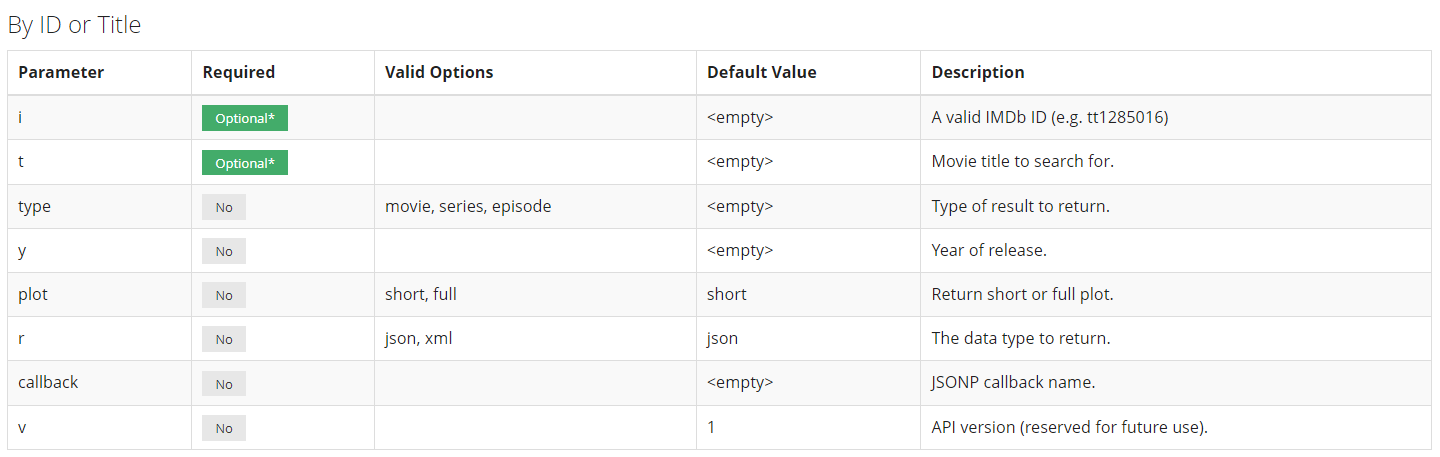
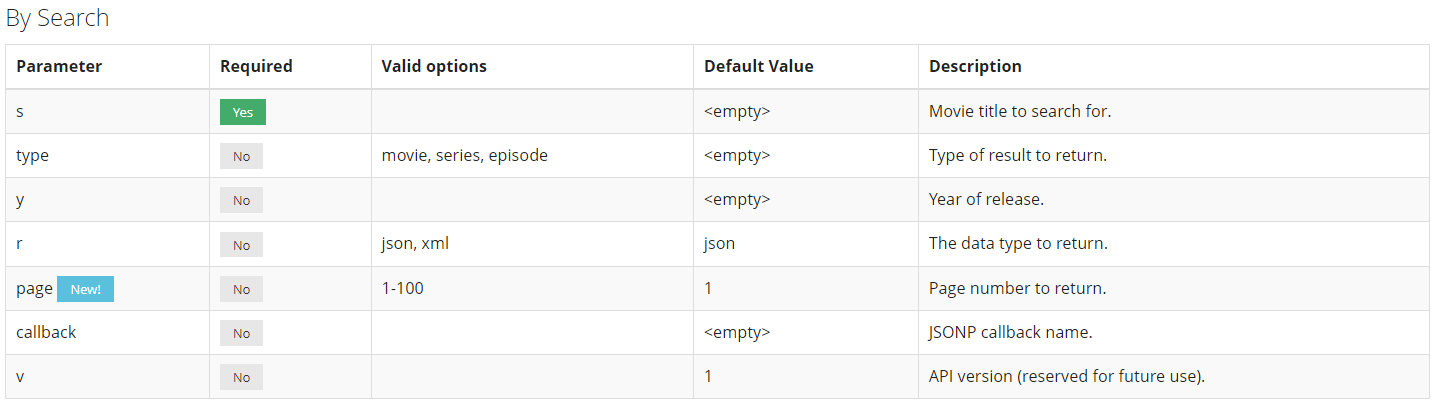
**Лабораторная работа №12. Тестирование API**

**Модульное тестирование:**

**Доступные методы:** GET, POST.



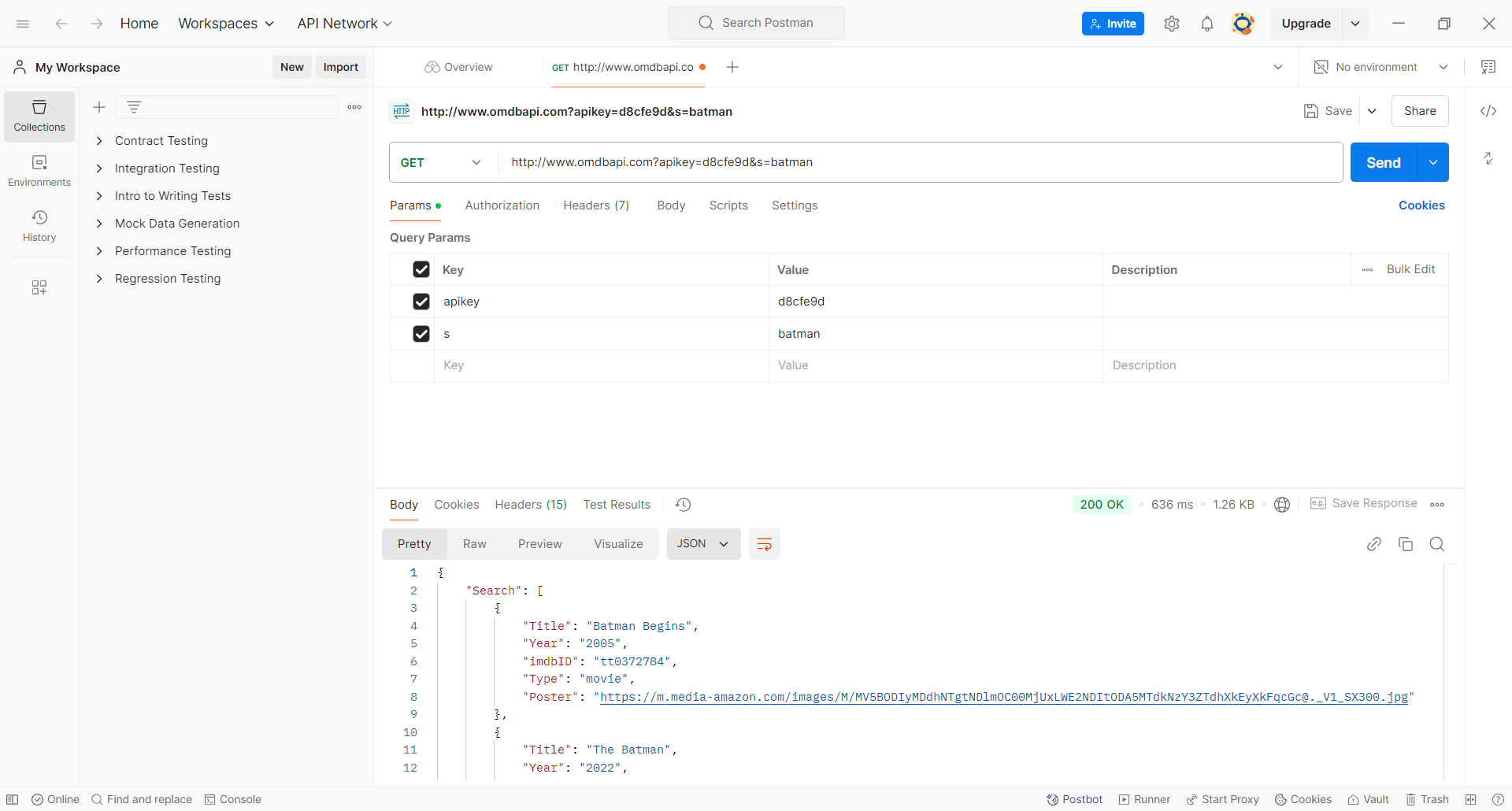


**Список тест-кейсов:**

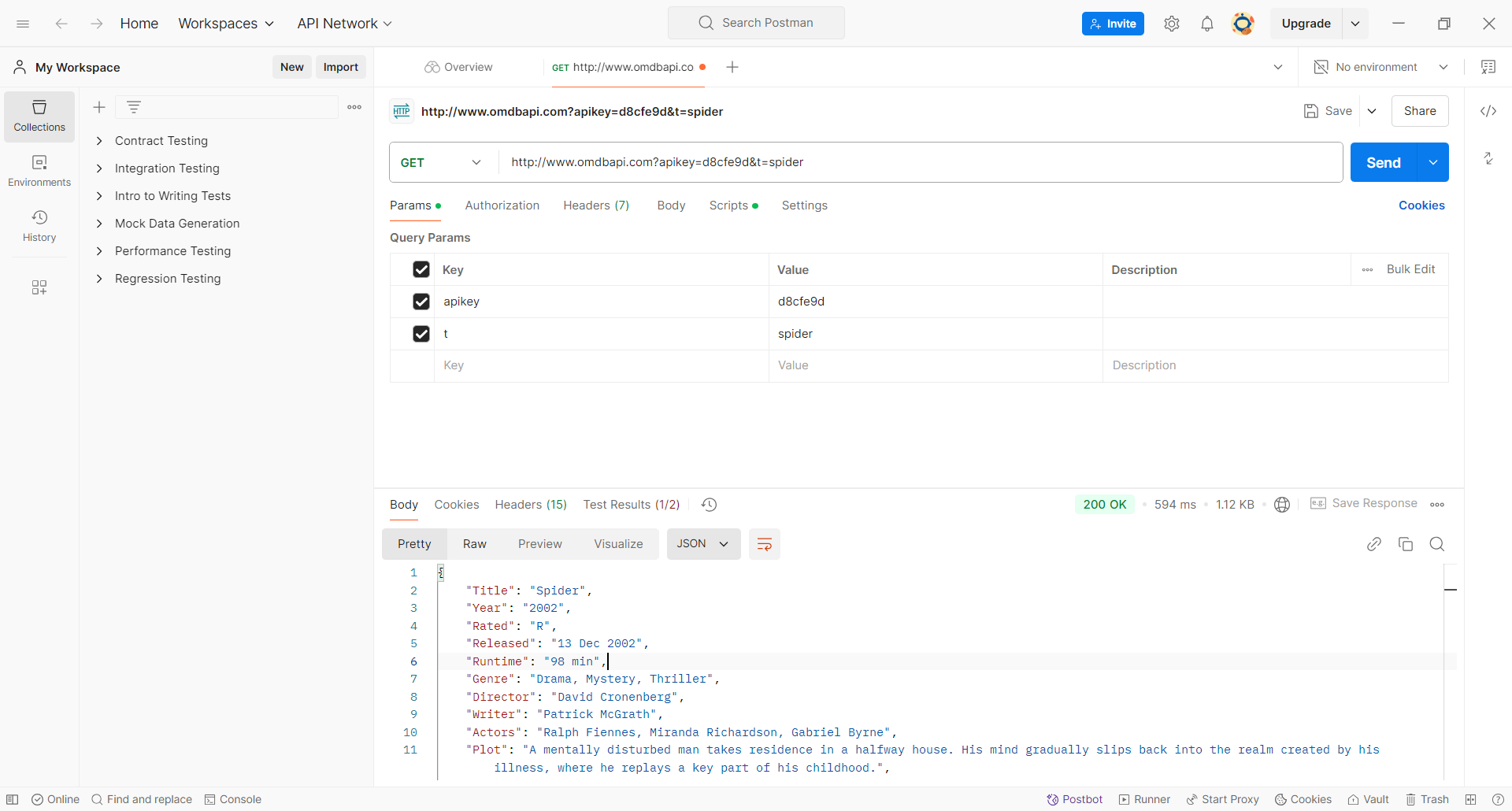
**Позитивные тесты для метода** GET**:**

* Корректный запрос с обязательными параметрами:

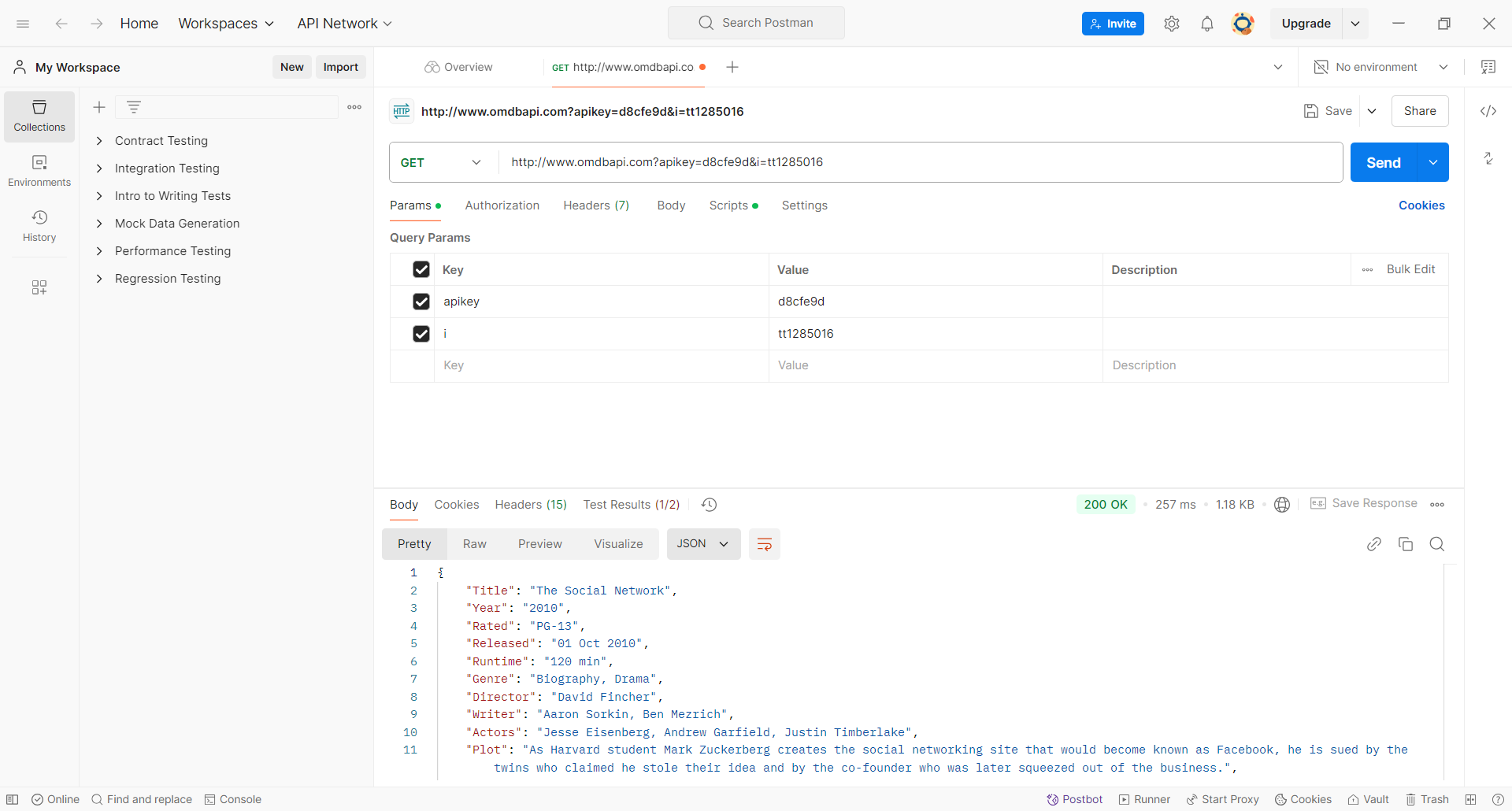
http://www.omdbapi.com?apikey=d8cfe9d&s=batman



http://www.omdbapi.com?apikey=d8cfe9d&t=spider

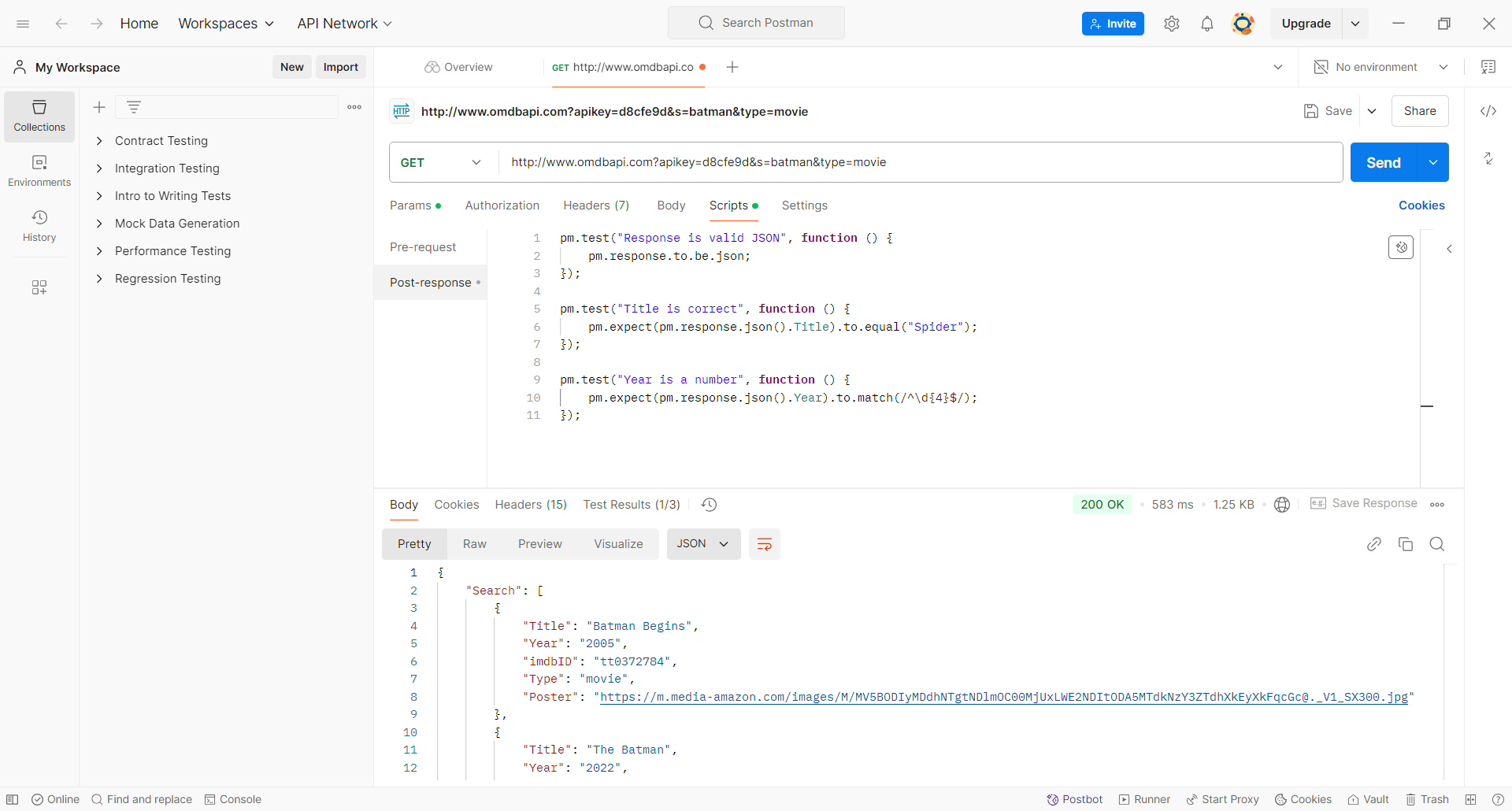


http://www.omdbapi.com?apikey=d8cfe9d&i=tt1285016

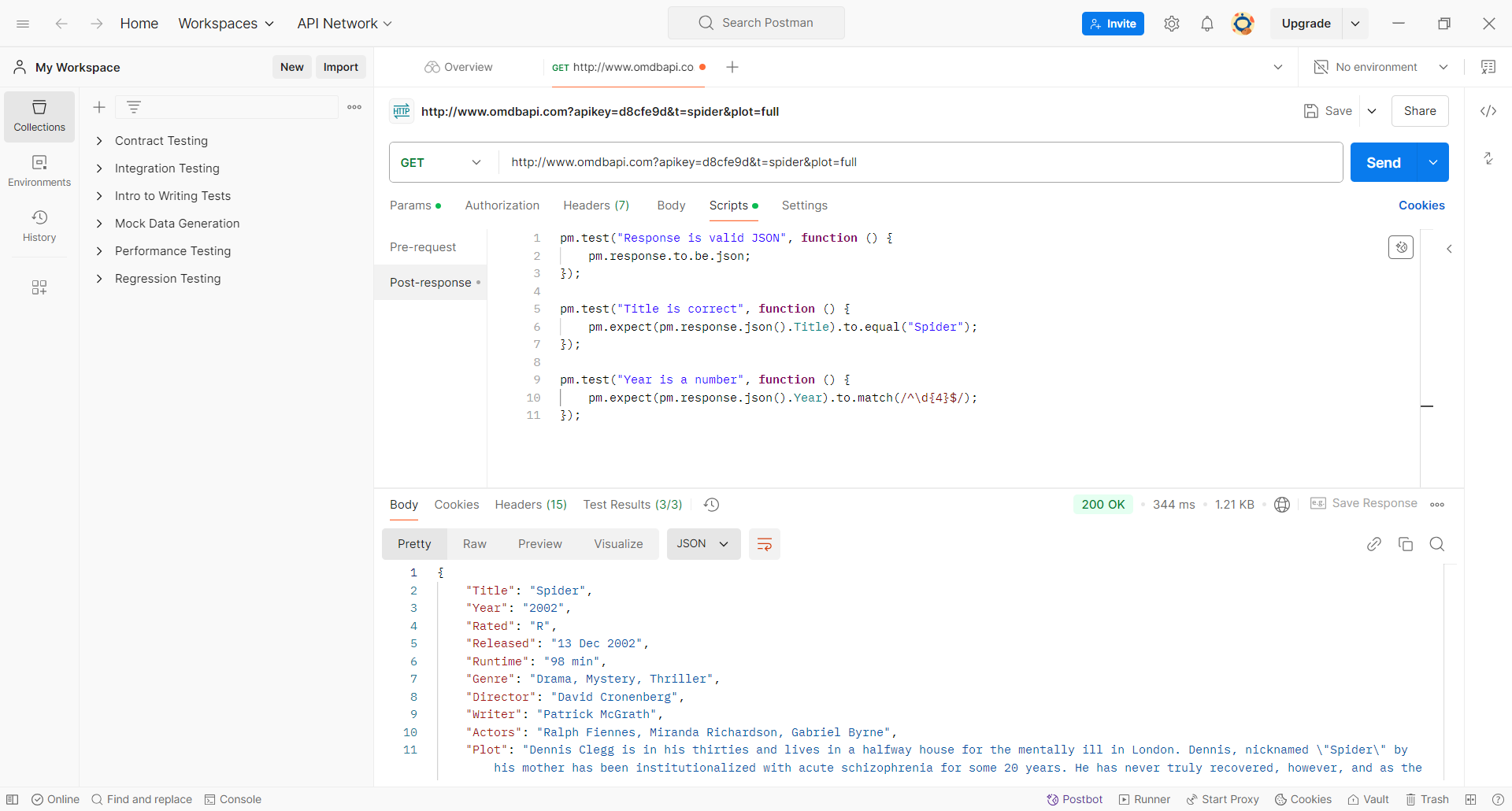


* Корректный запрос с обязательными и необязательными параметрами:

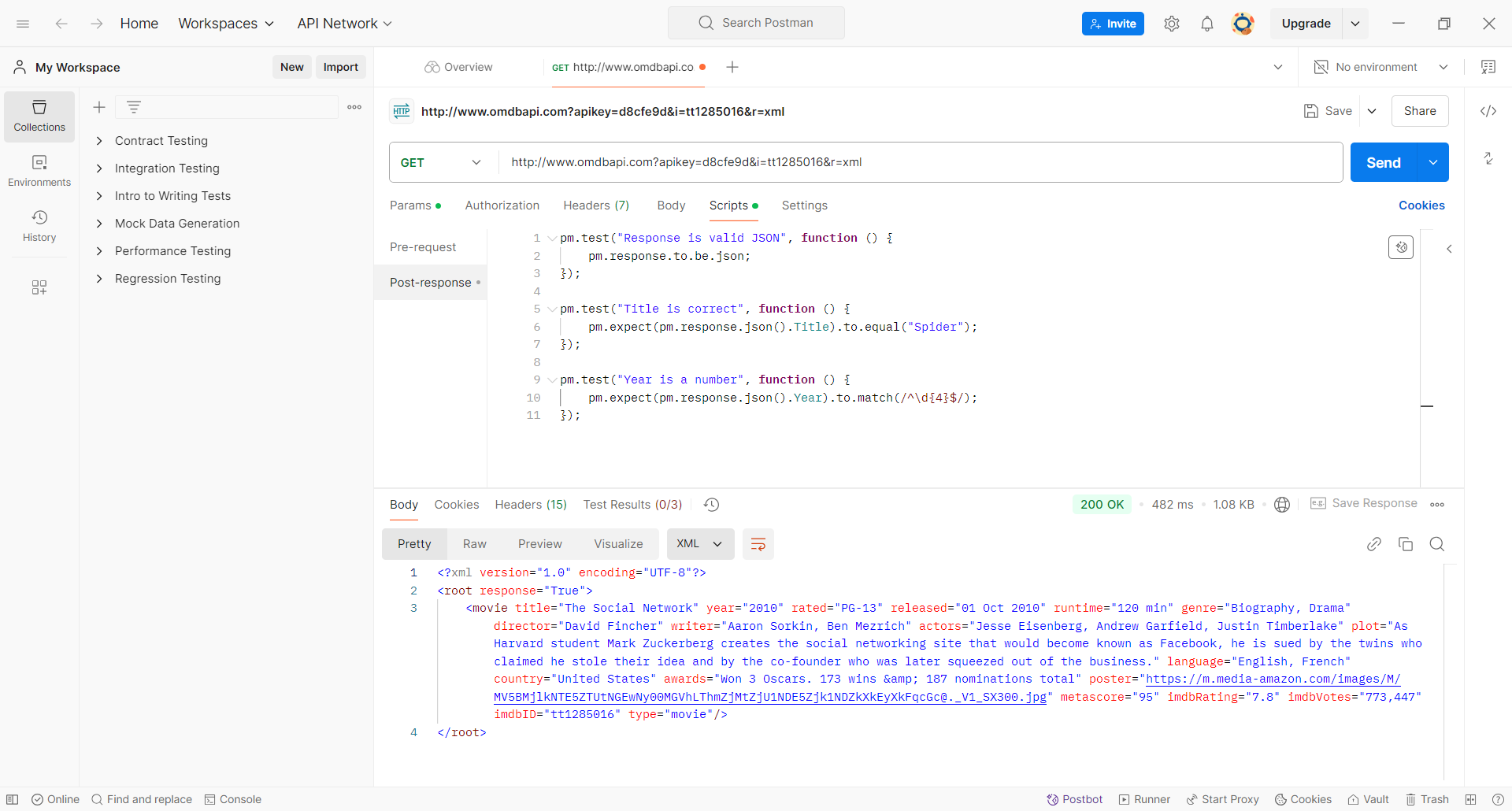
http://www.omdbapi.com?apikey=d8cfe9d&s=batman&type=movie



http://www.omdbapi.com?apikey=d8cfe9d&t=spider&plot=full



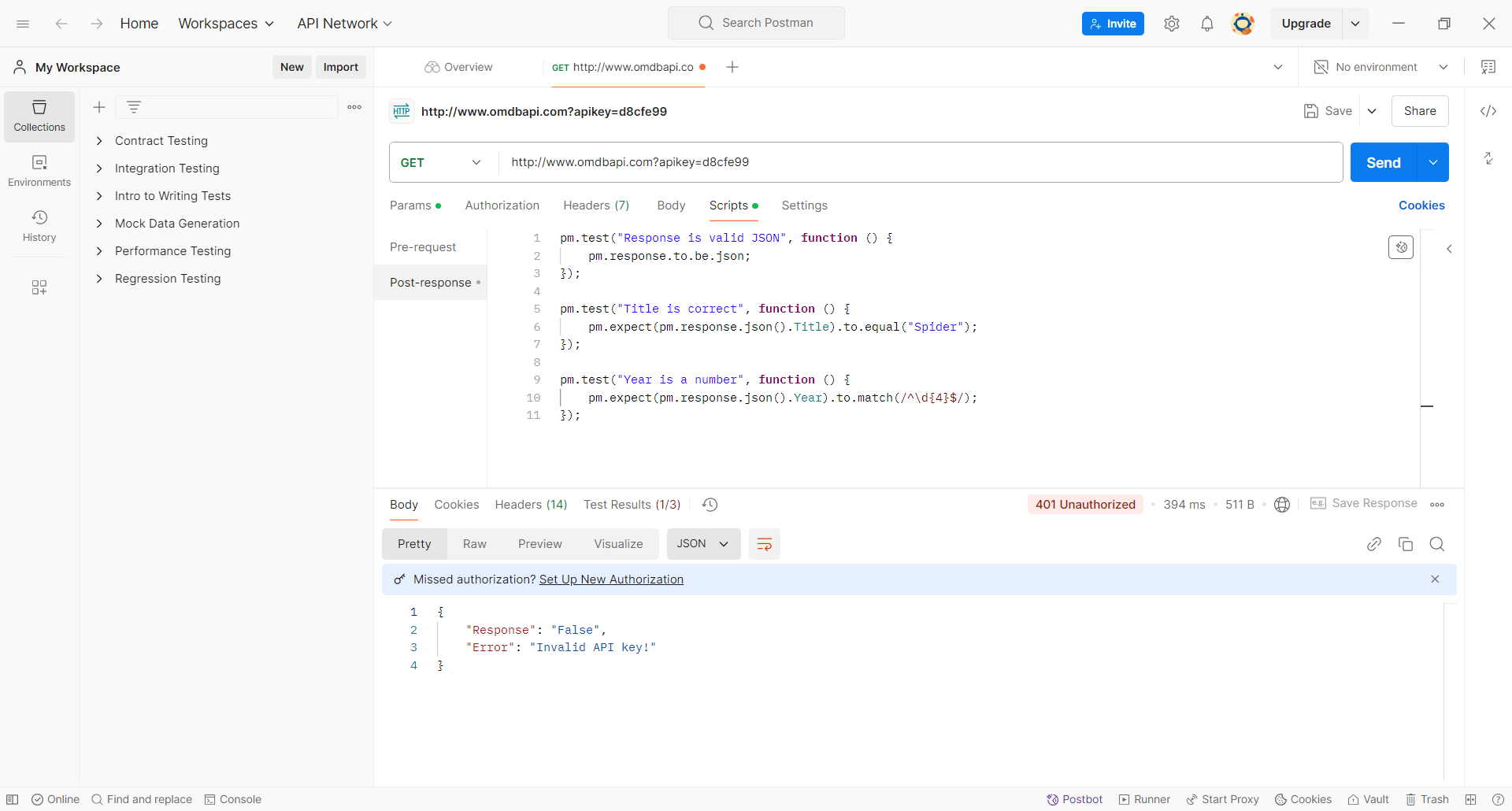
http://www.omdbapi.com?apikey=d8cfe9d&i=tt1285016&r=xml



**Негативные тесты для метода** GET**:**

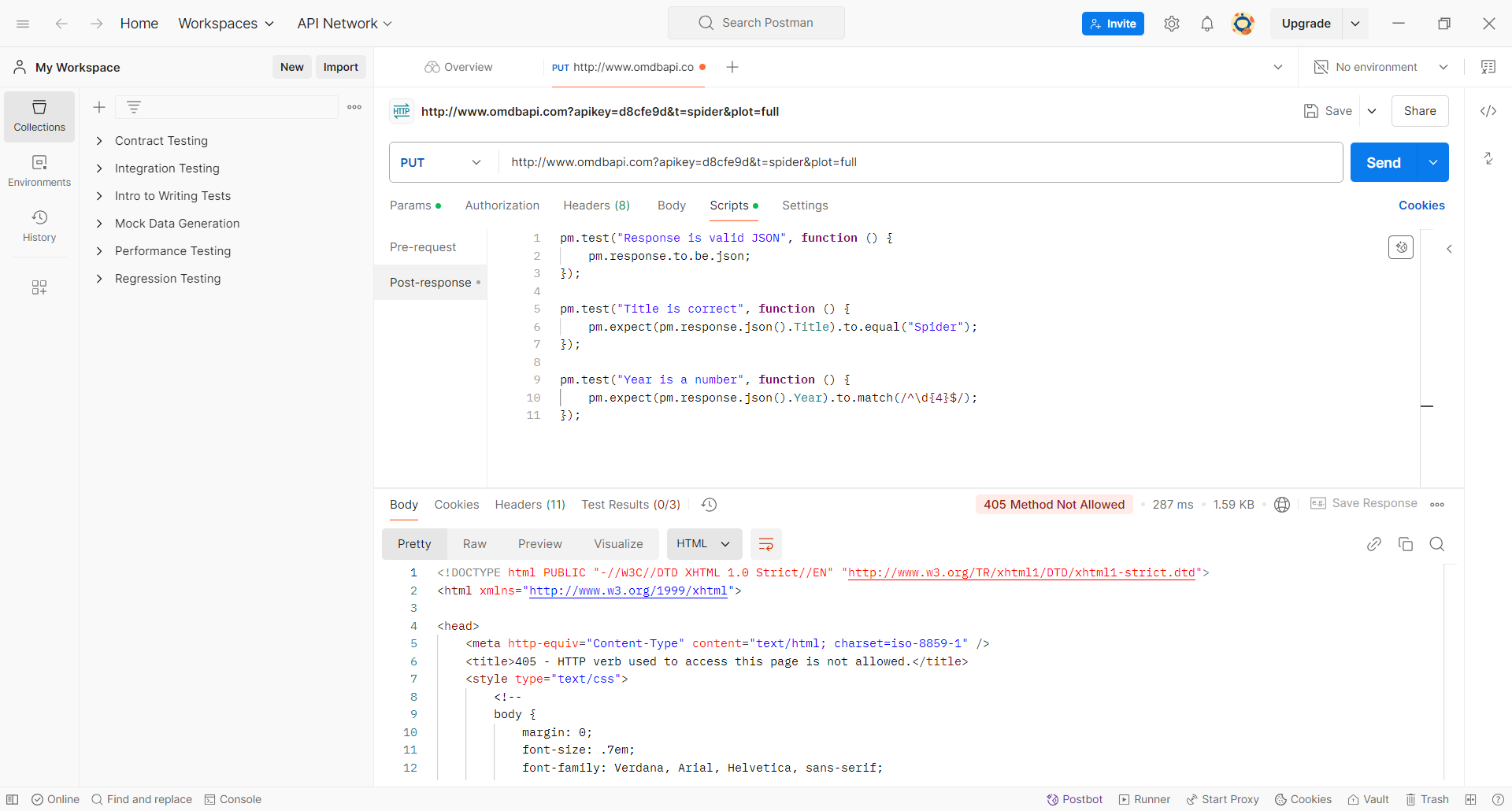
* Неверный ключ API:

http://www.omdbapi.com?apikey=d8cfe99



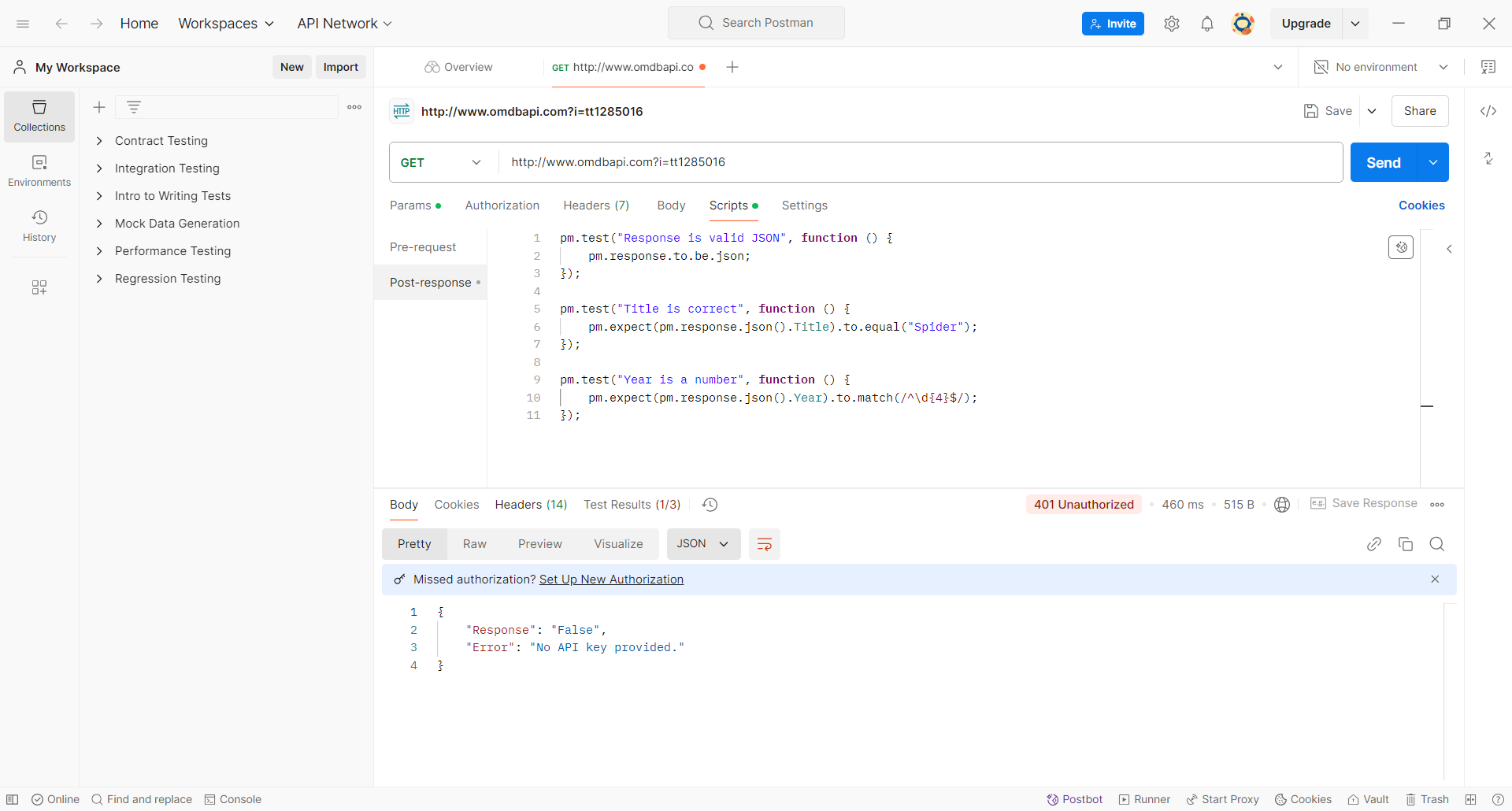
* Ошибка метода:

http://www.omdbapi.com?apikey=d8cfe9d&t=spider&plot=full (PUT)



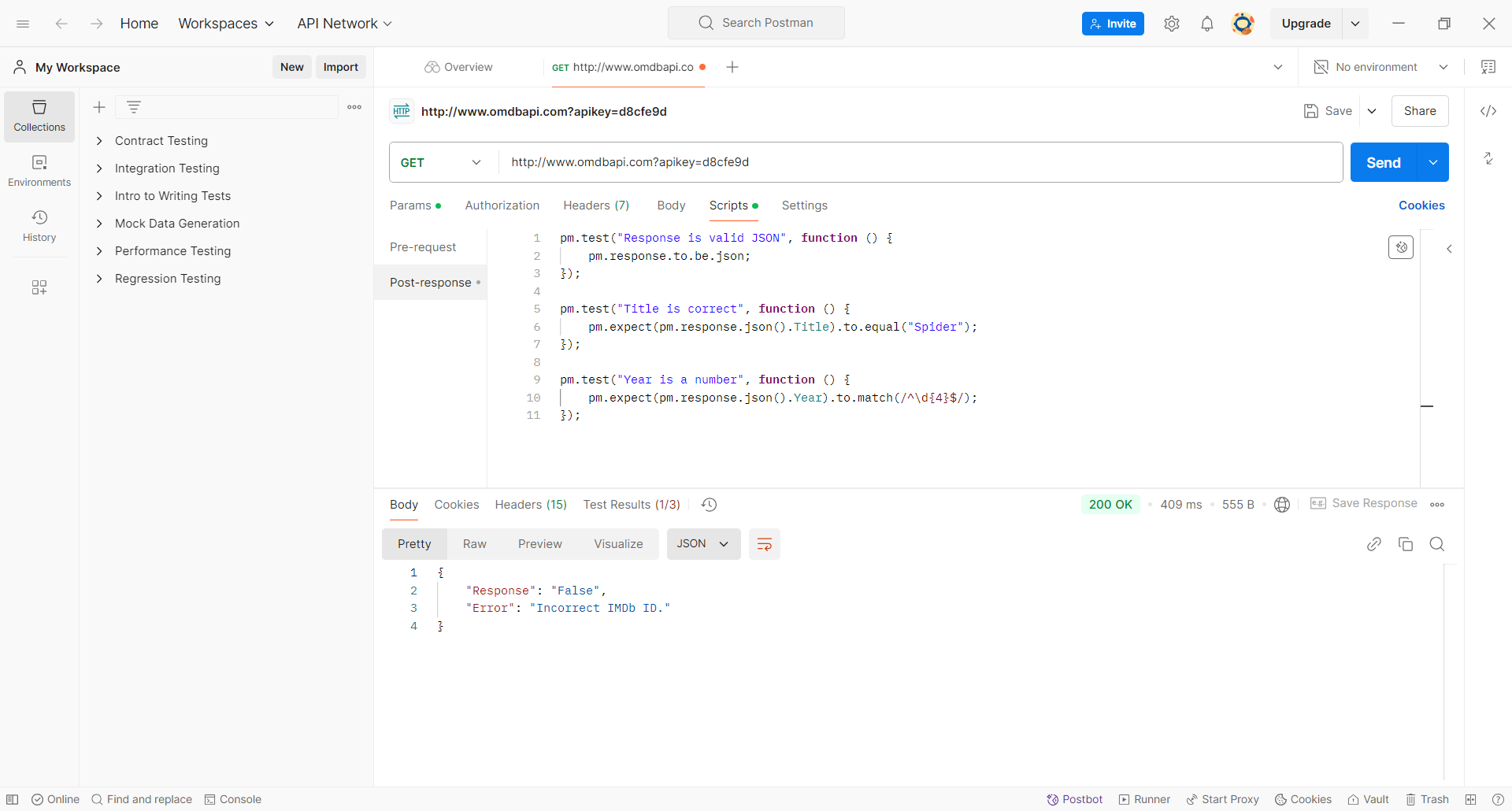
* Отсутствие ключа API:

http://www.omdbapi.com?i=tt1285016



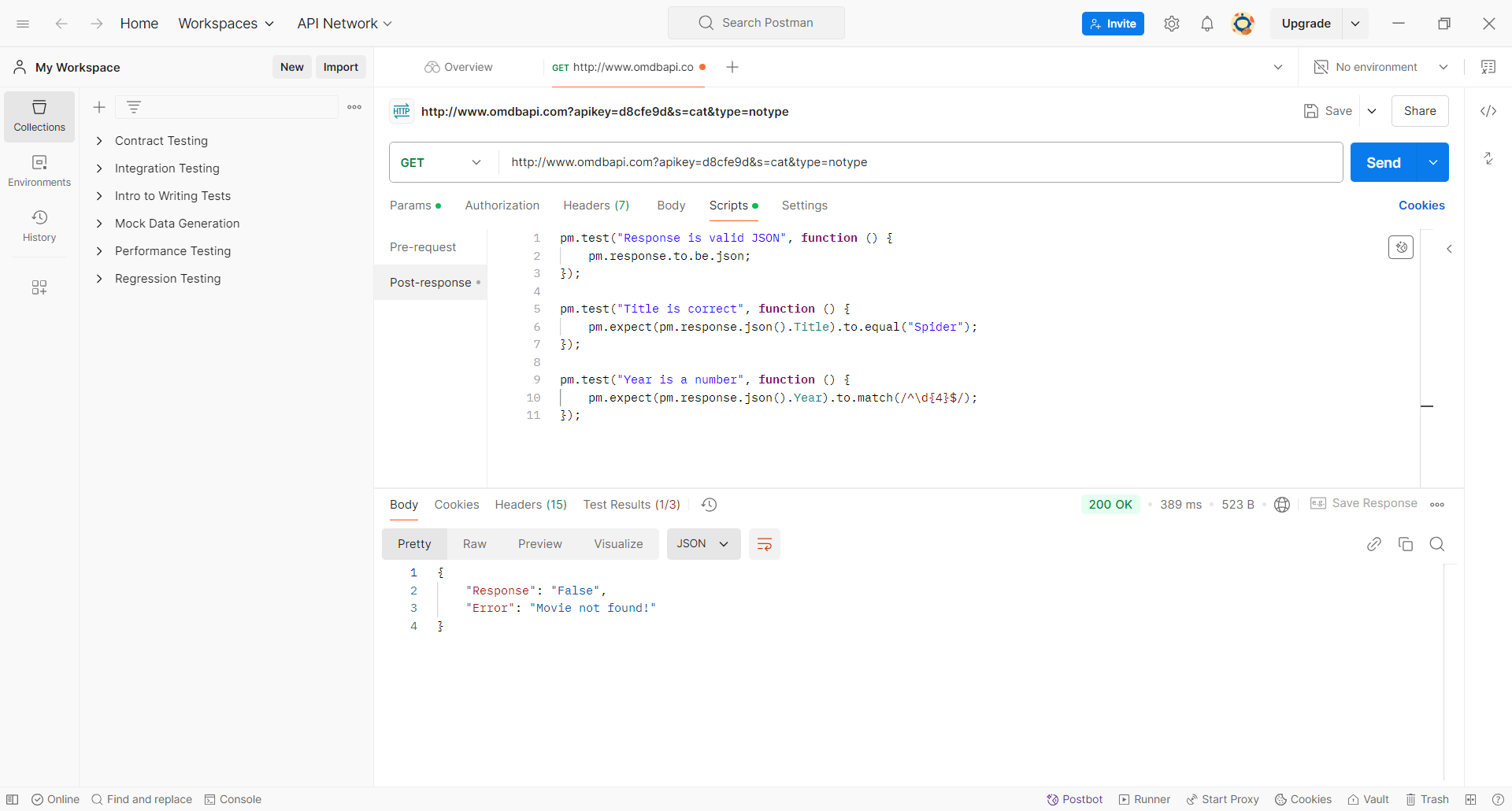
* Отсутствие обязательного параметра (s, t или i):

http://www.omdbapi.com?apikey=d8cfe9d



* Некорректное значение параметра:

http://www.omdbapi.com?apikey=d8cfe9d&s=cat&type=notype



**Автоматизированные тесты:**

**Позитивные тесты для метода** GET**:**

* Корректный запрос с обязательными параметрами:

http://www.omdbapi.com?apikey=d8cfe9d&s=batman

http://www.omdbapi.com?apikey=d8cfe9d&t=spider

http://www.omdbapi.com?apikey=d8cfe9d&i=tt1285016

pm.test("Response is valid JSON", function () {

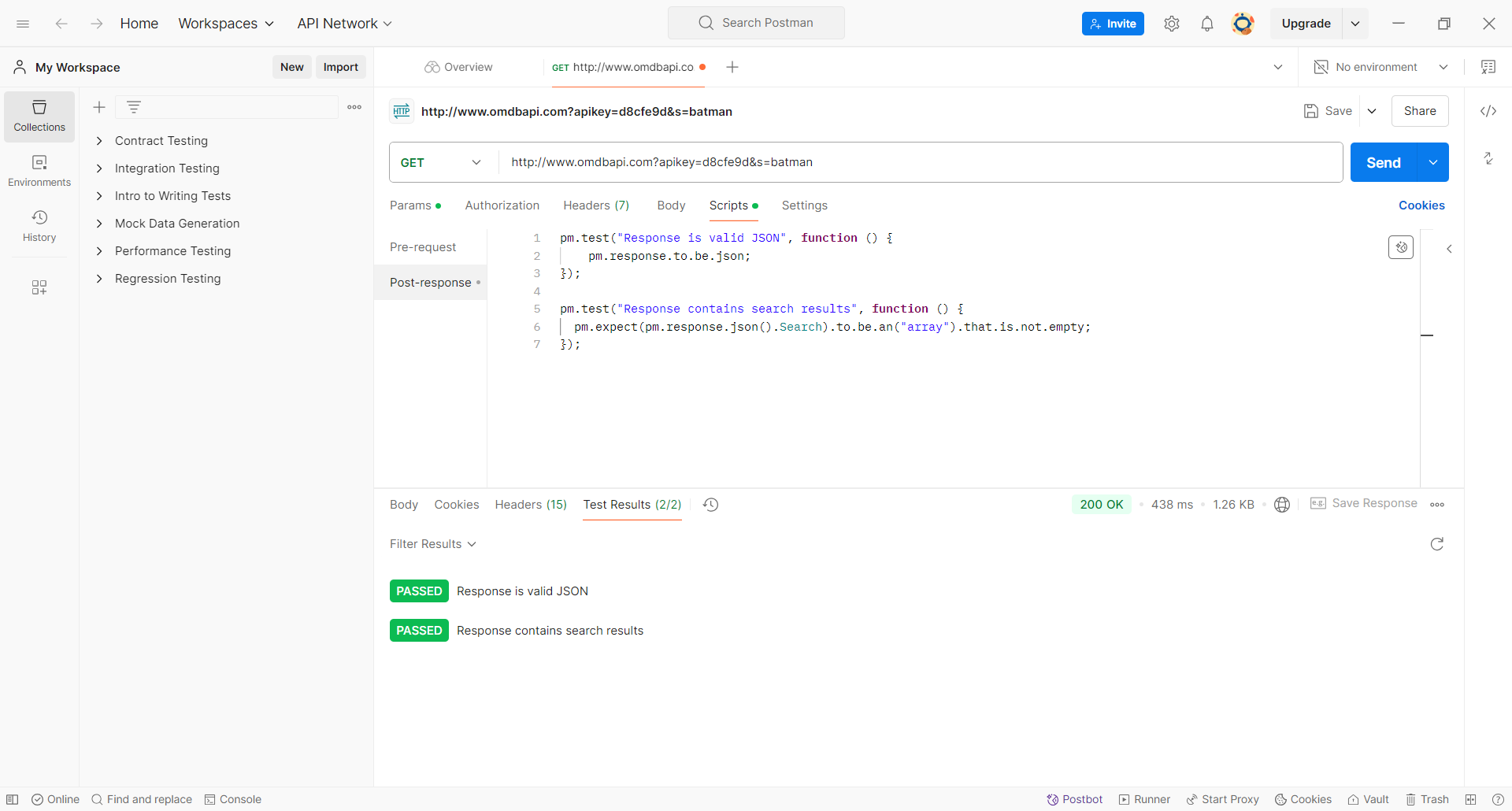
pm.response.to.be.json;

});

pm.test("Response contains search results", function () {

pm.expect(pm.response.json().Search).to.be.an("array").that.is.not.empty;

});



* Корректный запрос с обязательными и необязательными параметрами:

http://www.omdbapi.com?apikey=d8cfe9d&s=batman&type=movie

http://www.omdbapi.com?apikey=d8cfe9d&t=spider&plot=full

http://www.omdbapi.com?apikey=d8cfe9d&i=tt1285016&r=xml

pm.test("Response is valid JSON", function () {

pm.response.to.be.json;

});

pm.test("Title is correct", function () {

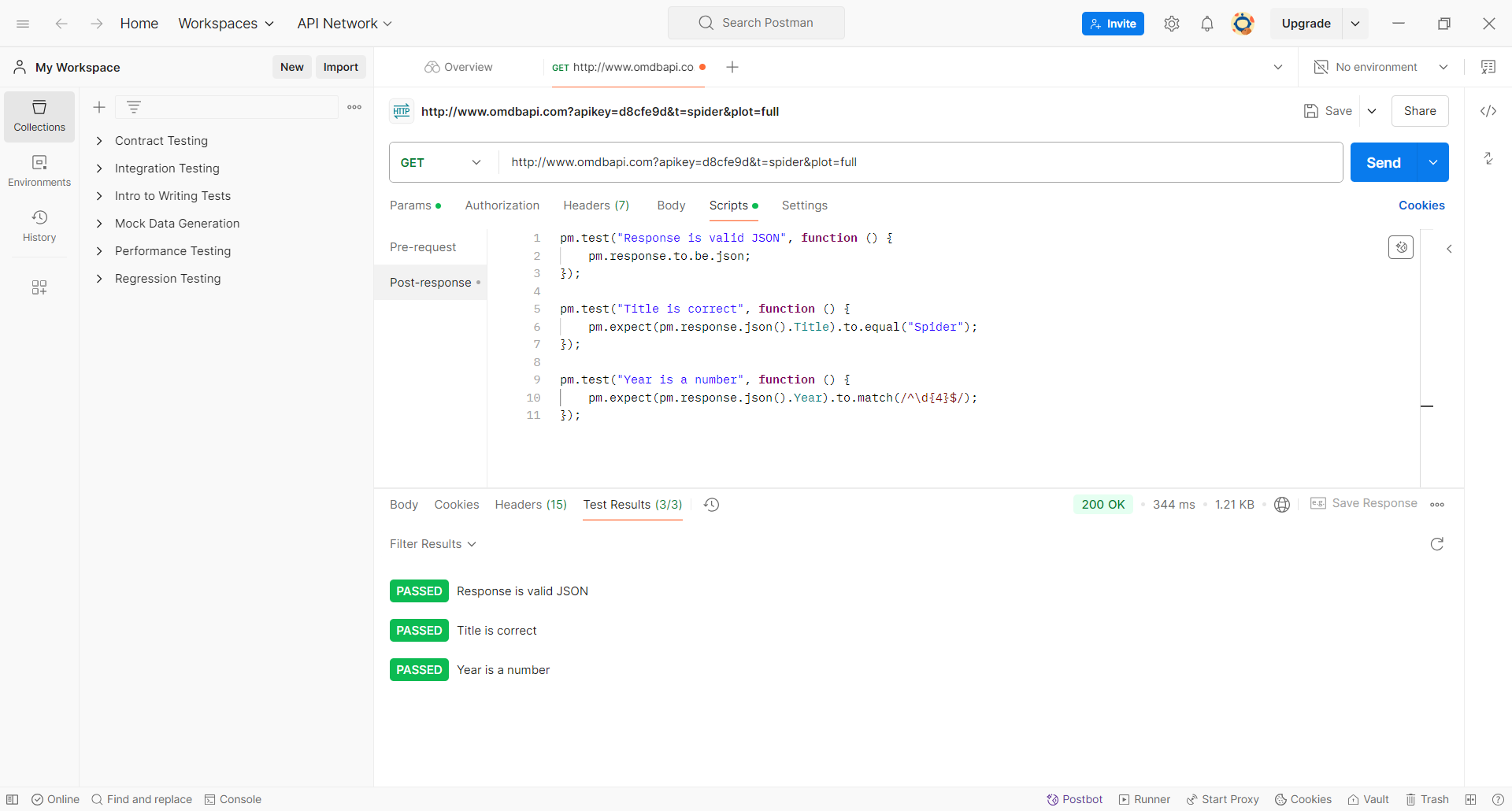
pm.expect(pm.response.json().Title).to.equal("Spider");

});

pm.test("Year is a number", function () {

pm.expect(pm.response.json().Year).to.match(/^\d{4}$/);

});



**Негативные тесты для метода** GET**:**

* Неверный ключ API:

http://www.omdbapi.com?apikey=d8cfe99

pm.test("Status code is 401", function () {

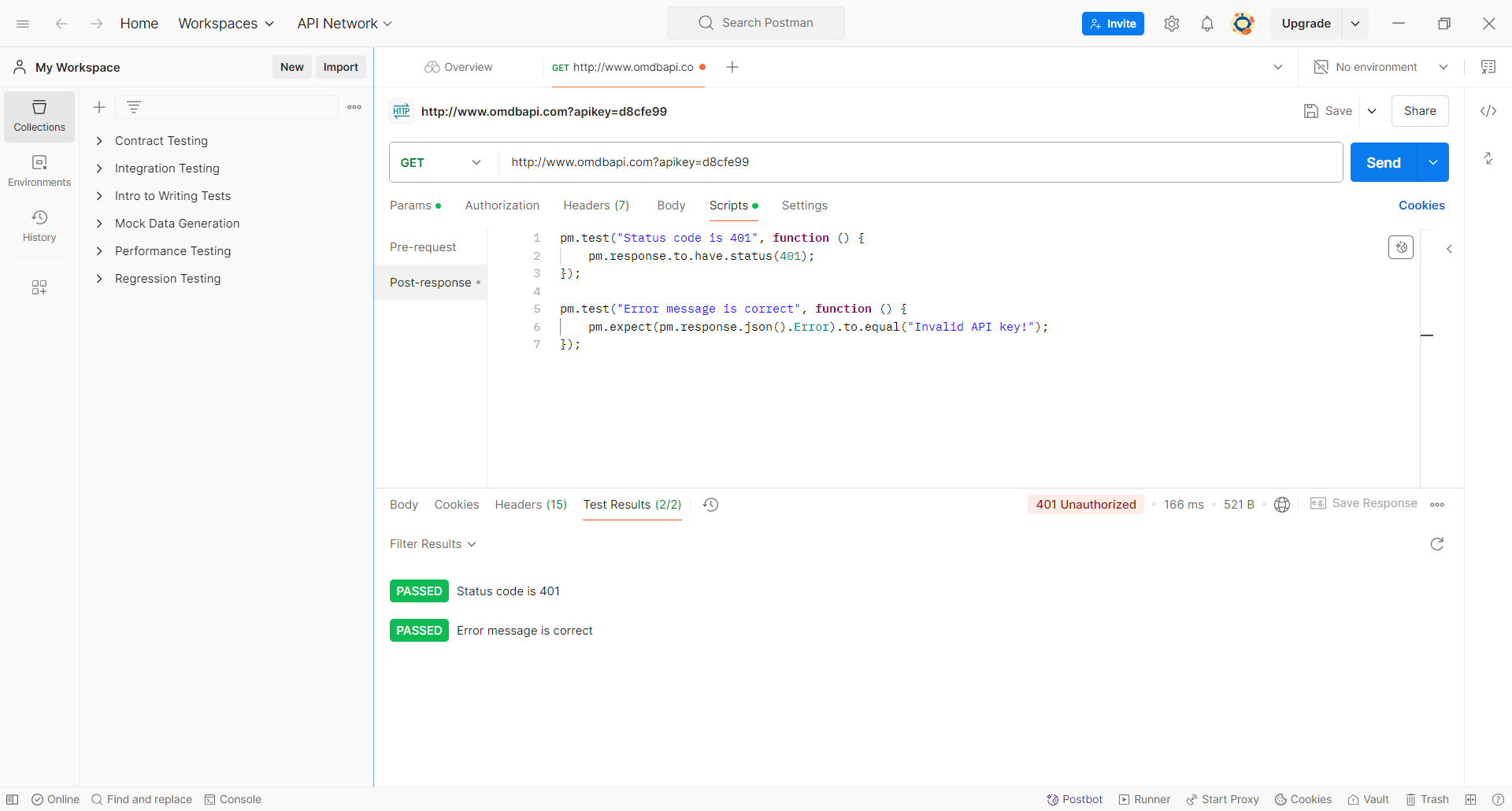
pm.response.to.have.status(401);

});

pm.test("Error message is correct", function () {

pm.expect(pm.response.json().Error).to.equal("Invalid API key!");

});



* Ошибка метода:

http://www.omdbapi.com?apikey=d8cfe9d&t=spider&plot=full (PUT)

pm.test("Status code is 405", function () {

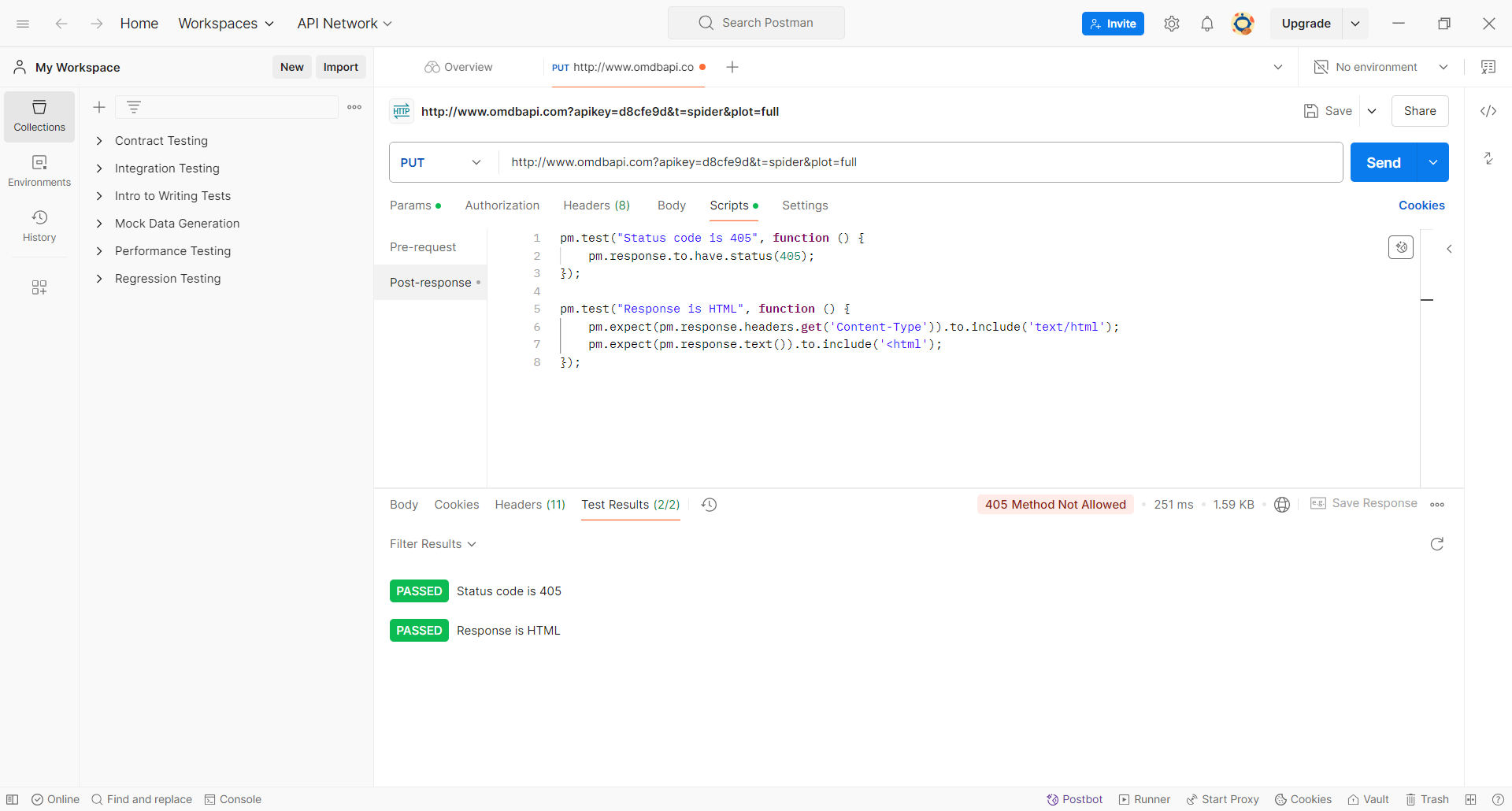
pm.response.to.have.status(405);

});

pm.test("Error message is correct", function () {

pm.expect(pm.response.json().Error).to.equal("Missing API key!");

});



* Отсутствие ключа API:

http://www.omdbapi.com?i=tt1285016

pm.test("Status code is 401", function () {

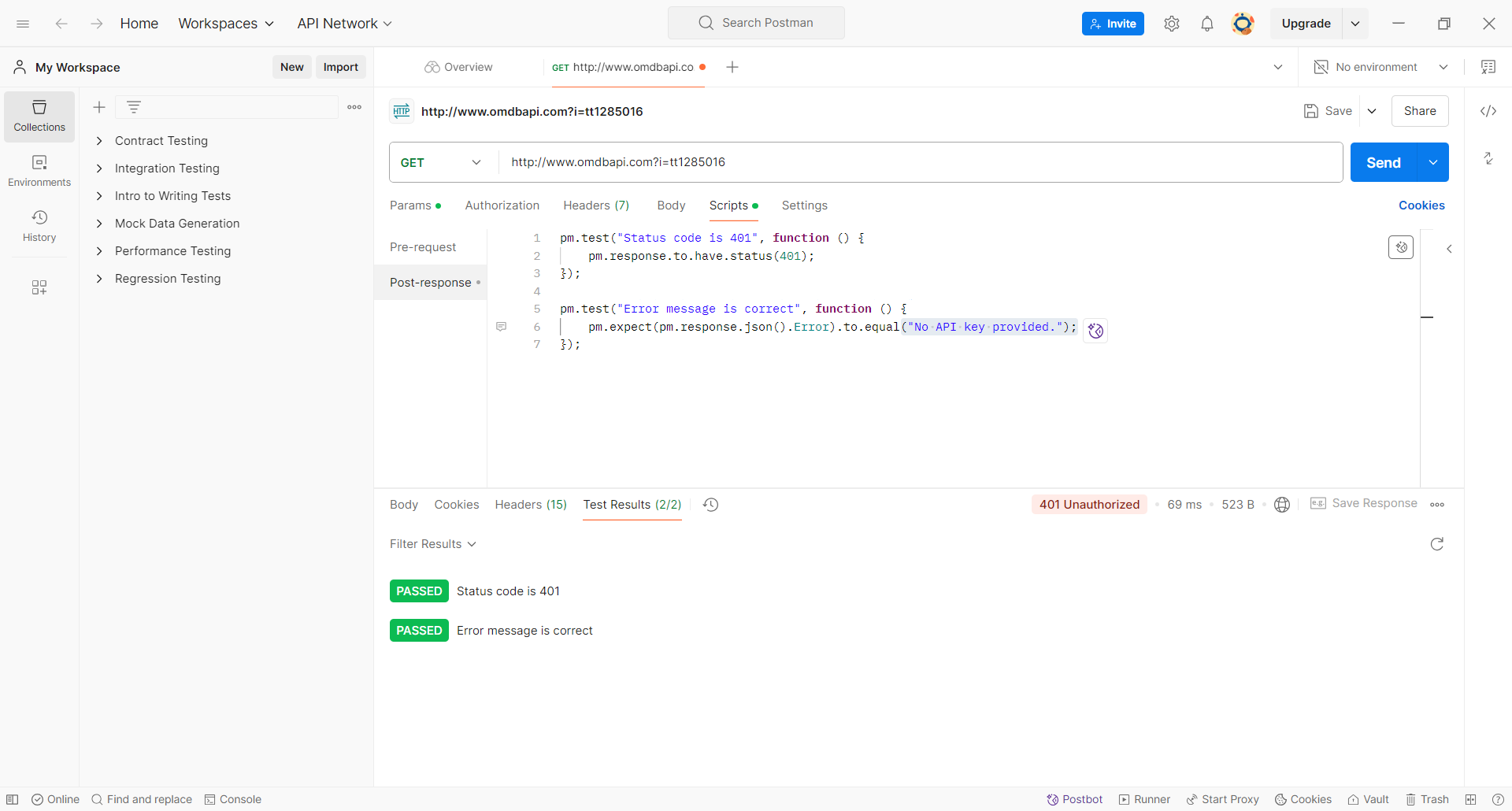
pm.response.to.have.status(401);

});

pm.test("Error message is correct", function () {

pm.expect(pm.response.json().Error).to.equal("No API key provided.");

});



* Отсутствие обязательного параметра (s, t или i):

http://www.omdbapi.com?apikey=d8cfe9d

pm.test("Status code is 200", function () {

pm.response.to.have.status(200);

});

pm.test("Response is false", function () {

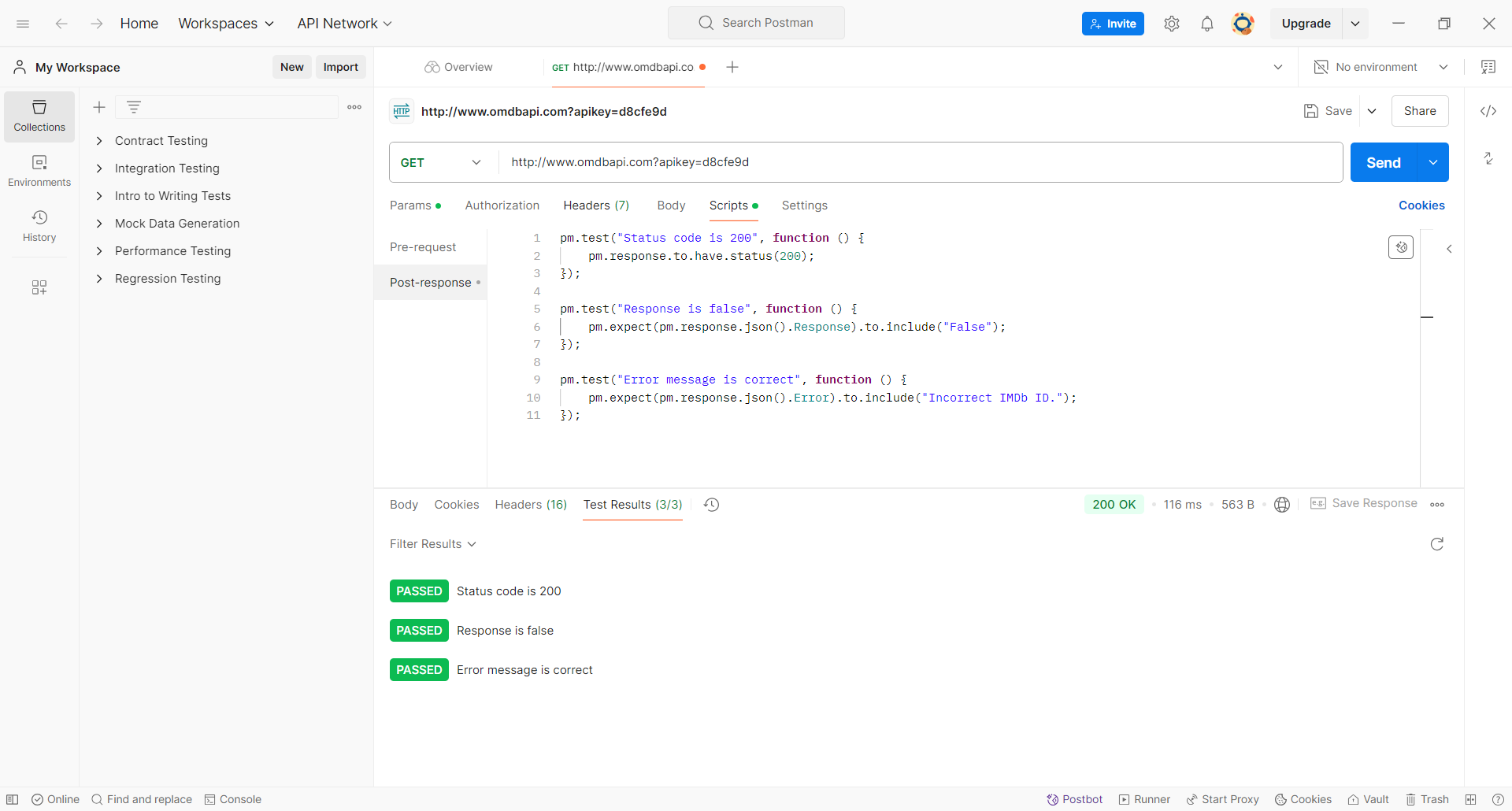
pm.expect(pm.response.json().Response).to.include("False");

});

pm.test("Error message is correct", function () {

pm.expect(pm.response.json().Error).to.include("Incorrect IMDb ID.");

});



* Некорректное значение параметра:

http://www.omdbapi.com?apikey=d8cfe9d&s=cat&type=notype

pm.test("Status code is 200", function () {

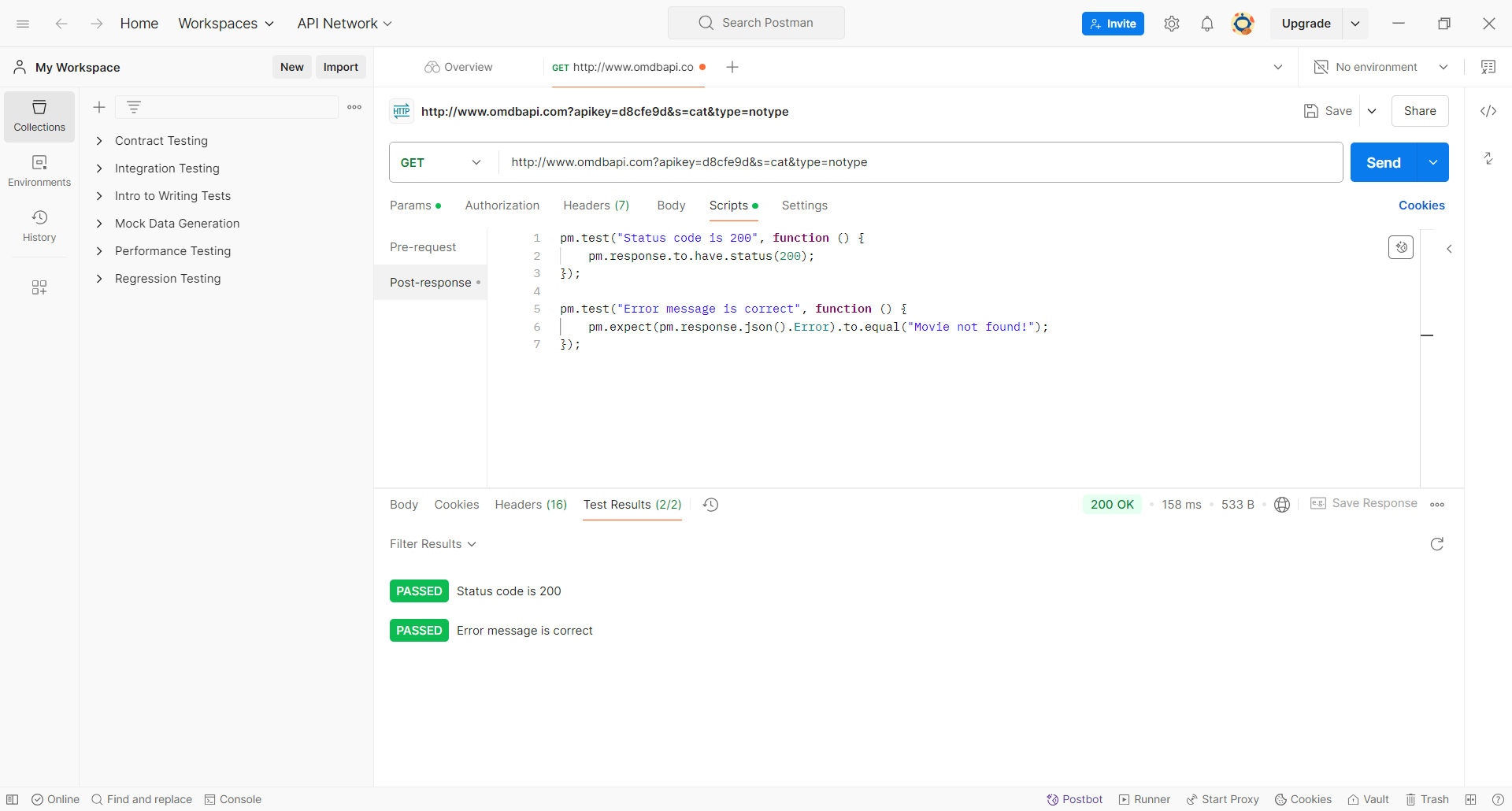
pm.response.to.have.status(200);

});

pm.test("Error message is correct", function () {

pm.expect(pm.response.json().Error).to.equal("Movie not found!");

});



**Интеграционное тестирование:**

**Тесты API для управления пользователями:**

* Создание нового пользователя:

https://reqres.in/api/users (POST)

Тело:

{ "username": "user1", "email": "user1@gmail.com", "password": "password1" }

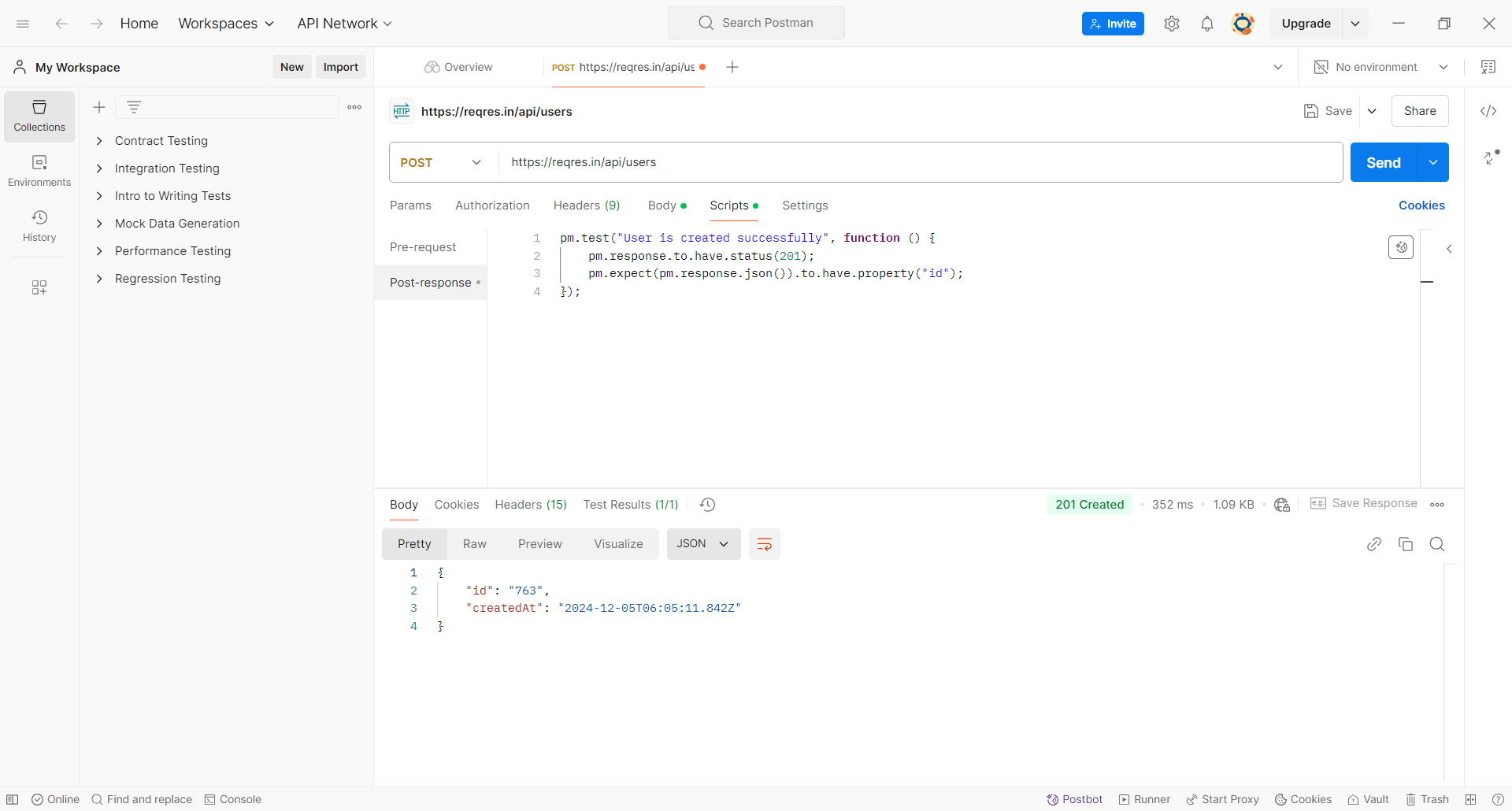
Тест:

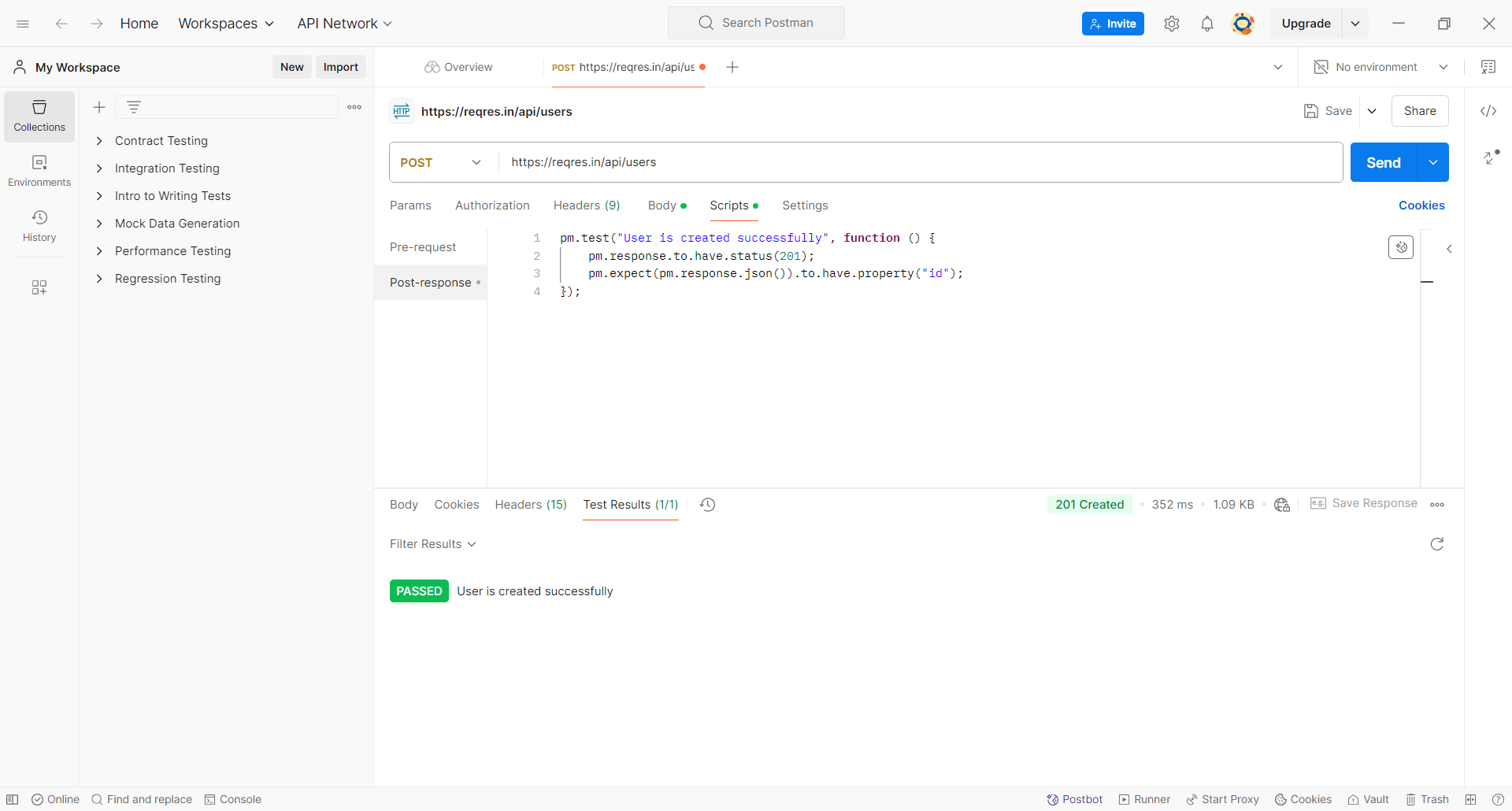
pm.test("User is created successfully", function () {

pm.response.to.have.status(201);

pm.expect(pm.response.json()).to.have.property("id");

});





* Получение информации о пользователе:

https://reqres.in/api/users/1 (GET)

Тест:

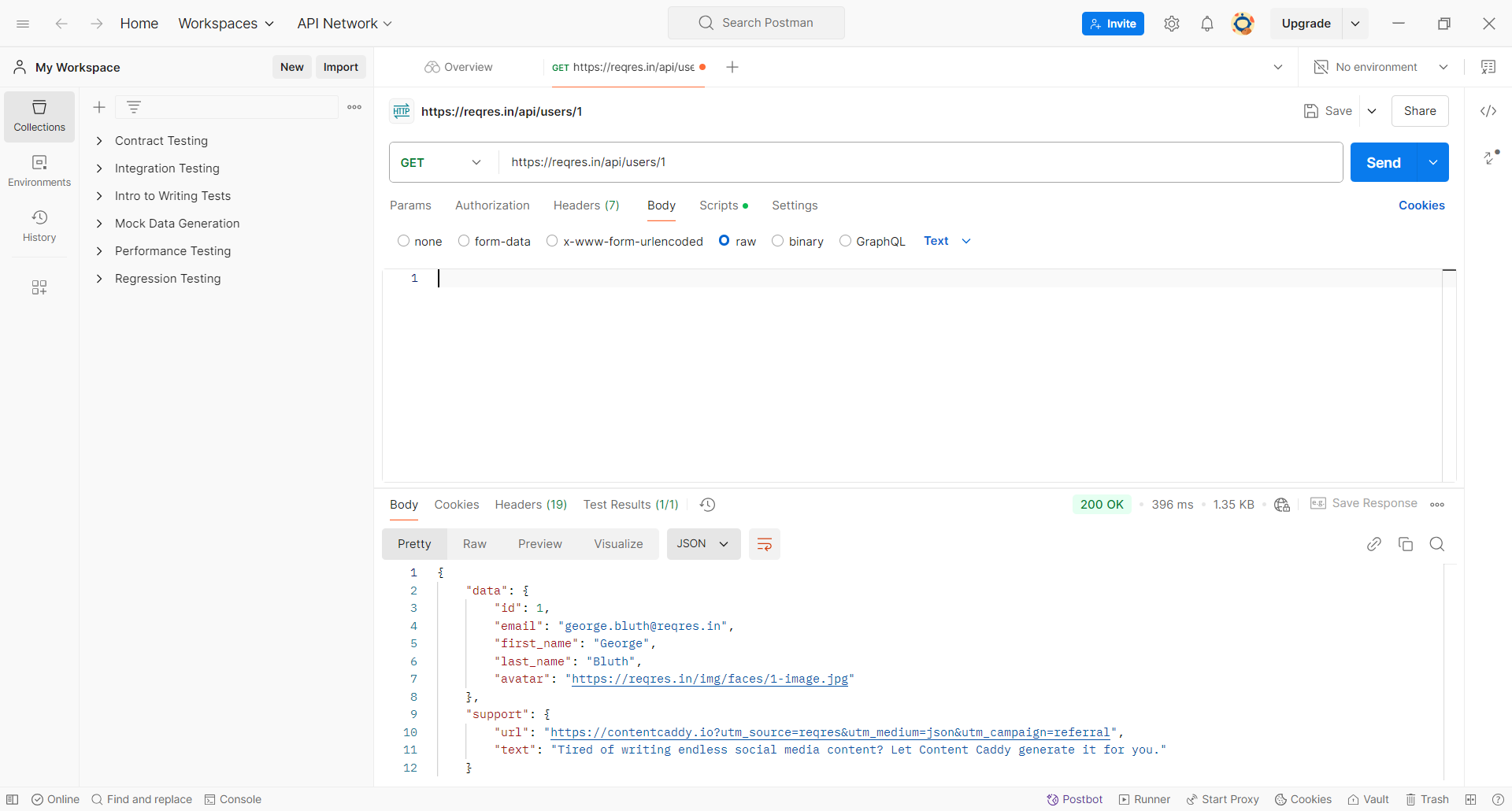
pm.test("User information is correct", function () {

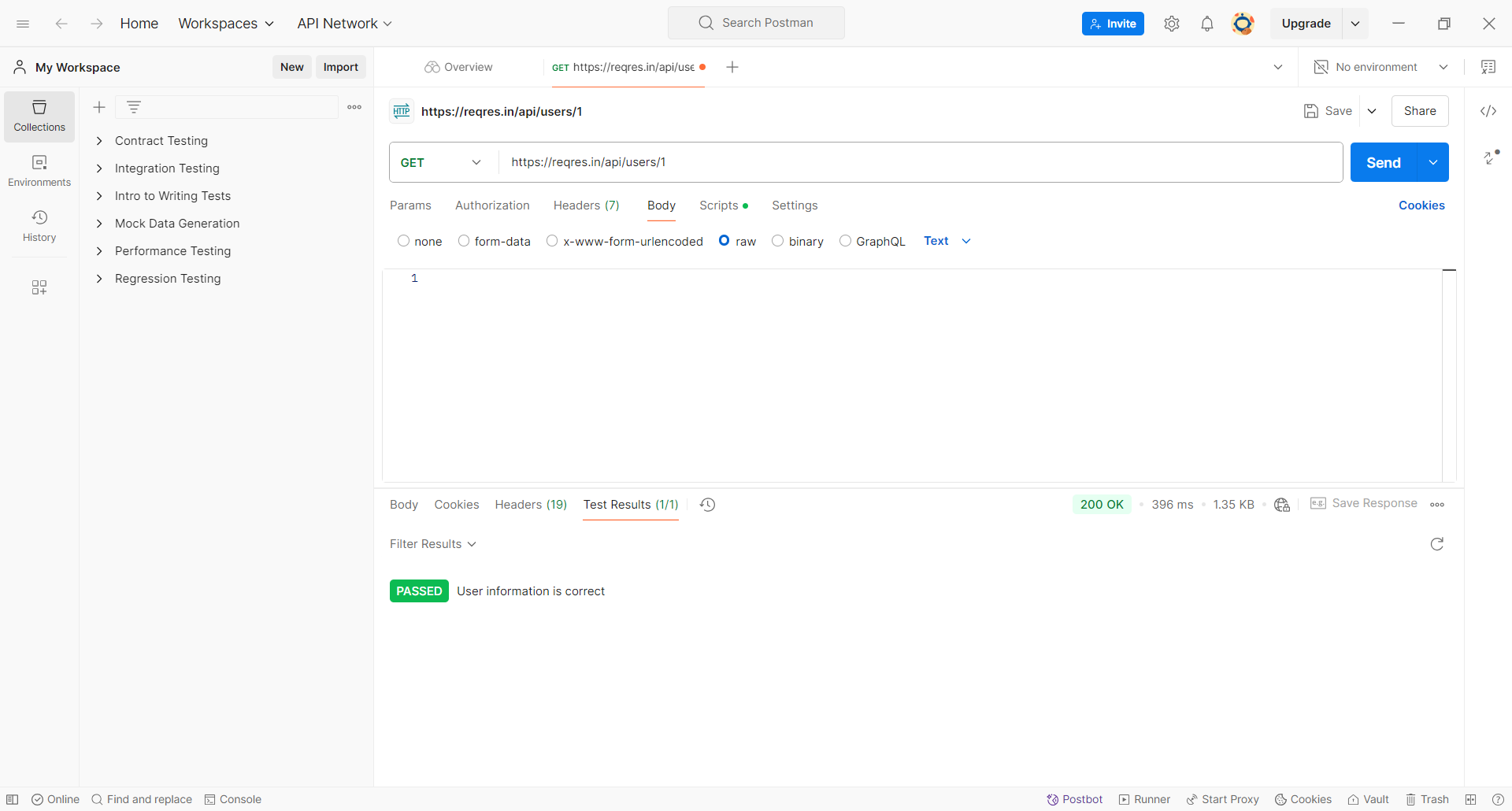
pm.response.to.have.status(200);

pm.expect(pm.response.json().data.first\_name).to.equal("George");

pm.expect(pm.response.json().data.email).to.equal("george.bluth@reqres.in");

});





* Обновление данных пользователя:

https://reqres.in/api/users/2 (PUT)

Тело:

{ "email": "janwea@reqres.in" }

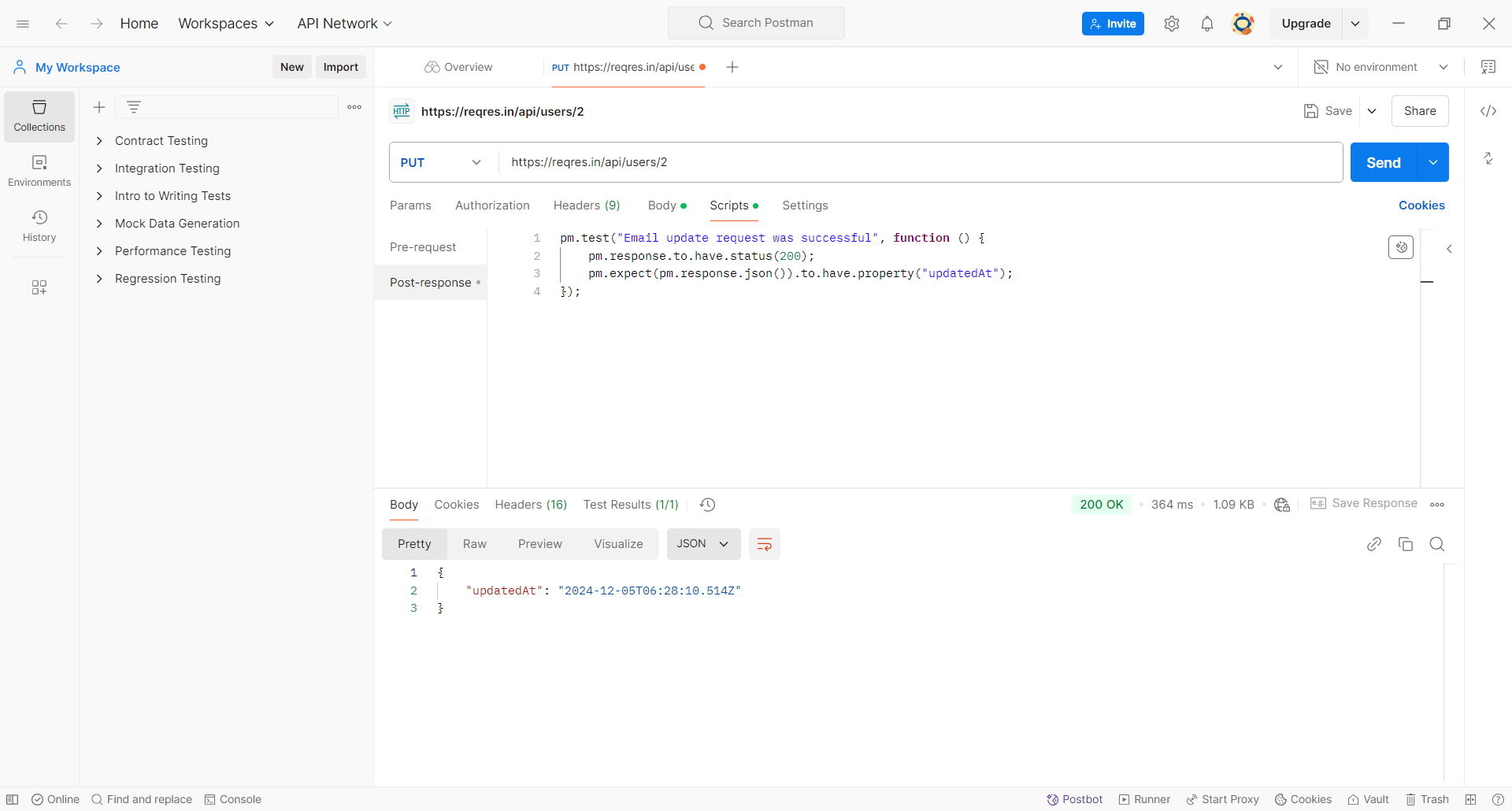
Тест:

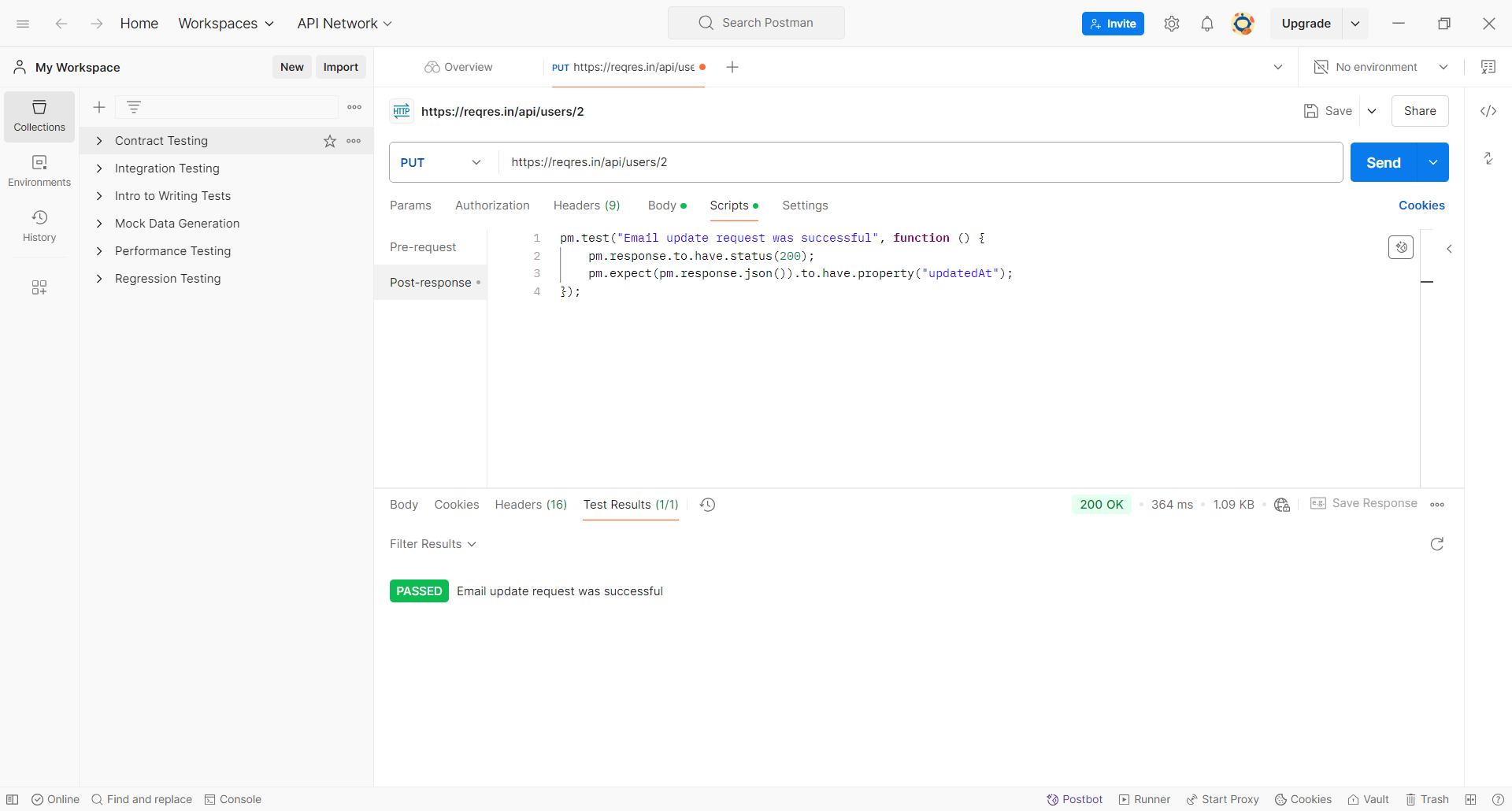
pm.test("Email update request was successful", function () {

pm.response.to.have.status(200);

pm.expect(pm.response.json()).to.have.property("updatedAt");

});





* Удаление пользователя:

https://reqres.in/api/users/6 (DELETE)

Тест:

pm.test("User is deleted successfully", function () {

pm.response.to.have.status(204);

});

pm.sendRequest({

url: "https://reqres.in/api/users/6",

method: 'GET'

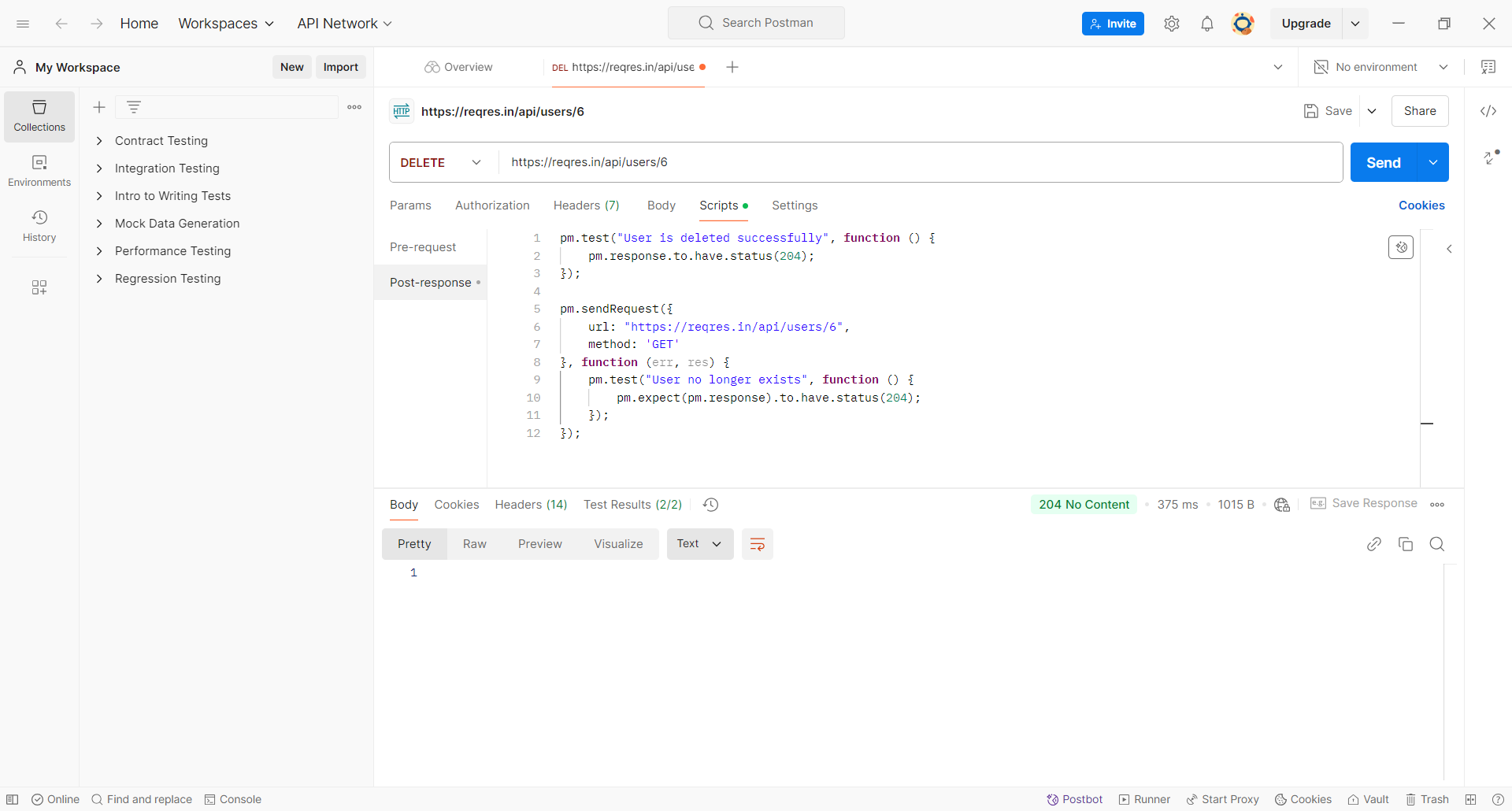
}, function (err, res) {

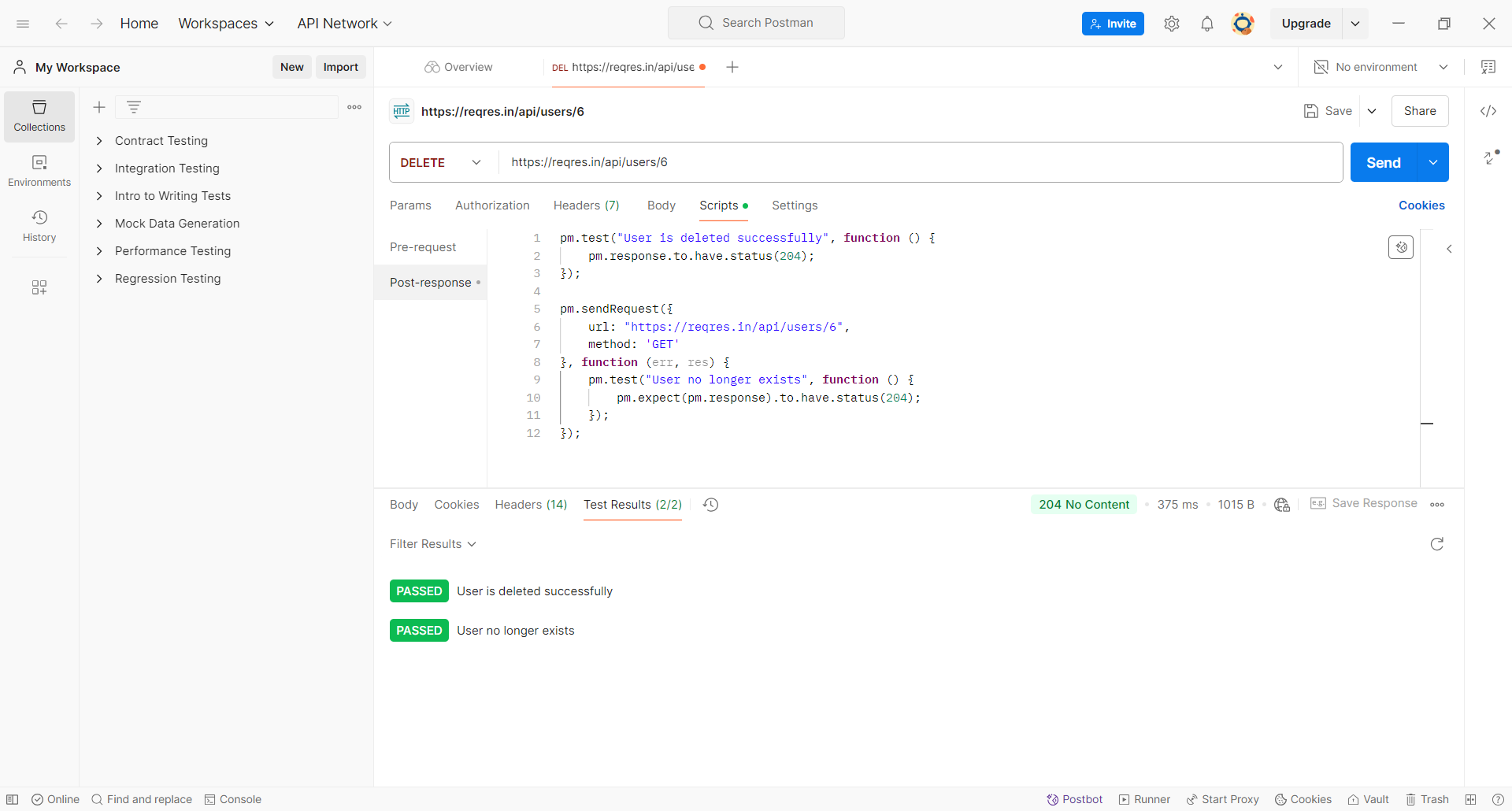
pm.test("User no longer exists", function () {

pm.expect(pm.response).to.have.status(204);

});

});





**Тестирование обработки ошибок:**

**Тесты для некорректных запросов:**

* Пустое тело:

https://reqres.in/api/products (POST)

Тело:

{ }

Тест:

pm.test("Empty body request returns 400", function () {

pm.response.to.have.status(400);

pm.expect(pm.response.json().error).to.exist;

});

* Некорректный формат данных:

https://reqres.in/api/products (POST)

Тело:

{ "id": 7, "name": 1, "year": "2 0 1 0", "color": "pink" }

Тест:

pm.test("Invalid data format returns 400", function () {

pm.response.to.have.status(400);

pm.expect(pm.response.json().error).to.exist;

});

* Несуществующая конечная точка:

https://reqres.in/api/nonexistent/point (GET)

Тест:

pm.test("Request to nonexistent endpoint returns 404", function () {

pm.response.to.have.status(404);

pm.expect(pm.response.json().error).to.exist;

});

**Тестирование валидации данных:**

**Валидация входных данных:**

* Слишком длинная строка в поле:

https://reqres.in/api/users (POST)

Тело:

{ "username": "user1".repeat(300), "password": "password1" }

Тест:

pm.test("Long string in name returns 400", function () {

pm.response.to.have.status(400);

pm.expect(pm.response.json().error).to.exist;

});

* Числа выходят за диапазон:

https://reqres.in/api/products (POST)

Тело:

{ "name": "new", "year": 20000, "color": "#98B2D1" }

Тест:

pm.test("Out of range number returns 400", function () {

pm.response.to.have.status(400);

pm.expect(pm.response.json().error).to.exist;

});

* Отсутствие обязательного поля:

https://reqres.in/api/products (POST)

Тело:

{ "year": 2005}

Тест:

pm.test("Missing required field returns 400", function () {

pm.response.to.have.status(400);

pm.expect(pm.response.json().error).to.exist;

});

**Тестирование пагинации**

**Запросы с параметрами пагинации:**

* Список объектов с параметрами:

https://reqres.in/api/users?page=2&limit=6 (GET)

Тест:

pm.test("Returns correct number of objects", function () {

pm.response.to.have.status(200);

const responseData = pm.response.json();

pm.expect(responseData.data.length).to.equal(6);

});

* Возвращаются только данные указанной страницы:

https://reqres.in/api/users?page=2&limit=6 (GET)

Тест:

pm.test("Returns correct data for page 2", function () {

pm.response.to.have.status(200);

const responseData = pm.response.json();

pm.expect(responseData.data.length).to.equal(6);

pm.expect(responseData.page).to.equal(2);

});

* Страница за пределами диапазона:

https://reqres.in/api/users?page=10 (GET)

Тест:

pm.test("Requesting a non-existent page returns empty data", function () {

pm.response.to.have.status(200);

const responseData = pm.response.json();

pm.expect(responseData.data.length).to.equal(0);

});