P O L I T E C H N I K A Ś W I Ę T O K R Z Y S K A

Wydział Elektrotechniki, Automatyki i Informatyki

Szymon Baran

Numer albumu: 88935

Aplikacja internetowa dla drużyn sportowych

Praca dyplomowa

na studiach I-go stopnia

na kierunku Informatyka

Opiekun pracy dyplomowej:

Dr inż. Ludomir Tuszyński

Katedra Informatyki Stosowanej

Kielce, 2022

POLITECHNIKA ŚWIĘTOKRZYSKA
WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI
Studia stacjonarne
Kierunek: INFORMATYKA

Zatwierdzam: **Rok akademicki: 2021/22**

PRODZIEKAN
ds. Studenckich i Dydaktyki
Wydziału Elektrotechniki, Automatyki i Informatyki
……
Prodziekan ds. studenckich i dydaktyki
dr inż. Barbara Łukawska

ZADANIE NA PRACĘ DYPLOMOWĄ STUDIÓW PIERWSZEGO STOPNIA

Wydano dla studenta:

Szymon Baran

I. Temat pracy:

Aplikacja internetowa dla drużyn sportowych
Web application for sports teams

II. Plan pracy:
1. Opis problematyki zarządzania drużynami sportowymi.
2. Przykładowe internetowe aplikacje dla drużyn sportowych.
3. Charakterystyka narzędzi do budowy stron internetowych.
4. Omówienie systemów baz danych.
5. Projekt aplikacji internetowej dla drużyn sportowych.
6. Realizacja programu komputerowego i weryfikacja jego funkcjonowania.
7. Uwagi i wnioski.

III. Cel pracy:
Celem pracy jest stworzenie aplikacji internetowej dla drużyn sportowych.

IV. Uwagi dotyczące pracy:

V. Termin oddania pracy: zgodnie z Regulaminem Studiów tj. do końca zajęć semestru dyplomowego.

VI. Konsultant:

Kierownik Katedry
KIEROWNIK
Katedry Informatyki Stosowanej
Wydziału Elektrotechniki Automatyki i Informatyki
dr hab. inż. Paweł Sitek, prof. PŚk
……
(pieczęć i podpis)

Promotor pracy dyplomowej
……
Dr inż. Ludomir Tuszyński

Temat pracy dyplomowej celem jej wykonania otrzymałem(am):

Kielce, dnia 31.05.2021 r. Baran Szymon
czytelny podpis studenta

Politechnika Świętokrzyska

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI

Kielce, dnia 12.01.2022

SZYMON BARAN 088935

Imię i nazwisko studenta nr albumu

SZCZUKOWICE 144 26-065 PIEKOSZÓW

Adres zamieszkania

INFORMATYKA, TELEINFORMATYKA, II, STACJONARNE

Kierunek, specjalność, rok studiów, rodzaj studiów (stacjonarne, niestacjonarne)

DR INŻ. LUDOMIR TUSZYŃSKI

Opiekun pracy dyplomowej inżynierskiej/~~magisterskiej~~*

OŚWIADCZENIE

Przedkładając w roku akademickim 2021/22. opiekunowi pracy dyplomowej inżynierskiej/~~magisterskiej~~*, powołanemu przez Dziekana Wydziału Elektrotechniki Automatyki i Informatyki Politechniki Świętokrzyskiej, pracę dyplomową inżynierską/~~magisterską~~* pod tytułem:

APLIKACJA INTERNETOWA DLA DRUŻYN SPORTOWYCH

oświadczam, że:

1) przedstawiona praca dyplomowa inżynierska/~~magisterska~~* została opracowana przeze mnie samodzielnie, stosownie do wskazówek merytorycznych opiekuna pracy,
2) przy wykonywaniu pracy dyplomowej inżynierskiej/~~magisterskiej~~* wykorzystano materiały źródłowe, w granicach dozwolonego użytku wymieniając autora, tytuł pozycji i miejsce jej publikacji,
3) praca dyplomowa inżynierska/~~magisterska~~* nie zawiera żadnych danych, informacji i materiałów, których publikacja nie jest prawnie dozwolona,
4) przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem stopnia zawodowego/naukowego w wyższej uczelni,
5) niniejsza wersja pracy jest identyczna z załączoną treścią elektroniczną (na CD i w systemie Archiwum Prac Dyplomowych).

Przyjmuję do wiadomości, iż w przypadku ujawnienia naruszenia przepisów ustawy o prawie autorskim i prawach pokrewnych, praca dyplomowa inżynierska/~~magisterska~~* może być unieważniona przez Uczelnię, nawet po przeprowadzeniu obrony pracy.

Zostałem uprzedzony:

1) o odpowiedzialności karnej wynikającej z art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. 1994 Nr 24, poz. 83, t.j. Dz. U. 2018 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.",
2) o odpowiedzialności dyscyplinarnej na podstawie art. 307 ust. 1 i 2 ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668, z późn. zm.) „Student podlega odpowiedzialności dyscyplinarnej za naruszenie przepisów obowiązujących w uczelni oraz za czyn uchybiający godności studenta. Student może być ukarany przez rektora lub komisję dyscyplinarną".

Prawdziwość powyższego oświadczenia potwierdzam własnoręcznym podpisem.

Szymon Baran

czytelny podpis studenta

**) niepotrzebne skreślić*

Aplikacja internetowa dla drużyn sportowych

Streszczenie

Celem niniejszej pracy dyplomowej, było utworzenie aplikacji internetowej dla drużyn sportowych. Jej zadaniem powinno być uproszczenie oraz usprawnienie procesów wykonywanych w obrębie drużyny przez jej zarząd oraz zawodników. Praca przedstawia problematykę zadania, wykorzystane technologie informatyczne oraz sposób ich implementacji, wyciągnięte wnioski, a także plany na dalszy rozwój aplikacji.

Słowa kluczowe: drużyna sportowa, aplikacja internetowa, .NET, Vue

Web application for sports teams

Summary

The main goal of this thesis was to create a web application for sports teams. Its main task is to simplify and improve all types of processes happening inside of sports team, concerning its board and players. This thesis presents issues of the task, used web technologies (including their implementation way), summary of the results and some ideas for further development of the application.

Keywords: sports team, web application, .NET, Vue

Spis treści

1. Wstęp

Internet jest nieodłączną częścią funkcjonowania społeczeństwa XXI wieku. Poza aspektami rozrywkowymi, pozwala usprawnić funkcjonowanie wielu dziedzin życia. Również w przypadku sportu, może odgrywać istotną rolę i podnieść jakość funkcjonowania drużyn oraz zawodników.

Celem niniejszej pracy jest utworzenie aplikacji internetowej dla drużyn sportowych. Jej zadaniem byłoby usprawnienie funkcjonowania takich drużyn, poprzez uproszczenie interakcji i zautomatyzowanie procesów zachodzących pomiędzy zawodnikami, członkami zarządu, nie tylko w obrębie jednego zespołu, ale również z rywalami.

W drugim rozdziale pracy opisano problematykę zarządzania drużynami sportowymi – na jakie wyzwania natrafiają poszczególne osoby podczas codziennej pracy. Porównano również utworzoną aplikację do rozwiązań istniejących obecnie na rynku. Trzeci rozdział skupia się na scharakteryzowaniu technologii informatycznych zastosowanych podczas tworzenia strony. Opisano zarówno technologie wykorzystywane po stronie klienta, jak i te, z których korzystano po stronie serwera aplikacji. W czwartym rozdziale przedstawione zostały założenia projektowe – to na ich podstawie zrealizowana została aplikacja internetowa. Część piąta jest najbardziej obszerna, gdyż to tutaj opisano funkcjonalności utworzonej strony. Szósty rozdział skupia się na testach aplikacji – w jaki sposób je wykonano oraz jaki przyniosły skutek. Część ostatnia pracy to wnioski oraz przedstawienie planów na dalszy rozwój aplikacji.

2. Problematyka zarządzania drużynami sportowymi

Operacje wykonywane przez drużyny sportowe przebiegają często w sposób nieoptymalny. Pomimo szybkiego rozwoju technologii informatycznych, w tej dziedzinie rozwiązania nie są chętnie implementowane. Szczególnie na niższych szczeblach rozgrywkowych, zespoły preferują metody komunikacyjne, które są albo przestarzałe, albo nieprzystosowane do wykonywanych procesów.

Fundamentalną rolę odgrywa zebranie wszystkich potrzebnych funkcjonalności w jednym miejscu. Problemem jest dezintegracja systemów – sytuacja, gdy nawet w obrębie jednej drużyny, wykorzystywane są różne aplikacje, ponieważ pojedyncza nie jest w stanie spełnić wszystkich postawionych wymagań.

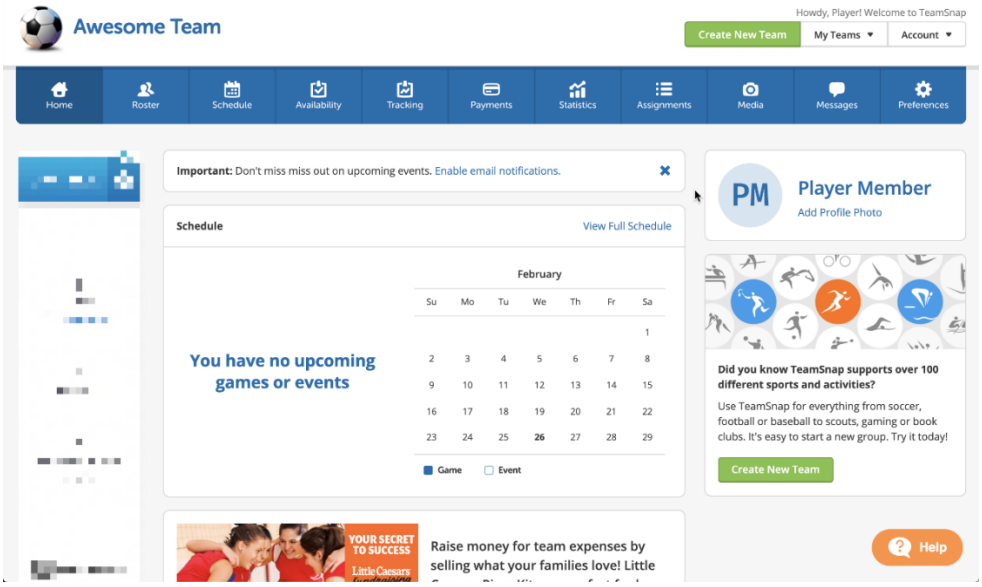
Podstawą funkcjonowania każdego zespołu jest komunikacja. Jej podstawową cechą powinna być niezawodność – zapewnienie nadawcy, że jego wiadomość została dostarczona oraz umożliwienie sprawdzenia, czy została odczytana przez odbiorcę. Przydatna jest możliwość formatowania wysyłanych wiadomości – możliwość załączenia tabeli z pożądanymi informacjami, listy, dodania nagłówka, zmiany rozmiaru tekstu. Użyteczną funkcjonalnością byłaby możliwość wysyłania wiadomości grupowych – od członka zarządu zespołu, do wszystkich jego zawodników. Mogłyby one zawierać informacje, takie jak harmonogram treningów na najbliższy tydzień, czy przekazanie ułożonej diety lub innych informacji organizacyjnych.

Istotnym zadaniem członków sztabu drużyn sportowych jest dokonywanie analizy. Dotyczy ona zarówno własnych zawodników, jak i potencjalnych rywali. Do osiągnięcia odpowiednich rezultatów, wymagana jest baza danych, zawierająca drużyny, zawodników oraz rozegrane mecze, wraz z wszelkimi statystykami oraz informacjami, które można filtrować oraz sortować. Dodatkiem upraszczającym te operacje, jest przedstawienie danych w postaci graficznej.

Przeprowadzanie rekrutacji do zespołu zawsze jest skomplikowaną operacją, bez względu na dziedzinę. Faktem jest, że żadna aplikacja nie jest w stanie zastąpić ludzkiej oceny potencjalnego kandydata, ale byłaby przydatna w wykonywaniu wstępnej selekcji oraz poszukiwaniu odpowiednich kandydatów o odpowiedniej charakterystyce. Również operacja ustalania transferów pomiędzy dwoma zespołami mogłaby zostać zautomatyzowana, poprzez określanie szczegółów, takich jak kwota odstępnego oraz pensja.

2.1. Analiza rozwiązań alternatywnych do zarządzania drużynami sportowymi

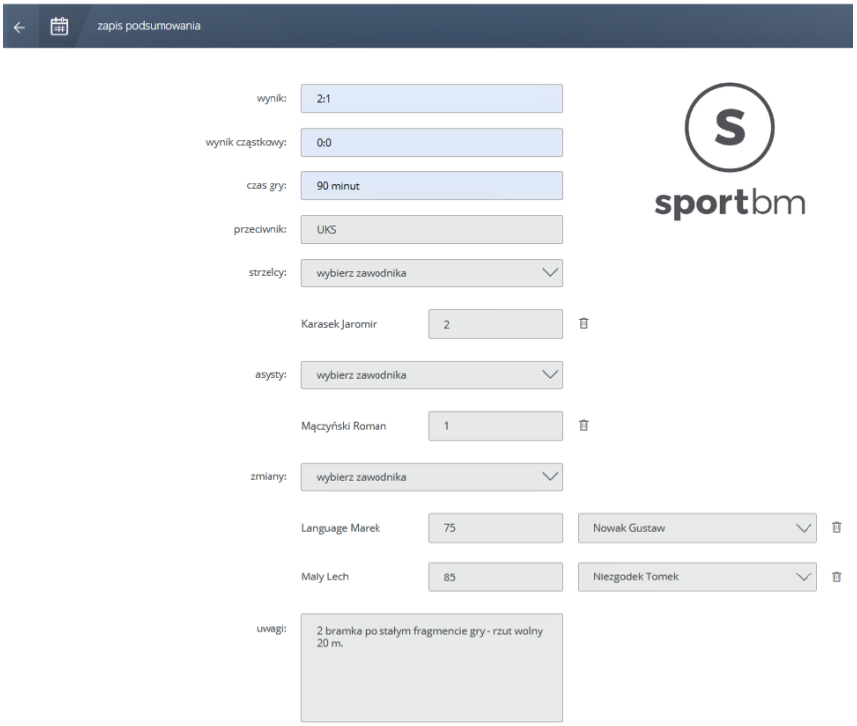
– TeamSnap – system umożliwiający zarządzanie drużyną sportową. Założony w 2009 roku, umożliwia korzystanie z 3 planów płatniczych – od 9,99 USD do 17,99 USD za miesiąc, gdzie wraz ze wzrostem opłat, użytkownik otrzymuje możliwość korzystania z większej liczby funkcjonalności, takich jak personalizacja profilu drużyny własnym herbem, dodawanie oznaczeń sponsorskich na profilu zespołu oraz śledzenie statystyk. Aplikacja nie obsługuje języka polskiego [1].



Rys. 2.1. Profil drużyny w aplikacji TeamSnap

Źródło: [1]

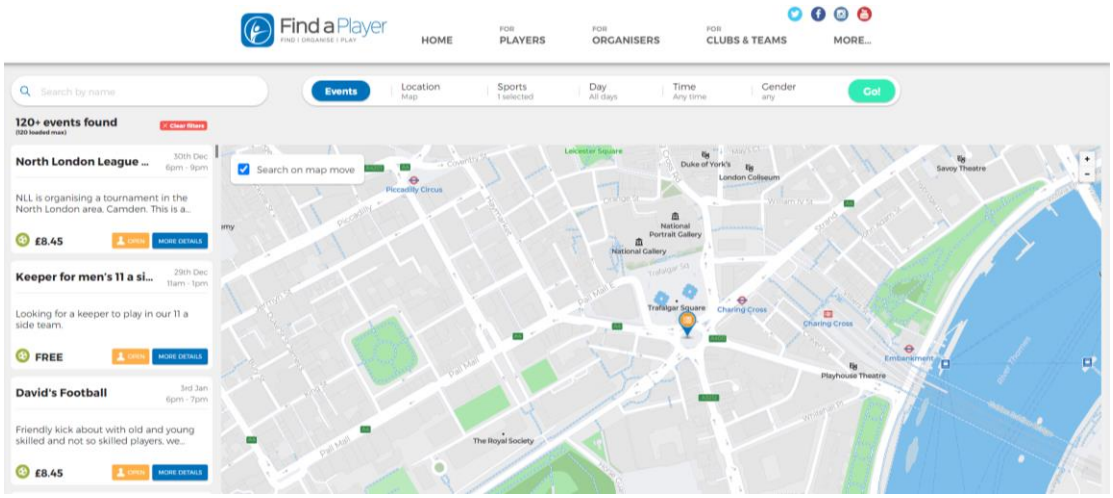
– Sportbm – polskie rozwiązanie, wspierające dużą liczbę funkcjonalności, takich jak planowanie meczy, wysyłanie wiadomości, utworzenie profilu drużyny i zarządzanie nim, tworzenie schematów treningów. Jej koszt wynosi 2zł za każdego zarejestrowanego zawodnika w drużynie za każdy miesiąc użytkowania. Funkcjonuje jako partner technologiczny organizacji takich jak Pomorski Związek Piłki Nożnej, Śląski Związek Piłki Nożnej oraz Polski Związek Unihokeja oraz akademii drużyn, takich jak Bytovia Bytów i Zagłębie Sosnowiec [2].



Rys 2.2. Podsumowanie meczu w aplikacji sportbm

Źródło: [2]

– Find a Player – aplikacja internetowa, utworzona przez szkocką firmę w celu poszukiwania znajomych do uprawiania sportu. Umożliwia poszukiwanie zawodników dla drużyn na stałe lub do jednorazowego meczu. Umożliwia wysyłanie wiadomości, ogłaszanie treningów otwartych dla kandydatów do dołączenia do zespołu, a następnie organizowanie meczy z ich udziałem [3].



Rys. 2.3. Wyniki wyszukiwania zawodników dostępnych do gry w piłkę nożną w Londynie w aplikacji Find a Player

Źródło: [3]

3. Charakterystyka narzędzi do budowy stron internetowych oraz systemów baz danych

Pierwszym krokiem wykonywania aplikacji internetowych jest dobór odpowiednich technologii informatycznych. Wszystkie z nich posiadają swoje zalety oraz wady. Dodatkowo, niektóre z nich lepiej współpracują ze sobą, dzięki czemu ich integracja jest prostsza. Podczas wykonywania tego procesu, należy wziąć także pod uwagę utworzone założenia projektowe i ocenić, czy wybrane narzędzie będzie odpowiednie do danego celu.

3.1. ASP.NET Core

ASP.NET Core jest platformą umożliwiającą tworzenie aplikacji, opartą na licencji open source. Została utworzona przez firmę Microsoft. Dzięki platformie .NET Core, tworzone aplikacje, mogą zostać wykorzystane na wielu systemach, w tym Windows, MacOS oraz Linux. Pozwala na tworzenie aplikacji internetowych, dzięki wykorzystaniu modelu Razor Pages oraz dobrze zoptymalizowanych i wydajnych Web API [4].

API (interfejs programowania aplikacji) jest nieodzowną częścią aplikacji tworzonych na bazie modelu single page application. Zakłada on sytuację, że część podstawowa strony jest tworzona z wykorzystaniem HTML + JavaScript + CSS, podczas gdy wszelkie czynności są wykonywane z wykorzystaniem REST API, a informacje uzyskane z bazy danych są otrzymywane w formacie XML lub JSON. Komunikacja odbywa się z wykorzystaniem protokołu HTTP, poprzez korzystanie z kontrolerów – klient wysyła zapytanie, na odpowiedni adres – endpoint, który jest przygotowaną metodą w kontrolerze. API przygotowuje i zwraca odpowiedź, którą otrzymuje następnie użytkownik [5].

Listing 3.1. Endpoint w *PlayersController*, przyjmujący opcjonalny parametr - identyfikator drużyny

```
[HttpGet]
public ActionResult<IEnumerable<Player>> GetPlayers(Guid? teamId)
{
    if (teamId.HasValue)
    {
        return Ok(PlayerService.GetPlayersByTeamId(teamId.Value));
    }
    else
    {
        return Ok(PlayerService.GetPlayers());
    }
}
```

Źródło: Opracowanie własne

Rozwiązanie przygotowane w ten sposób, może korzystać z usprawnień oraz funkcjonalności, takich jak asynchroniczność (async/await), wyrażenia lambda oraz LINQ.

LINQ (Language Integrated Query) to zintegrowane zapytania językowe, upraszczające pracę z danymi. Dzięki temu rozwiązaniu, programista nie musi wykorzystywać innego języka zapytań dla każdego źródła danych. Zamiast tego pracuje na obiektach, niezależnie czy aktualnie wykorzystuje bazę danych SQL czy jedną z kolekcji dostępnych w C# [6].

Listing 3.2. Metoda, wykorzystująca LINQ do pobrania zawodników należących do wybranej drużyny

```
public List<Player> GetPlayersInTeam(Guid teamId)
{
	return _context.Players.Where(x => x.UserType == UserType.Player &&
	x.TeamId == teamId).ToList();
}
```

Źródło: Opracowanie własne

Pakiety NuGet są kolejnym przydatnym mechanizmem zawartym w platformie .NET Core. Umożliwiają dodawanie do własnej aplikacji zewnętrznych bibliotek utworzonych i udostępnionych przez innych programistów. Przykładami bibliotek wykorzystanych w projekcie serwera są:

- AutoMapper – prosta biblioteka, służąca do automatycznego mapowania klas.
- AWSSDK.S3 – biblioteka, dostarczająca metody integrujące API z wykorzystanym nośnikiem danych – Amazon AWS S3.
- BCrypt.NET Next – dostarcza narzędzia do szyfrowania uzupełnionych haseł, zgodnie z funkcją bcrypt.

Jako IDE do tworzenia API wykorzystano w projekcie program Visual Studio 2019. Środowisko zostało utworzone – podobnie jak sam .NET Core – przez firmę Microsoft, dzięki czemu praca przebiega w sposób optymalny. Obsługuje rozszerzenia oraz motywy, które pozwalają na przystosowanie środowiska pracy do własnych potrzeb. Edytor wspiera również Intellisense – narzędzie dostarczające podpowiedzi w czasie rzeczywistym podczas pisania kodu oraz rozbudowany debugger.

3.2. Vue.js

JavaScript jest językiem programowania, umożliwiającym stronie dynamiczną modyfikację wyświetlanych informacji, implementację rozbudowanych animacji oraz odtwarzanie filmów. Obsługuje nasłuchiwanie zdarzeń na stronie (np. kliknięcie przycisku). Umożliwia także wykorzystywanie zewnętrznych API [7].

Vue jest jednym z najpopularniejszych frameworków, opartych właśnie na JavaScripcie, zbudowanym na licencji opensource. Został zaprojektowany tak, aby być jak najbardziej przystępny do nauki. Aby rozpocząć pracę, wymagana jest jedynie podstawowa wiedza na temat HTML, JavaScript oraz CSS i takie właśnie podstawowe sekcje można wydzielić w komponencie w Vue.

Listing 3.3. Podział prostego komponentu w Vue na 3 podstawowe sekcje

```
<template>
  <p>{{ test }}</p>
</template>

<script>
export default {
  data() {
    return {
      test: 10,
    };
  },
};
</script>

<style>
p {
  font-size: 20px;
}
</style>
```

Źródło: Opracowanie własne

Działanie Vue, oparte jest na podziale strony na małe, często wielokrotnie używane komponenty, które są ładowane tylko w momencie, kiedy są potrzebne. Jego istotną zaletą jest więc szybkość. Autorzy frameworka udostępniają również rozszerzenie do narzędzi deweloperskich, w znaczny sposób upraszczające proces debugowania tworzonej aplikacji internetowej. Zalecanym środowiskiem programistycznym jest Visual Studio Code z rozszerzeniem Volar [8].

Funkcjonalności Vue mogą zostać rozszerzone, poprzez skorzystanie z pakietów bibliotek zamieszczonych w katalogu npm. Niektóre z wykorzystanych w ramach projektu aplikacji, opisano poniżej:

- Vuex – wykorzystywany do zarządzania stanem aplikacji. Upraszcza komunikację między tworzonymi komponentami. Jego schemat działania, polega na wywołaniu akcji w komponencie, która wywołuje mutację na stanie w storze, który następnie może z wykorzystaniem gettera zostać pobrany na komponent. Takie podejście jest intuicyjne podczas pracy z Web API, gdy wywoływane asynchronicznie akcje, wykonują zapytanie do API, aby uzyskać odpowiedź, która zostanie zapisana w stanie aplikacji. Dane przechowywane w taki sposób mogą być odczytywane przez wiele komponentów [9].
- Axios – biblioteka, dostarczająca klient HTTP, umożliwiający komunikację z API. Jego działanie jest oparte na asynchroniczności [10].
- Vue Router – oficjalny router wykorzystywany w aplikacjach Vue. Umożliwia tworzenie aplikacji opartych o model single-page application [11].
- DevExtreme – rozbudowana biblioteka, dostarczająca gotowe komponenty UI. Do użytku niekomercyjnego, jest ona dostępna za darmo. Udostępnia dużą liczbę tabel z danymi, wykresów, przycisków nawigacyjnych oraz paneli, a wszystkie z nich są responsywne oraz wysoce konfigurowalne do własnych potrzeb. Twórcy biblioteki udostępniają również bardzo rozbudowaną dokumentację z demonstracyjnymi przykładami zastosowań ich rozwiązań, co znacznie usprawnia proces ich implementacji [12].
- Bootstrap – stanowi duże uproszczenie prac w pozycjonowaniu elementów na widoku, dzięki wbudowanej siatce. Jest znacznym wsparciem w nadawaniu odpowiednich styli elementom na stronie. Biblioteka oferuje także własny katalog ikon oraz motywów, które mogą zostać wykorzystane do personalizacji strony internetowej [13].

3.3. HTML i CSS

HTML to język znaczników, niezbędny do tworzenia stron internetowych. Służy do tworzenia całej struktury strony i wszystkich elementów, które się w niej znajdują, dodawanych za pośrednictwem odpowiednich tagów. Pierwsza wersja języka HTML powstała w roku 1991, aktualnie obowiązująca jest wersja 5 [14].

Listing 3.4. Przykładowy element w języku HTML – nagłówek h3

```
<h3>Wiadomości</h3>
```

Źródło: Opracowanie własne

Kaskadowe arkusze stylów (CSS) są językiem arkuszy stylów, służącym do personalizowania wyglądu utworzonych elementów na stronie internetowej, dodanych wcześniej za pomocą HTML. Posiadają unikalną składnię, służącą tworzeniu dyrektyw [15]. Można dołączyć je do swojego kodu na kilka różnych sposobów. Najpopularniejszym z nich jest utworzenie osobnego pliku z kodem CSS. Drugi sposób to umieszczenie kodu w odpowiedniej sekcji, bezpośrednio w pliku, który powinien zostać zmodyfikowany. Ostatni sposób to dołączenie atrybutu style w wybranym obiekcie HTML (tzw. inline CSS).

Listing 3.5. Przykładowa dyrektywa w języku CSS – stylowanie nagłówków h2

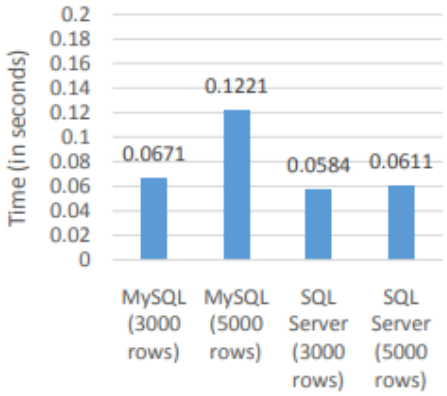
```
h2 {
  font-size: 150%;
  margin: 0;
  font-weight: bold;
  letter-spacing: 0.06em;
}
```

Źródło: Opracowanie własne

3.4. Microsoft SQL Server

SQL Server jest systemem zarządzania relacyjną bazą danych, utworzonym przez firmę Microsoft. Wykorzystuje własną interpretację języka SQL o nazwie Transact-SQL (T-SQL). Zadebiutował w 1989 roku, natomiast jego aktualna wersja stabilna miała swoją premierę w 2019 roku [16]. T-SQL jest rozszerzeniem możliwości standardowego języka SQL, między innymi o procedury składowane. Relacyjność bazy danych, polega na organizacji danych w wielu tabelach, gdzie każda z nich posiada wiele kolumn, natomiast każdy wiersz tabeli jest unikalnym rekordem, przykładowo w tabeli Users, każdy wiersz oznacza jednego, unikalnego użytkownika [17].

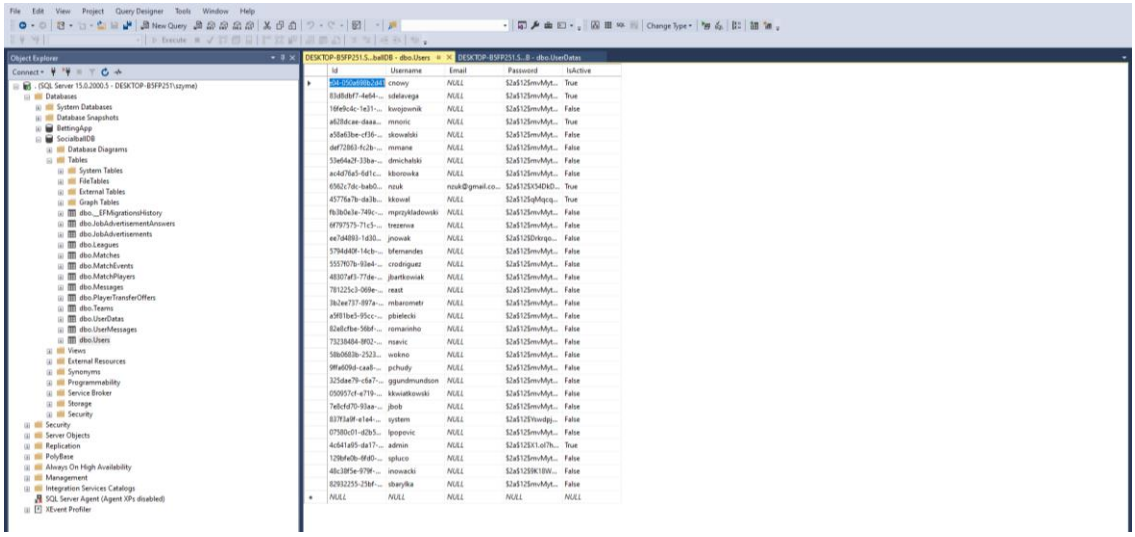
Głównymi rozwiązaniami konkurencyjnymi dla wybranego rozwiązania są MySQL oraz Oracle Database, oba stworzone przez firmę Oracle Corporation. Są to jedne z najpopularniejszych systemów bazodanowych. SQL Server jest jednak rozwiązaniem znacznie szybszym podczas wykonywania zapytań [18].



Rys. 3.1. Średni czas potrzebny obu systemom bazodanowym na wykonanie zapytania SELECT z JOIN oraz klauzulą ORDER BY

Źródło: [18]

SQL Server integruje się znacznie lepiej z wykorzystanym w projekcie środowiskiem ASP.NET Core oraz Windows, niż rozwiązania konkurencyjne takie jak wspomniane MySQL czy Oracle, ze względu na fakt, że ich autorem jest ta sama firma (Microsoft). T-SQL jest uznawany za prostszy w użyciu w porównaniu do języka Oracle PL/SQL, który z kolei posiada więcej wbudowanych funkcjonalności. Kolejną zaletą jest intuicyjne i proste w obsłudze graficzne narzędzie do zarządzania bazami danych – SQL Server Management Studio (SSMS).



Rys. 3.2. Edycja użytkowników z wykorzystaniem narzędzia SSMS

Źródło: Opracowanie własne

3.5. Git

Systemy kontrolowania wersji są niezbędnym narzędziem podczas tworzenia aplikacji. Dzięki ich możliwościom obserwacji zmian wykonywanych na plikach, umożliwiają, w przypadku wystąpienia błędów, powrót jednej z wybranych poprzednich wersji. Umożliwiają też dołączanie w prosty sposób kolejnych osób do pracy nad projektem, dzięki funkcjonalności klonowania (clone) aktualnej wersji aplikacji z repozytorium na własne urządzenie. Git na tle konkurencji wyróżnia się rozwiniętym wsparciem dla tzw. gałęzi (branches) – odejść od stabilnej wersji aplikacji w celu dodawania nowych funkcjonalności, przy jednoczesnym zachowaniu oryginału. Po zakończeniu prac oraz przetestowaniu nowych rozwiązań, wykonuje się wtedy tzw. merge, czyli połączenie gałęzi z wersją główną. Jest to przydatna funkcja, a w Gicie, charakteryzuje się szybkością oraz prostotą działania [19].

4. Założenia projektowe

4.1. Projekt aplikacji

Założono wydzielenie 3 kont użytkownika (+ dodatkowe konto systemowe, służące wyłącznie do wysyłania automatycznych wiadomości-powiadomień, unikalne w obrębie całej bazy danych). Każdy zalogowany użytkownik posiada dostęp do przeglądania otrzymanych oraz wysyłania nowych wiadomości. Poza tym powinien móc przeglądać profile zawodników i zespołów, tabele ligowe oraz szczegóły rozegranych meczy. Ma także możliwość edycji danych wprowadzonych podczas rejestracji.

Podstawowy typ konta to Zawodnik. Może zostać zarejestrowany przez każdego, jedynie wybór klubu powinien zostać potwierdzony przez innego użytkownika z odpowiednimi uprawnieniami. Do jego możliwości unikalnych należy możliwość dodawania ogłoszeń o poszukiwaniu zespołu. Może również wyświetlać ogłoszenia drużyn, które poszukują zawodników na jego pozycję (wybraną w momencie rejestracji lub podczas edycji profilu). Ten typ użytkownika powinien móc akceptować propozycje dotyczące jego osoby o dołączenie do zespołu (w momencie, gdy nie należy do innej drużyny) oraz oferty transferowe (aby zostały zatwierdzone, wymagają akceptacji zarówno zawodnika, jak i jego aktualnej drużyny). Poza tym, w przypadku gdy zawodnik posiada swój zespół, otrzymuje wiadomości wewnętrzne specjalnego typu, wysyłane przez członków jego zarządu.

Bardziej zaawansowanym rodzajem konta jest Członek zarządu drużyny. Taki użytkownik posiada uprawnienia do podejmowania decyzji w imieniu zespołu, do którego jest przypisany. Każdy istniejący zespół musi posiadać co najmniej jedną taką osobę i jest ona tworzona w momencie rejestracji drużyny. Proces ten – aby zapobiec potencjalnym oszustwom – musi zostać zatwierdzony przez administratora serwisu. Jedną z funkcjonalności tego konta jest dodawanie oraz usuwanie ogłoszeń o poszukiwaniu zawodników do zespołu. Powinien móc również odpowiadać na ogłoszenia wystawione przez innych oraz zarządzać odpowiedziami na istniejące ogłoszenia drużyny – akceptować je lub odrzucać. Drugą możliwością, również związaną z zarządzaniem składem zespołu, jest zarządzanie transferami. Składa się na to wysyłanie ofert transferowych do innego zespołu, które muszą zostać zaakceptowane przez członka zarządu drugiej drużyny oraz przez samego zawodnika. W przypadku odrzucenia przez którąś z osób, oferta powinna zostać usunięta, a stosowne powiadomienie wysłane do nadawcy. Jeżeli obie strony dokonają akceptacji,

system automatycznie przeprowadzi transfer oraz uzupełni profil odpowiednimi danymi, takimi jak ustalona pensja dla zawodnika. Kolejną z ról członka zarządu jest dodawanie meczy. Jest to rozbudowany proces, wymagający dodania składów obu zespołów oraz zdarzeń meczowych, takich jak bramki, czy kary indywidualne. Przebieg takiego meczu, musi zostać zaakceptowany przez zarządcę przeciwnej drużyny i dopiero wtedy staje się widoczny. Członek zarządu powinien także mieć możliwość wprowadzania zmian na liście zawodników swojego zespołu – zarządzać kontuzjami zawodników oraz – w skrajnych przypadkach – usunąć ich uprawnienia jako zawodnika zespołu.

Ostatnim typem konta jest Administrator. Domyślnie w bazie znajduje się jeden użytkownik tego typu i nie ma możliwości zmiany tego stanu w żaden sposób. Jest to specjalny typ konta, posiadający wyjątkowe uprawnienia. Należy do nich możliwość akceptacji zarejestrowanej drużyny oraz ręczne blokowanie użytkowników i zespołów w przypadku złamania zasad korzystania z serwisu.

Projekt aplikacji zostanie utworzony w oparciu o wzorzec architektoniczny Model-View-Controller. W praktyce, polega na tym, że te 3 główne elementy współpracują ze sobą, tworząc spójną całość. Model zajmuje się reprezentacją stanu aplikacji, obiektów oraz logiki biznesowej – ma postać klas, które odpowiadają tabelom w bazie danych. Rolą widoku jest przedstawienie zawartości strony poprzez interfejs użytkownika. Kontroler jest pośrednikiem pomiędzy widokiem i modelem. Ma za zadanie odebrać żądania od użytkownika [20].

4.2. Projekt bazy danych

Opis tabel w zaprojektowanej bazie danych:

- `Leagues` – tabela przechowuje wszystkie ligi używane w bazie danych. Jej kolumny zawierają informacje, takie jak unikalny identyfikator, nazwa, kraj oraz poziom rankingowy ligi w danym kraju (stosowany do sortowania lig od najlepszej do najgorszej).
- `Teams` – tabela odpowiada za przechowywanie wszystkich drużyn w aplikacji. Przechowuje informacje o identyfikatorze, nazwie, liczbie punktów zdobytych w aktualnym sezonie, informację typu bit, czy drużyna jest aktywna (niezablokowana) oraz identyfikator ligi, do jakiej jest przypisany dany zespół.
- `Users` – w tej tabeli, przechowywane są konta użytkowników w aplikacji. Zawiera informacje techniczne, takie jak identyfikator, nazwa użytkownika, jego adres e-mail,

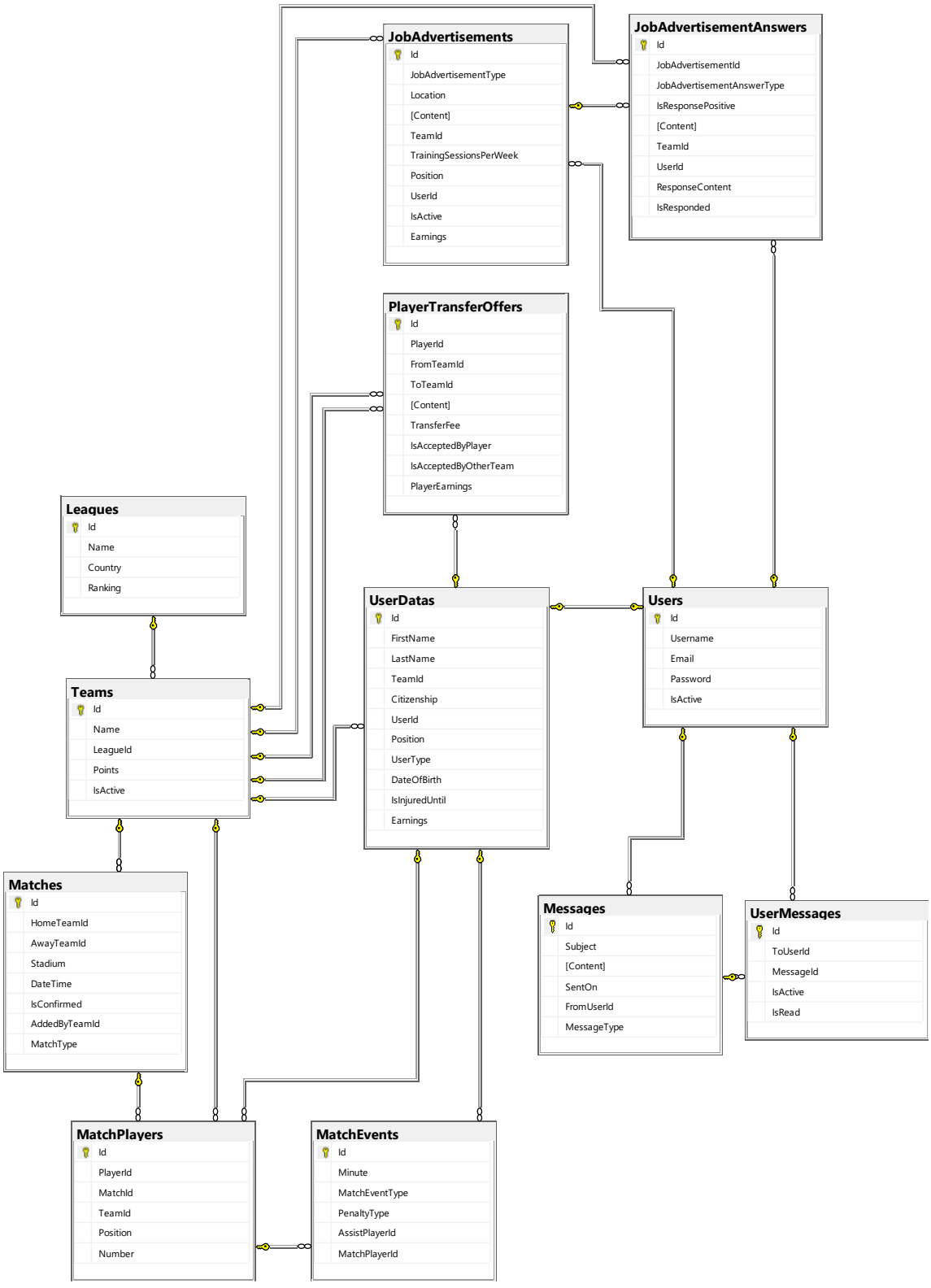
zaszyfrowane hasło oraz informację, czy użytkownik jest aktywny (czy nie jest zablokowany przez administratora serwisu).

- `UserDatas` – tabela zawiera informacje na temat użytkownika (występuje tu relacja 1-1). Jej kolumny przechowują m.in. identyfikator, imię i nazwisko, identyfikator drużyny, do której jest przypisany użytkownik, jego narodowość, id użytkownika, typ użytkownika (zawodnik/członek zarządu/administrator/system), preferowaną pozycję i przewidywaną datę zakończenia kontuzji (obie wypełniane, gdy dotyczy), datę urodzenia oraz zarobki.
- `Matches` – każdy wpis do tej tabeli odpowiada jednemu meczowi utworzonemu w aplikacji. Zawiera swój unikalny identyfikator, dwa identyfikatory drużyn – po jednym dla każdej biorącej udział w meczu, nazwę stadionu, gdzie rozgrywane jest spotkanie, dokładną datę i czas rozpoczęcia, typ meczu, określający, czy spotkanie jest towarzyskie, czy ligowe. Każdy mecz posiada także informację, czy został potwierdzony przez przeciwną drużynę oraz o tym, która drużyna wprowadziła wpis do bazy.
- `MatchPlayers` – jest to tabela, zbierająca informacje na temat występów zawodnika w meczu. Zawiera swój identyfikator, referencję do meczu i danych użytkownika (czyli zawodnika), a także informację o tym, na jakiej pozycji zagrał oraz z jakim numerem (stosowane podczas tworzenia makiety składów w widoku szczegółów).
- `MatchEvents` – tabela, określająca zdarzenia, które nastąpiły w meczu, takie jak bramki, czy kary indywidualne. W jej kolumnach, przechowywany jest identyfikator zdarzenia, identyfikator występu zawodnika w meczu, minuta zdarzenia, typ zdarzenia, typ kary (ustawiany tylko, gdy konieczne), identyfikator zawodnika asystującego przy bramce (również wypełniany, tylko gdy dotyczy).
- `Messages` – zawiera informacje na temat wysłanej wiadomości: identyfikator, jej temat oraz treść (w formie HTML), data wysłania, nadawca oraz typ wiadomości (prywatna/drużynowa/do innej drużyny). Jako, że aplikacja wspiera wysyłanie wiadomości grupowych, w tej tabeli nie ma informacji o odbiorcy.
- `UserMessages` – tabela, w której przechowywana jest relacja pomiędzy wiadomością, a jej odbiorcą. Zawiera również takie informacje, jak jej identyfikator, aktywność (czy nie została usunięta) oraz czy została odczytana.
- `JobAdvertisements` – w tej tabeli, są przechowywane ogłoszenia poszukiwania zawodników/drużyn. Jej kolumny zawierają informacje, takie jak unikalny identyfikator, typ ogłoszenia, lokalizacja, treść, identyfikator drużyny (gdy dotyczy), liczba sesji

treningowych w tygodniu, poszukiwana rola w drużynie, identyfikator użytkownika (gdy dotyczy), określenie aktualności ogłoszenia oraz potencjalne zarobki zawodnika.

– `JobAdvertisementAnswers` – każdy wpis w tabeli zawiera informacje: identyfikator, identyfikator ogłoszenia, na który wystawiono odpowiedź, typ odpowiedzi, czy jest ona potwierdzająca, czy zaprzeczająca, treść, identyfikatory drużyny oraz użytkownika, których dotyczy oraz reakcję drugiej strony (jeśli dotyczy).
– `PlayerTransferOffers` – tabela odpowiada za przechowywanie złożonych ofert transferowych. Jej kolumny zawierają takie informacje jak identyfikator, identyfikator zawodnika, którego dotyczy, identyfikatory obu drużyn – z jakiej pochodzi i do jakiej jest wysyłana propozycja, treść dołączonej wiadomości, kwota transferu i proponowanych – obie wyrażone w PLN oraz dwa pola typu bit, określające, czy oferta jest zaakceptowana przez aktualną drużynę i samego zawodnika.

Zdjęcia dodawane przez użytkowników i wykorzystywane w obrębie aplikacji (avatary oraz herby drużyn) nie są przechowywane w bazie danych. Wykorzystano w tym celu narzędzie dostarczane przez firmę Amazon – AWS S3. Jest to serwis, umożliwiający przechowywanie plików (w tym przypadku zdjęć) w chmurze. Amazon dostarcza odpowiednie biblioteki, dzięki czemu integracja API z nośnikiem danych odbywa się bezproblemowo [21]. Zdjęcia wykorzystywane bezpośrednio w obrębie systemu, pochodzą ze strony Adobe Stock Free na licencji standardowej oraz są przechowywane na serwerze, razem z aplikacją.

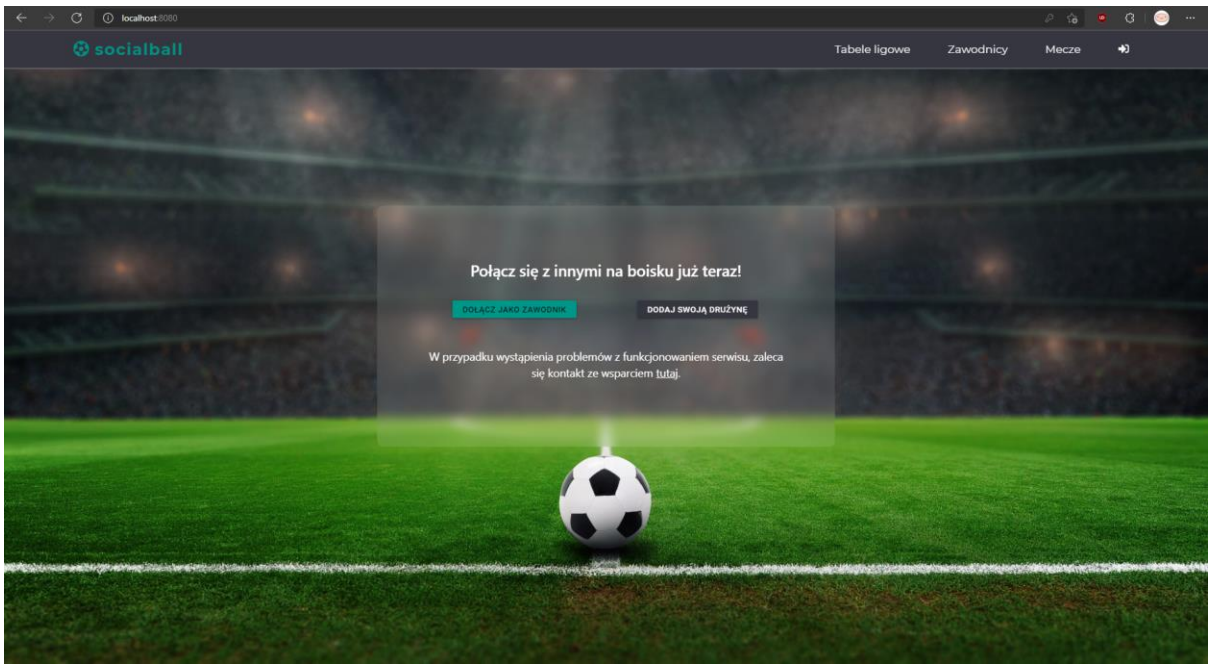


Rys. 4.1. Schemat bazy danych

Źródło: Opracowanie własne

5. Realizacja aplikacji internetowej

Projekt został zrealizowany zgodnie z założeniami. Po wejściu na stronę główną portalu, niezalogowany użytkownik widzi stronę główną.

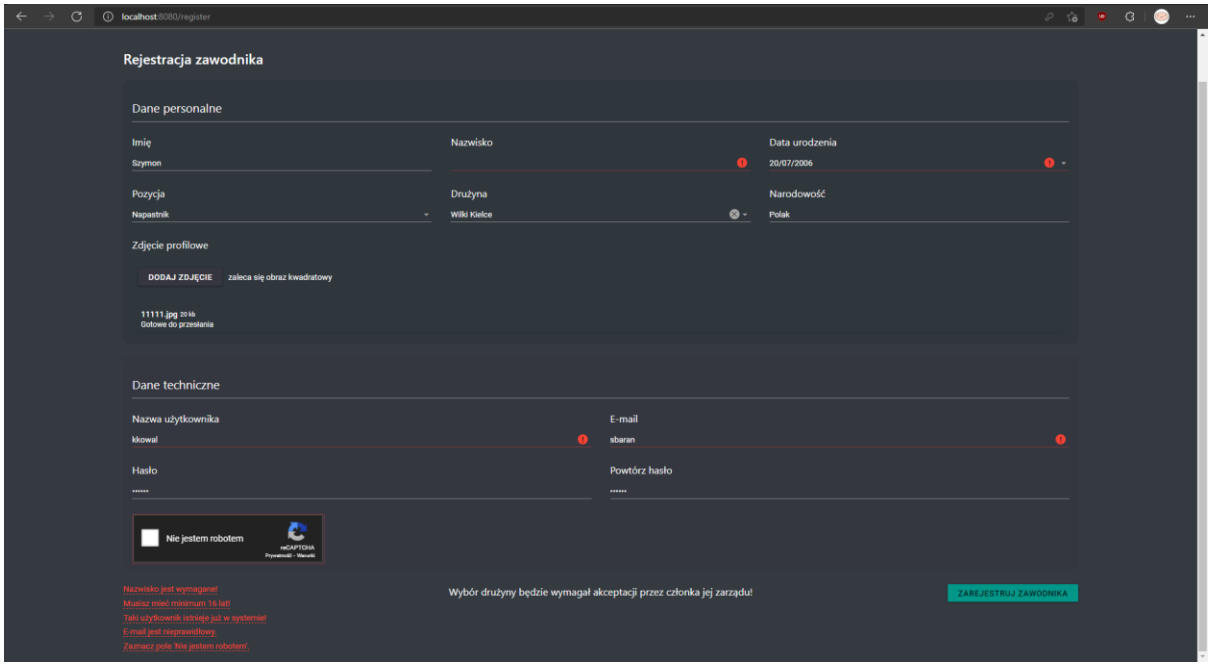


Rys. 5.1. Strona główna

Źródło: Opracowanie własne (tło: Igor Link, stock.adobe.com)

Funkcjonalnością, z jaką większość użytkowników spotka się jako pierwszą jest rejestracja jako zawodnik. Formularz został podzielony na dwie części: dane personalne i techniczne, a część pól podlega walidacji według reguł:

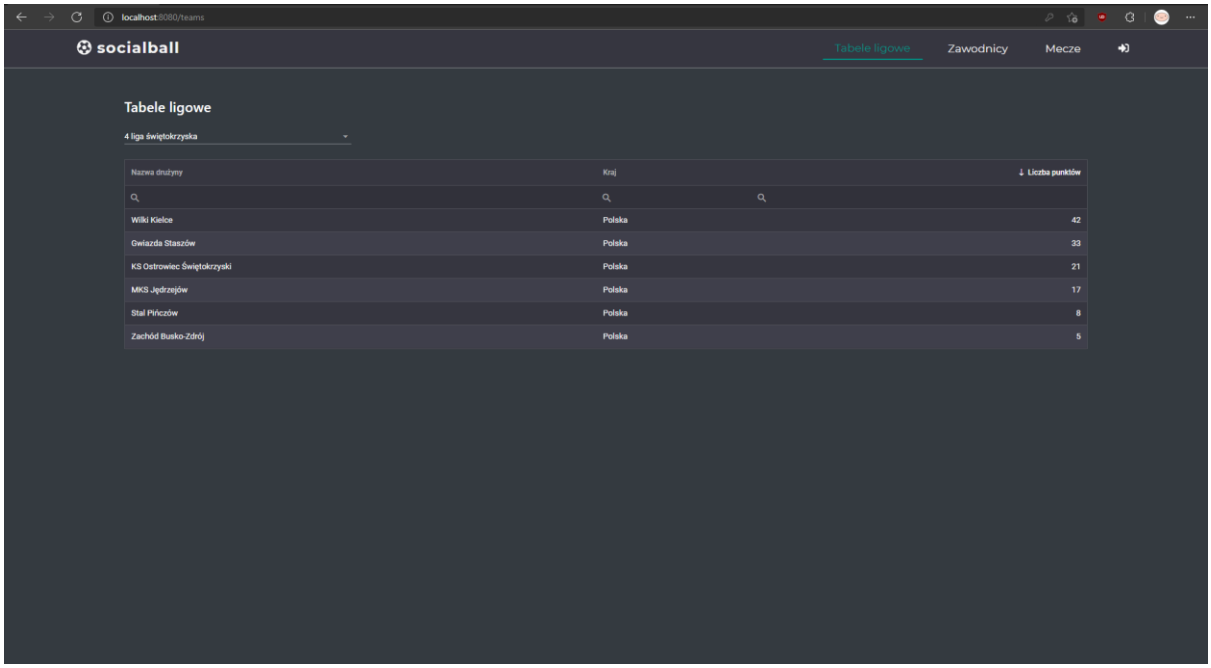
- Niektóre pola (przykładowo „Nazwisko” lub „Nazwa użytkownika”) są wymagane. Jest to najprostszy, a zarazem najczęściej stosowany typ reguły w aplikacji.
- Rejestrowany zawodnik musi mieć co najmniej 16 lat.
- Nazwa użytkownika musi być unikalna w obrębie systemu.
- Adres e-mail musi być prawidłowy.
- Pola z hasłami muszą być takie same.
- Należy zaznaczyć pole „Nie jestem robotem”, które jest rozwiązaniem firmy Google o nazwie reCAPTCHA V2 i chroni stronę przed atakami botów. Jeżeli algorytm uzna to za konieczne, użytkownik będzie musiał rozwiązać krótki test.
- Dodatkową regułą jest wybór drużyny - jeśli użytkownik zaznaczy, że należy do zespołu, to zanim system go do niego przypisze, członek zarządu tej drużyny musi to zatwierdzić.



Rys. 5.2. Walidacja procesu rejestracji

Źródło: Opracowanie własne

Niezalogowany użytkownik ma do wyboru kilka innych opcji – może przejrzeć tabele ligowe (sortowane domyślnie według zdobytej liczby punktów), zobaczyć bazę wszystkich zawodników oraz rozegranych meczy.



Rys. 5.3. Tabele ligowe

Źródło: Opracowanie własne

Dane w tabelach oraz formularzach w obrębie całej aplikacji po stronie użytkownika są przechowywane w tzw. storze, będącym częścią biblioteki Vuex. Vue po wejściu przez użytkownika na widok lub wykonaniu konkretnej akcji (tak jak w przypadku tabel ligowych, po wyborze ligi), wywołuje akcję ze store'a, której zadaniem jest wysłanie zapytania do API, z prośbą o otrzymanie danych. Jest ona wykonywana asynchronicznie, zatem aplikacja czeka, aż otrzyma dane od serwera, w tym czasie wykonując inne polecenia.

Listing 5.1. Akcja, wysyłająca zapytanie do API o przesłanie drużyn występujących w wybranej lidze

```
setTeamsByLeague({ commit }, leagueId) {
  axios
    .get("https://localhost:44369/api/teams/getTeamsByLeague", {
      params: { leagueId: leagueId },
    })
    .then((response) => {
      commit("SET_TEAMS", response.data);
    });
},
```

Źródło: Opracowanie własne

API przesyła przygotowaną odpowiedź do klienta, który po jej otrzymaniu przechodzi do wykonania mutacji na danych w stanie aplikacji.

Listing 5.2. Mutacja, ustawiająca drużyny w stanie store'a, danymi otrzymanymi od serwera

```
SET_TEAMS(state, payload) {
  state.teams = payload;
},
```

Źródło: Opracowanie własne

W tym momencie dane zostały już zapisane, a komponenty, które powinny mieć do nich dostęp, mogą skorzystać z utworzonego w tym celu gettera.

Listing 5.3. Getter, zwracający drużyny ze stanie store'a

```
getTeams(state) {
  return state.teams;
},
```

Źródło: Opracowanie własne

Chcąc uzyskać uprawnienia do zobaczenia pozostałych możliwości w aplikacji, należy się zalogować na wcześniej założone konto. Autentykacja następuje poprzez sprawdzenie kilku warunków w bazie – istnienia użytkownika o podanej kombinacji nazwy oraz hasła, sprawdzeniu, czy konto ma status „aktywne" oraz czy typ konta to nie „System" (konto tego typu służy wyłącznie do wysyłania automatycznych powiadomień).

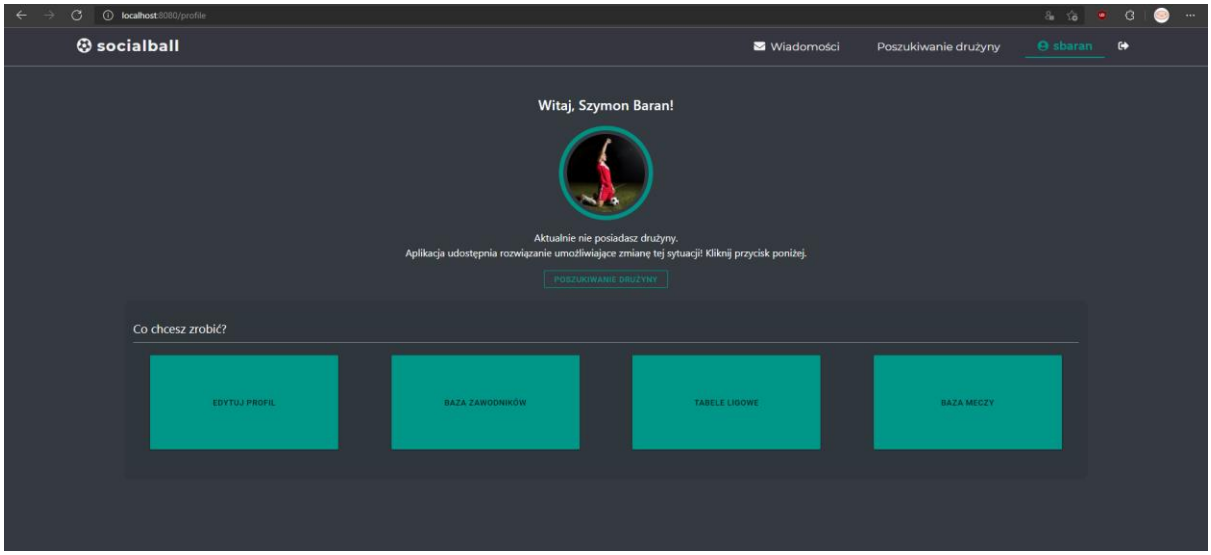
Listing 5.4. Metoda z *UserService*, służąca autentykacji użytkownika

```
public AuthenticateResponse Authenticate(AuthenticateRequest model)
{
    User user = _userRepository.GetUserDetailsByUsername(model.Username);
    if (user == null ||
        !BCrypt.Net.BCrypt.Verify(model.Password, user.Password) ||
        user.UserData.UserType == UserType.System ||
        !user.IsActive) return null;
    var token = generateJwtToken(user);
    return new AuthenticateResponse(user, token);
}
```

Źródło: Opracowanie własne

Po poprawnym przejściu wszystkich warunków, API generuje i zwraca unikalny JWT token, który będzie używany do autoryzacji operacji, a użytkownik zostaje zalogowany. W innym przypadku, zostanie zwrócony komunikat o błędzie, a logowanie się nie powiedzie.

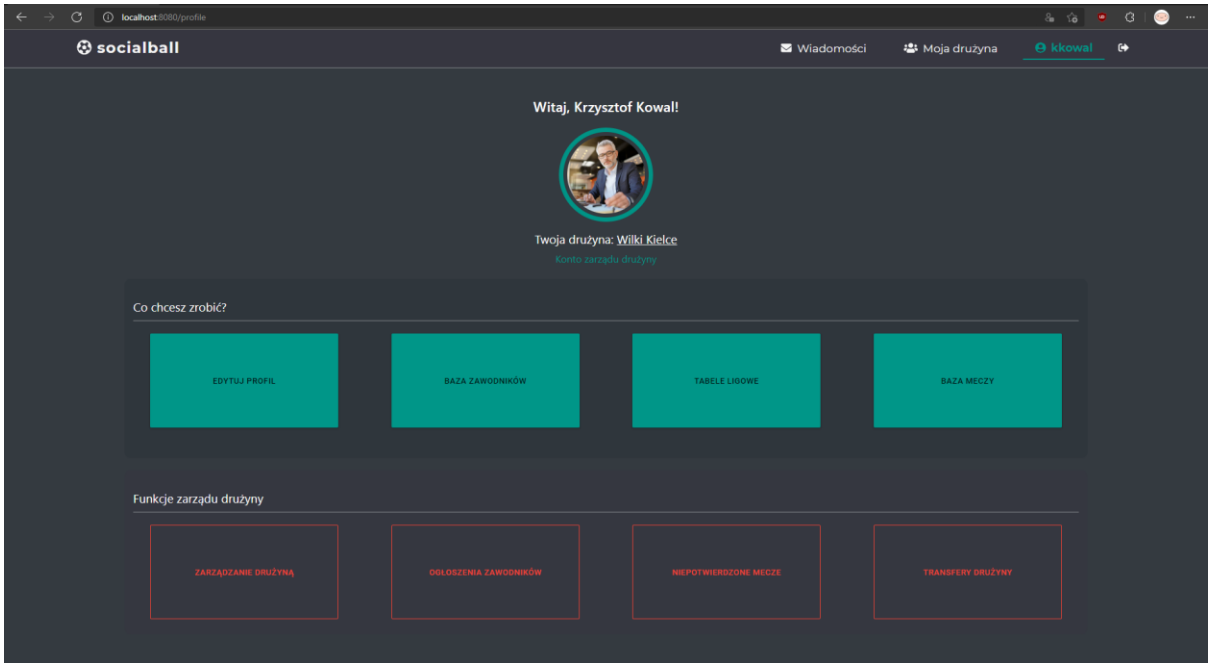
Po zalogowaniu na konto zawodnika, strona główna zostaje zastąpiona panelem użytkownika. Jest to strona prywatna, dostępna wyłącznie dla osoby zalogowanej. Z tego miejsca (oraz górnego paska nawigacyjnego, który również uległ zmianie po zalogowaniu), użytkownik ma możliwość szybkiego przejścia do różnych funkcjonalności aplikacji. Niezależnie od roli, każda zalogowana osoba ma możliwość edycji swojego profilu, gdzie może wprowadzić poprawki do danych podanych przy rejestracji oraz zmienić zdjęcie profilowe. Niektóre dane (przykładowo nazwa użytkownika) nie mogą zostać zmienione. Jeżeli zawodnik posiada drużynę, to widnieje tutaj również informacja o zarobkach za ten miesiąc, ustalana przez jej zarząd. Aktualnie zalogowany zawodnik nie posiada zespołu, więc widzi komunikat, informujący o możliwości skorzystania z narzędzia poszukiwania drużyny.



Rys. 5.4. Panel zawodnika

Źródło: Opracowanie własne

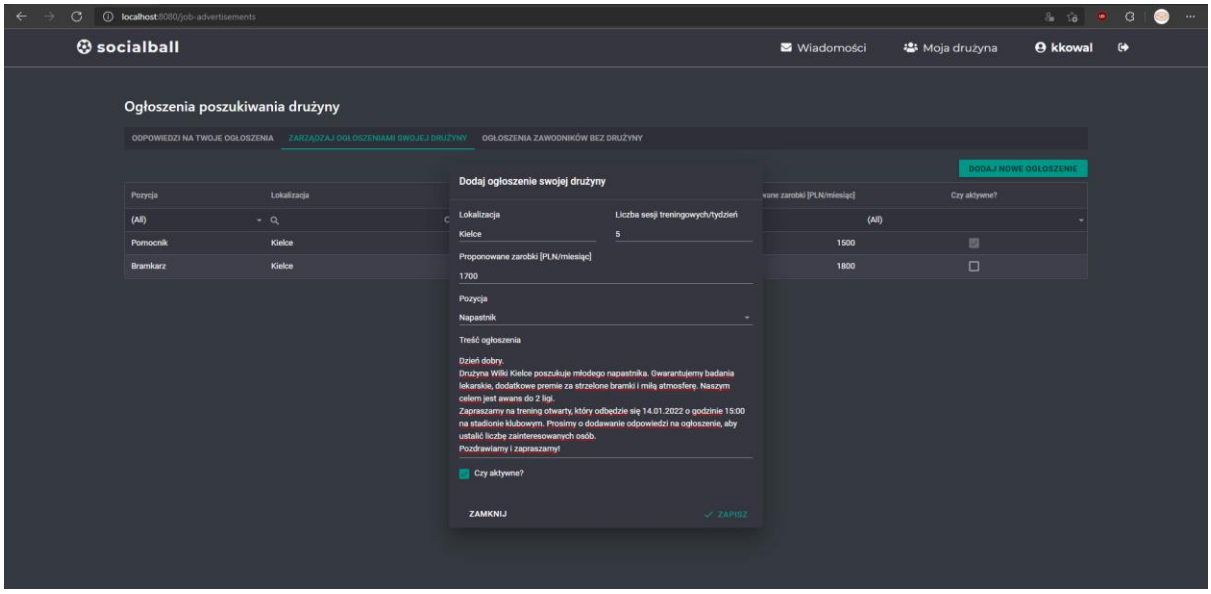
W zależności od typu konta widok panelu może się różnić. Panel po zalogowaniu na konto zarządu drużyny prezentuje się tak jak na rysunku 5.5. Posiada dodatkową sekcję ze specjalnymi funkcjonalnościami, umożliwiającymi wprowadzanie zmian w zarządzanym zespole.



Rys. 5.5. Panel zarządcy drużyny

Źródło: Opracowanie własne

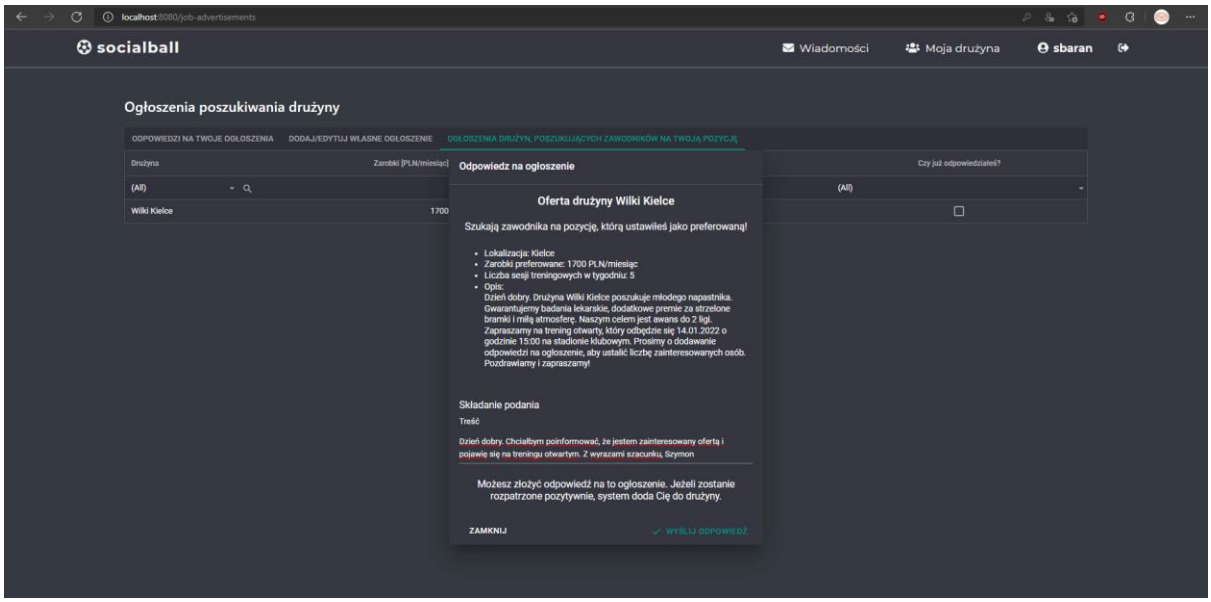
Funkcjonalność poszukiwania drużyny może zostać wykorzystana na dwa sposoby. Interakcja może rozpocząć się od strony zawodnika, który utworzy ogłoszenie, w którym będzie oferował swoje usługi lub od strony drużyny, która będzie chciała zaoferować pracę. Zawodnik może mieć w danym momencie aktywne jedno ogłoszenie, natomiast drużyna nieskończoną ich ilość.



Rys. 5.6. Dodawanie nowego ogłoszenia drużyny

Źródło: Opracowanie własne

Po zalogowaniu na konto nowoutworzonego zawodnika, widzi on ogłoszenie, ponieważ pasuje ono do roli, którą wybrał w procesie rejestracji.



Rys. 5.7. Odpowiedź zawodnika na ogłoszenie drużyny

Źródło: Opracowanie własne

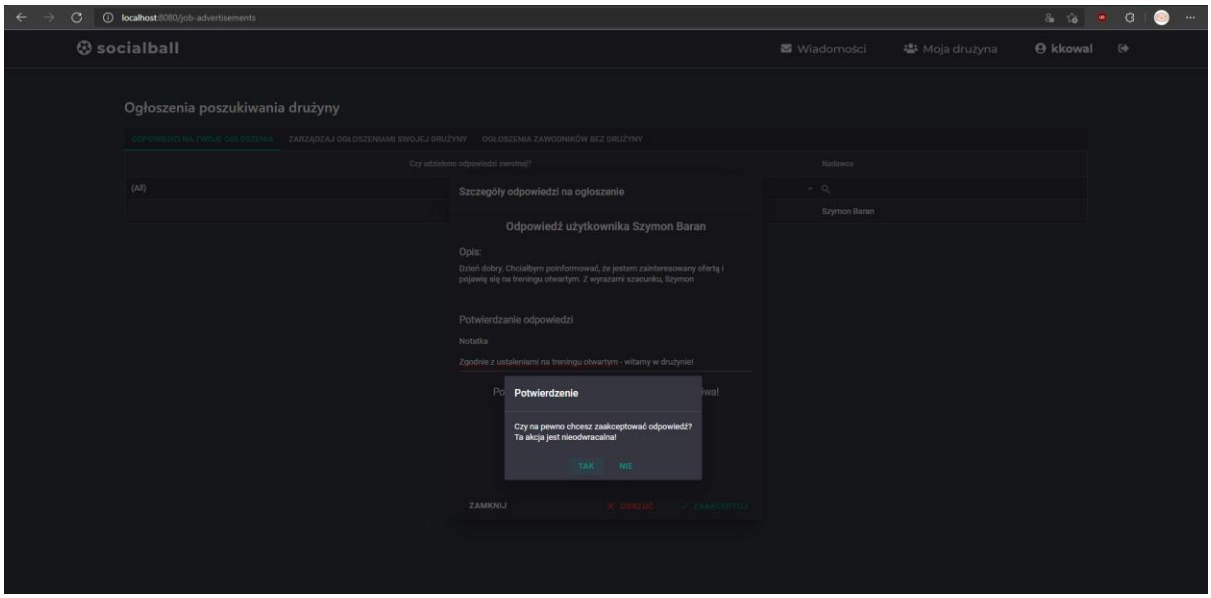
Wprowadzone zostało zabezpieczenie, uniemożliwiające złożenie dwukrotnej odpowiedzi na to samo ogłoszenie, jeżeli druga strona się w żaden sposób do niej nie ustosunkowała.

Listing 5.5. Metoda w *JobAdvertisementService* służąca do pobierania ogłoszeń pasujących dla zawodnika i przypisująca odpowiednią wartość do właściwości *IsAlreadyAnswered*

```
public object GetUserJobAdvertisements(Guid userId)
{
    Player player = _playerRepository.GetPlayerDetailsByUserId(userId);
    return  _jobAdvertisementRepository.GetFromTeamJobAdvertisementsByPosition
    ((PositionType)player.Position).Select(x => new
    {
        x.Id,
        x.TeamId,
        x.Earnings,
        x.TrainingSessionsPerWeek,
        x.Location,
        x.JobAdvertisementType,
        IsAlreadyAnswered = x.JobAdvertisementAnswers
        .Any(y      =>      y      is      JobAdvertisementUserAnswer      ?
        ((JobAdvertisementUserAnswer)y).UserId == userId && y.IsResponded ==
        false : false)
    }).ToList();
}
```

Źródło: Opracowanie własne

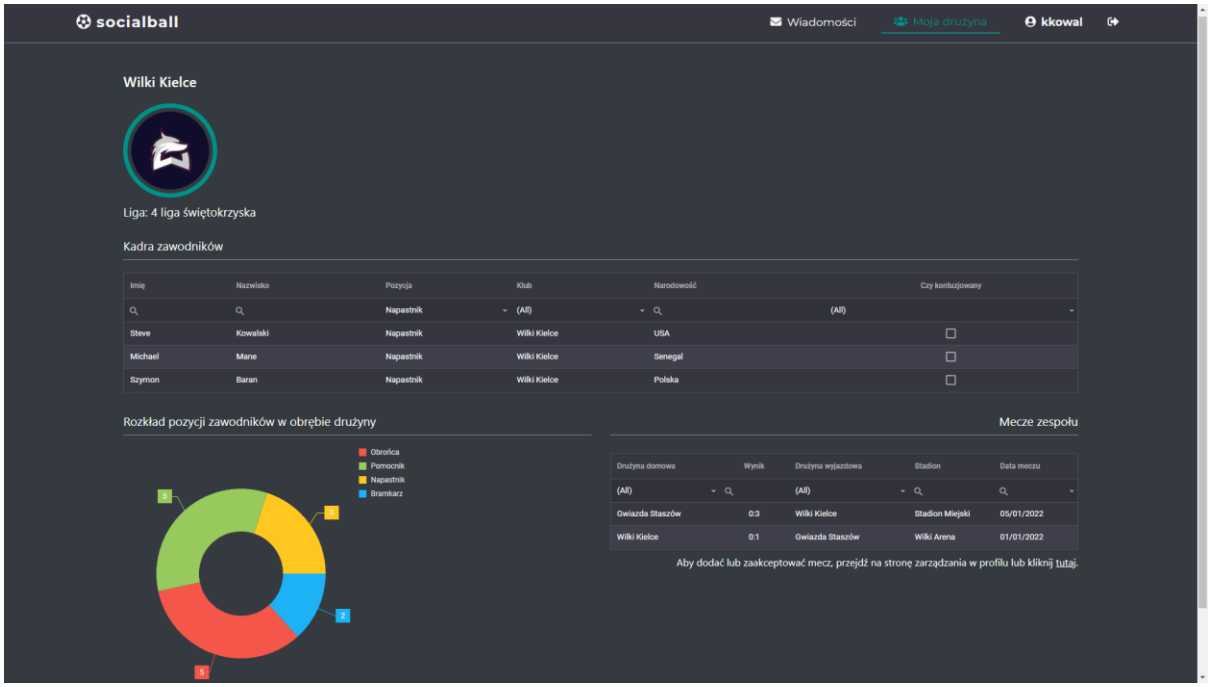
Ostatni krok wymaga dodatkowego potwierdzenia, gdyż w przypadku odpowiedzi pozytywnej, system doda zawodnika do drużyny.



Rys. 5.8. Potwierdzenie dodania zawodnika do drużyny

Źródło: Opracowanie własne

Po przejściu na profil drużyny, zawodnik powinien od razu być widoczny w kadrze oraz zostać uwzględniony w wykresie prezentującym rozkład pozycji zawodników. Z tego miejsca, użytkownik może przejść na profil każdego zawodnika oraz zobaczyć szczegóły każdego meczu rozegranego przez drużynę.



Rys. 5.9. Profil drużyny

Źródło: Opracowanie własne

Listing 5.6. Metoda z *TeamService*, której zadaniem jest zwrócenie danych do umieszczenia w wykresie na Rys. 5.9.

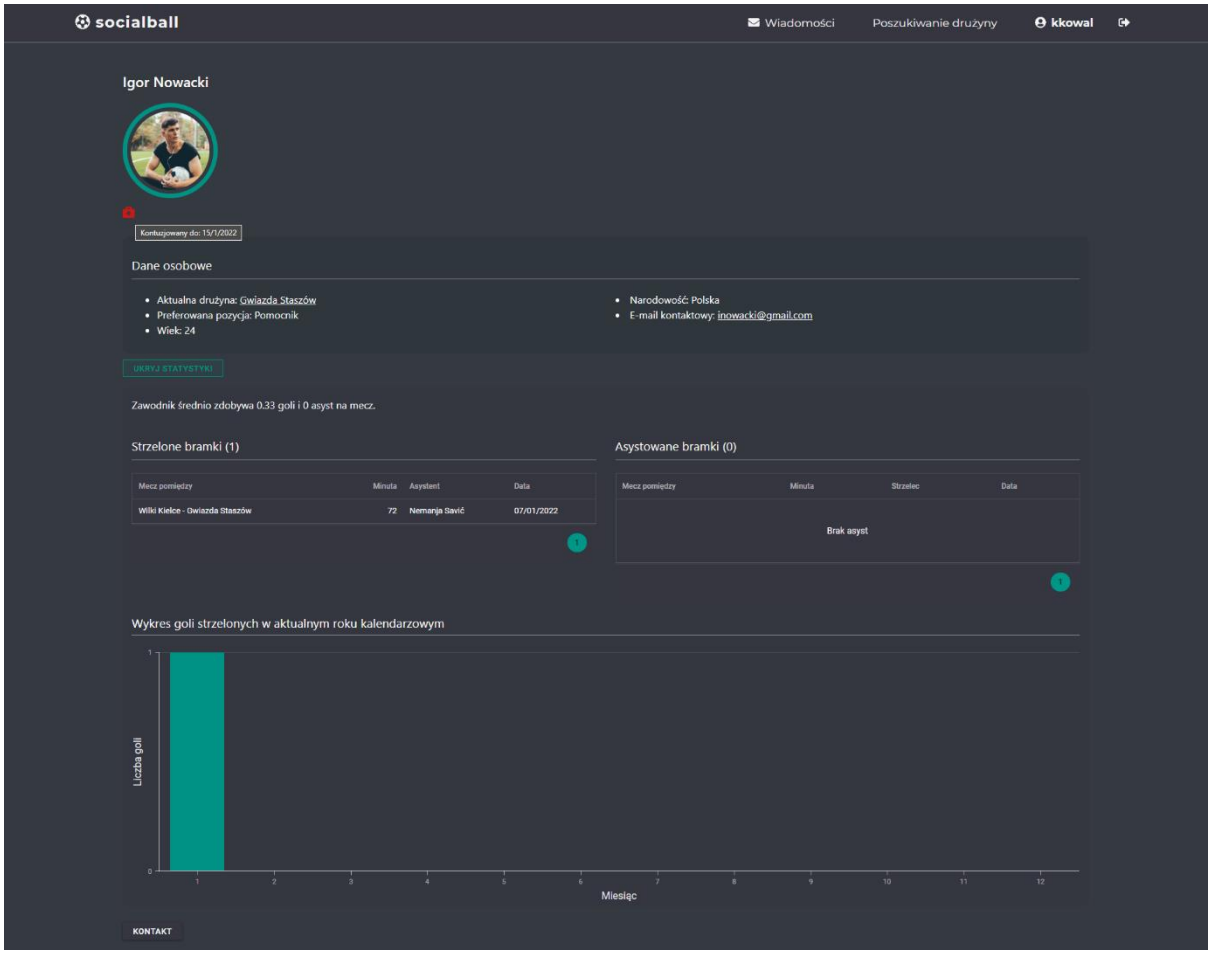
```
public List<PositionsInTeam> GetPositionsCountToChart(Guid teamId)
{
    List<PositionsInTeam> positionsInTeams = new List<PositionsInTeam>();

    foreach (PositionType p in Enum.GetValues(typeof(PositionType)))
    {
        positionsInTeams.Add(new PositionsInTeam
        {
            Position = p,
            NumberOfPlayers = _playerRepository.GetPlayersInTeam(teamId)
            .Where(x => x.Position == (int)p)
            .Count()
        });
    }

    return positionsInTeams;
}
```

Źródło: Opracowanie własne

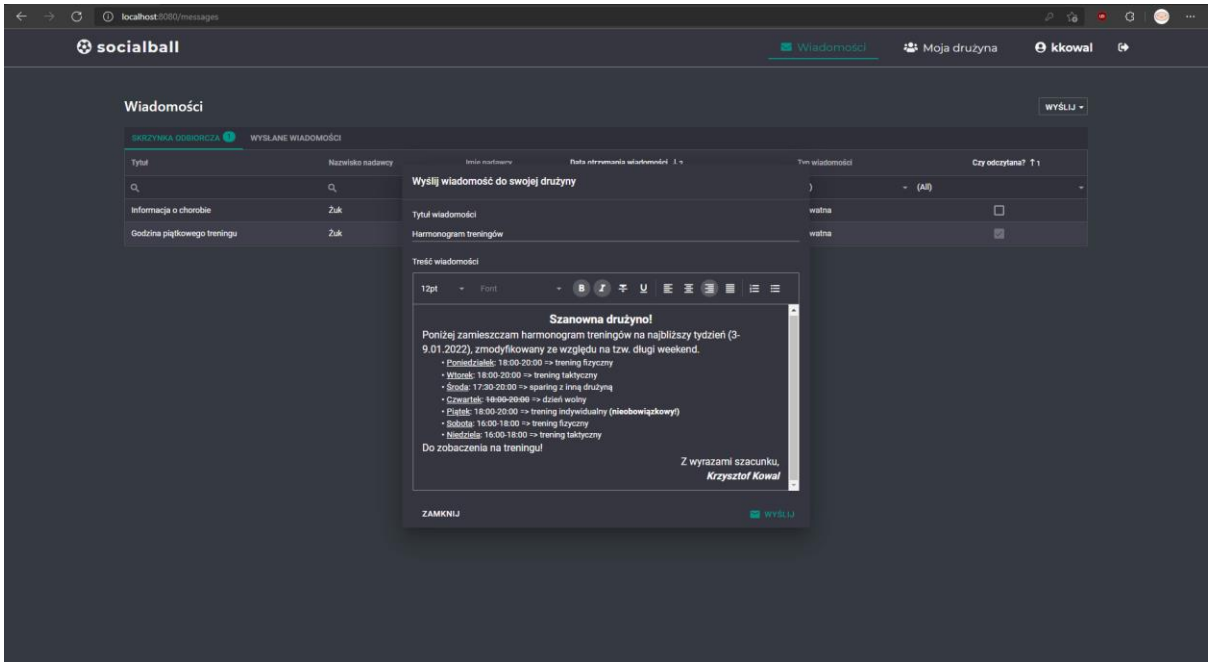
Każdy z zawodników posiada swój profil publiczny, na którym znajdują się jego dane osobowe oraz statystyki. Znajduje się tam również możliwość szybkiego kontaktu, poprzez kliknięcie odpowiedniego przycisku. W przypadku kontuzji zawodnika, wyświetlana jest ikona apteczki. Gdy użytkownik na nią najedzie, otrzyma informacje o planowanej dacie powrotu po urazie.



Rys. 5.10. Profil zawodnika

Źródło: Opracowanie własne

Funkcjonalnością aplikacji, do której dostęp posiada każdy zalogowany użytkownik są wiadomości. Częścią wspólną są dwie zakładki – skrzynka odbiorcza oraz wiadomości wysłane. Wiadomości są domyślnie sortowane według zasady „najpierw nieprzeczytane”, w drugiej kolejności od najnowszych. Możliwe typy wysyłki są różne, w zależności od typu konta. Zawodnik może wysyłać wyłącznie wiadomości prywatne, podczas gdy członek zarządu ma dodatkowo możliwość nawiązania kontaktu ze wszystkimi członkami swojej drużyny jednocześnie (formularz różni się wtedy brakiem opcji wyboru odbiorcy) lub z zarządem innego zespołu. Pole z treścią wiadomości posiada wbudowany edytor HTML, umożliwiający formatowanie tekstu oraz tworzenie list.



Rys. 5.11. Formularz wysyłania wiadomości do wszystkich członków drużyny

Źródło: Opracowanie własne

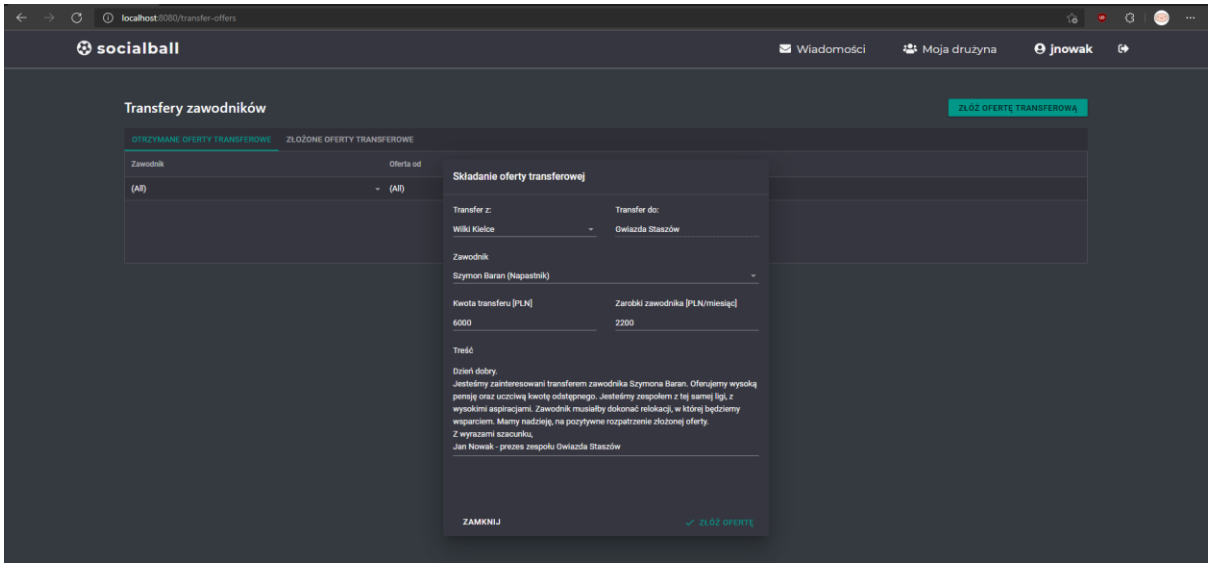
W momencie pierwszego otworzenia wiadomości, zostaje ona oznaczona jako *!IsRead*, natomiast po usunięciu jako *!IsActive* i przestaje być brana pod uwagę przy uzupełnianiu skrzynki odbiorczej.

Listing 5.7. Metoda z *MessageRepository*, pobierająca nieusunięte wiadomości użytkownika.

```
public object GetUserMessages(Guid userId)
{
    return _context.UserMessages
        .Include(x => x.Message)
        .ThenInclude(x => x.FromUser)
        .ThenInclude(x => x.UserData)
        .Where(x => x.IsActive && x.ToUserId == userId).ToList();
}
```

Źródło: Opracowanie własne

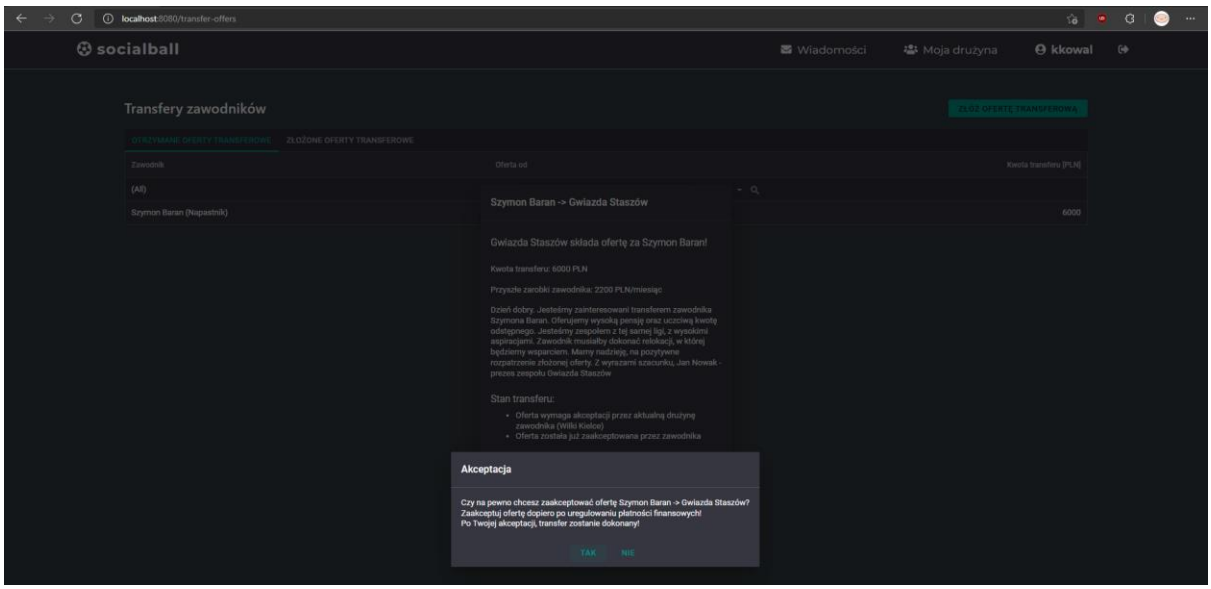
Funkcja transferu zawodników jest dostępna wyłącznie dla członków zarządu. Po złożeniu oferty, musi ona zostać zaakceptowana zarówno przez zawodnika, jak i przez jego aktualny zespół. Każdy zawodnik może uzyskać dostęp do ofert, które dotyczą jego osoby, poprzez zakładkę na liście wiadomości. W przypadku odrzucenia przez którąkolwiek ze stron, oferta zostaje anulowana, a zespół, który ją złożył otrzyma powiadomienie systemowe.



Rys. 5.12. Formularz składania oferty transferowej za zawodnika

Źródło: Opracowanie własne

Jeżeli oferta została już zaakceptowana przez jedną ze stron, to konieczne jest potwierdzenie dodatkowego komunikatu, że ustalone płatności zostały uregulowane.

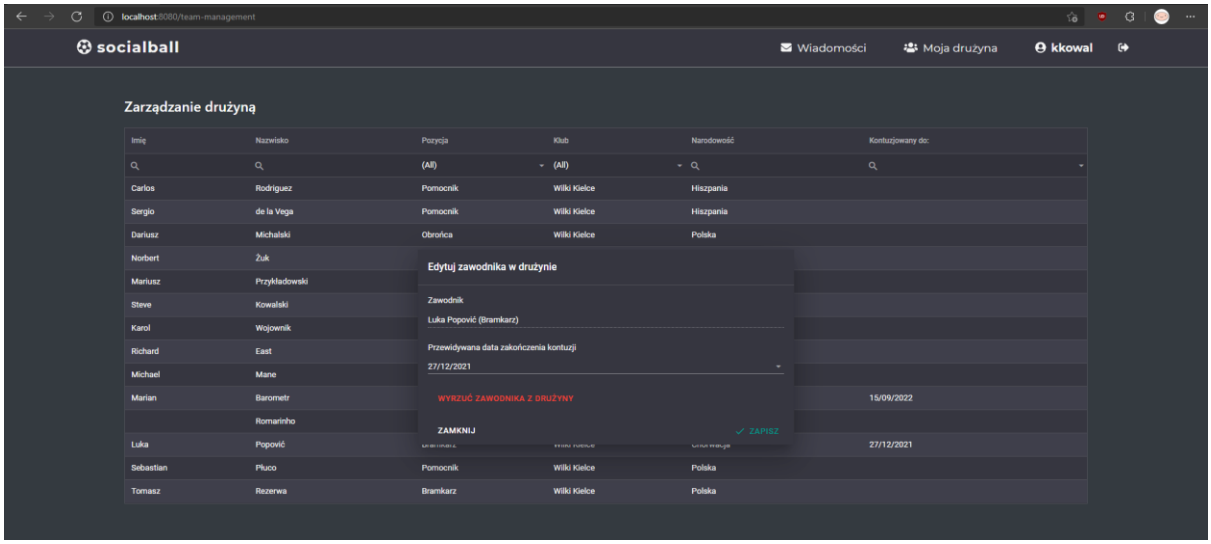


Rys. 5.13. Akceptacja oferty transferowej

Źródło: Opracowanie własne

Po dokonaniu akceptacji, zawodnik zostaje przeniesiony do nowego zespołu, a jego pensja ulega zmianie, zgodnie z ustaleniami w ofercie transferowej.

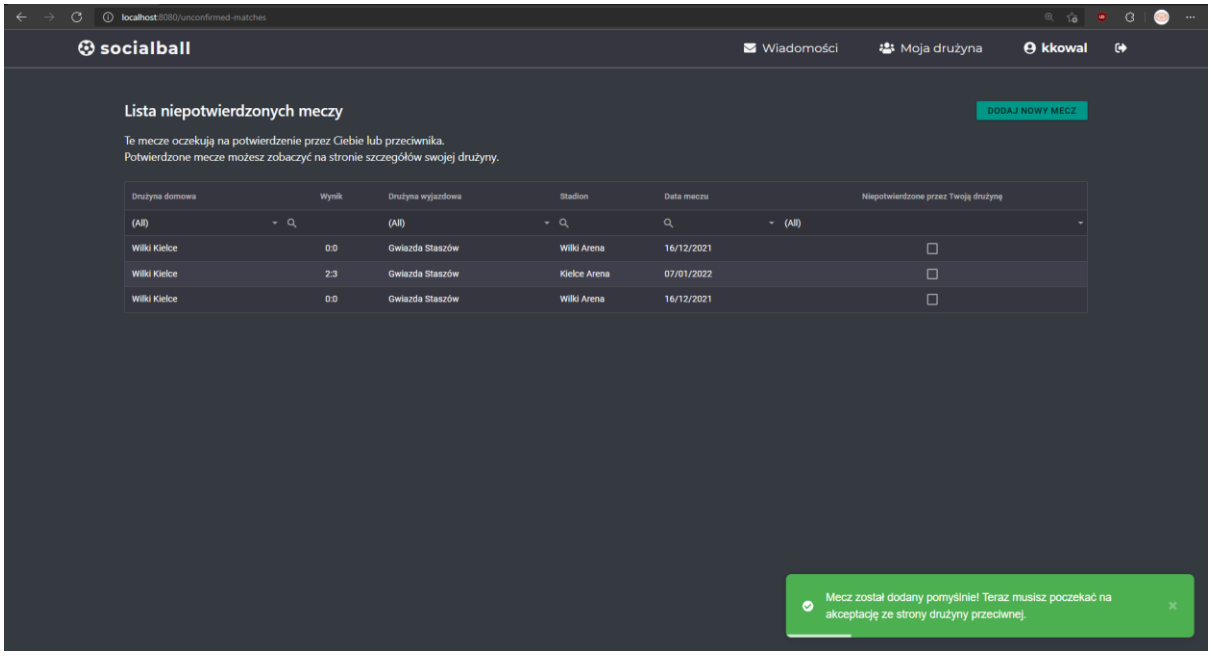
Członek zarządu ma możliwość edycji kontuzji zawodnika w zespole, a także usunięcia go z drużyny w systemie.



Rys. 5.14. Zarządzanie zawodnikiem w drużynie

Źródło: Opracowanie własne

Ostatnim nieomówionym widokiem, dostępnym dla członków zarządu jest dodawanie meczy. Dodany przebieg spotkania musi zostać zaakceptowany przez drugi zespół, aby był potwierdzony i wyświetlany przez aplikację.

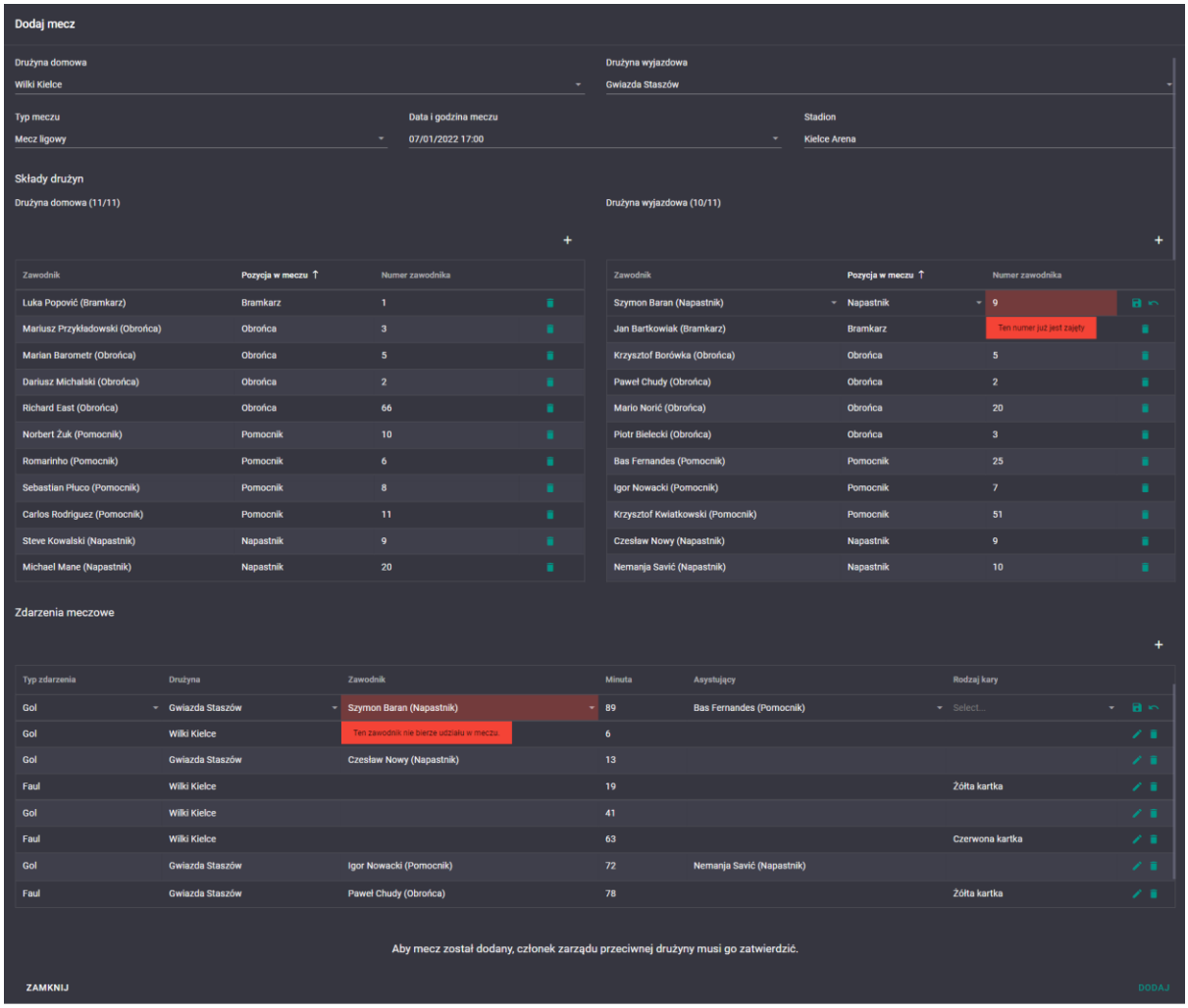


Rys. 5.15. Lista niepotwierdzonych meczy wraz z komunikatem potwierdzającym dodanie

Źródło: Opracowanie własne

Formularz dodawania nowego spotkania, musi przejść kilka reguł walidacyjnych, zanim zostanie zatwierdzony przez aplikację:

- Mecz musi posiadać dwie drużyny, wybrany typ, datę i godzinę oraz stadion.
- Jedną z drużyn biorących udział w spotkaniu, musi być zespół zalogowanego członka zarządu.
- Każda z drużyn musi wystawić po 11 zawodników, w tym 1 bramkarza.
- Każdy zawodnik musi mieć unikalny numer (może nie posiadać go wcale, ale nie mogą się powtarzać).
- Udział w zdarzeniu meczowym, może wziąć wyłącznie zawodnik, który gra w meczu.

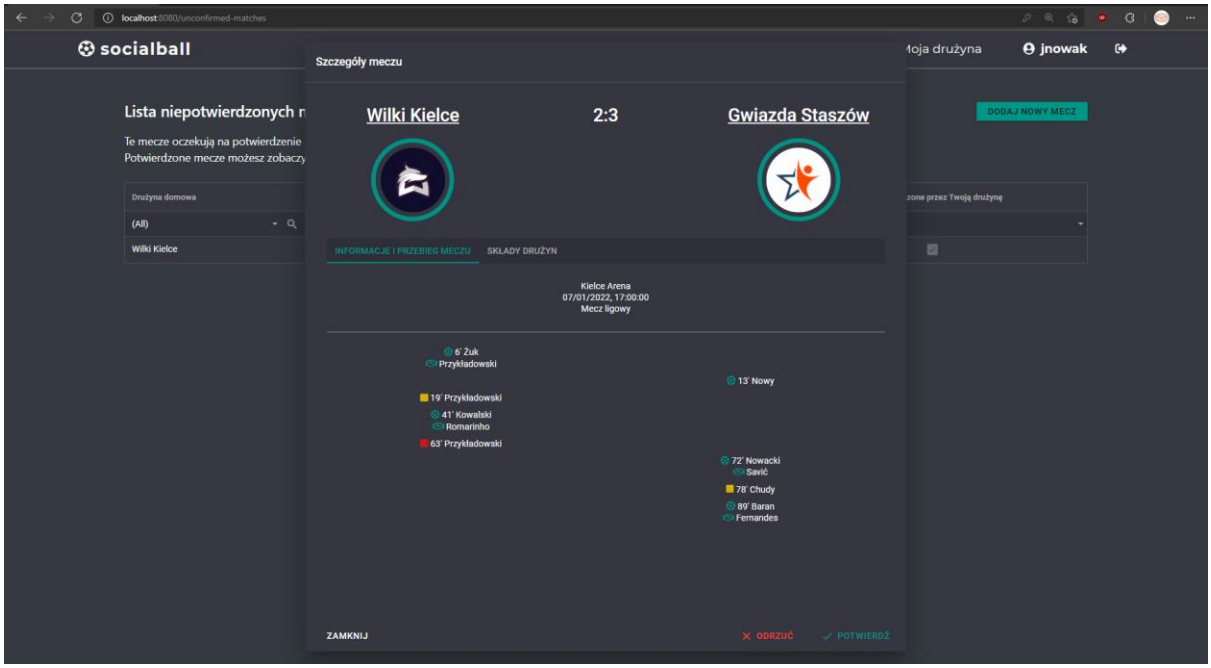


Rys. 5.16. Formularz dodawania meczu oraz jego reguły walidacyjne

Źródło: Opracowanie własne

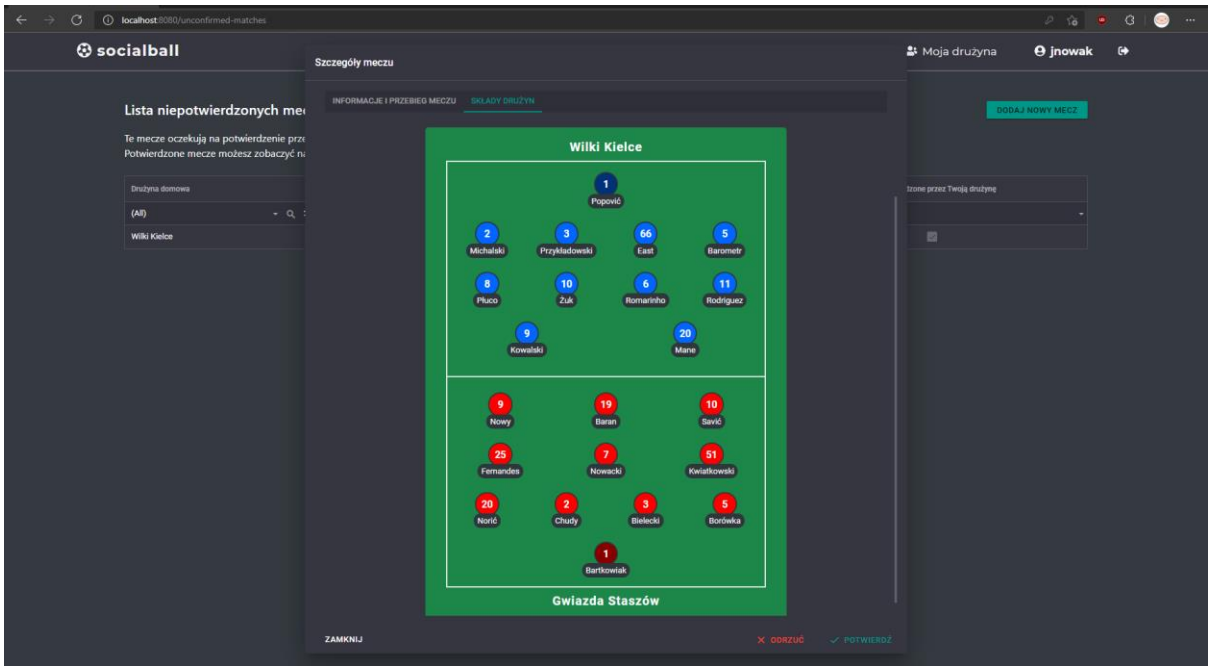
Po dodaniu meczu, nie jest on jeszcze wyświetlany w bazie meczy, ani na profilu drużyny. Widzi go jedynie członek zarządu drużyny przeciwnej, gdy wejdzie na widok „Niepotwierdzone mecze”. Może tutaj przejrzeć informacje oraz przebieg spotkania, a także

składy obu zespołów i zadecydować, czy są one zgodne z prawdą, czy jednak chce je odrzucić – jak na rysunkach 5.17 oraz 5.18.



Rys. 5.17. Informacje i przebieg meczu

Źródło: Opracowanie własne



Rys. 5.18. Składy drużyn w meczu

Źródło: Opracowanie własne

Jeżeli zarządca zespołu zdecyduje się na akceptację przebiegu, mecz zostanie oficjalnie dodany i pojawi się również w innych miejscach aplikacji. Dodatkowo,

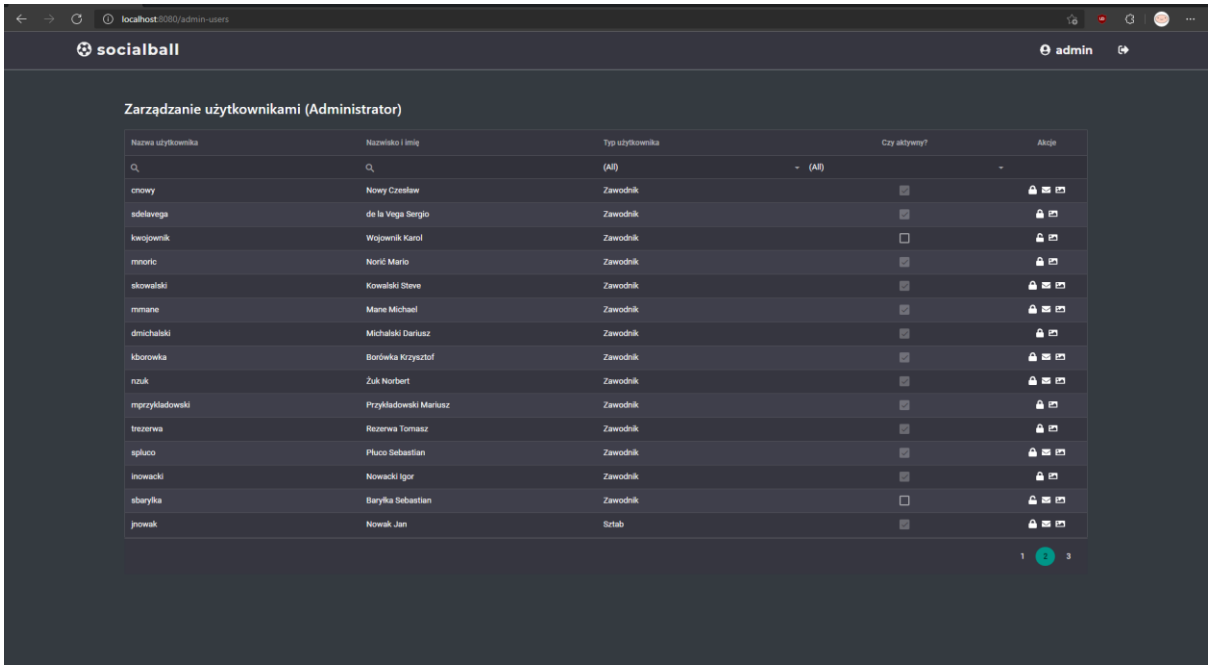
w przypadku spotkania ligowego, odpowiednia liczba punktów trafi na konto drużyny zwycięskiej. Jeśli jednak uzna, że coś się nie zgadza i odrzuci ten wpis, system wyśle odpowiednie powiadomienie twórcy meczu, który zostanie usunięty.

Ostatni typ konta dostępny w aplikacji to konto administracyjne. Jest unikalne w skali systemu i nie ma możliwości rejestracji.

Rys. 5.19. Panel administracyjny

Źródło: Opracowanie własne

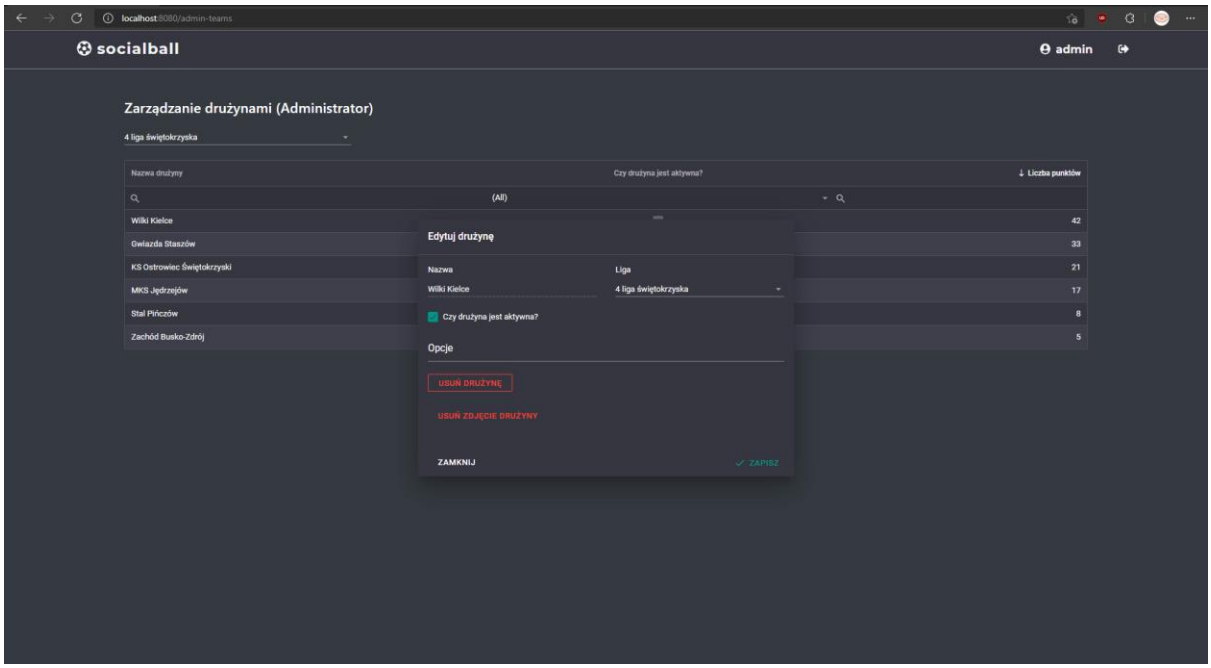
Na widoku zarządzania użytkownikami, administrator ma prawo zablokowania konta użytkownika, jeżeli ten złamał zasady korzystania z serwisu. Może także usunąć jego zdjęcie profilowe. Posiada również odnośnik do kontaktu mailowego, jeżeli użytkownik posiada uzupełniony adres e-mail w profilu.



Rys. 5.20. Administracyjne zarządzanie użytkownikami

Źródło: Opracowanie własne

Widok zarządzania drużynami prezentuje się analogicznie. Dodatkową możliwością jest zmiana ligi, w której występuje zespół.



Rys. 5.21. Administracyjne zarządzanie drużynami

Źródło: Opracowanie własne

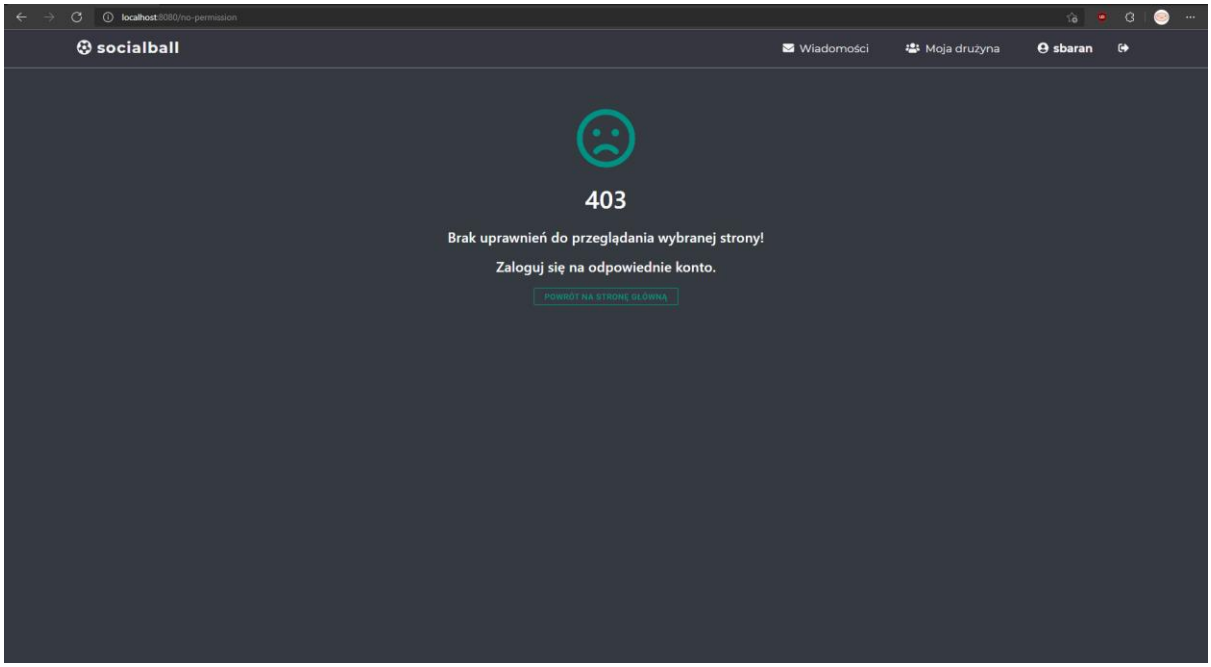
6. Testy aplikacji

Proces weryfikacji działania ukończonej aplikacji jest niemal tak samo istotny jak sam proces jej tworzenia. Jako rodzaj testów, wybrano testy manualne, na podstawie uprzednio utworzonych scenariuszy z opisanymi kolejno krokami do odtworzenia sytuacji.

Test 1.1. Sprawdzanie uprawnień dostępu do aplikacji do strony członka zarządu klubu
Kroki do wykonania testu:

1. Zalogować się na konto zawodnika.
2. Wpisać w przeglądarce URL do strony dostępnej wyłącznie dla członków zarządu (przykładowo /unconfirmed-matches).

Oczekiwany rezultat:
Przeglądarka powinna przekierować użytkownika na stronę sygnalizującą brak uprawnień (/no-permission).
Otrzymany rezultat:
Nastąpiło przekierowanie na stronę, informującą użytkownika o braku uprawnień.



Rys. 5.22. Wynik testu wejścia na stronę przy braku uprawnień
Źródło: Opracowanie własne

Test 1.2. Sprawdzanie uprawnień dostępu do aplikacji do strony administratora

Kroki do wykonania testu:

1. Zalogować się na konto lub członka zarządu klubowego.
2. Wpisać w przeglądarce URL do strony dostępnej wyłącznie dla administratora (przykładowo /admin).

Oczekiwany rezultat:

Przeglądarka powinna przekierować użytkownika na stronę sygnalizującą brak uprawnień (/no-permission).

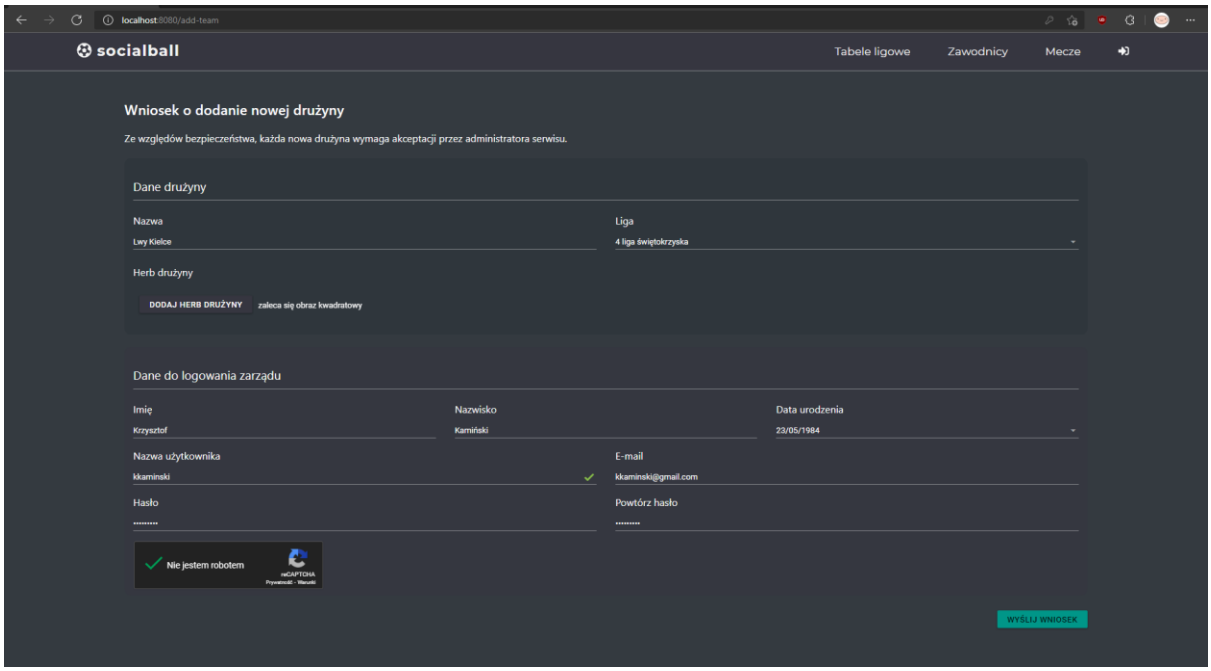
Otrzymany rezultat:

Nastąpiło przekierowanie na stronę, informującą użytkownika o braku uprawnień.

Test 2. Sprawdzenie konieczności akceptacji nowej drużyny przez administratora

Kroki do wykonania testu:

1. Wypełnić poprawnie formularz dodawania nowej drużyny.



Rys. 5.23. Formularz wniosku o dodanie nowej drużyny

Źródło: Opracowanie własne

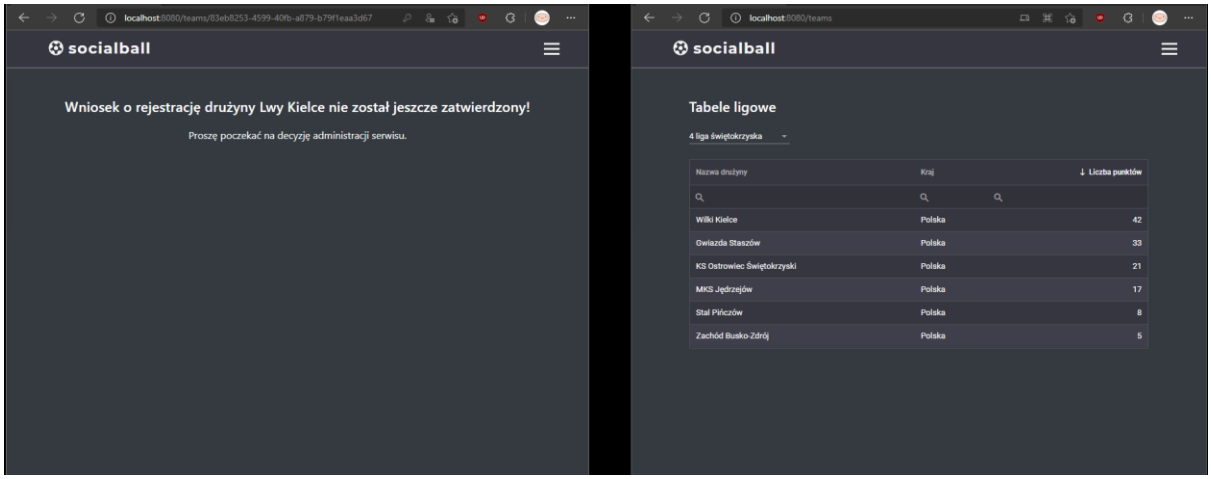
2. Zalogować się na nowoutworzone konto zarządu

Oczekiwany rezultat:

Logowanie powinno przebiec pomyślnie, jednak profil drużyny powinien wyświetlać komunikat o braku zatwierdzenia przez administrację, a drużyna nie powinna być wyświetlana w tabeli ligowej.

Otrzymany rezultat:

Funkcjonalność działa zgodnie z założeniem.



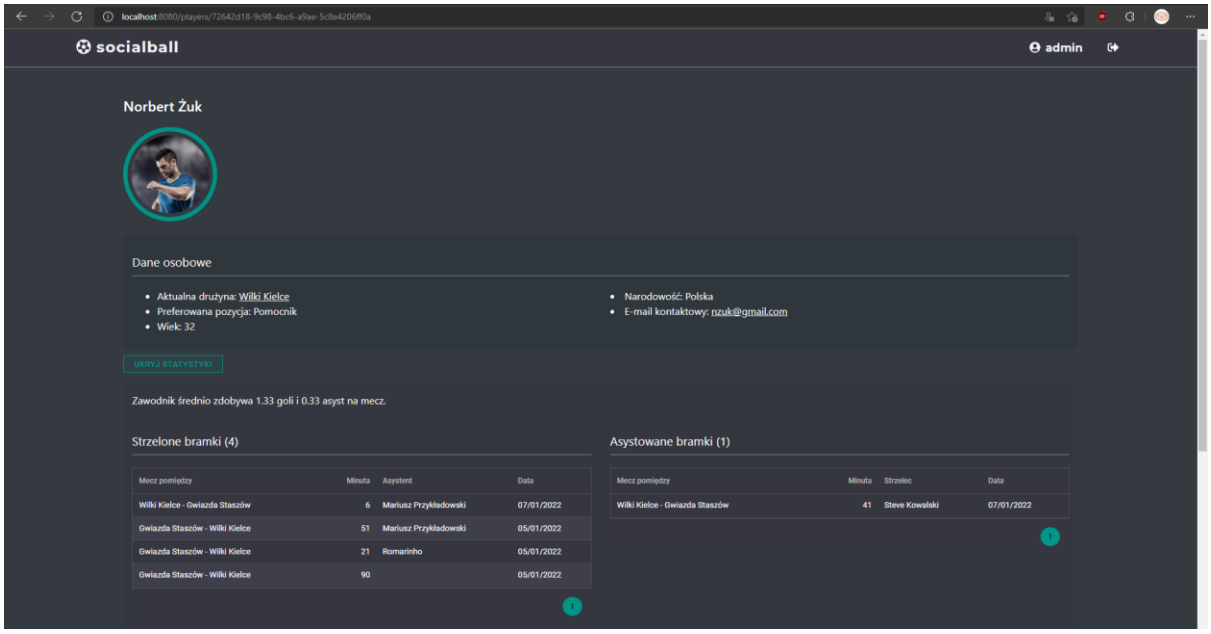
Rys. 5.24. Wynik testu rejestracji drużyny przy braku akceptacji administratora

Źródło: Opracowanie własne

Test 3.1. Sprawdzenie poprawności wyliczania średniej bramek i asyst zawodnika

Kroki do wykonania testu:

1. Przejść na profil zawodnika.
2. Sprawdzić wyliczoną przez system średnią liczbę zdobytych bramek i asyst.



Rys. 5.25. Profil zawodnika ze statystykami

Źródło: Opracowanie własne

3. Wyliczyć średnie wartości ręcznie i porównać z tymi w profilu.

Oczekiwany rezultat:

Wartości wyliczone przez aplikację oraz ręcznie powinny być identyczne.

Otrzymany rezultat:

Zawodnik zagrał w sumie w 3 meczach. W pierwszym nie brał udziału przy żadnej z bramek, w drugim strzelił 3 gole, natomiast w trzecim zdobył gola i zaliczył asystę, zatem:

$\frac{4}{3} = 1\frac{1}{3} \approx 1.33\ oraz \frac{1}{3} \approx 0.33$, więc wartości wyliczone przez aplikację są poprawne.

Test 3.2. Sprawdzenie poprawności utworzenia wykresu rozkładu pozycji zawodników w obrębie drużyny.

Kroki do wykonania testu:

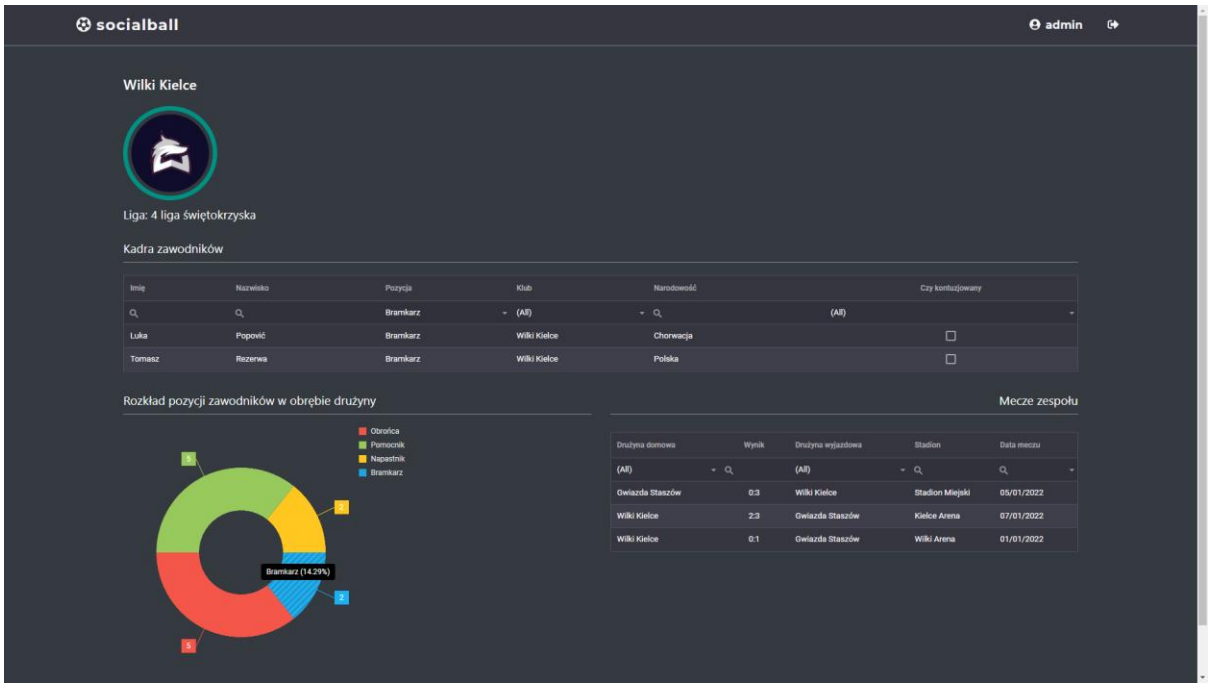
1. Przejść na profil jednej z drużyn.
2. Porównać kolejno każdy z argumentów (pozycji) znajdujących się na wykresie z danymi w tabeli „kadra zawodników", korzystając z opcji filtrowania według pozycji.

Oczekiwany rezultat:

Dane z wykresu powinny się zgadzać z tymi w tabeli.

Otrzymany rezultat:

Dane z wykresu są zgodne z danymi w tabeli.



Rys. 5.26. Wynik testu porównania liczby bramkarzy na wykresie z liczbą w tabeli

Źródło: Opracowanie własne

Test 4. Przetestowanie funkcjonowania powiadomień systemowych na przykładzie odrzucenia transferu zawodnika

Kroki do wykonania testu:

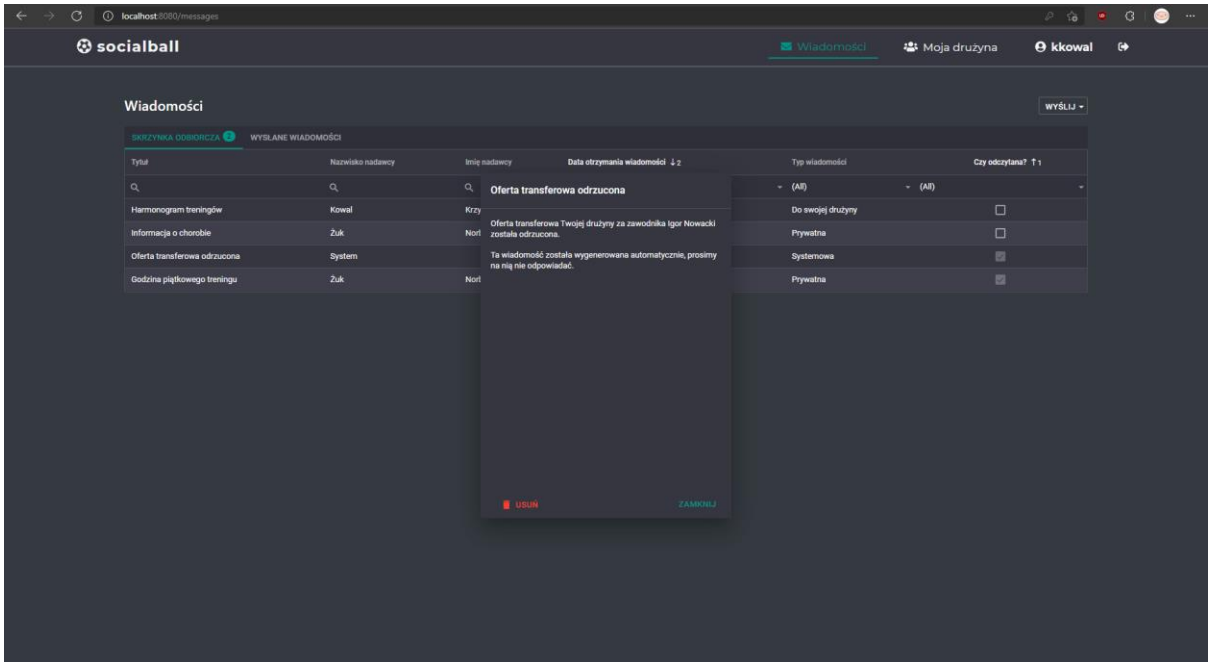
1. Zalogować się na konto zarządu drużyny.
2. Wybrać z panelu opcję „transfery drużyny” oraz nacisnąć przycisk „Złóż ofertę transferową”.
3. Wypełnić formularz i zatwierdzić go.
4. Zmienić konto na zarząd drużyny, do której aktualnie należy zawodnik.
5. Ponownie przejść do widoku transferów.
6. Wybrać otrzymaną ofertę i odrzucić ją.
7. Wrócić na pierwsze konto i sprawdzić skrzynkę odbiorczą wiadomości.

Oczekiwany rezultat:

Użytkownik powinien otrzymać powiadomienie systemowe, o fakcie, że jego oferta została odrzucona.

Otrzymany rezultat:

Otrzymano oczekiwaną wiadomość o specjalnym typie „Systemowa”.



Rys. 5.27. Wynik testu funkcjonowania powiadomień systemowych

Źródło: Opracowanie własne

Test 5. Sprawdzenie poprawności funkcjonowania mechanizmu wysyłania prośby o dodanie w przypadku wyboru drużyny podczas rejestracji zawodnika

Kroki do wykonania testu:

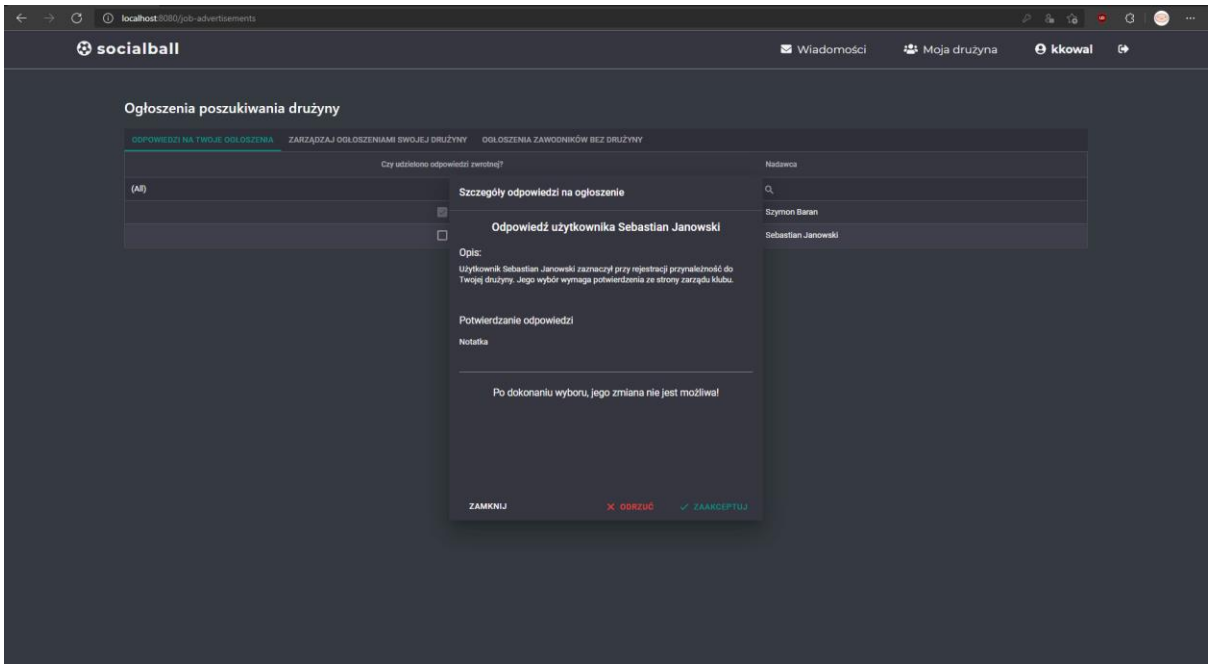
1. Uzupełnić i przesłać formularz rejestracji wraz z wyborem drużyny.
2. Sprawdzić, czy zawodnik zgodnie z oczekiwaniami jeszcze nie został przypisany do zespołu.
3. Zalogować się na konto zarządcy wybranej drużyny.
4. Przejść na widok „Ogłoszenia zawodników”.
5. Znaleźć wpis dotyczący nowoutworzonego konta zawodnika.

Oczekiwany rezultat:

System powinien utworzyć wpis na widoku „Ogłoszenia zawodników”, dotyczący nowoutworzonego konta, który członek zarządu może potwierdzić i zaakceptować prośbę zawodnika lub odrzucić.

Otrzymany rezultat:

Oczekiwany wpis został dodany.



Rys. 5.28. Wynik testu rejestracji zawodnika z drużyną

Źródło: Opracowanie własne

Test 6. Przetestowanie działania funkcjonalności poszukiwania drużyny przez zawodnika

Kroki do wykonania testu:

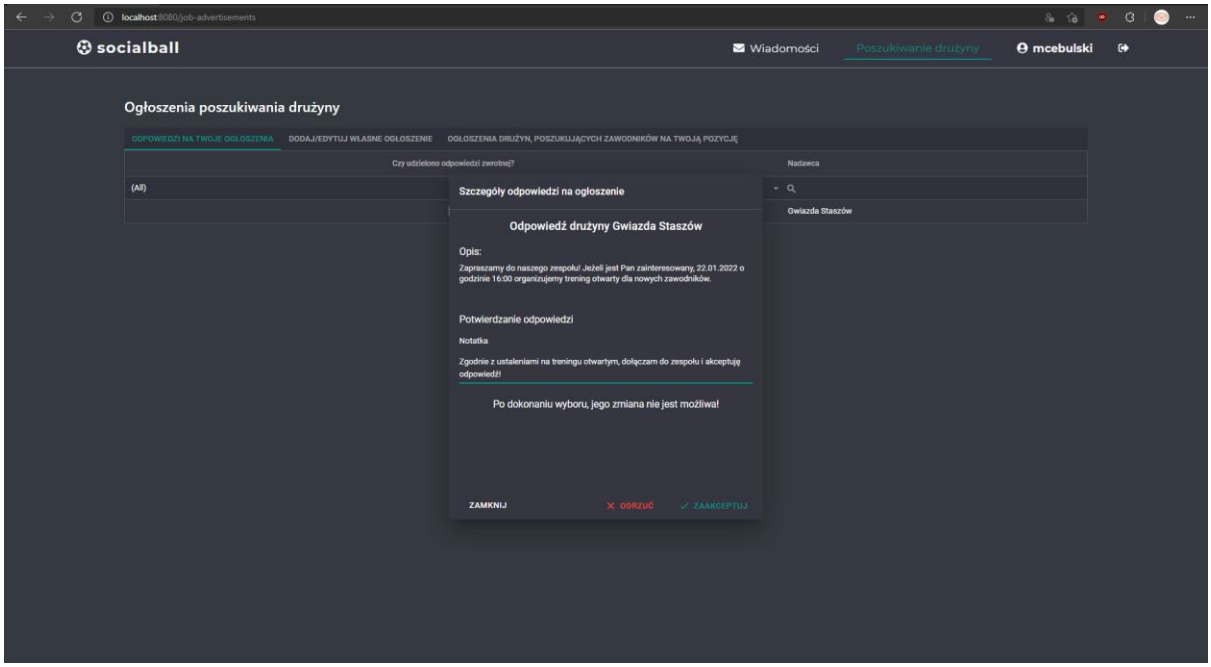
1. Zalogować się na konto zawodnika bez przypisanej drużyny.
2. Utworzyć ogłoszenie o poszukiwaniu drużyny.
3. Zalogować się na konto zarządu drużyny.
4. Przejść na widok „Poszukiwanie drużyny” oraz wybrać zakładkę „Ogłoszenia zawodników bez drużyny”, gdzie powinien być widoczny utworzony wpis.
5. Złożyć odpowiedź na ogłoszenie zawodnika.
6. Zalogować się ponownie na konto zawodnika.

Oczekiwany rezultat:

Odpowiedź na ogłoszenie powinna być widoczna, a po dokonaniu akceptacji, system powinien przydzielić zawodnika do drużyny.

Otrzymany rezultat:

Odpowiedź jest widoczna, a po jej akceptacji, zawodnik został przypisany do drużyny.



Rys. 5.29. Wynik testu funkcjonalności poszukiwania drużyny

Źródło: Opracowanie własne

7. Wnioski

Zaprojektowana oraz utworzona aplikacja internetowa w pełni spełnia postawione założenia projektowe. Wybrane technologie informatyczne sprawdziły się bardzo dobrze, a ich implementacja nie sprawiła problemów. Wykonane testy potwierdziły niezawodność oraz sprawne działanie funkcjonalności utworzonej aplikacji, która jest prosta w obsłudze. Stworzony projekt posiada dużą liczbę możliwości, a przy tym wciąż posiada potencjał na rozbudowę.

Pierwszym planem dalszego rozwoju jest dodanie wysyłania automatycznych wiadomości e-mail. Sprawdziłyby się one jako powiadomienia systemowe, informacje o otrzymanej nowej wiadomości w systemie, potwierdzenie w procesie rejestracji lub przypomnienia o treningach. Drugim z pomysłów byłaby ewolucja istniejących rozwiązań – rozbudowa statystyk wyświetlanych w aplikacji, rozwój responsywności, umożliwienie większej personalizacji, zarówno zawodnikowi, jak i drużynie, poprzez dostosowywanie kolorystyki aplikacji do zespołu zalogowanego użytkownika, dodawanie oznaczeń sponsorskich. Dodatkowym usprawnieniem mogłoby być oznaczanie frekwencji, gdzie zawodnicy przed treningiem oznaczaliby swoją obecność, która następnie byłaby weryfikowana przez członka zarządu. Trzecim planem dalszego rozwoju, jest dodanie obsługi dla innych języków niż polski, skupiając się w pierwszej kolejności na języku angielskim.

Literatura

[1] TeamSnap, https://www.teamsnap.com/, data dostępu: 30.12.2021

[2] sportbm, https://sportbm.com/, data dostępu: 30.12.2021

[3] Find a Player, https://www.findaplayer.com/, data dostępu: 30.12.2021

[4] Wprowadzenie do platformy ASP.NET Core, https://docs.microsoft.com/pl-pl/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-6.0, data dostępu: 04.01.2022

[5] A. Freeman: ASP.NET MVC 5 Zaawansowane programowanie, Helion, Gliwice 2015

[6] Microsoft Docs: Wprowadzenie do zapytań LINQ, https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/introduction-to-linq-queries, data dostępu: 04.01.2022

[7] MDN Web Docs: Co to jest JavaScript?, https://developer.mozilla.org/pl/docs/Learn/JavaScript/First_steps/What_is_JavaScript, data dostępu: 05.01.2022

[8] Vue 3 Guide: Vue Introduction, https://v3.vuejs.org/guide/introduction.html, data dostępu: 05.01.2022

[9] Vuex Docs: What is Vuex?, https://vuex.vuejs.org/, data dostępu: 05.01.2022

[10] Axios Docs: Getting Started, https://axios-http.com/docs/intro, data dostępu: 05.01.2022

[11] Vue Router Guide: Getting Started, https://router.vuejs.org/guide/, data dostępu: 05.01.2022

[12] DevExtreme Docs: DevExtreme Vue Components, https://js.devexpress.com/Documentation/Guide/Vue_Components/DevExtreme_Vue_Components/, data dostępu: 05.01.2022

[13] Bootstrap Docs: Introduction, https://getbootstrap.com/docs/5.1/getting-started/introduction/, data dostępu: 05.01.2022

[14] MDN Web Docs: HTML podstawy, https://developer.mozilla.org/pl/docs/Learn/Getting_started_with_the_web/HTML_basics, data dostępu: 03.01.2022

[15] MDN Web Docs: CSS podstawy, https://developer.mozilla.org/pl/docs/Learn/Getting_started_with_the_web/CSS_basics, data dostępu: 03.01.2022

[16] Richard Peterson: What is SQL Server? Introduction, History, Types, Versions, https://www.guru99.com/sql-server-introduction.html, data dostępu: 03.01.2022

[17] Microsoft Docs: Wprowadzenie do Transact-SQL, https://docs.microsoft.com/pl-pl/learn/modules/introduction-to-transact-sql/1-introduction, data dostępu: 03.01.2022
[18] A. Saikia, S. Joy, D. Dolma, R. Mary. R: Comparative Performance Analysis of MySQL and SQL Server Relational Database Management Systems in Windows Environment, 2015, https://www.ijarcce.com/upload/2015/march-15/IJARCCE%2039.pdf, data dostępu: 05.01.2022
[19] S. Chacon, B. Straub: Pro Git, E-book, https://git-scm.com/book/pl/v2, Second Edition, Apress Open, 2021, data dostępu: 03.01.2022
[20] Microsoft Docs: Omówienie platformy ASP.NET Core MVC - Wzorzec MVC, https://docs.microsoft.com/pl-pl/aspnet/core/mvc/overview?view=aspnetcore-6.0, data dostępu: 05.01.2022
[21] AWS Amazon Docs: AWS SDK for .NET, https://docs.aws.amazon.com/sdk-for-net/v3/developer-guide/welcome.html, data dostępu: 06.01.2022