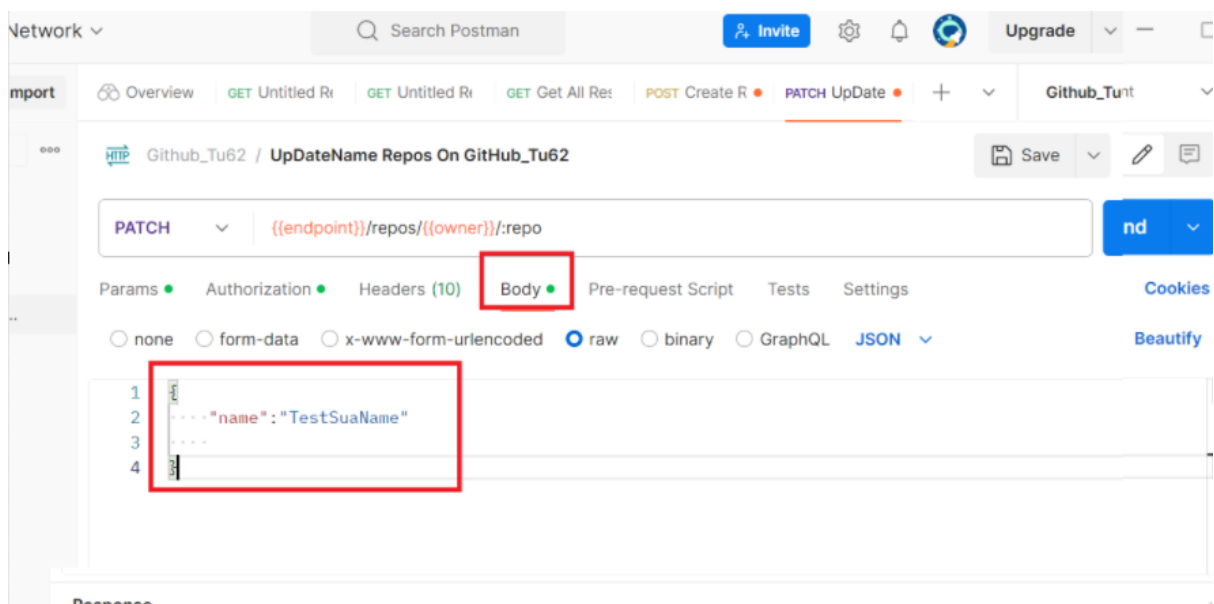
Bước 4: Nhập tên của Repo muốn sửa



Hình II.26. Nhập Repo muốn sửa tên

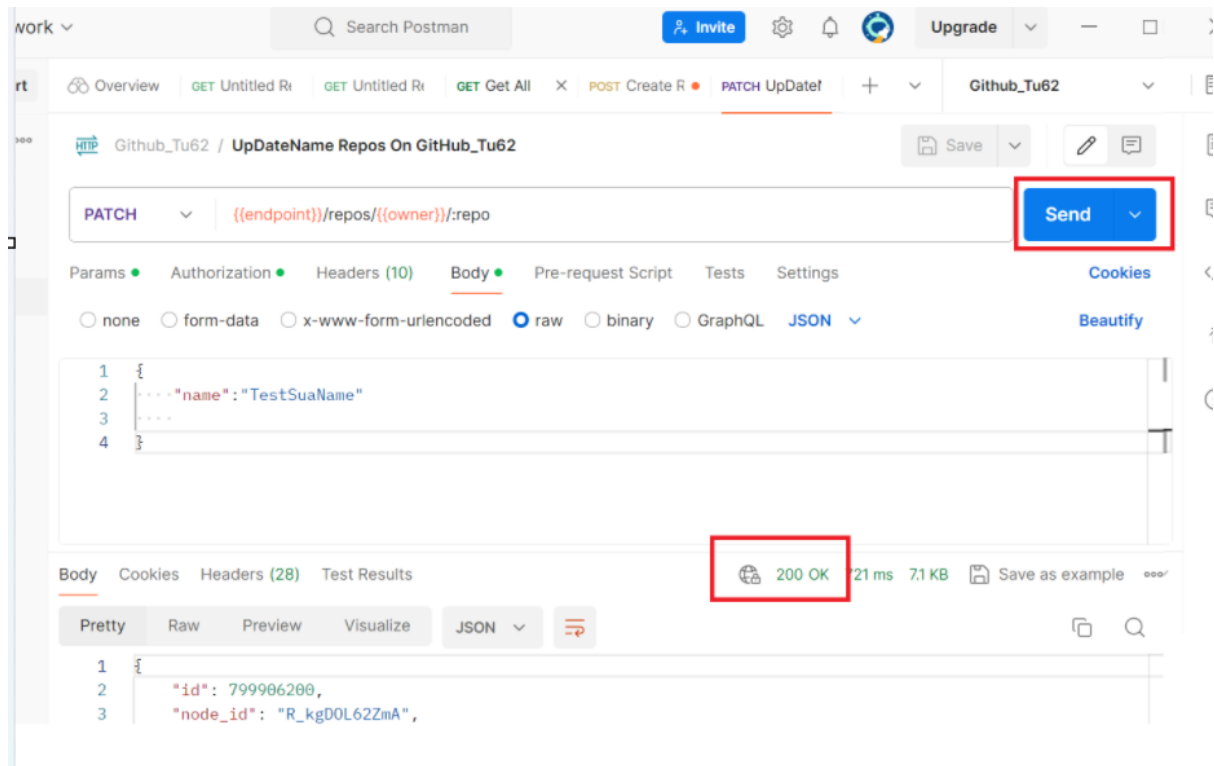
Bước 5: Cập nhật lại tên trong body

```
{  
  "name": "TestSuaName"  
}
```



Hình II.27. Update lại tên mới cho Repo

Bước 6: Send và kiểm tra kết quả đã sửa tên thành công



Bước 7: Viết testcase API kiểm tra

TC3.1-Tu62-Response status code is 200: Kiểm tra status code là 200

```
pm.test("TC3.1-Tu62-Response status code is 200", function () {  
  pm.response.to.have.status(200);  
});
```

TC3.2-Tu62-Response type is JSON : Kiểm tra dữ liệu trả về có phải là JSON

// Test to check if the response type is JSON

```
pm.test("TC3.2-Tu62-Response type is JSON", function () {  
  pm.expect(pm.response).to.have.header('Content-Type',  
'application/json; charset=utf-8');
```

```
});
```

TC3.3-Tu62-Name field has been changed to 'TestSuaName': Kiểm tra RePo đã được đổi tên

```
// Test to check if the 'name' field has been changed to 'TestSuaName'
```

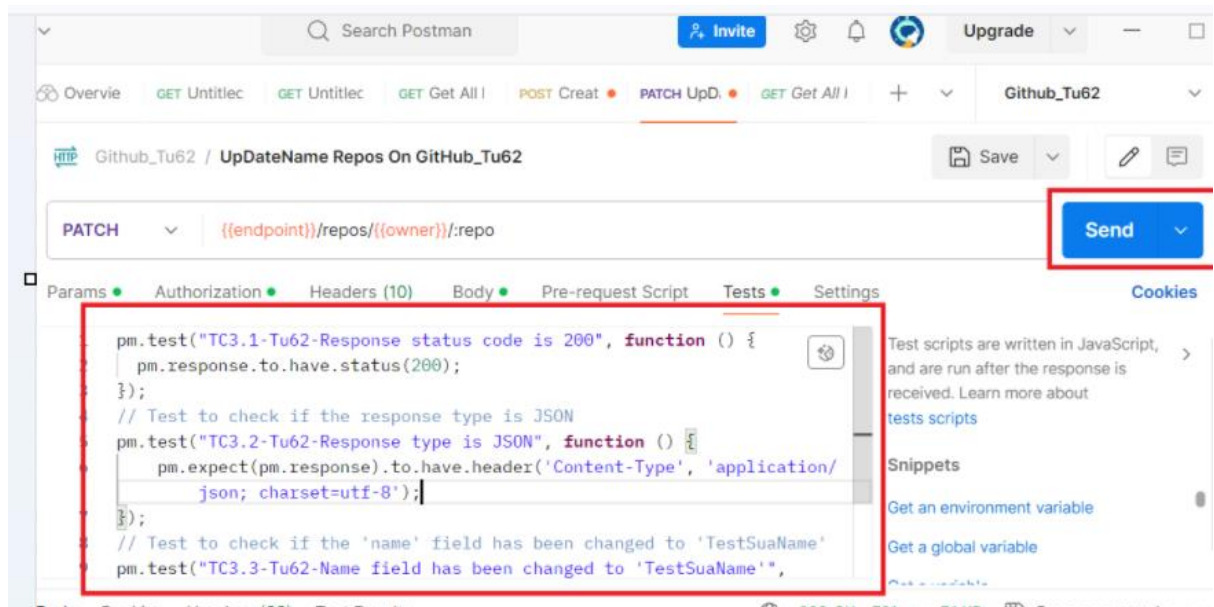
```
pm.test("TC3.3-Tu62-Name field has been changed to
```

```
'TestSuaName'", function () {
```

```
    var responseData = pm.response.json();
```

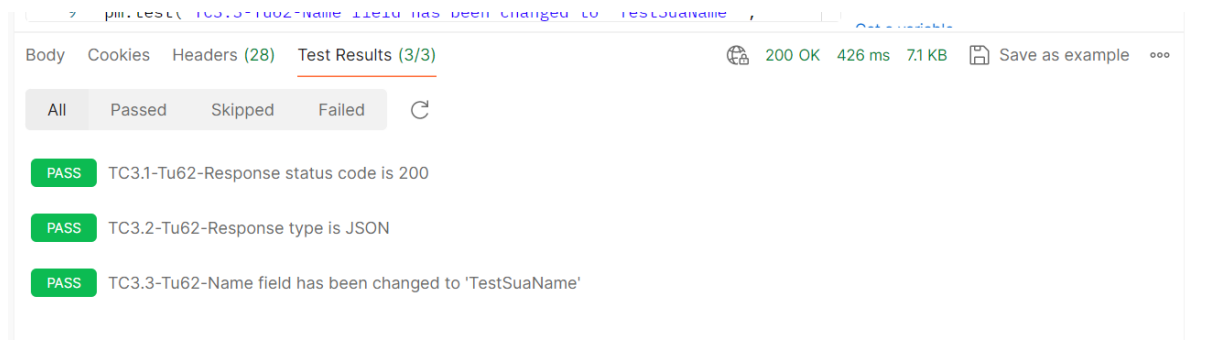
```
    pm.expect(responseData.name).to.eql('TestSuaName');
```

```
});
```



Hình II.28. TestCase API kiểm tra sửa tên repo

Bước 8: Chạy kiểm tra các TestCase



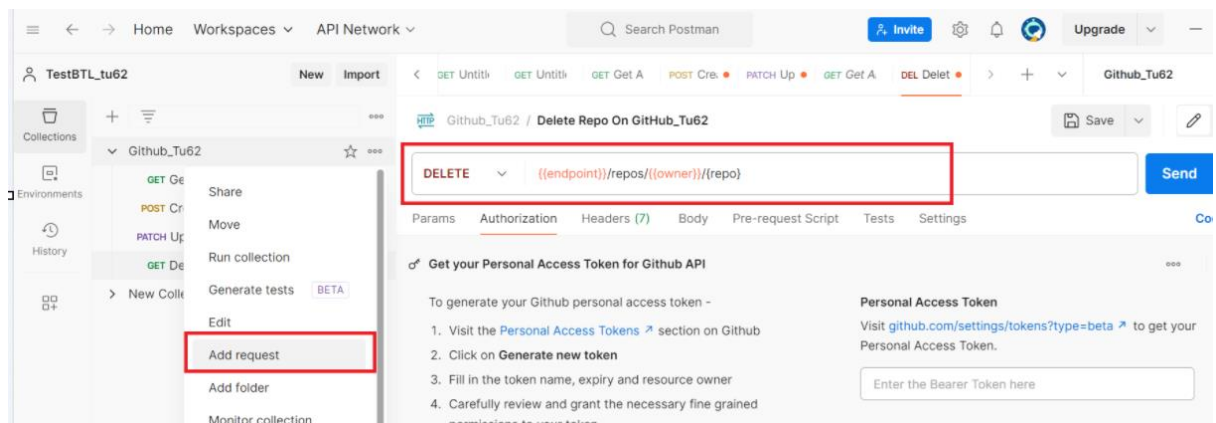
Hình II.29. Kết quả testcase API

2.4. Chức năng 4: Kiểm thử chức năng xóa một Repos

Mô tả chức năng xóa một repository (repo) trên GitHub cho phép người dùng vào tab "Settings" của repo, cuộn xuống phần "Danger Zone", nhấn "Delete this repository", xác nhận tên repo và nhấn "I understand the consequences, delete this repository" để xóa vĩnh viễn.

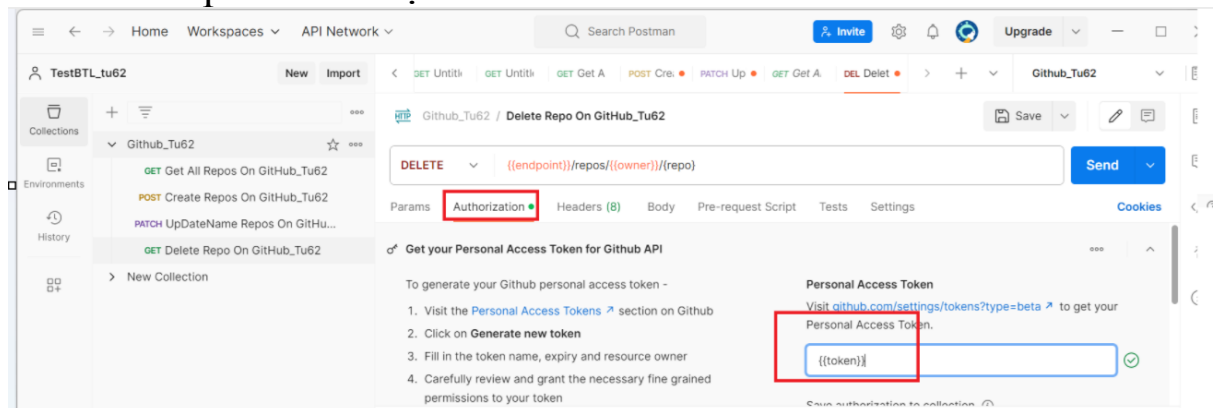
Bước 1: Lấy API:
`https://api.github.com/repos/{owner}/{repo}`

Bước 2: Add Request mới -> Đặt tên cho Request, chỉnh phương thức và thêm API vào



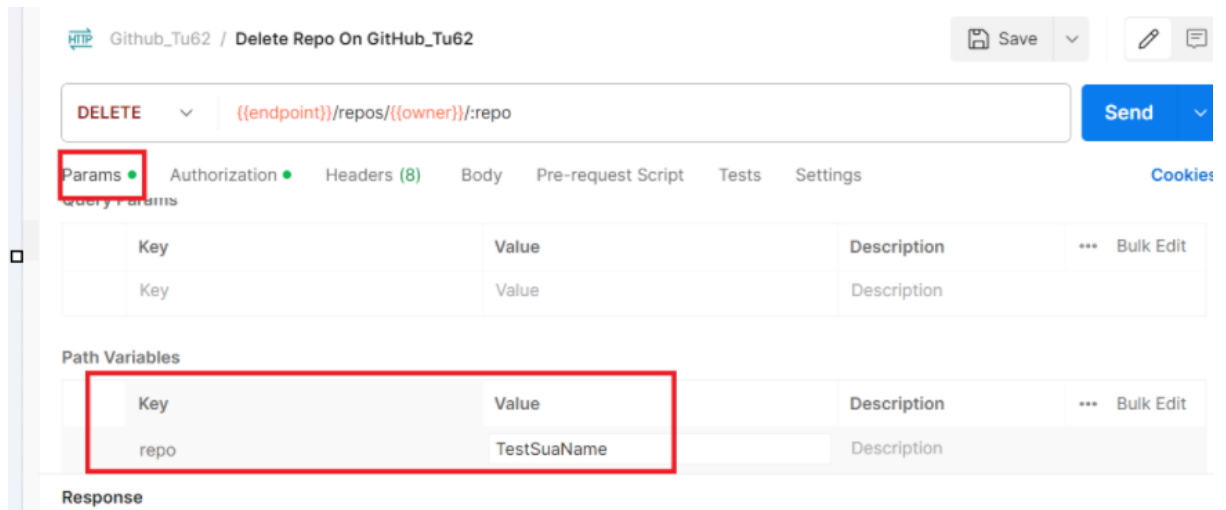
Hình II.30. Add request phương thức Delete

Bước 3: Cấp token đã tạo



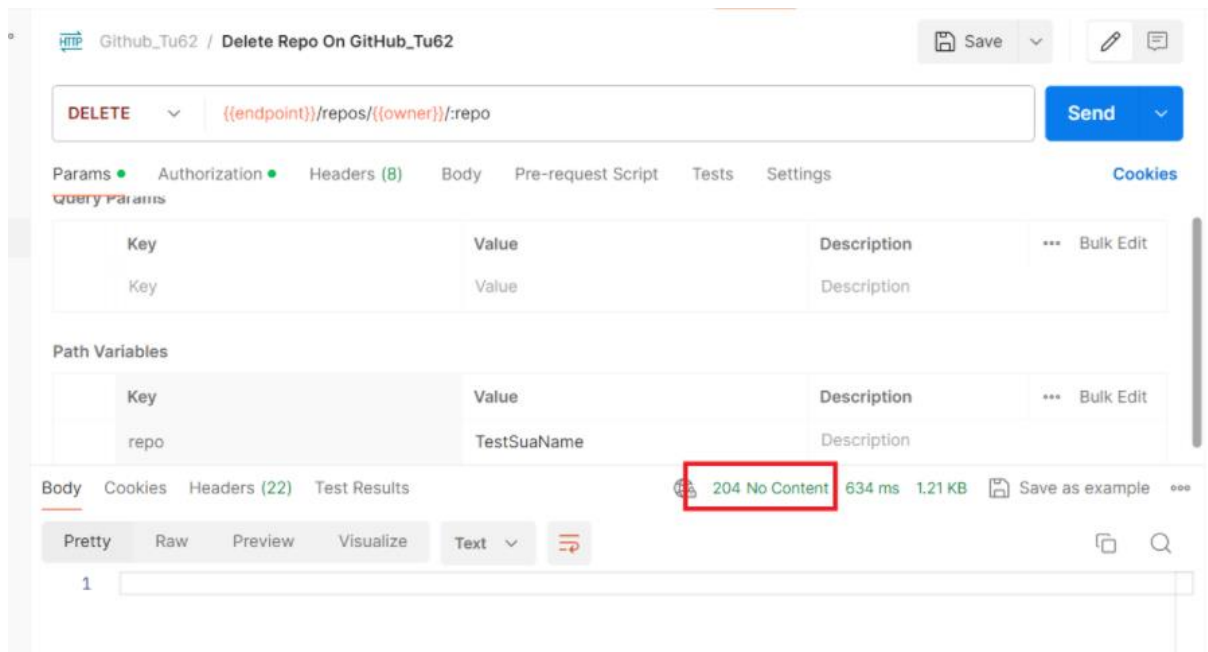
Hình II.31. Cấp token đã tạo trước đó

Bước 3: Chọn tên repo muốn Delete trong Params



Hình II.32. Chọn Repo muốn delete

Bước 4: Send gửi và kiểm tra kết quả đã xóa thành công



Hình II.33. Send kiểm tra kết quả delete thành công

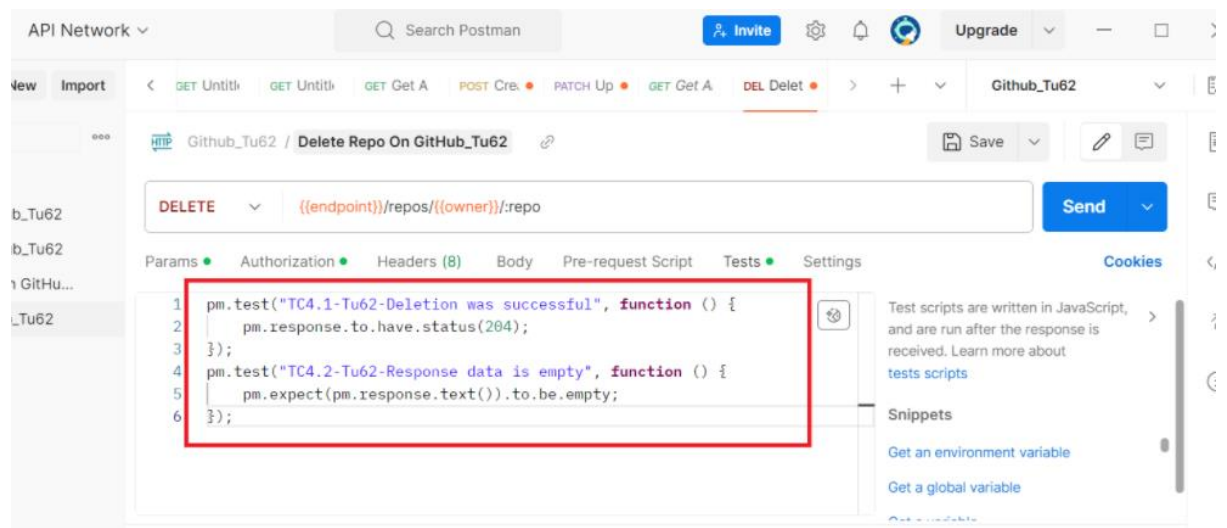
Bước 5: Viết các TestCase chức năng Delete Repo

TC4.1-Tu62-Deletion was successful : Kiểm tra status code 204

```
pm.test("TC4.1-Tu62-Deletion was successful", function () {  
    pm.response.to.have.status(204);  
});
```

TC4.2-Tu62-Response data is empty: Kiểm tra RePo rỗng

```
pm.test("TC4.2-Tu62-Response data is empty", function () {  
    pm.expect(pm.response.text()).to.be.empty;  
});
```



Hình II.34. Testcase API chức năng Delete

Bước 6: Send và gửi xem kết quả của testcase

Kết quả chạy các testcase API

Github_Tu62 - Run results

Run Again
Automate Run
+ New Run
Export Results

Ran today at 08:29:28 · [View all runs](#)

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	Github_Tu62	1	4s 131ms	10	856 ms

All Tests
Passed (10)
Failed (0)
Skipped (0)
[View Summary](#)

Iteration 1

1

GET
Get All Repos On GitHub_Tu62
https://api.github.com/user/repos
200 OK
1030 ms
91.775 KB

PASS
TC1.1-Tu62-Response status code is 200
PASS
TC1.2-TU62-Response type is JSON
PASS
TC1.3-Tu62-The number of returned repos is equal to 18

POST
Create Repos On GitHub_Tu62
https://api.github.com/user/repos
201 Created
1122 ms
7.141 KB

PASS
TC2.1-Tu62-Response status code is 201_Tu62
PASS
TC2.2-Tu62-Response type is JSON_Tu62

Hình II.35. Kết quả chạy các testcase API

Github_Tu62 - Run results

Run Again
Automate Run
+ New Run
Export Results

Ran today at 08:29:28 · [View all runs](#)

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	Github_Tu62	1	4s 131ms	10	856 ms

All Tests
Passed (10)
Failed (0)
Skipped (0)
[View Summary](#)

https://api.github.com/user/repos
201 Created
1122 ms
7.141 KB

PASS
TC2.1-Tu62-Response status code is 201_Tu62
PASS
TC2.2-Tu62-Response type is JSON_Tu62

PATCH
UpDateName Repos On GitHub_Tu62
https://api.github.com/repos/tupham2803/TestAPI_Tu62
200 OK
653 ms
7.273 KB

PASS
TC3.1-Tu62-Response status code is 200
PASS
TC3.2-Tu62-Response type is JSON
PASS
TC3.3-Tu62-Name field has been changed to 'TestSuaName'


DELETE
Delete Repo On GitHub_Tu62
https://api.github.com/repos/tupham2803/TestSuaName
204 No Content
619 ms
1.235 KB

PASS
TC4.1-Tu62-Deletion was successful
PASS
TC4.2-Tu62-Response data is empty

Hình II.36. Kết quả chạy các testcase API

Github_Tu62 - Run results

Ru

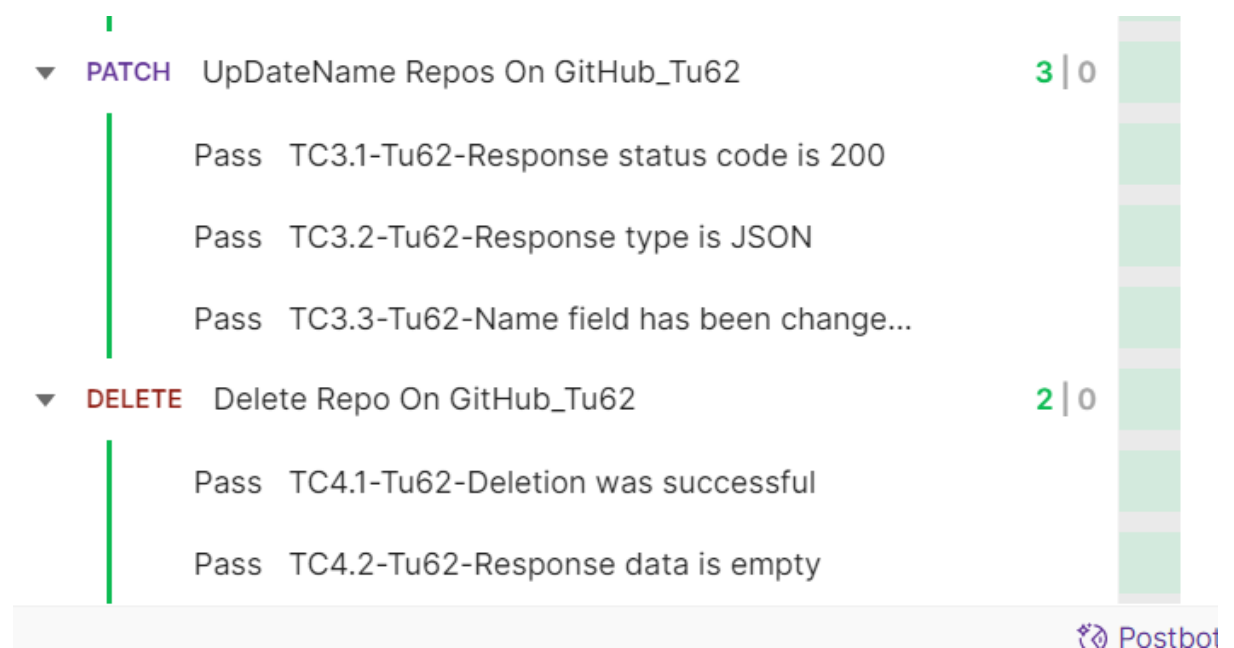
 Ran today at 08:29:28 · [View all runs](#)

Source	Environment	Iterations	Duration
Runner	Github_Tu62	1	4s 131ms

RUN SUMMARY

▼	GET	Get All Repos On GitHub_Tu62	3 0
		Pass TC1.1-Tu62-Response status code is 200	
		Pass TC1.2-TU62-Response type is JSON	
		Pass TC1.3-Tu62-The number of returned repo...	
▼	POST	Create Repos On GitHub_Tu62	2 0
		Pass TC2.1-Tu62-Response status code is 201_...	
		Pass TC2.2-Tu62-Response type is JSON_Tu62	

Hình II.37. Kết quả chạy các testcase API



Hình II.38. Kết quả chạy các testcase API