SOFTWARE TEST DESCRIPTION

(STD)

NASA API website

{NASA APIs}

Nidaa AbuSaleh

TABLE OF CONTENTS

INTRODUCTION

1. Purpose:

The purpose of this Software Test Plan (STP) is to establish a framework for conducting comprehensive QA automation tests on the NASA API website (https://api.nasa.gov/). The STP is designed to ensure that the API meets predefined quality standards, with a focus on identifying and effectively addressing any potential defects.

2.scope:

The project for the NASA API website encompasses critical objectives, emphasizing key functionalities and user interactions. The scope of testing includes:

- Endpoint Functionality: Validate the correctness and reliability of API endpoints, ensuring accurate data retrieval.
- Data Accuracy: Verify the precision and consistency of data obtained from different API endpoints.
- Documentation Clarity: Assess the clarity and completeness of API documentation to facilitate developers' understanding and implementation.
- Response Time: Evaluate the response time of API requests under normal and peak load conditions to ensure optimal performance.
- Compatibility: Validate the API's compatibility across various programming languages and environments.
- Authentication and Authorization: Ensure effective authentication and authorization mechanisms to control access to restricted data and functionalities.
- Data Format Support: Test the API's ability to handle different data formats (e.g., JSON, XML) during requests and responses.

- Documentation Examples: Validate the accuracy and relevance of code examples provided in the API documentation for seamless integration.

.OVERVIEW

1.Background:

The NASA API website, available at https://api.nasa.gov/, serves as a gateway for developers and enthusiasts to access a rich collection of data and resources provided by NASA. This API offers programmable access to various information, including imagery, astronomy data, and space-related content. Developers can leverage the API to retrieve details about space missions, celestial bodies, and other astronomical phenomena, contributing to educational and research purposes.

2. Goals:

Ensuring API Reliability and Functionality:

- Striving for an API that operates seamlessly, providing accurate and reliable data to developers.
- Conducting comprehensive testing to identify and rectify any potential issues, ensuring a robust and dependable API experience.

Cross-Platform Compatibility:

- Verifying and optimizing the NASA API for compatibility across different programming languages and environments.
- Ensuring consistent performance and functionality regardless of the development environment used by developers.

Developer-Focused User Experience (UX):

- Focusing on continuous improvement of the overall developer experience when interacting with the API.
- Emphasizing clear and comprehensive API documentation, intuitive API design, and ease of integration to cater to a diverse developer community.

Data Accuracy and Consistency:

- Ensuring the accuracy and consistency of data retrieved from various API endpoints to support reliable research and development efforts.
- Providing developers with dependable and up-to-date information through the API.

3.Glossary for NASA API Website:

- Endpoint:

  A specific URL within the NASA API that serves as a gateway for retrieving specific data or performing actions programmatically.

- Documentation:

  Comprehensive written material accompanying the NASA API, providing details on available endpoints, request formats, response structures, and example use cases to guide developers.

- Authentication:

  The process of verifying the identity of a user or application interacting with the NASA API to ensure secure access to restricted data and functionalities.

- Authorization:

The process of granting specific permissions and access rights to authenticated users or applications based on their roles and responsibilities.

- API User:

  An individual or application interacting with the NASA API to retrieve information, contributing to research, education, and development efforts.

Test Cases

1. Endpoint Access:
   - Verify that developers can access specific API endpoints with valid credentials.
   - Confirm that the API returns the expected data in response to valid requests.

2. Data Accuracy:
   - Verify the accuracy and consistency of data retrieved from different API endpoints.
   - Confirm that the data corresponds to the specified parameters and is up-to-date.

3. Documentation Validation:
   - Verify the clarity and completeness of API documentation.
   - Confirm that the documentation accurately reflects available endpoints, request formats, and response structures.

4. Authentication and Authorization:
    - Verify that users can successfully authenticate and receive authorization to access restricted data and functionalities.
    - Confirm that unauthorized users are denied access to sensitive information.

5. Response Time:
    - Evaluate the response time of API requests under normal load conditions.
    - Confirm that the API responds efficiently to deliver timely data.

6. Data Format Support:
    - Test the API's ability to handle different data formats (e.g., JSON) during both requests and responses.
    - Confirm that the API supports the specified data formats without errors.

7. Code Examples:
    - Validate the accuracy and relevance of code examples provided in the API documentation.
    - Confirm that developers can successfully implement API requests using the provided code snippets.