**Test Plan Document**

Petstore API Validation

**Project Name:** Petstore API Validation  
**Version:** 1.0  
**Date:** 18th November 2024  
**Prepared by:** Azri Mangsor  
**Approval Status:** Pending Approval

Table of Contents

[1. Document Control 3](#_Toc182927336)

[1.1. Document Version History 3](#_Toc182927337)

[1.2. Document Review History 3](#_Toc182927338)

[2. Introduction 3](#_Toc182927339)

[2.1. Scope 3](#_Toc182927340)

[2.2. Objective 3](#_Toc182927341)

[2.3. Assumptions 3](#_Toc182927342)

[2.4. Risks 5](#_Toc182927343)

[3. Test Approach 7](#_Toc182927344)

[3.1. Test Design Techniques 7](#_Toc182927345)

[3.2. Levels of Testing 7](#_Toc182927346)

[4. Test Deliverables 7](#_Toc182927347)

[5. Test Environment 7](#_Toc182927348)

[5.1. Hardware 7](#_Toc182927349)

[5.2. Software 7](#_Toc182927350)

[5.3. Test Data 8](#_Toc182927351)

[6. Staffing 8](#_Toc182927352)

[7. Test Entry & Exit Criteria 8](#_Toc182927353)

[7.1. Entry Criteria 8](#_Toc182927354)

[7.2. Exit Criteria 8](#_Toc182927355)

[7.3. Suspension Criteria 8](#_Toc182927356)

[7.4. Resume Criteria 9](#_Toc182927357)

[8. Schedule 9](#_Toc182927358)

[9. Test Plan Sign-Off 9](#_Toc182927359)

[Annex A: Test Design Specification 10](#_Toc182927360)

[A.1 Test Cases Overview 10](#_Toc182927361)

# Document Control

## Document Version History

This table shows records of changes made to the document.

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Descriptions** |
| 1.0 | 19/11/2024 | Azri Mangsor | Initial |

## Document Review History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Reviewer** | **Description** |
|  |  |  |  |  |
|  |  |  |  |  |

# Introduction

This Test Plan outlines the approach, resources, and schedule for the validation of the Petstore API (https://petstore.swagger.io/v2) in accordance with ISO/IEEE 29119 standards.

## Scope

|  |  |
| --- | --- |
| **In Scope** | **Out Off Scope** |
| **Authorization:** Verifying endpoint access with and without an API key. | **UI** testing of Swagger Petstore documentation |
| **CRUD Operations:** End-to-end testing of create, read, update, and delete operations for /pet. | Other API groups such as /store and /user |
| **Data Validation:** Ensuring responses match defined schema (petSchema.json). |  |
| **Error Handling:** Testing API responses for invalid inputs and edge cases |  |

## Objective

The objective is to ensure that the API meets functional, non-functional, and contract compliance requirements.

## Assumptions

1. API Key Functionality:

* The provided API key (special-key) is valid and has the required permissions for all API operations (e.g., CRUD).
* The API key usage follows rate-limiting rules (if applicable).

1. Environment Availability:

* The Petstore API endpoints (https://petstore.swagger.io/v2) are publicly accessible and stable during the testing period.
* There are no planned or unexpected downtimes during the testing phase.

1. Schema Consistency:

* The API responses adhere to the schema defined in the Swagger documentation without unexpected changes or omissions.

1. Test Data Setup:

* Test data used for CRUD operations (e.g., petId and category) can be successfully created, retrieved, updated, and deleted without affecting other users or causing conflicts.

1. Performance Benchmarks:

* Response time thresholds for performance testing align with industry standards for publicly accessible APIs.

1. Third-Party Dependencies:

* Any external resources (e.g., images in photoUrls) used in the test scenarios are reliably accessible.

## Risks

The following table provides a comprehensive risk assessment for the test plan. Each risk is analyzed based on its severity (seriousness of the impact) and the likelihood of its occurrence. Additionally, mitigation strategies are defined for each identified risk to address potential issues effectively.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Risk ID** | **Risk Description** | **Impact** | **Likelihood** | **Severity Level** | **Mitigation Strategies** |
| R1 | Environment Instability | API might be unavailable due to downtime, maintenance, or high traffic. | Medium | High | Monitor API availability; schedule tests during low-traffic hours; notify stakeholders. |
| R2 | Data Persistence Issues | Test data might not persist or conflict with other testers accessing the public API. | Medium | Medium | Use unique identifiers in test data to avoid conflicts; clean up data after tests. |
| R3 | Schema Changes | Unexpected changes in API schema could cause schema validation tests to fail. | Low | High | Regularly review the Swagger documentation; automate schema validation updates. |
| R4 | Security Restrictions | Invalid or expired API keys could block authorized tests. | Medium | High | Validate API keys before testing; include fallback handling for token expiration. |
| R5 | Performance Degradation | High concurrency could slow response times, leading to inaccurate performance results. | Medium | Medium | Perform load testing during off-peak hours; simulate realistic user loads. |
| R6 | Rate Limiting and Quotas | Excessive requests could trigger rate-limiting, interrupting tests. | Medium | High | Implement retry mechanisms and ensure tests comply with API limits. |
| R7 | Unsupported Edge Cases | The API may not handle invalid data gracefully, resulting in unexpected failures. | Medium | Low | Define edge cases explicitly and validate error handling in API responses. |
| R8 | Third-Party Dependency Issues | External resources (e.g., images) might be inaccessible during testing. | Low | Medium | Use mock or predefined URLs for external resources; check resource availability first. |
| R9 | Limited Control Over Public API Behavior | Dependency on API design may limit testing flexibility (e.g., rate-limiting configurations). | Medium | Medium | Coordinate with API maintainers or set realistic test scenarios. |
| R10 | Data Privacy Concerns | Sensitive test data might become publicly accessible if not anonymized. | Low | High | Use mock or non-sensitive data; validate input fields for anonymization. |

# Test Approach

## Test Design Techniques

* ***Equivalence Partitioning:*** Test different categories of valid and invalid inputs.
* ***Boundary Value Analysis:*** Ensure endpoints handle edge cases effectively.
* ***Contract Testing:*** Validate API responses against the defined schema using Ajv.

## Levels of Testing

* ***Component Testing:*** Test individual endpoints (e.g., /pet, /pet/{id}).
* ***Integration Testing:*** Validate interactions between the CRUD operations and the API database.

# Test Deliverables

* **Test Plan (This document)**
* **Test Design Specifications** (Annex A)
* **Test Scripts:** Implemented in TypeScript.
* **Execution Logs:** Recorded in Cypress.
* **Defect Reports (if applicable).**

# Test Environment

## Hardware

Local machine with a stable internet connection.

## Software

|  |  |  |
| --- | --- | --- |
| **Tool/Component** | **Version** | **Purpose** |
| Cypress | 12.12 | Test execution framework. |
| TypeScript | 4.x | Scripting language for tests. |
| Ajv | Latest | JSON schema validation library. |

## Test Data

* **petSchema.json:** Schema for the /pet object.
* **petData.json:** Dynamic storage for pet IDs during testing.

# Staffing

|  |  |  |
| --- | --- | --- |
| **Role** | **Name** | **Responsibility** |
| Test Manager | John Doe | Approve test plan, manage resources. |
| Test Analyst | Jane Doe | Develop and execute test cases. |

# Test Entry & Exit Criteria

The test entry, exit, and suspension criteria outline the conditions for initiating, concluding, and temporarily pausing testing activities. Testing begins when preconditions such as test environment readiness, stable builds, and signed-off test plans are met. It concludes when objectives like executing all planned test cases, resolving critical defects, and meeting coverage thresholds are achieved. Testing is paused if critical blockers, environment instability, or unavailable dependencies arise, with defined protocols for resuming once issues are resolved. Together, these criteria ensure a structured approach to maintaining the efficiency and focus of the testing process.

## Entry Criteria

* API is accessible at https://petstore.swagger.io/v2.
* Test environment and fixtures (e.g., petSchema.json, petData.json) are set up.

## Exit Criteria

* All planned test cases have been executed.
* Zero high-severity defects remain open.
* Response schemas for all endpoints pass validation.

## Suspension Criteria

* API downtime or connectivity issues exceeding 1 hour.
* Critical issues in test environment setup.

## Resume Criteria

Test stop because of suspension will resume only when the problem (s) that caused the suspension has been resolved.

# Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task Name** | **Description** | **Start Date** | **End Date** | **Estimae Hours + / -** | **Notes** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Test Plan Sign-Off

|  |  |  |
| --- | --- | --- |
|  | **Project Manager** | **QA Manager** |
| **Name** |  |  |
| **Signature** |  |  |
| **Date** |  |  |

# Annex A: Test Design Specification

## A.1 Test Cases Overview

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Objective** | **Input/Actions** | **Expected Outcome** |
| TC-001 | Unauthorized access | GET /pet/findByStatus (no key) | 401 Unauthorized. |
| TC-002 | Add pet with valid data | POST /pet (valid payload) | 200 OK, pet created. |
| TC-003 | Fetch pet by ID | GET /pet/{id} | 200 OK, correct pet data. |
| TC-004 | Schema validation | GET /pet/{id} | Valid schema. |