**Test Strategy & Plan: Reqres API Testing**

This document outlines the testing strategy and plan for the Reqres API, a fake API used for testing and prototyping. The provided test suite is built using Java and the RestAssured library to validate the functionality of the API's user management features.

**1. What is Being Tested**

This test plan covers the functional testing of the Reqres API endpoints related to user management. The tests verify the correct behavior of the following API operations:

* **User Registration:** Successful and unsuccessful user registration.
* **User Data Retrieval:** Fetching lists of users.
* **User Creation:** Creating a new user.
* **User Updates:** Modifying an existing user's information.
* **User Deletion:** Removing a user.

The tests focus on validating:

* HTTP status codes for different requests.
* The structure and content of JSON responses.
* Data integrity after create, update, and delete operations.

**2. Test Coverage Areas**

The test suite aims to provide comprehensive coverage across the following areas of the Reqres API:

* **Happy Path Testing:** This ensures the API functions as expected under normal conditions. This includes:
  + Successful user registration with valid credentials.
  + Creating a new user with all required fields.
  + Updating an existing user.
  + Retrieving a list of users and verifying their data.
  + Successfully deleting a user.
* **Negative Testing:** This validates how the API handles error conditions and invalid input. This includes:
  + Attempting to register a user with a missing password.
* **Data Validation:** This ensures the integrity and correctness of the data returned by the API. This includes:
  + Verifying that user avatars and IDs are correctly associated.
  + Ensuring user email addresses follow the expected format.
  + Confirming that the updatedAt timestamp is correctly set after a user update.

**3. Tools Used and Why**

The following tools have been selected for this test suite, each serving a specific purpose in ensuring the quality and reliability of the Reqres API:

* **Java:** A robust, object-oriented programming language that provides a solid foundation for building scalable and maintainable test automation frameworks.
* **RestAssured:** A Java-based library that simplifies the testing of RESTful APIs. It offers a Behavior-Driven Development (BDD) style syntax (given/when/then), making the tests easy to read and write.
* **JUnit 5:** A popular testing framework for Java that provides annotations for identifying and configuring test methods. It also includes features for assertions, which are crucial for verifying expected outcomes.
* **Maven:** A build automation tool used for managing project dependencies, such as RestAssured and JUnit, and for building and running the project.

This combination of tools provides a powerful and flexible environment for API test automation, allowing for the creation of clear, concise, and maintainable tests.

**4. How to Run the Tests**

To execute the tests in this suite, you will need to have a Java Development Kit (JDK) and a build tool such as Maven or Gradle installed on your system.

1. **Clone the Repository:** Obtain the project code by cloning the repository from its source.
2. **Build the Project:** Open a terminal or command prompt, navigate to the project's root directory, and run the following command:
   * **For Maven:** mvn clean install
3. **Run the Tests:** The tests can be executed using the following commands:
   * **For Maven:** mvn test

The test results will be displayed in the console, and a detailed report will be generated in the target/surefire-reports (for Maven) or build/reports/tests/test (for Gradle) directory.

**5. Assumptions and Limitations**

**Assumptions:**

* The Reqres API is available and accessible at the specified URL (https://reqres.in).
* The API endpoints and their expected responses remain consistent with the documentation.
* The test environment has a stable internet connection.

**Limitations:**

* **Limited Negative Testing:** The current test suite primarily focuses on happy path scenarios, with only one negative test case for unsuccessful registration. To improve coverage, more negative tests should be added to validate the API's error handling capabilities for other endpoints.
* **No Performance Testing:** This test plan does not include performance testing. To ensure the API can handle a high volume of requests, separate performance tests should be designed and executed using tools like JMeter or Gatling.
* **No Security Testing:** The suite does not cover security testing. It is recommended to perform security testing to identify and address potential vulnerabilities, such as SQL injection or cross-site scripting (XSS).
* **Static Test Data:** The tests currently use hardcoded data. To make the tests more robust and reusable, the test data should be externalized and managed separately.