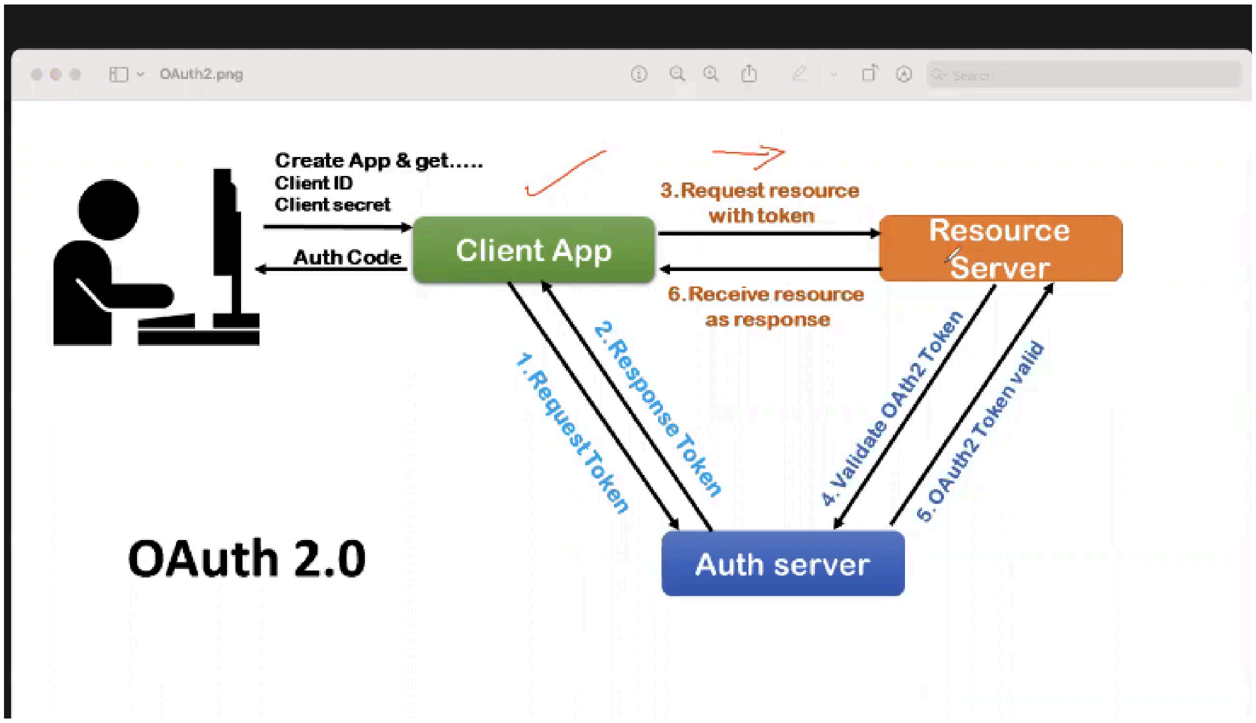
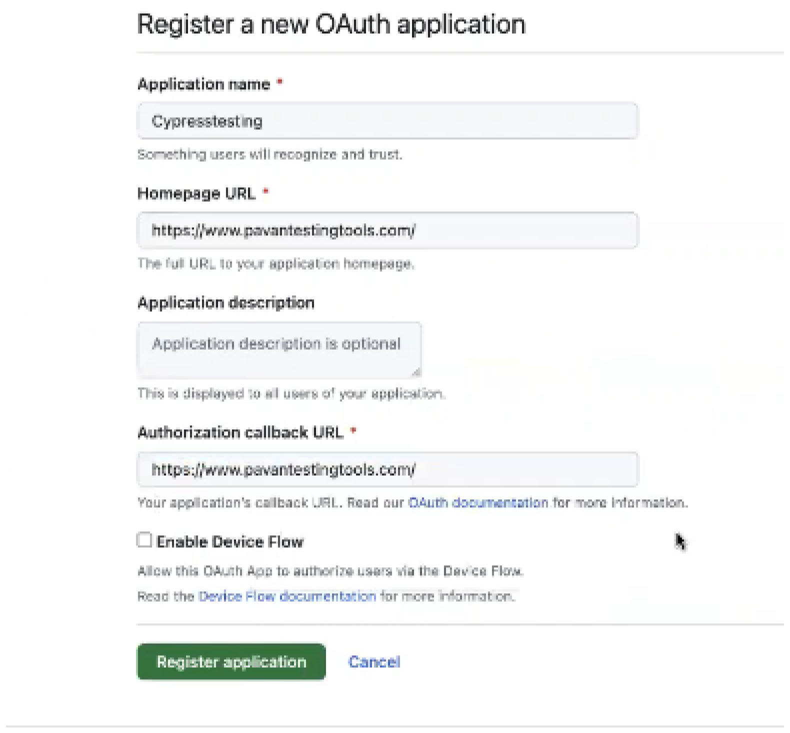


Go to github.
Settings.
Developer settings.
Oauth apps.
Create new app.
Give any name url etc.
Click register.



Register a new OAuth application

Application name *

Cypresstesting

Something users will recognize and trust.

Homepage URL *

https://www.pavantestingtools.com/

The full URL to your application homepage.

Application description

Application description is optional

This is displayed to all users of your application.

Authorization callback URL *

https://www.pavantestingtools.com/

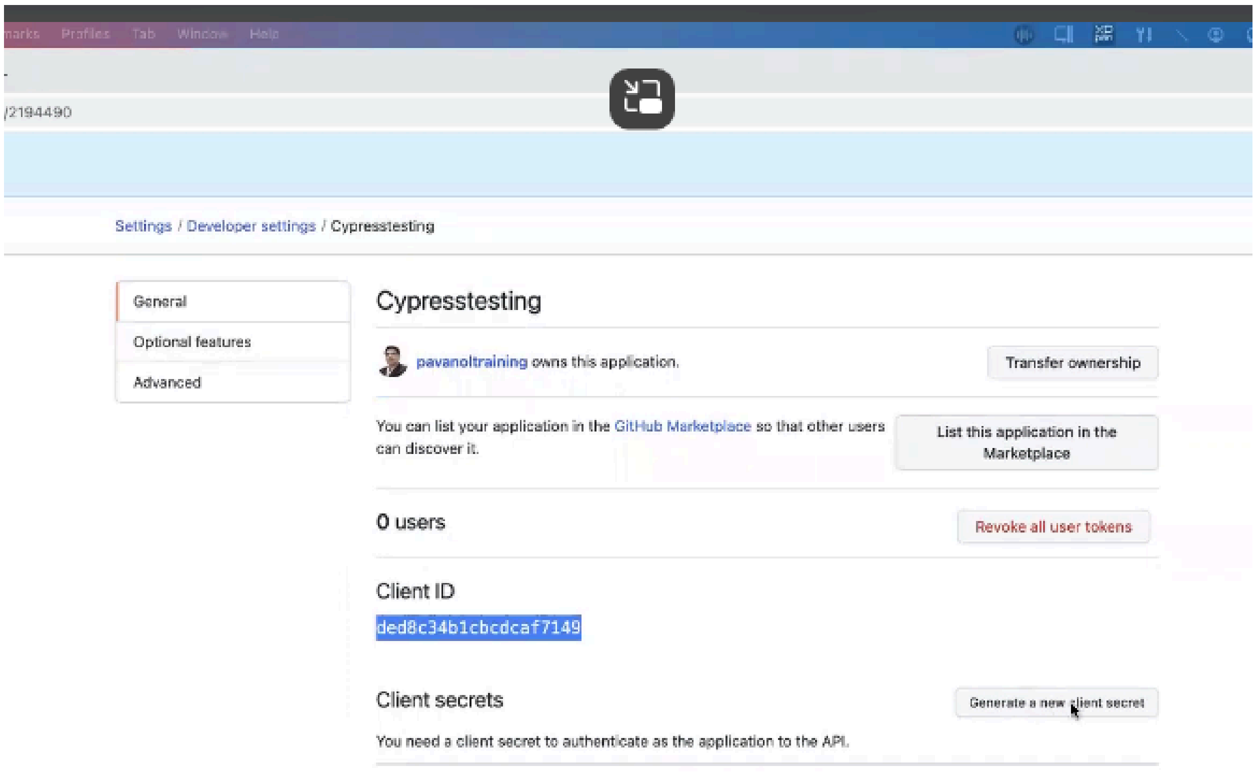
Your application's callback URL. Read our OAuth documentation for more information.

☐ Enable Device Flow

Allow this OAuth App to authorize users via the Device Flow.

Read the Device Flow documentation for more information.

Client id generated. Click generate client secret.



/2194490
Settings / Developer settings / Cypresstesting
General
Optional features
Advanced

Cypresstesting

pavanoltraining owns this application.

You can list your application in the GitHub Marketplace so that other users can discover it.

0 users

Client ID

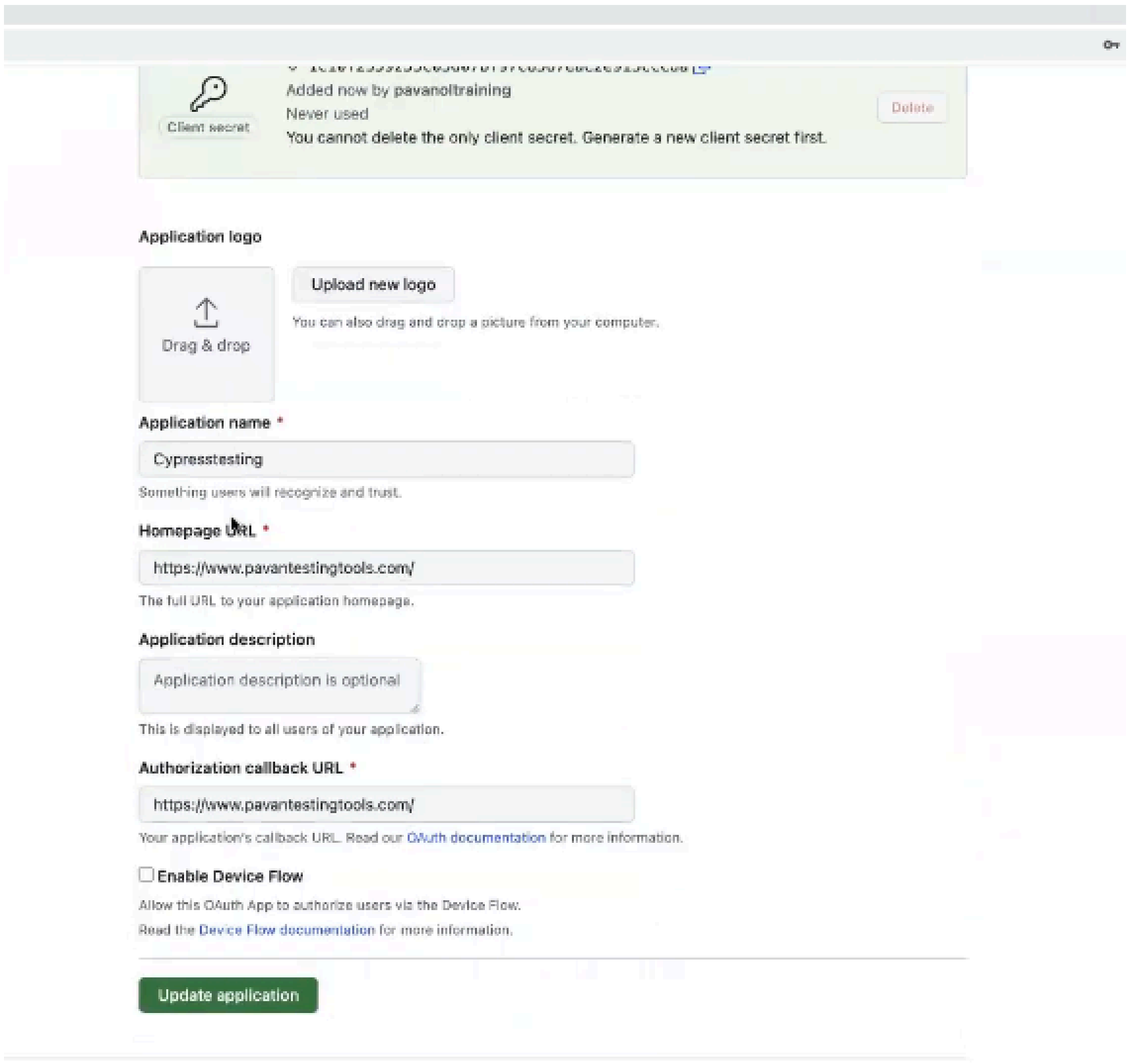
ded8c34b1cbcdcaf7149

Client secrets

You need a client secret to authenticate as the application to the API.

Transfer ownership
List this application in the Marketplace
Revoke all user tokens
Generate a new client secret

Enter github credentials.
Copy and keep it.
Click update application.



Client secret

Added now by pavanoltraining

Never used

You cannot delete the only client secret. Generate a new client secret first.

Delete

Application logo

Drag & drop

Upload new logo

You can also drag and drop a picture from your computer.

Application name *

Cypresstesting

Something users will recognize and trust.

Homepage URL *

https://www.pavantestingtools.com/

The full URL to your application homepage.

Application description

Application description is optional

This is displayed to all users of your application.

Authorization callback URL *

https://www.pavantestingtools.com/

Your application's callback URL. Read our OAuth documentation for more information.

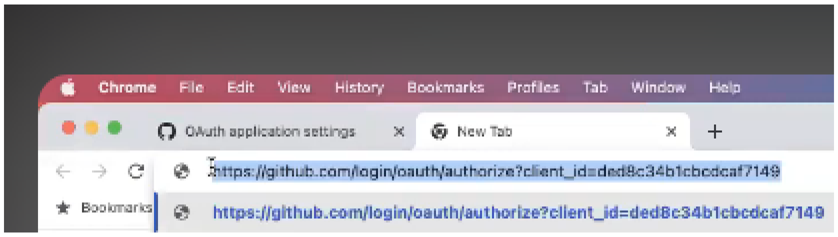
☐ Enable Device Flow

Allow this OAuth App to authorize users via the Device Flow.

Read the Device Flow documentation for more information.

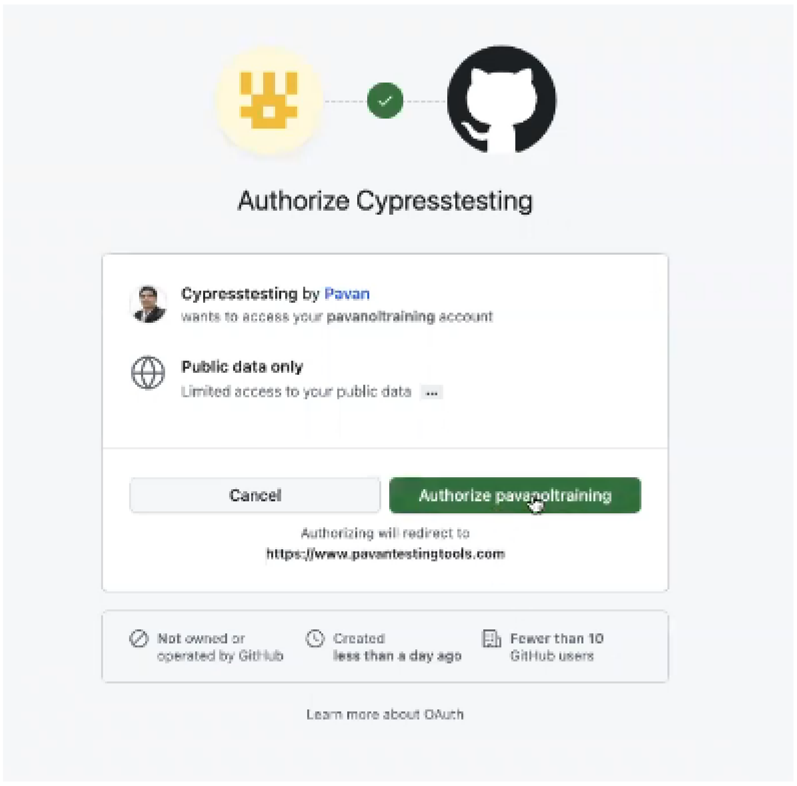
Update application

To get the auth code-
Go to the url provided by gihub.
It has client id.


Chrome File Edit View History Bookmarks Profiles Tab Window Help
OAuth application settings
New Tab
Bookmarks

https://github.com/login/oauth/authorize?client_id=ded8c34b1cbcdcaf7149

Click authorise.


Authorize Cypresstesting

Cypresstesting by Pavan
wants to access your pavanoltraining account

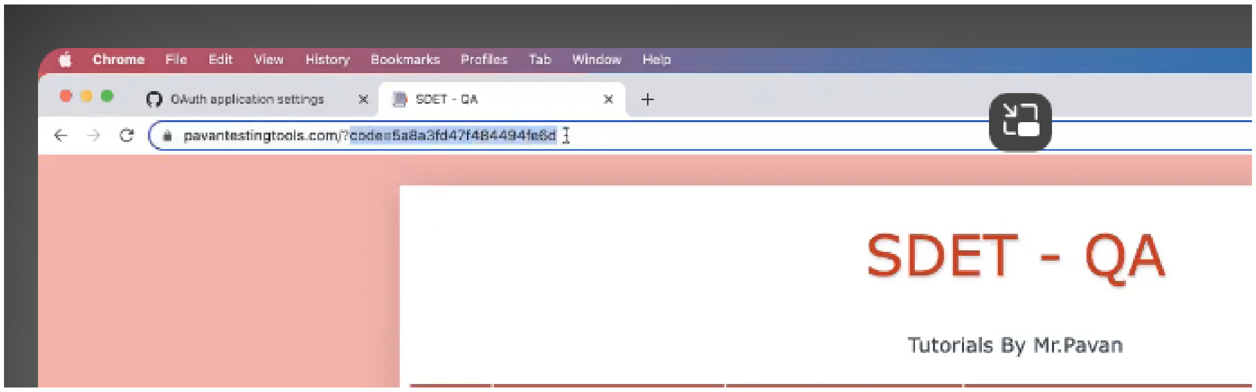
Public data only
Limited access to your public data

Cancel
Authorize pavanoltraining

Authorizing will redirect to
https://www.pavantestingtools.com

Not owned or operated by GitHub
Created less than a day ago
Fewer than 10 GitHub users
Learn more about OAuth

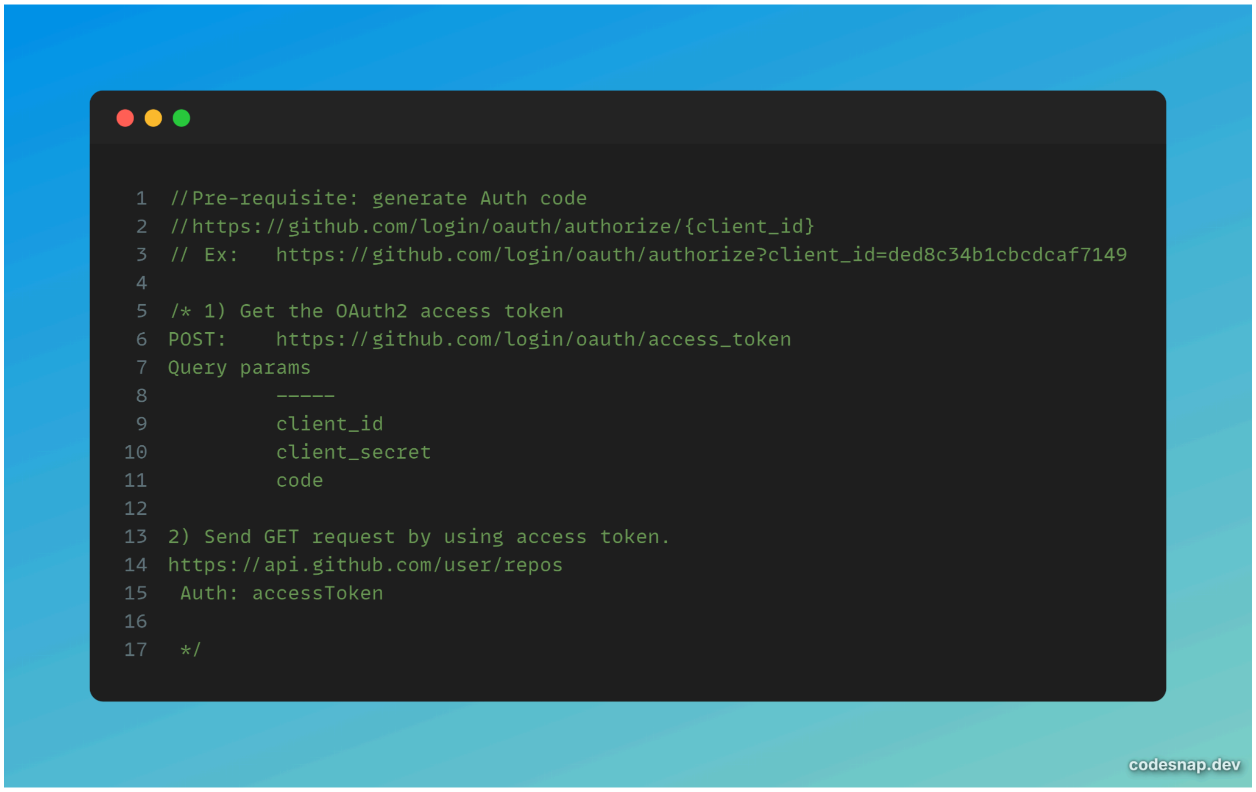
Now see the redirected url.
It contains the auth code.


Chrome File Edit View History Bookmarks Profiles Tab Window Help
OAuth application settings
SDET - QA
SDET - QA
Tutorials By Mr.Pavan

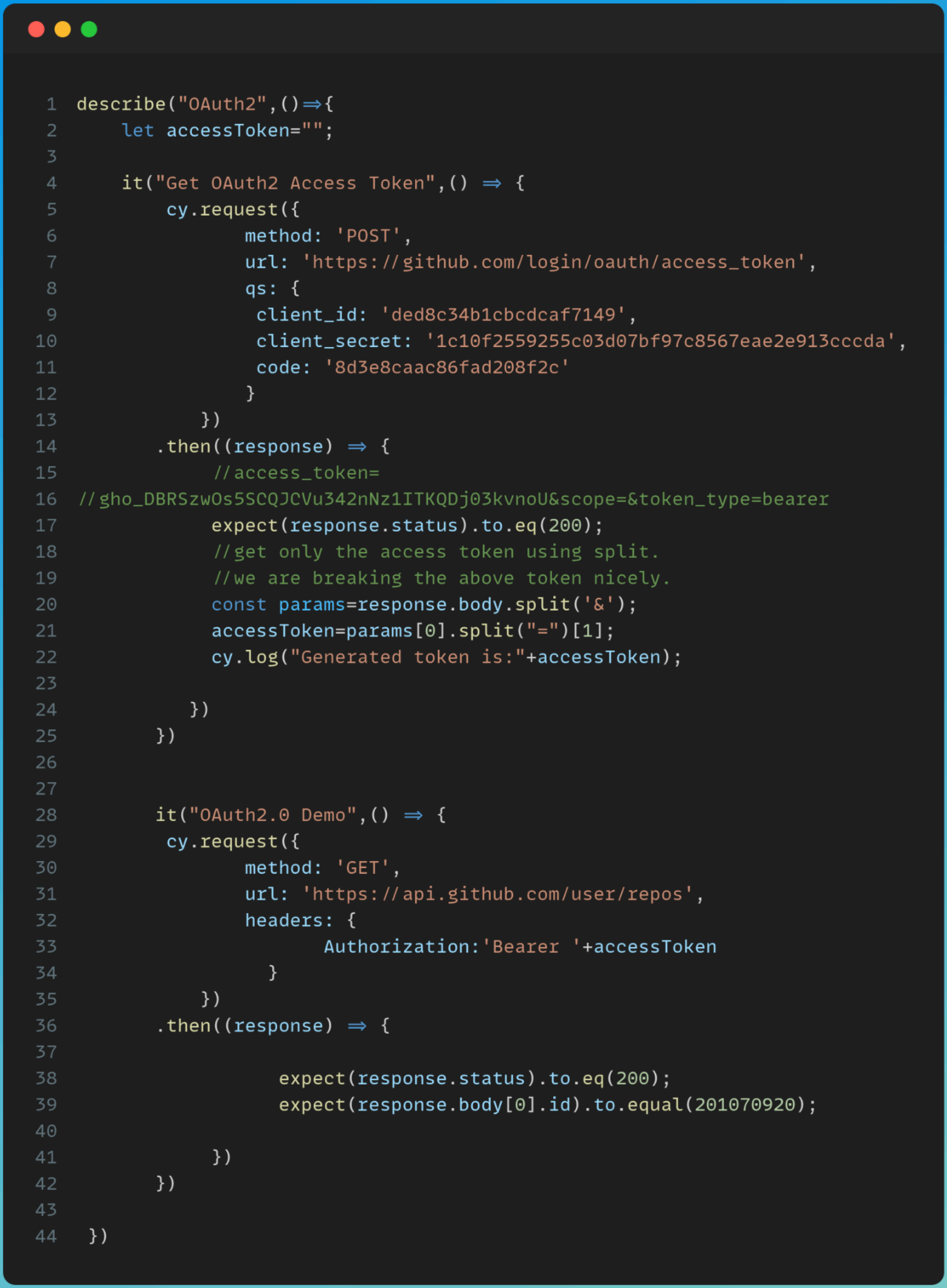
pavantestingtools.com/?code=5a8a3fd47f484494fe8d

Auth code newly created every time.

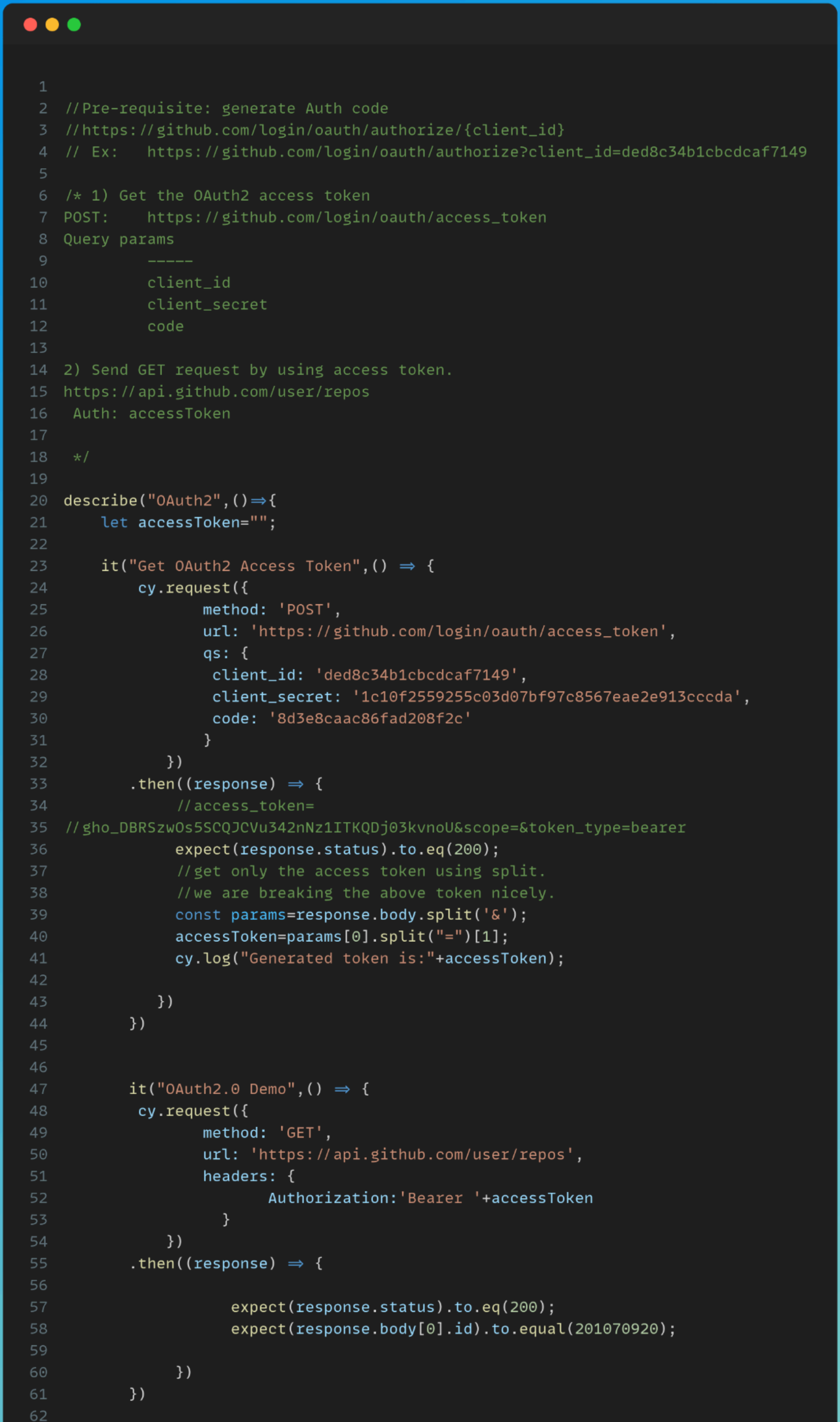
Now we want oauth 2 token.



```
//Pre-requisite: generate Auth code
//https://github.com/login/oauth/authorize/{client_id}
// Ex:   https://github.com/login/oauth/authorize?client_id=ded8c34b1cbcdcaf7149

/* 1) Get the OAuth2 access token
POST:    https://github.com/login/oauth/access_token
Query params
         -----
         client_id
         client_secret
         code

2) Send GET request by using access token.
https://api.github.com/user/repos
 Auth: accessToken

 */
```



```javascript
describe("OAuth2",()=>{
    let accessToken="";

    it("Get OAuth2 Access Token",() => {
        cy.request({
               method: 'POST',
               url: 'https://github.com/login/oauth/access_token',
               qs: {
                client_id: 'ded8c34b1cbcdcaf7149',
                client_secret: '1c10f2559255c03d07bf97c8567eae2e913cccda',
                code: '8d3e8caac86fad208f2c'
               }
           })
       .then((response) => {
            //access_token=
//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&scope=&token_type=bearer
            expect(response.status).to.eq(200);
            //get only the access token using split.
            //we are breaking the above token nicely.
            const params=response.body.split('&');
            accessToken=params[0].split("=")[1];
            cy.log("Generated token is:"+accessToken);

          })
       })


       it("OAuth2.0 Demo",() => {
        cy.request({
               method: 'GET',
               url: 'https://api.github.com/user/repos',
               headers: {
                      Authorization:'Bearer '+accessToken
                 }
           })
       .then((response) => {

                  expect(response.status).to.eq(200);
                  expect(response.body[0].id).to.equal(201070920);

            })
       })

 })
```



```javascript

//Pre-requisite: generate Auth code
//https://github.com/login/oauth/authorize/{client_id}
// Ex:   https://github.com/login/oauth/authorize?client_id=ded8c34b1cbcdcaf7149

/* 1) Get the OAuth2 access token
POST:    https://github.com/login/oauth/access_token
Query params
         -----
         client_id
         client_secret
         code

2) Send GET request by using access token.
https://api.github.com/user/repos
 Auth: accessToken

 */

describe("OAuth2",()=>{
    let accessToken="";

    it("Get OAuth2 Access Token",() => {
        cy.request({
               method: 'POST',
               url: 'https://github.com/login/oauth/access_token',
               qs: {
                client_id: 'ded8c34b1cbcdcaf7149',
                client_secret: '1c10f2559255c03d07bf97c8567eae2e913cccda',
                code: '8d3e8caac86fad208f2c'
               }
           })
       .then((response) => {
            //access_token=
//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&scope=&token_type=bearer
            expect(response.status).to.eq(200);
            //get only the access token using split.
            //we are breaking the above token nicely.
            const params=response.body.split('&');
            accessToken=params[0].split("=")[1];
            cy.log("Generated token is:"+accessToken);

          })
       })


       it("OAuth2.0 Demo",() => {
        cy.request({
               method: 'GET',
               url: 'https://api.github.com/user/repos',
               headers: {
                      Authorization:'Bearer '+accessToken
                 }
           })
       .then((response) => {

                  expect(response.status).to.eq(200);
                  expect(response.body[0].id).to.equal(201070920);

            })
       })

```