

```
Sublime Text  File  Edit  Selection  Find  View  Goto  Tools  Project  Window  Help
Create User - POST REST CALL

1 Create User - POST REST CALL
2
3 1. using hard coded json payload in script
4 2. using random email id in payload
5 3. read json from cyypress fixture and use it in request body
6   a. using require --> const
7   b. using cy.fixture - callback
8
```

```
PostUser.js - CypressAP:Training

JS PostUser.js 2 ●

PITests > JS PostUser.js > describe('post user request') callback > it('create user test') callback >
1  /// <reference types="Cypress" />
2
3 describe('post user request', () => {
4   let accessToken = '007526d9efdbc07e084ff7a6d4cfcc90588fbe20641c00faebf45:
5
```

```
Window  Help
PostUser.js - CypressAP:Training

JS PostUser.js 1 ●

PostUser.js > describe('post user request') callback > it('create user test') callback >
5
   Open Cypress | Set ".only"
6   it('create user test', () => {
7     cy.request({
8       method: 'POST',
9       url: 'https://gorest.co.in/public/v1/users',
10      headers: {
11        'Authorization': 'Bearer ' + accessToken
12      },
13      body: {
14        "name": "Test Automation",
15        "gender": "male",
16        "email": "nwontest@gmail.com",
17        "status": "active"
18      }
19    })
20
```

```

        "gender": "male",
        "email": "cytesting@gmail.com",
        "status": "active"
    }

    }).then((res)=>{
        expect(res.status).to.eq(201)
        expect(res.body.data).has.property('email', 'cytesting@gmail.com')
        expect(res.body.data).has.property('name', 'Test Automation')
        expect(res.body.data).has.property('status', 'active')
        expect(res.body.data).has.property('gender', 'male')
    })

```

Added21 to log response

```

13         body: {
14             "name": "Test Automation",
15             "gender": "male",
16             "email": "cytesting12@gmail.com",
17             "status": "active"
18         }
19
20     }).then((res)=>{
21         cy.log(res)
22         expect(res.status).to.eq(201)
23         expect(res.body.data).has.property('email', 'cyt
24         expect(res.body.data).has.property('name', 'Test
25         expect(res.body.data).has.property('status', 'ac

```

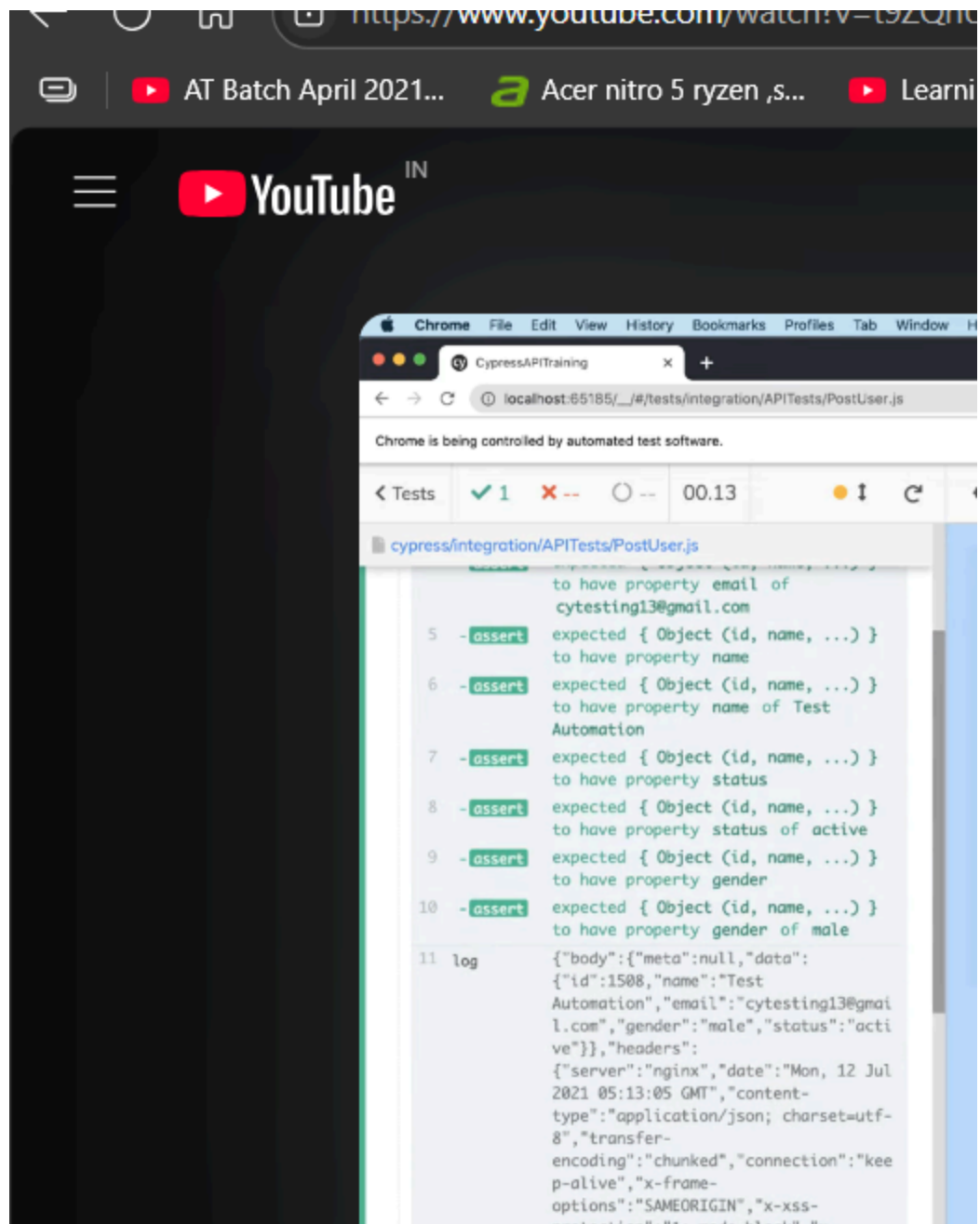
```

8 - assert expected { Object (
9 - assert expected { Object (
10 - assert expected { Object (
log Object{9}

```

Updated21

```
14         "name": "Test Automation",
15         "gender": "male",
16         "email": "cytesting13@gmail.co
17         "status": "active"
18     }
19
20     }).then((res) => {
21         cy.log(JSON.stringify(res))
22         expect(res.status).to.eq(201)
23         expect(res.body.data).has.property
```



Random generator

```
Terminal Window Help
PostUser.js — CypressAPITraining

JS PostUser.js • JS GetUser.js

cypress > integration > APITests > JS PostUser.js > describe('post user request') callback > it('create user te
1  /// <reference types="Cypress" />
2
3  describe('post user request', () => {
4    let accessToken = '007526d9efdbc07e084ff7a6d4cfcc90588fbe20641c00faebf45a7f3b2eaf33'
5    let randomText = ""
6    let testEmail = ""
7
8    Open Cypress | Set ".only"
9    it('create user test', () => {
10
11      var pattern = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"
12      for (var i = 0; i < 10; i++)
13        randomText+=pattern.charAt(Math.floor(Math.random() * pattern.length));
14      testEmail = randomText + '@gmail.com'
15
16
```

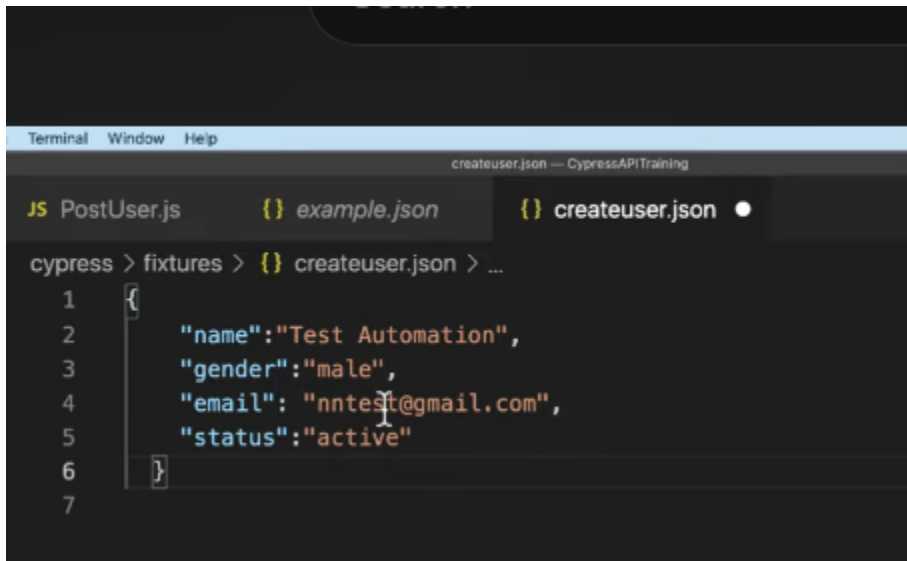
24

```
15  cy.request({
16    method: 'POST',
17    url: 'https://gorest.co.in/public/v1/users',
18    headers: {
19      'Authorization': 'Bearer ' + accessToken
20    },
21    body: {
22      "name": "Test Automation",
23      "gender": "male",
24      "email": testEmail,
25      "status": "active"
26    }
27  }).then((res) => {
28
```

31

```
24      "email": testEmail,
25      "status": "active"
26    }
27  }).then((res) => {
28    cy.log(JSON.stringify(res))
29    expect(res.status).to.eq(201)
30    expect(res.body.data).has.property('email', testEmail)
31    expect(res.body.data).has.property('name', 'Test Automation')
32    expect(res.body.data).has.property('status', 'active')
33
```

Create user json

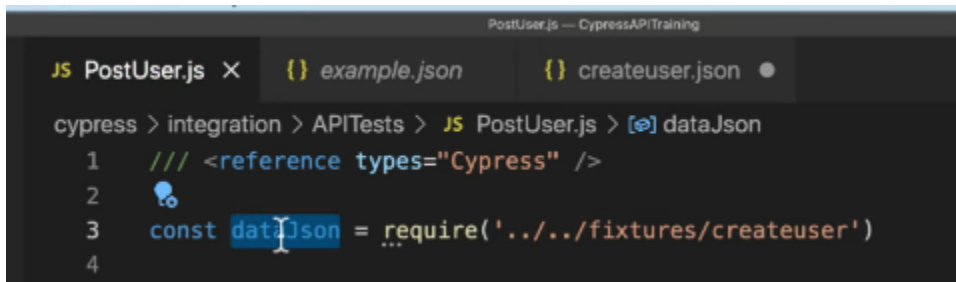


A screenshot of a code editor window titled "createuser.json — CypressAPITraining". The editor shows a JSON file named "createuser.json" with the following content:

```
1 {  
2   "name": "Test Automation",  
3   "gender": "male",  
4   "email": "nntest@gmail.com",  
5   "status": "active"  
6 }  
7
```

Added3

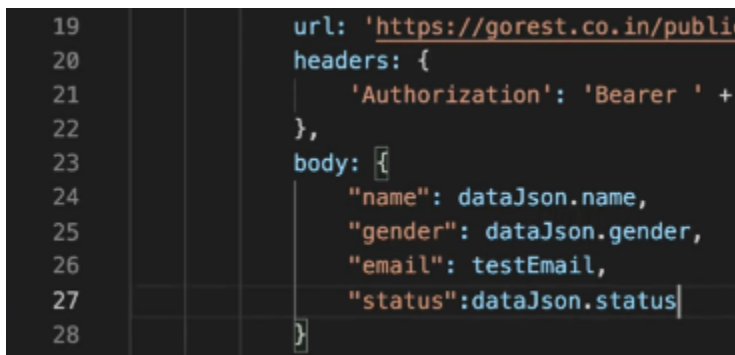
No need of extension



A screenshot of a code editor window titled "PostUser.js — CypressAPITraining". The editor shows a JavaScript file named "PostUser.js" with the following content:

```
1 // <reference types="Cypress" />  
2  
3 const dataJson = require('../../fixtures/createuser')  
4
```

Changed body



A screenshot of a code editor window showing the body of a REST client request. The content is as follows:

```
19 url: 'https://gorest.co.in/public/v2/users/  
20 headers: {  
21   'Authorization': 'Bearer ' +  
22 },  
23 body: {  
24   "name": dataJson.name,  
25   "gender": dataJson.gender,  
26   "email": testEmail,  
27   "status": dataJson.status  
28 }
```

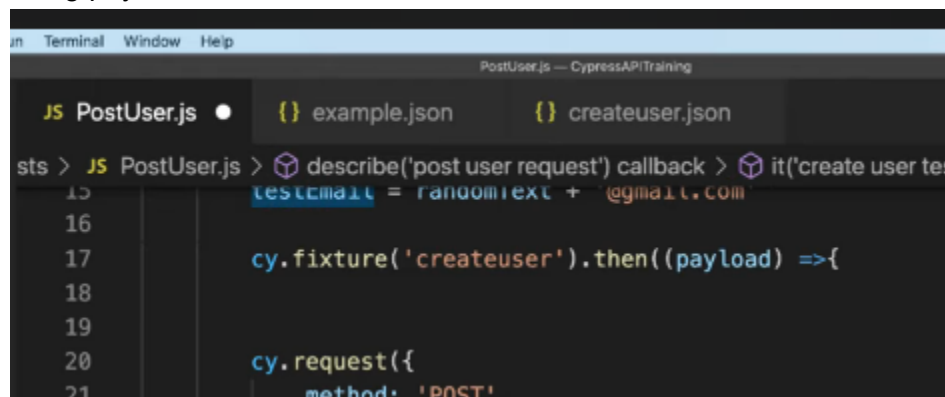
Changed response

```

27         "status":dataJson.status
28     }
29
30     }).then((res)=>{
31         cy.log(JSON.stringify(res))
32         expect(res.status).to.eq(201)
33         expect(res.body.data).has.property('email', testEmail)
34         expect(res.body.data).has.property('name',dataJson.name)
35         expect(res.body.data).has.property('status',dataJson.status)
36         expect(res.body.data).has.property('gender',dataJson.gender)
37     })
38 }

```

Using pay load



```

PostUser.js — CypressAPITraining
JS PostUser.js • {} example.json {} createuser.json
sts > JS PostUser.js > describe('post user request') callback > it('create user test')
15     testEmail = randomText + '@gmail.com'
16
17     cy.fixture('createuser').then((payload) =>{
18
19
20     cy.request({
21         method: 'POST',

```

From url to test to assert should be inside fixture if we are using.

Pass in body

```

25     },
26     body: {
27         "name": payload.name,
28         "gender": payload.gender,
29         "email": testEmail,
30         "status":payload.status
31     }
32 }

```

Assert with fixture

```
30         "status":payload.status
31     }
32
33 }).then((res)=>{
34     cy.log(JSON.stringify(res))
35     expect(res.status).to.eq(201)
36     expect(res.body.data).has.property('email', testEmail)
37     expect(res.body.data).has.property('name',payload.name)
38     expect(res.body.data).has.property('status',payload.status)
39     expect(res.body.data).has.property('gender',payload.gender)
40
41 })
```