**WYDZIAŁ MATEMATYKI I INFORMATYKI UNIWERSYTETU ŁÓDZKIEGO**

Kierunek studiów: Informatyka

Specjalizacja: Grafika Komputerowa i Projektowanie Gier

**Aplikacja internetowa „System zgłaszania błędów”**

Mikołaj Szymański

Numer albumu: 359892

Praca licencjacka napisana  
w Katedrze Funkcji Rzeczywistych  
pod kierunkiem  
dr Sebastiana Lindnera

**Łódź 2018**

Spis treści

[Wstęp 3](#_Toc513246509)

[1. Wzorce architektury używane w projektowaniu aplikacji. 3](#_Toc513246510)

[**1.1.** Opis wzorca architektury MVC 3](#_Toc513246511)

[**1.2.** Opis wzorca architektury MVP 3](#_Toc513246512)

[**1.3.** Opis wzorca architektury MVVM 3](#_Toc513246513)

[**1.4.** Który wzorzec jest odpowiedni dla mojej aplikacji? 3](#_Toc513246514)

[2. Dokumentacja użytkownika 4](#_Toc513246515)

[**2.1.** Do czego służy aplikacja 4](#_Toc513246516)

[**2.2.** Instrukcja obsługi aplikacji 4](#_Toc513246517)

[**2.2.2.** Tworzenie zgłoszenia 6](#_Toc513246518)

[**2.2.3.** Przeglądanie zgłoszeń 7](#_Toc513246519)

[**2.2.4.** Edycja zgłoszeń 10](#_Toc513246520)

[3. Dokumentacja programisty 14](#_Toc513246521)

[**3.2.** Użyte narzędzia programistyczne 14](#_Toc513246522)

[**3.2.1.** Instalacja 14](#_Toc513246523)

[**3.2.2.** Konfiguracja 14](#_Toc513246524)

[**3.3.** Opis działania kodu 14](#_Toc513246525)

[**3.4.** Klasy 14](#_Toc513246526)

[**3.4.1.** Klasa Account 14](#_Toc513246527)

[**3.4.2.** Klasa Ticket 15](#_Toc513246528)

[**3.4.3.** Klasa FileUploadHelperExtension 15](#_Toc513246529)

[**3.5.** Opis działania wybranych funkcji 15](#_Toc513246530)

[**3.5.1.** Funkcja DownloadFile 15](#_Toc513246531)

[**3.6.** Graficzna reprezentacja schematu bazy danych 15](#_Toc513246532)

[4. Podsumowanie 15](#_Toc513246533)

[5. Bibliografia 15](#_Toc513246534)

# Wstęp

Tu będzie wstęp.

# Wzorce architektury używane w projektowaniu aplikacji.

## Opis wzorca architektury MVC

Wzorzec Model-View-Controller jest przeznaczony do projektowania aplikacji internetowych. **Model** reprezentuje naszą warstwę biznesową, czyli nie tylko klasy mapujące dane pobrane z bazy danych. To również funkcjonalności potrzebne do działania naszej aplikacji . **View** jest odpowiedzialny za prezentowanie danych użytkownikowi, pobieranych z Modelu, który jest dla niego tylko do odczytu. **Controller** przekierowuje wszelkie żądania użytkownika na wywołanie metod konkretnego Modelu, oraz przygotowuje dane dla Widoku i żądaniem jego wygenerowania. Jeden kontroler może obsługiwać kilka widoków.

## Opis wzorca architektury MVP

Wzorzec Model-View-Presenter używany jest głównie do projektowania aplikacji mobilnych. **Presenter** w tym wzorcu ma podobne zastosowanie co kontroler w MVC, lecz dodatkowo posiada on w sobie logikę biznesową, która w MVC jest pod Modelem. Tutaj dane nie są przekazywane bezpośrednio z modelu na widok, ale prezenter wykonuje zapytania do modelu o pewne wartości, które później są przetwarzane i wysyłane na widok. **Model** jest zwykłą reprezentacją danych, zawierająca powiązania i struktury danych. **View** ma to samo zadanie co w architekturze MVC, wyświetlić nasze żądania. W tej architekturze, presenter odnosi się do jednego widoku, nie może obsługiwać wielu.

## Opis wzorca architektury MVVM

Wzorzec Model-View-ViewModel to wzorzec wykorzystywany głównie do aplikacji graficznych, w których dane są mocno ze sobą powiązane. Powstał na bazie architektury MVC, i stąd **Model** jest niezmienny w swoich zastosowaniach. **View** jest interfejsem użytkownika, który nie ma pojęciao **ViewModel-u**. **ViewModel** pobiera dane z Modelu i przygotowuje je do wyświetlenia na możliwie wiele Widoków. Ten wzorzec nie ogranicza się do ilości obsługiwanych widoków.

## Który wzorzec jest odpowiedni dla mojej aplikacji?

Każdy z opisanych wzorców ma swoje wady i zalety. Zastosowanie **MVP** ogranicza nas do obsługi tylko jednego widoku, przy użyciu **MVC** bądź **MVVM** nie musimy się martwić o limit obsługi widoków. Jednak w porównaniu do **MVVM**, popularność wśród aplikacji internetowych architektury **MVC** jest zauważalnie większa**.** Użycie tej architektury pomaga programistom współpracować ze sobą nad jednym kodem. Dzięki modułowej budowie (Model – View- Controller) modyfikacja kodu jest prosta a sam kod jest przejrzystszy.

# Dokumentacja użytkownika

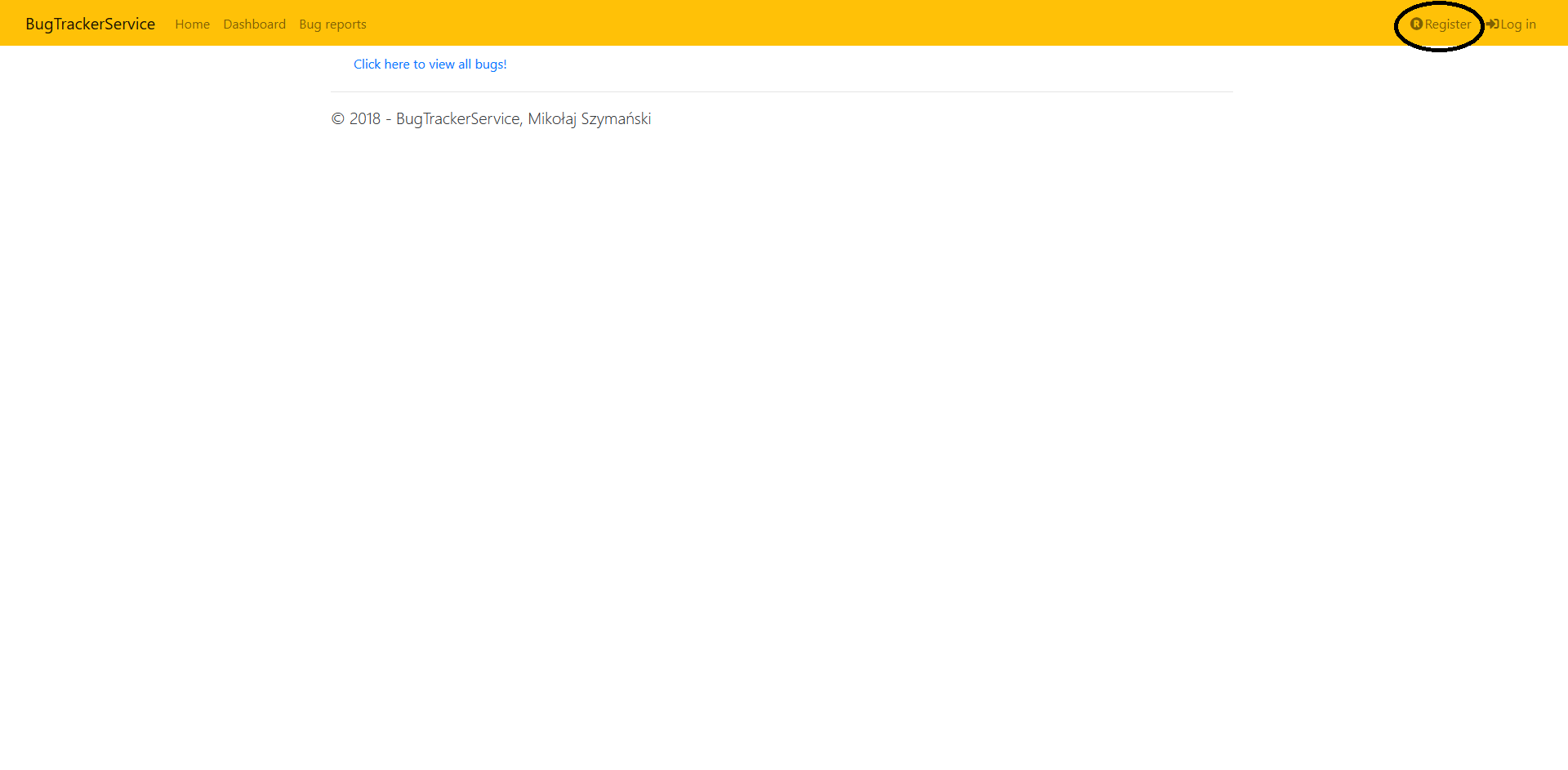
## Do czego służy aplikacja

Aplikacja została stworzona do użytku przez firmy informatyczne, które mogą kontrolować ilość błędów w programach tworzone, bądź użytkowane przez nie. Użytkownik, czyli osoba posiadająca do własnego użytku konkretny program, po zarejestrowaniu się na stronę, posiada możliwość zgłoszenia błędu wynikłego podczas korzystania z programu komputerowego. Podając szczegóły, wrzucając zdjęcia obrazujące powstały ambaras, użytkownik zaznajamia serwisantów. Serwisanci po przeanalizowaniu problemu, podejmują po swojej stronie odpowiednie kroki, żeby zażegnać kłopot.

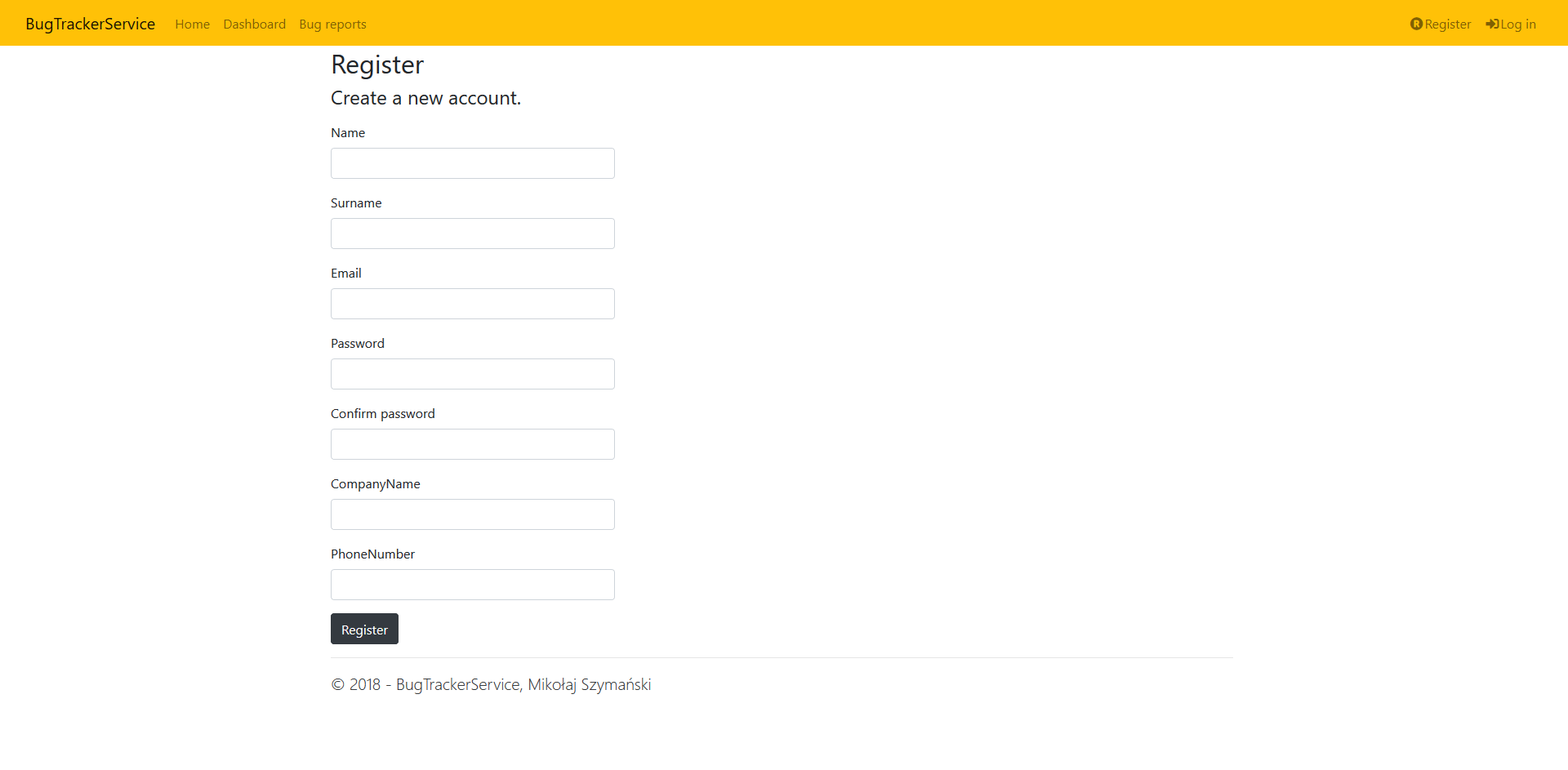
## Instrukcja obsługi aplikacji

* + 1. Rejestracja użytkownika

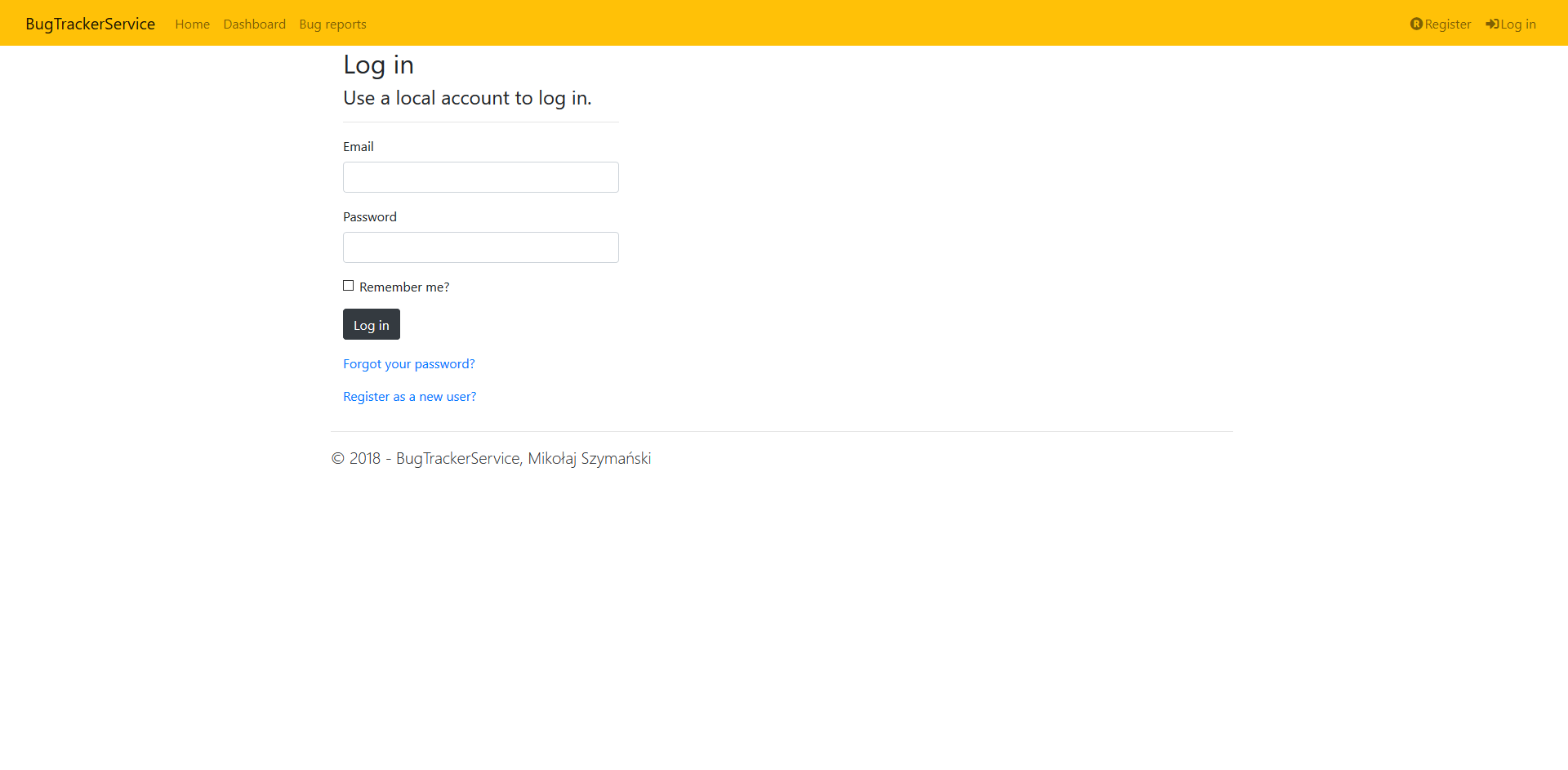
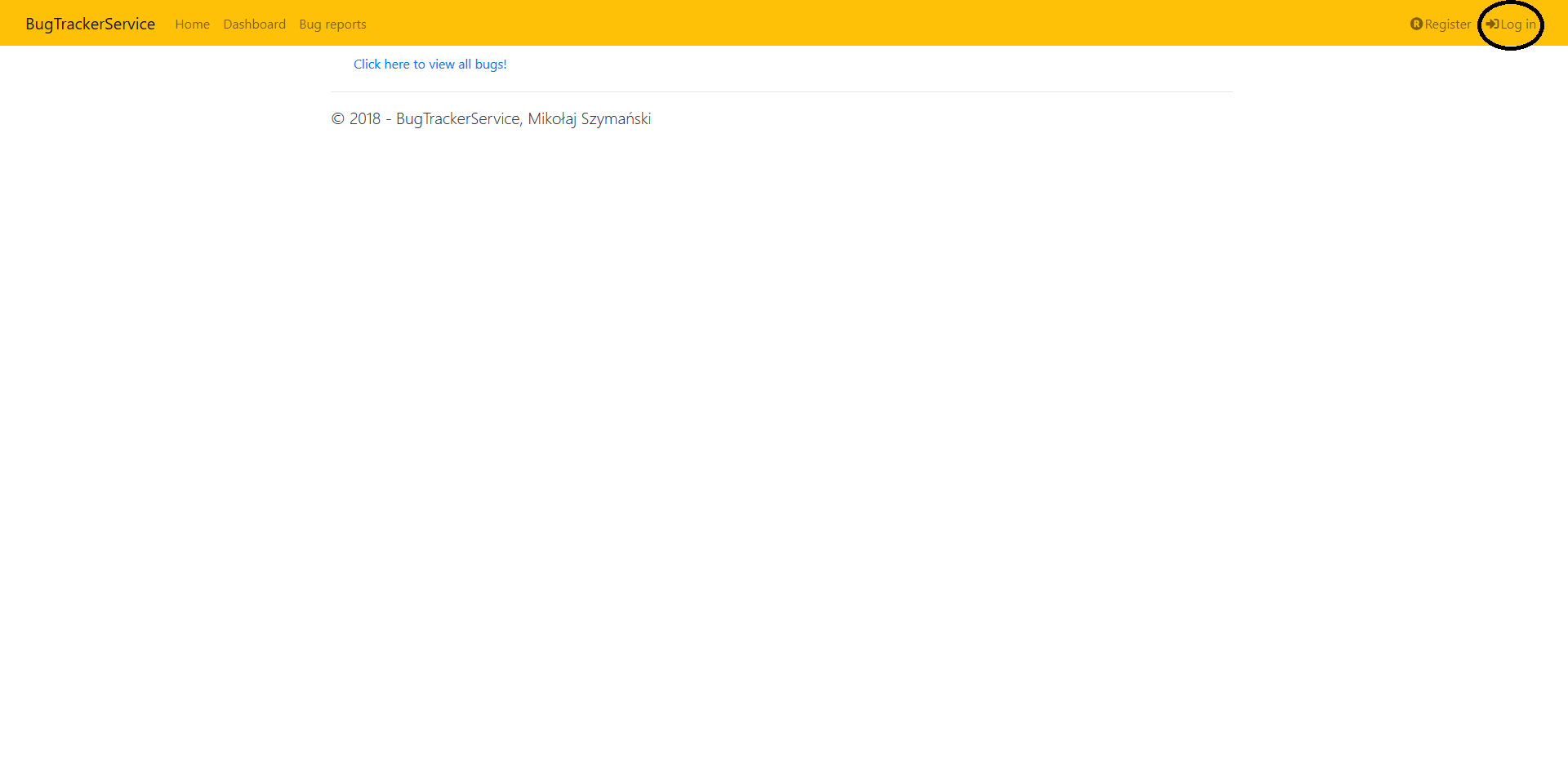
Użytkownik, który posiada oprogramowanie wystawione przez konkretną firmę, może zarejestrować się na stronę przez kliknięcie w górnym prawym rogu **Register**.



Po ukazaniu się formularza rejestracji, użytkownik podaje swoje dane, czyli imię i nazwisko, swojego e-maila potrzebnego do potwierdzenia (bez tego nie można korzystać z aplikacji), swoje hasło oraz jego potwierdzenie składające się z przynajmniej 8 znaków w tym: jedna wielka litera oraz znak specjalny, nazwa firmy, w której pracuje oraz telefon kontaktowy.

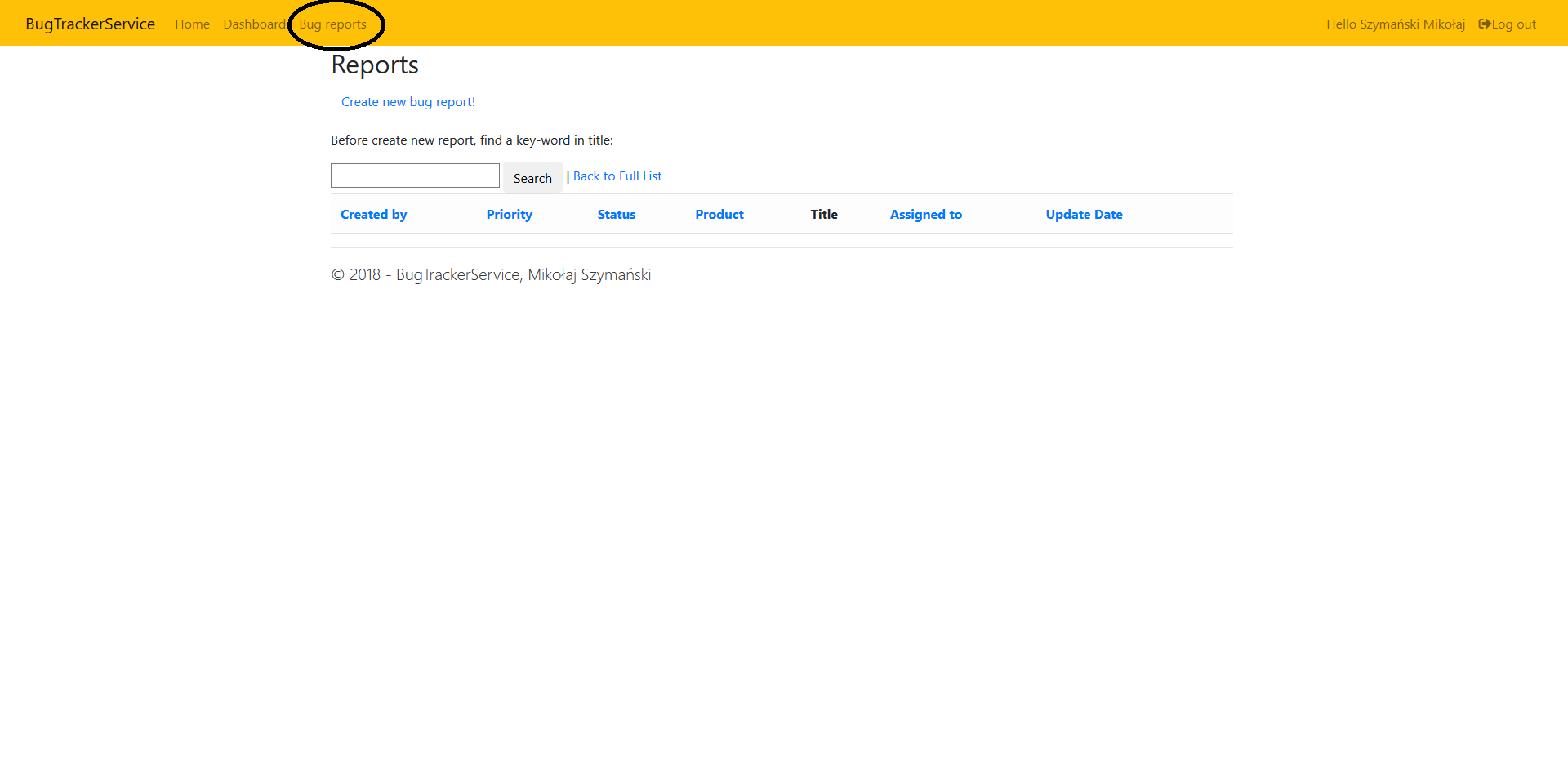


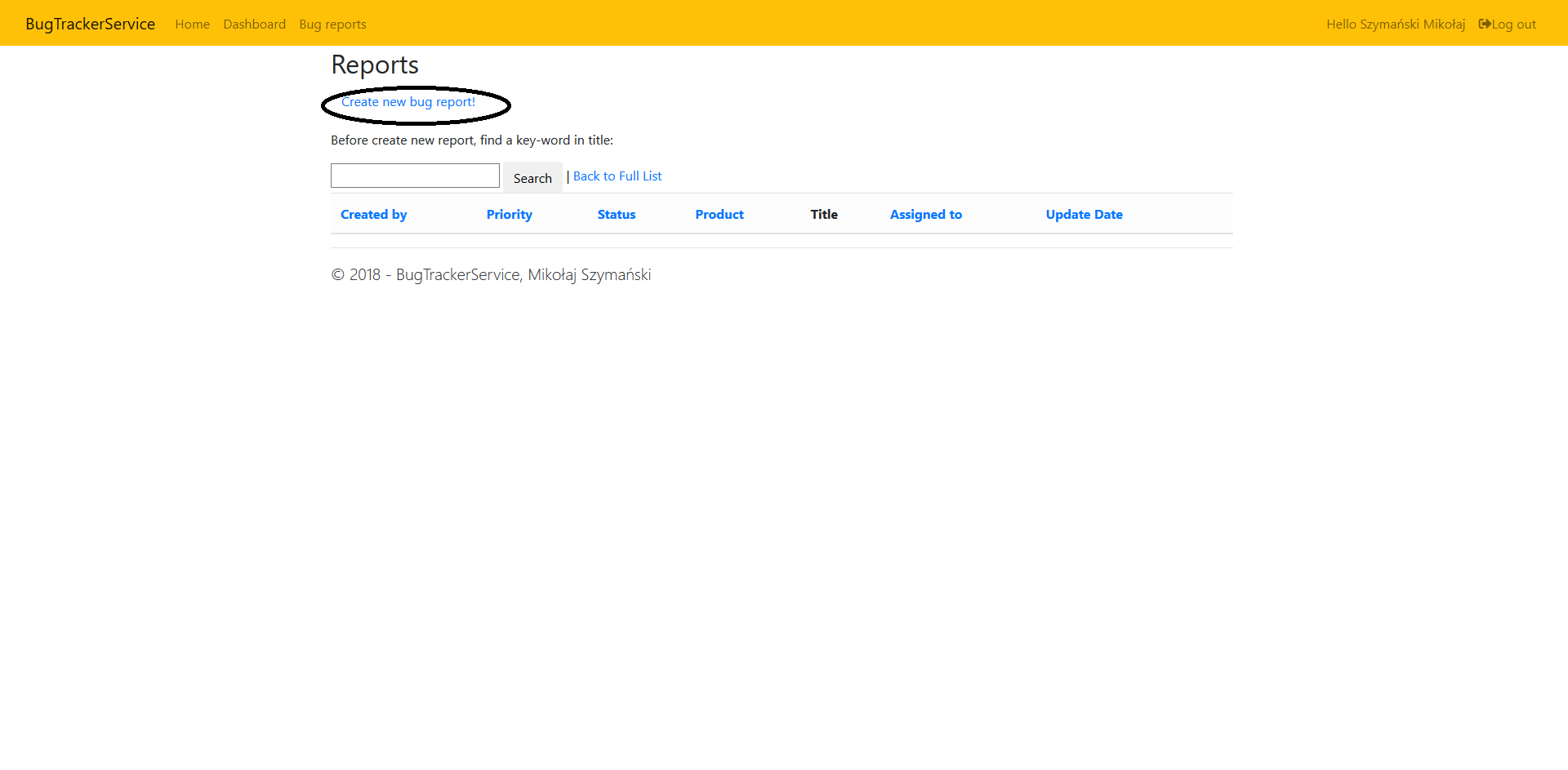
Po kliknięciu przycisku **Register** na dole formularza, użytkownik dostaje e-maila z linkiem do potwierdzenia swojej tożsamości. Po pozytywnej weryfikacji, użytkownik może zalogować się na stronę klikając w prawym górnym rogu ikonkę **Log in**. Podaje w formularzu swój login i hasło podane podczas rejestracji i cieszy się z możliwości oferowanych przez aplikację.



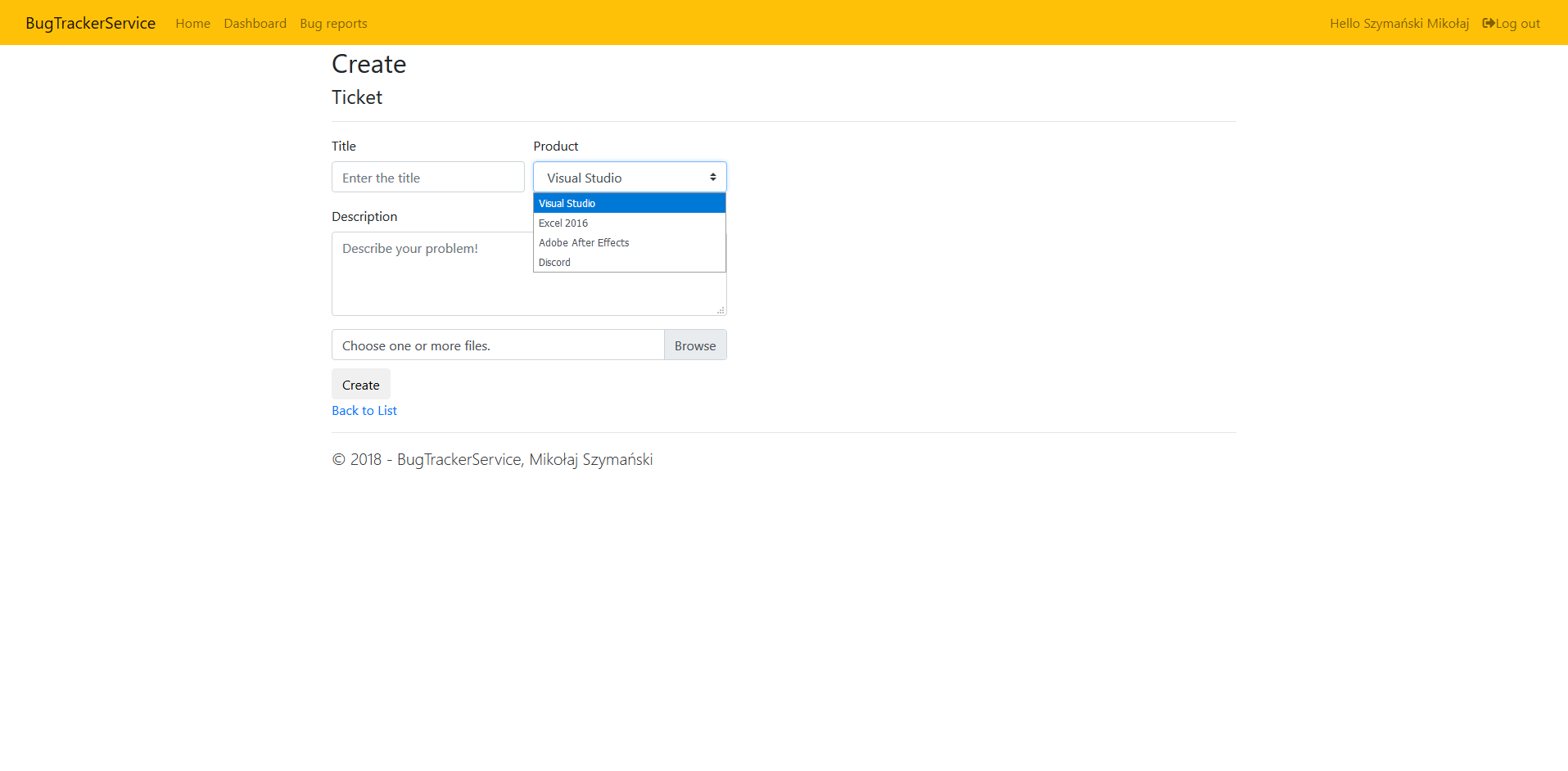
### Tworzenie zgłoszenia

Aby móc stworzyć zgłoszenie, użytkownik musi być zalogowany do serwisu. Na górnym pasku strony kliknąć w zakładkę **Bug Reports.**

 Po kliknięciu w **Create new bug report! o**tworzy nam się prosty formularz do zgłoszenia problemu napotkanego podczas korzystania z jednej z aplikacji.



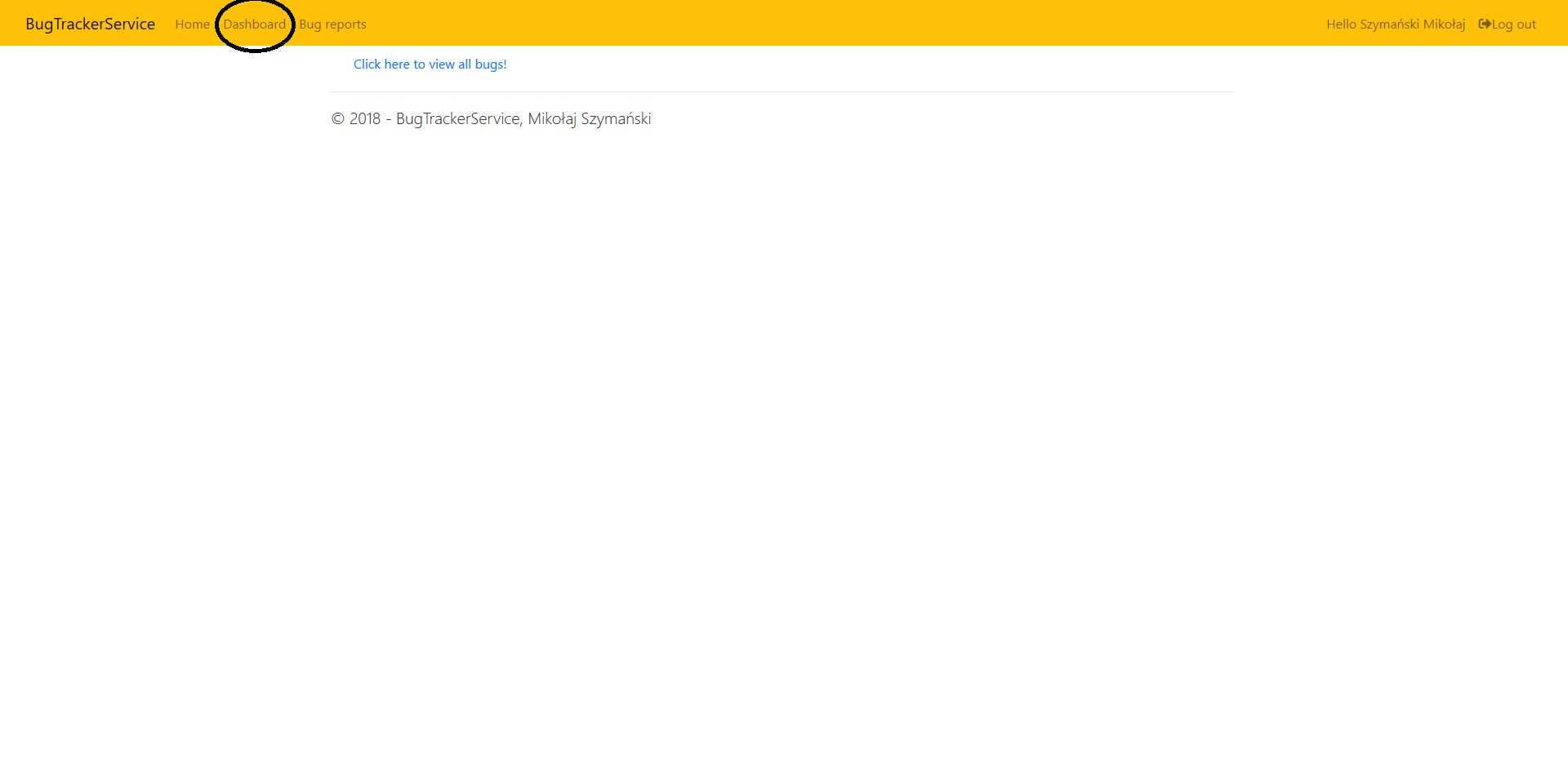
Podajemy w nim tytuł (tak, żeby się dobrze kojarzył z danym problemem), jego szczegółowy opis, wybieramy jakiego oprogramowania się problem tyczy, i jeżeli jest możliwość, to pokazać zdjęcia obrazujące dany problem.



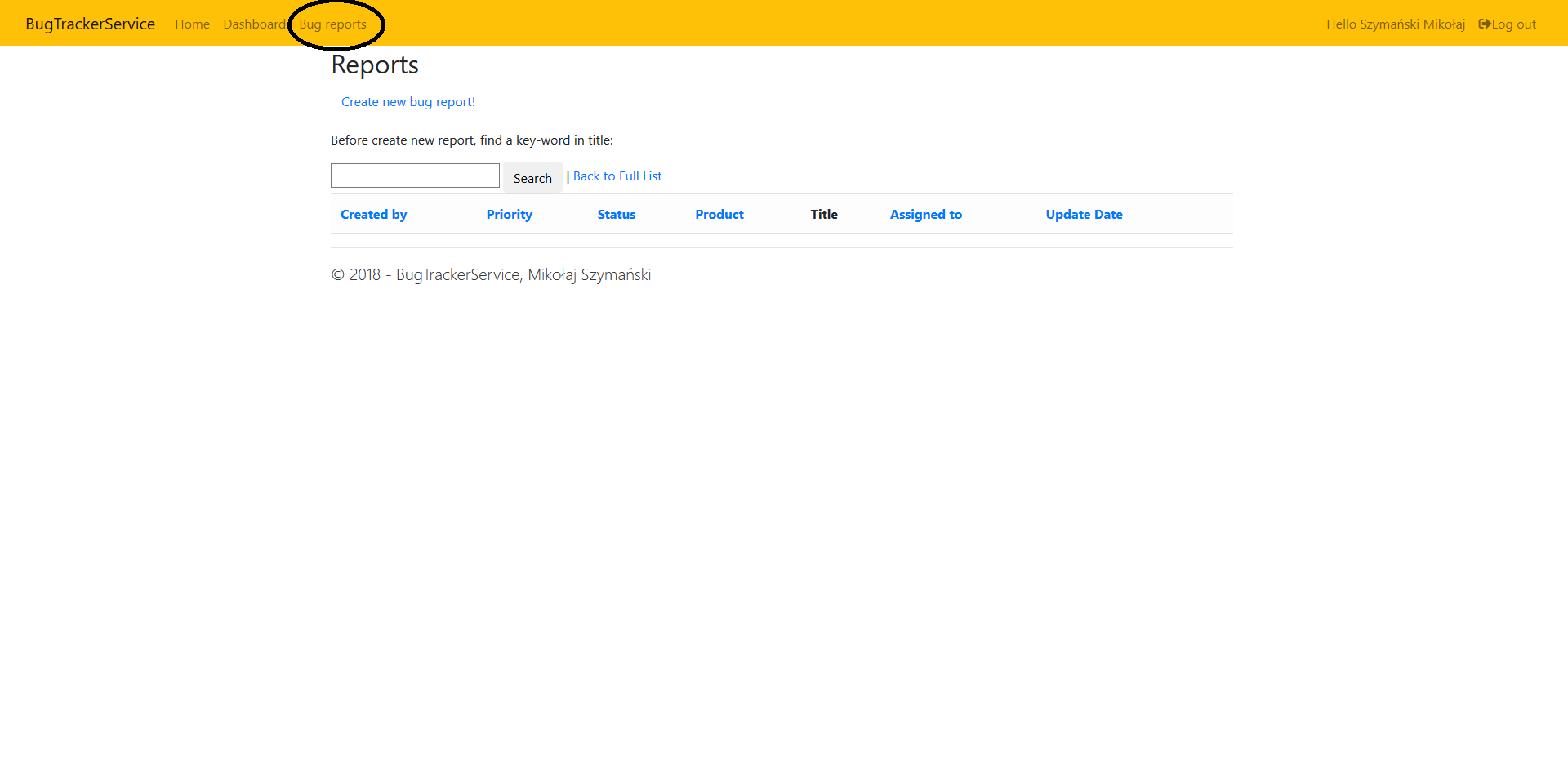
Po kliknięciu przycisku **Create** nasze zgłoszenie zostaje utworzone i każdy kto jest zalogowany może je zobaczyć.

### Przeglądanie zgłoszeń

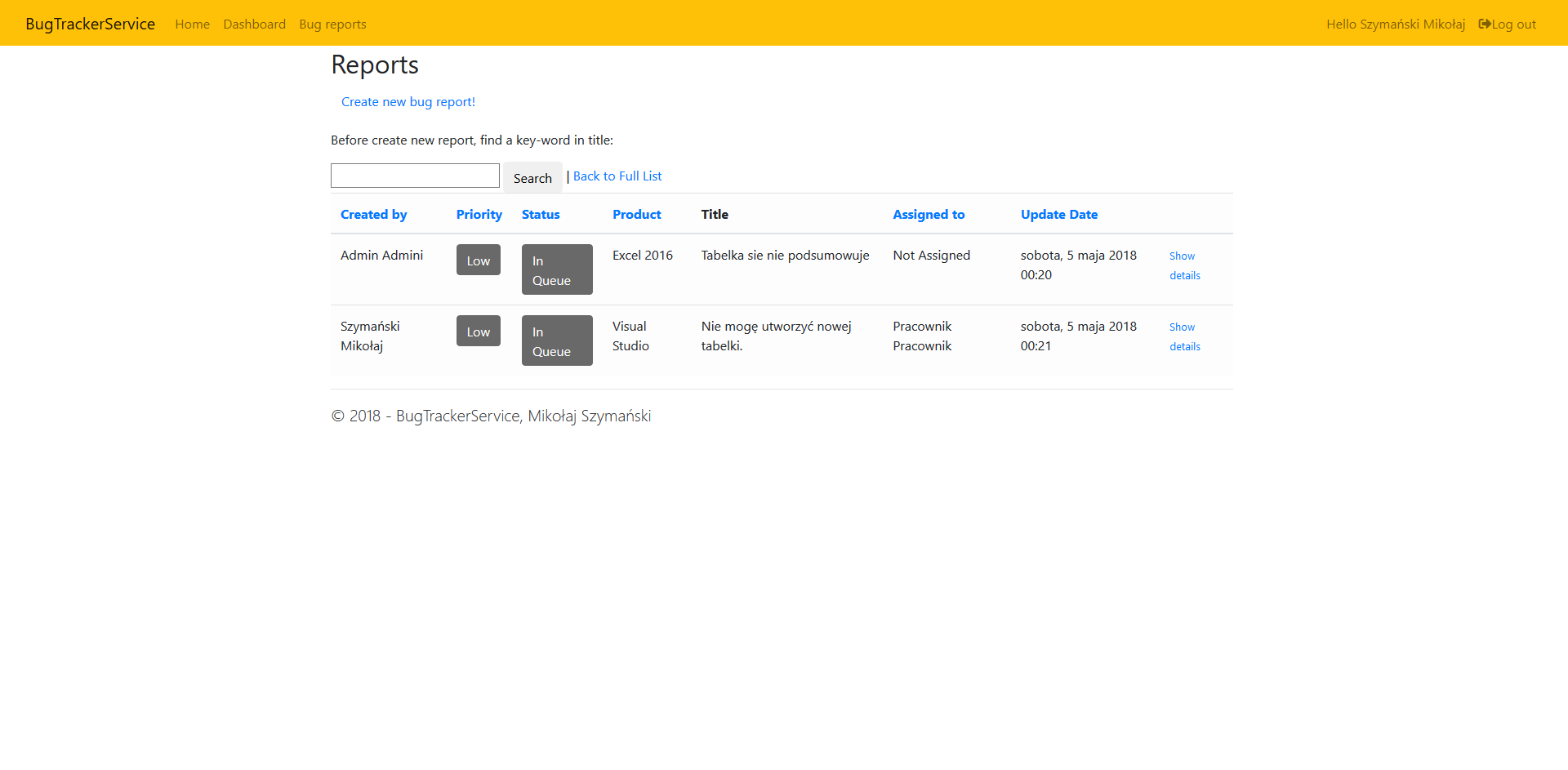
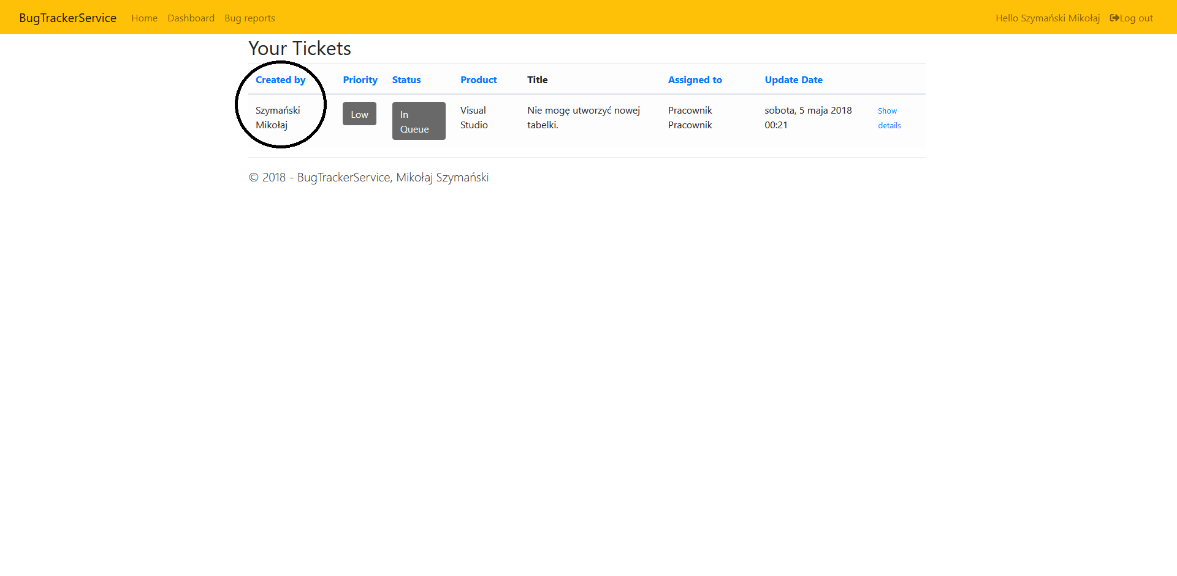
Mamy możliwość przeglądania zgłoszeń swoich, oraz wszystkich. Po kliknięciu w **Dashboard** ukazują nam się wszystkie zgłoszenia stworzone tylko i wyłączne przez nas. Jeżeli jesteśmy serwisantem, pokażą nam się wszystkie zgłoszenia do których jesteśmy przypisani.

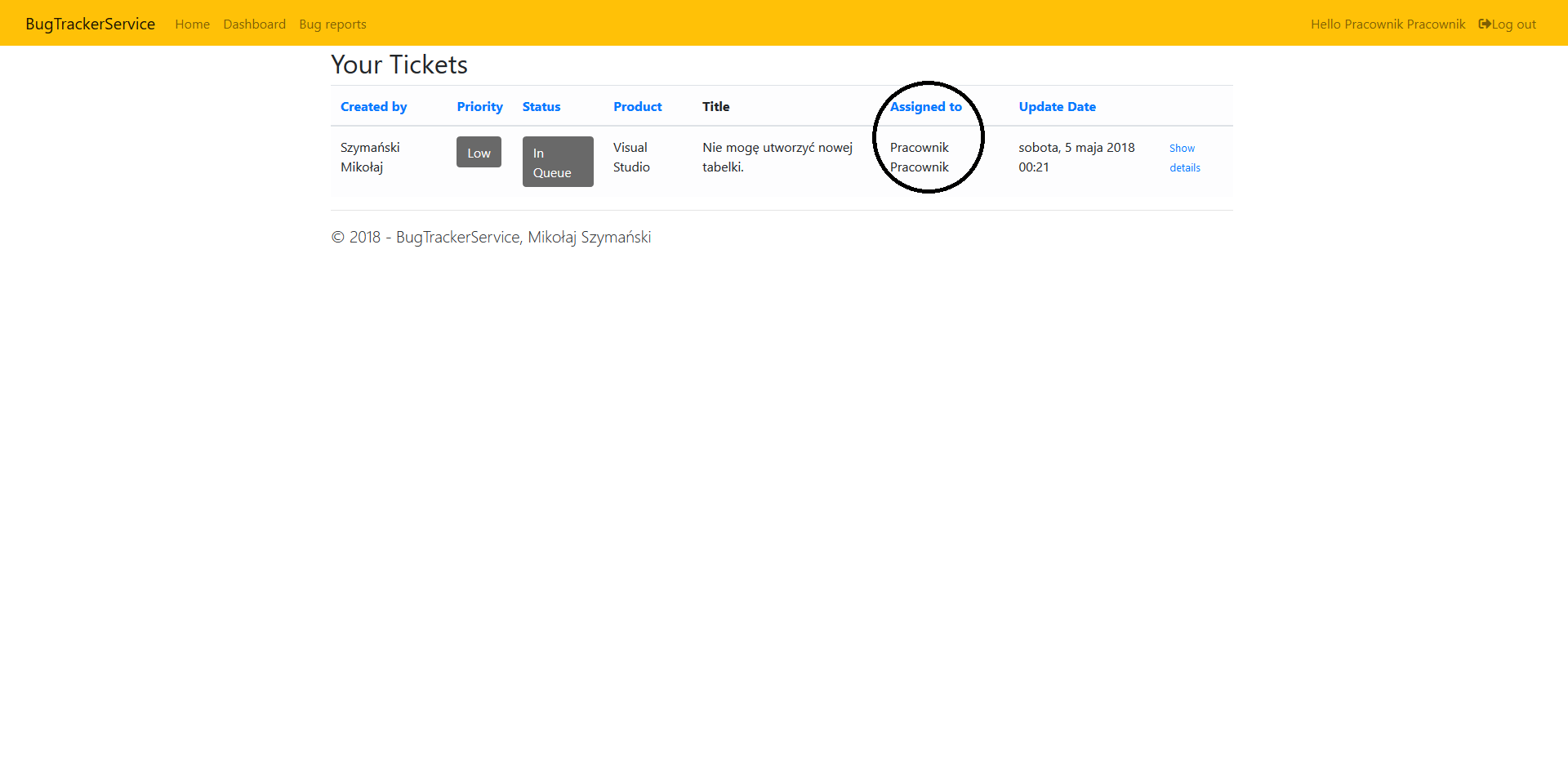


Natomiast jeżeli klikniemy w ikonkę **Bug** **Reports,** to ukażą nam się wszystkie zgłoszenia.

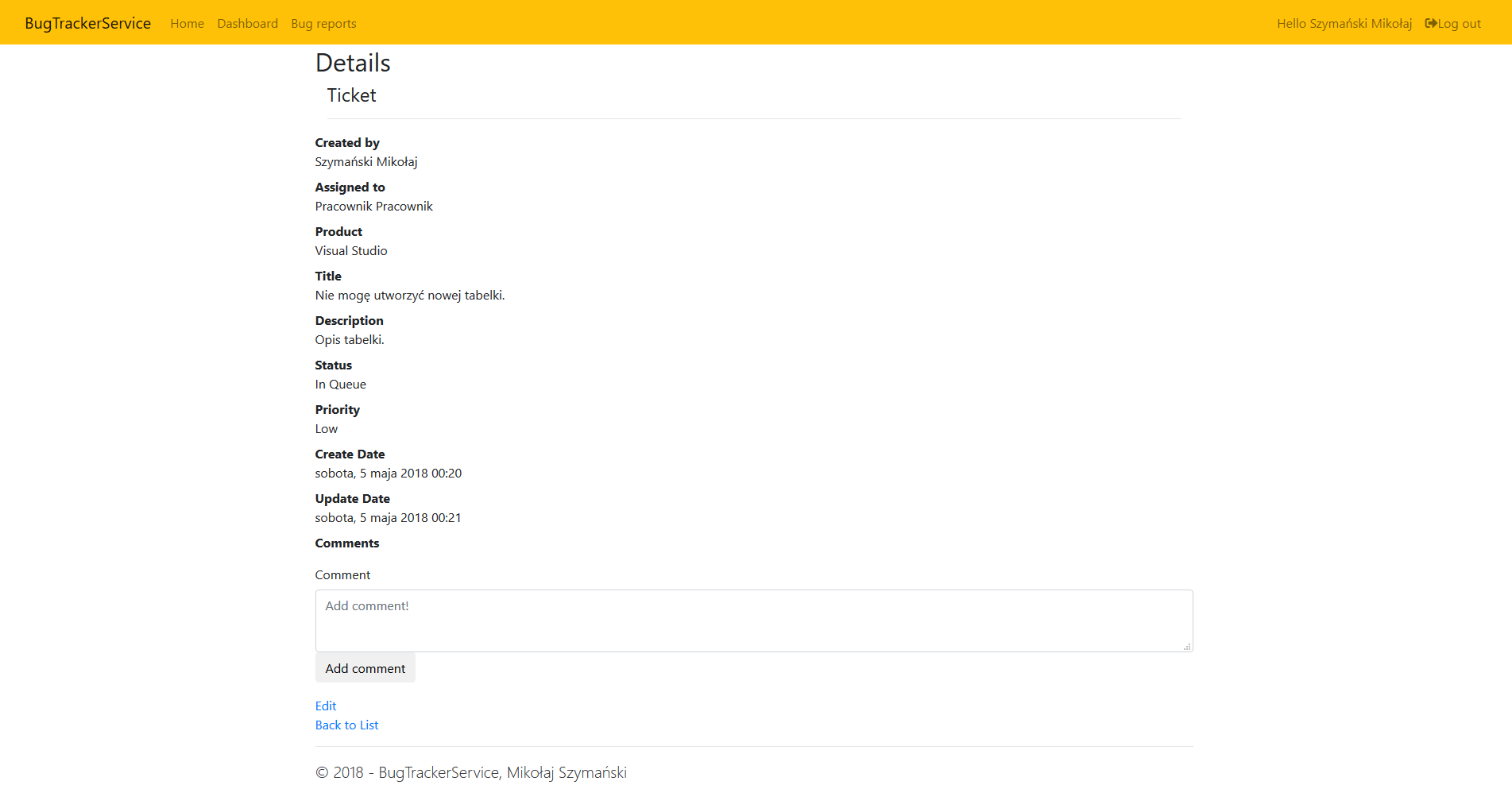


Tabelka pokazuje nam przez kogo został stworzony problem, jego priorytet (od niskiego po wysoki), jego status (od oczekującego do skończonego), jakiego produktu się tyczy, jego tytuł, do kogo jest przypisany i jego data ostatniej aktualizacji.

1. Przegląd wszystkich zgłoszeń
2. Przegląd zgłoszeń stworzonych przez użytkownika
3. Przegląd zgłoszeń przypisanych do pracownika

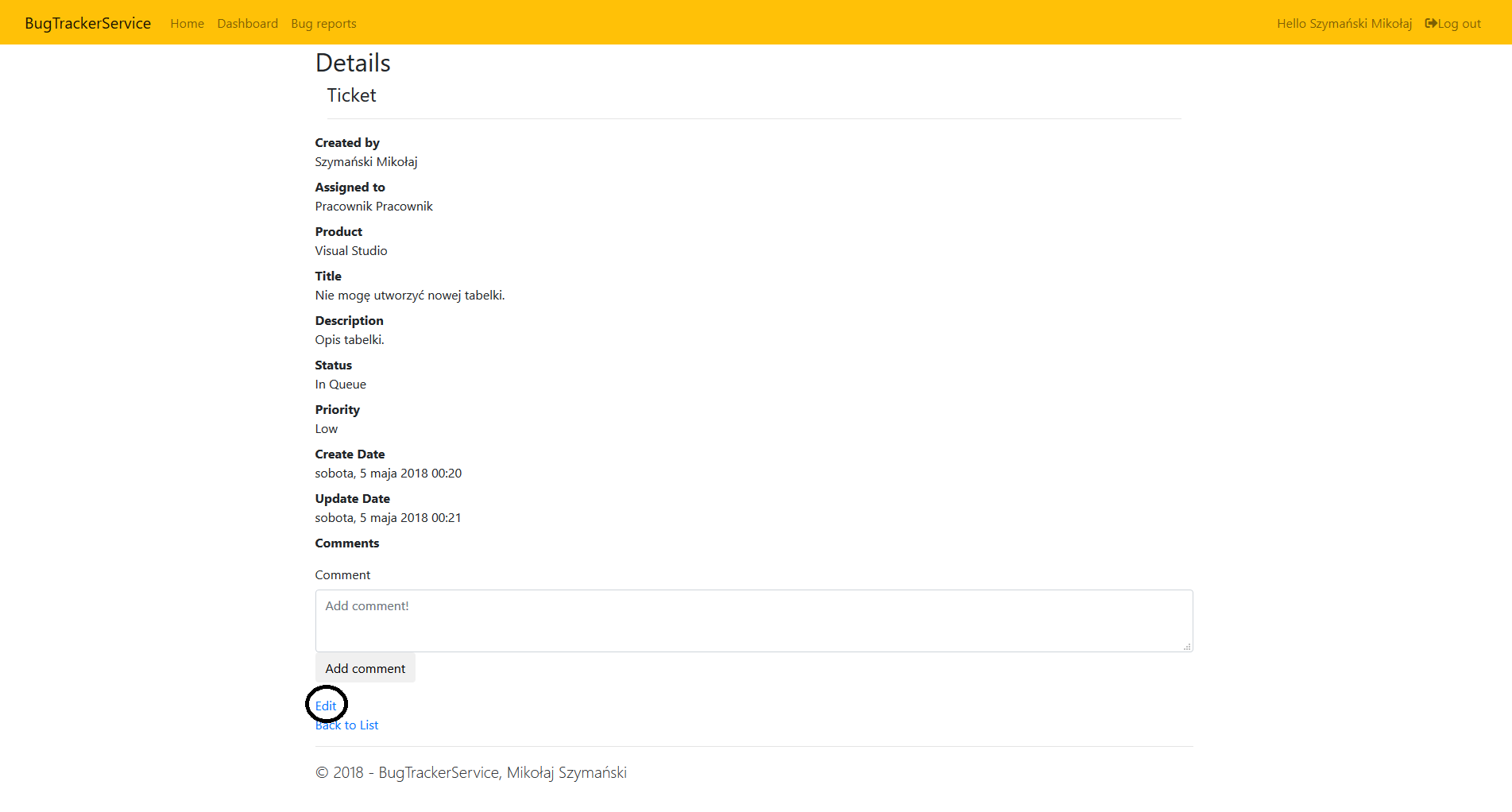


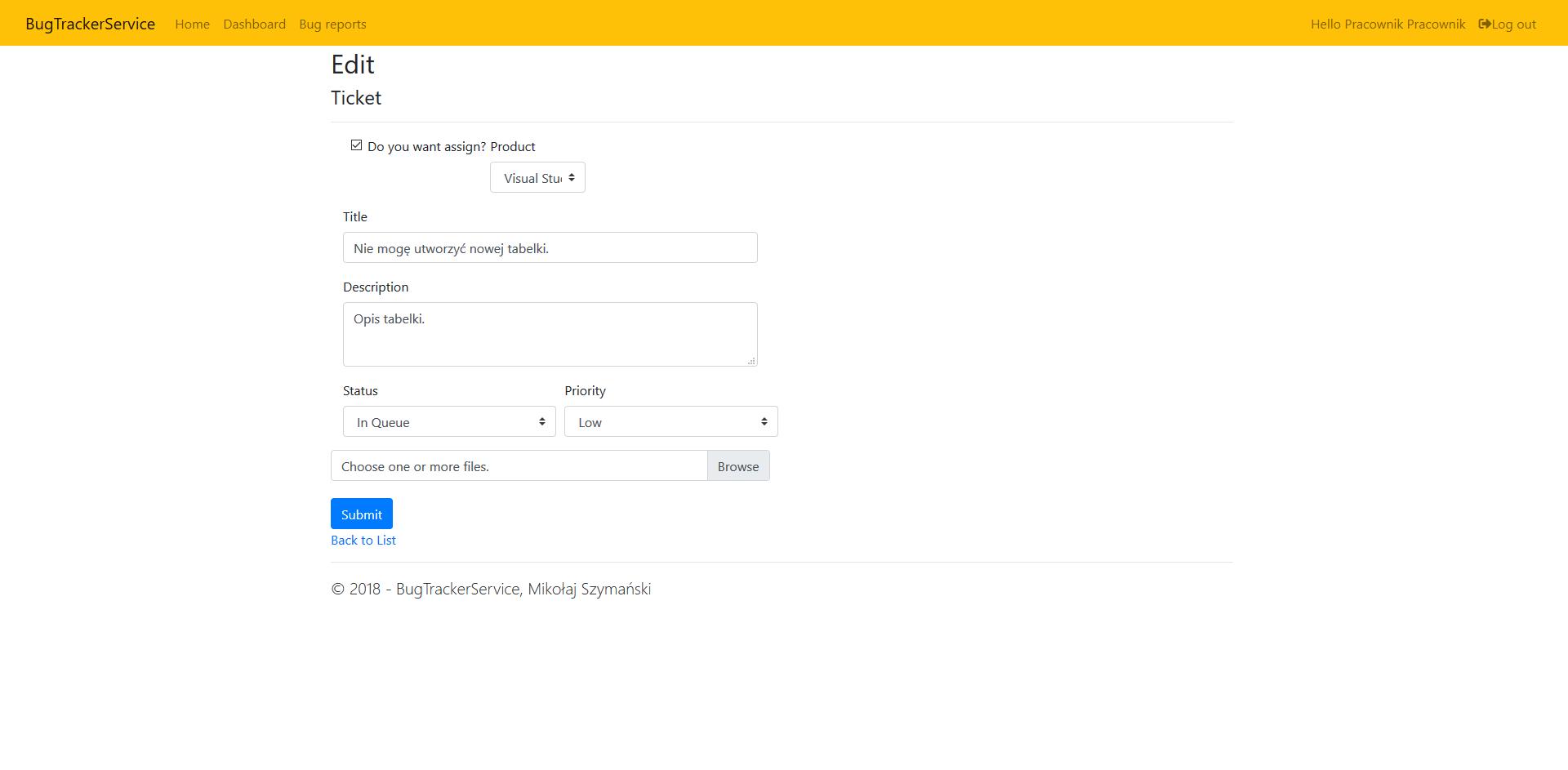
Po kliknięciu w opcję **Show details** ukaże nam się cały opis naszego zgłoszenia. Kto jest przypisany do danego zgłoszenia, jego status oraz priorytet. Możemy również dodawać komentarze do zgłoszenia, co jest szybką formą komunikacji między użytkownikiem a serwisantem.

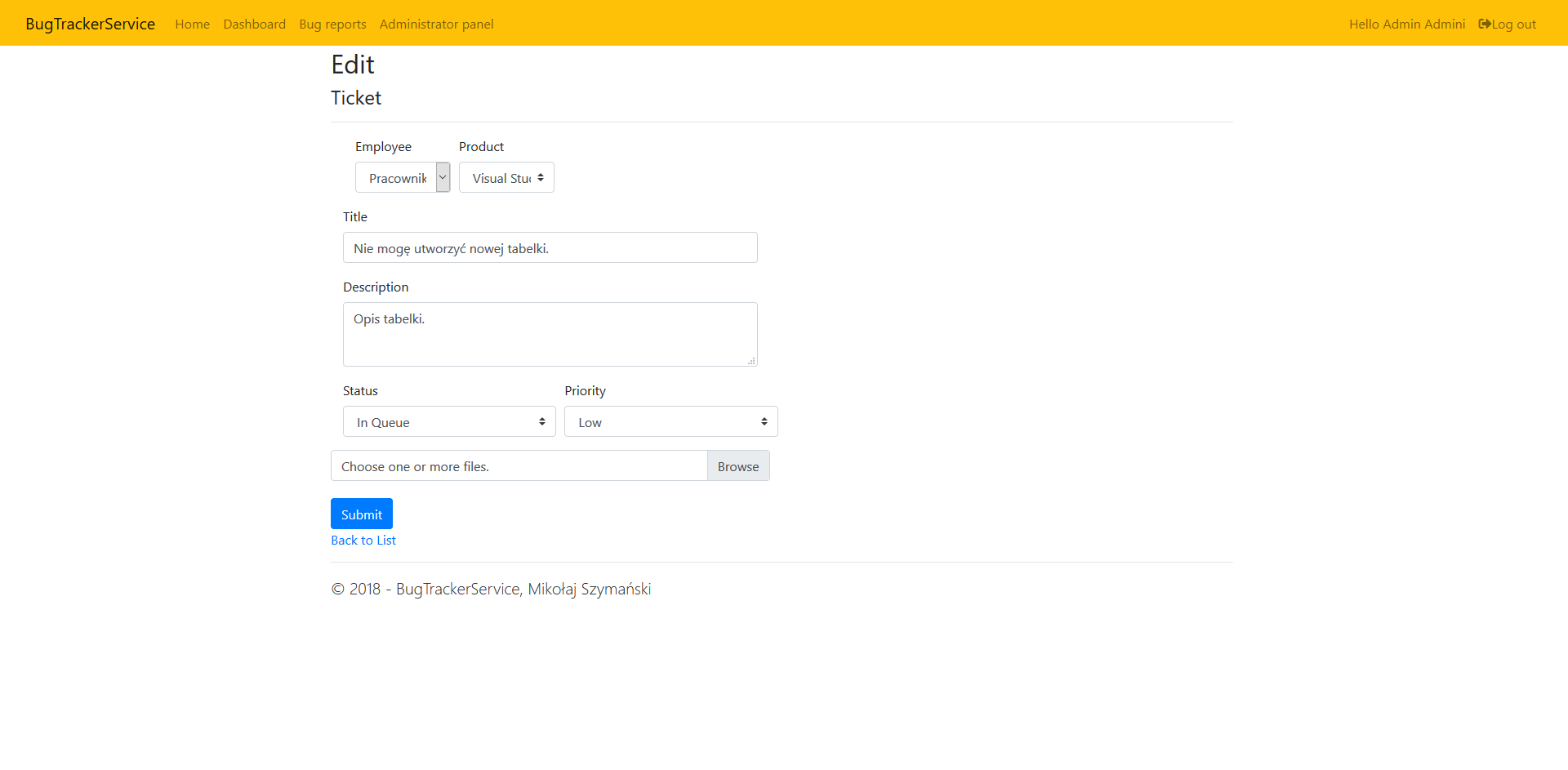


### Edycja zgłoszeń

Jeżeli jakaś rzecz w naszym zgłoszeniu jest zła (zły opis) bądź brakuje pewnych zdjęć do zobrazowania problemu dokładniej, dobrą opcją jest edycja takiego zgłoszenia. Drugą możliwością edycji zgłoszeń jest przypisanie się serwisanta do konkretnego zgłoszenia. Serwisanta może przypisać również główny administrator. Aby dostać się do edycji zgłoszenia, trzeba wejść przez opcję   
**Show details,** a następnie na dole zgłoszenia kliknąć **Edit.**

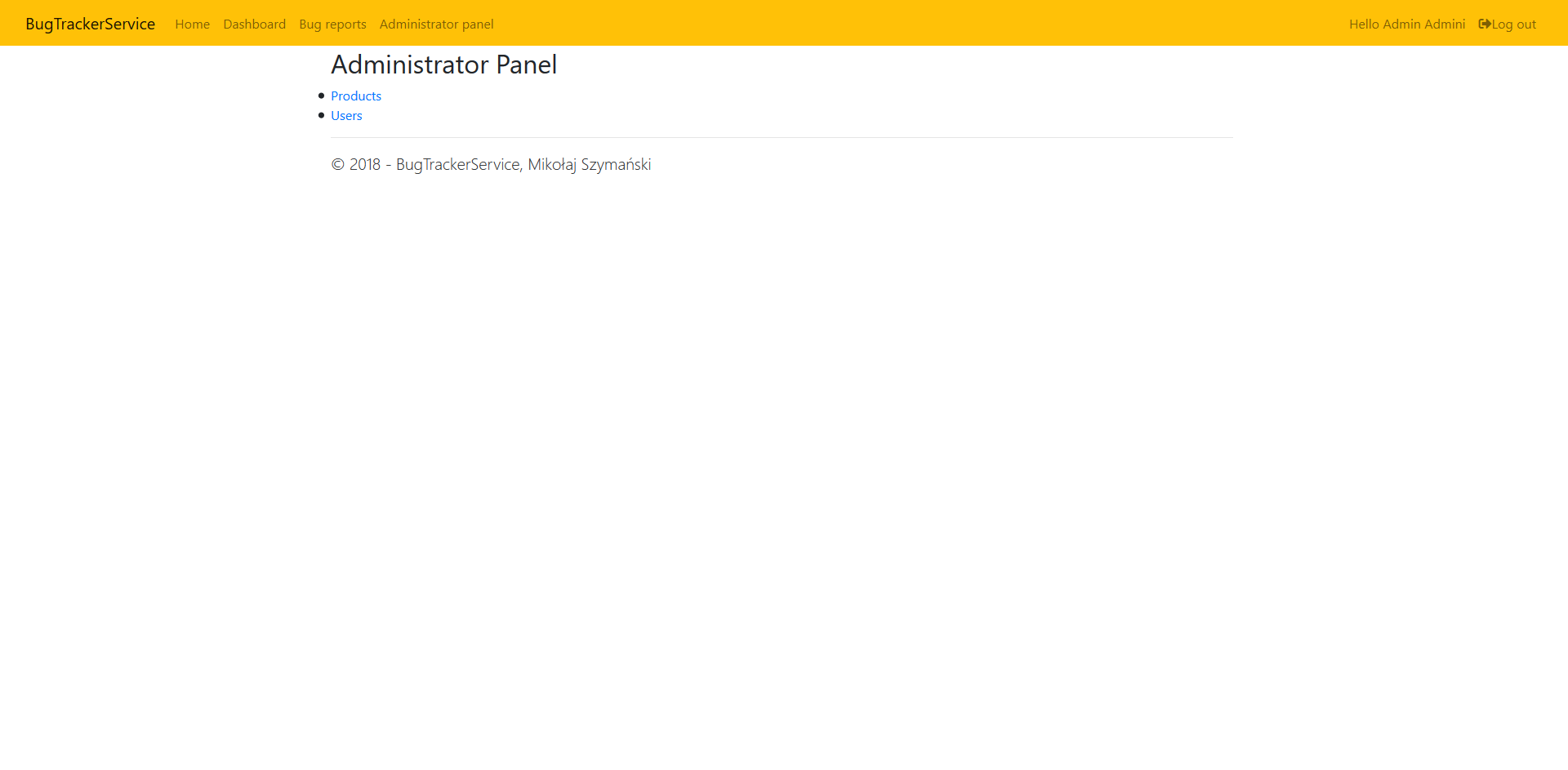


1. Edycja przez serwisanta
2. Edycja przez administratora



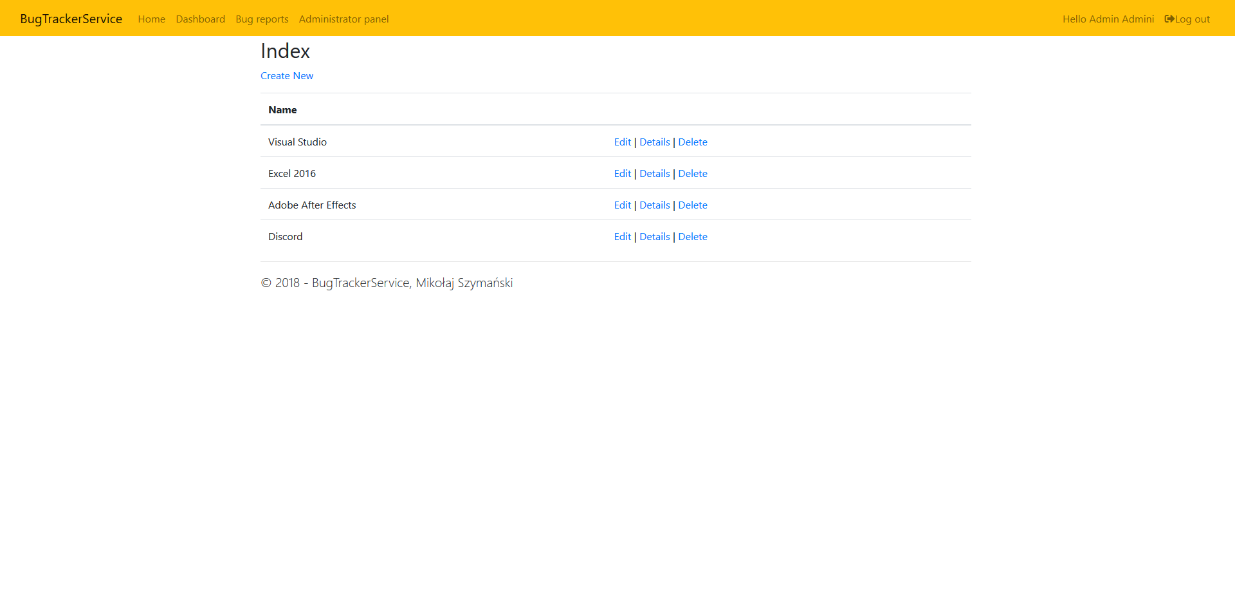
* + 1. Panel administratora

Będąc głównym administratorem serwisu, mamy dostęp do **Panelu administratora,** który umożliwia nam dodanie Produktów (oprogramowania) oraz dodawania pracowników, usuwania użytkowników oraz pracowników.

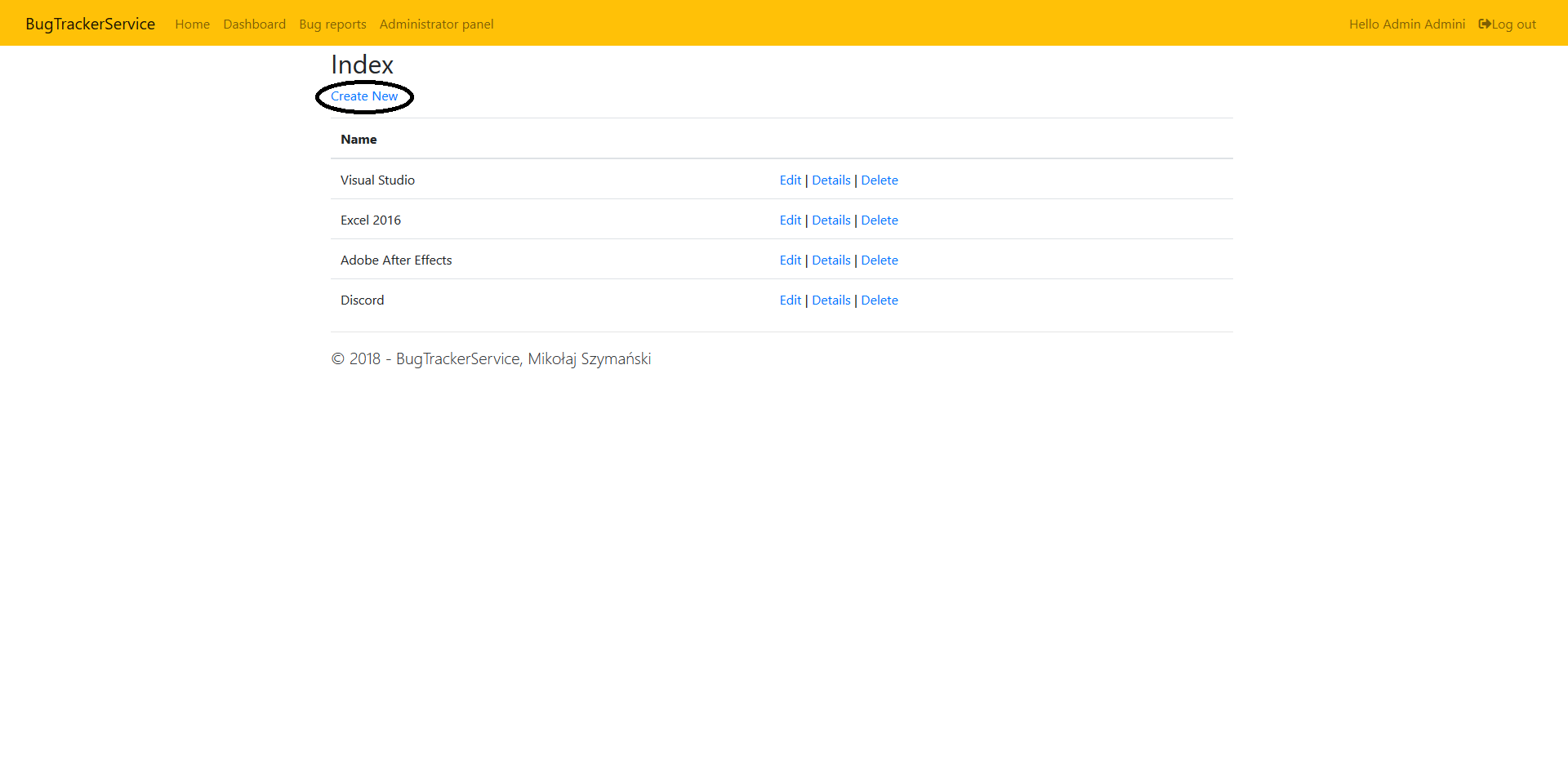


1. Zakładka **Products**
   1. Przegląd

Po kliknięciu w opcję **Products** możemy zedytować nasz produkt (zmienić jego nazwę),

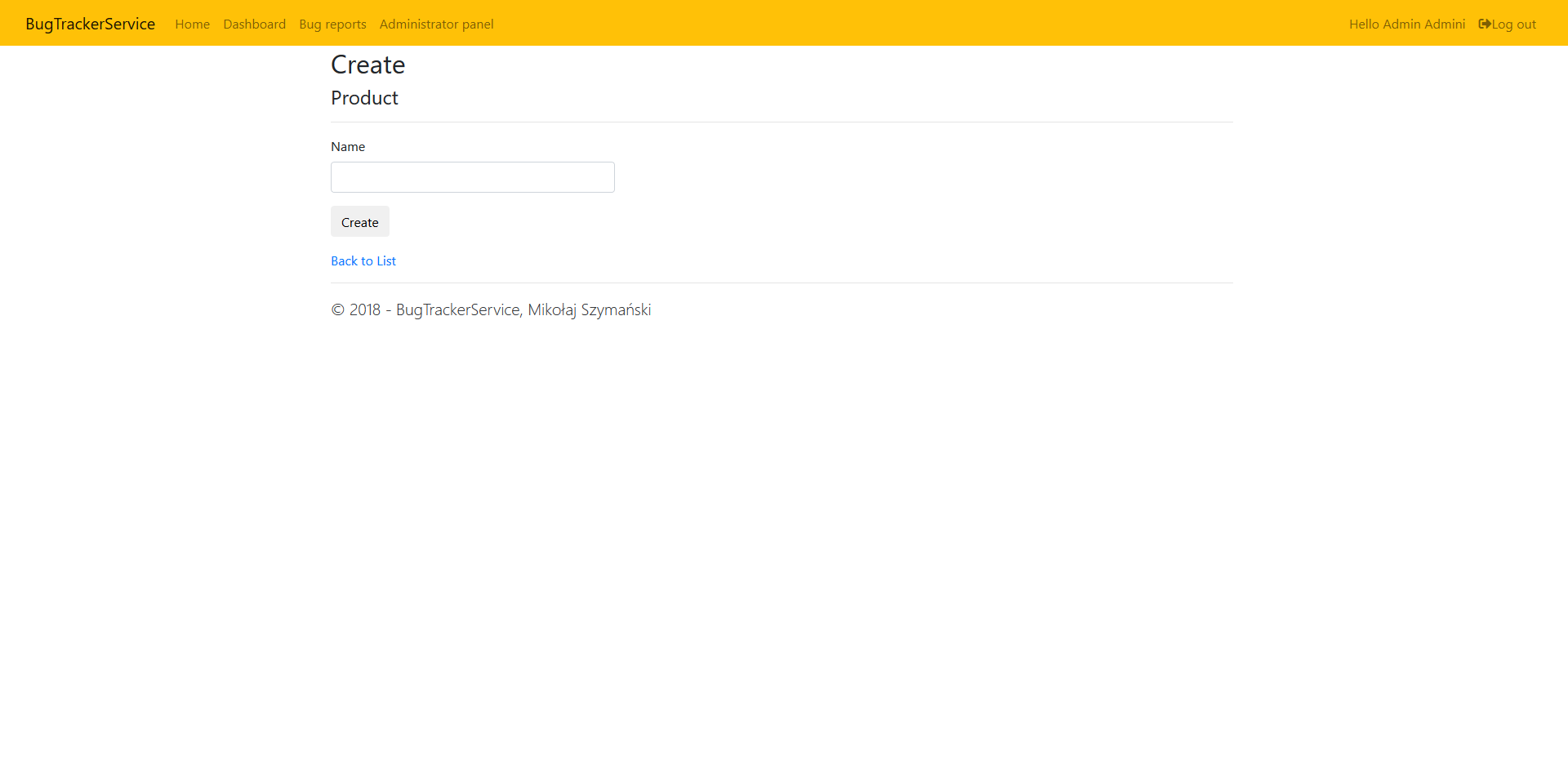


usunąć go z listy produktów oraz dodać nowy (po kliknięciu **Create New**).



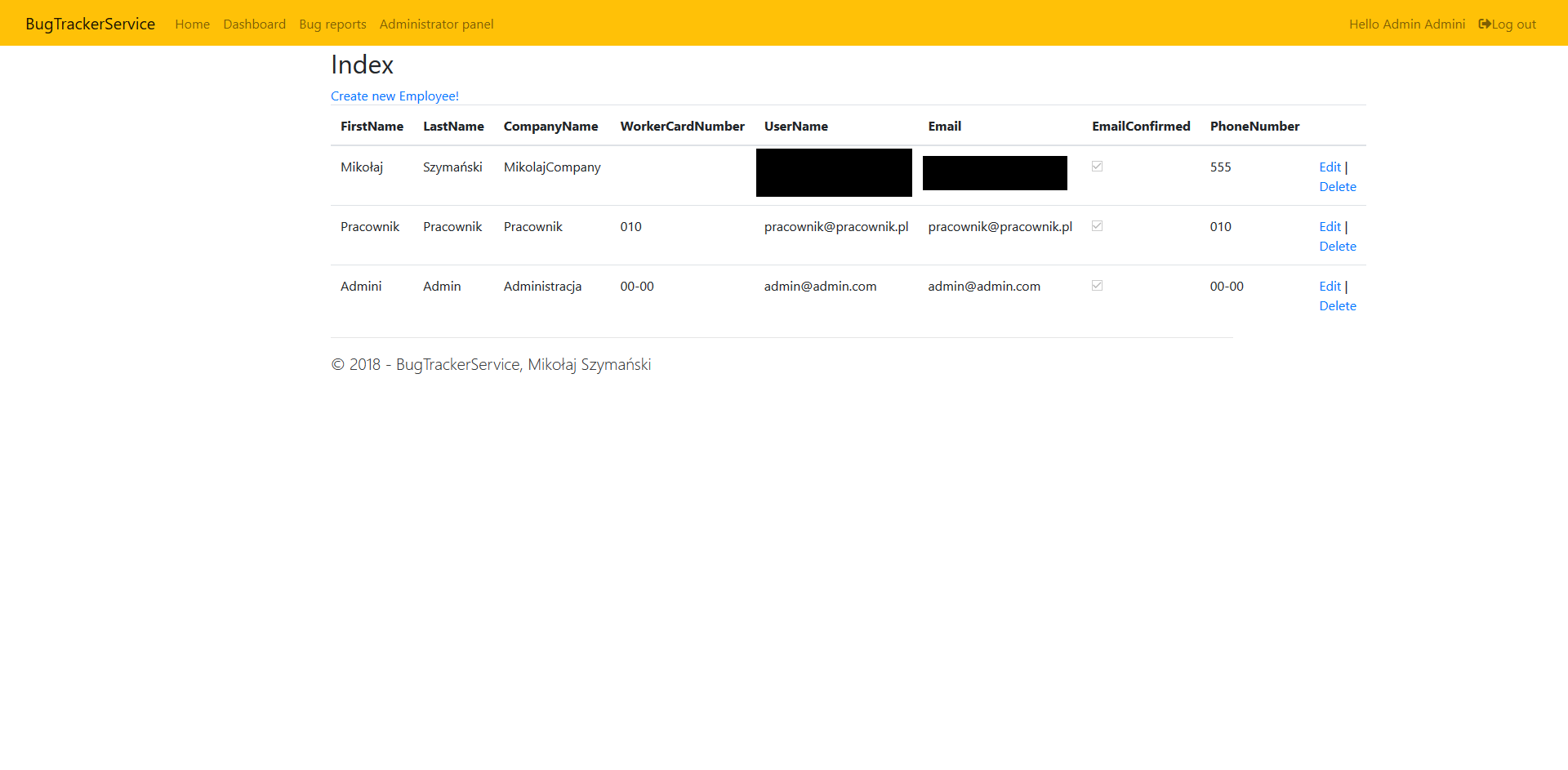
* 1. Tworzenie

Po kliknięciu w opcję **Create new** możemy dodać nowy produkt do naszej listy, wpisując jego nazwę i zatwierdzając przyciskiem **Create**.



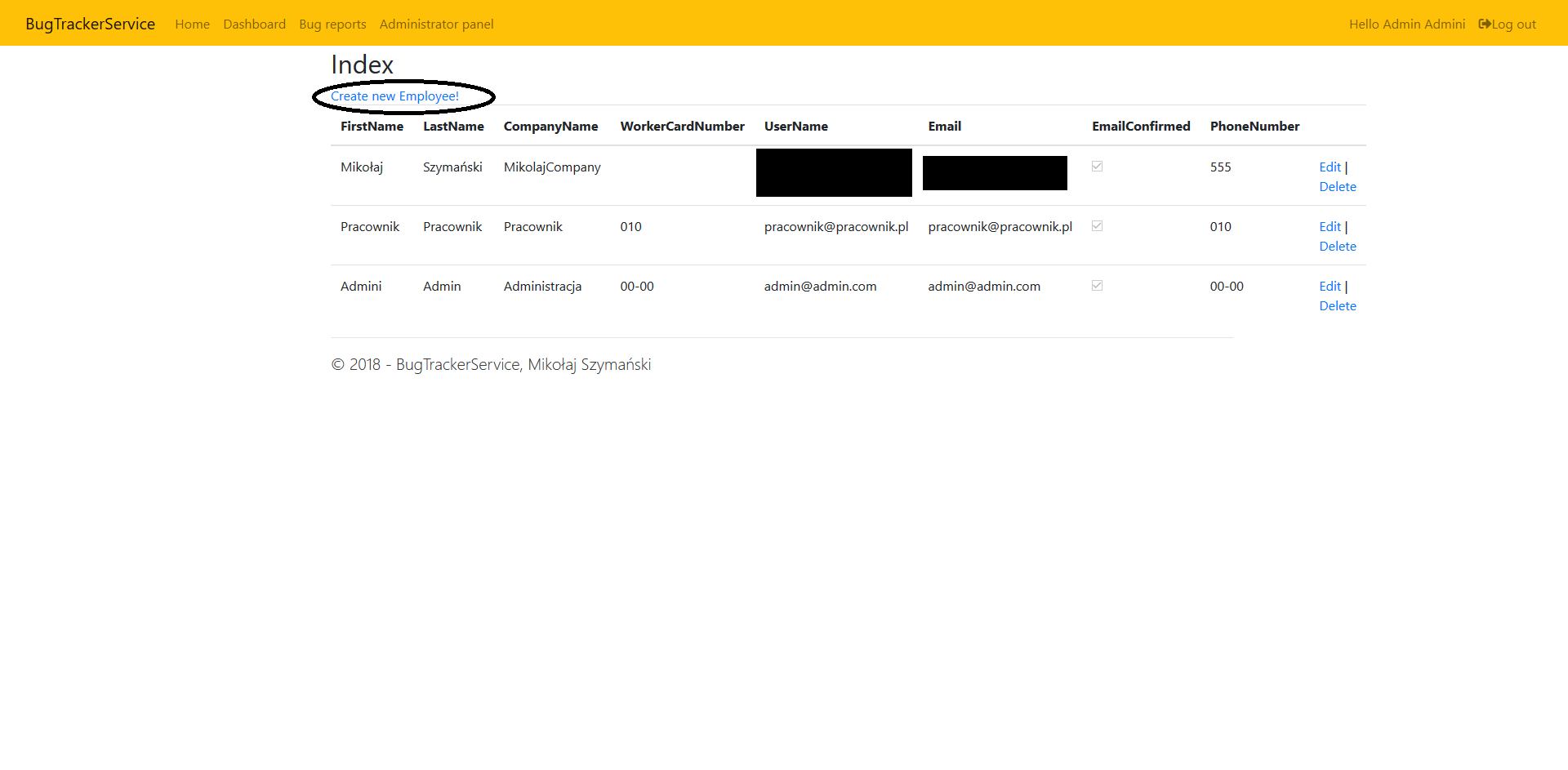
1. Zakładka **Users**
   1. Przegląd

Otwierając zakładkę **Users** ukazuje nam się lista wszystkich zarejestrowanych użytkowników, oraz ich podstawowe dane podawane podczas rejestracji.



* 1. Tworzenie

Chcąc dodać nowego pracownika należy kliknąć w **Create new Employee,** a link przekieruje nas na stronę do dobrze znanego nam formularza rejestracji nowego użytkownika (2.2.1)



# Dokumentacja programisty

## Użyte narzędzia programistyczne

### Instalacja

### Konfiguracja

## Opis działania kodu

## Klasy

## Klasa Account

#### Controller

#### Model

* + - 1. View

### Klasa Ticket

#### Controller

#### Model

* + - 1. View
    1. Klasa FileUploadHelperExtensions

## Opis działania wybranych funkcji

### Funkcja DownloadFile

## Graficzna reprezentacja schematu bazy danych

# Podsumowanie

A tu będzie podsumowanie

# Bibliografia