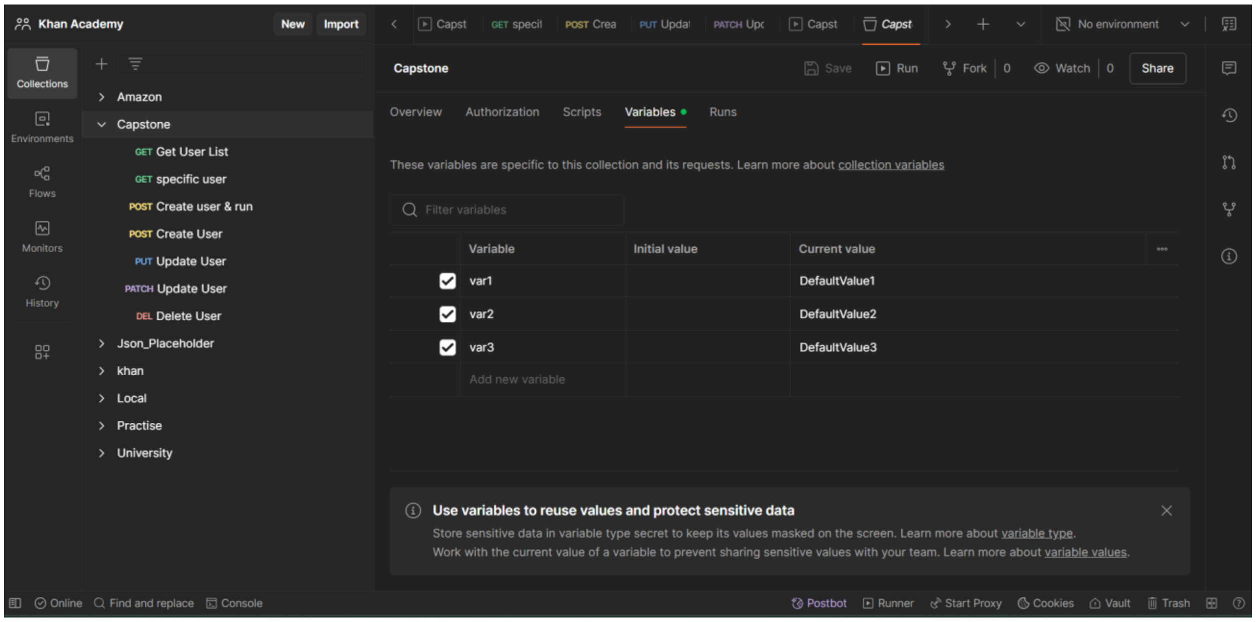
POSTMAN USING JSON PLACEHOLDER

1.Create a collection for accessing the methods (GET,POST,PUT,PATCH,DELETE)

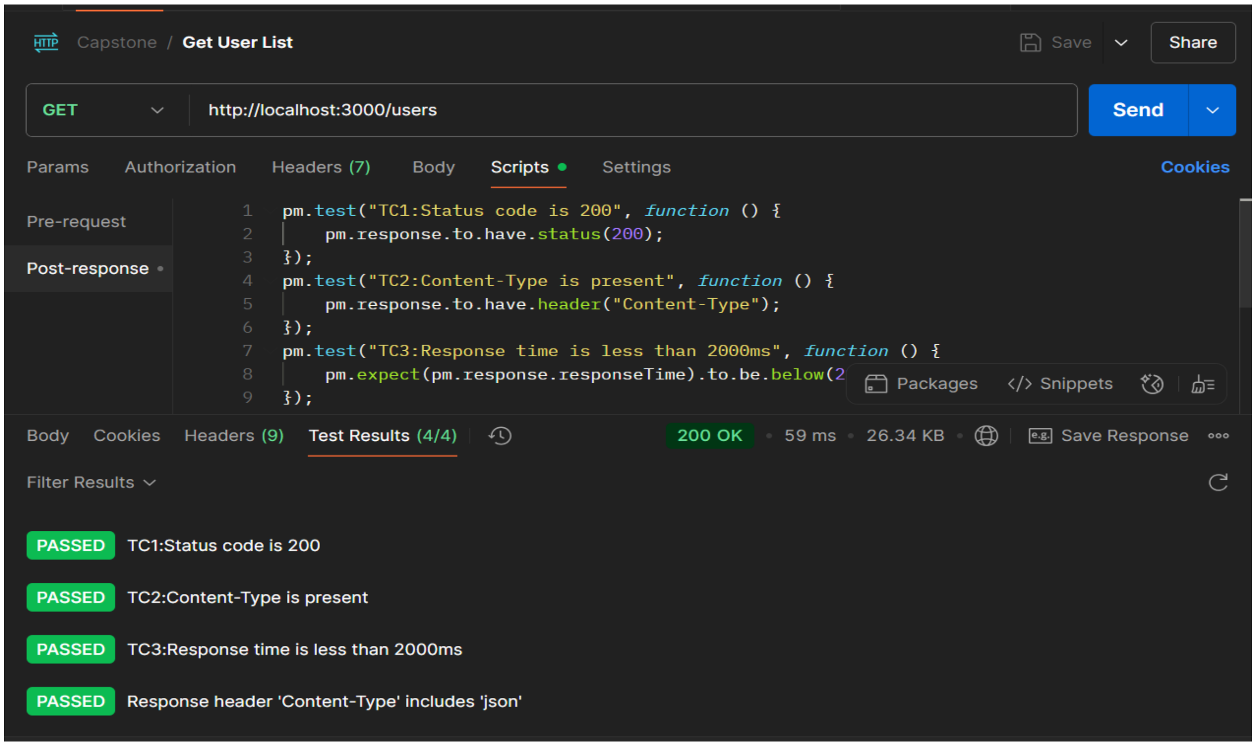


These variables are specific to this collection and its requests. Learn more about collection variables

	Variable	Initial value	Current value
✓	var1		DefaultValue1
✓	var2		DefaultValue2
✓	var3		DefaultValue3
	Add new variable		

Use variables to reuse values and protect sensitive data
Store sensitive data in variable type secret to keep its values masked on the screen. Learn more about variable type.
Work with the current value of a variable to prevent sharing sensitive values with your team. Learn more about variable values.

1.GET Method: Retrieves data from the server without changing it. Used to fetch information like user details.



GET http://localhost:3000/users

```
pm.test("TC1:Status code is 200", function () {
    pm.response.to.have.status(200);
});
pm.test("TC2:Content-Type is present", function () {
    pm.response.to.have.header("Content-Type");
});
pm.test("TC3:Response time is less than 2000ms", function () {
    pm.expect(pm.response.responseTime).to.be.below(2
});
```

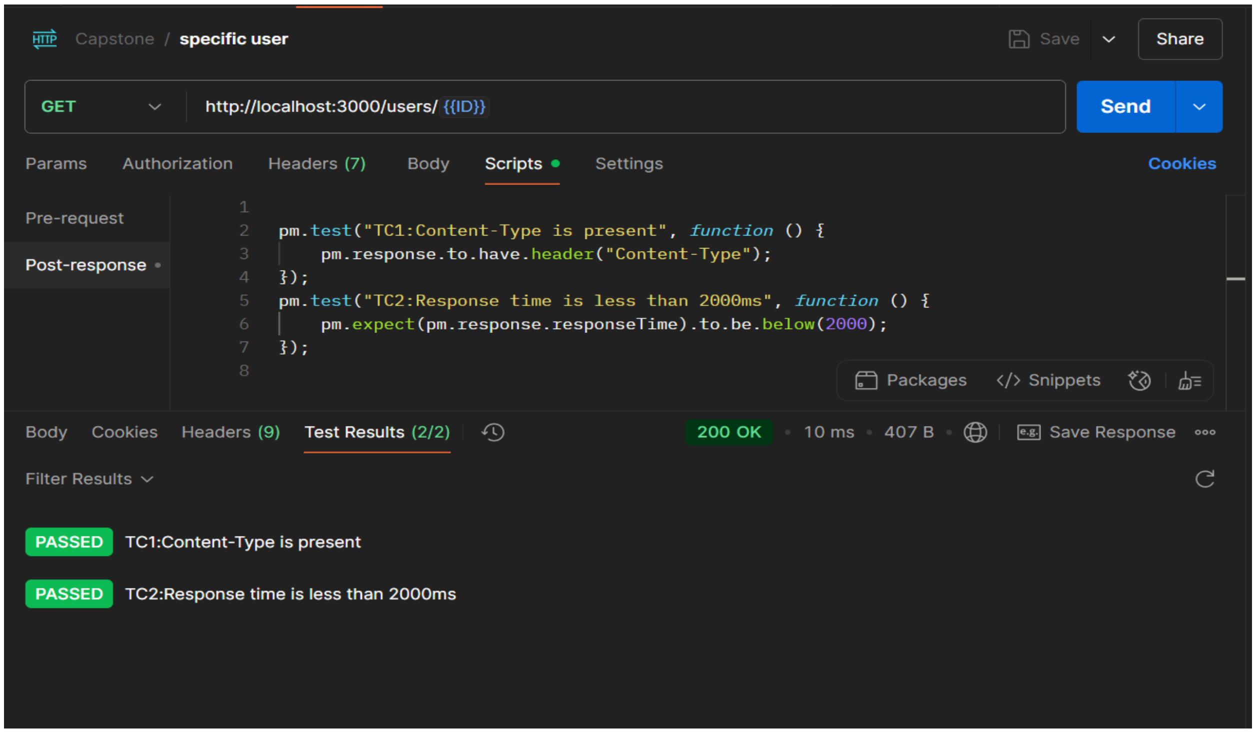
200 OK • 59 ms • 26.34 KB

PASSED TC1:Status code is 200

PASSED TC2:Content-Type is present

PASSED TC3:Response time is less than 2000ms

PASSED Response header 'Content-Type' includes 'json'


Capstone / specific user

GET http://localhost:3000/users/ {{ID}}

Scripts — Post-response

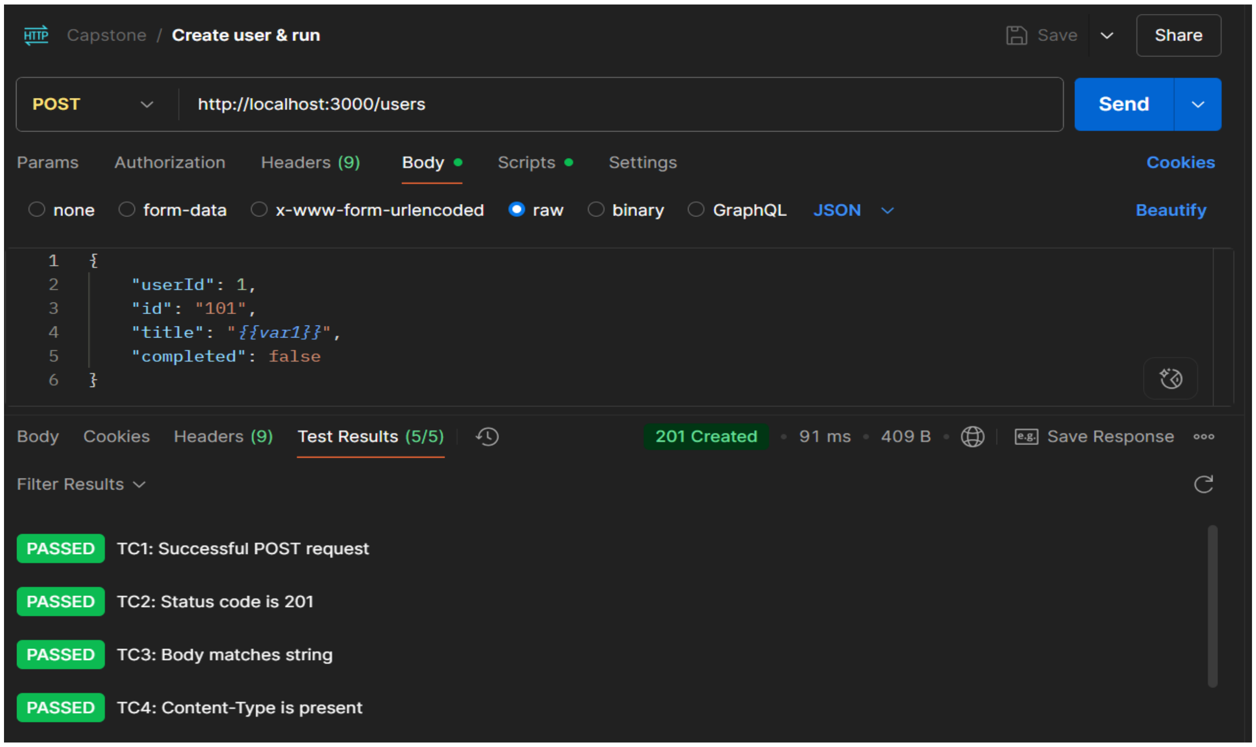
```
pm.test("TC1:Content-Type is present", function () {
    pm.response.to.have.header("Content-Type");
});
pm.test("TC2:Response time is less than 2000ms", function () {
    pm.expect(pm.response.responseTime).to.be.below(2000);
});
```

Test Results (2/2) — 200 OK • 10 ms • 407 B

PASSED TC1:Content-Type is present

PASSED TC2:Response time is less than 2000ms

2.POST Method: Sends new data to the server to create a resource. Used when adding ne items like users.


Capstone / Create user & run

POST http://localhost:3000/users

Body — raw JSON

```
{
    "userId": 1,
    "id": "101",
    "title": "{{var1}}",
    "completed": false
}
```

Test Results (5/5) — 201 Created • 91 ms • 409 B

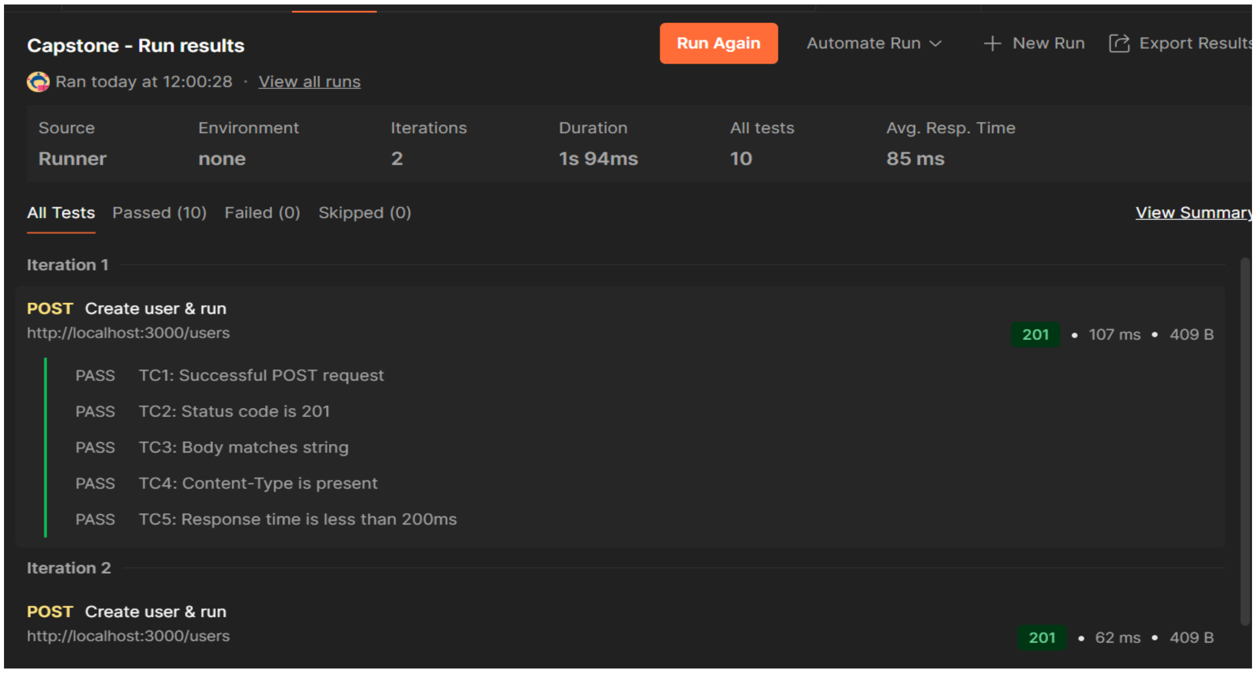
PASSED TC1: Successful POST request

PASSED TC2: Status code is 201

PASSED TC3: Body matches string

PASSED TC4: Content-Type is present

POST Method: Sends new data to the server to create a resource. Used when adding new items like users using csv file.



Capstone - Run results

Ran today at 12:00:28 · View all runs

```
Source      Environment   Iterations   Duration   All tests   Avg. Resp. Time
Runner      none          2            1s 94ms    10          85 ms
```

All Tests   Passed (10)   Failed (0)   Skipped (0)   View Summary

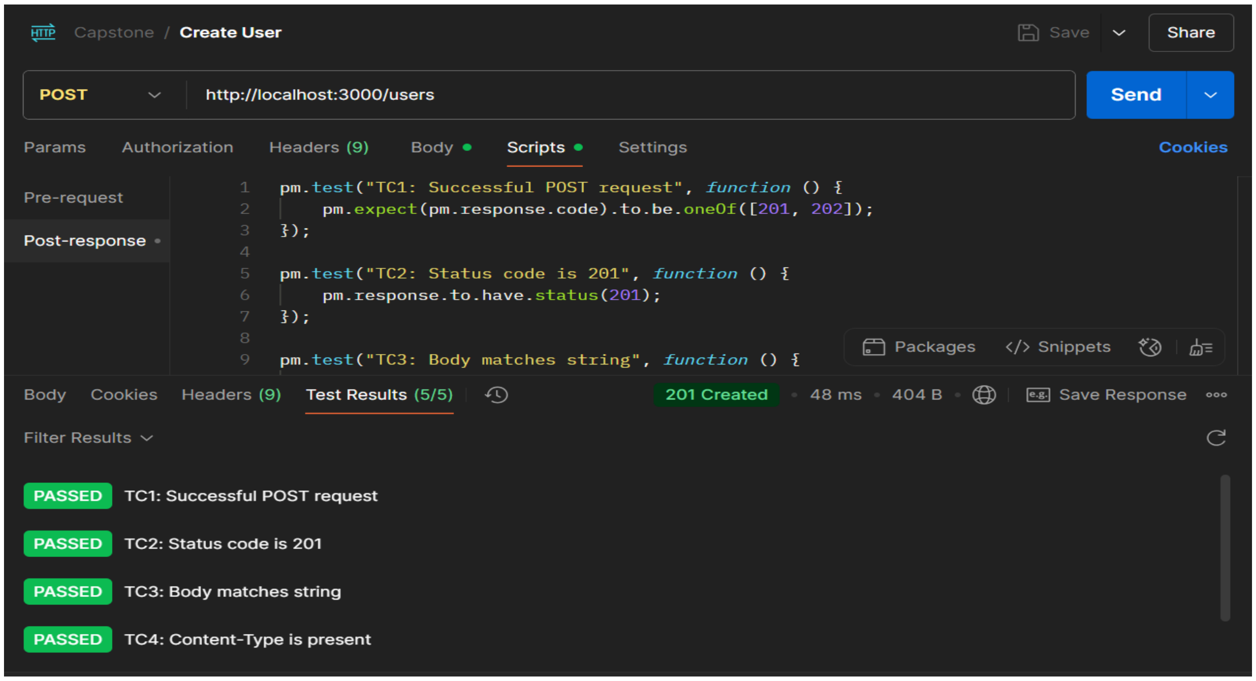
Iteration 1

POST Create user & run
http://localhost:3000/users   201 • 107 ms • 409 B

```
PASS   TC1: Successful POST request
PASS   TC2: Status code is 201
PASS   TC3: Body matches string
PASS   TC4: Content-Type is present
PASS   TC5: Response time is less than 200ms
```

Iteration 2

POST Create user & run
http://localhost:3000/users   201 • 62 ms • 409 B



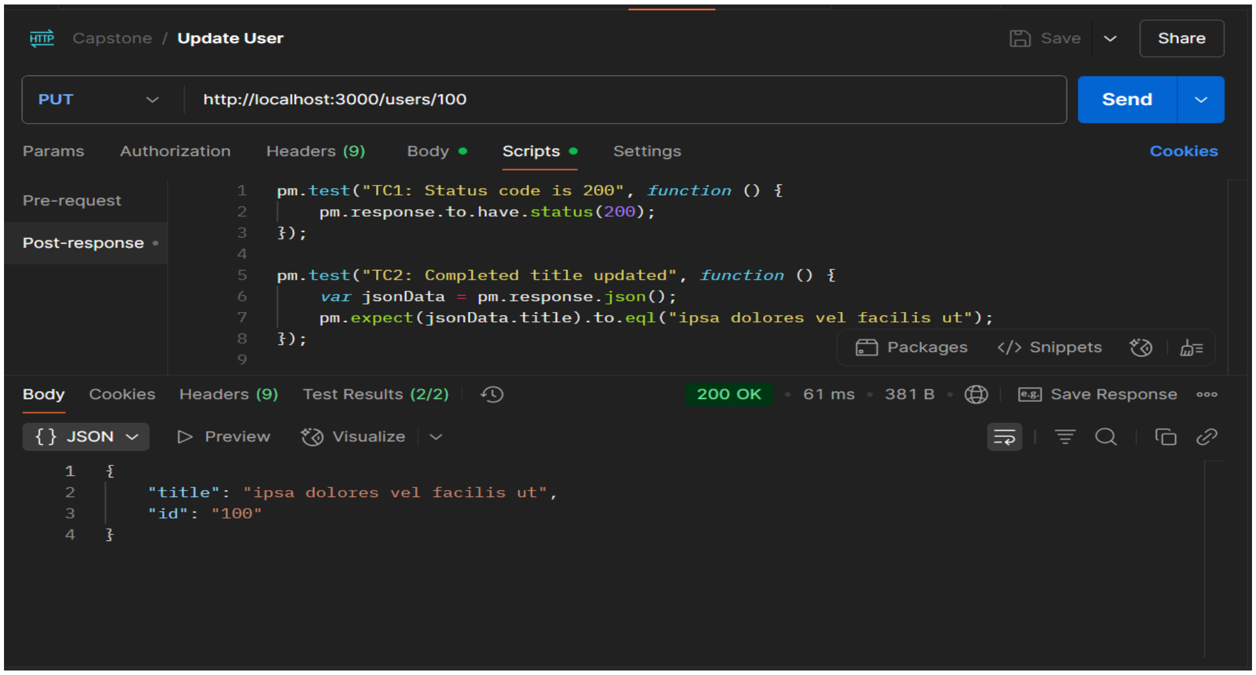
POST http://localhost:3000/users

```
pm.test("TC1: Successful POST request", function () {
    pm.expect(pm.response.code).to.be.oneOf([201, 202]);
});

pm.test("TC2: Status code is 201", function () {
    pm.response.to.have.status(201);
});

pm.test("TC3: Body matches string", function () {
```

Body   Cookies   Headers (9)   Test Results (5/5)   201 Created • 48 ms • 404 B

```
PASSED  TC1: Successful POST request
PASSED  TC2: Status code is 201
PASSED  TC3: Body matches string
PASSED  TC4: Content-Type is present
```

3.PUT Method: Replaces an existing resource entirely with new data. Ideal for full updates where all fields are changed.

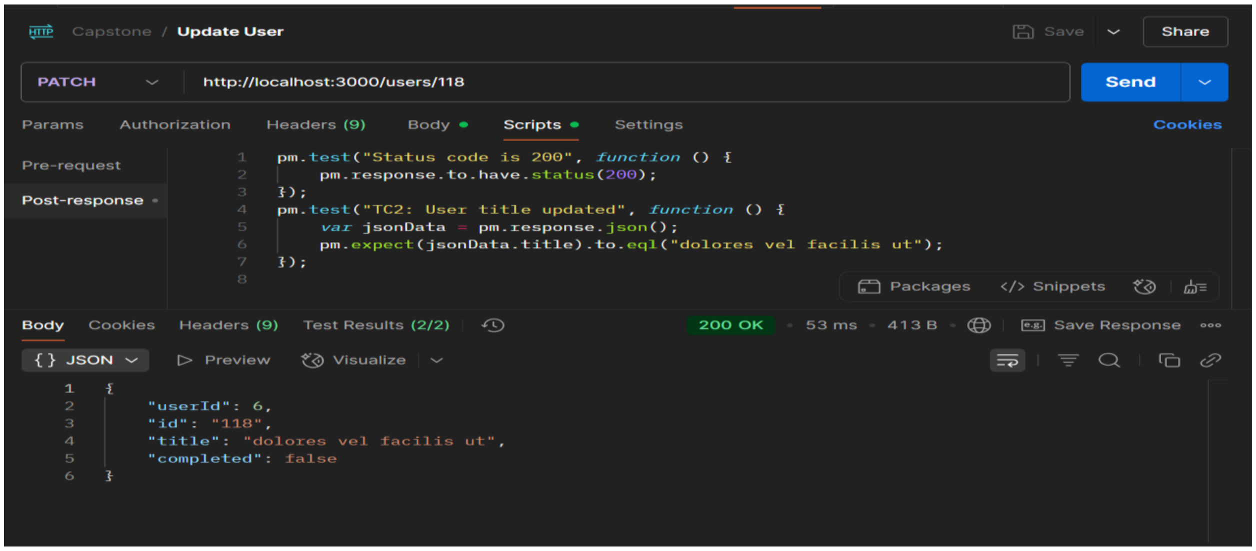


```
http://localhost:3000/users/100
```

```
pm.test("TC1: Status code is 200", function () {
    pm.response.to.have.status(200);
});

pm.test("TC2: Completed title updated", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.title).to.eql("ipsa dolores vel facilis ut");
});
```

```
{
    "title": "ipsa dolores vel facilis ut",
    "id": "100"
}
```

4.PATCH Method: Updates part of an existing resource with partial data. Useful for making small changes without replacing everything.

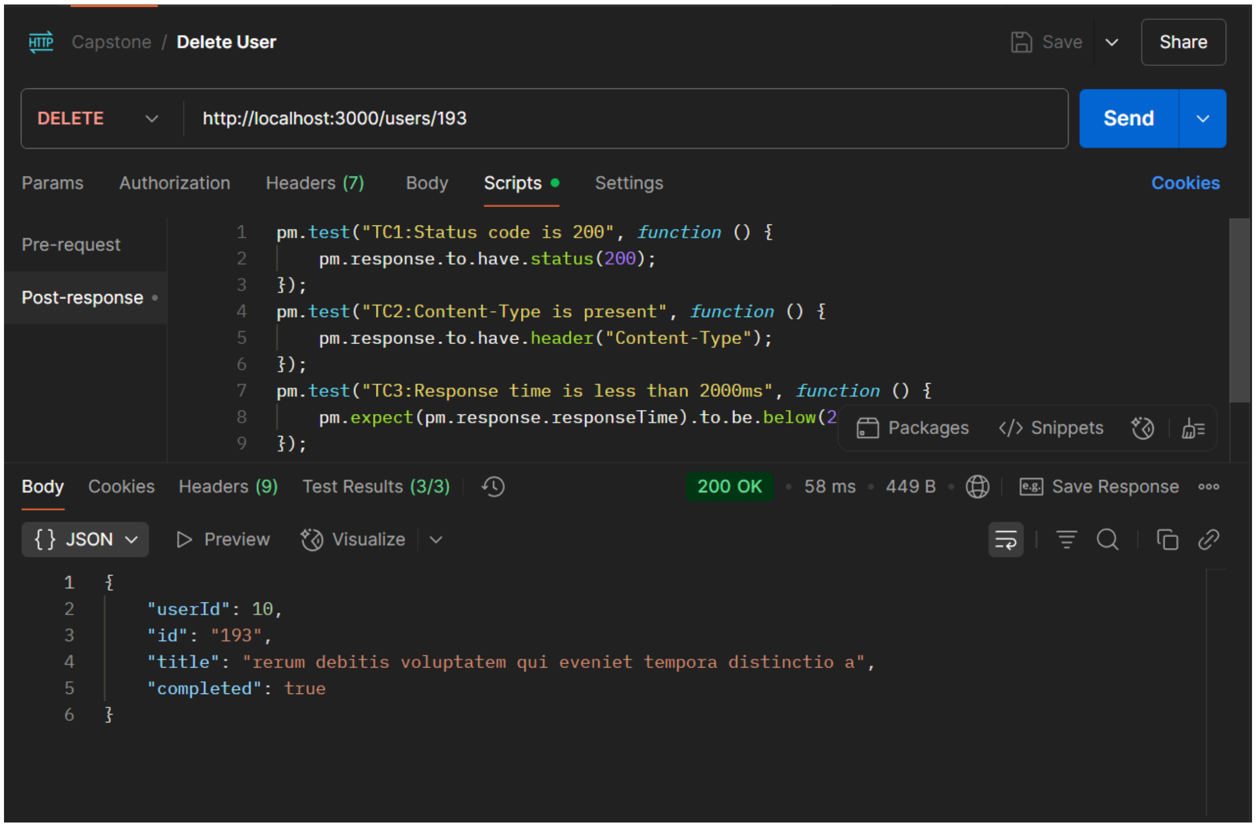


```
http://localhost:3000/users/118
```

```
pm.test("Status code is 200", function () {
    pm.response.to.have.status(200);
});
pm.test("TC2: User title updated", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.title).to.eql("dolores vel facilis ut");
});
```

```
{
    "userId": 6,
    "id": "118",
    "title": "dolores vel facilis ut",
    "completed": false
}
```

5.DELETE Method: Removes a resource from the server permanently. Used to delete data like user posts.



DELETE http://localhost:3000/users/193

```
pm.test("TC1:Status code is 200", function () {
    pm.response.to.have.status(200);
});
pm.test("TC2:Content-Type is present", function () {
    pm.response.to.have.header("Content-Type");
});
pm.test("TC3:Response time is less than 2000ms", function () {
    pm.expect(pm.response.responseTime).to.be.below(2
});
```

200 OK • 58 ms • 449 B

```
{
    "userId": 10,
    "id": "193",
    "title": "rerum debitis voluptatem qui eveniet tempora distinctio a",
    "completed": true
}
```