

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа №4

Выполнил:

Захарчук Александр

К3341

Проверил:

Добряков Д. И.

Санкт-Петербург

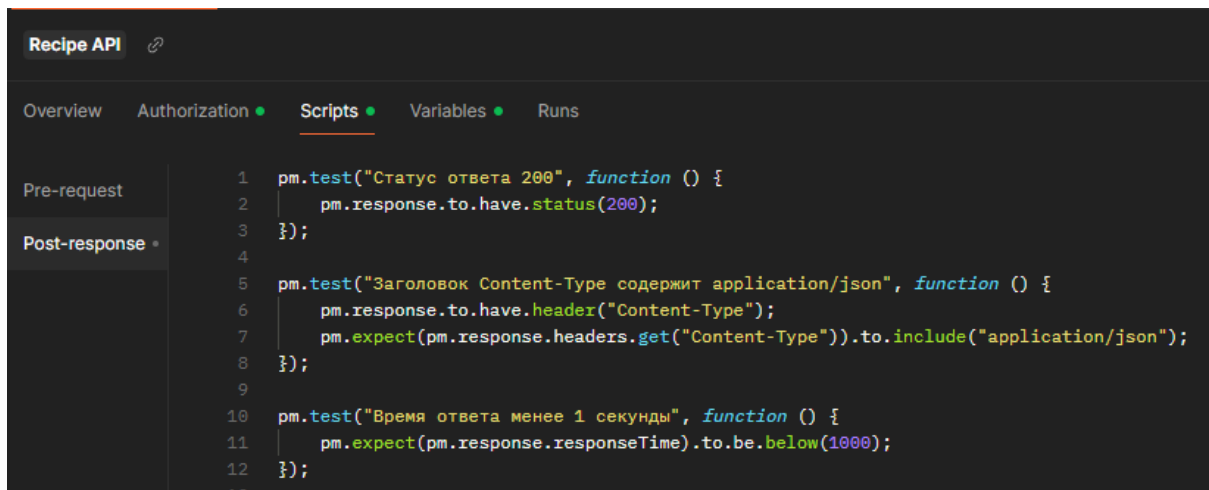
2025 г.

Задача

- Реализовать тестирование API средствами Postman;
- Написать тесты внутри Postman.

Ход работы

На уровне коллекции были реализованы тесты для всех запросов. Эти тесты проверяют статус код ответа, наличие заголовков для JSON, а также время ответа.



```
Recipe API 🔗
Overview Authorization ● Scripts ● Variables ● Runs
Pre-request
Post-response *
1  pm.test("Статус ответа 200", function () {
2      pm.response.to.have.status(200);
3  });
4
5  pm.test("Заголовок Content-Type содержит application/json", function () {
6      pm.response.to.have.header("Content-Type");
7      pm.expect(pm.response.headers.get("Content-Type")).to.include("application/json");
8  });
9
10 pm.test("Время ответа менее 1 секунды", function () {
11     pm.expect(pm.response.responseTime).to.be.below(1000);
12 });
13
```

Для отдельных эндпоинтов были добавлены специализированные тесты, которые проверяют данные в ответе.

Тест для логина пользователя также устанавливает полученный токен в переменную окружения для авторизации в последующих запросах.

POST {{baseUrl}} /users/login

Params Authorization Headers (11) Body ● Scripts ● Settings

Pre-request

Post-response *

```
1 const jsonData = pm.response.json();
2
3 pm.test("Ответ содержит ключ token с непустой строкой", function () {
4     pm.expect(jsonData).to.have.property("token");
5     pm.expect(jsonData.token).to.be.a("string").and.to.not.be.empty;
6 });
7
8 pm.environment.set("authToken", jsonData.token);
```

Body Cookies Headers (7) Test Results (4/4) ↻

Filter Results ▾

- PASSED** Статус ответа 200
- PASSED** Заголовок Content-Type содержит application/json
- PASSED** Время ответа менее 1 секунды
- PASSED** Ответ содержит ключ token с непустой строкой

Тесты для GET запросов проверяют корректность полученных данных.

GET

{{baseUrl}} /users/:username

Params • Authorization • Headers (9) Body Scripts • Settings

Pre-request

Post-response •

```
1 pm.test("Ответ соответствует ожидаемой структуре", function () {
2     const jsonData = pm.response.json();
3
4     pm.expect(jsonData).to.be.an("object");
5
6     pm.expect(jsonData).to.have.all.keys(
7         "created_at",
8         "updated_at",
9         "username",
10        "email",
11        "profile_picture",
12        "bio"
13    );
14
15    pm.expect(jsonData.created_at).to.be.a("string");
16    pm.expect(jsonData.updated_at).to.be.a("string");
17    pm.expect(jsonData.username).to.be.a("string").and.to.not.be.empty;
18    pm.expect(jsonData.email).to.be.a("string").and.to.include("@");
19    pm.expect(jsonData.profile_picture).to.be.a("string");
20    pm.expect(jsonData.bio).to.be.a("string");
21
22    const iso8601regex = /^\\d{4}-\\d{2}-\\d{2}T\\d{2}:\\d{2}:\\d{3}Z$/;
23
24    pm.expect(jsonData.created_at).to.match(iso8601regex);
25    pm.expect(jsonData.updated_at).to.match(iso8601regex);
26    // ...
27 }
```

Body Cookies Headers (7) Test Results (4/4) ↻

Filter Results ▾

- PASSED** Статус ответа 200
- PASSED** Заголовок Content-Type содержит application/json
- PASSED** Время ответа менее 1 секунды
- PASSED** Ответ соответствует ожидаемой структуре

GET {{baseUrl}} /users

Params Authorization Headers (9) Body Scripts Settings

Pre-request

Post-response

```
1 pm.test("Ответ — массив пользователей с корректной структурой", function () {
2   const jsonData = pm.response.json();
3   pm.expect(jsonData).to.be.an("array").that.is.not.empty;
4   const iso8601regex = /^[\d]{4}-[\d]{2}-[\d]{2}T[\d]{2}:[\d]{2}:[\d]{3}Z$/;
5   jsonData.forEach((user, index) => {
6     pm.expect(user).to.be.an("object");
7     pm.expect(user).to.have.all.keys(
8       "created_at",
9       "updated_at",
10      "username",
11      "email",
12      "profile_picture",
13      "bio"
14    );
15    pm.expect(user.created_at, `created_at в элементе ${index}`).to.match(iso8601regex);
16    pm.expect(user.updated_at, `updated_at в элементе ${index}`).to.match(iso8601regex);
17    pm.expect(user.username, `username в элементе ${index}`).to.be.a("string").and.to.not.be.empty;
18    pm.expect(user.email, `email в элементе ${index}`).to.be.a("string").and.to.include("@");
19    pm.expect(user.profile_picture, `profile_picture в элементе ${index}`).to.be.a("string");
20    pm.expect(user.bio, `bio в элементе ${index}`).to.be.a("string");
21  });
22 });
```

Body Cookies Headers (7) Test Results (4/4)

Filter Results

- PASSED Статус ответа 200
- PASSED Заголовок Content-Type содержит application/json
- PASSED Время ответа менее 1 секунды
- PASSED Ответ — массив пользователей с корректной структурой

Запрос для создания пользователя предварительно генерирует случайное имя, которое будет использовано в теле.

POST {{baseUrl}} /users

Params Authorization Headers (11) Body Scripts Settings

Pre-request

Post-response

```
1 const randomString = Math.random().toString(36).substring(2, 12);
2 pm.environment.set("randomUsername", randomString);
```

```
1 {
2   "username": "{{randomUsername}}",
3   "email": "{{randomUsername}}@mail.ru",
4   "password": "{{randomUsername}}12345",
5   "profile_picture": "test picture",
6   "bio": "test bio"
7 }
```

Тест проверяет, что переданное имя соответствует тому, что было получено в ответе.

The screenshot displays a REST client interface for a POST request to the endpoint `{{baseUrl}}/users`. The 'Scripts' tab is active, showing a series of 14 lines of JavaScript code for testing the response. The code includes assertions for the response body's structure and content, such as checking for a 'username' property and validating ISO 8601 timestamps for 'created_at' and 'updated_at'.

```
1 pm.test("Пользователь создан с правильным именем и корректным ответом", function () {
2   const jsonData = pm.response.json();
3   const expectedUsername = pm.environment.get("randomUsername");
4
5   pm.expect(jsonData).to.be.an("object");
6   pm.expect(jsonData).to.have.property("username", expectedUsername);
7   pm.expect(jsonData).to.have.property("email").that.includes(expectedUsername);
8   pm.expect(jsonData).to.have.property("profile_picture").that.is.a("string");
9   pm.expect(jsonData).to.have.property("bio").that.is.a("string");
10
11   const iso8601regex = /^\\d{4}-\\d{2}-\\d{2}T\\d{2}:\\d{2}:\\d{2}\\.\\d{3}Z$/;
12   pm.expect(jsonData.created_at).to.match(iso8601regex);
13   pm.expect(jsonData.updated_at).to.match(iso8601regex);
14 });
```

Below the script editor, the 'Test Results' tab shows four passed tests:

- PASSED** Статус ответа 200
- PASSED** Заголовок Content-Type содержит application/json
- PASSED** Время ответа менее 1 секунды
- PASSED** Пользователь создан с правильным именем и корректным ответом

Тесты изменения и удаления пользователя также используют username, сохраненный в переменной окружения.

PUT {{baseUrl}} /users/:username

Params • Authorization • Headers (12) Body • Scripts • Settings

```
1 pm.test("Пользователь создан с правильным именем и корректным ответом", function () {
2   const jsonData = pm.response.json();
3   const expectedUsername = pm.environment.get("randomUsername");
4
5   pm.expect(jsonData).to.be.an("object");
6   pm.expect(jsonData).to.have.property("username", expectedUsername);
7   pm.expect(jsonData).to.have.property("email", `${expectedUsername}@new.mail.ru`);
8   pm.expect(jsonData).to.have.property("profile_picture").that.is.a("string");
9   pm.expect(jsonData).to.have.property("bio").that.is.a("string");
10
11   const iso8601regex = /^\\d{4}-\\d{2}-\\d{2}T\\d{2}:\\d{2}:\\d{2}\\.\\d{3}Z$/;
12   pm.expect(jsonData.created_at).to.match(iso8601regex);
13   pm.expect(jsonData.updated_at).to.match(iso8601regex);
14 });
```

Body Cookies Headers (7) Test Results (4/4) | 🕒

Filter Results ▾

- PASSED** Статус ответа 200
- PASSED** Заголовок Content-Type содержит application/json
- PASSED** Время ответа менее 1 секунды
- PASSED** Пользователь создан с правильным именем и корректным ответом

DELETE

▼

{{baseUrl}} /users/:username

Params

Authorization

Headers (8)

Body

Scripts

Settings

Query Params

	Key	Value
	Key	Value

Path Variables

	Key	Value
	username	{{randomUsername}}

Body

Cookies

Headers (7)

Test Results (3/3)

🕒

Filter Results ▼

PASSED

Статус ответа 200

PASSED


Заголовок Content-Type содержит application/json

PASSED

Время ответа менее 1 секунды

Результат запуска коллекции:

Recipe API - Run results

 Ran today at 03:05:09 · [View all runs](#)

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	test env	1	2s 799ms	23	147 ms

RUN SUMMARY

					1
▼	POST	Логин пользователя			4 0
		Pass Статус ответа 200			
		Pass Заголовок Content-Type содержит arr...			
		Pass Время ответа менее 1 секунды			
		Pass Ответ содержит ключ token с непустой ...			
▼	GET	Получить список всех пользователей			4 0
		Pass Статус ответа 200			
		Pass Заголовок Content-Type содержит arr...			
		Pass Время ответа менее 1 секунды			
		Pass Ответ — массив пользователей с корре...			
▼	GET	Получить пользователей по имени			4 0
		Pass Статус ответа 200			
		Pass Заголовок Content-Type содержит arr...			
		Pass Время ответа менее 1 секунды			
		Pass Ответ соответствует ожидаемой структ...			
▼	POST	Создание нового пользователя			4 0
		Pass Статус ответа 200			
		Pass Заголовок Content-Type содержит arr...			
		Pass Время ответа менее 1 секунды			
		Pass Пользователь создан с правильным им...			
▼	PUT	Обновить данные пользователя			4 0
		Pass Статус ответа 200			
		Pass Заголовок Content-Type содержит arr...			
		Pass Время ответа менее 1 секунды			
		Pass Пользователь создан с правильным им...			
▼	DELETE	Удалить пользователя			3 0
		Pass Статус ответа 200			
		Pass Заголовок Content-Type содержит arr...			
		Pass Время ответа менее 1 секунды			

Вывод

В ходе выполнения данной работы были изучены и применены на практике тесты API с помощью Postman, как для всей коллекции, так и для отдельных запросов. Также были использованы переменные окружения для передачи данных между запросами. Изученный инструмент позволит автоматизировать тесты API в будущем и снизить количество ошибок в них.