Twitter API Test Suite

In this test suite we will test and check if end points of the API work properly at creating a tweet, looking for that tweet, deleting the tweet, hide replies and search for user, blocking users, checking the list of the current blocked users and unblocking some of them. All tests will start by checking if the status code is correct to the test done, for example if we want to properly publish a tweet the code needed in the test should be 201. The second part of the tests are done to make sure that the body of the response is properly done and have the correct information from the Valid Request.

The tests are divided in two sections and will be performs in order. Section one tests are “Valid Requests” and section two test are “Invalid or Unauthorized tests”. This was made to test if the methods and endpoints of our API would work and respond accordingly to documentation when a valid request was made and when an invalid or unauthorized request was done.

First, we will go through our “Valid Requests” tests.

1. ***Creating a tweet***

Before the test was done a random word or name was generated to be published inside the tweet and was saved inside an environment variable in Postman and used in the body of the request.

To check if the tweet was published properly and the API response is properly done, we will check the status code test and also important part of the body response that include the tweet id and tweet content or text.

* Status code should be 201.
* Should include tweet id
* Should include tweet text

After the tests are done the tweet id and text are saved into environments variables to use the upcoming tests.

1. ***Checking tweet***

Using the variables created in our last test we will search for the tweet by its id. We will test if the id finds the proper response from this endpoint. First, we will check the status code and then check if the response body includes the tweet id and the tweet text.

* Status code should be 200
* Should include tweet id
* Should tweet text

1. ***Delete a Tweet***

Using the previous tweet id, we will delete it and confirm that it was deleted by the response body from our endpoint. First, we will check the status code and then we will check if the response includes ‘{“deleted”:”true”}’, confirming that the tweet was deleted.

* Status code should be 200
* Should confirm that tweet is deleted

1. ***Search User by Username***

In this simple test we will test that the endpoint can successfully search a user by its username. In this test, in our ‘*pre-request script*’, we will create a function that randomly selects a username from a list and it will be saved into an environment variable that can be used in our parameters to look for a user. To test this endpoint, we will first check the status code and then check if the response includes the user id, user and username.

* Status code should be 200
* Should include User ID
* Should include name
* Should include username

1. ***Blocks look up***

In this test we will look the users blocked by the owner of the API key. To test this endpoint, we will check the status code, check if the response includes the result\_count or the number of users blocked by the user and also the number of users blocked. In this case should be zero.

* Status code should be 200
* Body response should include ‘result\_count’
* ‘result\_count’ should be 0

Now, we will test some of the same endpoints and check if they respond properly to an invalid or unauthorized request from an user.

*\*For most of these tests we will have an Unauthorized version, those will work the same. This will be a simulation of the response that the API has when an user uses key that does not have permission to read or write. First, we will check for the status code that should be 401 and then we will check for the response body and look to find the “title” section that should say “Forbidden”.*

* *Status code should be 401*
* *Title should say ‘’Unauthorized”*

1. ***Create a Tweet***
   1. ***Unauthorized\****
   2. ***Invalid Input***

In this test we will check if this endpoint responds properly to the user trying to publish a tweet with the same text inside it. We will first check the Status code that should be 403 Forbidden and we will check the error details and title. Details should describe that the user is trying to publish a duplicated tweet. The tittle should say Forbidden.

* Status code should be 403
* Should include warning on duplicate content
* Title should say ‘Forbidden’

1. ***Check Tweet***
   1. ***Unauthorized\****
   2. ***Tweet Unavailable***

In this test we will check the response when we search for a tweet, and it doesn't exist and can’t be found. First, we will check the Status code that should be 200 and then we will check the title section of the response body and it should say “Not Found Error”

* Status code should be 200
* Title should indicate ‘Not Found Error’

1. ***Delete Tweet***
   1. ***Unauthorized\****
2. ***Search User by Username***
   1. ***Unauthorized\****
   2. ***Invalid Input***

This test will check what happens when we look for a user by its username, but we use an invalid character in the parameters. First, we will check status code, it should be 400, then we will check the title section that should say ‘Invalid Request’ and finally we will check for the error details and confirm that the message says the one of the parameters is invalid.

* Status code should be 400
* Title should say ‘Invalid Request’
* Details should declare parameters invalid
  1. ***User not Found***

Before the test is done, we created a function that takes a username from an array at random and saves it inside an environment variable so we can use it in the parameters. All of the usernames inside the array will cause the same response from the endpoint, which is “user not found” error. To test this, we will first check the status code that should be 200 and then we will look for the response message that indicate the user could not be found using the parameters given by the user. Then we will check the title section that should say “Not Found Error”.

* Status code should be 200
* Should include message ‘Could not fund user’
* Title should say ‘Not Found Error’

1. ***Block look up***

This test will check the response given by this endpoint when the user inputs a parameter that is not his own account id to check the blocked users list, or the number of users blocked. According to the documentation the parameter should use the id from the user that is also the owner of the API keys. To test this, we will first check the status code that should 400, then we will check for the title section inside the response that should say “Invalid Request” and the finally we will confirm through the error details that the parameter is invalid.

* Status code should be 400
* Title should say ‘Invalid Request’
* Should confirm one or more parameters are invalid

Here is the link to join the workspace and check the test suite.

<https://app.getpostman.com/join-team?invite_code=3c1e810daff1f7c995dd8752d4503140&target_code=b52a031d5ae8df2d83af7ad619b2d09b>

NBA.com Test Cases

This time we will test the NBA.com site and do a couple of simple tests. We will do 5 simple automation tests using Selenium. Before starting the first test the program will automatically accept cookies, so the element doesn’t interfere with the various tests.

* **Sign Up a new User**

The user will Sign up for a new free NBA.com account.

User will start at home page and will signup to a new account by clicking into the ‘Sign In’ tab where a dropdown menu will pop up and he will click into ‘Sign into NBA.com’. A window will pop up and he will click into the top of the window where a link will get him to Sign Up a new account. At the Sign-Up form will input email, password, first name, last name, his birth of day and country, then will accept and check the Term of Service box and create a new account. After that the site will ask for the user favorite team and the user will select his team and continue to the next page that is letter options and the user will choose to check boxes of daily mail and nba.com promotional material.

After that the test will check that the user arrived at the final windows where it will ask the user if they want to buy NBA League Pass.

* **Search for Historic Player Stats**

The user will look up the players with more than 30 rebounds in a game and wants to know the player with the most rebounds in a single game.

User will start at the home page and will click the Stats tab in the navigation bar and that will bring him into Stats Page where he will click into ‘Tools’ the navigation bar and will look for ‘Players Box Score Search’. That will bring him into the tool to search for the information he is looking for. The user will add a filter where he will look for player with more than 30 rebounds in a single game. After adding input to the filter he will click the “Run It” button and that will make the box score appear. Then the user will organize the table by the player with the most rebounds in a single game and he will click the rebounds columns to check that.

The test will check if the table is visible to confirm the test was successful.

* **Search for a Single Player**

The user is looking for his favorite player “Stephen Curry” and want to look for his profile

User will start at the home page where he will click the ‘Players’ Tab and then he will find himself in the players page where he will input “Curry” to look up a single player. After his input players with this last name will appear and he will click in the ‘Stephen Curry’ row. That will bring him into the player’s profile where the user will get all the information, news and statistics he wants.

The test will check if the title matches the name ‘Stephen’ to prove if the user is in the right page.

* **Search for a Box Score from a Game in the Past**

The user is interested in knowing the box score of a game in the past of his favorite team.

User will start at the home page where he will click into the ‘Schedule’ Tab. There he will input and modify the filters to look for games from a specific team such as the ‘Miami Heat’ and then look up games from February. Then he will scroll down and find a specific game where he will click into the ‘Box Score’ link and that will bring him into the game box score.

The test will check if the box score table is present to prove if the user is in the right page.

* **Search for a News of MVP Award**

The user wants to know the latest news on the NBA awards for this season and wants to know who won season MVP.

User will start at the home page and will click into the ‘News’ Tab. There he will click the Awards Tab in the navigation bar inside the News Page and that will bring him into the Awards Page where can see all the news about every award this season. He will scroll down to find the article announcing the seasons’ MVP. He will click on the article to read the announcement.

The test will check the title of the article and will check if it contains the word ‘Jokic’ to prove that the user is inside the article page.