

Postman

Testy API zostały sporządzone na podstawie dokumentacji na stronie <https://petstore.swagger.io/#/>.

Utworzono nową kolekcję.

1. POST – dodawanie nowego użytkownika

The image shows the Swagger UI for the `POST /user` endpoint. The title is `POST /user Create user`. A note states: "This can only be done by the logged in user." There is a "Try it out" button. The "Parameters" section shows a required `body` parameter of type `object` (body), described as "Created user object". An example JSON value is displayed in a dark box:

```
{  "id": 0,  "username": "string",  "firstName": "string",  "lastName": "string",  "email": "string",  "password": "string",  "phone": "string",  "userStatus": 0}
```

 The "Parameter content type" is set to `application/json`. The "Responses" section shows a default response with status `200` and description "successful operation", with a response content type of `application/json`.

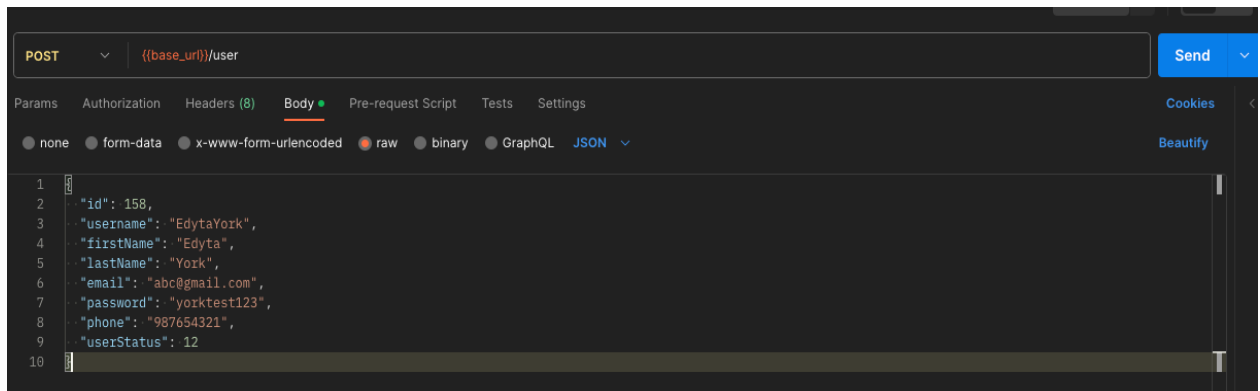
Z dokumentacji rozpoczęto dodawanie nowego użytkownika przy pomocy POST

The image shows the Postman interface for the `POST /user` endpoint. The title is `POST POST - Create user`. The URL is `{{base_url}}/user`. The "Body" tab is selected, and the content type is set to `JSON`. The JSON body is:

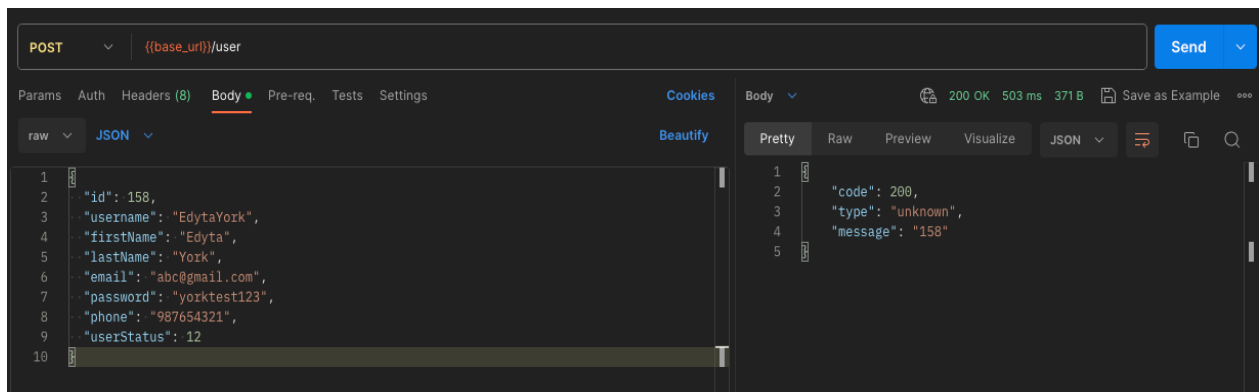
```
{  "id": 0,  "username": "string",  "firstName": "string",  "lastName": "string",  "email": "string",  "password": "string",  "phone": "string",  "userStatus": 0}
```

 The interface includes tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. There are also buttons for Save, Send, Cookies, and Beautify.

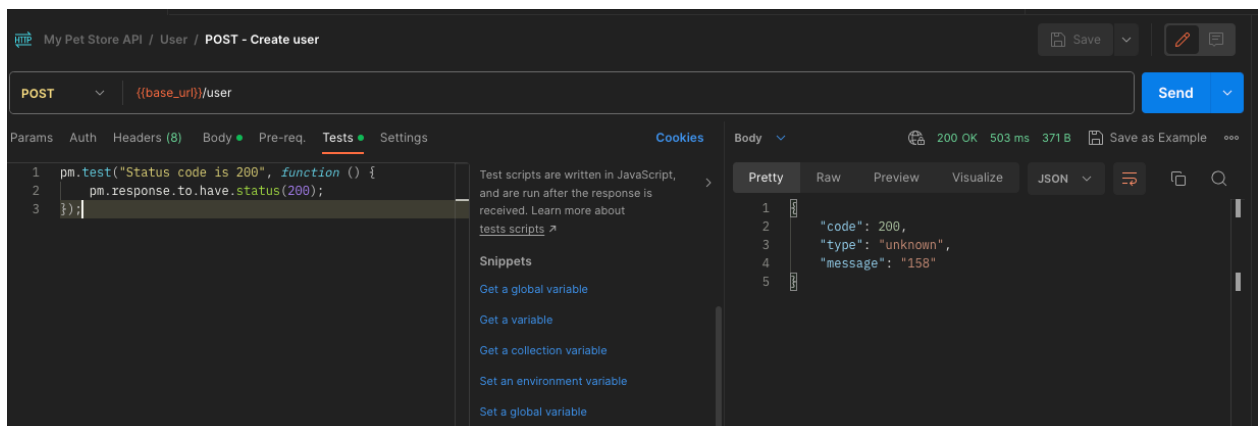
Wprowadzono dane dla nowego użytkownika:

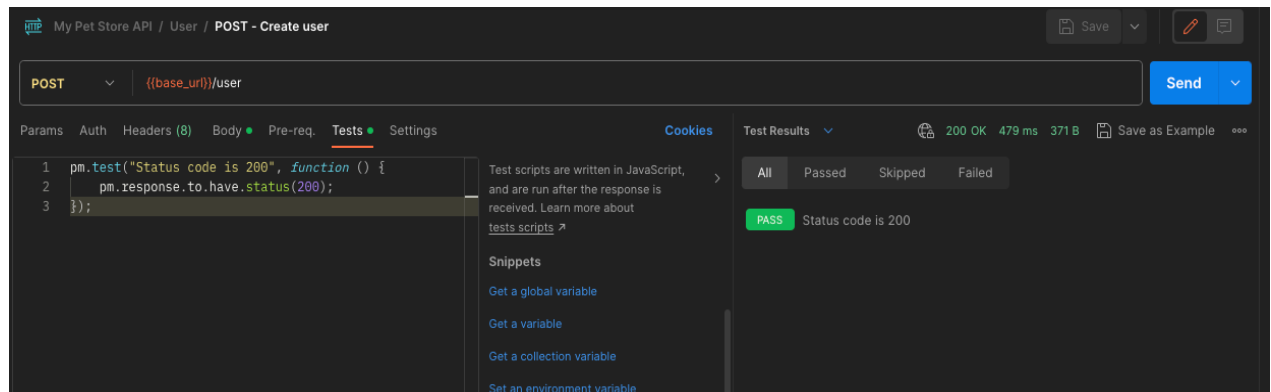


Wysłano żądanie i otrzymano status code 200 OK. Oznacza to, że nowy użytkownik został dodany.

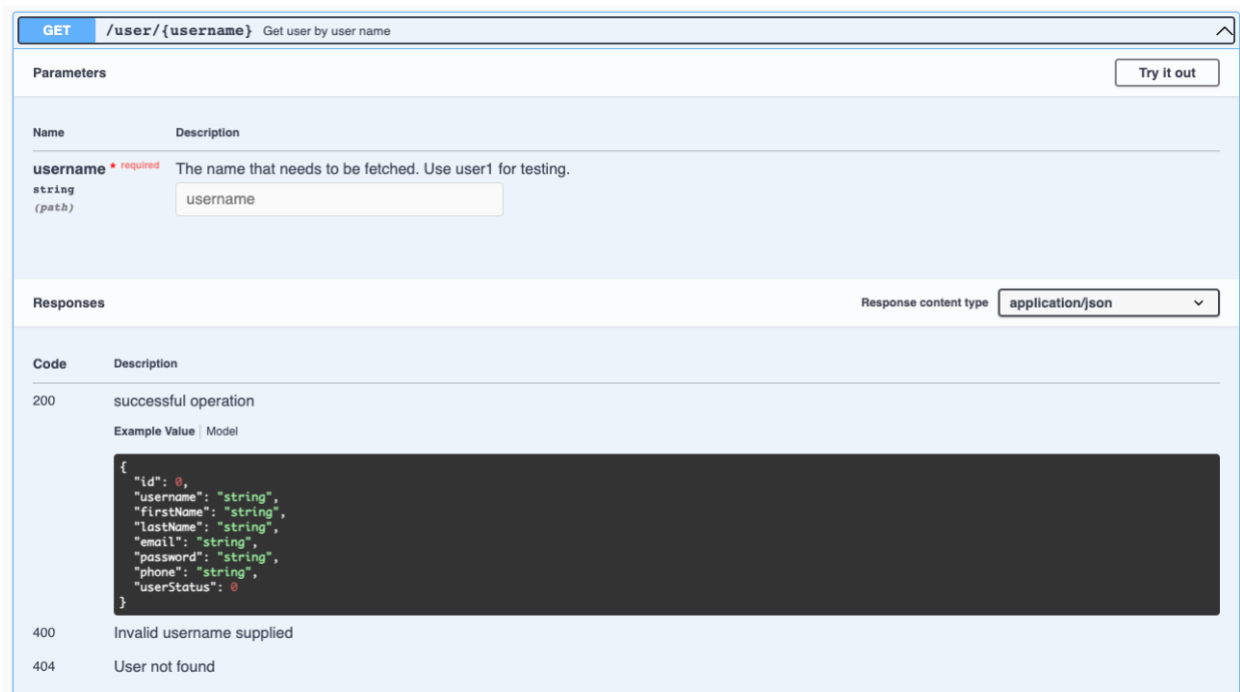


Wyniki wysłanych testów pokazują status 200.

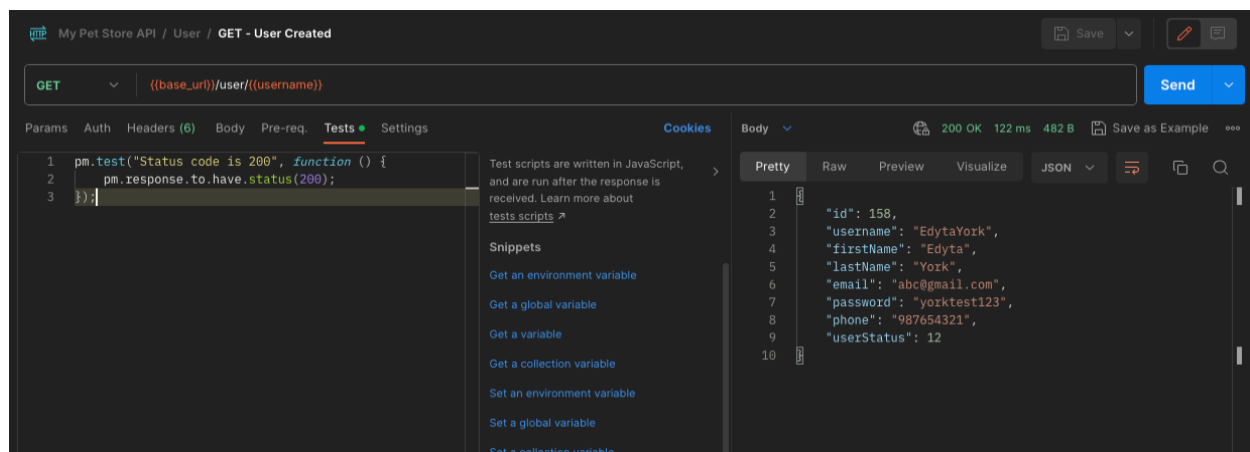




2. GET – pobieranie danych dodanego użytkownika



Z dokumentacji przy pomocy komendy GET sprawdzono, czy został dodany nowy użytkownik:



Otrzymano status 200 OK, czyli użytkownik został dodany prawidłowo.

3. PUT – aktualizacja nazwy użytkownika

Zgodnie z dokumentacją należy zmienić nazwę użytkownika:

The screenshot shows a REST client interface for a PUT request to the endpoint `/user/{username}`. The request is labeled "Updated user". A note states: "This can only be done by the logged in user." The "Parameters" section includes a "Try it out" button. The "Name" and "Description" table lists two parameters:

Name	Description
username * required string (path)	name that need to be updated
body * required object (body)	Updated user object

Below the table, an "Example Value" for the body is shown as a JSON object:

```
{  "id": 0,  "username": "string",  "firstName": "string",  "lastName": "string",  "email": "string",  "password": "string",  "phone": "string",  "userStatus": 0}
```

The "Parameter content type" is set to `application/json`.

Zamieniono nazwę użytkownika:

The screenshot shows a REST client interface for a PUT request to the endpoint `{{(base_url)}}/user/{{(username1)}}`. The "Body" tab is selected, and the JSON content is displayed in a code editor:

```
1  { 2    "id": 158, 3    "username": "NowyYork", 4    "firstName": "Edyta", 5    "lastName": "York", 6    "email": "abc@gmail.com", 7    "password": "yorktest123", 8    "phone": "987654321", 9    "userStatus": 12 10 }
```

Wykonano test i otrzymano status 200, czyli aktualizacja przebiegła pomyślnie:

The screenshot shows a REST client interface for a PUT request to the endpoint `{{(base_url)}}/user/{{(username1)}}`. The "Tests" tab is selected, and the test script is shown:

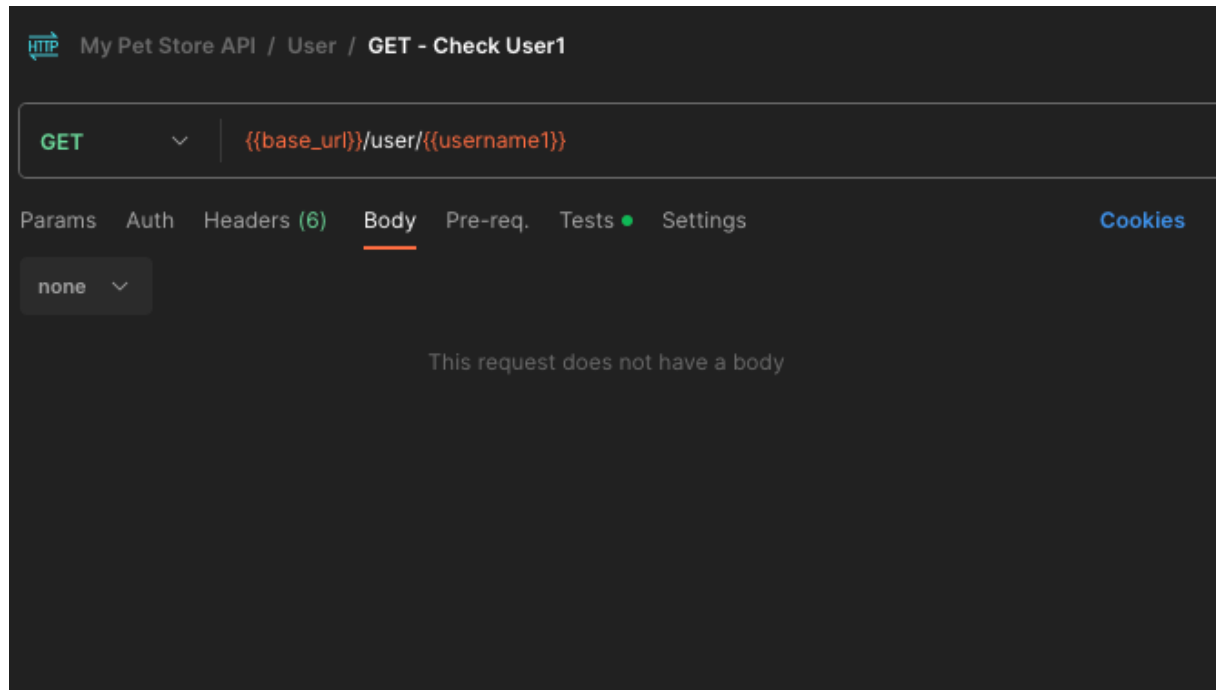
```
1  pm.test("Status code is 200", function () { 2    pm.response.to.have.status(200); 3  });
```

The test result is displayed on the right, showing a status of `200 OK` with a response body:

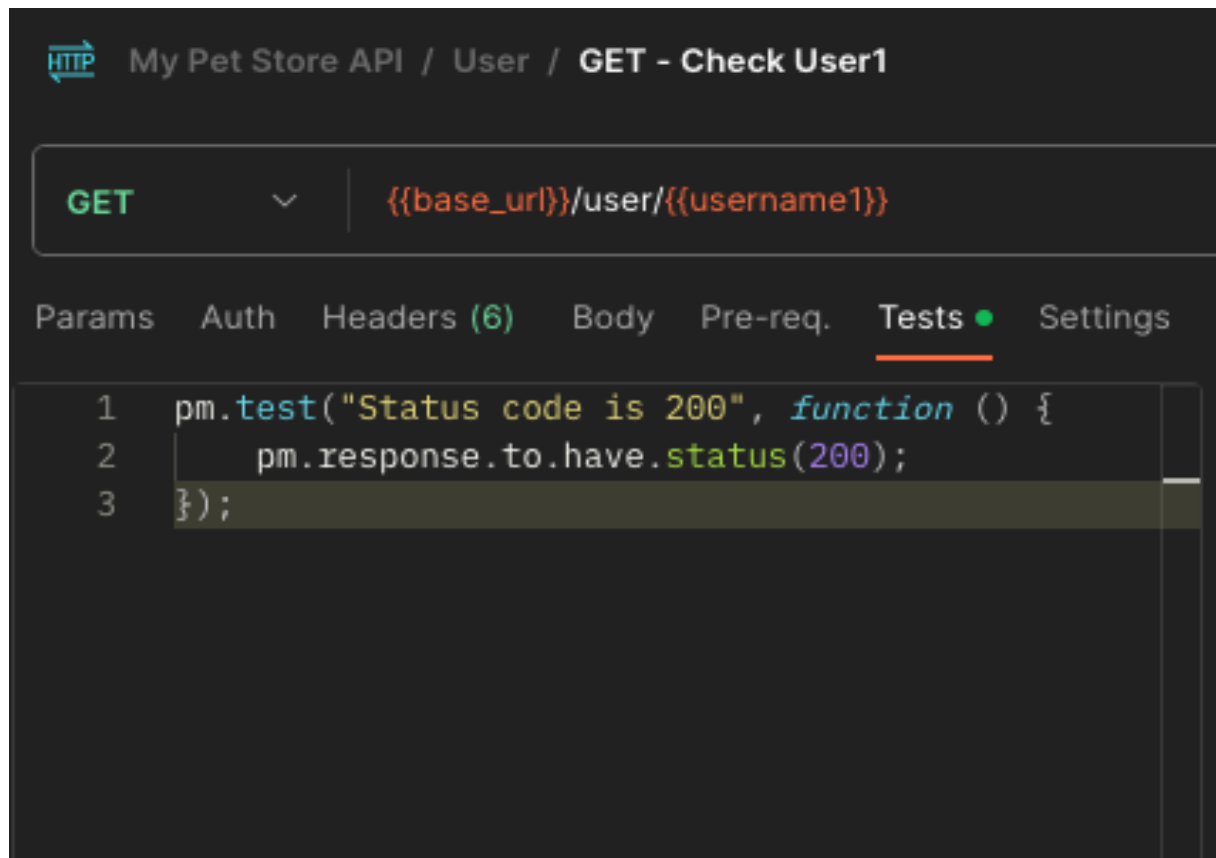
```
1  { 2    "code": 200, 3    "type": "unknown", 4    "message": "158" 5 }
```

4. GET – sprawdzenie, czy zmieniła się nazwa użytkownika

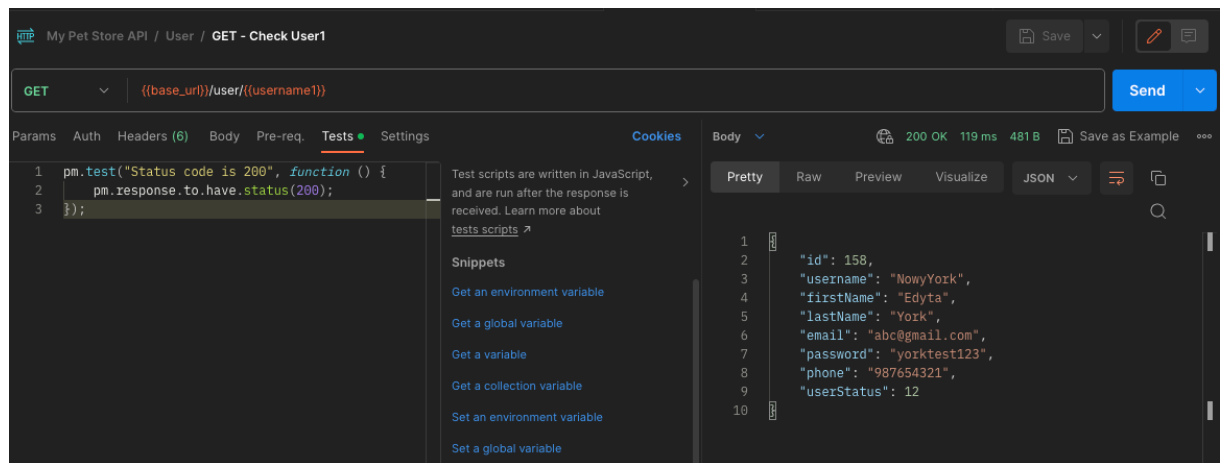
Sprawdzenie, czy zmieniła się nazwa użytkownika:



Wybrano test 200:

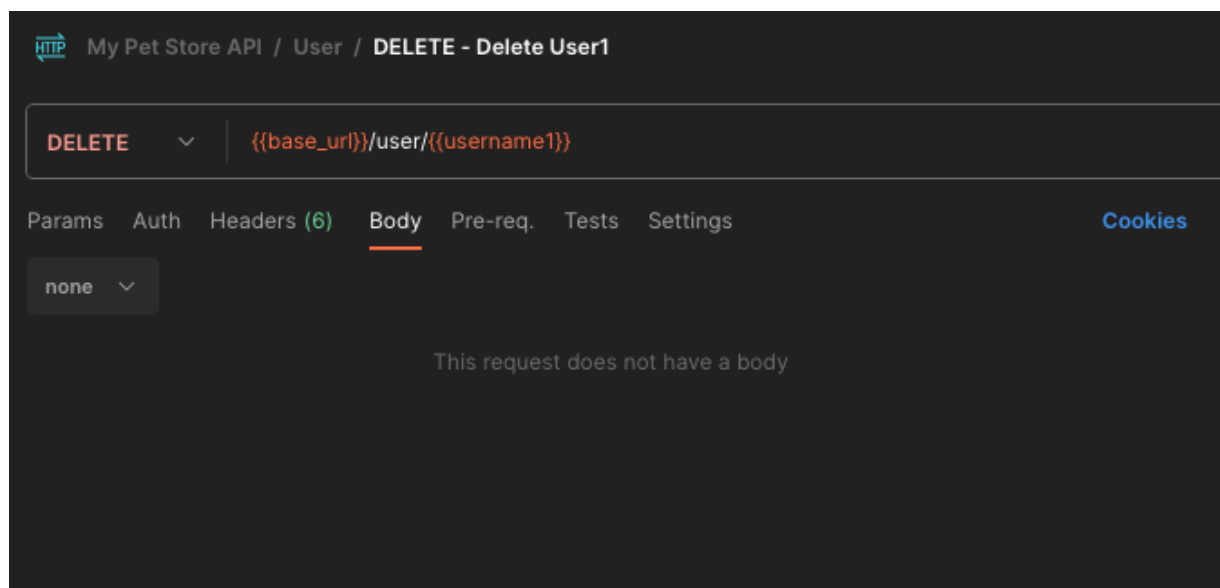


Nazwa użytkownika przebiegła pomyślnie o czym informuje nas status 200:

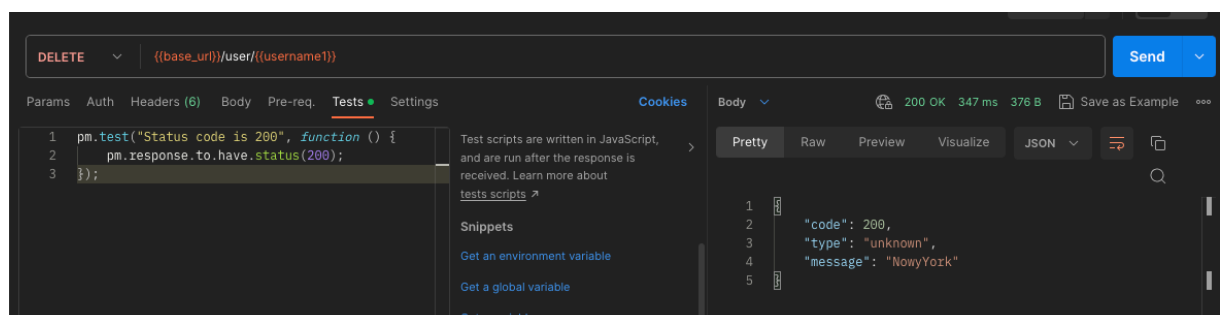


5. DELETE – usunięcie użytkownika, któremu zmieniono nazwę

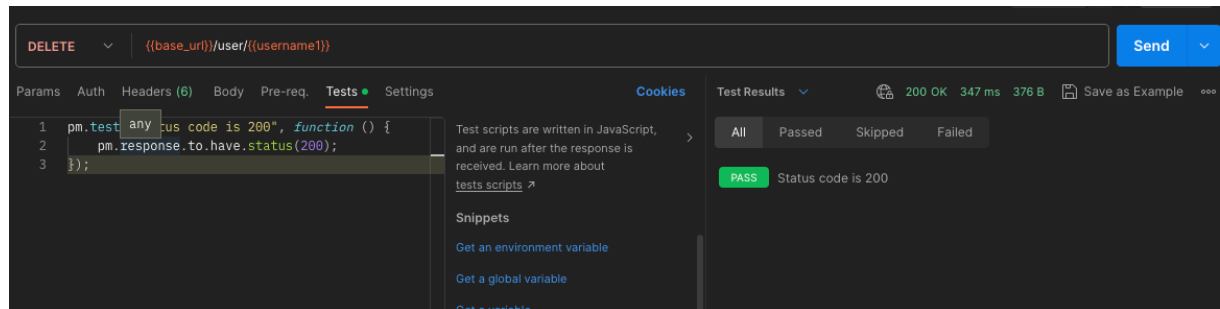
Usunięcie ostatnio modyfikowanego użytkownika przy pomocy DELETE:



Sprawdzenie, czy usunięto użytkownika przy pomocy testu „Status code is 200”:

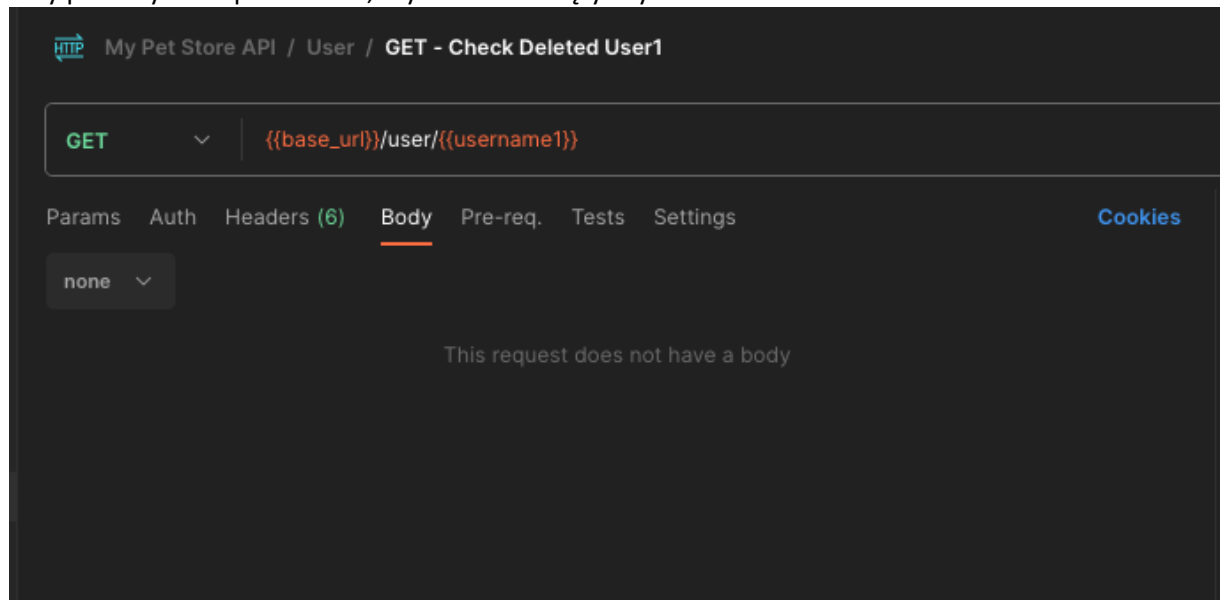


Otrzymano status 200 OK, co oznacza, że pomyślnie usunięto użytkownika:

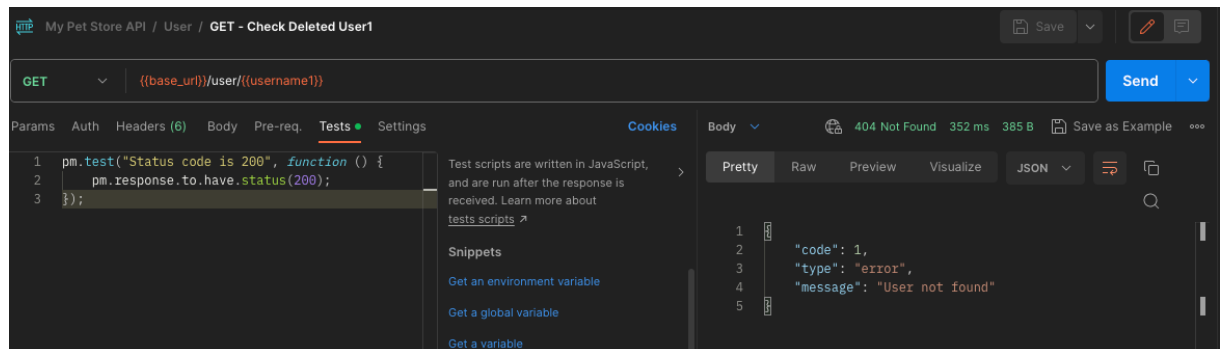


6. GET – sprawdzenie, czy usunięto użytkownika

Przy pomocy GET sprawdzono, czy został usunięty użytkownik:



Sprawdzenie, czy usunięto użytkownika przy pomocy testu „Status code is 200”:



Otrzymano status 404 Not Found, co oznacza, że użytkownika nie znaleziono i został on usunięty:

The screenshot shows the Postman interface for an API endpoint `GET {{base_url}}/user/{{username1}}`. The test script in the `Tests` tab is as follows:

```
1 pm.test("Status code is 200", function () {  
2   pm.response.to.have.status(200);  
3 });
```

The `Test Results` panel on the right indicates a failure with the message: `FAIL Status code is 200 | AssertionError: expected response to have status code 200 but got 404`. The status bar at the top of the results panel shows `404 Not Found`, `352 ms`, and `385 B`.