Nataraj Perisetty

Abstract

This document describes QA automation process for Swagger PetStore Business Requirements. And document also consists of test plans, Test cases and Automation test process

QA Engineer techincal chalLenge

Contents

[1 Document History 3](#_Toc4732320)

[2 Approvals 3](#_Toc4732321)

[3 Introduction 3](#_Toc4732322)

[3.1 Summary 4](#_Toc4732323)

[3.1.1 Objectives 4](#_Toc4732324)

[3.1.2 Scope 4](#_Toc4732325)

[3.1.3 Out of Scope Functionality 4](#_Toc4732326)

[3.2 Test Approach 5](#_Toc4732327)

[3.2.1 Manual testing: 5](#_Toc4732328)

[3.2.2 Automation testing: 5](#_Toc4732329)

[3.3 Business requirements: 5](#_Toc4732330)

[3.3.1 User Story 1: Find Pet in store. 5](#_Toc4732331)

[3.3.2 User Story 2: Add a new Pet to Store 6](#_Toc4732332)

[3.3.3 User Story 3: Delete Pet from store 6](#_Toc4732333)

[3.4 Test bed setup: 7](#_Toc4732334)

[3.5 Test cases for User Story 1: Find Pet in store. 8](#_Toc4732335)

[3.6 Automation Test cases for User Story 1: Find Pet in store. 14](#_Toc4732336)

# Document History

|  |  |  |
| --- | --- | --- |
| **Date** | **Version** | **Comments** |
| 25/03/2019 | 1 | Draft Test plan/Specification document |
|  |  |  |

# Approvals

|  |  |  |
| --- | --- | --- |
| **Role** | **Name** | **Date** |
| Project Sponsor | ABC | 28/03/2019 |
| Business Owner | XYZ | 28/03/2019 |
| Project Manager | 123 | 29/03/2019 |

# Introduction

## Summary

### Objectives

The aim of this document is to specify the Test Plan for sample micro service which is Petstore server in swagger.

This document describes the approaches and methodologies that will apply to functional tests to test “<https://petstore.swagger.io>”

This document includes the objectives, test responsibilities, Acceptance criteria, scope and Automation approach

### Scope

The document aims towards manual API testing and automation of API parameters as per business requirements.

Functions to be tested Manual: end points are listed below

PET (Everything about your Pets)

POST /pet (Add a new pet to store)

GET /pet/{pet id}

GET /pet/{pet findbystatus}

POST /pet{pet id}

DELETE /pet{pet id}

Functions to be tested with Automation: end points are listed below

PET (Everything about your Pets)

POST /pet (Add a new pet to store)

GET /pet/{pet id}

DELETE /pet{pet id}

All the responses are made with json type.

### Out of Scope Functionality

Not other than mentioned above in section

## Test Approach

### Manual testing:

Pet store is sample REST service. The definition is informat Open API 2.0. Hence To test REST service ReadyAPI or Postman tools can be used.

For each functional test coverage can be extended to but not limited to validation of response status code, header parameter and presence of mandatory tag etc..

Petstore swagger definition is added below.

### Automation testing:

Behavioral Data Driven (BDD) framework can be build up to monitor the smooth run of APIs through an automated way. Below are the pre-requisites to achieve automation process.

Maven-Cucumber Framework

Rest Assured or

Extent Reporting or Jenkins reporting

JXL jar configuration to perform Excel Read/Write Operations or Apache POI jars

Jenkins integration to achieve CI

## Business requirements:

Business requirements for Petstore API as follows only 3 requirements are considered.

### User Story 1: Find Pet in store.

The requirement is User should be able to find a Pet in petstore, he/she should be able to do it by either by Pet ID or by with availability of Pets in store.

Only parameter required is pet ID if find operation is done with Pet ID.



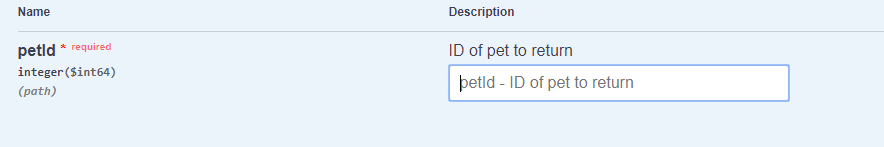
Only parameter required is availability i.e. available or pending or sold if find operations is done with FindByStatus in store.



**Acceptance criteria:**

In case find Pet request is made with Pet ID- Check response code for 200 is returned.

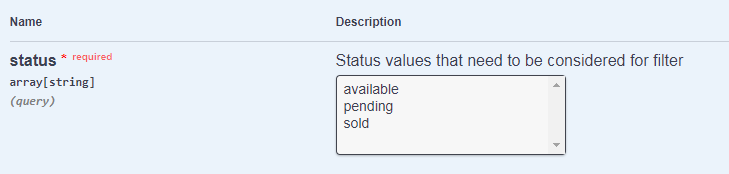
Check response/json data contains proper requested ID and all information corresponds to requested Pet.



**Error scenario:** Enter invalid Pet ID in request, i.e. enter a pet ID which is not available,

Check response code. Response should be 404 with error code as not found.

In case find Pet request is made with findByStatus- Check response code for 200 is returned.

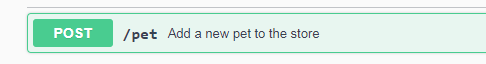


Check response/json data contains proper requested ID and all information corresponds to requested Pet.

Check response time is lesser than 1 sec.

### User Story 2: Add a new Pet to Store

The requirement is User should be able to add a new Pet in petstore, he/she should be able to do it by providing new Pet ID and Pet Name.



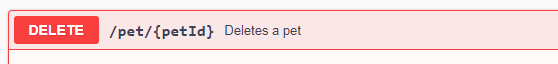
**Acceptance Criteria:**

In case new Pet is created, check the response code 200

**Error Scenario**: In case new pet is not created due to invalid input then check response code 405

### User Story 3: Delete Pet from store

The requirement is user should be able to delete the existing pet from store. He/she should be able to do it by providing only Pet ID.



Acceptance Criteria:

In case Pet is deleted, check the response code 200.

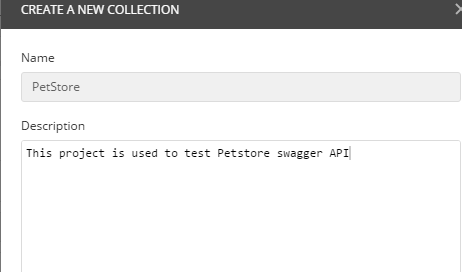
In case Pet is not available, check for pet not found code 404

**Error Scenario:** In case Pet ID is invalid input, check for Invalid IP supplied code 400

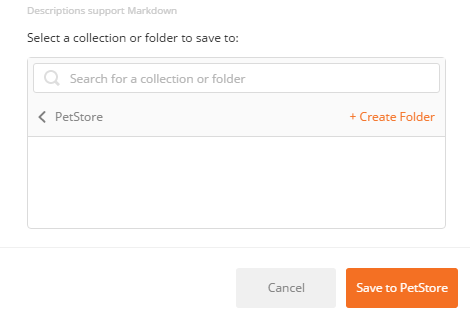
## Test bed setup:

For manual testing: Postman will be used to Test API.

Step1: Create new project “PetStore”

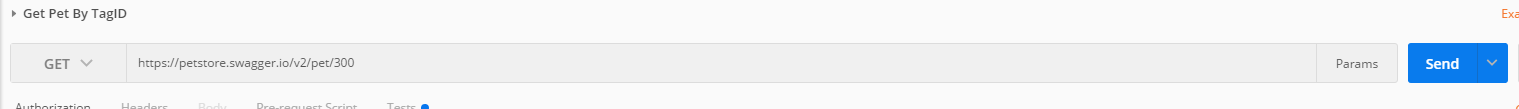


Step 2: Set collection of test cases to PetStore folder.



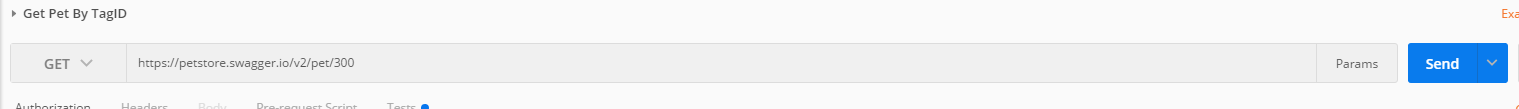
## Test cases for User Story 1: Find Pet in store.

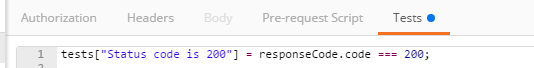
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Name** | **Description** | **Expected Result** | **Priority** |
| 001\_Find Pet with Tag ID as 300 | From postman, create a new test  Use GET request with URL as  <https://petstore.swagger.io/v2/pet/300>  Select send request  Check Body | Json format/xml format data should be retrieved. | P0 |

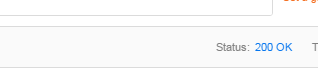


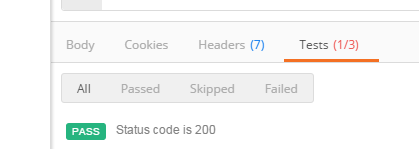


|  |  |  |  |
| --- | --- | --- | --- |
| **Test Name** | **Description** | **Expected Result** | **Priority** |
| 002\_Check status response code 200 for Get Pet by Tag ID | From postman,  Use GET request with URL as  <https://petstore.swagger.io/v2/pet/300>  From Tests field add below command.  tests["Status code is 200"] = responseCode.code === 200;  Select send request  Check Body, verify success code test from Tests | When request is successful, 200 code should return. | P0 |

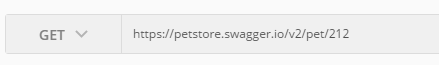








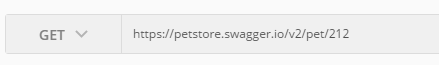
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Name** | **Description** | **Expected Result** | **Priority** |
| 003\_Check data contains proper values i.e. verify pet name form the response | From postman,  Use GET request with URL as  <https://petstore.swagger.io/v2/pet/300>  From Tests field add below command.  varjsonData = JSON.parse(responseBody);  tests["Name of Pet"] = jsonData.name === "neee";Select send request  Check Body, and verify pet name test case from Tests | When request is successful, Pet name should match with name === "neee"; | P0 |







|  |  |  |  |
| --- | --- | --- | --- |
| **Test Name** | **Description** | **Expected Result** | **Priority** |
| 004\_Check response time is with in valid duration | From postman,  Use GET request with URL as  <https://petstore.swagger.io/v2/pet/300>  From Tests field add below command.  tests["Response time is less than 200ms"] = responseTime< 300;  Check Body, and verify time take taken to complete the response. | Time taken to complete the response should be with in acceptable duration.(<300 ms) | P0 |





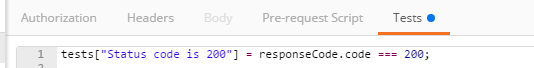




Test case failed because of actual time to complete the response is greater than 300ms.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Name** | **Description** | **Expected Result** | **Priority** |
| 005\_find pet By Status ‘available’ | From postman,  Use GET request with URL as  <https://petstore.swagger.io/v2/pet/findByStatus?status=available>  From Tests field add below command.  tests["Status code is 200"] = responseCode.code === 200;  Check Body, and verify all available pets data is retrieved | Json format/xml format data should be retrieved with all available pets information | P0 |

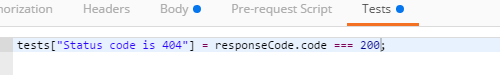


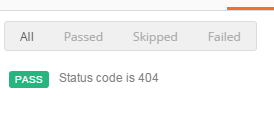




Similarly tests 2 to 4 can be executed with various combinations of availability.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Name** | **Description** | **Expected Result** | **Priority** |
| 006\_Error Scenario: Verify Tag By ID when ID is not present. Verify 404 code | From postman,  Use GET request with URL as  <https://petstore.swagger.io/v2/pet/300>  Select send request  Check Body, Verify for 404 code. | Check for failure of test since no pet is available with requested ID. | P0 |





Test report:

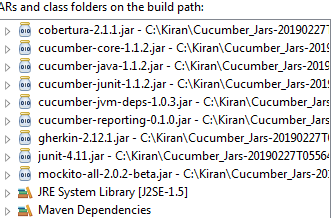
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test campaign** | **Total Tests Executed** | **Passed** | **Failed** | **Comments** |
| User Story 1: Find Pet in store | 6 | 5 | 1 | Tests related to response time is failed. |

## Automation Test cases for User Story 1: Find Pet in store.

Automation test cases to get pet By ID by BDD:

Cucumber feature is defined as below:

Create a maven project and add build path additional jars which are required to cucumber



Make sure cucumber core version is 1.2.3, so that annotations work properly.

A new feature file is defined with below scenario under resource folder.

Feature: Get pet by ID

# Given the Swagger definition at

# http://petstore.swagger.io/v2/swagger.json

Scenario: Get pet by ID

Given User with ID 300

When user retrieves the details by id

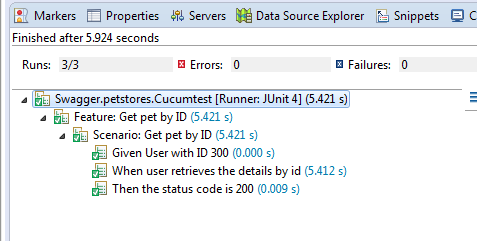
Then the status code is 200

**Pom.xml:**

Add all dependent jars to pom.xml file as shown below,



**Test Results:**



Similar approach can be used to implement remaining API test scenarios.

And results can be enhanced and published to extent reporting or Jenkins reporting

And in case inputs are maintained in external xls file, JXL jar configuration can be done to perform Excel Read/Write Operations or Apache POI jars

**End of Document**