Testowanie backendu narzędziami Postman, SoapUI.

SoapUI - Zadania

Zadanie S_01 Pobierz listę ras

Utwórz TestCase o nazwie “Exercise S_01 – Get all races”, a w nim przygotuj request, który będzie pobierał listę wszystkich ras dostępnych pod zasobami “dictionaries”

Asercje:

A. Dodaj asercję sprawdzającą czy http code = 200
B. dodaj asercję sprawdzającą czy ilość elementów na liście w JSONie zwrotnym **jest równa 10**
C. dodaj asercję sprawdzającą czy na 5 pozycji nazwa kota to “Kot egzotyczny”

Hint: Stronka wspierająca w czytaniu JsonPath: https://jsonpath.com/

Zadanie S_02 Zarejestruj nowego użytkownika

Utwórz nowy TestCase o nazwie “Exercise S_02 – Register new user”, który zawierać będzie request wywołujący utworzenie nowego użytkownika.

Hint: Przykład zawartości body requestu tworzącego użytkownika odnajdziesz w dokumentacji Swagger na stronie https://norad-beta.duckdns.org/norad/swagger-ui.html

Krok 1. Zmień TestCase w taki sposób, ażeby można było wywołać request utworzenia **wielokrotnie.**

Czyli e-mail nie może się powtarzać, za każdym wywołaniem powinien generować się nowy unikalny ze stałym prefixem - przyjmijmy Twoja pierwsza litera imienia i nazwisko. Na przykład: ***mszpiler***

Hint: Przeszukaj projekt ***postman-soapui-training*** w katalogu “**groovy”** znajdują się przykłady, które mogą Tobie pomóc rozwiązać zadanie.

Krok 2. Dodaj asercję sprawdzającą, że w odpowiedzi będzie w strukturze json w polu **email** istniała wartość oczekiwana wygenerowana przed wywołaniem requesta tworzącego nowego użytkownika.

Krok 3. Dodaj asercję sprawdzającą za pomocą skryptu Groovie czy rekord w bazie danych istnieje

- Połącz się z bazą danych

- Pobierz parametr email z kroku pierwszego, w którym email jest generowany i wykorzystany podczas rejestracji konta
- Wykonaj zapytanie SQL
- Zastosuj polecenie assert

Krok 4. Zapisz UUID użytkownika w propertiesach **w kontekście Test Suite** pod zmienną **userUUID**.

Zadanie S_03 Dodaj nowe zwierzę do nowego użytkownika

Utwórz nowy TestCase "Exercise S_03 – Add new animal for new user", który zawierać będzie następujące kroki:

Krok 1. Rejestracja nowego użytkownika.

Krok 2. Utworzenie losowej nazwy zwierzęcia ze stałym prefixem na przykład pierwsza litera imienia i nazwisko np. **mszpiler .** Zapisz wygenerowany string pod zmienną **animalName** w kontekście **TestCase'a**

Prefix pozwoli lepiej identyfikować Twoje dane w bazie danych, która jest wspólna dla wszystkich uczestników szkolenia.

Krok 3. Pobierz wybrany losowo element z listy ras i zastosuj go przy tworzeniu zwierzęcia. Zapisz go w kontekście Test Case'a w propertiesach pod zmienną **raceId**

Krok 4. Pobierz Access Token i zapisz w propertiesach pod zmienną **ACCESS_TOKEN.**

Krok 5. Dodaj nowe zwierzę dla użytkownika z kroku 1, dodane zwierzę powinno zawierać **3 zdjęcia** oraz rasę ustaloną w kroku 3. W headerze requesta wykorzystaj access token pobrany z serwera autoryzacyjnego w kroku 4.

Krok 6. Zapisz uuid zwierzęcia w propertiesach w kontekście **Test Suite** pod zmienną **animalUUID**.

Krok 7. Zapisz uuid **pierwszego zdjęcia** w propertiesach w kontekście **całego Testu Suite** pod zmienną **imageUUID**

Krok 8. Zapisz **email** użytkownika w propertiesach w kontekście całego **Testu Suite** pod zmienną **email**

Hint: Zobacz jaka relacja jest między encjami user, animal i image. Zobacz w dokumentacji zawartość DTO dla utworzenia nowego zwierzęcia.

Asercje

A. Sprawdź czy w bazie zwierzę posiada oczekiwaną wartość określającą rasę.
B. Sprawdź czy w bazie zwierzę powiązane jest z użytkownikiem utworzonym w pierwszym kroku.
C. Sprawdź czy w odpowiedzi lista zdjęć jest oczekiwana i wynosi 3
D. Sprawdź czy każde ze zdjęć posiada niepuste pole uuid

Zadanie S_04 - Wykonaj upload zdjęcia - załącznik wskazany ręcznie

Przygotuj Test Case, który będzie wykonywał test uploadu wybranego contentu zdjęcia na serwer. Wykorzystaj encję zdjęcia utworzonego w zadaniu S_03.

Krok 1. Pobierz Access Token i zapisz w propertiesach pod zmienną ACCESS_TOKEN.

Krok 2. Dodaj step “Upload file”, który wywołuje endpoint odpowiedzialny za upload plików na serwer.

Krok 3. Dodaj załącznik ręcznie i wykonaj request, który wykona upload obrazka na serwer.

Asercje

A. Sprawdź czy odpowiedź http = 200
B. Utwórz request pobrania obrazka i sprawdź czy uzyskany obrazek posiada oczekiwany rozmiar - czyli zgodny z tym rozmiarem pliku jaki został użyty podczas uploadu zdjęcia.

Zadanie S_05 - Wykonaj upload zdjęcia - załącznik wskazany za pomocą skryptu Groovy

Przygotuj Test Case, który będzie wykonywał test uploadu wybranego contentu zdjęcia na serwer. Wykorzystaj encję zdjęcia utworzonego w zadaniu S_03.

Krok 1. Pobierz Access Token i zapisz w propertiesach pod zmienną ACCESS_TOKEN.

Krok 2. Dodaj step “Upload file”, który wywołuje endpoint odpowiedzialny za upload plików na serwer.

Krok 3. Określ dwa custom properties na poziomie projektu:

- ***imageFilePath*** - śwcieżka do pliku ze zdjęciem
- ***imageFileName*** **-** nazwa pliku ze zdjęciem

Krok 4. Dodaj załącznik za pomocą skryptu Groovy i wykonaj request, który wykona upload obrazka na serwer.

Asercje

A. Sprawdź czy odpowiedź http = 200
B. Utwórz request pobrania obrazka i sprawdź czy uzyskany obrazek posiada oczekiwany rozmiar - czyli zgodny z tym rozmiarem pliku jaki został użyty podczas uploadu zdjęcia.

Zadanie S_06 - Dodaj zwierzęta pobierając dane z pliku CSV

Utwórz nowy TestCase, który utworzy wiele zwierząt do konkretnego użytkownika.

Wykorzystaj metodę gotoStep oraz korzystanie z pętli w Groovy.

Lista zwierząt znajduje się w pliku CSV. Dane na temat zwierzęcia w pliku, to nazwa zwierzęcia oraz rasa.

Krok 1. Utwórz step rejestrujący nowego użytkownika.

Krok 2. Utwórz step pobierający Access Token.

Krok 3. Utwórz step tworzący jedno zwierzę - ciało requesta ma sparametryzowane wartości **animalName**, **raceId.** Parametry **animalName** i **raceId** będą określane w skrypcie groovy, który w pętli będzie czytać zawartość pliku CSV.

Krok 4. Utwórz step ze skryptem Groovy, który odczytuje plik CSV i w pętli przechodzi po wierszach pliku. Dla każdego wiersza - za wyjątkiem pierwszego który stanowi nagłówek danych – skrypt tworzy zwierzę powiązane z użytkownikiem utworzonym w kroku pierwszym.

Asercje

A. W pętli odczytaj zawartość pliku CSV i sprawdź czy istnieją dla każdego wiersza rekordy **w bazie danych**. Zastosuj połączenie do bazy danych i przygotuj zapytanie SQL, które wiąże użytkownika ze zwierzęciem, parametrami zapytania będą
    a. nazwa zwierzęcia
    b. Identyfikator rasy
    c. UUID zarejestrowanego użytkownika z kroku pierwszego

Zadanie S_07 - Wiele kroków

Utwórz nowy TestCase “Exercise S_07 – Many steps”, który wykona następujący scenariusz:

Krok 1. Zarejestruj nowego użytkownika.

Krok 2. Utwórz step pobierający Access Token.

Krok 3. Utwórz step tworzący jedno zwierzę - ciało requesta ma sparametryzowane wartości **animalName**, **raceId.** Parametry **animalName** i **raceId** będą określane w skrypcie groovy, który w pętli będzie czytać zawartość pliku CSV.

Krok 4. Utwórz step ze skryptem Groovy, który odczytuje plik CSV i w pętli przechodzi po wierszach pliku. Dla każdego wiersza - za wyjątkiem pierwszego, który stanowi nagłówek danych – skrypt tworzy zwierzę powiązane z użytkownikiem utworzonym w kroku pierwszym.

Dodatkowo w pętli, po każdym wywołaniu requesta z utworzeniem zwierzęcia odczytaj w responsie uuid utworzonego zwierzęcia i dodaj uuid do mapy, która przechowywana będzie w kontekście TestCase’a. Kluczem mapy będzie uuid zwierzęcia, a wartością **raceId** tego zwierzęcia.

Krok 5. Wejdź w szczegóły każdego zwierzęcia i sprawdź czy ma oczekiwaną rasę - nie korzystaj ze sprawdzenia bazy danych lecz wykorzystaj restowe API.

Hint: Przygotuj step ***REST Request*** wyciągający szczegóły jednego zwierzęcia. Przygotuj step ***Groovy Script*****,** który w pętli wywoła step ***REST Request***. Pętla będzie dotyczyć listy uuid zwierząt utworzonych w kroku 3.