**Obraz zawierający tekst, clipart

Opis wygenerowany automatycznie**

Sebastian Kolański

laboratorium 2

Nr albumu 122942

**Aplikacja do kupowania biletów**

Programowanie obiektowe

2022/2023

# Spis treści

[1. Wprowadzenie 2](#_Toc125673576)

[1.1 Cel oraz ogólny opis aplikacji 2](#_Toc125673577)

[1.2 Wymagania systemowe oraz konfiguracja 2](#_Toc125673578)

[1.3 Baza danych – diagram ERD oraz opis tabel 3](#_Toc125673579)

[1.3.1 Opis tabeli „events” 3](#_Toc125673580)

[1.3.2 Opis tabeli „users” 4](#_Toc125673581)

[1.3.3 Opis tabeli „bought” 5](#_Toc125673582)

[1.4 Diagram przepływu danych 6](#_Toc125673583)

[2. Interfejs użytkownika i funkcjonalności 7](#_Toc125673584)

[2.1 Okno logowania 7](#_Toc125673585)

[2.2 Panel główny 8](#_Toc125673586)

[2.3 Panel zarządzania bazą danych 9](#_Toc125673587)

[2.4 Panel do zakupu biletów 10](#_Toc125673588)

[2.5 Panel użytkownika 11](#_Toc125673589)

[3. Dokumentacja API 12](#_Toc125673590)

[3.1 Klasy w programie 12](#_Toc125673591)

[3.2 Metody w „HelloController” 13](#_Toc125673592)

[3.3 Metody w „MainPanelController” 14](#_Toc125673593)

[4. Bezpieczeństwo, zalety i ograniczenia aplikacji 18](#_Toc125673594)

[4.1 Bezpieczeństwo 18](#_Toc125673595)

[4.2 Zalety 18](#_Toc125673596)

[4.3 Ograniczenia 18](#_Toc125673597)

1. Wprowadzenie
   1. Cel oraz ogólny opis aplikacji

Celem aplikacji o nazwie „NEW EVENT” jest możliwość zakupu biletu na wydarzenia oraz zarządzanie bazą danych, w której są zawarte informacje o nadchodzących wydarzeniach. W przyszłości, chciałbym dodać możliwość komentowania przez użytkowników wydarzeń i dyskutowanie o nich. W aplikacji postawiłem na schludny, prosty w obsłudze interfejs, który będzie zrozumiały i czytelny zarówno dla zwykłego użytkownika jak i dla osoby, która będzie nią zarządzać. Całość kodu źródłowego została zrealizowana w języku JavaFX, z interfejsem JDBC, umożliwiającym połączenie z bazą danych. Do zaprojektowania interfejsu użyłem aplikacji SceneBuilder, która znacznie przyśpieszyła ten proces i była przydatna przy ewentualnych modernizacjach. Dodatkowo zaimportowałem bibliotekę JFoenix, która przekłada się jedynie na wygląd aplikacji. Oprócz pobierania informacji z bazy danych na temat wydarzeń, dodałem funkcję umożliwiającą logowanie i rejestrowanie użytkowników z wykorzystaniem połączenia z bazą danych.

* 1. Wymagania systemowe oraz konfiguracja

Żeby uruchomić aplikacje będziemy musieli zainstalować na naszym komputerze:  
- JAVA 9

- XAMPP – przykładowa aplikacja do zarządzania bazą danych.

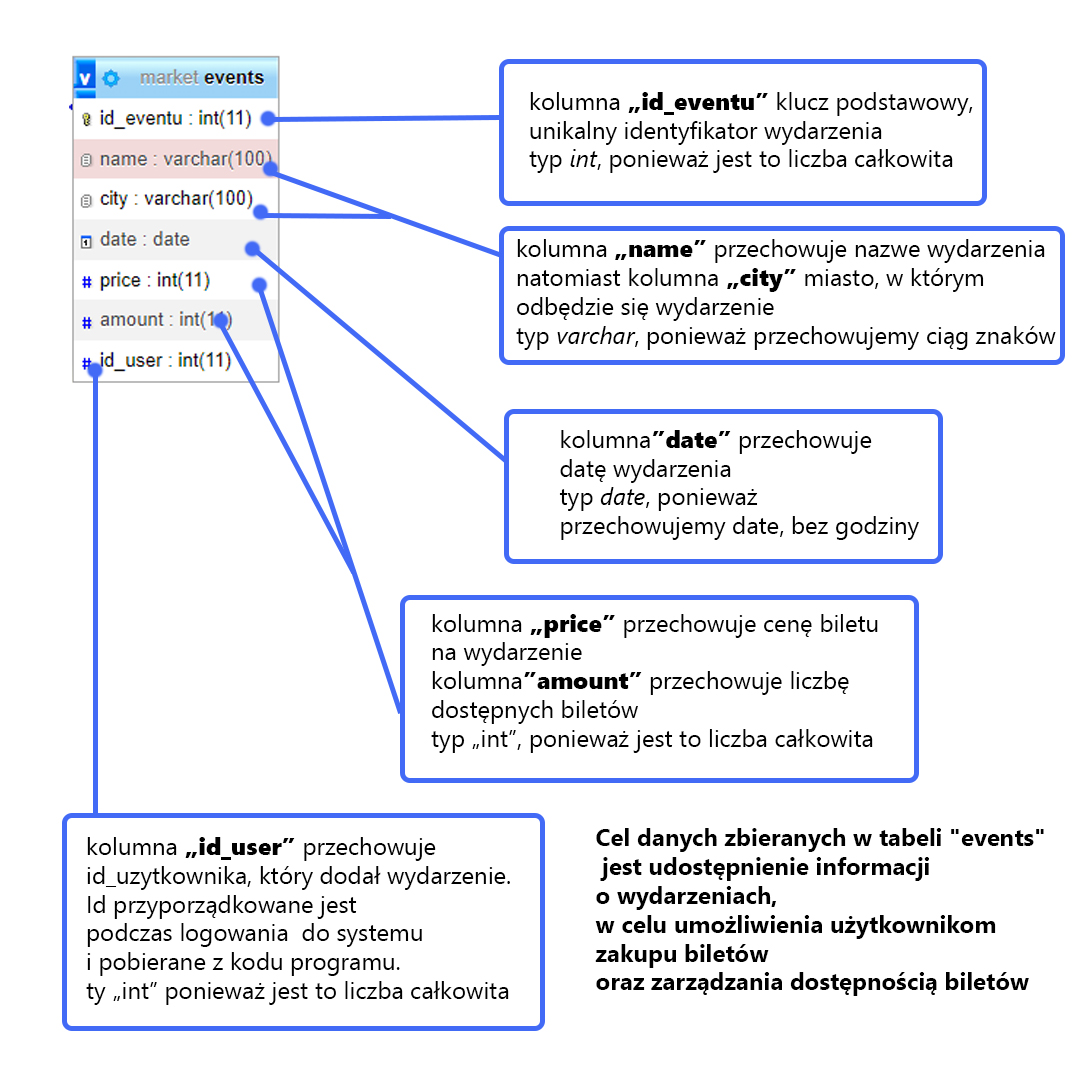
Żeby skonfigurować nasz program, będziemy musieli zaimportować dołączoną bazę danych za pomocą programu XAMPP. Używamy przeglądarki, wprowadzamy adres „localhost/phpmyadmin”. Tworzymy nową bazę danych o nazwie „market” , a w zakładce import przekazujemy dołączony plik „market.sql”.

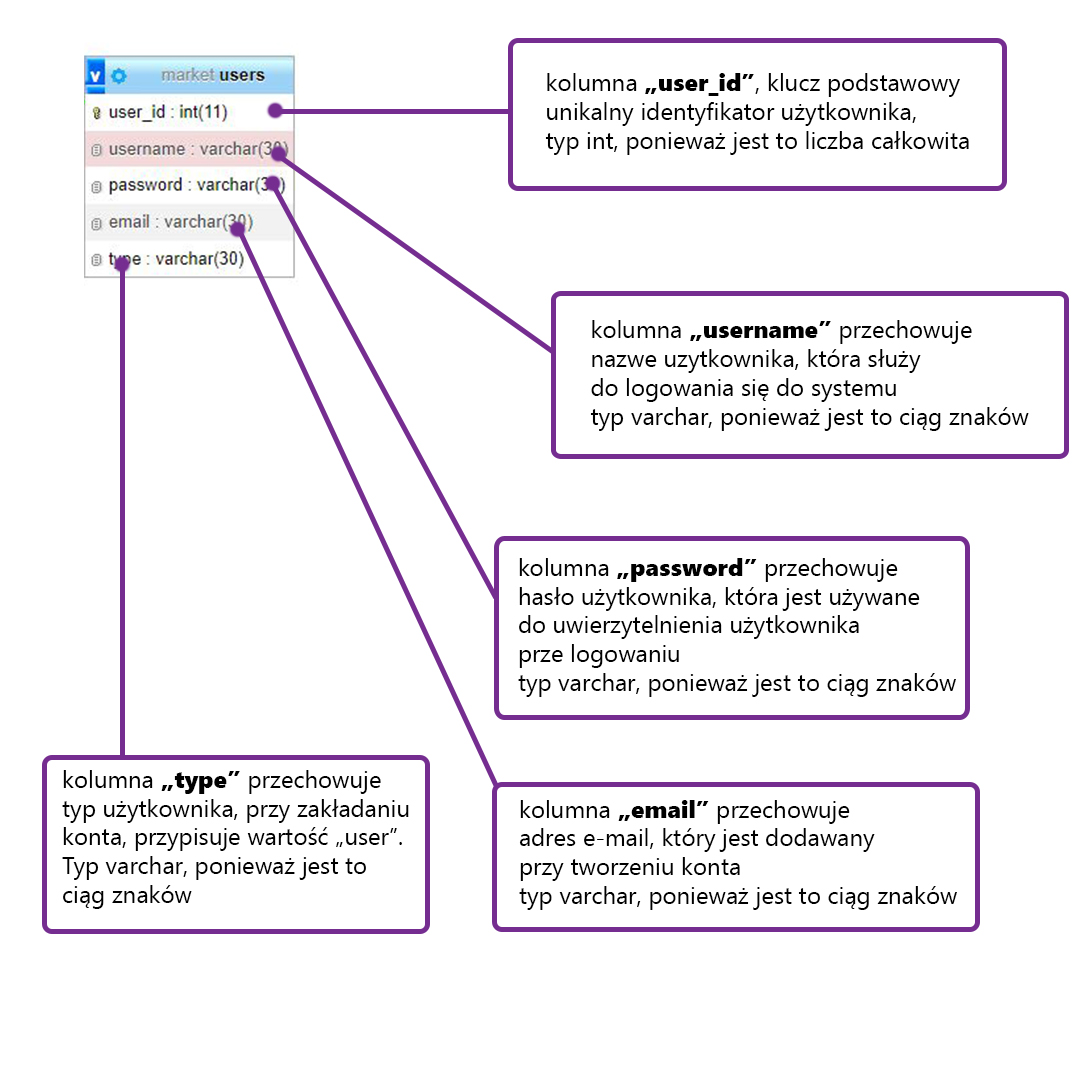
Żeby uruchomić program używamy „New Event.exe”, który znajduje się w głównym katalogu z aplikacją

* 1. Baza danych – diagram ERD oraz opis tabel

Baza danych składa się z trzech tabel połączonych relacjami. 

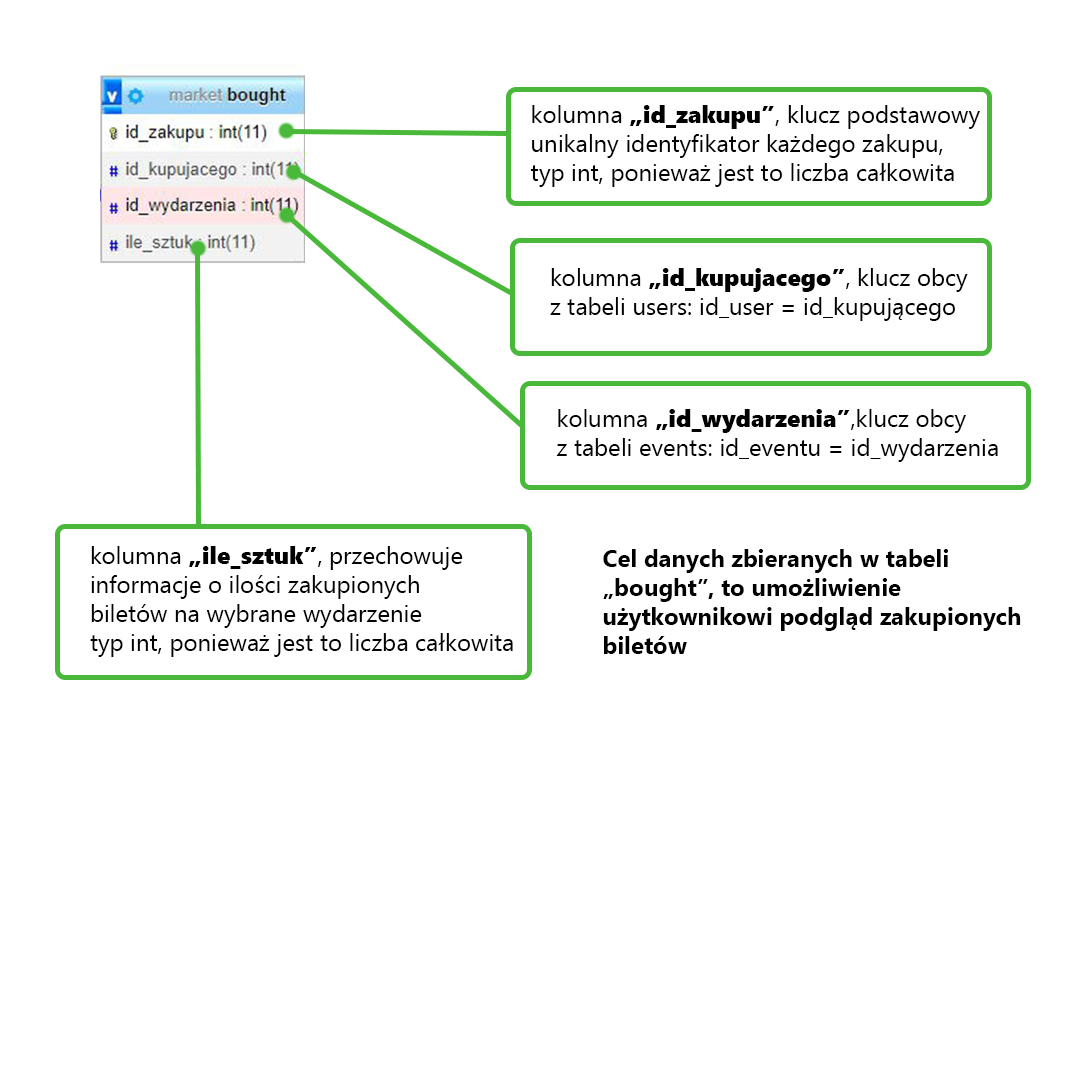
* + 1. Opis tabeli „events”



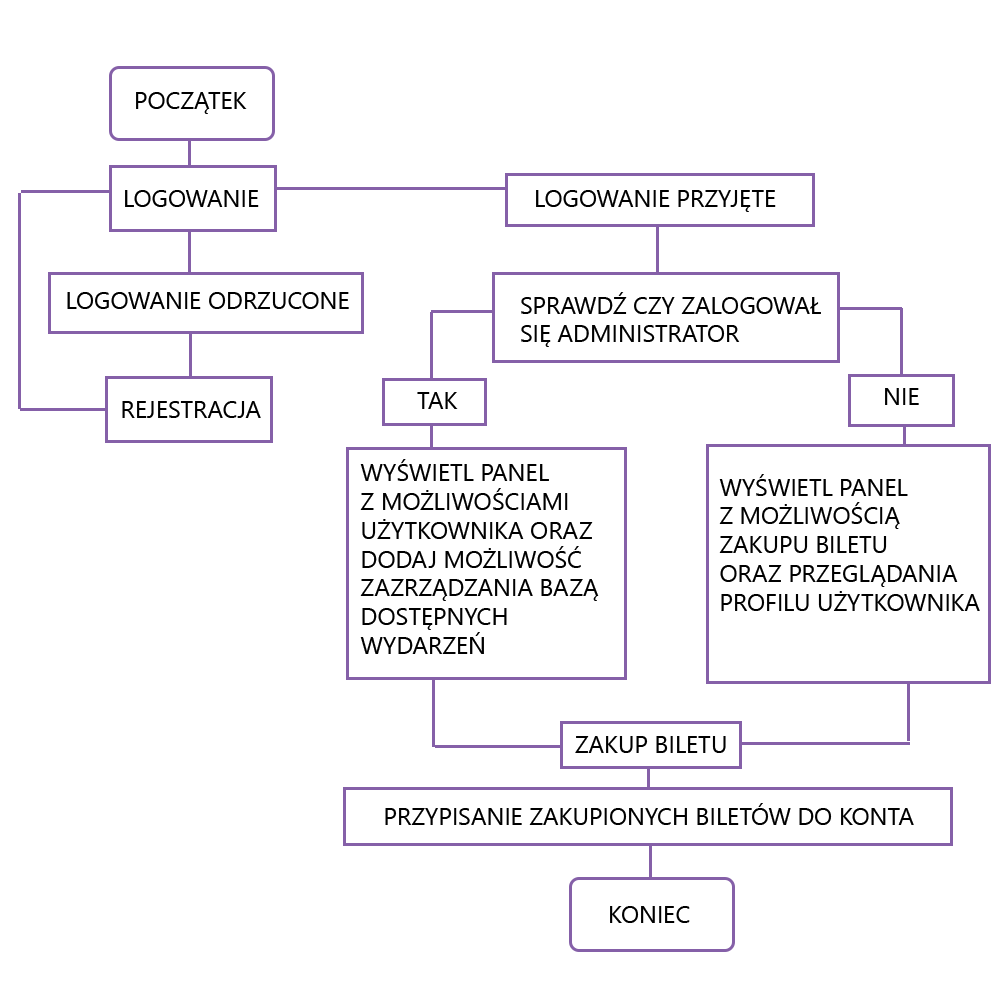
* + 1. Opis tabeli „users” 

Dane te są zbierane, aby umożliwić użytkownikom logowanie się do systemu, personalizację treści i udostępnianie odpowiednich funkcji zależnie od poziomu dostępu oraz do komunikacji z użytkownikami

* + 1. Opis tabeli „bought”



* 1. Diagram przepływu danych



1. Interfejs użytkownika i funkcjonalności
   1. Okno logowania

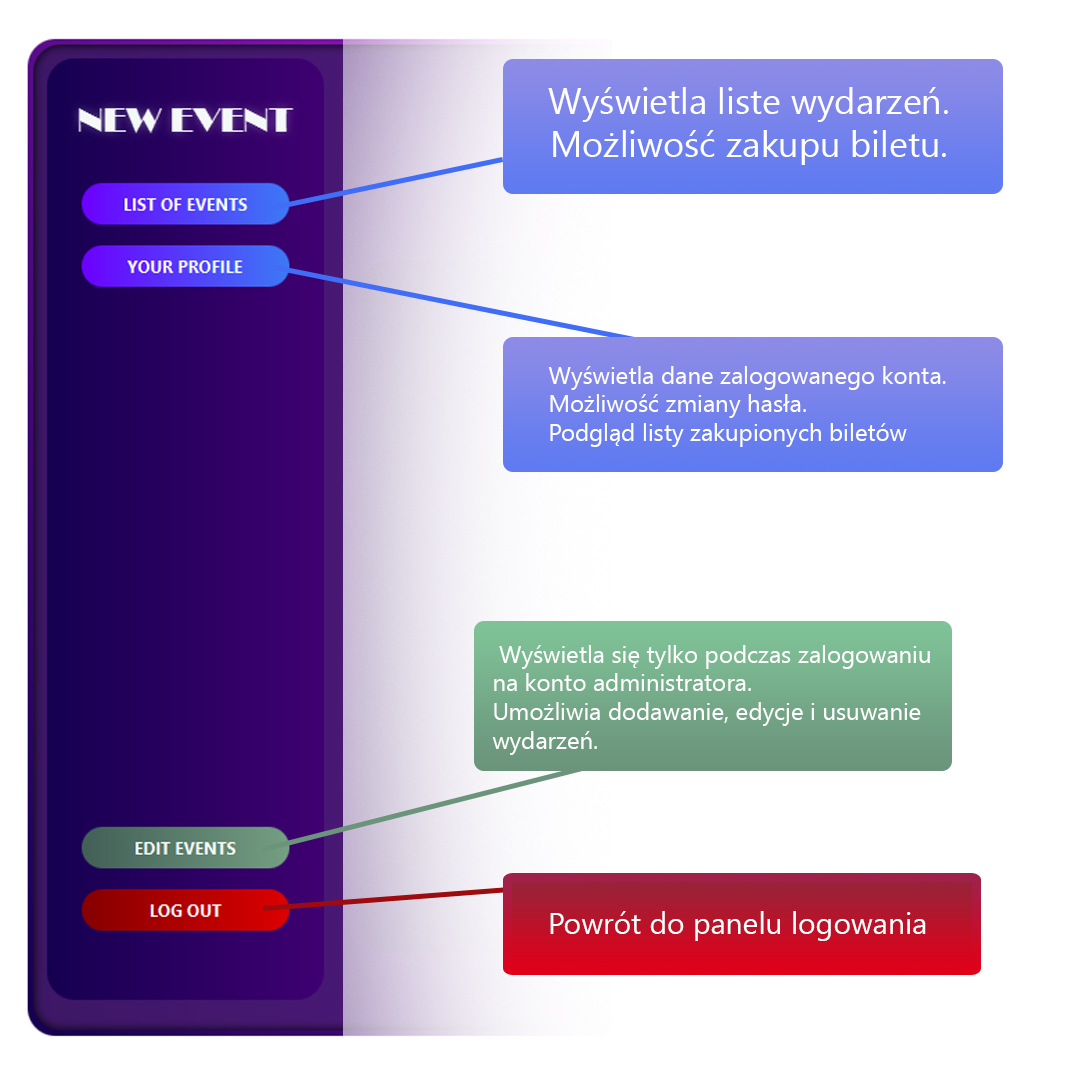


Panel logowania Panel rejestracji

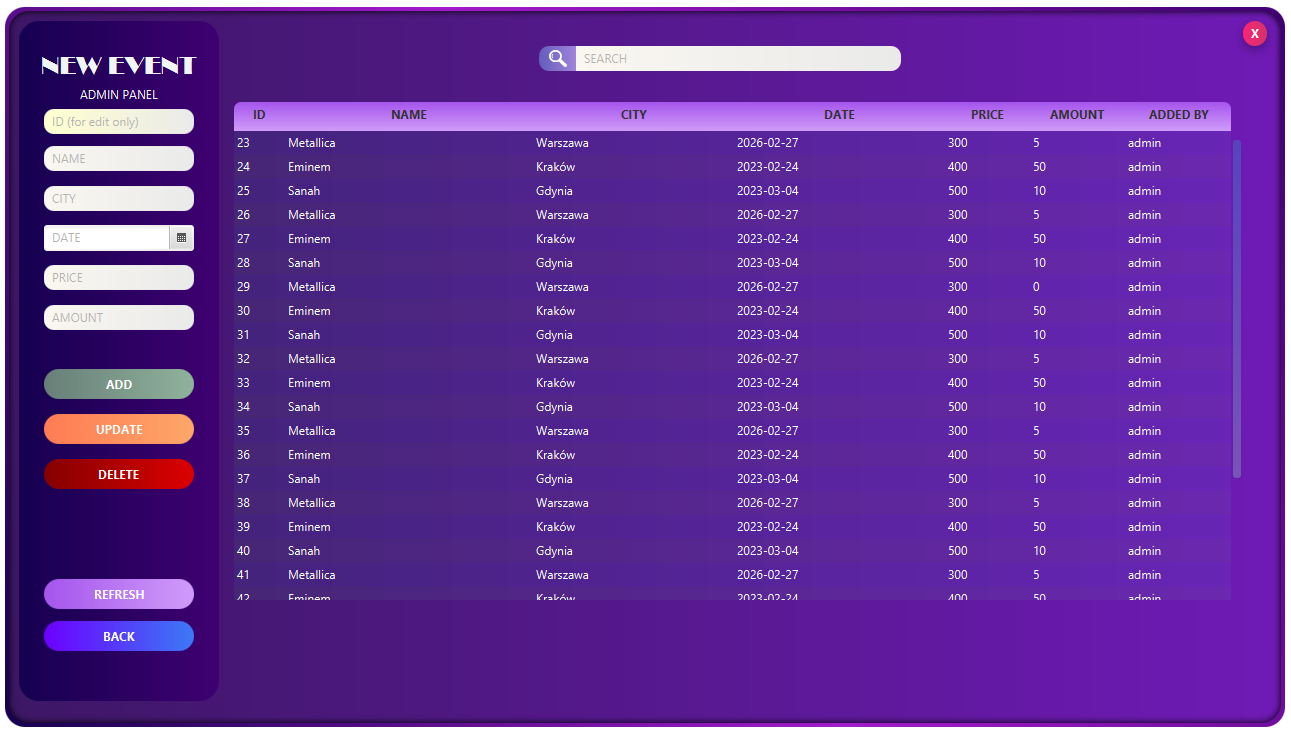
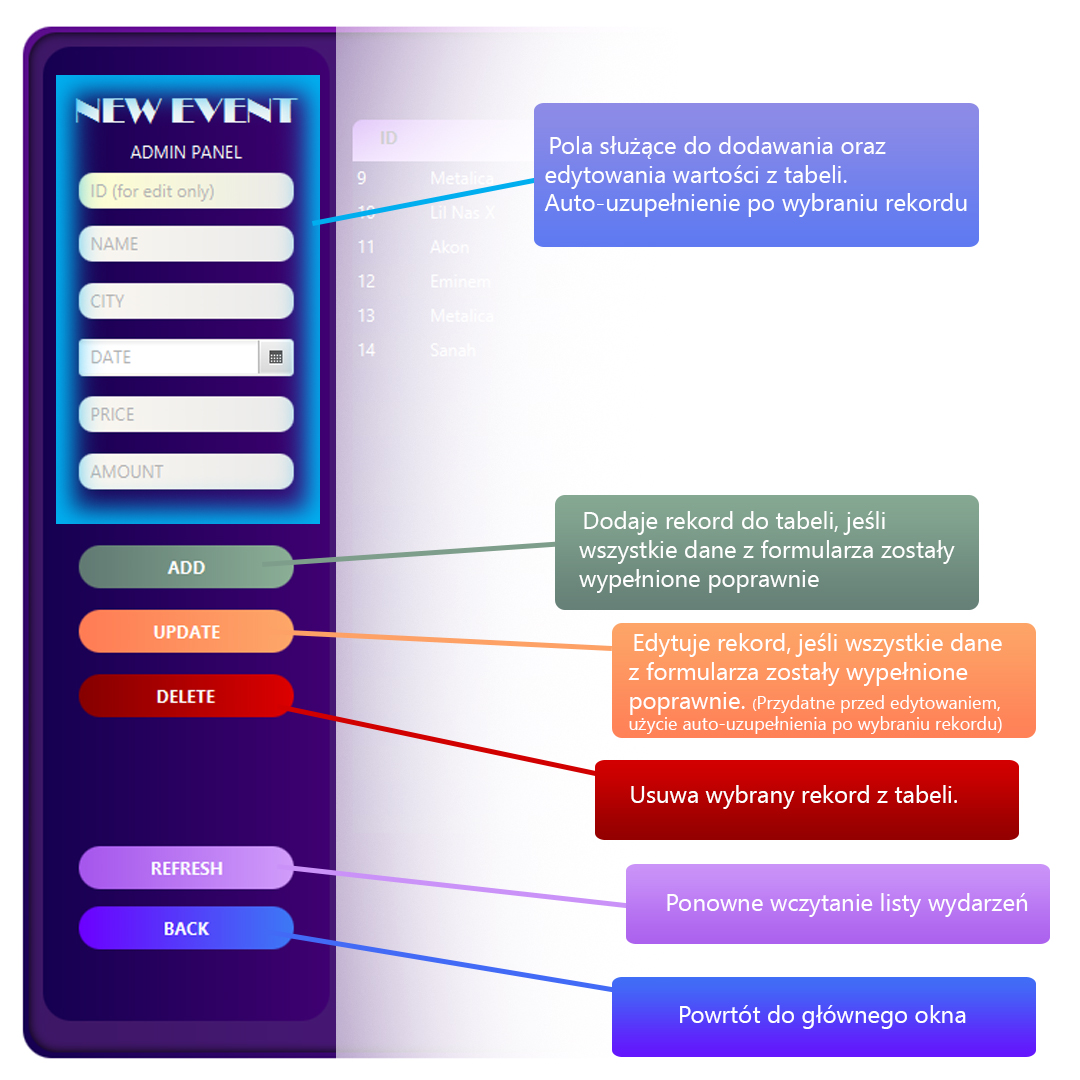
Program po uruchomieniu pokazuje ekran logowania. Żeby zalogować się musimy podać nasze dane logowania(tj. login i hasło), z wcześniej utworzonego konta. Jeśli jednak jeszcze tego nie zrobiliśmy, przechodzimy do ekranu rejestracji, w którym po podaniu nazwy użytkownika, e-mailu oraz hasła stworzymy nowe konto.

* 1. Panel główny

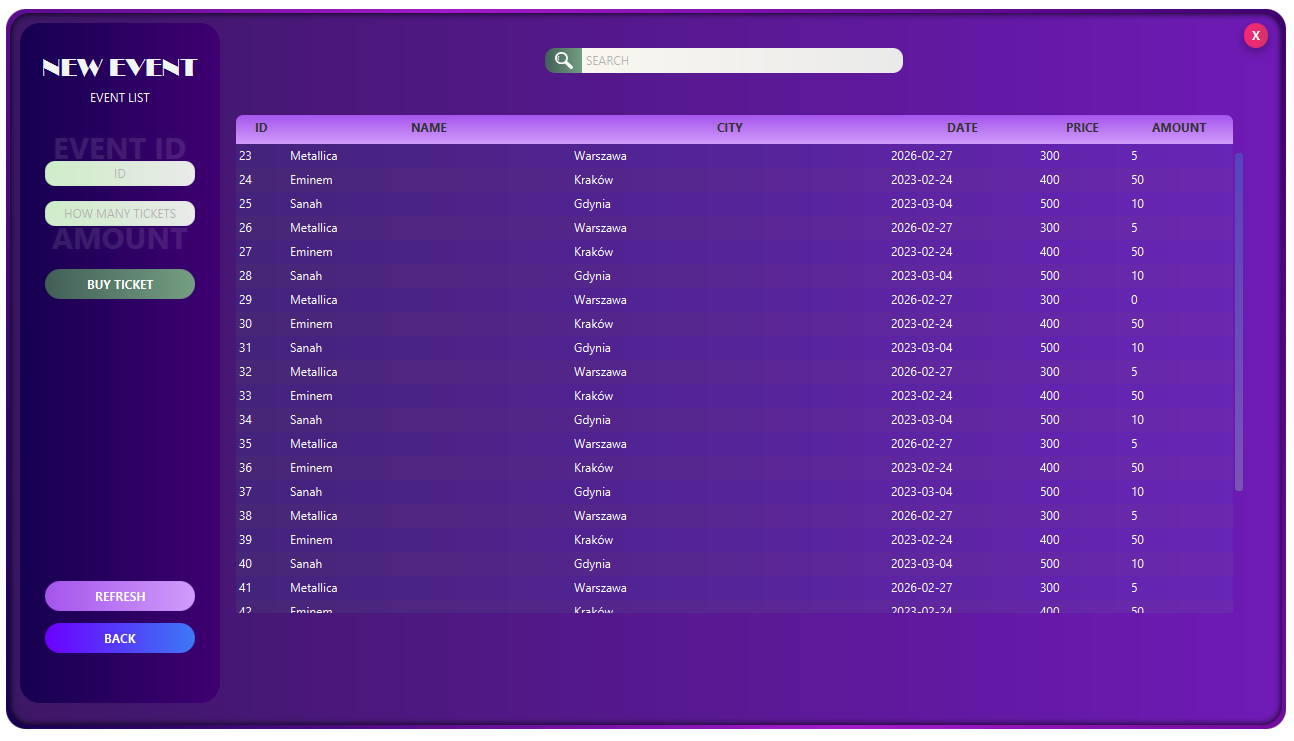
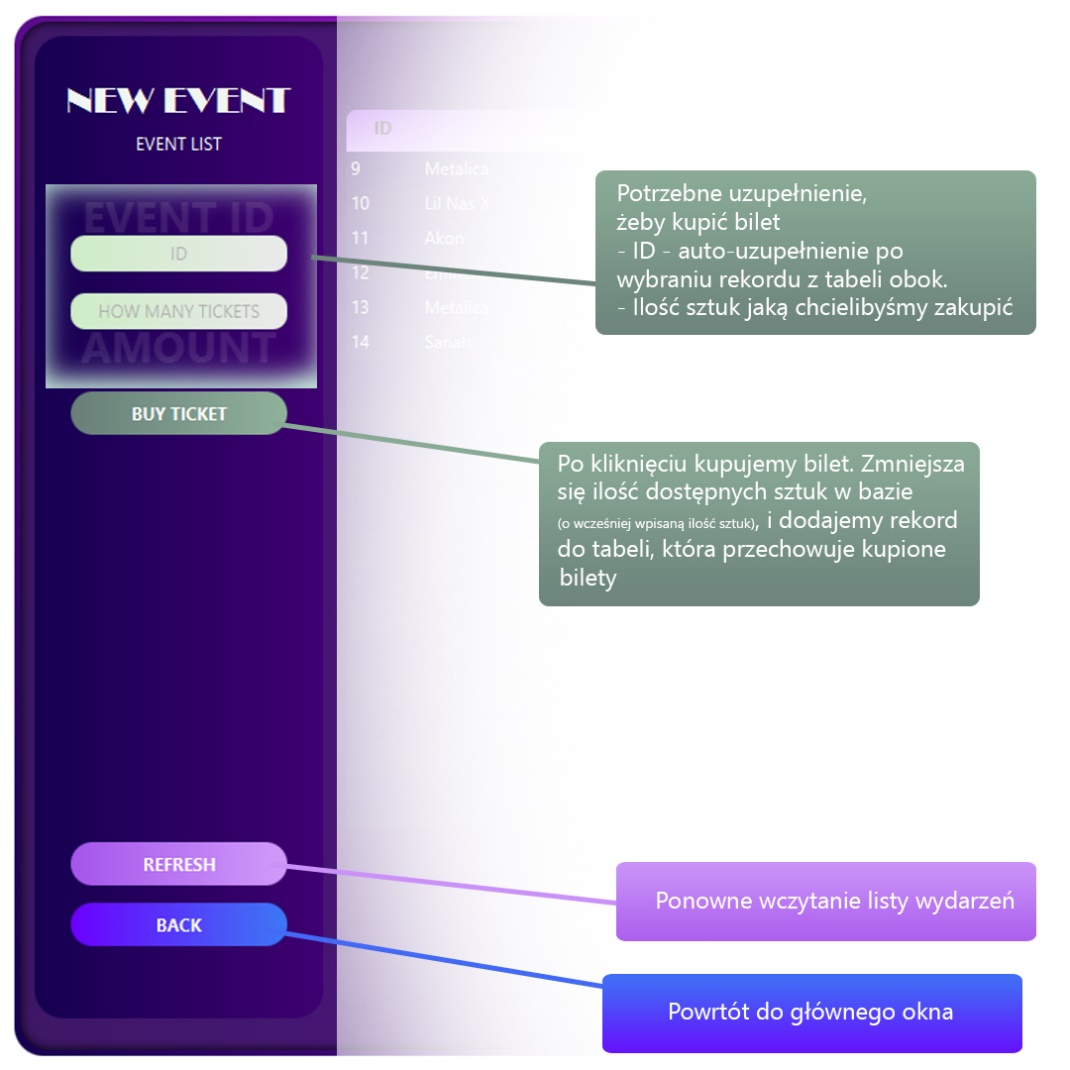
Obraz zawierający tekst

Opis wygenerowany automatyczniePo zalogowaniu, program zmienia scenę na ekran główny, w którym występują funkcje 

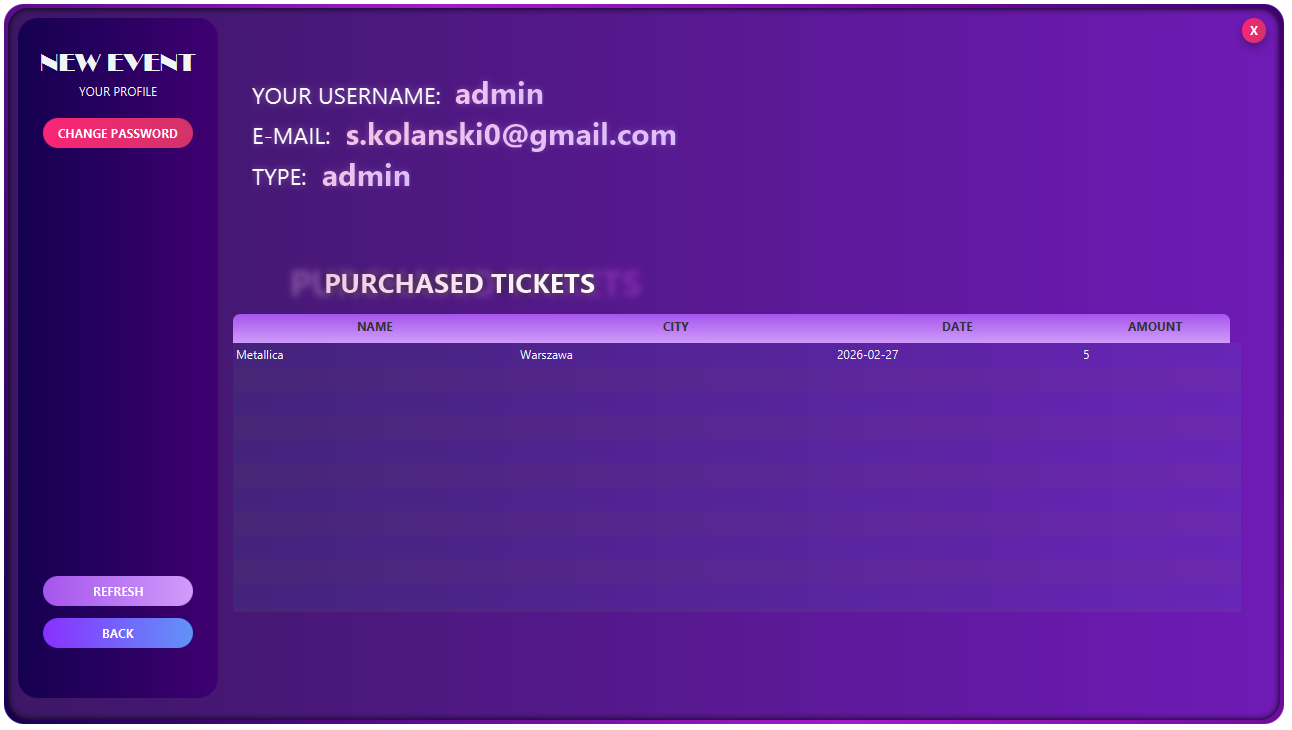
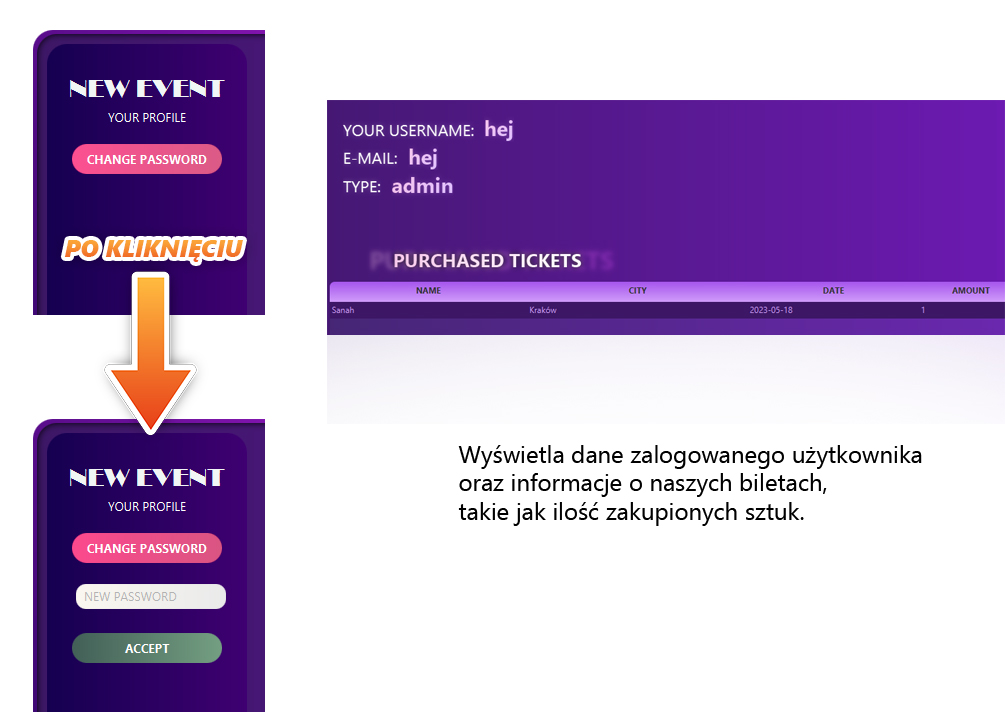
* 1. Panel zarządzania bazą danych

Po kliknięciu „EDIT EVENTS” uruchamia się nam scena, w której wyświetla się panel zarządzania oraz lista wydarzeń wpisanych do bazy danych. Oprócz wyszukiwania funkcjami które ta scena obsługuje jest

* 1. Panel do zakupu biletów

Wcześniej dodane wydarzenia, możemy przeglądać również jako zwykły użytkownik. Służy do tego przycisk „LIST OF EVENTS”, który przenosi nas do okna z listą eventów. Oprócz wyszukiwania w tym oknie dostępnymi funkcjami są:

* 1. Panel użytkownika



Po wprowadzeniu nowego hasła

oraz kliknięciu „ACCEPT”

aktualizujemy hasło naszego konta.

1. Dokumentacja API
   1. Klasy w programie

Obraz zawierający tekst

Opis wygenerowany automatycznie

* Klasa **Events** jest skonstruowana na potrzeby przekazywania informacji o wydarzeniach.
* Klasa **HelloController** jest skonstruowana na potrzeby obsługi zdarzeń okna logowania, tj. logowanie do aplikacji i rejestracja użytkownika, przez dodanie informacji o nim do bazy danych.
* Klasa **Main** obsługuje uruchomienie sceny okna logowania.
* Klasa **MainPanelController** jest skonstruowana na potrzeby obsługi sceny po poprawnym zalogowaniu przez użytkowniku.
* Klasa **mysqlconnect** jest skonstruowana na potrzeby obsługi zdarzenia połączenia programu z bazą danych.
* Klasa **User** nie ma zbyt wielu funkcji, potrzebna jedynie do zwrócenia nazwy użytkownika przy zmianie sceny.
  1. Metody w „HelloController”

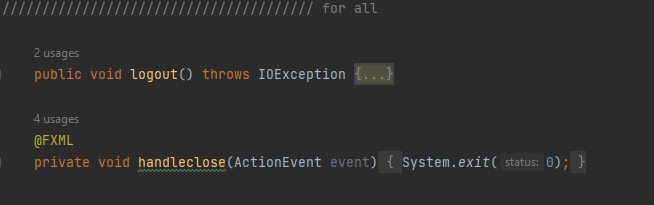
Obraz zawierający tekst

Opis wygenerowany automatycznie

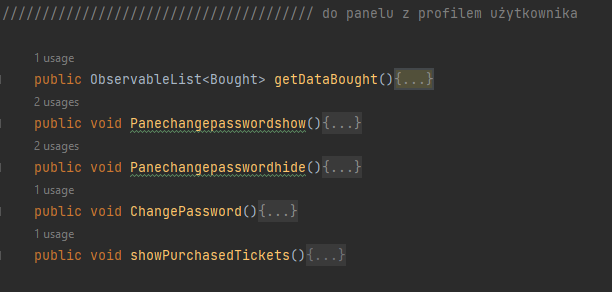
* Metoda **Loginpaneshow()** oraz **Signuppaneshow()** umożliwia przełączenie między oknem logowania a oknem rejestracji
* Metoda **getUsername()** zwraca login
* Metoda **Login()** po przekazaniu kliknięcia, w guzik, który jest na ekranie logowania, wykonuje się procedura: łączenie z bazą, sprawdzanie podanych loginów, a jeśli wszystko się zgadza, przełączanie na okno główne.
* Metoda **add\_users()** po przekazaniu kliknięcia, w guzik, który jest na ekranie rejestracji, wykonuje się procedura: łączenie z bazą, dodanie użytkownika do bazy, jeśli dane się zgadzają, po sukcesie tej operacji przechodzimy na okno logowania.
  1. Metody w „MainPanelController”



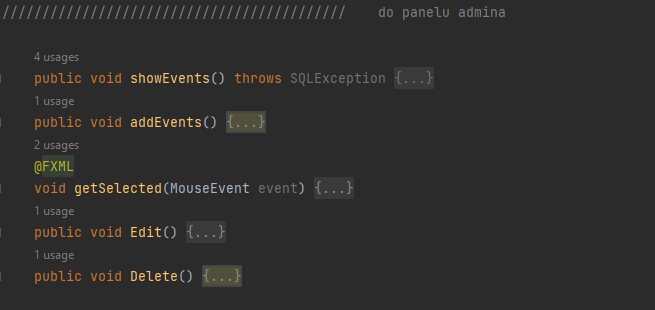
* Metody te przełączają widoki okien. Ponieważ cały ekran startowy obsługuje jeden plik FXML.



* Metoda **logout()** zmienia okno na ekran logowania.
* Metoda **handleclose()** służy do obsługi przycisku zamknięcia programu.

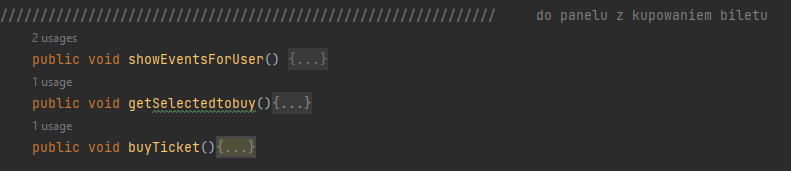


* Metoda **getDataBought()** łączy się z bazą danych, wybiera odpowiednie rekordy za pomocą zapytania SELECT oraz wstawia je do ObservableList.
* Metoda **Panechangepasswordshow()** oraz **Panechangepasswordhide()** umożliwia przełączanie się widoku (pokazane w ostatnim zdjęciu przy opisach interfejsu).
* Metoda **ChangePassword()** aktualizuje bazę danych po zmianie hasła przez użytkownika.
* Metoda **showPurchasedTickets()** uzupełnia listę w oknie profilu użytkownika.



Metody te służą do zarządzania panelem administratora.

* **showEvents()** pokazuje listę wydarzeń.
* **addEvents()** dodaje wydarzenia po uzupełnieniu formularza i kliknięciu przycisku add,
* **getSelected()** służy do auto-uzupełniania po wybraniu rekordu z listy,
* **Edit()** obsługuje aktualizacje bazy po kliknięciu przycisku update,
* **Delete()**, usuwa zaznaczony rekord z bazy danych.



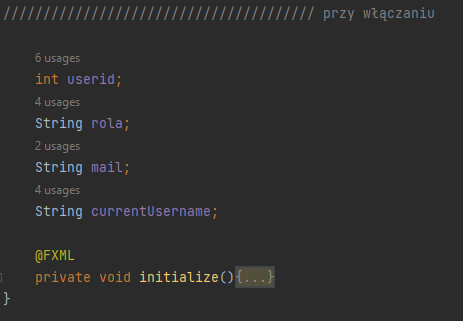
Metody te służą do pokazywania listy dostępnych wydarzeń i zakupu biletu.

* **showEventsForUser(),** wyświetla listę z wydarzeniami
* **getSelectedtobuy(),** przekazuje ID w pole formularza, po wybraniu rekordu z listy
* **buyTicket(),** aktualizuje bazę danych po zakupie biletu.

Obraz zawierający tekst

Opis wygenerowany automatycznie

* **searchEvents()** – wyszukuje wydarzeń po nazwie lub mieście
* **validationAdd()** oraz **validationUpdate()** korzysta z **validationAdminPane()** który ustala poprawność wprowadzonych danych podczas dodawania/edycji wydarzeń



Po włączeniu okna wykonuje się **initialize**, które wyświetla nazwe użytkownika na ekranie głównym, oraz uzupełnia zmienne, które znajdują się powyżej metody.

1. Bezpieczeństwo, zalety i ograniczenia aplikacji
   1. Bezpieczeństwo

Dla bezpieczeństwa aplikacji, został wykonany system uwierzytelniania użytkowników za pomocą hasła oraz wprowadzenie funkcji użytkownika i administratora, żeby zwykły użytkownik nie mógł ingerować w zarządzanie bazą danych.

* 1. Zalety

Zaletą aplikacji na pewno jest to, że nie powinna sprawiać kłopotu w obsłudze dla zwykłego użytkownika oraz w prosty sposób możemy zarządzać nią jako administrator.

* 1. Ograniczenia

Ograniczeniami aplikacji z pewnością jest brak znacznej rozbudowy, jednakże w przyszłości chciałbym ją rozbudować dodając wszelakie funkcje.