

UNIwersYTET RZESZOWSKI
WYDZIAŁ NAUK ŚCISŁYCH I TECHNICZNYCH
INSTYTUT INFORMATYKI



Jakub Pelic
134957
Informatyka

Dokumentacja Aplikacji Do Zarządzania Drużyną Piłkarską

Praca projektowa
Praca wykonana pod kierunkiem
mgr inż. Ewa Żesławska

Rzeszów 2025r.

Spis treści

1. Opis założeń projektu	3
1.1. Wymagania funkcjonalne oraz нефункционалне.....	3
2. Opis struktury projektu	5
2.1. Opis metod kontrolerów	5
3. Harmonogram realizacji projektu	8
4. Prezentacja warstwy użytkowej projektu.....	9
5. Podsumowanie.....	14
Spis rysunków	15

1. Opis założeń projektu

Celem projektu jest stworzenie aplikacji desktopowej umożliwiającej użytkownikowi zarządzanie drużynami piłkarskimi i ich zawodnikami. Aplikacja została napisana w języku Java z wykorzystaniem biblioteki Swing do tworzenia interfejsu graficznego oraz JDBC do komunikacji z bazą danych MySQL. Program umożliwia tworzenie kont użytkowników, logowanie się, tworzenie i edycję drużyn oraz zarządzanie listą zawodników. Głównym problemem, który rozwiązuje aplikacja, jest brak prostych i intuicyjnych narzędzi desktopowych dla amatorskich trenerów lub pasjonatów piłki nożnej, którzy chcą w prosty sposób śledzić skład i strukturę drużyn. Potrzeba ta została zaobserwowana na poziomie amatorskich lig, szkółek piłkarskich oraz entuzjastów lokalnych rozgrywek, którzy dotychczas korzystają z arkuszy kalkulacyjnych lub ręcznych zapisków, co nie jest efektywne ani wygodne.

W celu rozwiązania tego problemu, projekt zakłada stworzenie kompletnego systemu desktopowego, który po zalogowaniu się użytkownika umożliwia zarządzanie jego drużynami. Każdy użytkownik może dodawać, edytować i usuwać drużyny, a także modyfikować ich skład osobowy poprzez dodawanie lub usuwanie zawodników. Proces rozpoczyna się od formularza logowania, skąd można przejść do rejestracji nowego konta. Po pomyślnym zalogowaniu użytkownik widzi listę swoich drużyn. Funkcje dodawania i edycji otwierają osobne okna formularzy. Całość została zaprojektowana w sposób obiektowy — każda klasa GUI dziedziczy po klasie nadrzędnej odpowiedzialnej za wspólne ustawienia graficzne (takie jak centrowanie okna). Logika interakcji została rozdzielona pomiędzy kontrolery przypisane do każdego widoku.

Aplikacja wymaga aktywnego połączenia z bazą danych MySQL, co oznacza, że użytkownik musi posiadać zainstalowany serwer bazy danych oraz odpowiednio skonfigurowany dostęp (login i hasło). Rezultatem prac projektowych jest w pełni funkcjonalna aplikacja komputerowa, którą można rozszerzać o nowe funkcjonalności w przyszłości (np. statystyki zawodników, harmonogramy meczów).

1.1. Wymagania funkcjonalne oraz нефункционалне

- Możliwość rejestracji nowego użytkownika z podaniem wszystkich niezbędnych danych.
- Możliwość logowania do aplikacji po uprzednim utworzeniu konta.
- Przechowywanie i weryfikacja danych logowania w bazie danych MySQL.
- Lista drużyn przypisana do zalogowanego użytkownika.
- Możliwość dodania nowej drużyny poprzez wypełnienie formularza.
- Możliwość edycji nazwy wybranej drużyny.
- Możliwość usunięcia wybranej drużyny.
- Wyświetlanie listy zawodników przypisanych do konkretnej drużyny.
- Możliwość dodania zawodnika do wybranej drużyny (z podaniem imienia i nazwiska).
- Możliwość usunięcia zawodnika z drużyny.
- Oddzielenie warstwy GUI od logiki poprzez zastosowanie kontrolerów.

oraz wymagania нефункционалне:

- Aplikacja powinna być dostępna na systemie operacyjnym Windows z zainstalowanym środowiskiem Java.
 - Interfejs użytkownika powinien być intuicyjny i spójny wizualnie (wspólna klasa Frame dla okien).
 - Wszystkie działania powinny być możliwe do wykonania w czasie krótszym niż 3 sekundy.
 - System musi być odporny na błędy związane z brakiem połączenia z bazą danych.
 - Aplikacja powinna przechowywać dane w lokalnej bazie danych MySQL.
 - Zmiany wprowadzane w interfejsie powinny być natychmiastowo odzwierciedlane w bazie danych.
 - Aplikacja powinna umożliwiać łatwe aktualizacje i konserwację kodu dzięki modularnej strukturze klas.
 - W przypadku błędnych danych logowania aplikacja powinna poinformować użytkownika o niepowodzeniu.
 - Okna aplikacji powinny być automatycznie centrowane na ekranie użytkownika.
-

2. Opis struktury projektu

Struktura projektu została zaprojektowana w sposób modularny z wyraźnym podziałem na warstwę logiki (kontrolery), interfejs użytkownika (GUI) oraz logikę biznesową (modele danych). Aplikacja została zrealizowana w języku Java z wykorzystaniem środowiska IntelliJ IDEA oraz bibliotek Swing (do tworzenia interfejsu graficznego) i JDBC (do obsługi połączenia z bazą danych MySQL).

Struktura klas

Projekt składa się z następujących typów plików:

- **GUI (.java + .form)** – klasy odpowiedzialne za interfejs graficzny: `AplikacjaGUI`, `DodajPilkarzaGUI`, `DodawanieDruzyznyGUI`, `EdytyujDruzyneGUI`, `LoginGUI`, `RejestracjaGUI`, każda posiada odpowiadający plik `.form` (projekt wizualny).
- **Kontrolery** – klasy zarządzające logiką aplikacji: `AplikacjaController`, `DodajPilkarzaController`, `DodawanieDruzyznyController`, `EdytyujDruzyneController`, `LoginController`, `RejestracjaController`.
- **Modele danych** – klasy reprezentujące dane aplikacji: `Druzyzna`, `Pilkarz`, `Uzytkownik`.
- **Pozostałe klasy wspomagające** – `Main` (punkt wejścia aplikacji), `Frame` (bazowa klasa okien aplikacji), `Komunikat` (wyświetlanie informacji użytkownikowi), `BazaDanych` (zarządzanie połączeniem z MySQL).

Hierarchia klas i logika aplikacji

- Wszystkie klasy graficzne dziedziczą po klasie `Frame`, która ustawia wspólne parametry okien, takie jak centrowanie na ekranie.
- Klasa `Main` uruchamia aplikację i wywołuje ekran logowania.
- Klasa `LoginGUI` umożliwia logowanie użytkownika, a `LoginController` obsługuje logikę uwierzytelniania (porównanie danych z bazą).
- Klasa `RejestracjaGUI` umożliwia rejestrację nowego użytkownika, a `RejestracjaController` zapisuje dane do bazy.
- Po zalogowaniu uruchamiane jest główne okno `AplikacjaGUI`, zarządzane przez `AplikacjaController`, wyświetlające drużyny i umożliwiające ich modyfikację.
- Dodawanie/edycja drużyn realizowana jest w osobnych oknach: `DodawanieDruzyznyGUI`, `EdytyujDruzyneGUI`.
- Analogicznie, zawodników obsługują: `DodajPilkarzaGUI`, `DodajPilkarzaController`.

2.1. Opis metod kontrolerów

AplikacjaController

- `OdswiezListeDruzyn()` – Pobiera z bazy danych listę drużyn przypisanych do zalogowanego użytkownika i wyświetla je w interfejsie.

- `DodajDruzyneObsluga()` – Otwiera nowe okno do dodania drużyny i zamyka bieżące.
- `UsunDruzyne()` – Usuwa zaznaczoną drużynę z bazy danych i z listy GUI.
- `EdytujDruzyne()` – Przechodzi do okna edycji wybranej drużyny.

RejestracjaController

- `Zarejestuj()` – Obsługuje proces rejestracji nowego użytkownika: sprawdza poprawność danych i zapisuje je do bazy.
- `CzyIstniejeTakiUzytkownik(String login)` – Sprawdza, czy użytkownik o podanym loginie już istnieje w bazie.
- `Wroc()` – Powrót do okna logowania.

LoginController

- `Zaloguj()` – Autoryzuje dane logowania użytkownika na podstawie zapytania SQL. W przypadku powodzenia przechodzi do głównego okna aplikacji.

DodawanieDruzynyController

- `DodajDruzyne()` – Dodaje nową drużynę do bazy, jeśli taka jeszcze nie istnieje.
- `CzyIstniejeTakaDruzyina(String nazwa)` – Sprawdza, czy drużyna o danej nazwie znajduje się już w bazie danych.
- `Wroc()` – Powrót do głównego okna aplikacji.

EdytujDruzyneController

- `ZmienNazwe()` – Zmienia nazwę wybranej drużyny w bazie danych.
- `DodajPilkarza()` – Otwiera okno dodawania nowego piłkarza do danej drużyny.
- `UsunPilkarza()` – Usuwa zaznaczonego piłkarza z bazy i z listy GUI.
- `Wroc()` – Powrót do głównego okna aplikacji.
- `OdswiezListePilkarzy()` – Wczytuje wszystkich piłkarzy przypisanych do danej drużyny i wyświetla ich w interfejsie.

DodajPilkarzaController

- `Dodaj()` – Dodaje nowego piłkarza do bazy danych i przypisuje go do konkretnej drużyny.
- `Wroc()` – Powrót do głównego okna aplikacji.

Wykorzystane technologie

- **Java 24** – język programowania.
 - **Swing** – biblioteka do budowy GUI.
 - **JDBC** – interfejs do obsługi bazy danych.
 - **MySQL 10.4.32-MariaDB** – relacyjna baza danych przechowująca dane użytkowników, drużyn i zawodników.
-

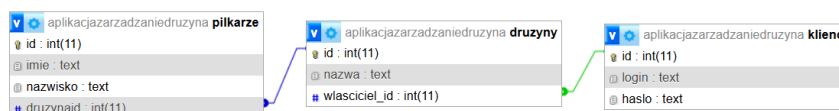
- **IntelliJ IDEA 2025.1** – środowisko programistyczne wykorzystywane do budowy projektu.

Zarządzanie danymi i baza danych

Baza danych znajduje się w pliku readme.md w repozytorium projektu zostało umieszczonym na platformie **GitHub** pod adresem:

https://github.com/KubaP43/PO_Java_UR

Poniżej zamieszczony zrzut ekranu przedstawia relację oraz strukturę bazy danych, powiązania oraz klucze główne.



Rys. 2.1. Diagram ERD przedstawiający relacje w bazie danych

Dane przechowywane są w lokalnej bazie danych MySQL. Struktura bazy obejmuje trzy tabele:

Tabela druzyzny

Tabela przechowuje informacje o drużynach, w tym ich nazwy oraz identyfikatora właściciela (odwołanie do tabeli klienci).

Tabela klienci

Tabela zawiera dane klientów systemu, takie jak login oraz hasło.

Tabela pilkarze

Tabela zawiera dane piłkarzy, w tym imię, nazwisko oraz identyfikator drużyny, do której należą.

Połączenie realizowane jest przez klasę BazaDanych przy użyciu sterownika JDBC.

Minimalne wymagania sprzętowe

- System operacyjny: Windows 10 lub nowszy.
- Procesor: minimum Intel Core i3 lub odpowiednik.
- RAM: minimum 4 GB.
- Dysk: co najmniej 200 MB wolnego miejsca.
- Zainstalowany i uruchomiony serwer MySQL.

Dodatkowe narzędzia

- IntelliJ IDEA (lub inny edytor wspierający .form Swing Designer).
- phpMyAdmin – do zarządzania bazą danych.
- JDBC Driver for MySQL (np. mysql-connector-java).

3. Harmonogram realizacji projektu

Poniżej przedstawiono harmonogram realizacji projektu w postaci diagramu Gantta. Diagram ten obrazuje podział prac nad projektem w czasie, z uwzględnieniem głównych etapów, takich jak projektowanie bazy danych i interfejsu, implementacja funkcjonalności, testowanie oraz przygotowanie dokumentacji.

Funkcjonalność	Tydzień 1	Tydzień 2	Tydzień 3	Tydzień 4	Tydzień 5	Tydzień 6
1. Projektowanie bazy danych	5 dni					
2. Projektowanie interfejsu GUI	3 dni	2 dni				
3. Implementacja logowania		3 dni				
4. Implementacja rejestracji		2 dni				
5. Implementacja listy drużyn			4 dni			
6. Dodawanie/usuwanie drużyn			3 dni			
7. Edycja drużyn i zarządzanie				4 dni		
8. Zarządzanie piłkarzami				3 dni		
9. Testowanie aplikacji					3 dni	
10. Poprawa błędów i optymalizacja					2 dni	
11. Przygotowanie dokumentacji						4 dni

Rys. 3.1. Diagram Gantta przedstawiający harmonogram realizacji projektu

Prace nad projektem rozpoczęto od zaprojektowania bazy danych oraz struktury klas aplikacji. Następnie stworzono graficzny interfejs użytkownika przy użyciu biblioteki Swing. W kolejnych etapach zaimplementowano funkcjonalności takie jak:

- rejestracja i logowanie użytkownika,
- dodawanie, edytowanie oraz usuwanie drużyn,
- zarządzanie piłkarzami w drużynach,
- komunikacja z bazą danych oraz odświeżanie danych w GUI.

Po zakończeniu implementacji przeprowadzono testowanie wszystkich funkcji, usunięto wykryte błędy oraz przygotowano dokumentację projektu.

Do zarządzania kodem źródłowym wykorzystano system kontroli wersji **Git**. Repozytorium projektu zostało umieszczone na platformie **GitHub** pod adresem:

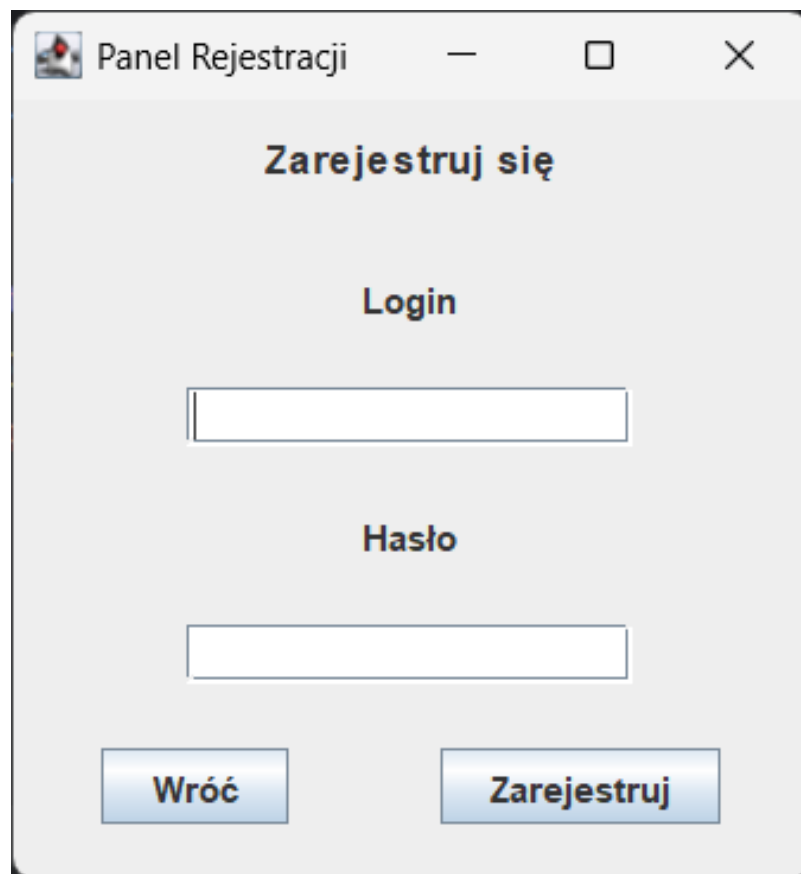
https://github.com/KubaP43/PO_Java_UR

4. Prezentacja warstwy użytkowej projektu

Aplikacja desktopowa została stworzona w języku Java z wykorzystaniem biblioteki Swing do implementacji warstwy graficznej (GUI). Umożliwia zarządzanie drużynami piłkarskimi – dodawanie, edycję oraz przypisywanie zawodników – z uwzględnieniem podziału na funkcjonalne moduły GUI.

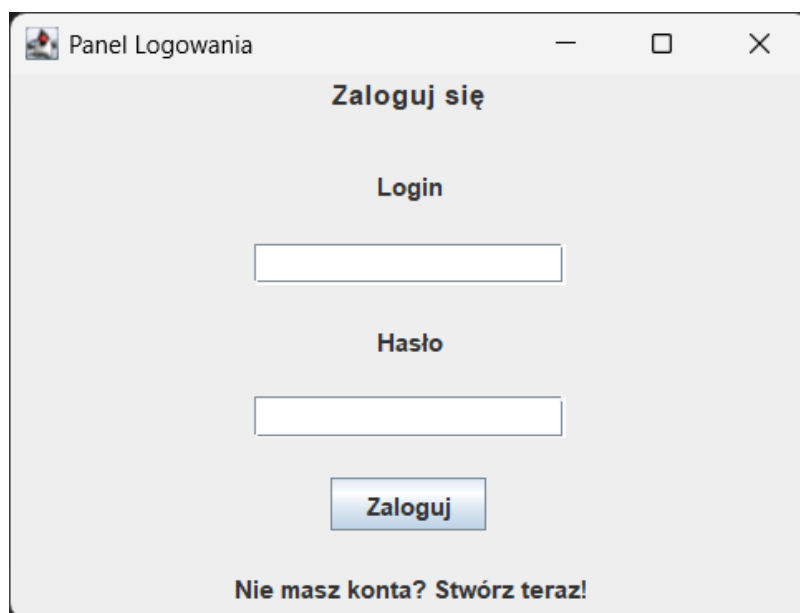
Poniżej przedstawiono opis poszczególnych paneli aplikacji wraz z odpowiednimi zrzutami ekranu.

- **RejestracjaGUI** – panel służący do zakładania nowego konta. Użytkownik wprowadza nazwę użytkownika i hasło, które po zatwierdzeniu zostają zapisane w bazie danych.



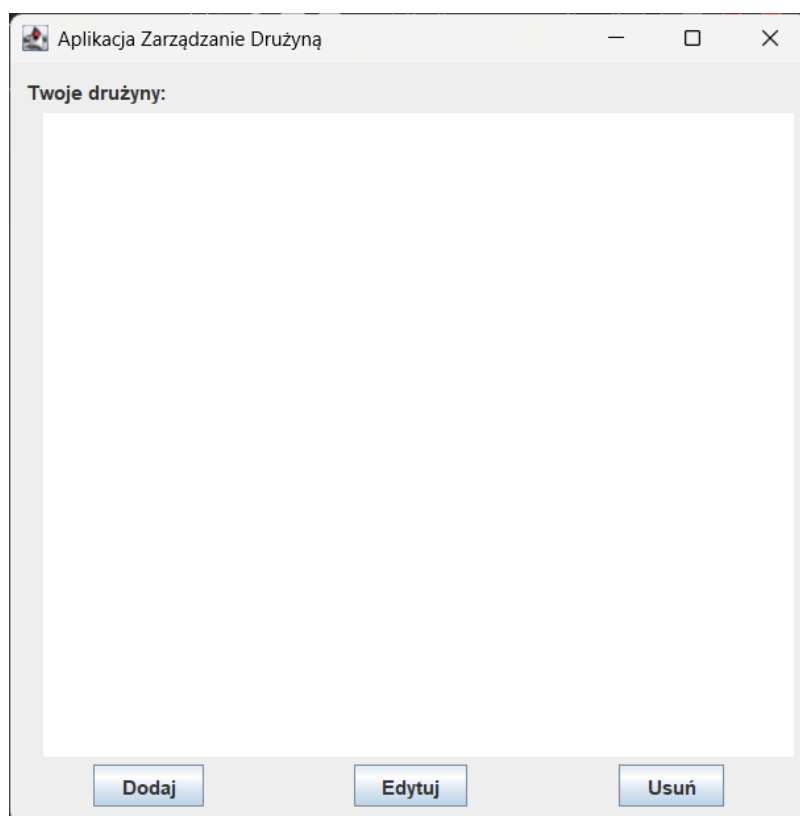
Rys. 4.1. Panel rejestracji użytkownika (RejestracjaGUI).

- **LoginGUI** – panel logowania dla zarejestrowanych użytkowników. Po podaniu prawidłowych danych, użytkownik zostaje przekierowany do głównego interfejsu.



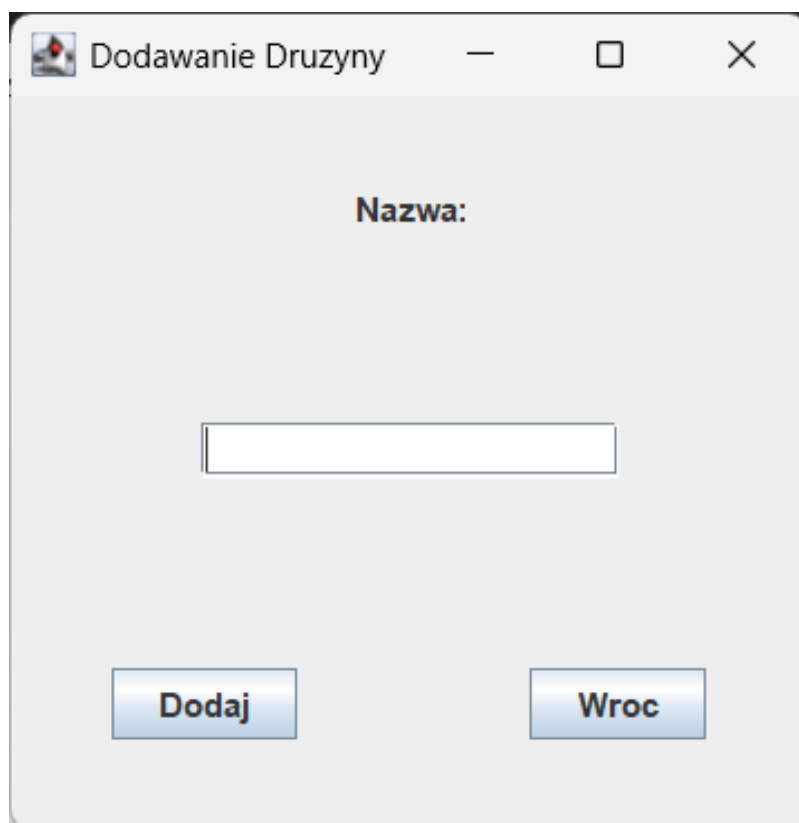
Rys. 4.2. Panel logowania użytkownika (LoginGUI).

- **AplikacjaGUI** – główny panel aplikacji. Z jego poziomu użytkownik może zarządzać drużynami i zawodnikami, a także przechodzić do pozostałych funkcji systemu.



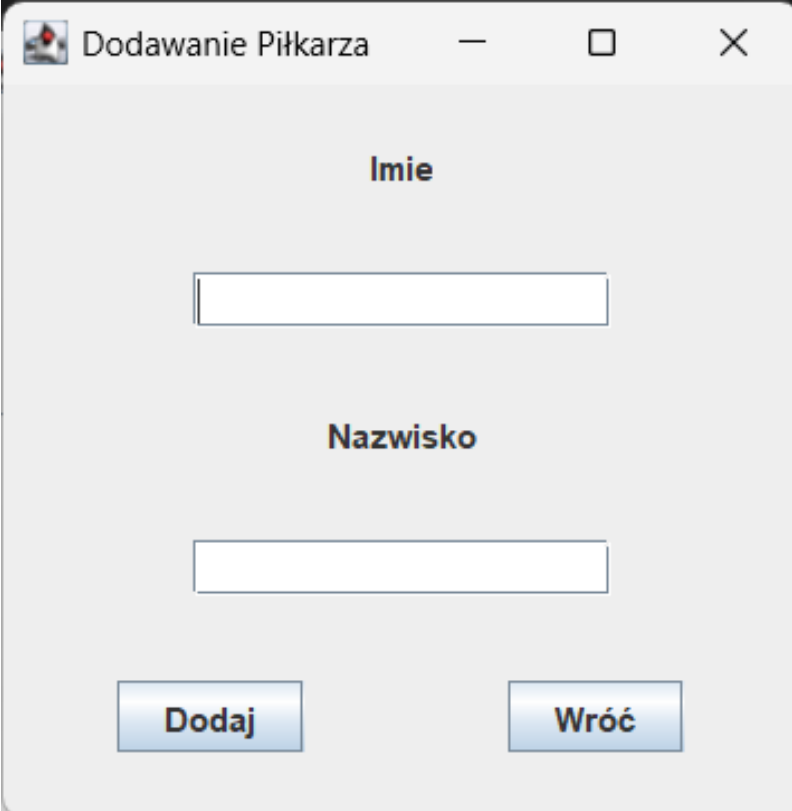
Rys. 4.3. Panel główny aplikacji (AplikacjaGUI).

- **DodawanieDruzynyGUI** – panel umożliwiający dodanie nowej drużyny do systemu. Użytkownik podaje nazwę, kraj oraz inne podstawowe informacje.



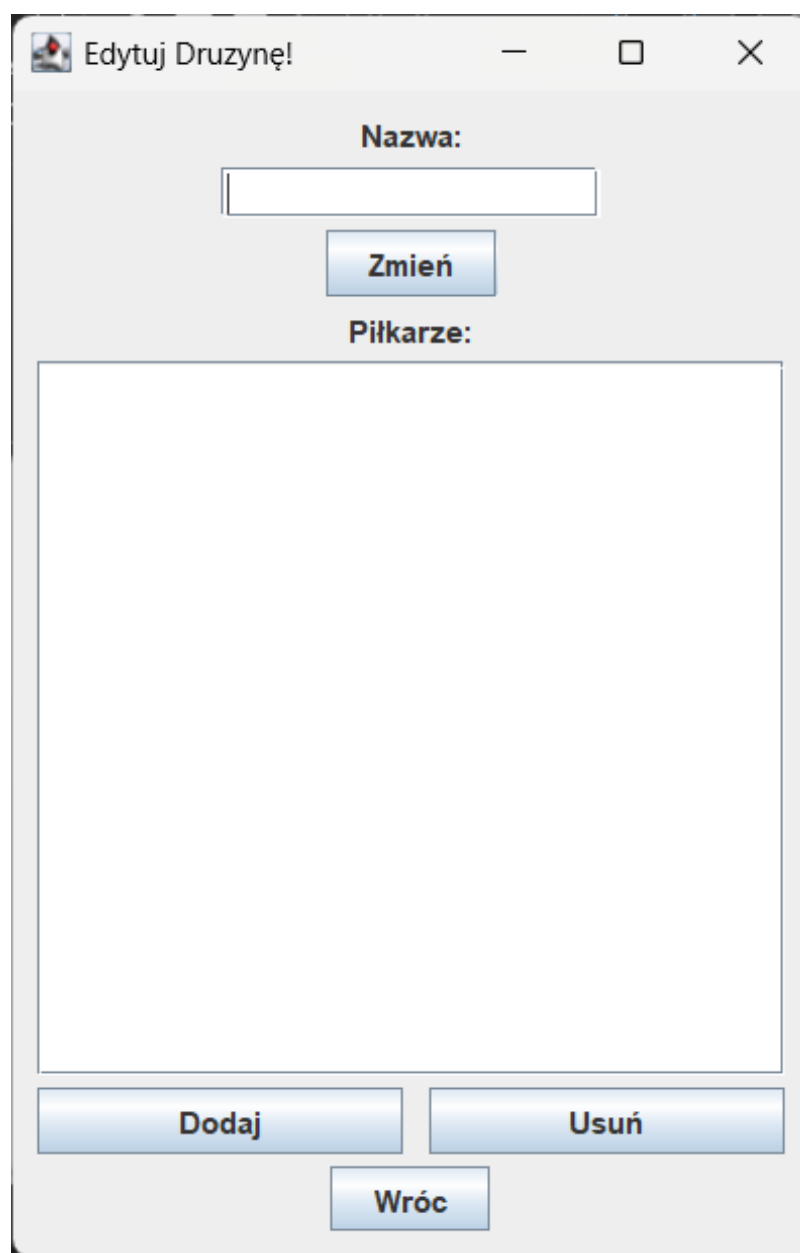
Rys. 4.4. Dodawanie nowej drużyny (DodawanieDruzynyGUI).

- **DodajPiłkarzaGUI** – interfejs służący do dodawania zawodników do drużyny. Można w nim określić imię, nazwisko, pozycję i inne dane piłkarza.



Rys. 4.5. Dodawanie piłkarza do drużyny (DodajPiłkarzaGUI).

- **EdytujDruzyneGUI** – panel edycji istniejącej drużyny. Użytkownik może zmieniać dane drużyny oraz usuwać ją z systemu.



Rys. 4.6. Edycja danych drużyny (EdytujDruzyneGUI).

Cały interfejs został zaprojektowany w sposób przejrzysty i intuicyjny, z podziałem na funkcjonalne panele umożliwiające użytkownikowi pełną kontrolę nad tworzonymi danymi.

5. Podsumowanie

W ramach projektu zrealizowano aplikację desktopową służącą do zarządzania drużynami piłkarskimi. Główne funkcjonalności obejmują możliwość rejestracji i logowania użytkownika, dodawania nowych drużyn oraz przypisywania do nich piłkarzy. Dodatkowo zaimplementowano mechanizmy edycji istniejących drużyn, a całość została zorganizowana w formie przejrzystych graficznych interfejsów użytkownika (GUI), opartych na bibliotece Swing.

Projekt został zrealizowany zgodnie z założeniami, a jego struktura umożliwia łatwą rozbudowę w przyszłości. Do najważniejszych osiągnięć należy:

- zaprojektowanie i implementacja funkcjonalnych paneli GUI,
- integracja z bazą danych za pomocą JDBC,
- pełna obsługa operacji CRUD (Create, Read, Update, Delete) dla drużyn i piłkarzy,
- zastosowanie systemu kontroli wersji Git i współpraca zdalna za pomocą repozytorium GitHub.

Możliwe kierunki rozwoju projektu obejmują:

- dodanie możliwości przeglądania szczegółowych statystyk piłkarzy i drużyn,
- implementację systemu logowania z poziomami dostępu (np. admin, użytkownik),
- rozbudowę interfejsu graficznego z wykorzystaniem nowoczesnych bibliotek JavaFX lub frameworków webowych,
- wdrożenie walidacji danych użytkownika i komunikatów błędów w czasie rzeczywistym,
- eksport danych do plików PDF lub Excel.

Zrealizowany projekt stanowi solidną podstawę pod dalszy rozwój i integrację z bardziej zaawansowanymi funkcjonalnościami w przyszłości.

Spis rysunków

2.1	Diagram ERD przedstawiający relacje w bazie danych	7
3.1	Diagram Gantta przedstawiający harmonogram realizacji projektu	8
4.1	Panel rejestracji użytkownika (RejestracjaGUI).	9
4.2	Panel logowania użytkownika (LoginGUI).	10
4.3	Panel główny aplikacji (AplikacjaGUI).	10
4.4	Dodawanie nowej drużyny (DodawanieDrużynyGUI).	11
4.5	Dodawanie piłkarza do drużyny (DodajPiłkarzaGUI).	12
4.6	Edycja danych drużyny (EdytujDrużynyGUI).	13

Załącznik nr 2 do Zarządzenia nr 228/2021 Rektora Uniwersytetu Rzeszowskiego z dnia 1 grudnia 2021 roku w sprawie ustalenia procedury antyplagiatowej w Uniwersytecie Rzeszowskim

OŚWIADCZENIE STUDENTA O SAMODZIELNOŚCI PRACY

Jakub Michał Relic

Imię (imiona) i nazwisko studenta

Wydział Nauk Ścisłych i Technicznych

Informatyka

Nazwa kierunku

13 495 7

Numer albumu

1. Oświadczam, że moja praca projektowa pt.: Przygotowanie dokumentacji do projektu w systemie L^AT_EX^{*}
 - 1) została przygotowana przeze mnie samodzielnie*,
 - 2) nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2021 r., poz. 1062) oraz dóbr osobistych chronionych prawem cywilnym,
 - 3) nie zawiera danych i informacji, które uzyskałem/am w sposób niedozwolony,
 - 4) nie była podstawą otrzymania oceny z innego przedmiotu na uczelni wyższej ani mnie, ani innej osobie.
2. Jednocześnie wyrażam ~~zgody~~^{zgoda}/~~nie wyrażam zgody~~^{nie wyrażam zgody}** na udostępnienie mojej pracy projektowej do celów naukowo-badawczych z poszanowaniem przepisów ustawy o prawie autorskim i prawach pokrewnych.

Przerzaw 14.06.2025

(miejscowość, data)

Jakub Relic

(czytelny podpis studenta)

* Uwzględniając merytoryczny wkład prowadzącego przedmiot

** – niepotrzebne skreślić