API Assignment

Devendra Kumar Vatsa

Vatsadev081@gmail.com

8629994177

Q.01: Find the endpoints and test with Postman using chai assertion and automate same in RESTAssured.

Answer 01: First I using manual testing from postman for all API of Q.1

1. Find pet by its status

GET: https://petstore.swagger.io/v2/pet/findByStatus?status=available

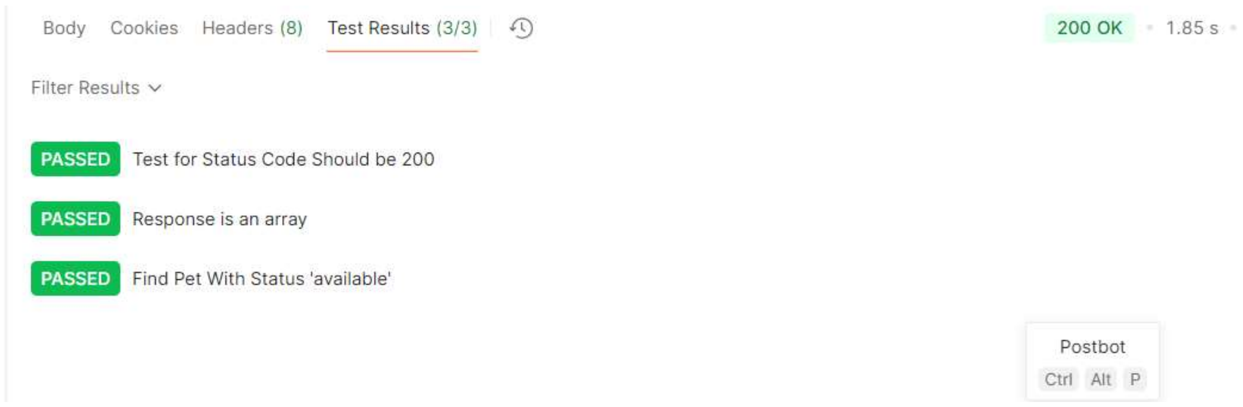
//Test for Status Code

Scripts:

```
pm.test("Test for Status Code Should be 200", ()=>
{
   pm.expect(pm.response.code).to.eql(200);
})

//Find Response is an array "

let res = pm.response.json();

pm.test("Response is an array",()=>
{
   pm.expect(res).to.be.an("array");
})
//Find Pet by Status "Available"
pm.test("Find Pet With Status 'available'", function () {
   let jsonData = pm.response.json(); // Parse the JSON response
```

```
  // Iterate over each pet in the response array
  jsonData.forEach((pet) => {
    if (pet.status === "available") {
      console.log(`Pet Name: ${pet.name}`);
    }
  });
});
```

Result in Postman:



PASSED Test for Status Code Should be 200
PASSED Response is an array
PASSED Find Pet With Status 'available'

---

2. Add new Pet to the store

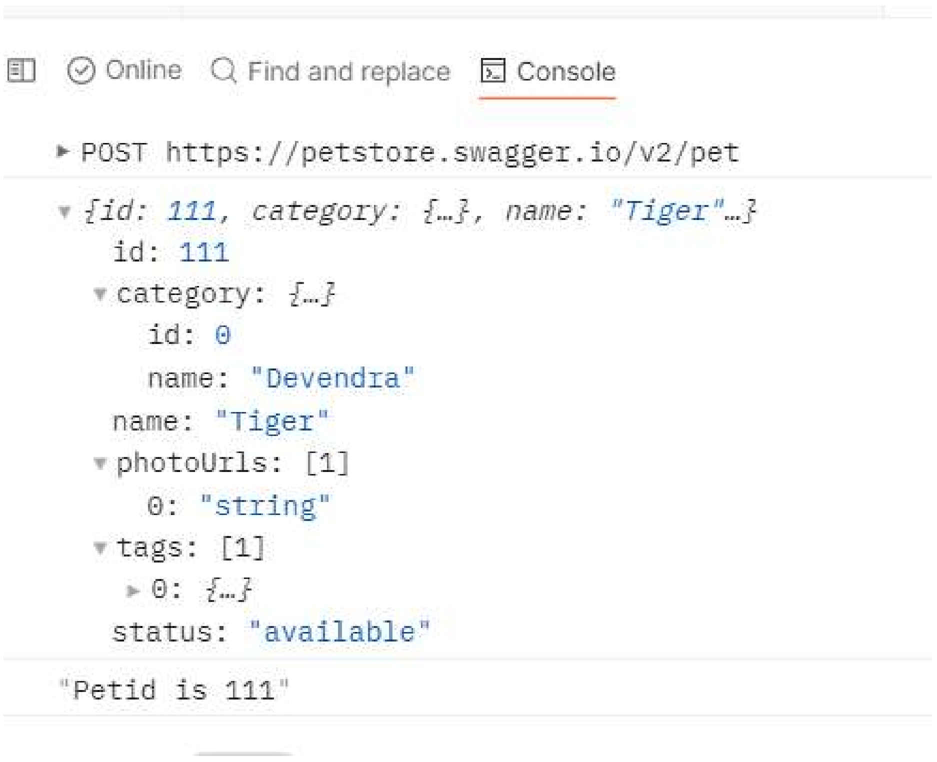
POST: https://petstore.swagger.io/v2/pet

Scripts:

```
//Store Response in Varaible

let res = pm.response.json();

//Test For Status Code
```

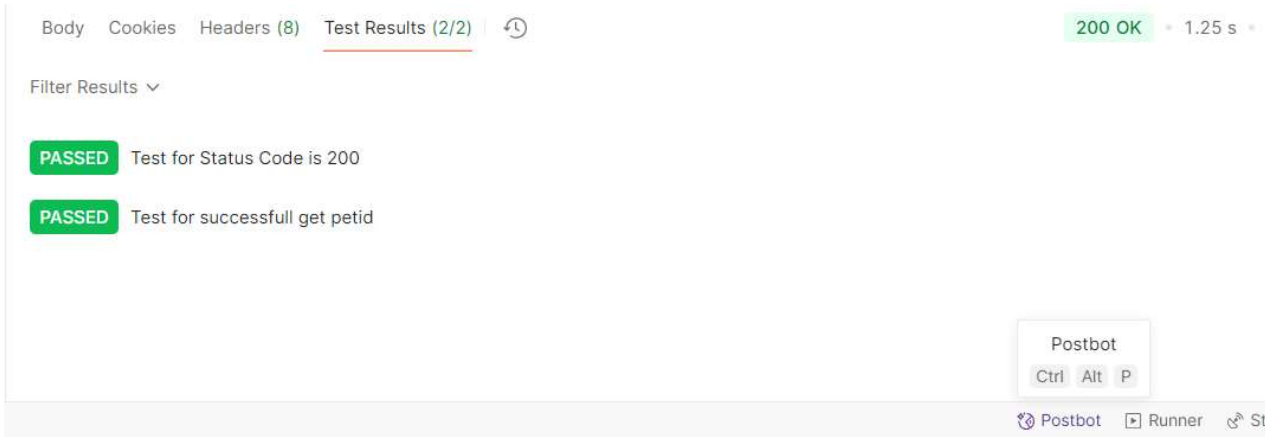
```
pm.test("Test for Status Code is 200", ()=>
{
   pm.expect(pm.response.code).to.eql(200);
})

let petid = res.id;

//print id in console

console.log(petid);

//Set Environment Variable

pm.environment.set("petid",petid);

// Test for PetId

pm.test("Test for successfull get petid",()=>
{
   pm.expect(petid).to.eql(111);
})
```

Console Result in Postman:



```
▸ POST https://petstore.swagger.io/v2/pet
▾ {id: 111, category: {…}, name: "Tiger"…}
    id: 111
  ▾ category: {…}
      id: 0
      name: "Devendra"
    name: "Tiger"
  ▾ photoUrls: [1]
      0: "string"
  ▾ tags: [1]
    ▸ 0: {…}
    status: "available"
"Petid is 111"
```

Result in Postman:



```
PASSED  Test for Status Code is 200
PASSED  Test for successfull get petid
```

---

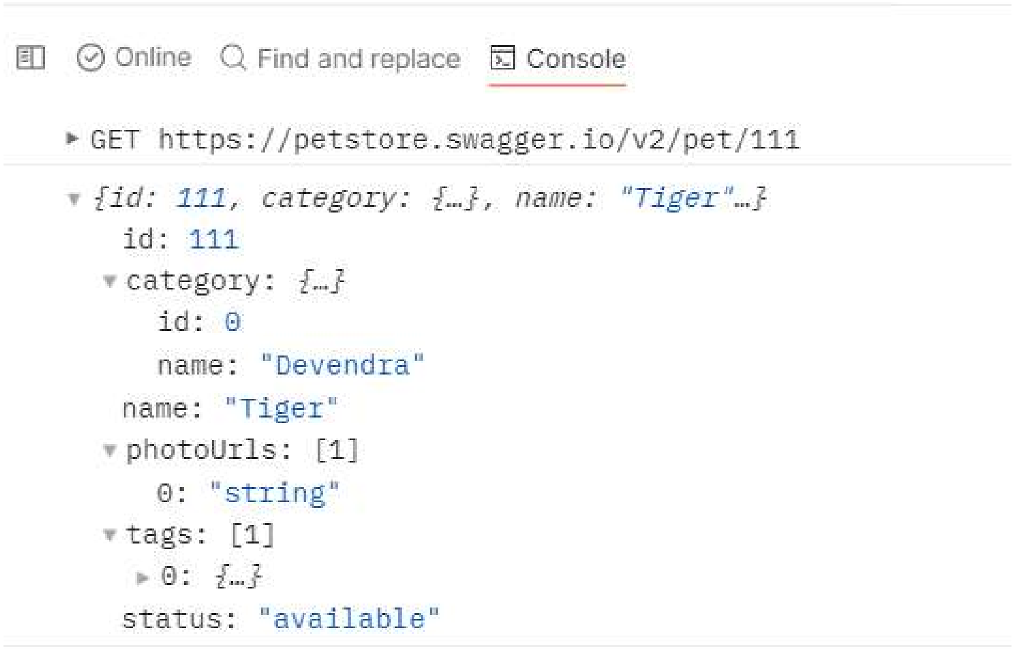
3. For the same generated id find the pet

GET: https://petstore.swagger.io/v2/pet/{{petid}}

Scripts:

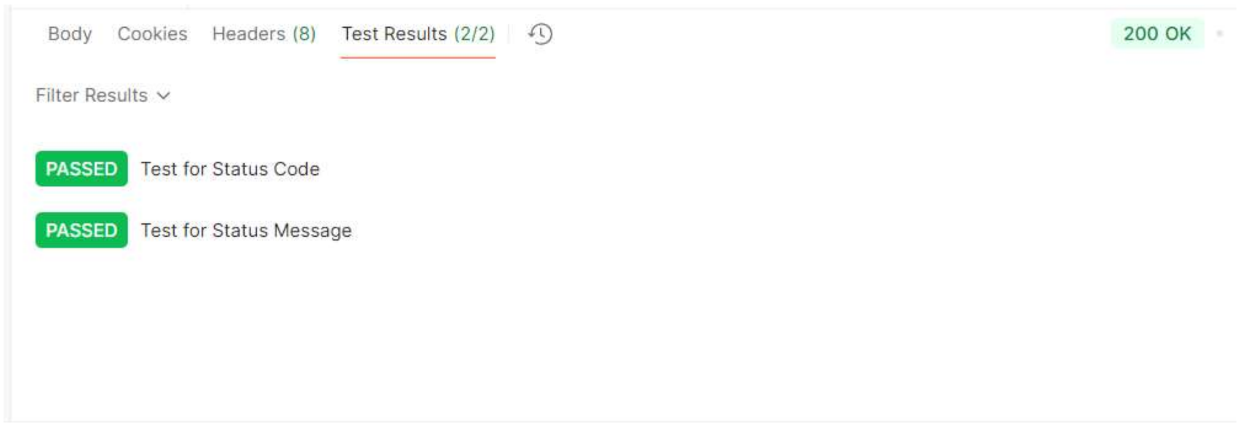
```
//Test for Status Code

pm.test("Test for Status Code",()=>
{
   pm.expect(pm.response.code).to.eql(200);
} )

// Test for Status Message

pm.test("Test for Status Message",()=>
{
   pm.expect(pm.response.status).to.eql("OK")
} );
```

Console Result in Postman:



```
▸ GET https://petstore.swagger.io/v2/pet/111
▾ {id: 111, category: {…}, name: "Tiger"…}
    id: 111
  ▾ category: {…}
      id: 0
      name: "Devendra"
    name: "Tiger"
  ▾ photoUrls: [1]
      0: "string"
  ▾ tags: [1]
    ▸ 0: {…}
    status: "available"
```

Result in Postman:



PASSED Test for Status Code

PASSED Test for Status Message

---

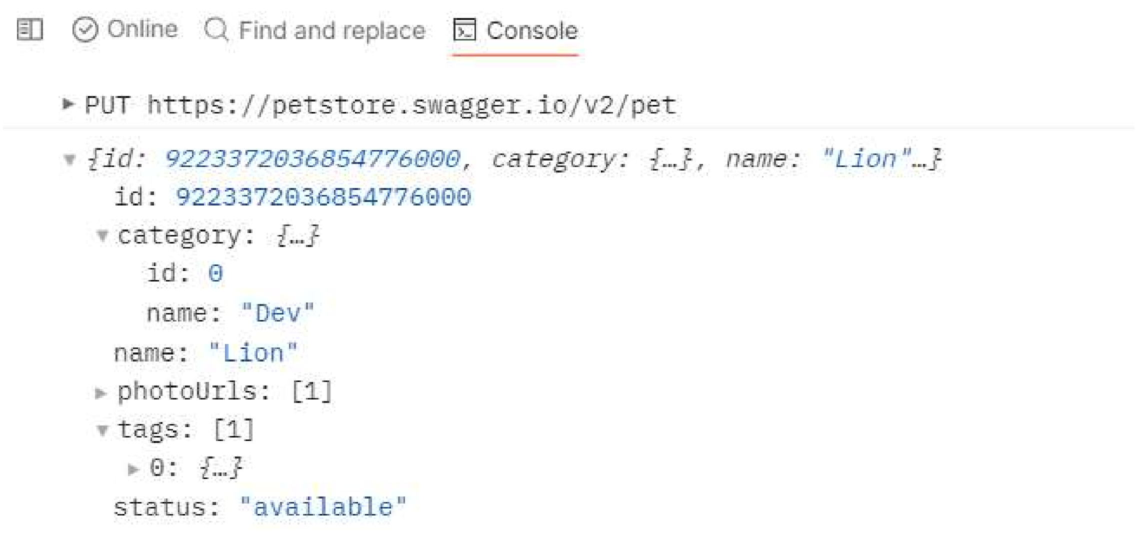
4. Update an existing pet

PUT: https://petstore.swagger.io/v2/pet

Scripts:

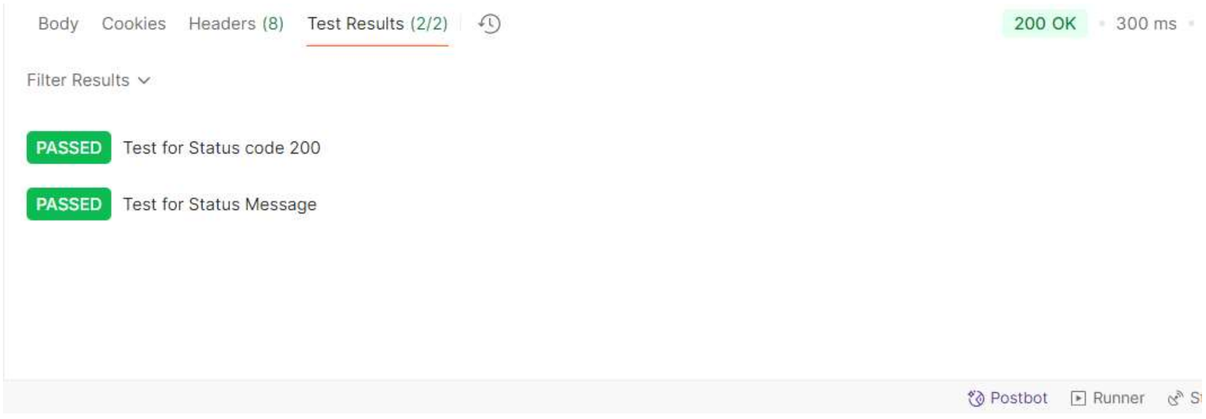
```
// Test for Status Code

pm.test("Test for Status code 200" , ()=>
{
   pm.expect(pm.response.code).to.eql(200)
})

// Test for Status Message

pm.test("Test for Status Message", ()=>
{
   pm.expect(pm.response.status).to.eql("OK")
});

// log the result in console

let res = pm.response.json();

console.log(res);
```

Console Result in Postman:



```
▸ PUT https://petstore.swagger.io/v2/pet
▾ {id: 9223372036854776000, category: {…}, name: "Lion"…}
    id: 9223372036854776000
  ▾ category: {…}
      id: 0
      name: "Dev"
    name: "Lion"
  ▸ photoUrls: [1]
  ▾ tags: [1]
    ▸ 0: {…}
    status: "available"
```

Result in Postman:



PASSED Test for Status code 200

PASSED Test for Status Message

---

5. Delete existing pet with same id

DELETE: https://petstore.swagger.io/v2/pet/{{petid}}

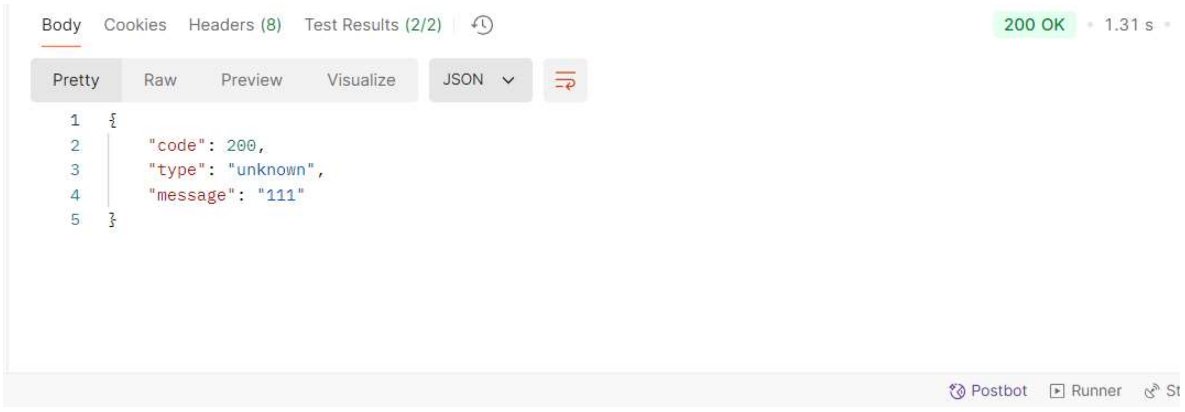
Scripts:

//Test for Status Code

pm.**test**("Test for Status Code", ()**=>**

```
{
   pm.expect(pm.response.code).to.eql(200)
} );

//Test for Status Message

pm.test("Test for Status Message",()=>
{
   pm.expect(pm.response.status).to.eql("OK");
} )
```

Body Message in Result window:

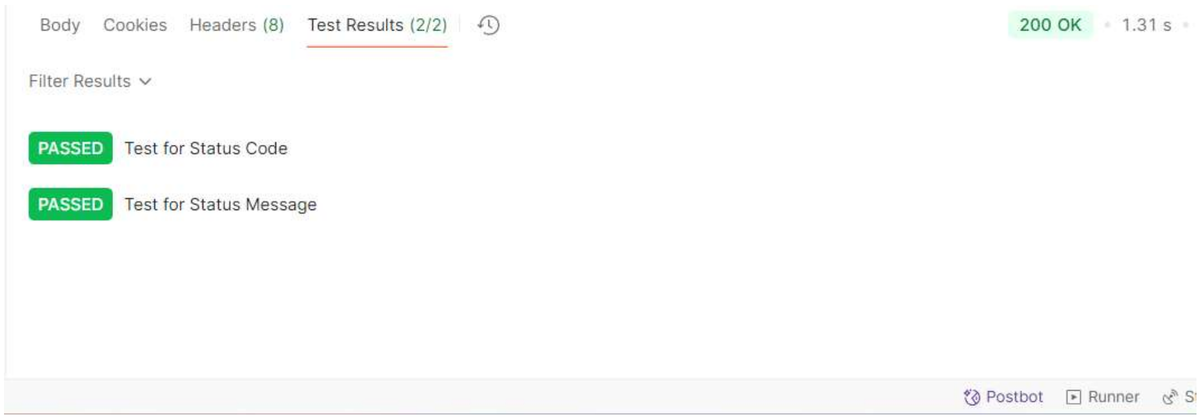


Body Cookies Headers (8) Test Results (2/2) 200 OK 1.31 s

Pretty Raw Preview Visualize JSON

```
{
    "code": 200,
    "type": "unknown",
    "message": "111"
}
```

Result in Postman:



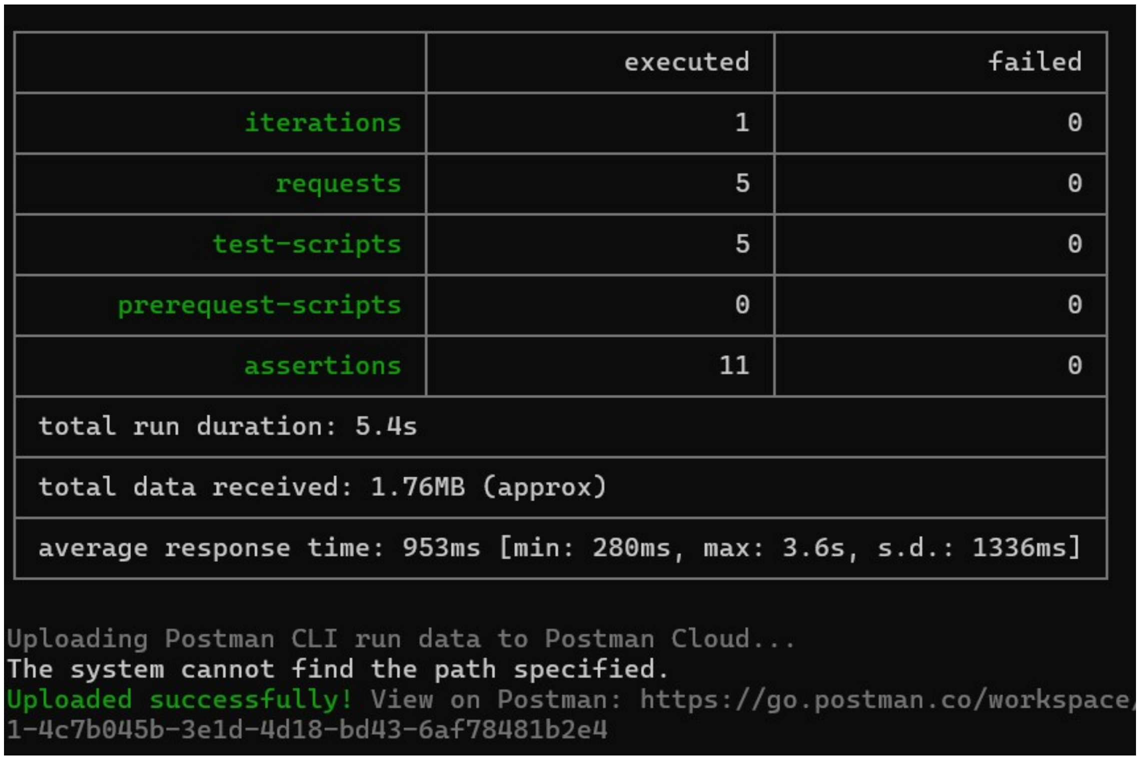
Body Cookies Headers (8) Test Results (2/2) 200 OK 1.31 s

Filter Results

PASSED Test for Status Code

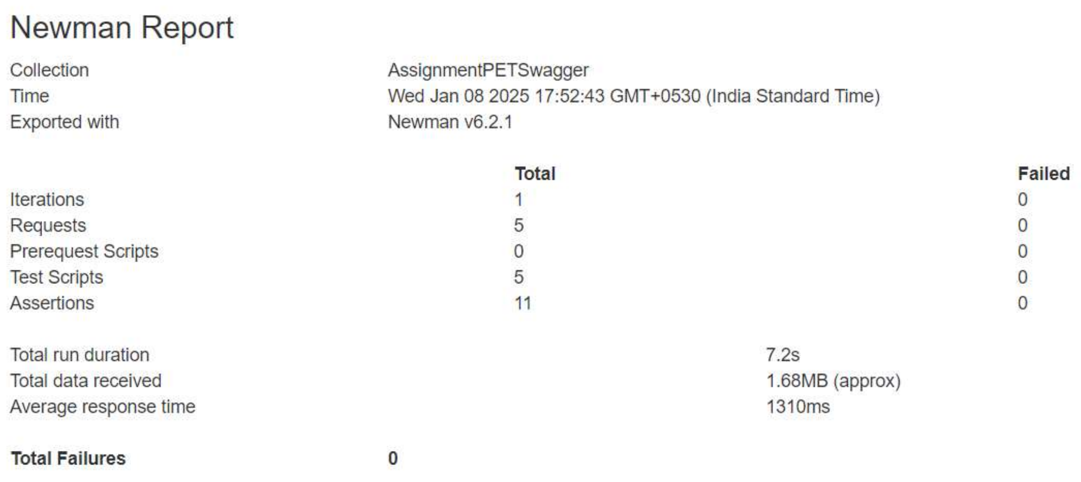
PASSED Test for Status Message

Collection Result of all API in Postman CLI:



```
┌─────────────────────────┬─────────────────────┬─────────────────────┐
│                         │            executed │              failed │
├─────────────────────────┼─────────────────────┼─────────────────────┤
│              iterations │                   1 │                   0 │
├─────────────────────────┼─────────────────────┼─────────────────────┤
│                requests │                   5 │                   0 │
├─────────────────────────┼─────────────────────┼─────────────────────┤
│            test-scripts │                   5 │                   0 │
├─────────────────────────┼─────────────────────┼─────────────────────┤
│      prerequest-scripts │                   0 │                   0 │
├─────────────────────────┼─────────────────────┼─────────────────────┤
│              assertions │                  11 │                   0 │
├─────────────────────────┴─────────────────────┴─────────────────────┤
│ total run duration: 5.4s                                            │
├─────────────────────────────────────────────────────────────────────┤
│ total data received: 1.76MB (approx)                                │
├─────────────────────────────────────────────────────────────────────┤
│ average response time: 953ms [min: 280ms, max: 3.6s, s.d.: 1336ms] │
└─────────────────────────────────────────────────────────────────────┘

Uploading Postman CLI run data to Postman Cloud...
The system cannot find the path specified.
Uploaded successfully! View on Postman: https://go.postman.co/workspace/
1-4c7b045b-3e1d-4d18-bd43-6af78481b2e4
```

Report in Newman:



Newman Report

Collection	AssignmentPETSwagger
Time	Wed Jan 08 2025 17:52:43 GMT+0530 (India Standard Time)
Exported with	Newman v6.2.1

	Total	**Failed**
Iterations	1	0
Requests	5	0
Prerequest Scripts	0	0
Test Scripts	5	0
Assertions	11	0

Total run duration	7.2s
Total data received	1.68MB (approx)
Average response time	1310ms

Total Failures **0**

Requests

Find pet by its status

Method	GET
URL	https://petstore.swagger.io/v2/pet/findByStatus?status=available
Mean time per request	5.4s
Mean size per request	1.67MB
Total passed tests	3
Total failed tests	0
Status code	200

Tests

Name	Pass count	Fail count
Test for Status Code Should be 200	1	0
Response is an array	1	0
Find Pet With Status 'available'	1	0

Add new Pet to the store

Method	POST
URL	https://petstore.swagger.io/v2/pet
Mean time per request	292ms
Mean size per request	140B
Total passed tests	2
Total failed tests	0
Status code	200

Tests

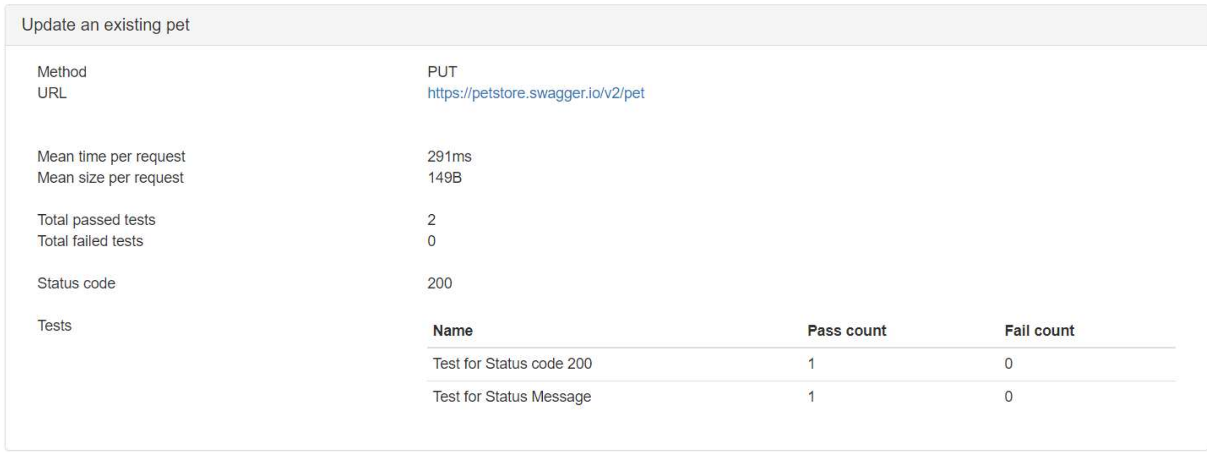
Name	Pass count	Fail count
Test for Status Code is 200	1	0
Test for successfull get petid	1	0

same generated id find the pet

Method	GET
URL	https://petstore.swagger.io/v2/pet/111
Mean time per request	285ms
Mean size per request	140B
Total passed tests	2
Total failed tests	0
Status code	200

Tests

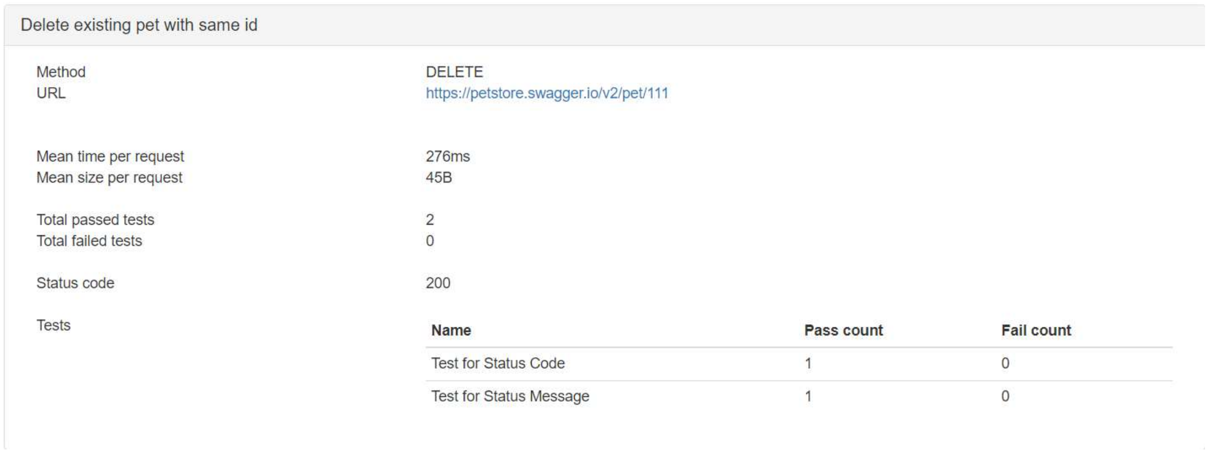
Name	Pass count	Fail count
Test for Status Code	1	0
Test for Status Message	1	0


Update an existing pet

Method	PUT
URL	https://petstore.swagger.io/v2/pet
Mean time per request	291ms
Mean size per request	149B
Total passed tests	2
Total failed tests	0
Status code	200

Tests

Name	Pass count	Fail count
Test for Status code 200	1	0
Test for Status Message	1	0

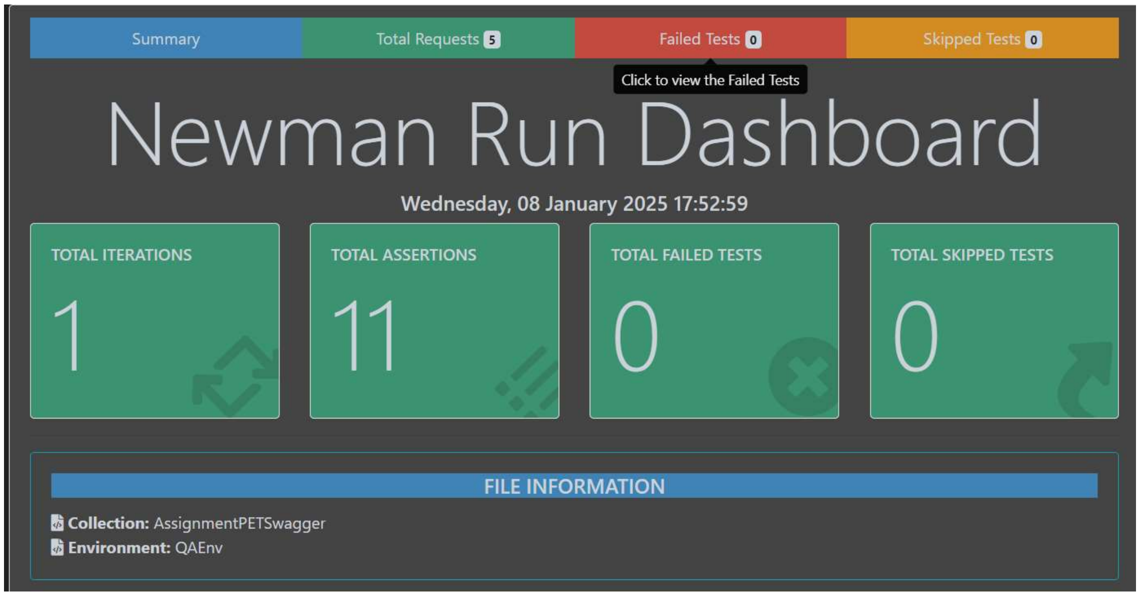

Delete existing pet with same id

Method	DELETE
URL	https://petstore.swagger.io/v2/pet/111
Mean time per request	276ms
Mean size per request	45B
Total passed tests	2
Total failed tests	0
Status code	200

Tests

Name	Pass count	Fail count
Test for Status Code	1	0
Test for Status Message	1	0

Newman Dash Board


Summary | Total Requests 5 | Failed Tests 0 | Skipped Tests 0

Click to view the Failed Tests

Newman Run Dashboard

Wednesday, 08 January 2025 17:52:59

TOTAL ITERATIONS 1

TOTAL ASSERTIONS 11

TOTAL FAILED TESTS 0

TOTAL SKIPPED TESTS 0

FILE INFORMATION

Collection: AssignmentPETSwagger

Environment: QAEnv

Automation of API in Rest Assured:

I used API Chaining Concept for all Requests API in Single TestNG File

Script in TestNG:

```
package com.Assignment;

import org.testng.annotations.Test;

import io.restassured.path.json.JsonPath;
import io.restassured.response.Response;

import static io.restassured.RestAssured.*;
import static io.restassured.matcher.RestAssuredMatchers.*;
import static org.hamcrest.Matchers.*;
import static org.testng.Assert.assertEquals;
import java.util.List;
import java.util.Map;


import org.testng.Assert;


public class APIQuestion1 {
        int expectedid;


 @Test (priority =1)
 public void  Find_pet_by_its_status() {


         System.out.println("Find Pet By Its Status");


        Response res =  given()
                .header("accept","application/json")


                .when().get("https://petstore.swagger.io/v2/pet/findByStatus?status=available");


                //log the result


                //res.then().log().body();


                //validate status code and Message



        Assert.assertEquals(res.getStatusCode(), 200);
        System.out.println("Status Code is: " + res.getStatusCode());



        Assert.assertEquals(res.getStatusLine(), "HTTP/1.1 200 OK");
```

```
            System.out.println("Status Message is: " + res.getStatusLine());


           // JSON path
   JsonPath json = res.jsonPath();


   // Extract the list of pets from the JSON response
   List<Map<String, Object>> pets = json.getList("");


   // Print pets with status "available"
   System.out.println("Pets with status 'available':");
   pets.forEach(pet -> {
       if ("available".equals(pet.get("status"))) { // Check status
           System.out.println("Pet Name: " + pet.get("name"));
       }
   });
}




@Test (priority =2)
public void  Add_new_Pet_to_the_store() {


            System.out.println("\n\nTest for Add New Pet to The Store:");


            Response res = given()
                                             .header("accept","application/json")
                                             .header("Content-Type","application/json")
                                             .body("{ \n"
                                                          + "\"id\": 111, \n"
                                                          + "\"category\": { \n"
                                                          + "\"id\": 0, \n"
                                                          + "\"name\": \"Devendra\" \n"
```

```
                                        + "}, \n"
                                        + "\"name\": \"Tiger\", \n"
                                        + "\"photoUrls\": [ \n"
                                        + "\"string\" \n"
                                        + "], \n"
                                        + "\"tags\": [ \n"
                                        + "{ \n"
                                        + "\"id\": 0, \n"
                                        + "\"name\": \"Vatsa\" \n"
                                        + "} \n"
                                        + "], \n"
                                        + "\"status\": \"available\" \n"
                                        + "}")

                                .when().post("https://petstore.swagger.io/v2/pet");

  //log the Body

  res.then().log().body();

//validate status code and Message



      Assert.assertEquals(res.getStatusCode(), 200);
      System.out.println("Status Code is: " + res.getStatusCode());



      Assert.assertEquals(res.getStatusLine(), "HTTP/1.1 200 OK");
      System.out.println("Status Message is: " + res.getStatusLine());

  // JSON Path
```

```
            JsonPath json = res.jsonPath();
            expectedid = json.getInt("id");
            int Actualid = 111;

            //Validate the Result

            Assert.assertEquals(expectedid, Actualid);
            System.out.println("Generated ID is: " + expectedid);

    }
```

@Test (priority =3)
public void same_generated_id_find_the_pet() {

```
            System.out.println("\n\nSame Generated id find the Pet: ");

           Response res =  given()
                                        .header("accept","application/json")


          .when().get("https://petstore.swagger.io/v2/pet/"+expectedid );

           //log the Body

           res.then().log().body();

          //validate status code and Message


                Assert.assertEquals(res.getStatusCode(), 200);
```

```
                        System.out.println("Status Code is: " + res.getStatusCode());



                        Assert.assertEquals(res.getStatusLine(), "HTTP/1.1 200 OK");
                        System.out.println("Status Message is: " + res.getStatusLine());
}
```

@Test (priority =4)

public void Update_an_existing_pet() {

```
             System.out.println("\n\nUpdate an existing Pet: ");


            Response res = given()
                                        .header("accept","application/json")
                                        .header("Content-Type","application/json")
                                        .body("{ \n"
                                                    + "\"id\": 0, \n"
                                                    + "\"category\": { \n"
                                                    + "\"id\": 0, \n"
                                                    + "\"name\": \"Dev\" \n"
                                                    + "}, \n"
                                                    + "\"name\": \"Lion\", \n"
                                                    + "\"photoUrls\": [ \n"
                                                    + "\"string\" \n"
                                                    + "], \n"
                                                    + "\"tags\": [ \n"
                                                    + "{ \n"
                                                    + "\"id\": 0, \n"
                                                    + "\"name\": \"Gotu\" \n"
                                                    + "} \n"
                                                    + "], \n"
```

```
                                                  + "\"status\": \"available\" \n"
                                                  + "}")

                                        .when().put("https://petstore.swagger.io/v2/pet");

                    //log the Body

                     res.then().log().body();

                    //validate status code and Message


                          Assert.assertEquals(res.getStatusCode(), 200);
                          System.out.println("Status Code is: " + res.getStatusCode());



                          Assert.assertEquals(res.getStatusLine(), "HTTP/1.1 200 OK");
                          System.out.println("Status Message is: " + res.getStatusLine());

}
```

@Test (priority =5)
public void Delete_existing_pet_with_same_id() {

```
      System.out.println("\n\nDelete existing pet with same id: ");

     Response res =  given()
                                        .header("accept","application/json")

                                        .when().delete("https://petstore.swagger.io/v2/pet/"+expectedid);
```

```
        //log the Body

         res.then().log().body();

        //validate status code and Message


                Assert.assertEquals(res.getStatusCode(), 200);

                System.out.println("Status Code is: " + res.getStatusCode());



                Assert.assertEquals(res.getStatusLine(), "HTTP/1.1 200 OK");

                System.out.println("Status Message is: " + res.getStatusLine());

  }

}
```

Console Result of all Requests:

```
[RemoteTestNG] detected TestNG version 7.7.1
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Find Pet By Its Status
Status Code is: 200
Status Message is: HTTP/1.1 200 OK
Pets with status 'available':
Pet Name: doggie
Pet Name: doggie
Pet Name: doggie
```

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: Doggie

Pet Name: Doggie

Pet Name: doggie

Pet Name: fish

Pet Name: jimmy

Pet Name: fish

Pet Name: fish

Pet Name: fish

Pet Name: doggie

Pet Name: doie

Pet Name: New Pet 08/01/2025, 14:46:24

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: siyaan

Pet Name: Doggie

Pet Name: Doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: Doggy

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: doggie

Pet Name: Puff

Pet Name: newname

Pet Name: Doggie

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: Doggie

Pet Name: Doggie

Pet Name: fish

Pet Name: Puff

Pet Name: doggie

Pet Name: Doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: fish

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: snopy

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: doggie

Pet Name: MiaouMiaou

Pet Name: fish

Pet Name: doggie

Pet Name: Puff

Pet Name: cat

Pet Name: doggie

Pet Name: Tom

Pet Name: fish

Pet Name: doggie

Pet Name: cat

Pet Name: test

Pet Name: Doggie

Pet Name: doggie new

Pet Name: pig

Pet Name: Dinosaur

Pet Name: doggie

Pet Name: snopy

Pet Name: doggie

Pet Name: fish

Pet Name: fishtest

Pet Name: test

Pet Name: snopy

Pet Name: doggie

Pet Name: cat

Pet Name: fish

Pet Name: doggie

Pet Name: Puff

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: fish

Pet Name: fish

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: fish

Pet Name: Puff

Pet Name: doggie

Pet Name: doggie

Pet Name: siyaan

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: Puff

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: fish

Pet Name: fish

Pet Name: jimmy

Pet Name: doggie

Pet Name: fish

Pet Name: fish

Pet Name: doggie

Pet Name: Puff

Pet Name: doggie

Pet Name: fish

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: doggie

Pet Name: Tyrannosaurus

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: fish

Pet Name: Tyrannosaurus

Pet Name: fish

Pet Name: Tyrannosaurus

Pet Name: Puff

Pet Name: doggie

Pet Name: doggie

```
Pet Name: Tyrannosaurus
Pet Name: doggie
Pet Name: doggie
Pet Name: doggie
Pet Name: doggie
Pet Name: Tyrannosaurus
Pet Name: fish
Pet Name: fish
Pet Name: fish
Pet Name: fish
Pet Name: Tyrannosaurus
Pet Name: Tyrannosaurus
Pet Name: Tyrannosaurus
Pet Name: newname
Pet Name: fish
Pet Name: doggie
Pet Name: doggie
Pet Name: fish
Pet Name: doggie
Pet Name: Puff
Pet Name: doggie
Pet Name: Dog
Pet Name: Cat
Pet Name: Dog
Pet Name: Cat
Pet Name: fish
Pet Name: Doggie
Pet Name: doggie
Pet Name: fish
Pet Name: doggie
Pet Name: newname
```

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: doggie

Pet Name: fish

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: Puff

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: Tyrannosaurus

Pet Name: fish

Pet Name: fish

Pet Name: fish

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: fish

Pet Name: fish

Pet Name: Tyrannosaurus

Pet Name: Puff

Pet Name: doggie

Pet Name: Tyrannosaurus

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: Tyrannosaurus

Pet Name: doggie

Pet Name: klyopa

Pet Name: fish

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: akshay

Pet Name: fish

Pet Name: akanksha

Pet Name: klyopa

Pet Name: Postman

Pet Name: Tommi the dog

Pet Name: Sabertooth

Pet Name: tex

Pet Name: Sabertooth

Pet Name: newname

Pet Name: PasBre

Pet Name: xyz

Pet Name: Cat

Pet Name: Hillard

Pet Name: javaRush

Pet Name: doggie

Pet Name: doggie

Pet Name: fish

Pet Name: KuKu

Pet Name: Fluffy

Pet Name: CatTest

Pet Name: fish

Pet Name: doggie

Pet Name: Puff

Pet Name: KuKu

Pet Name: test

Pet Name: newname

Pet Name: doggie

Pet Name: newname

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: newname

Pet Name: Catty

Pet Name: Browny

Pet Name: kitty

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: Morty

Pet Name: doggie

Pet Name: nq4uxhsilU

Pet Name: Jimmy

Pet Name: doggie

Pet Name: Bilbo

Pet Name: pussy

Pet Name: doggie

Pet Name: Frodo

Pet Name: abhishek

Pet Name: wwluvvqroh

Pet Name: Sabertooth

Pet Name: pussy

Pet Name: doggie

Pet Name: ponpon

Pet Name: doggie

Pet Name: Connor

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: dummyDoggie

Pet Name: Sobaken

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: test

Pet Name: runneradmin

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: Jeromy

Pet Name: Floy

Pet Name: Candace

Pet Name: Andrew

Pet Name: Brenna

Pet Name: doggie

Pet Name: Fluffy

Pet Name: doggie

Pet Name: Jon

Pet Name: suri

Pet Name: tona

Pet Name: doggie

Pet Name: Doggo

Pet Name: Alis

Pet Name: Tommi the dog

Pet Name: Tommi the dog

Pet Name: Tommi the dog

Pet Name: Tommi the dog

Pet Name: doggie

Pet Name: doggie

Pet Name: Bobik

Pet Name: Lilika

Pet Name: Ignatius

Pet Name: Nemo

Pet Name: Ramson

Pet Name: Артур

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Pet Name: doggie

Test for Add New Pet to The Store:

```
{
  "id": 111,
  "category": {
    "id": 0,
    "name": "Devendra"
  },
  "name": "Tiger",
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "Vatsa"
    }
  ],
  "status": "available"
}
Status Code is: 200
Status Message is: HTTP/1.1 200 OK
Generated ID is: 111
```

Same Generated id find the Pet:

```
{
  "id": 111,
  "category": {
    "id": 0,
    "name": "Devendra"
  },
  "name": "Tiger",
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "Vatsa"
    }
  ],
  "status": "available"
}
Status Code is: 200
Status Message is: HTTP/1.1 200 OK
```

Update an existing Pet:

```
{
  "id": 9223372036854775807,
  "category": {
    "id": 0,
    "name": "Dev"
  },
  "name": "Lion",
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "Gotu"
```

```
        }
    ],
    "status": "available"
}
Status Code is: 200
Status Message is: HTTP/1.1 200 OK



Delete existing pet with same id:
{
    "code": 200,
    "type": "unknown",
    "message": "111"
}
Status Code is: 200
Status Message is: HTTP/1.1 200 OK
PASSED: com.Assignment.APIQuestion1.same_generated_id_find_the_pet
PASSED: com.Assignment.APIQuestion1.Find_pet_by_its_status
PASSED: com.Assignment.APIQuestion1.Update_an_existing_pet
PASSED: com.Assignment.APIQuestion1.Delete_existing_pet_with_same_id
PASSED: com.Assignment.APIQuestion1.Add_new_Pet_to_the_store

===============================================
    Default test
    Tests run: 5, Failures: 0, Skips: 0
===============================================



===============================================
Default suite
Total tests run: 5, Passes: 5, Failures: 0, Skips: 0
```

==

Q.2 Write an automation script to create POST Request using HashMap and POJO class.

Answer 02 A : An automation script to create POST Request using HashMap.

```
package com.BDD;


import org.testng.Assert;

import org.testng.annotations.Test;


import io.restassured.response.Response;


import static io.restassured.RestAssured.*;

import static io.restassured.matcher.RestAssuredMatchers.*;

import static org.hamcrest.Matchers.*;

import static org.testng.Assert.assertEquals;


import java.util.HashMap;


public class CreatePostUJsingHashMap {

  @Test

  public void hashmap() {


              //Request Payload
```

```
System.out.println("Post Using HashMap");

//child payload

HashMap<String,Object>map1 = new HashMap<String,Object>();

map1.put("year",2019);
map1.put("price",1849.99);
map1.put("CPUmodel","Intel Core i9");
map1.put("Harddisksize","1 TB");


//origional payload
HashMap<String,Object> map = new HashMap<String,Object>();
map.put("name", "Apple MacBook Pro 16");
map.put("data",map1);




Response res =  given()
                    .header("Content-Type", "application/json")
                    .body(map)

                    .when().post("https://api.restful-api.dev/objects");

//log the Result

res.then().log().body();
```

```
            //Validate Status

            Assert.assertEquals(res.getStatusCode(),200);
            System.out.println("Status Code is: " + res.getStatusCode());


            //jsonpath
            String actualid = "ff808181932badb601943abc57676b2e";
            String expectedid = res.jsonPath().get("id");

            //Validation Dynamic id & Print Dynamic id

           Assert.assertFalse(expectedid.contains(actualid));

            System.out.println("Generated id is: " + expectedid);



  }
}
```

Console Result For create POST Request using HashMap.

```
[RemoteTestNG] detected TestNG version 7.7.1
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Post Using HashMap
```

```
{
  "id": "ff808181932badb601944615c3aa01cb",
  "name": "Apple MacBook Pro 16",
  "createdAt": "2025-01-08T13:23:39.051+00:00",
  "data": {
    "CPUmodel": "Intel Core i9",
    "Harddisksize": "1 TB",
    "year": 2019,
    "price": 1849.99
  }
}
Status Code is: 200
Generated id is: ff808181932badb601944615c3aa01cb
PASSED: com.BDD.CreatePostUJsingHashMap.hashmap

===============================================
  Default test
  Tests run: 1, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
===============================================
```

Answer 02 B: An automation script to create POST Request using POJO class.

Script 01: Original Payload

```
package com.pojo;

public class PostUsingPOJO1 {


	private String name;
	private PostUsingPOJO2 data;
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public PostUsingPOJO2 getData() {
		return data;
	}
	public void setData(PostUsingPOJO2 data) {
		this.data = data;
	}

}
```

Script 02: Child PayLoad

```
package com.pojo;

public class PostUsingPOJO2 {
```

```
private int year;
private double price;
public int getYear() {
	return year;
}
public void setYear(int year) {
	this.year = year;
}
public double getPrice() {
	return price;
}
public void setPrice(double price) {
	this.price = price;
}
public String getCPUmodel() {
	return CPUmodel;
}
public void setCPUmodel(String cPUmodel) {
	CPUmodel = cPUmodel;
}
public String getHarddisksize() {
	return Harddisksize;
}
public void setHarddisksize(String harddisksize) {
	Harddisksize = harddisksize;
}
private String CPUmodel;
private String Harddisksize;
```

```
}
```

Script 03: POST Request using TestNG POJO class

```
package com.BDD;

import static io.restassured.RestAssured.given;

import org.testng.annotations.Test;

import com.pojo.PostUsingPOJO1;
import com.pojo.PostUsingPOJO2;

import io.restassured.response.Response;

import static io.restassured.RestAssured.*;
import static io.restassured.matcher.RestAssuredMatchers.*;
import static org.hamcrest.Matchers.*;
import static org.testng.Assert.assertEquals;

import org.testng.Assert;

public class CreatedPostUsingPOJO {
  @Test
  public void pojo() {

        System.out.println("Post Call Using POJO");
        //Request Payload

        //Child Payload
```

```
 PostUsingPOJO2 data = new PostUsingPOJO2();

 data.setYear(2019);
 data.setPrice(1849.99);
 data.setCPUmodel("Intel Core i9");
 data.setHarddisksize("1 TB");

 //origional payload


 PostUsingPOJO1 payload = new PostUsingPOJO1();

 payload.setName("Apple MacBook Pro 16");
 payload.setData(data);

Response res=  given()
                        .header("Content-Type","application/json")
                        .body(payload)

                        .when().post("https://api.restful-api.dev/objects ");


//log the Result

res.then().log().body();


//validate Status

Assert.assertEquals(res.getStatusCode(), 200);
```

```
            System.out.println("Status Code is: " + res.getStatusCode());
            System.out.println("Status Line is: " + res.getStatusLine());

            //validate id & print id

            String actualid= "ff808181932badb601943afa258b6c63";
            String expectedid = res.jsonPath().getString("id");

            Assert.assertFalse(expectedid.contains(actualid));

            System.out.println("Dynamic Genereted id is: " + expectedid);


  }
}
```

Console Result For create POST Request using POJO.

```
[RemoteTestNG] detected TestNG version 7.7.1
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Post Call Using POJO
{
  "id": "ff808181932badb601944627eea00202",
  "name": "Apple MacBook Pro 16",
  "createdAt": "2025-01-08T13:43:29.696+00:00",
  "data": {
    "year": 2019,
    "price": 1849.99,
        "cpumodel": "Intel Core i9",
        "harddisksize": "1 TB"
    }
}
Status Code is: 200
Status Line is: HTTP/1.1 200
Dynamic Genereted id is: ff808181932badb601944627eea00202
PASSED: com.BDD.CreatedPostUsingPOJO.pojo

===============================================
    Default test
    Tests run: 1, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
===============================================
```

Q.03: Write a test script to verify the authentication process of an API using OAuth 2.0.

Answer 03: Architecture Overview

The architecture consists of interactions among the above entities, using secure communication protocols (typically HTTPS). OAuth 2.0 defines four types of grant flows, but the **Authorization Code Grant** is the most common. Below is its architecture:

Step-by-Step Process

1. Client Registration

- The client (application) registers with the Authorization Server.
- The Authorization Server issues:
  - A **Client ID** (public identifier for the app).
  - A **Client Secret** (private key, if confidential).

---

2. Request Authorization

- The Client redirects the Resource Owner (User) to the Authorization Server.
- The Resource Owner authenticates (e.g., logs in) and grants permission to the Client.

---

3. Authorization Grant

- The Authorization Server provides the Client with an **Authorization Code** (short-lived and single-use).

---

4. Exchange Code for Token

- The Client exchanges the Authorization Code for an **Access Token** by sending a request to the Authorization Server.
- The request includes:
  - Authorization Code.
  - Client ID and Secret.
  - Redirect URI (to ensure security).

---

5. Access Resource

- The Client uses the Access Token to request resources from the Resource Server.
- The Resource Server validates the Access Token with the Authorization Server (if needed).

---

6. Response with Resource

- Upon successful validation of the Access Token, the Resource Server provides access to the requested resource.

Script: Automation Script for OAuth2.0 Using GitHub

```
package com.Authentication;

import org.testng.annotations.Test;

import io.restassured.response.Response;

import static io.restassured.RestAssured.*;
import static io.restassured.matcher.RestAssuredMatchers.*;
import static org.hamcrest.Matchers.*;
import static org.testng.Assert.assertEquals;

import org.testng.Assert;

public class OAuth2 {
  @Test
  public void OAuth() {

          System.out.println("Test of an API using OAuth 2.0.");

          String oauthtoken =
"github_pat_11BOFWZYQ0Ach8AXEHRr7l_LclEn3ztifCWUlVNd3s8FdFLfZ4VoDUZhTtBe0qD9srP6R5G
MFTPr6ogwrL";

         Response res =  given()
                                        .auth().oauth2(oauthtoken)
                                      .when().get("https://api.github.com/user/repos");



          //validate Status

          Assert.assertEquals(res.getStatusCode(), 200);
          System.out.println("Status Code is: " + res.getStatusCode());
          System.out.println("Status Line is: " + res.statusLine());



  }
}
```

Console Result of OAuth2.0

```
[RemoteTestNG] detected TestNG version 7.7.1
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Test of an API using OAuth 2.0.
Status Code is: 200
Status Line is: HTTP/1.1 200 OK
PASSED: com.Authentication.OAuth2.OAuth

===============================================
    Default test
    Tests run: 1, Failures: 0, Skips: 0
```

```
===============================================



===============================================
Default suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
===============================================
```

_______*************_______________*************_____________*********_____

Thank You

Devendra Kumar Vatsa

Vatsadev081@gmail.com

8629994177

_____________*************_______________*************_____________*********______