API Test Automation Technical Document

Company Name: HMH

Author: Suganya

Date:09-02-2024

For this Technical assessment, I have developed a project to validate the main functionality of the **FAV QUOTE** and **LIST QUOTES** endpoints provided by the **Favqs API**. The tests are implemented using **Java and TestNG**, leveraging the **Rest Assured** library for REST API testing. Each test case is thoroughly documented to ensure easy understanding.

I have created a **maven project** and added the essential dependencies in the pom.xml for this project.

**Pom.xml**

The dependencies section in the Maven POM file specifies the external libraries required by the project. Here is why each dependency is included.

1. Used **RestAssured** library, which is a Java DSL for testing RESTful APIs which allows testing of HTTP requests and responses.
2. Used **TestNG** for Parameterizing tests and for structured execution of tests using annotations and assertions.
3. Used **Hamcrest** as it provides a set of matchers that is used to create assertions in the tests.
4. Below is the Navigation path of my project.

HMH (Maven Project)

|-> src/test/java

|-> hmh\_Favqs

-> BaseClass.java

-> TC01\_QuoteOfTheDay.java

-> TC02\_FavQuote.java

-> TC03\_FavQuote\_InvalidData.java

-> TC04\_FavQuoteWithNoSessionId.java

-> TC05\_UnFavQuote.java

-> TC06\_UnFavQuoteWithNoSessionId.java

-> TC07\_ListAllQuotes.java

-> TC08\_GetQuoteContainingWords.java

-> TC09\_GetQuoteByFilter.java

|-> src/test/resources

-> Login Credentials. Json

|-> test-output

Pom.xml

Testng-hmh.xml

**Note:**

1. **QuoteID Retrieval:**

From the API documentation, I observed that the QuoteID serves as a unique identifier required for subsequent endpoint interactions. To ensure uniqueness, the QuoteID is retrieved from the Quoteoftheday Endpoint.

1. **Colon: Removal:**

During testing of the Fav and Unfav endpoints, it was noted that inclusion of a colon resulted in a 404 error. Consequently, the testing approach was adjusted by removing the colon from the endpoints as follows:

PUT /api/quotes/:quote\_id/fav

PUT /api/quotes/:quote\_id/unfav

1. **Marking TestCase03 as failure:**

Testing the Fav Quote functionality involved the provision of an invalid QuoteID. According to the documentation, this action should yield an error code 40 accompanied by the corresponding error message. However, the response received was a 404 error code, indicating that the resource was not found. As a result, the testcase was marked as a failure.

**Base Class Overview:**

* The **base URI is initialized, and global variables are set**. **Content-Type and API key** are configured as **basic headers** for all test cases.
* Executes a POST request to create a **new user session** before test case execution, utilizing the **@BeforeSuite annotation**. The session creation request body, containing JSON of the username and password, is sourced from a file named **"LoginCredentials.json" stored under the "src/test/resources" folder.**
* Extracts the **Session\_ID** from the response and appends it in the **getAddHeaders()** method along with the basic headers.
* Utilizes the **@AfterSuite** annotation to **destroy the session ID** after all test cases finish executing.
* Implements **error handling using Try-Catch blocks** to ensure robustness in handling exceptions.

**Note**: **Baseclass is extended by all the test cases below, each annotated with @Test along with dependsOnMethod and Priority annotations.**

**TC01\_QuoteOfTheDay**

* The HTTP GET request is utilized to retrieve the Quote\_id from the Quote of the Day API endpoint.
* The **Quote ID is dynamically captured** from the response received.
* The system proceeds to validate the HTTP status code, ensuring it matches the expected value of **200 OK.**
* Additionally, it checks for the **absence of error codes** in the response.
* The **Quote ID is then logged in the TestNG report** for reference.
* To prioritize its execution, a **priority of 0** is assigned to this particular task.

**TC02\_FavQuote:**

* The **QuoteID obtained from TestCase01 is utilized in TestCase02** to mark the quote as a **favorite**.
* This action involves **appending the QuoteID to the FavQuoteURI endpoint.**
* The test procedure then proceeds to verify the success status code, ensuring it aligns with the expected value of **200 OK.**
* Additionally, it checks for the **absence of error codes** in the response.
* Furthermore, an assertion is made to confirm that the "**Favorite" field is set to true,** indicating that the quote has been successfully marked as a favorite.
* To ensure proper execution sequence, the annotation **@Test(dependsOnMethods = "hmh\_Favqs.TC01\_QuoteOfTheDay.quoteOfDay") is set. This informs TestNG that TestCase02 depends on TestCase01 and should execute after it**. If TestCase01 fails, TestCase02 will be skipped.

**TC03\_FavQuote\_InvalidData: Known Failure:**

* **An invalid QuoteId is initiated** and combined with valid Quote ID to have unique ID for each run.
* **This combined ID is then used in the FavQuote URI** to attempt marking a quote as a favorite for validation purposes.
* The objective of this test is **to simulate marking a quote as a favorite using an invalid Quote ID**, with expectations aligned with the API documentation.
* Per the documentation, **the expected error code and message are:**

"error\_code": 40,

"message": "Quote not found."

* However, the **received response deviates from expectations, presenting a different status code and error message:**

"status": 404,

message“”:”NotFound”

**TC04\_FavQuoteWithNoSessionId:**

* The test case **attempts to mark a quote as a favorite without including a session ID** in the request headers.
* It utilizes the dynamically retrieved Quote ID obtained from a previous test case.
* The expected outcome is a status code of 200 with the following error code and message:

"error\_code": 20,

"message": "No user session found."

* The annotation **@Test(dependsOnMethods = "hmh\_Favqs.TC01\_QuoteOfTheDay.quoteOfDay") is set to indicate that TestCase04 depends on TestCase01**. Consequently, it executes after TestCase01 and will be skipped if TestCase01 fails.

**TC05\_UnFavQuote:**

* It aims to mark the "**Favourite" field as false** by sending a request to the corresponding endpoint.
* Verification is performed on the status code **200** to ensure the successful execution of the request.
* Additionally, the **absence of any error code** in the response is confirmed.
* The annotation **@Test(dependsOnMethods = "hmh\_Favqs.TC02\_FavQuote.checkfavQuote") is set to indicate that TestCase05 depends on TestCase02.** Therefore, it executes after TestCase02, and if TestCase02 fails, TestCase05 will be skipped.
* An attempt is made to validate that the **"Favourite" field is set to false**, indicating a transition from true to false.

**TC06\_UnFavQuoteWithNoSessionId:**

* The test case attempts to **remove a quote from favorites without including a session ID** in the request headers.
* Verification is performed on the status code **200** to ensure the execution of the request.
* Additionally, the presence of **error codes in the response is confirmed**.

"error\_code": 20,

"message": "No user session found."

* This ensures that the system appropriately handles the scenario where a session ID is missing.
* The test aims to maintain consistency **and compliance with the expected behaviour** defined by the API specifications.

**TC07\_ListAllQuotes:**

* The test case sends a request to the API endpoint to **retrieve all available quotes**.
* Upon receiving the response, it verifies that the response contains **25 quotes,** indicating the expected quantity.
* The status code **200** is validated to ensure the successful execution of the request.
* Additionally, the **absence of any error codes** in the response is confirmed.
* This comprehensive validation ensures that the system properly retrieves and delivers the expected number of quotes without encountering any errors.
* By confirming the adherence to expected behaviour, the test contributes to maintaining the reliability and functionality of the API endpoint.
* **@Test(priority = 1)** is set and it is not dependent on any test case.

**TC08\_GetQuoteContainingWords:**

* The test case retrieves quotes containing **specific words** from the server.
* It verifies the response status code **200** to ensure the request was successful.
* Additionally, the **absence of error codes** in the response is confirmed, indicating a smooth operation.
* To enable testing with various data, the test case utilizes **TestNG's Data Provider feature**.
* **TestNG's data Provider supplies dynamic query parameters to the test method, allowing verification of multiple test data scenarios.**
* The annotation @Test(dataProvider = "queryParamProvider", priority = 2) specifies the use of the data provider and assigns priority 2 to the test method

@DataProvider(name = "queryParamProvider")

**public** Object[][] queryParamProvider() {

**return** **new** Object[][] {

{"funny"},

{"humor"},

};

**TC09\_GetQuoteByFilter:**

* The test case retrieves quotes from the API based on various **filters such as tag, author, or user.**
* It validates the response status code **200** to ensure the request was successful.
* Additionally, the **absence of error codes** in the response is confirmed, indicating a smooth operation.
* To cover a range of scenarios, the test case verifies **three different test data scenarios, each representing a unique filter example.**
* **TestNG's data Provider feature is utilized to supply dynamic query parameters** to the test method, enabling testing with different filter combinations.
* The annotation **@Test(dataProvider = "queryParamsProvider", priority = 3) is applied to the test method, indicating the use of the data provider and assigning priority 3**

@DataProvider(name = "queryParamsProvider")

**public** Object[][] queryParamsProvider() {

**return** **new** Object[][] {

{"funny", "tag"},

{"Nassim Taleb","author"},

{"gose","user"}

};