Małgorzata Janicka, Katarzyna Kuchcińska, Małgorzata Miąsik, Aneta Zaniewicz

Wprowadzenie do testowania usług REST API z użyciem Postman i Newman

Wstęp

REST API to usługi sieciowe wykorzystujące architekturę REST (ang. *Representational State Transfer*), stanowiącą dominującą metodę komunikacji pomiędzy aplikacjami rozproszonymi. Popularność tego typu rozwiązań powoduje, że coraz istotniejsze staje się zapewnienie efektywnych testów usług REST API.

Celem artykułu jest omówienie zasad projektowania testów usług REST API z użyciem narzędzia Postman API Development Environment oraz ich automatycznego wykonywania z wykorzystaniem oprogramowania Newman.

Artykuł powstał na podstawie pracy dyplomowej opracowanej w ramach studiów podyplomowych w Wyższej Szkole Bankowej w Gdańsku na kierunku Tester oprogramowania.

Celami przedstawionego procesu automatycznego testowania oprogramowania są między innymi:

- wykonywanie testów bez nadzoru,
- uzyskanie powtarzalności testów,
- znajdowanie błędów regresji,
- pomiar jakości oprogramowania,
- zwiększenie pokrycia testami,
- zmniejszenie kosztów testowania,
- szybsza realizacja procesów,
- testy na wielu środowiskach/platformach.

Osiągnięcie tych celów jest możliwe pod warunkiem zastosowania narzędzia pozwalającego na wydajne i efektywne opracowywanie przypadków testowych, a następnie ich wykonywanie bez bezpośredniego udziału testera.

REST API - co to jest?

REST jest zbiorem następujących reguł komponowania architektury i budowania rozproszonych aplikacji sieciowych:

- *Uniform Interface* - jednolity interfejs oparty na zasobach,
- *Stateless* - bezstanowość, informacje o stanie konieczne do obsługi znajdują się w żądaniu klienta,
- *Cacheable* - możliwość cache'owania danych przez serwer,
- *Client-Server* - jednoznaczny podział na klienta i serwer,
- *Layered System* - warstwowość, oznacza separację poszczególnych warstw po stronie serwera,
- *Code on Demand* - kod przesyłany na żądanie (element opcjonalny).

RESTful API to zestaw usług sieciowych wykorzystujących do komunikacji protokół HTTP i implementujących wszystkie zasady wzorca REST. Jeśli dane REST API łamie którąś z wymienionych zasad, to nie może być określane terminem RESTful.

Postman

Postman to zintegrowane środowisko wyposażone w GUI, służące do definiowania i wywoływania usług REST API oraz tworzenia testów tych usług z użyciem języka JavaScript. Pisanie testów wspierane jest gotowymi przykładami w postaci szablonów (ang. *snippetów*).

Postman zawiera środowisko uruchomieniowe oparte o Node.js, które umożliwia dodawanie dynamicznie tworzonych wywołań do żądań i kolekcji. Pozwala to na pisanie przypadków testowych i budowanie zapytań zawierających dynamiczne parametry czy przekazywanie danych pomiędzy poszczególnymi żądaniami.

Postman zapewnia również pełne wsparcie procesu automatyzacji, łącznie z dokumentacją i integracją z zewnętrznymi repozytoriami np. GitHub, GitLab czy Dropbox.

Produkt dostępny jest pod adresem https://www.getpostman.com/.

Newman

Newman to narzędzie rozwijane w ramach projektu Postman, dostępne z wiersza poleceń. Umożliwia ono uruchamianie kolekcji testów zdefiniowanych w Postman, zapewniając jednocześnie łatwą integrację z narzędziami do automatyzacji procesu budowania oprogramowania (ciągła integracja).

Produkt dostępny jest pod adresem https://github.com/postmanlabs/newman.

Testowane usługi REST API

Procesowi automatyzacji testów poddałyśmy usługi udostępnione w ramach platformy Taiga (https://taiga.io/). Taiga to narzędzie do zarządzania projektami prowadzonymi w oparciu o metodyki zwinne.

Podejście do testów zorganizowałyśmy pod kątem weryfikacji poprawności działania usług w ramach procesów biznesowych realizujących funkcjonalności platformy, a definiowanie scenariuszy i przypadków zrealizowałyśmy w oparciu o dokumentację usług dostępną na stronie Taiga REST API.

Przygotowanie scenariuszy i przypadków testowych w Postman

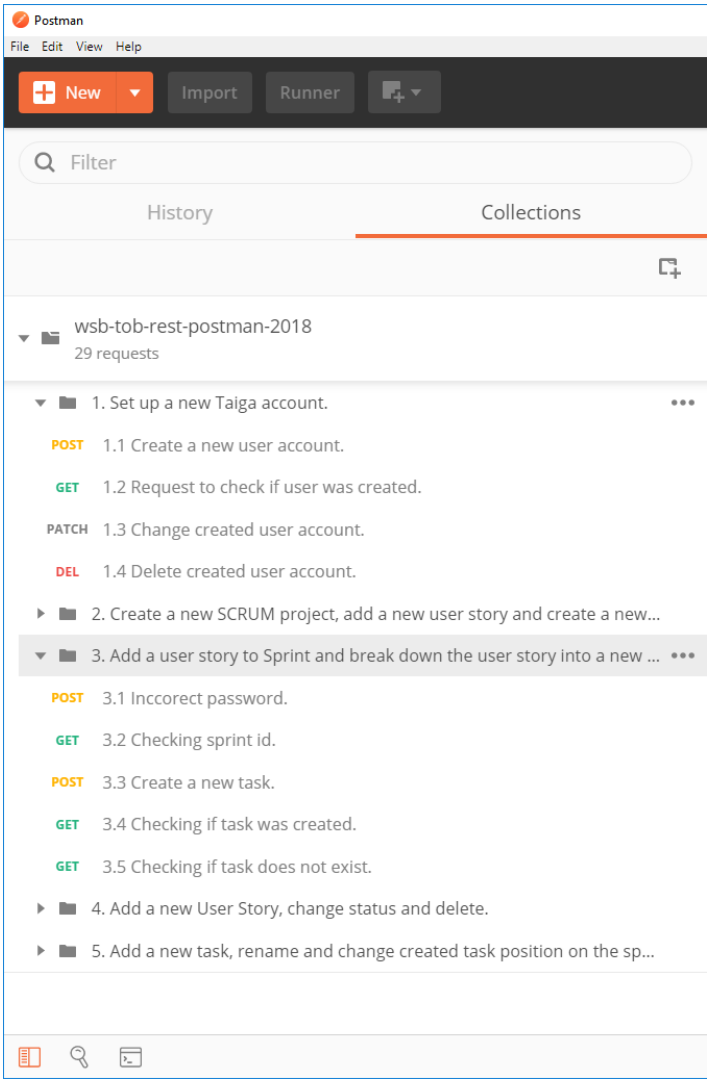
Opracowanie scenariuszy i przypadków testowych w Postman zaczęłyśmy od zdefiniowania kolekcji oraz poszczególnych przypadków testowych w ramach kolekcji.

Postman pozwala na grupowania wielu przypadków testowych w kolekcje (ang. *Collection*), które odzwierciedlają scenariusze testowe, co ułatwia zarządzanie i utrzymywanie przypadków testowych dla projektów.

Definiowanie scenariuszy testowych

Definiowanie kolekcji wykonywane jest z poziomu menu głównego Postman, gdzie określa się nazwę kolekcji oraz jej opis i globalne wartości dostępne następnie dla wszystkich przypadków testowych w ramach danej kolekcji.

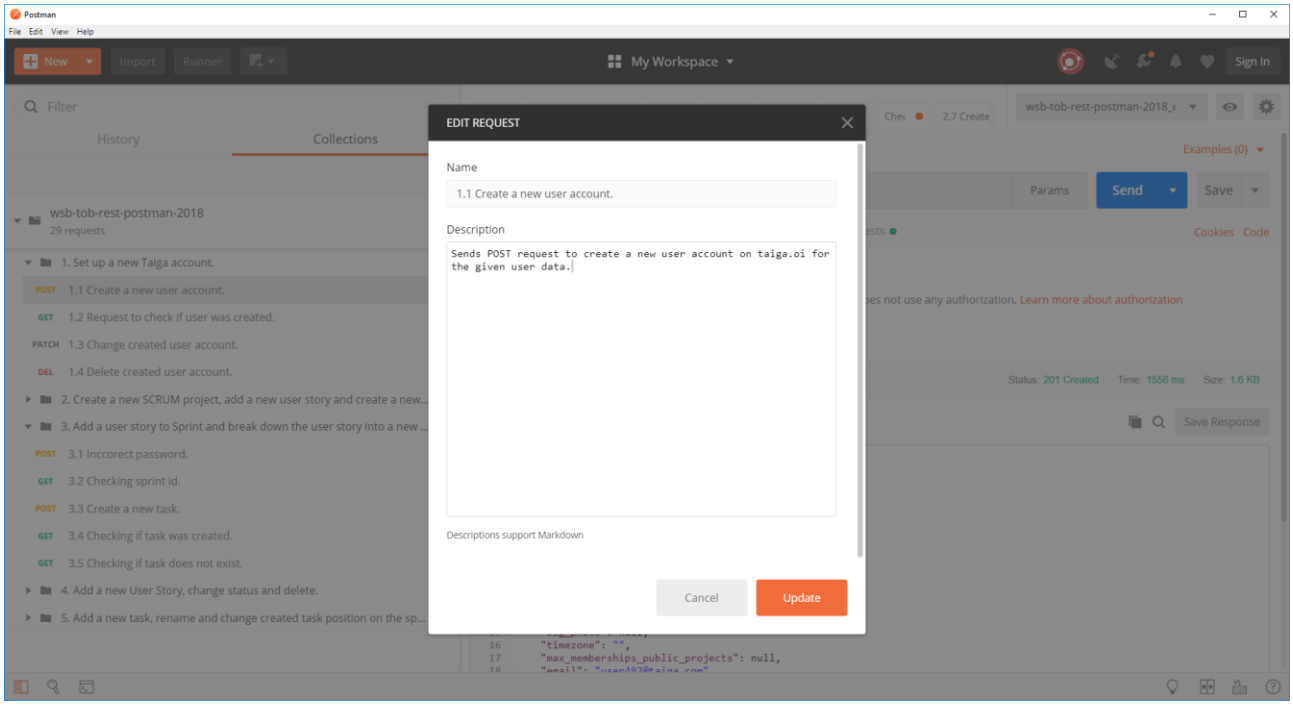
Kolekcje testów w naszej pracy stanowią dokładne odzwierciedlenie scenariuszy biznesowych, zgodnie z poniższym zrzutem ekranu.



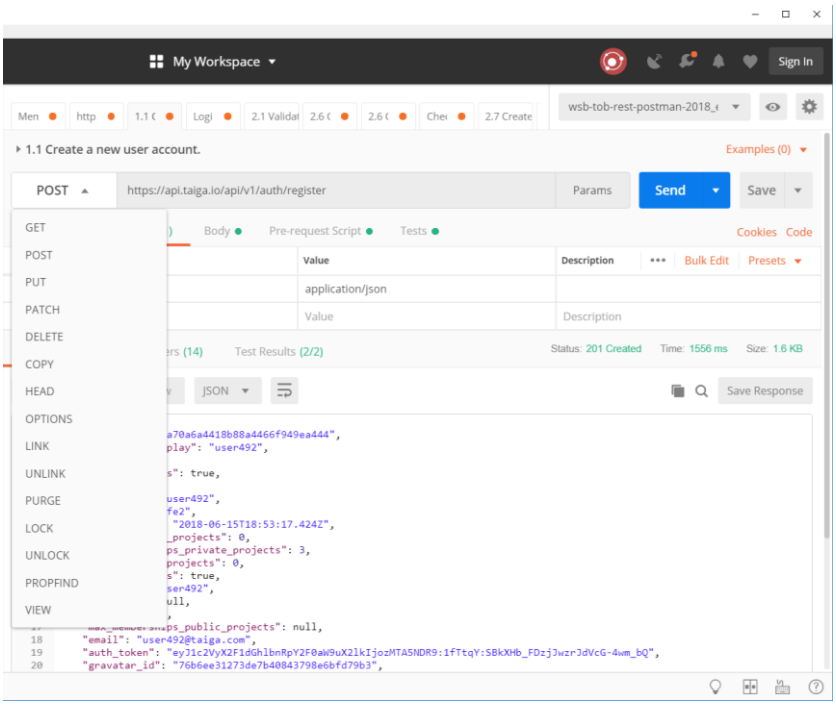
Warto również nadmienić, że na podstawie opisu kolekcji można automatycznie wygenerować dokumentację testów, co znacznie upraszcza proces zarządzania dokumentacją.

Definiowanie przypadków testowych

Po zdefiniowaniu kolekcji należy określić przypadki testowe, co realizowane jest w menu *New Request* z poziomu wybranej kolekcji:



Po zdefiniowaniu nazwy i opisu testu należy skonfigurować wywołanie usługi REST API poprzez wskazanie wykonywanej metody, adresu usługi oraz wymaganych parametrów.



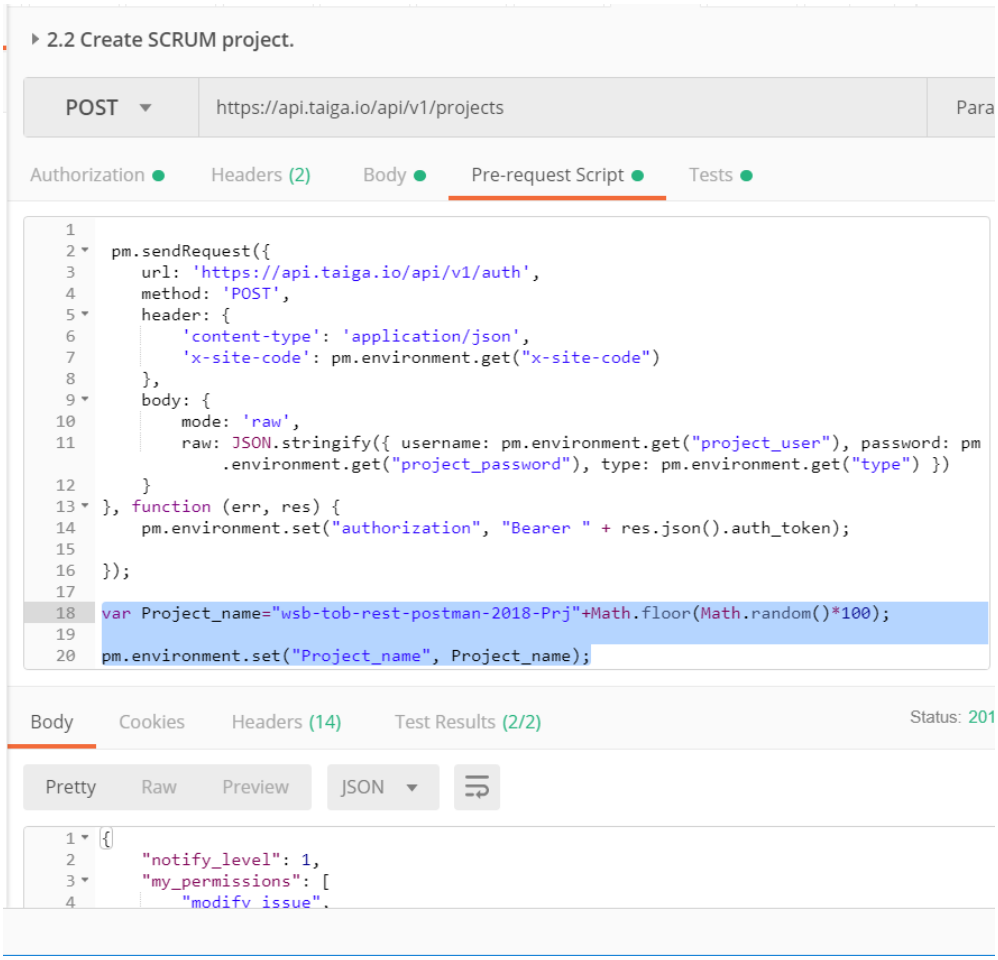
Przygotowanie *pre-request script*

Pre-request script to *snippet* wykonywany przed uruchomieniem przypadku testowego, który może wykonywać różne zadania, na przykład ustawiać wartości zmiennych środowiskowych, wartości wektora testowego (zbioru danych, które są używane przez przypadek testowy) oraz wykonywać żądania HTTP.

Pre-request script w ramach naszej pracy jest wykorzystywany do:

- generowania unikalnych identyfikatorów na potrzeby parametrów wejściowych usługi,
- generowania tokenu wymaganego do uwierzytelnienia wykonania usługi.

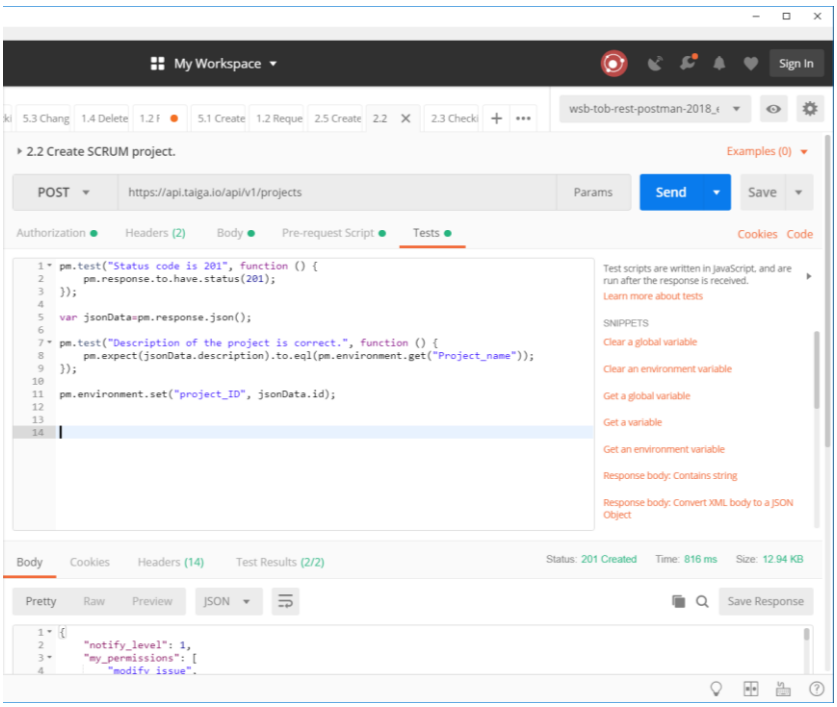
Poniższy przykład prezentuje kod generujący losowy identyfikator projektu. Do stałej nazwy dołączana jest liczba pseudolosowa, zapewniająca unikalność identyfikatora w ramach danego scenariusza testowego.



Przygotowanie asercji

Asercje służą do weryfikacji zgodności rzeczywistego działania testowanego oprogramowania z oczekiwanych wynikiem. Definiowane są w zakładce *Tests* przypadku testowego, pisane są w języku JavaScript z użyciem gotowego PM API Postman, dostarczającego gotowe funkcje testowe.

Dla jednego przypadku testowego możliwe jest wywołanie wielu asercji weryfikujących różne wartości zwracane w ramach wykonanej usługi. Najwygodniejszą metodą definiowania asercji jest skorzystanie z przykładowych *snippetów* dostępnych w menu Postman.



Na potrzeby naszego projektu zastosowane zostały m.in. następujące asercje:

- Weryfikacja statusów odpowiedzi HTTP:

```
pm.test("Status code is 200.", function ()
{
    pm.response.to.have.status(200);
    /*wynik testu: Weryfikacja statusu odpowiedzi http status =200.*/
});
```

```
pm.test("Status code is 201.", function ()
{
    pm.response.to.have.status(201);
    /*wynik testu: Weryfikacja statusu odpowiedzi http status =201.*/
});
```

```
pm.test("Status code is not found (404).", function ()
{
    pm.response.to.have.status(404);
    /*wynik testu: Weryfikacja statusu odpowiedzi http status =404.*/
});
```

- Weryfikacja wartości zwracanej w obiekcie JSON na zgodność ze zmienną środowiskową:

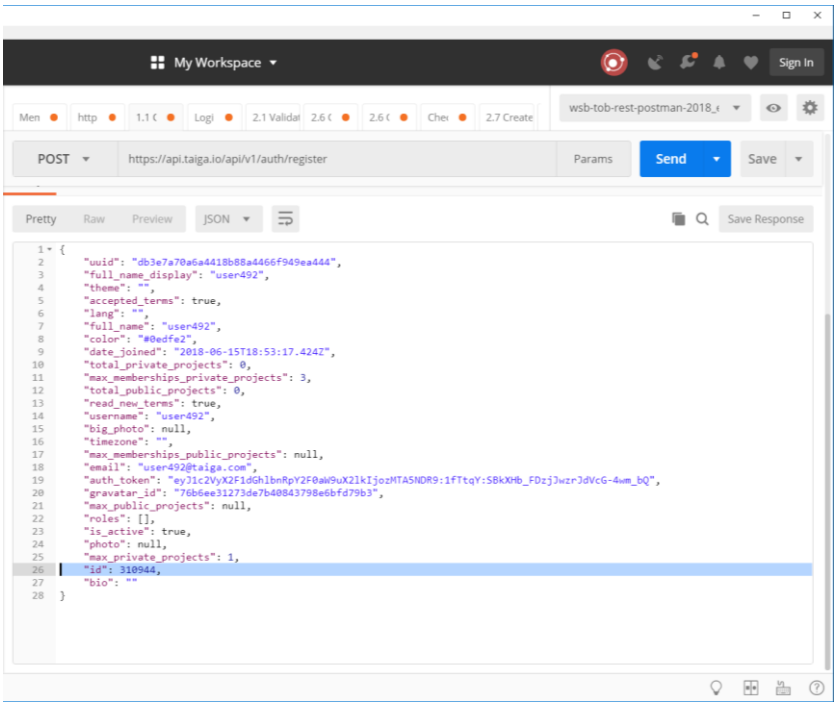
```
var jsonData = pm.response.json();
/*przypisanie do zmiennej zwracanej przez zapytanie wartości w formacie
json.*/
pm.test("Project name is correct.", function ()
{
      pm.expect(jsonData.description).to.eql(pm.environment.get("Project
_name"));
    /*wynik testu: Weryfikacja zgodności wartości description wyniku
żądania ze zmienną środowiskową Project_name.*/
});
```

Wykorzystanie zmiennych środowiskowych

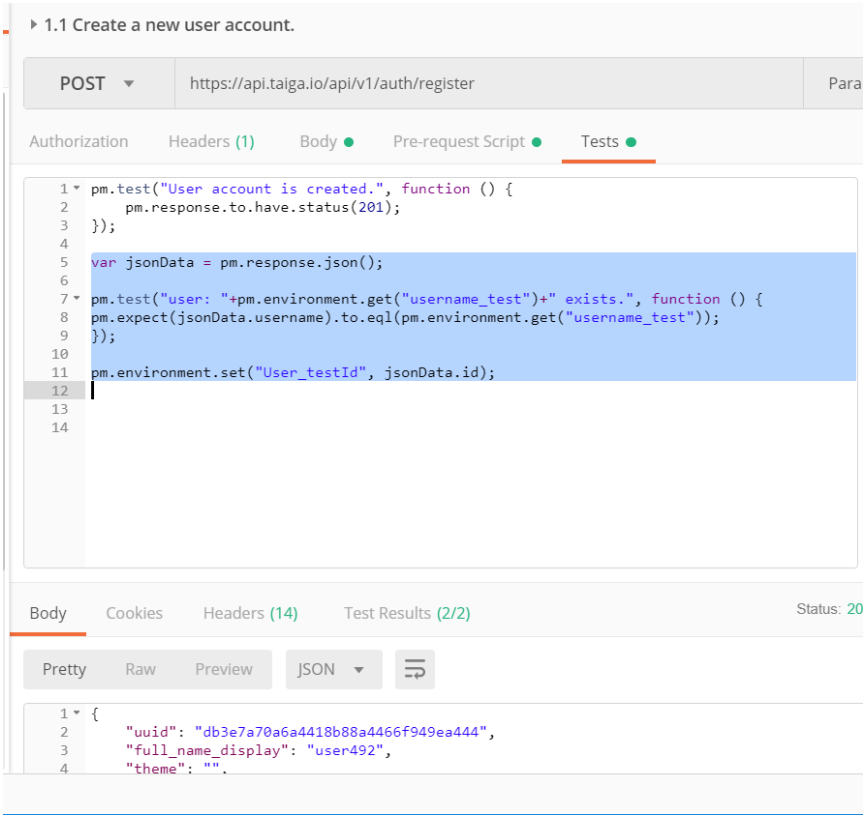
Jak już wspomniałyśmy, Postman zapewnia mechanizm własnego zarządzania zmiennymi środowiskowymi, które służą między innymi do przekazywania pomiędzy przypadkami testowymi wartości zwracanych przez usługi uruchamiane w ramach scenariuszy testowych, zgodnie z poniższym przykładem.

W ramach scenariusza realizowane są kolejno następujące przypadki testowe:

- Założenie konta użytkownika – metoda POST

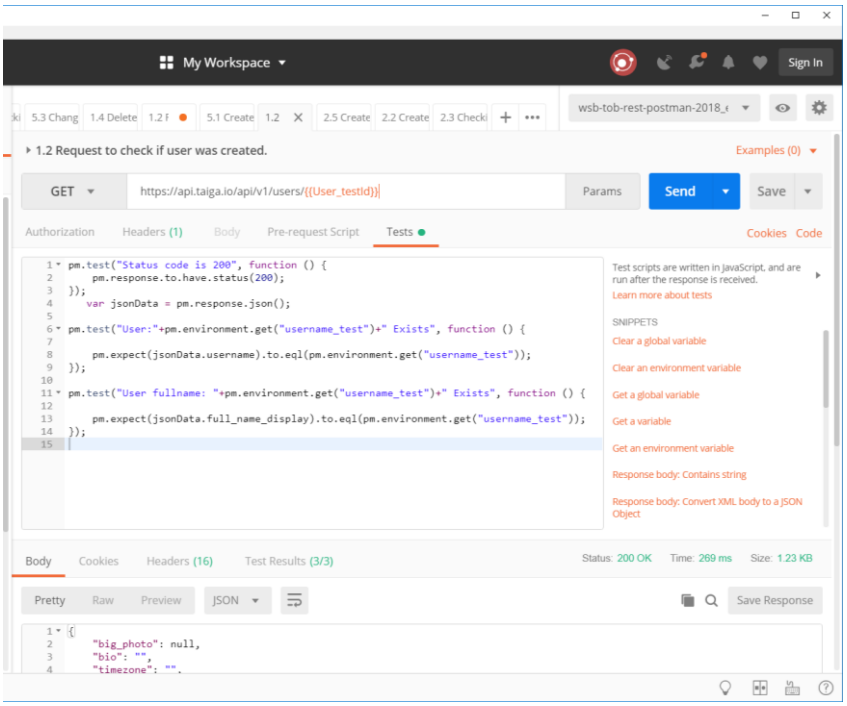


Ponieważ każde założone konto otrzymuje unikalny identyfikator, zwracany jako jedna z wartości obiektu JSON, to w ramach asercji wykonywanej po wywołaniu usługi założenia klienta wartość identyfikatora jest zapisywana do zmiennej środowiskowej.



- Weryfikacja utworzenia użytkownika – metoda GET

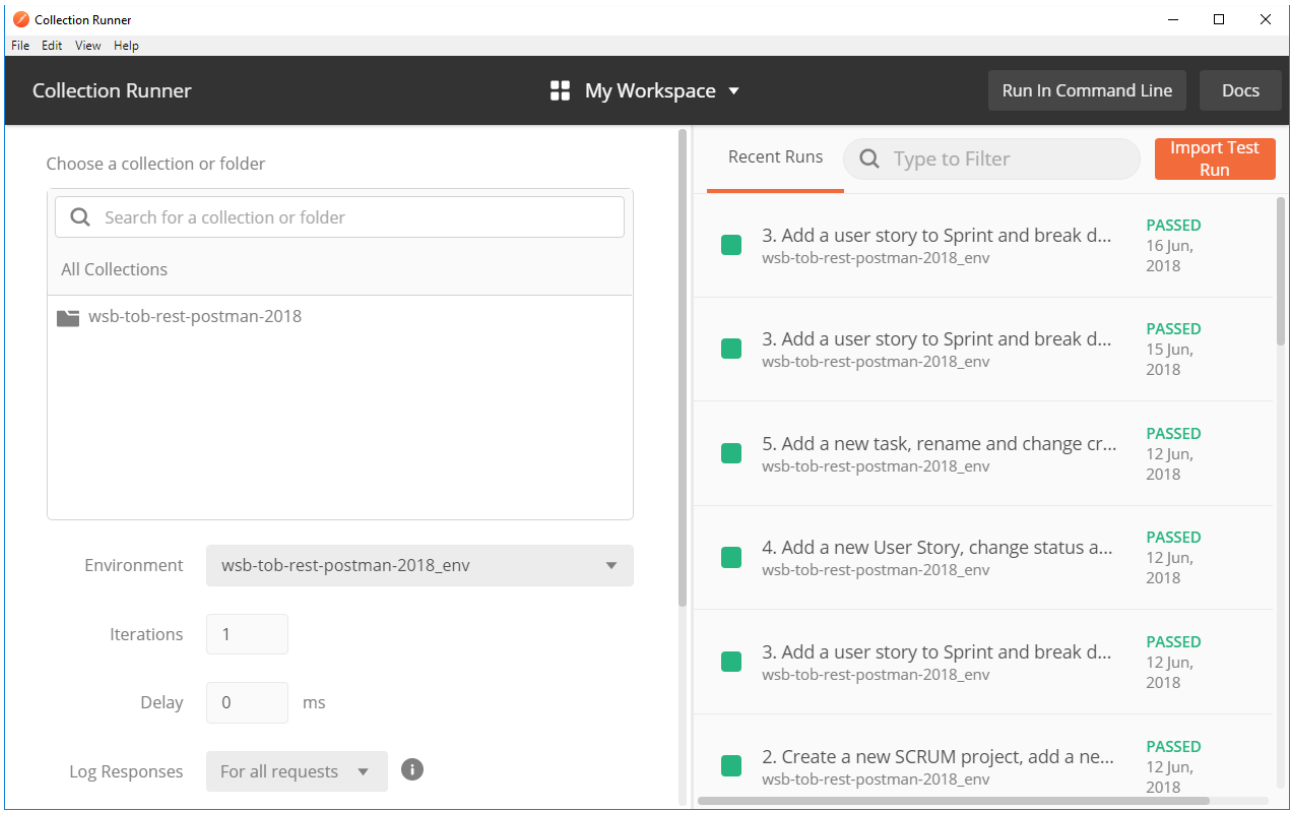
Wywołanie usługi weryfikacji istnienia użytkownika wykorzystuje jako parametr zmienną środowiskową zawierającą identyfikator użytkownika przekazany z testu założenia konta użytkownika.



Weryfikacja poprawności działania scenariuszy

Weryfikacja poprawności zdefiniowanych kolekcji testów wykonywane jest poprzez uruchomienie z menu Postman narzędzia *Collection Runner*.

Collection Runner pozwala na uruchomienie kolekcji (scenariuszy) oraz dokumentuje wyniki poszczególnych przypadków testowych, co pozwala na weryfikację poprawności działania całego scenariusza testowego.



Newman - automatyczne uruchamianie scenariuszy

Newman umożliwia uruchomienie scenariuszy testowych utworzonych w Postman przy pomocy linii poleceń. Otwiera to możliwość użycia przygotowanych w ten sposób testów w środowisku ciągłej integracji.

Opracowane kolekcje i przypadki testowe należy zapisać do pliku wraz z plikiem ustawień środowiska testowego w wybranym katalogu. Uruchomienie scenariuszy i przypadków testowych realizowane jest przy pomocy narzędzia Newman komendą:

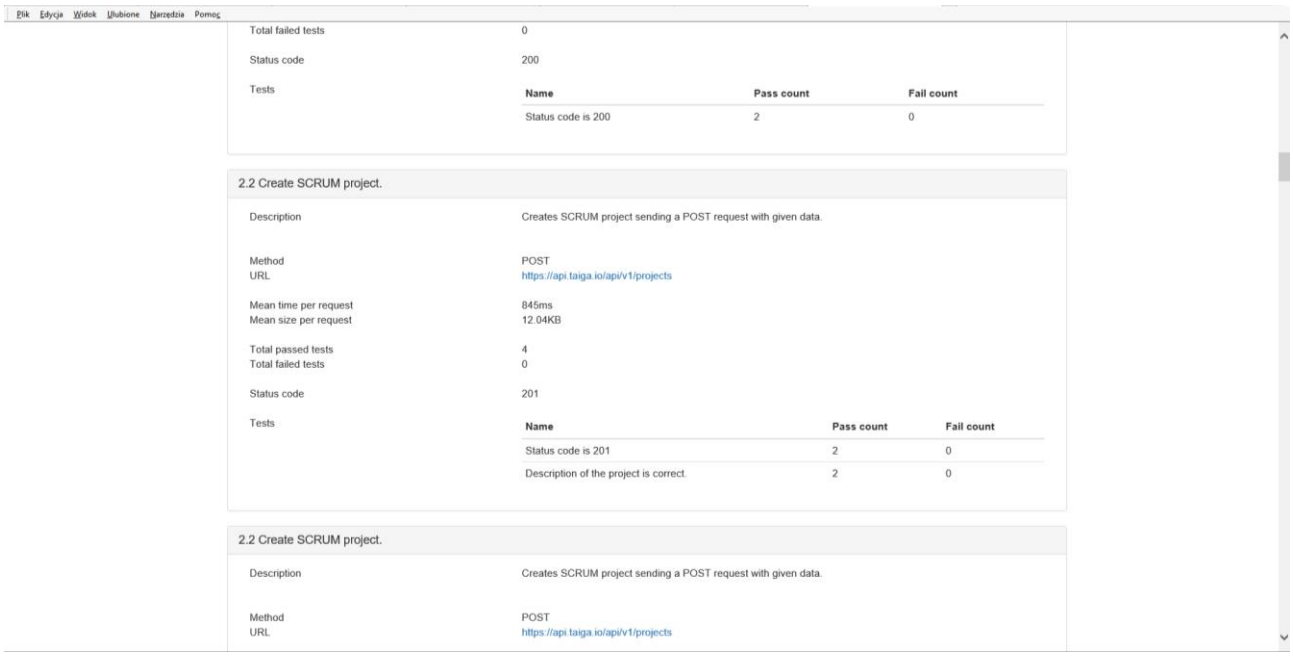
```
newman run [katalog projektu]/<kolekcja>.json
-e [katalog projektu]/ <ustawienia środowiska>.json
--reporters html,cli
```

gdzie:

- `[katalog projektu]/` **`<kolekcja>.json`** - ścieżka do pliku ze scenariuszami testowymi,
- `[katalog projektu]/` **`<ustawienia środowiska>.json`** – ścieżka do pliku ze zmiennymi środowiskowymi,
- **`--reporters html,cli`** – tworzy raport z przebiegu testów w katalogu newman w formacie HTML oraz wyświetla raport bezpośrednio na ekran.

Wyniki wykonania testów automatycznych dostępne są w pliku: `[katalog projektu]/newman/newman-run-report-[yyyy-mm-dd-hh-mm-ss-ms-n].html`.

Fragment raportu z wykonania testów:


Plik Edycja Widok Ulubione Narzędzia Pomoc

Total failed tests 0

Status code 200

Tests

Name	Pass count	Fail count
Status code is 200	2	0

2.2 Create SCRUM project.

Description Creates SCRUM project sending a POST request with given data.

Method POST
URL https://api.taiga.io/api/v1/projects

Mean time per request 845ms
Mean size per request 12.04KB

Total passed tests 4
Total failed tests 0

Status code 201

Tests

Name	Pass count	Fail count
Status code is 201	2	0
Description of the project is correct.	2	0

2.2 Create SCRUM project.

Description Creates SCRUM project sending a POST request with given data.

Method POST
URL https://api.taiga.io/api/v1/projects

Raport z przebiegu testów może być na przykład umieszczony w repozytorium lub przesłany na skrzynkę e-mail osoby odpowiedzialnej za przebieg testów.

Newman został zbudowany od podstaw jako biblioteka, co oznacza że może być rozbudowywany i wykorzystywany na różne sposoby, na przykład w samodzielnie napisanym kodzie Node.js.

Newman może być również wykorzystany w systemach ciągłej integracji, takich jak Jenkins czy Bamboo. Uruchamianie testów w ten sposób oznacza możliwość szybkiego wykrywania błędów w oprogramowaniu, każdorazowo po jego kompilacji zgodnie z ustalonym harmonogramem.

Przykład takiej integracji dostępny jest na stronie:
https://www.getPostman.com/docs/v6/Postman/collection_runs/integration_with_jenkins.

Wnioski

Postman jest narzędziem pozwalającym na efektywne i elastyczne tworzenie i zarządzanie scenariuszami testowymi. W porównaniu z innymi rozwiązaniami oferuje przyjazny interfejs użytkownika dla definiowania i weryfikacji działania przypadków testowych, łatwy dostęp do debugowania ich działania oraz automatyczne uruchamianie wraz z tworzeniem raportów zawierających wyniki testów.

Głównymi adresatami rozwiązania są testerzy i zespoły testerów, którzy dzięki niemu mogą tworzyć i zarządzać scenariuszami i przypadkami testowymi, dokumentować i uruchamiać je oraz umieszczać przypadki w repozytorium celem wersjonowania i udostępniania scenariuszy do automatycznego uruchamiania przez Newman. Postman stanowi alternatywę w stosunku do rozwiązań SoapUI, Ranorex, Katalon Studio czy Tricentis.

Użycie Newman pozwala natomiast na wsadowe uruchamianie zdefiniowanych scenariuszy testów w ramach wykorzystywanego rozwiązania do ciągłej integracji, umożliwiając jednocześnie dostarczanie raportów z wykonania testów w sposób przyjęty w danej organizacji, na przykład poprzez umieszczenie ich w repozytorium lub wysłanie na skrzynki e-mail osób odpowiedzialnych za kontrolę jakości oprogramowania.