TEST PLAN

- GAME OF THRONES QUOTES API -

Valpovo, May 2024.

Table of Contents

[1. Introduction 3](#_Toc167738414)

[2. Scope of Testing 3](#_Toc167738415)

[3. Testing Environment Setup 3](#_Toc167738416)

[3.1. Tools and Frameworks 3](#_Toc167738417)

[3.2. Installation and Setup 3](#_Toc167738418)

[4. Dependencies 4](#_Toc167738419)

[5. Test Execution 4](#_Toc167738420)

[6. Test Plan Outline 4](#_Toc167738421)

[7. Additional Considerations 6](#_Toc167738422)

[8. Conclusion 6](#_Toc167738423)

# 1. Introduction

This document outlines a test plan for automated testing of the Game of Thrones Quotes API (<https://gameofthronesquotes.xyz/>). The test plan includes the scope, testing methodologies, environment setup, different dependencies, and responsibilities. The objective is to ensure the API’s functionality, reliability, and error handling through systematic testing.

# 2. Scope of Testing

|  |  |
| --- | --- |
| Test type | Description |
| Functionality Testing | Verify API endpoints for retrieving quotes randomly, by character, and by episode. |
| Compatibility Testing | Ensure the API consistently returns the expected responses. |
| Error Handling | Validate appropriate error responses for invalid requests. |
| Performance Testing | Check the API’s performance under various loads. |

# 3. Testing Environment Setup

## 3.1. Tools and Frameworks

* Postman
  + Tool for API testing and automation.
* Newman
  + Command-line tool for running Postman collections
* Node.js
  + JavaScript runtime for executing tests
* Mocha and Chai
  + Testing frameworks for writing and running test cases
* Swagger
  + API documentation tool to understand and verify API endpoints and responses

## 3.2. Installation and Setup

* Install Postman and configure API endpoints
* Install Node.js and npm
* Install Newman globally using npm
* Set up Mocha and Chai for test scripting

# 4. Dependencies

* Internet access for API endpoint requests
* Access to the API documentation for endpoint details
* Test data including valid and invalid character names, season, and episode numbers

# 5. Test Execution

* Tester: Branimir Križanec
* Responsibilities:
  + Write and execute test cases
  + Document and track test results
  + Analyze and report any issues found during testing
* Execution Schedule:
  + Tests will be executed after each API update
  + Regular testing to ensure API stability and performance

# 6. Test Plan Outline

The following tables serve to roughly outline the Test Scenarios for the web page: [https://api.gameofthronesquotes.xyz](https://api.gameofthronesquotes.xyz/v1/random)

|  |  |  |  |
| --- | --- | --- | --- |
| # | Scenario | Test Case | Expected Result |
| 1. | **Retrieve Random Quote** | 1.1: Send a GET request to ‘/v1/random’ | Status code 200 (OK)  Response body contains a random object (quote).  Each quote has the following properties: *sentence, character.name, character.slug, character.house.name, character.house.slug* |
| 2. | **Retrieve Multiple Random Quotes** | 2.1: Send a GET request to ‘/v1/random/{number}’ | Status code 200 (OK)  Response body contains an array of objects (quotes). Each element of the array has the following properties:  *sentence, character.name, character.slug, character.house.name, character.house.slug* |
| 2.2 Send a GET request to ‘/v1/random/{number}’ with an invalid number (e.g., negative number or non-numeric value) | Response status code is 400 (Bad Request) |
| 3. | **Retrieve Quotes by Character** | 3.1: Send a GET request to ‘v1/author/{character}’ with a valid character name | Status code 200 (OK)  Response body contains an object (quote) related to the specified character.  Each quote has the following properties: *sentence, character.name, character.slug, character.house.name, character.house.slug* |
| 3.2: Send a GET request to ‘/v1/author/{character}/{number} with a valid character name and a valid number | Status code 200 (OK)  Response body contains an array of objects (quotes) for a character. Each element of the array has the following properties:  *sentence, character.name, character.slug, character.house.name, character.house.slug* |
| 3.3: Send a GET request to ‘/v1/author/{character}/{number} with a valid character name and an invalid number  NOTE:  *The {number} parameter is optional, and by default the response body returns 1 object (quote). When the endpoint contains an invalid number, the API returns the default response body.* | Status code 200 (OK)  Response body contains an object (quote) related to the specified character.  Each quote has the following properties: *sentence, character.name, character.slug, character.house.name, character.house.slug* |
| 3.4: Send a GET request to ‘v1/author/{character}’ with an invalid character name  NOTE:  *The character parameter does not have a way of handling incorrect inputs.* | Response body is empty. |
| 4 | **Retrieve All Characters** | 4.1: Send a GET request to ‘/v1/characters’ | Status code is 200 (OK)  Response body contains an array of characters.  Each character object contains ‘name’ and ‘slug’ fields. |
| 5. | **Retrieve All Houses** | 5.1: Send a GET request to ‘/v1/houses’ | Status code is 200 (OK)  Response body contains an array of houses.  Each element of the array has the following properties: *slug, name.* The array also has a nested array of all the members (objects) from the house. |

# 7. Additional Considerations

* Data-Driven Testing: Use data driven approaches to test with multiple sets of input data
  + Implement data-driven tests using Mocha’s ‘describe’ and ‘it’ functions along with data from external sources like JSON files

# 8. Conclusion

This test plan serves as a foundational guide for automated testing of the Game of Thrones Quotes API. By following the outlined steps and considering additional factors, we aim to ensure the API’s reliability, performance, and error handling. Regular reviews and updates to the test plan will help maintain its effectiveness and relevance as the project evolves. Detailed test scenarios are supplemented in separate documents or tools such as Postman collections.