



**Politechnika  
Śląska**

Dokumentacja zrealizowanego projektu

2022/2023

## **Bazy Danych**

Kierunek: Informatyka

Członkowie zespołu:

*Bartosz Bugla*

*Kamil Grabowski*

*Michał Bober*

*Bartosz Pacia*

Gliwice, 2022/2023

# Spis treści

<b>1</b>	<b>Wprowadzenie</b>	<b>2</b>
1.1	Cel projektu . . . . .	2
1.2	Role w projekcie . . . . .	2
<b>2</b>	<b>Założenia projektowe</b>	<b>2</b>
2.1	Opis wymagań . . . . .	2
2.2	Encje . . . . .	3
2.3	Związki . . . . .	3
2.4	Diagram związków encji . . . . .	4
2.5	Model relacyjny . . . . .	5
2.6	Dodatkowe funkcje . . . . .	6
2.6.1	Trigger'y . . . . .	6
2.6.2	Widoki . . . . .	9
2.7	Obsługa bazy . . . . .	10
<b>3</b>	<b>Wnioski</b>	<b>14</b>
3.1	Podsumowanie projektu . . . . .	14
3.2	Potencjał rozwoju . . . . .	14

# 1 Wprowadzenie

## 1.1 Cel projektu

Projekt zakłada stworzenie bazy danych do zarządzania statystykami meczów. Sam program przeznaczony jest dla kibiców, którzy chcą podejrzeć aktualne statystyki z meczu, ligi lub swojego ulubionego zawodnika. Natomiast osoba z dostępem do bazy jest w stanie tworzyć nowe ligi oraz zarządzać planem rozgrywek.

## 1.2 Role w projekcie

1. Kamil Grabowski - utworzenie struktury, tworzenie dokumentacji
2. Bartosz Bugla - uzupełnienie danych, tworzenie dokumentacji
3. Michał Bober - utworzenie struktury, tworzenie dokumentacji
4. Bartosz Pacia - uzupełnienie danych, stworzenie funkcji oraz trigger'ów

# 2 Założenia projektowe

## 2.1 Opis wymagań

- Nasza baza w założeniu ma za zadanie pomagać kibicom oraz organizatorom lig lub turniejów. Naszym celem jest stworzenie bazy, która będzie w stanie wyświetlać informacje oraz statystyki z danego meczu.
- Dzięki naszej bazie będzie możliwość stworzenia ligi w dyscyplinie sportowej Lacrosse wraz z drużynami oraz rozpiskom stopkań.
- Baza przechowuje informacje o statystykach z danego meczu, o statystykach każdego zawodnika, informacje o przebiegu spotkania oraz ogólne informacje dotyczące ligi.
- Obsługiwanie bazy będzie możliwe dla każdej osoby, ponieważ będzie się to odbywało przez przeglądarkę. Oznacza to, że kibic jak i organizator będzie w stanie w każdym momencie podejrzeć lub zmienić dane.

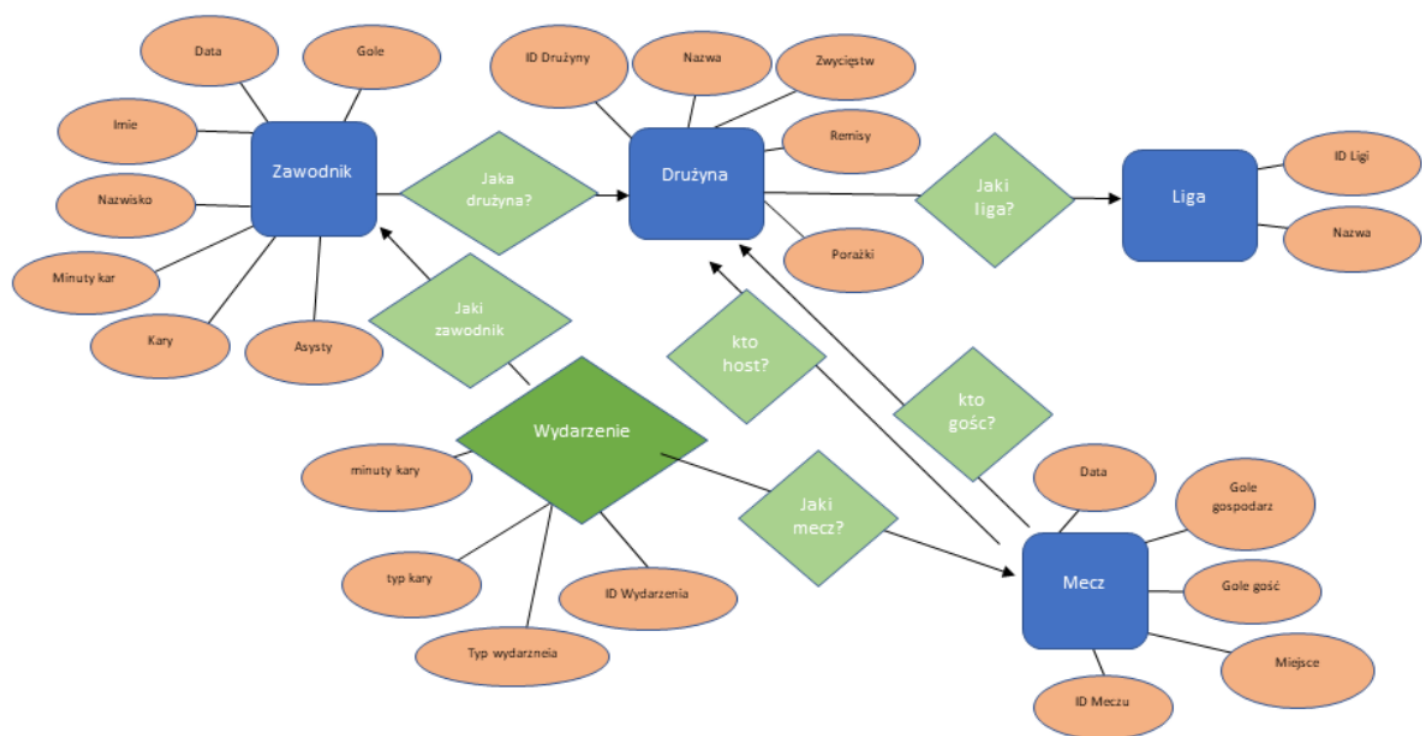
## 2.2 Encje

1. Liga - ID ligi[PK], Nazwa.
2. Drużyna - ID Drużyny[PK], Nazwa, Zwycięstwa, Remisy, Porażki, Punkty.
3. Mecz - ID MeczU[PK], Data, Miejsce, Gole gospodarz, Gole gość
4. Zawodnik - ID Zawodnika[PK], Imie, Nazwisko, Data urodzenia, Pozycja, Gole, Asysty, kary, Minuty kar.
5. Wydarzenia - ID Wydarzenia[PK], typ wydarzenia(gol lub faul), asysta, typ kary, minuty kary.

## 2.3 Związki

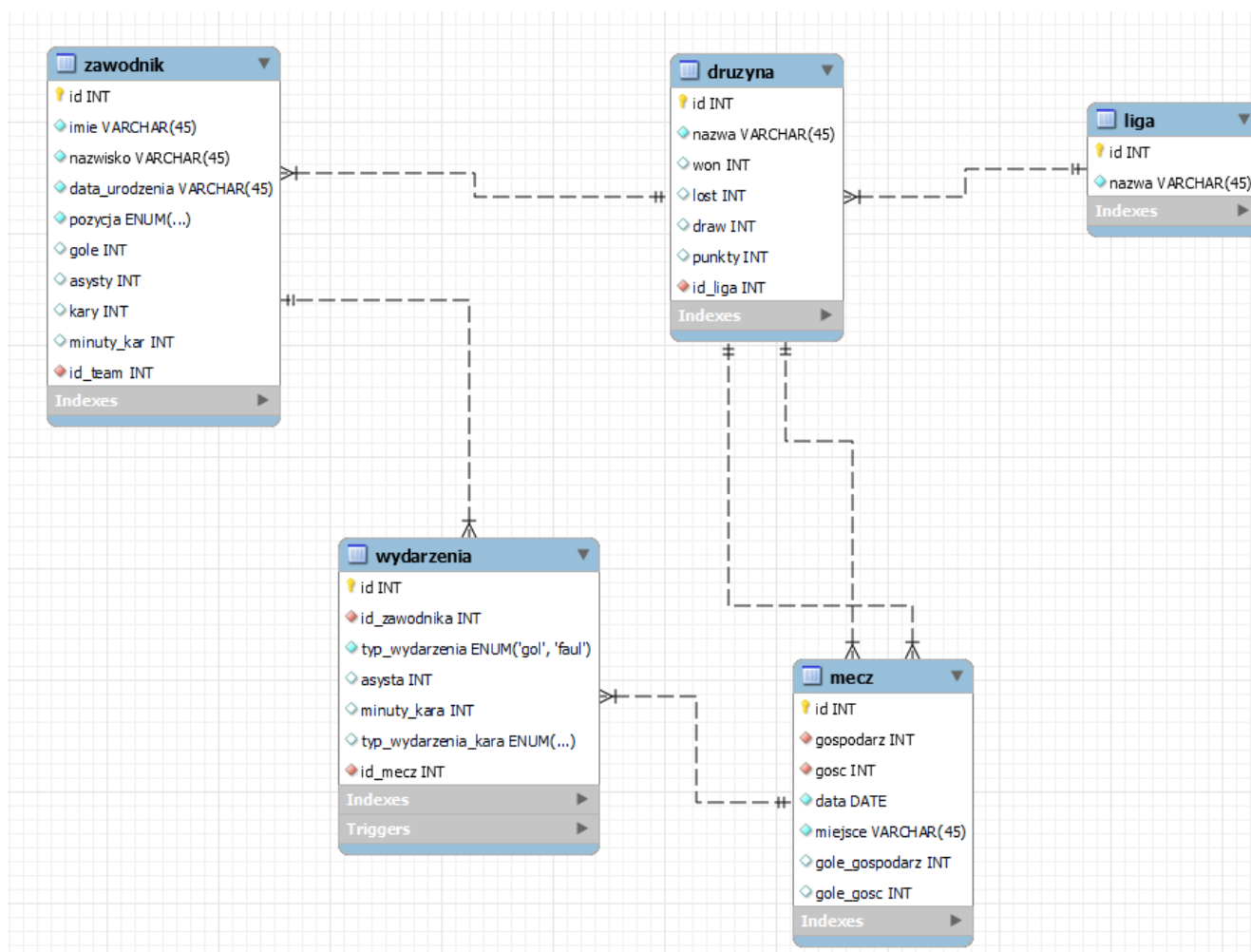
1. Zawodnik
  - Jak drużyna? - Zawodnik - Drużyna 1 : n, każdy zawodnik należy do drużyny.
2. Drużyna
  - W jakiej lidze? - Drużyna - Liga 1 : n, każda drużyna należy do ligi.
3. Wydarzenia
  - Jaki zawodnik? - Wydarzenia - Zawodnik(gol, faul) 1 : n, wydarzenie dotyczy danego zawodnika.
  - Kto ma asystę? - Wydarzenia - Zawodnik(asysta) 1 : n, asysta dotyczy danego zawodnika.
  - Jaki mecz? - Wydarzenia - Mecz 1 : n, wydarzenie należy do danego meczu.
4. Mecz
  - Przydział gospodarza - Mecz (gospodarz) - Drużyna 1 : n, drużyna gospodarzy należy do drużyny.
  - Przydział gościa - Mecz (gość) - Drużyna 1 : n, drużyna gości należy do drużyny.

## 2.4 Diagram związków encji



Rysunek 1: Model encji

## 2.5 Model relacyjny



Rysunek 2: Model relacyjny

## 2.6 Dodatkowe funkcje

### 2.6.1 Trigger'y

Dla tabeli 'wydarzenia' zastosowaliśmy trigger (AFTER INSERT), który

- automatycznie aktualizuje statystyki zawodników po dodaniu wydarzenia. Aktualizuje w ten sposób kolumny goli, fauli, asyst oraz minut kar.
- automatycznie aktualizuje wynik meczu. Po dodaniu wydarzenia, który jest golem dodaje dla strzelającej drużyny bramkę.

```
CREATE DEFINER='root'@'localhost' TRIGGER
'wydarzenia_AFTER_INSERT_zawodnik_and_mecz' AFTER INSERT
ON 'wydarzenia' FOR EACH ROW BEGIN
/*Aktualizacja tabeli zawodnik*/
    IF NEW.typ_wydarzenia = 'Gol' THEN
        UPDATE zawodnik SET gole = gole + 1
        WHERE id = NEW.id_zawodnika;
        IF (new.asysta is not null) THEN
            UPDATE zawodnik set asysty = asysty + 1
            WHERE id = new.asysta;
        END IF;
    ELSEIF NEW.typ_wydarzenia = 'Faul' THEN
        UPDATE zawodnik SET kary = kary + 1
        WHERE id = new.id_zawodnika;
        UPDATE zawodnik set minuty_kar = minuty_kar + new.minuty_kara
        where id = new.id_zawodnika;
    END IF;

/*Aktualizacja tabeli mecz*/
    IF NEW.typ_wydarzenia = 'Gol' THEN
        IF EXISTS (SELECT 1 FROM mecz WHERE id = new.id_mecz
        AND gospodarz in (select druzyna.id from zawodnik
        INNER JOIN druzyna on druzyna.id = zawodnik.id_team
        WHERE zawodnik.id = new.id_zawodnika)) THEN
            UPDATE mecz SET gole_gospodarz = gole_gospodarz + 1
            WHERE id = NEW.id_mecz;
        ELSEIF EXISTS (SELECT 1 FROM mecz
        WHERE id = NEW.id_mecz
        AND gosc in (select druzyna.id from zawodnik
        INNER JOIN druzyna on druzyna.id = zawodnik.id_team
```

```

WHERE zawodnik.id = new.id_zawodnika)) THEN
    UPDATE mecz SET gole_gosc = gole_gosc + 1
    WHERE id = NEW.id_mecz;
END IF;
END IF;
END

```

- Również dla tabeli 'wydarzenia' trigger (AFTER UPDATE), który aktualizuje w przypadku zmiany statystyki zawodników oraz wynik meczu.

```

CREATE DEFINER='root'@'localhost' TRIGGER 'wydarzenia_AFTER_UPDATE'
AFTER UPDATE ON 'wydarzenia' FOR EACH ROW BEGIN
/*Zakładamy, że user zmienia tylko na zawodników
   z danej drużyny. W innym wypadku usuwa wartość*/
IF (OLD.id_zawodnika != new.id_zawodnika) then
UPDATE zawodnik set gole = gole - 1
    where zawodnik.id = OLD.id_zawodnika;
END IF;
IF (OLD.asysta != new.asysta) then
UPDATE zawodnik SET asysty = asysty + 1
    where zawodnik.id = new.id_zawodnika;
END IF;
IF (OLD.id_zawodnika != new.id_zawodnika) then
UPDATE zawodnik set kary = kary - 1
    where zawodnik.id = OLD.id_zawodnika;
END IF;
IF (OLD.id_zawodnika != new.id_zawodnika) then
UPDATE zawodnik set minuty_kar = minuty_kar - old.minuty_kara
    where zawodnik.id = OLD.id_zawodnika;
END IF;

END

```



- Również dla tabeli 'wydarzenia' trigger (AFTER DELETE), który aktualizuje w przypadku usunięcia statystyki zawodników oraz wynik meczu.

```

CREATE DEFINER='root'@'localhost' TRIGGER 'wydarzenia_AFTER_DELETE'
AFTER DELETE ON 'wydarzenia' FOR EACH ROW BEGIN
IF OLD.typ_wydarzenia = 'Gol' THEN
    IF EXISTS (SELECT 1 FROM mecz
    WHERE id = OLD.id_mecz
    AND gospodarz in (select druzyna.id from zawodnik
    INNER JOIN druzyna on druzyna.id = zawodnik.id_team
    WHERE zawodnik.id = OLD.id_zawodnika)) THEN
        UPDATE mecz SET gole_gospodarz = gole_gospodarz - 1
        WHERE id = OLD.id_mecz;
    ELSEIF EXISTS (SELECT 1 FROM mecz
    WHERE id = OLD.id_mecz AND gosc
    in (select druzyna.id from zawodnik
    INNER JOIN druzyna on druzyna.id = zawodnik.id_team
    WHERE zawodnik.id = OLD.id_zawodnika)) THEN
        UPDATE mecz SET gole_gosc = gole_gosc - 1
        WHERE id = OLD.id_mecz;
    END IF;
end if;

IF OLD.typ_wydarzenia = 'Gol' THEN
    UPDATE zawodnik SET gole = gole - 1
    WHERE id = OLD.id_zawodnika;
    IF (OLD.asysta is not null) THEN
        UPDATE zawodnik set asysty = asysty - 1
        WHERE id = OLD.asysta;
    END IF;
ELSEIF OLD.typ_wydarzenia = 'Faul' THEN
UPDATE zawodnik SET kary = kary - 1
    WHERE id = OLD.id_zawodnika;
        UPDATE zawodnik set minuty_kar = minuty_kar - OLD.minuty_kara
        where id = OLD.id_zawodnika;
    END IF;
END

```

### 2.6.2 Widoki

Do wyświetlenia drużyny, która strzeliła największą ilość bramek w danej lidze zastosowaliśmy widok. Wyświetla nazwę drużyny, ilość zdobytych goli oraz nazwę ligi w której gra dana drużyna.

```
CREATE
  ALGORITHM = UNDEFINED
  DEFINER = 'root'@'localhost'
  SQL SECURITY DEFINER
VIEW 'best_team' AS
  SELECT
    'druzyna'.'nazwa' AS 'druzyna',
    SUM('zawodnik'.'gole') AS 'zdobyte_gole',
    'liga'.'nazwa' AS 'nazwa'
  FROM
    (('zawodnik'
    JOIN 'druzyna' ON (('druzyna'.'id' = 'zawodnik'.'id_team')))
    JOIN 'liga' ON (('druzyna'.'id_liga' = 'liga'.'id')))
  WHERE
    (('druzyna'.'id_liga' = 1)
    OR ('druzyna'.'id_liga' = 2))
  GROUP BY 'zawodnik'.'id_team'
  ORDER BY SUM('zawodnik'.'gole') DESC
```

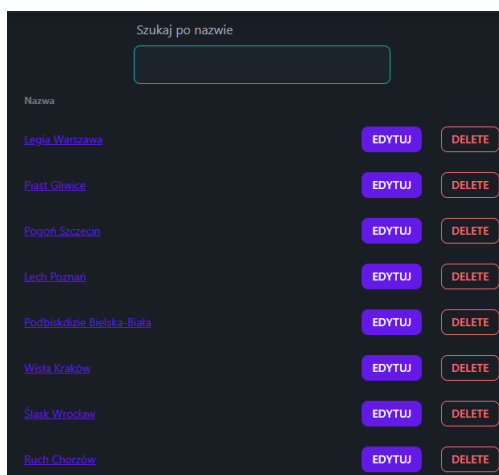
## 2.7 Obsługa bazy

Strona startowa wyświetla rozegrane oraz te jeszcze nierozegrane mecze. Prezentuje się to w następujący sposób.

Witaj w LaStats					
Najlepsza strona z drużynami oraz meczami					
Home	Result	Away		Time	Liga
<a href="#">KAC Vienna</a>	0-0	<a href="#">Bats Bratislava</a>		11-08-2023 09:00	<a href="#">Czeska Liga</a>
<a href="#">Legia Warszawa</a>	2-1	<a href="#">Piast Gliwice</a>	Ended	24-06-2023 11:41	<a href="#">Ekstraklasa</a>
<a href="#">Lech Poznań</a>	2-3	<a href="#">Pogoń Szczecin</a>	Ended	23-06-2023 11:41	<a href="#">Ekstraklasa</a>
<a href="#">LCC Custodes</a>	0-0	<a href="#">Polish Eagles</a>	Ended	07-05-2023 14:00	<a href="#">Czeska Liga</a>
<a href="#">LCC Custodes</a>	0-0	<a href="#">LCC Wolfs</a>	Ended	05-02-2023 19:00	<a href="#">Czeska Liga</a>

Rysunek 3: Wyświetlenie meczów

Wszystkie drużyny oraz ich informacje, kto w nich gra możemy podrzeć w zakładce drużyny



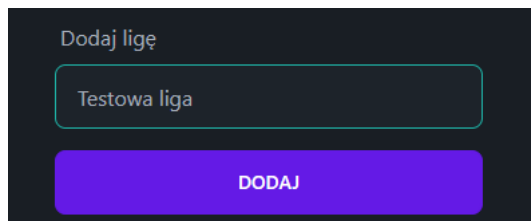
Rysunek 4: Wyświetlenie meczów

Dodawanie drużyny

A screenshot of a web application form for adding a new football team. The form is titled "Dodawanie drużyny" and contains the following fields: "Nazwa \*" (required), "Wins", "Draws", and "Losses". The "Nazwa \*" field contains the text "Testowa drużyna". Below the "Wins", "Draws", and "Losses" fields, there is a note: "Pola poniżej muszą być liczbowe w przypadku nie podania wartości automatycznie jest 0". At the bottom right of the form, there are two buttons: "ANULUJ" and "DODAJ". The interface has a dark theme.

Rysunek 5: Dodawanie drużyny

Dodawanie ligi



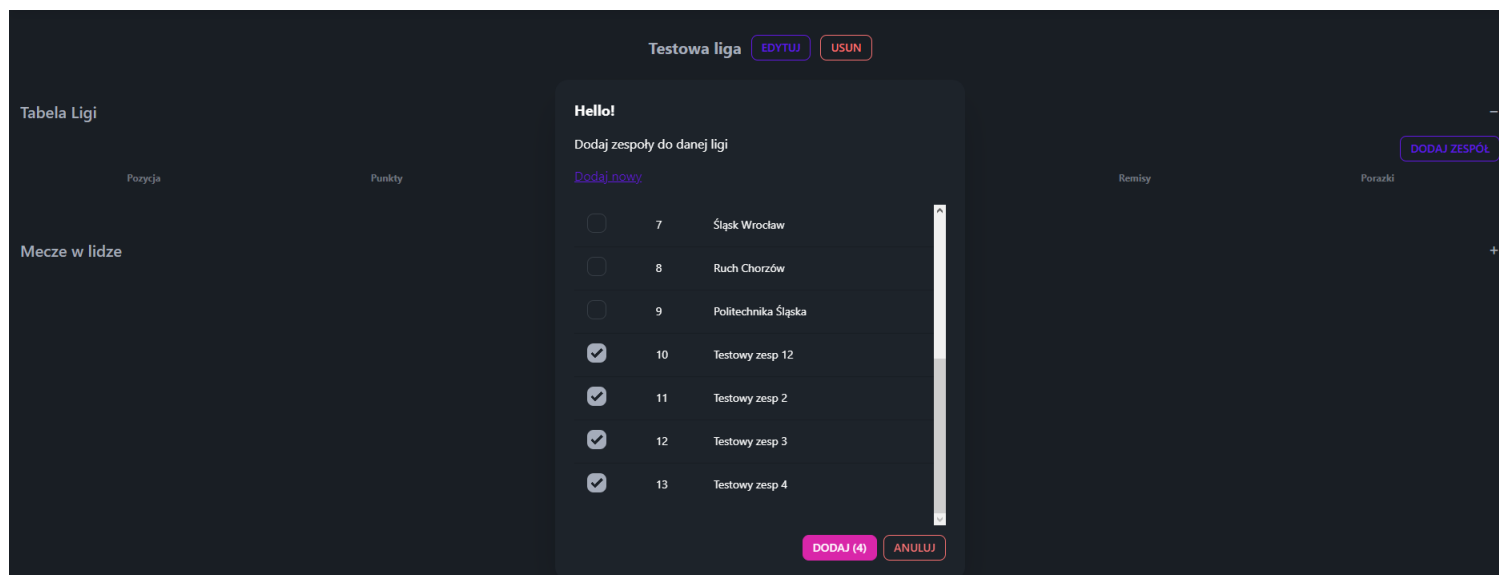
Dodaj ligę

Testowa liga

DODAJ

Rysunek 6: Dodawanie ligi

Gdy już dodaliśmy drużyny oraz lige do bazy. Możemy je przydzielić do ligi w następujący sposób.



Testowa liga [EDYTUJ](#) [USUN](#)

Tabela Ligi

Pozycja Punkty

Mecze w lidze

Remisy Porazki

[DODAJ ZESPÓŁ](#)

**Hello!**

Dodaj zespoły do danej ligi

[Dodaj nowy](#)

<input type="checkbox"/>	7	Śląsk Wrocław
<input type="checkbox"/>	8	Ruch Chorzów
<input type="checkbox"/>	9	Politechnika Śląska
<input checked="" type="checkbox"/>	10	Testowy zesp 12
<input checked="" type="checkbox"/>	11	Testowy zesp 2
<input checked="" type="checkbox"/>	12	Testowy zesp 3
<input checked="" type="checkbox"/>	13	Testowy zesp 4

[DODAJ \(4\)](#) [ANULUJ](#)

Rysunek 7: Przydzielenie drużyny do ligi

Gdy już dodaliśmy drużyny, zawodników oraz ligę jesteśmy w stanie ustawić spotkanie. Możemy wybrać z listy drużyn, które zostały przydzielone do danej ligi.

Testowa liga [EDYTUJ](#) [USUN](#)

Tabela Ligi

Mecze w lidze

Gospodarz meczu

Testowy zesp 2

Gość meczu

Testowy zesp 3

Lokalizacja Spotkania

Polska

Czas Spotkania

08.07.2023 23:38

[ANULUJ](#) [ZATWIERDŹ](#)

[DODAJ MECZ](#)

Rysunek 8: Tworzenie spotkania

## **3    Wnioski**

### **3.1    Podsumowanie projektu**

Udało nam się stworzyć bazę, która ma potencjał do tworzenia realnych turniejów oraz lig. Baza może zostać wykorzystana dla większości sportów lecz trzeba ją odpowiednio zmienić pod względem statystyk. Dla naszej bazy udało się stworzyć rozpiske meczów, statystyki zawodników w danym meczu, możliwość podejrzenia najelspzych zawodników, statystyki drużyn. Do bazy danych użyliśmy realnych statystyk z czeskiej ligi lacrosse w 2023 roku, która wciąż trwa.

### **3.2    Potencjał rozwoju**

W każdym sporcie jest pełno statystyk. Od ilości zebranych piłek po procent posiadanie piłki. W ten sposób jest możliwe rozwinięcie bazy. Wprowadzanie kolejnych statystyk spowoduje rozszerzenie się bazy o kolejne rekordy, tabele oraz kolumny w nich. Baza jest gotowa do wykorzystania w realnych komercyjnych projektach.