**Ali Ahmed**

**Manual Tesing :**

Using Gherkin language:

**I – The Checkers Game**

Feature: Checkers Game

Scenario: Navigate to the Checkers game site

**Given** I am on the Checkers game site

**When** I navigate to "https://www.gamesforthebrain.com/game/checkers/"

**Then** the site should be up

Scenario: Make five legal moves as orange

**Given** I am on the game page

**When** I make five legal moves as orange

**Then** the game should allow me to continue after each move

Scenario: Include taking a blue piece

**Given** I am making legal moves as orange

**When** I take a blue piece during my moves

**Then** the move should be valid

Scenario: Use "Make a move" as confirmation

**Given** I am making a move

**When** I use "Make a move" as confirmation

**Then** the next step should be allowed

Scenario: Restart the game after five moves

**Given** I have completed five moves

**When** I restart the game

**Then** the game should be reset

Scenario: Confirm successful game restart

**Given** I have restarted the game

**When** I confirm the restart

**Then** the game should be ready for a new session

**Automation Testing :**

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

public class CheckersGameTest {

public static void main(String[] args) {

// Set the path to the ChromeDriver executable

System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");

// Create a new instance of the ChromeDriver

WebDriver driver = new ChromeDriver();

try {

// Scenario 1: Navigate to the Checkers game site

driver.get("https://www.gamesforthebrain.com/game/checkers/");

// Scenario 2: Confirm that the site is up

if (driver.getTitle().contains("Checkers"))

System.out.println("Scenario 2: Site is up");

else

System.out.println("Scenario 2: Site is not up");

// Scenario 3: Make five legal moves as orange

for (int i = 0; i < 5; i++) {

makeLegalMove(driver);

}

// Scenario 4a: Include taking a blue piece

takeBluePiece(driver);

// Scenario 4b: Use "Make a move" as confirmation

useMakeMoveConfirmation(driver);

// Scenario 4c: Restart the game after five moves

restartGame(driver);

// Scenario 4d: Confirm that the restarting had been successful

confirmRestart(driver);

} finally {

// Close the browser window

driver.quit();

}

}

private static void makeLegalMove(WebDriver driver) {

// Your code to make a legal move as orange

}

private static void takeBluePiece(WebDriver driver) {

// Your code to take a blue piece during the moves

}

private static void useMakeMoveConfirmation(WebDriver driver) {

// Your code to use "Make a move" as confirmation

}

private static void restartGame(WebDriver driver) {

// Your code to restart the game after five moves

}

private static void confirmRestart(WebDriver driver) {

// Your code to confirm that the restarting had been successful

}

}

**II – The Card Game**

Explain:

1. **Create a new request:**

* Open Postman and create a new request by clicking on the "New" button.

1. **Enter the API URL:**

* Enter the URL for the API:

https://deckofcardsapi.com/. This is the base URL for the deck of cards API.

1. **Confirm the site is up:**

* To confirm that the site is up, you can simply send a GET request to the base URL. Click on the "Send" button to make the GET request.

1. **Get a new deck:**

* To get a new deck, you can make a GET request to the /api/deck/new/ endpoint. After sending the request, you'll receive a JSON response containing a deck ID.

1. **Shuffle the deck:**

* To shuffle the deck, make a request to the /api/deck/{deck\_id}/shuffle/ endpoint, replacing {deck\_id} with the actual deck ID you obtained in the previous step.

1. **Deal three cards to each of two players:**

* Make a request to the /api/deck/{deck\_id}/draw/ endpoint twice (for each player) with the query parameter count=3 to draw three cards for each player.

1. **Check whether either has blackjack:**

* Examine the cards drawn for each player to check if they have blackjack. Blackjack typically involves an Ace and a 10-point card (10, Jack, Queen, or King).

1. **Write out which one has blackjack:**

* Based on the cards drawn, determine if either player has blackjack. If one does, you can print a message indicating which player has blackjack.

1. **Test and document:**

* Run the entire collection to make sure each step works as expected. Document the request and response details for reference.

Here's a summary of the Postman requests you will make:

**Request 1:**

* Method: GET
* URL: https://deckofcardsapi.com/
* Purpose: Confirm the site is up.

**Request 2:**

* Method: GET
* URL: https://deckofcardsapi.com/api/deck/new/
* Purpose: Get a new deck.

**Request 3:**

* Method: GET
* URL:

https://deckofcardsapi.com/api/deck/{deck\_id}/shuffle/ (replace {deck\_id} with the actual deck ID)

* Purpose: Shuffle the deck.

**Request 4 and 5:**

* Method: GET
* URL:

https://deckofcardsapi.com/api/deck/{deck\_id}/draw/?count=3 (twice, once for each player)

* Purpose: Draw three cards for each player.

After completing these steps, I can analyze the drawn cards to check for blackjack and print out the results accordingly.