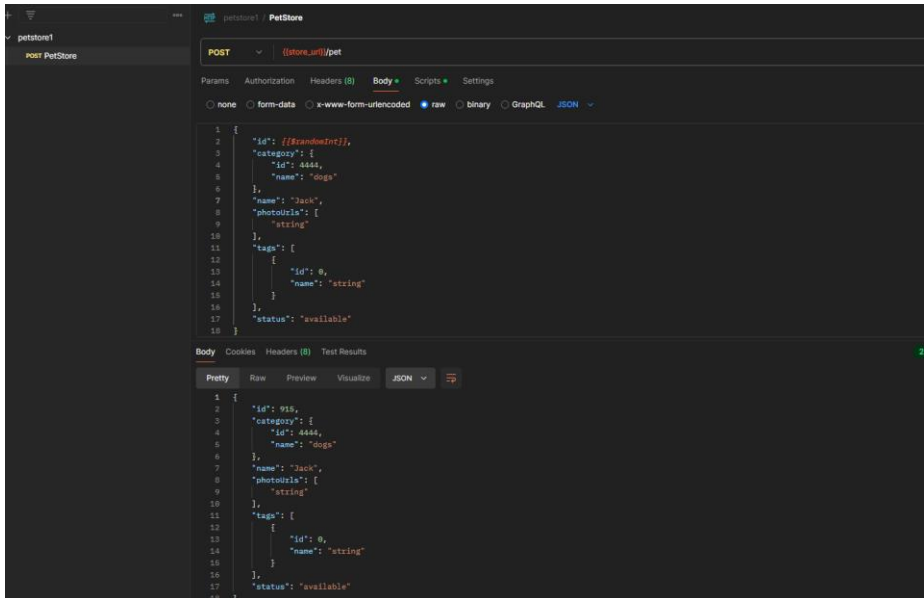
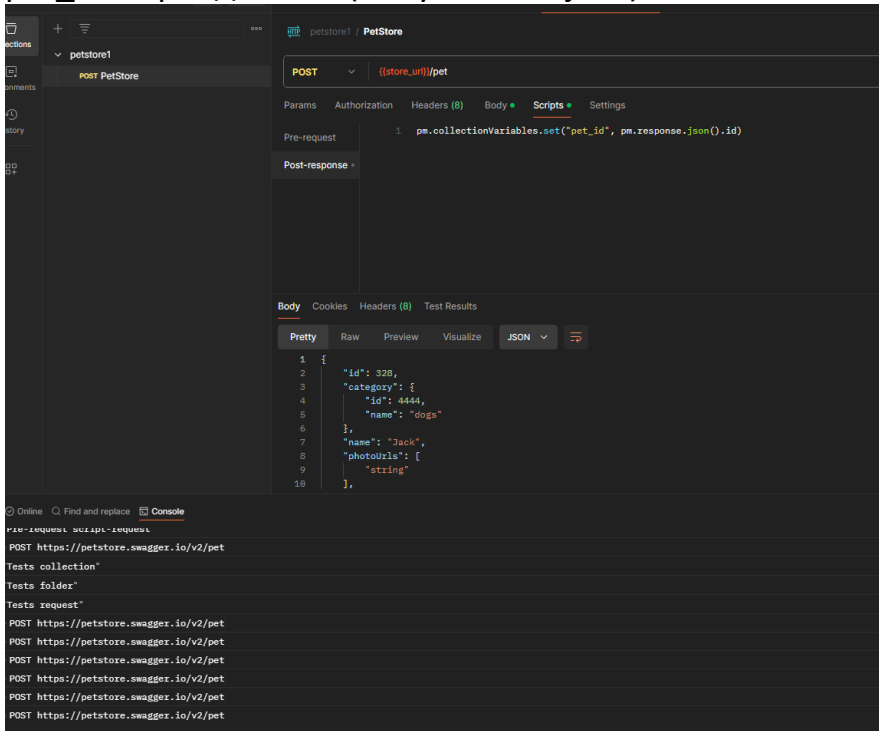
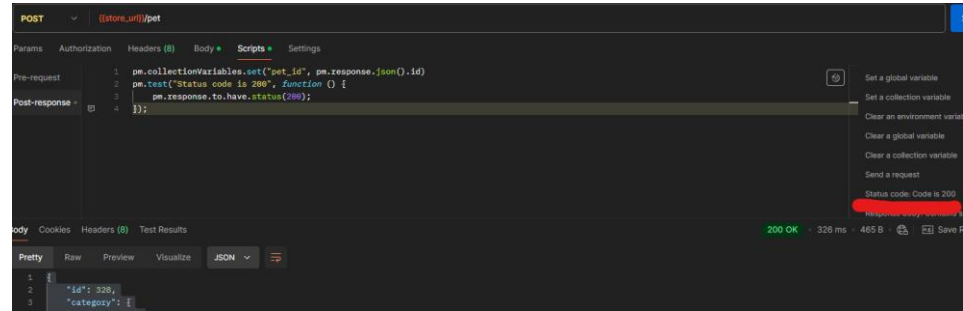
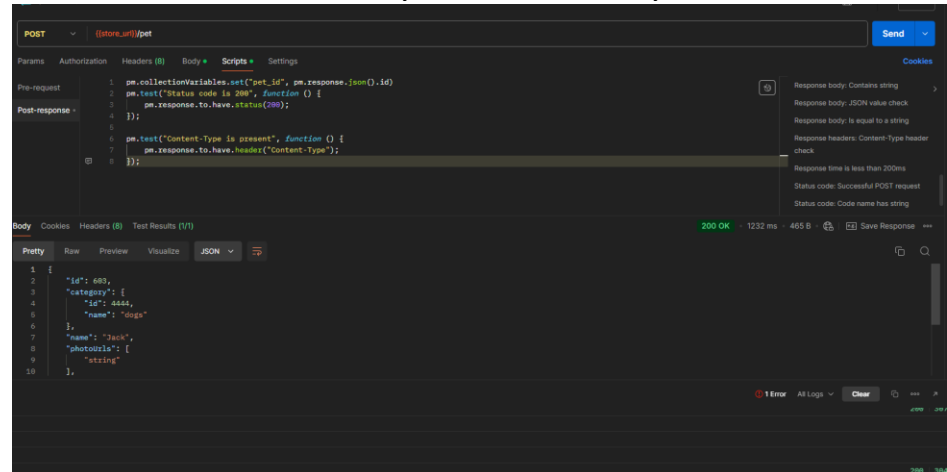


Тема Лабораторной Работы	Lb_7
Выполняющий	Коваленко Кирилл ИС221
Помогающий	Лавренов Александр ИС221
Ход работы	<div><div>1. Создали новый воркспейс, коллекцию и реквест для новой лабораторной и добавляем в id <code>{{\${randomInt}}}</code> вместо статичного id</div><div></div><div>2. Так же в tests (post-response) меняем значение переменной pet_id на случайное (получаем из json)</div><div></div></div>

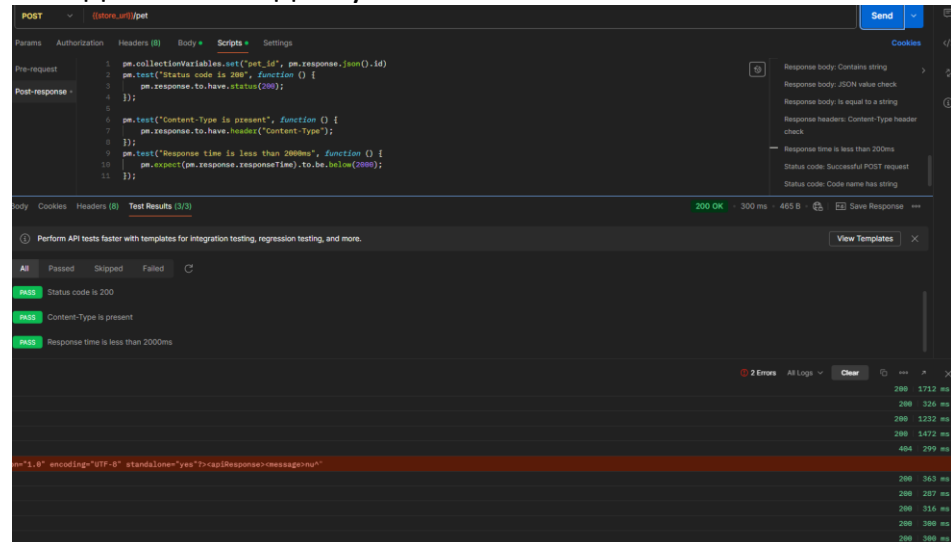
3. Используя snippet, добавляем ответ на код 200.



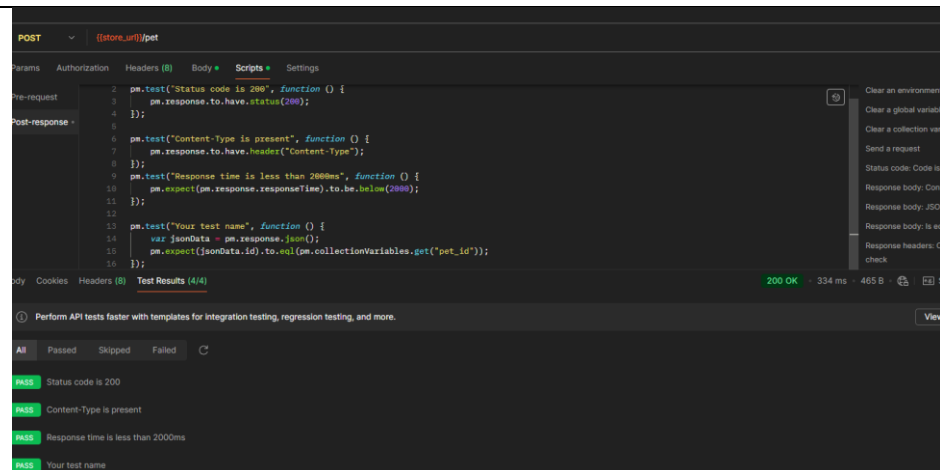
4. Точно так же как и в пункте 3 используем новый snippet



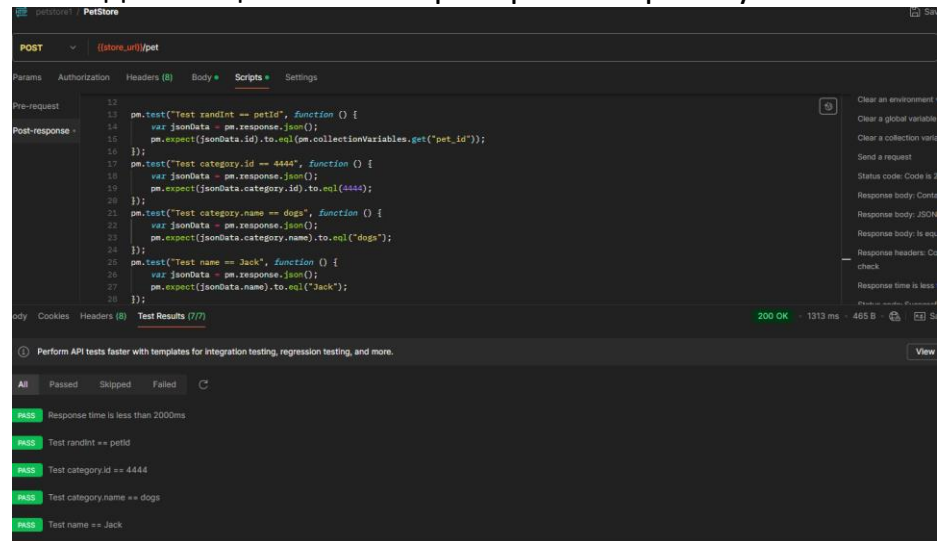
5. Добавляем проверку на время ответа сервера (2000мс) и во вкладке Tests видим успешные тесты.



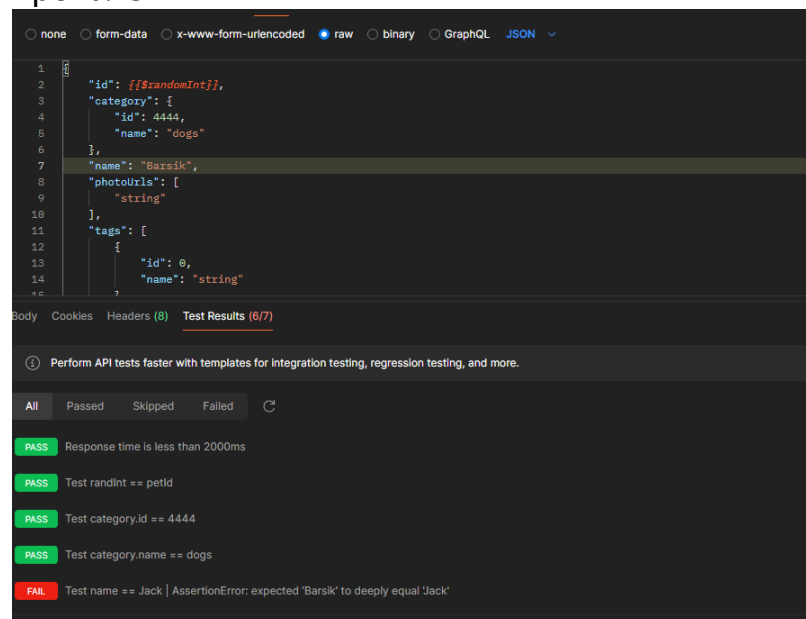
6. Так же добавляем тест на присутствие id и видим, что все тесты прошли успешно



7. Создаем еще 3 теста и проверяем их работу



8. Пробуем поменять имя пета на Barsik и видим, что тест провален



Так же проверяем имя Veeeeeee

Params Authorization Headers (8) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▾

```
1 {
2   "id": {{$randomInt}},
3   "category": {
4     "id": 4444,
5     "name": "dogs"
6   },
7   "name": "Beeeeeeee",
8   "photoUrls": [
9     "string"
10  ],
11  "tags": [
12    {
13      "id": 0,
14      "name": "string"
15    }
16  ]
17 }
```

Body Cookies Headers (8) **Test Results (6/7)**

① Perform API tests faster with templates for integration testing, regression testing, and more.

All Passed Skipped Failed ↻

PASS Response time is less than 2000ms

PASS Test randInt == petId

PASS Test category.id == 4444

PASS Test category.name == dogs



FAIL Test name == Jack | AssertionError: expected 'Beeeeeeee' to deeply equal 'Jack'

9. Создаем новый реквест с загрузкой файла.


POST ▾ `{{store_url}}/pet/{petId}/uploadImage`

Params Authorization Headers (8) **Body** Scripts Settings

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

Key	Value	Description
<input checked="" type="checkbox"/> additionalMetadata	Text ▾ avatar	
<input checked="" type="checkbox"/> file	File ▾  logo.jpg 	
Key	Value	Description

Body Cookies Headers (8) Test Results **200 OK**

Pretty Raw Preview Visualize **JSON** ▾ 

```
1 {
2   "code": 200,
3   "type": "unknown",
4   "message": "additionalMetadata: avatar\nfile uploaded to ./logo.jpg, 1587469 bytes"
5 }
```

POST ▾ `{{store_url}}/pet/{petId}/uploadImage`

Params Authorization Headers (8) **Body** Scripts Settings

Query Params

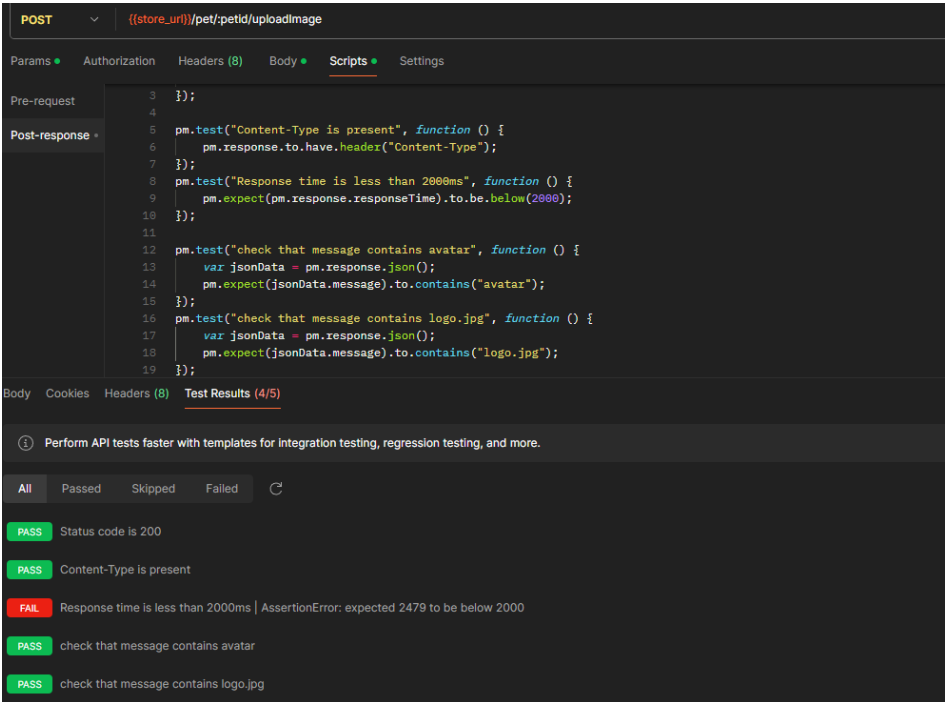
Key	Value
Key	Value

Path Variables

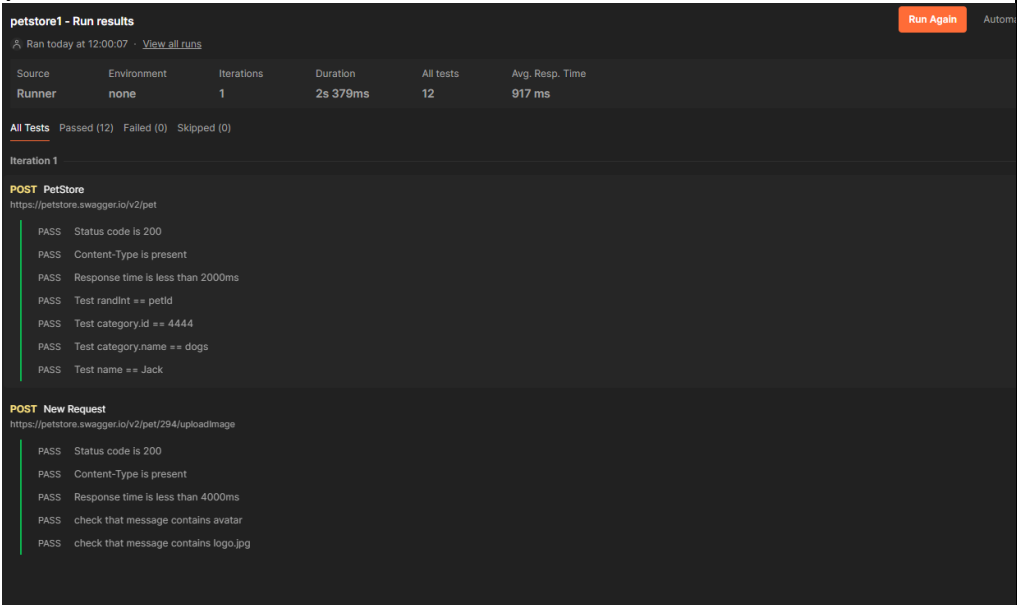
Key	Value
petId	<code>{{pet_id}}</code>

Body Cookies Headers (8) Test Results

10. Пишем тесты для проверки на наличие названия и самого лого. (ошибка из-за медленного интернета, т. е. долгий ответ сервера)



11. Запускаем коллекцию и видим, что все тесты прошли успешно



Результат

В ходе данной работы мы научились запускать автотесты и понимать как они работают

Оценка

Листинг:

Номер 2.

```
{
  "id": 328,
  "category": {
    "id": 4444,
    "name": "dogs"
  },
  "name": "Jack",
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "string"
    }
  ],
  "status": "available"
}
```

Листинг кода автотеста добавления питомца

```
pm.collectionVariables.set("pet_id", pm.response.json().id)
pm.test("Status code is 200", function () {
  pm.response.to.have.status(200);
});
```

```

pm.test("Content-Type is present", function () {
    pm.response.to.have.header("Content-Type");
});

pm.test("Response time is less than 2000ms", function () {
    pm.expect(pm.response.responseTime).to.be.below(2000);
});

pm.test("Test randInt == petId", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.id).to.eql(pm.collectionVariables.get("pet_id"));
});

pm.test("Test category.id == 4444", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.category.id).to.eql(4444);
});

pm.test("Test category.name == dogs", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.category.name).to.eql("dogs");
});

pm.test("Test name == Jack", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.name).to.eql("Jack");
});

```

Листинг кода автотеста загрузки аватара для питомца

```

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

```

```
});
```

```
pm.test("Content-Type is present", function () {  
    pm.response.to.have.header("Content-Type");
```

```
});
```

```
pm.test("Response time is less than 4000ms", function () {  
    pm.expect(pm.response.responseTime).to.be.below(4000);
```

```
});
```

```
pm.test("check that message contains avatar", function () {
```

```
    var jsonData = pm.response.json();
```

```
    pm.expect(jsonData.message).to.contain("avatar");
```

```
});
```

```
pm.test("check that message contains logo.jpg", function () {
```

```
    var jsonData = pm.response.json();
```

```
    pm.expect(jsonData.message).to.contain("logo.jpg");
```

```
});
```