Politechnika Poznańska

Wydział Elektryczny, Informatyka

Semestr VII

BAZY DANYCH - Projekt

Rafał Jendraszak

Indeks: 121286

# **Charakterystyka Projektu**

## Opis

Celem projektu jest zaimplementowanie systemu zarządzania meczami dla sędziów piłki ręcznej. Zadaniami do wykonania są zaprojektowanie bazy danych, implementacja logiki aplikacji po stronie bazy danych oraz stworzenie prostego interfejsu dla użytkownika, który będzie umożliwiał wykorzystanie funkcji stworzonych po stronie bazy danych.

## Wykorzystane technologie

Do realizacji wykorzystano język Python wraz z biblioteką Django oraz baza danych PostgreSQL.

Język Python został wybrany, ponieważ jest łatwy do nauczenia, a kod napisany w tym jeżyku jest przejrzysty i czytelny. Biblioteka Django zostanie wykorzystana do stworzenia interfejsu graficznego w postaci strony internetowej, umożliwia ona w prosty sposób manipulację na obiektach bazodanowych, np. dodawanie, edycja, usuwanie, czytanie danych z tabel.

Początkowo wybrana została baza danych MySQL, jednak na etapie tworzenia projektu, napotkano przeszkodę w postaci połączenia kodu aplikacji z bazą danych, dlatego zdecydowano na zmianę na PostgreSQL. Jest to jeden z najpopularniejszych otwartych systemów zarządzania relacyjnymi bazami danych.

## Architektura oprogramowania

Poniższy diagram przedstawia w prosty sposób architekturę aplikacji.

Użytkownik (Klient) łączy się ze stroną internetową, która jest generowana dla niego przez aplikację (Aplikacja w Django), korzystając z danych dostępnych w bazi (Baza danych PostgreSQL)

  
Rysunek 1: Architektura oprogramowania

# **Baza Danych**

## Skrypty tworzące strukturę SZBD

### Drużyna

**create table** referee\_team  
(  
 **id serial not null**  
 **constraint** referee\_team\_pkey  
 **primary key**,  
 **name varchar**(100) **not null**,  
 **city varchar**(50) **not null**,  
 **telephone varchar**(11)  
);

### Sędziowie szczegóły

**create table** referee\_refereedetails  
(  
 **id serial not null**  
 **constraint** referee\_refereedetails\_pkey  
 **primary key**,  
 **referee\_level integer not null**,  
 **city varchar**(50),  
 **telephone varchar**(11),  
 **user\_id integer not null**  
 **constraint** referee\_refereedetails\_user\_id\_key  
 **unique**  
 **constraint** referee\_refereedetails\_user\_id\_c9200df3\_fk\_auth\_user\_id  
 **references** auth\_user  
 **deferrable initially deferred**  
);

### Status meczu

**create table** referee\_matchstatus  
(  
 **id serial not null**  
 **constraint** referee\_matchstatus\_pkey  
 **primary key**,  
 **name varchar**(100) **not null**,  
 **next\_status integer**  
);

### Kategoria meczu

**create table** referee\_matchcategory  
(  
 **id serial not null**  
 **constraint** referee\_matchcategory\_pkey  
 **primary key**,  
 **name varchar**(50) **not null**  
);

### Mecz

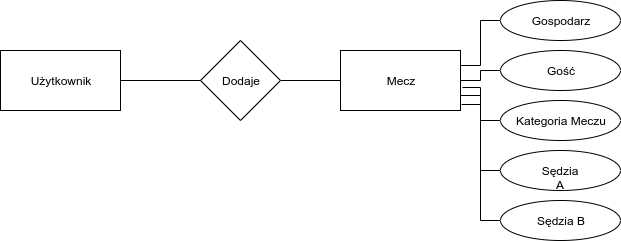
**create table** referee\_match  
(  
 id **serial not null**  
 **constraint** referee\_match\_pkey  
 **primary key**,  
 match\_number **varchar**(10) **not null**,  
 date\_time **timestamp with time zone not null**,  
 away\_team\_id **integer**  
 **constraint** referee\_match\_away\_team\_id\_fkey  
 **references** referee\_team  
 **deferrable initially deferred**,  
 home\_team\_id **integer**  
 **constraint** referee\_match\_home\_team\_id\_fkey  
 **references** referee\_team  
 **deferrable initially deferred**,  
 match\_category\_id **integer**  
 **constraint** referee\_match\_match\_category\_id\_fkey  
 **references** referee\_matchcategory  
 **deferrable initially deferred**,  
 referee\_a\_id **integer**  
 **constraint** referee\_match\_referee\_a\_id\_fkey  
 **references** auth\_user  
 **deferrable initially deferred**,  
 referee\_b\_id **integer**  
 **constraint** referee\_match\_referee\_b\_id\_fkey  
 **references** auth\_user  
 **deferrable initially deferred**,  
 match\_status\_id **integer**  
 **constraint** referee\_match\_match\_status\_id\_fkey  
 **references** referee\_matchstatus  
 **deferrable initially deferred**  
);

### Wyniki meczu

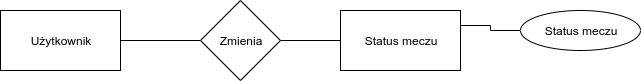
**create table** referee\_matchresult  
(  
 **id serial not null**  
 **constraint** referee\_matchresult\_pkey  
 **primary key**,  
 **home\_team\_goals integer**,  
 **away\_team\_goals integer**,  
 **winner varchar**(1),  
 **match\_id integer not null**  
 **constraint** referee\_matchresult\_match\_id\_ef3a6230\_fk\_referee\_match\_id  
 **references** referee\_match  
 **deferrable initially deferred**  
);

## Diagramy ER

### Dodanie meczu

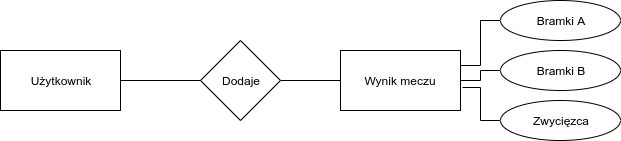
  
Diagram 1

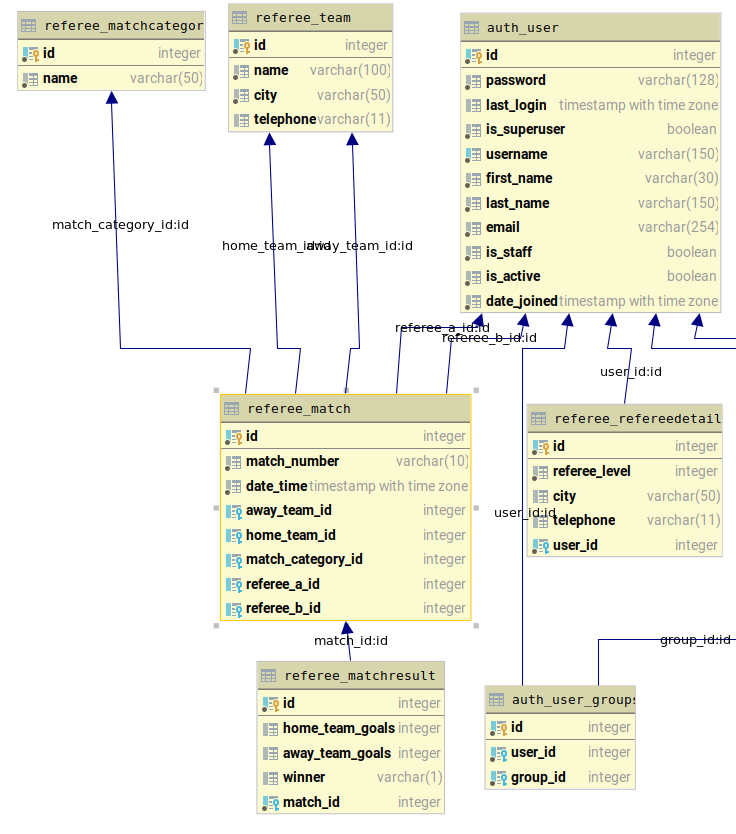
### Zmiana statusu meczu

  
Diagram 2

### Dodanie wyniku meczu

## Diagram DB

  
Diagram 3

  
Diagram 4

# Przykłady rozwiązań

## Triggery

### Sprawdzenie wyniku i wpisanie do tabeli zwycięzcy meczu

**create trigger** t\_insert\_winner **after insert or update on** referee\_matchresult  
 **for each row**  
 **execute procedure** f\_insert\_winner();  
  
**drop trigger** t\_insert\_winner **on** referee\_matchresult;  
  
**create or replace function** f\_insert\_winner()  
**returns trigger as**  
 **$BODY$**  
 **BEGIN**  
 **if new**.home\_team\_goals > **new**.away\_team\_goals **then**  
 **update** referee\_matchresult **set** winner = **'A' where** id = **new**.id;  
 **else**  
 **update** referee\_matchresult **set** winner = **'B' where** id = **new**.id;  
 **end if**;  
  
 **return new**;  
 **end**;  
 **$BODY$**  
  
**language** plpgsql **volatile**;

Podczas prezentacji projektu na zajęciach, powyższy trigger działał tylko w przypadku, gdy dokonano edycji meczu i dodania wyniku. Aktualnie działa również w przypadku dodania nowego rekordu do bazy danych.

### Ustawienie statusu meczu na ‘Nowy’, jeśli status nie został przypisany podczas dodawania meczu

**create trigger** t\_set\_new\_status **after insert on** referee\_match  
 **for each row**  
 **execute procedure** f\_set\_new\_status();  
  
**create or replace function** f\_set\_new\_status()  
**returns trigger as**  
 **$BODY$**  
 **BEGIN**  
 **if new**.match\_status\_id **isnull then**  
 **update** referee\_match **set** match\_status\_id = 1 **where** id = **new**.id;  
 **end if**;  
  
 **return new**;  
 **end**;  
 **$BODY$**  
**language** plpgsql **volatile**;

## Widoki

### Wyświetlenie meczy wraz z wynikami oraz sędziami

**create or replace view** v\_match\_details **as**  
**select**  
rm.**match\_number AS** MATCH\_NUMBER,  
 rm.**date\_time AS** DATE\_TIME,  
 rmc.**name AS** MATCH\_CATEGORY,  
 rtA.**name AS** HOME\_TEAM,  
 rtB.**name as** AWAY\_TEAM,  
 auA.**last\_name as** REFEREE\_A,  
 auB.**last\_name as** REFEREE\_B,  
 rmr.**home\_team\_goals as** HOME\_TEAM\_GOALS,  
 rmr.**away\_team\_goals as** AWAY\_TEAM\_GOALS,  
 **null as** id,  
 rm.**id as** match\_id,  
 rms.**name as** match\_status,  
 rmr.**winner as** match\_winner  
**from**  
referee\_match rm  
**join** referee\_matchcategory rmc **on** rm.**match\_category\_id** = rmc.**id**  
**join** referee\_team rtA **on** rm.**home\_team\_id** = rtA.**id**  
**join** referee\_team rtB **on** rm.**away\_team\_id** = rtB.**id**  
**left join** referee\_matchresult rmr **on** rm.**id** = rmr.**match\_id**  
**join** auth\_user auA **on** rm.**referee\_a\_id** = auA.**id**  
**join** auth\_user auB **on** rm.**referee\_b\_id** = auB.**id**  
**left join** referee\_matchstatus rms **on** rm.**match\_status\_id** = rms.**id**;

### Wyświetlanie danych sędziego

### **create or replace view** v\_ref\_details **as** **select** au.**first\_name**, au.**last\_name**, rr.**city**, rr.**referee\_level**, rr.**telephone** **from** auth\_user au **join** referee\_refereedetails rr **on** au.**id** = rr.**user\_id**;

## Procedury

### Wyświetlenie drużyn z danego miasta

**create or replace function** *f\_show\_teams\_from\_city*(t\_city **varchar**)  
**returns setof** referee\_team  
 **language sql**  
 **as**  
 **$$**  
 **select** *\** **from** referee\_team rt **where** rt.**city** = t\_city;  
 **$$**;

### Wyświetlenie **sędziów** z danego miasta

**create or replace function** *f\_show\_ref\_from\_city*(r\_city **varchar**)  
**returns setof** v\_ref\_details  
**language sql**  
**as**  
 **$$**  
 **select** *\** **from** v\_ref\_details vrd **where** vrd.**city** = r\_city;  
 **$$**;

### Wyświetlenie **meczy, gdzie dana drużyna jest gospodarzem/gościem**

**create or replace function** *f\_show\_matches\_where\_team\_is\_home*(f\_team **varchar**)  
**returns setof** v\_match\_details  
**language sql**  
**as**  
 **$$**  
 **select** *\** **from** v\_match\_details vmd **where** vmd.**home\_team** = f\_team;  
 **$$**;

**create or replace function** *f\_show\_matches\_where\_team\_is\_away*(f\_team **varchar**)  
**returns setof** v\_match\_details  
**language sql**  
**as**  
 **$$**  
 **select** *\** **from** v\_match\_details vmd **where** vmd.**away\_team** = f\_team;  
 **$$**;

### Zmiana statusu meczu

**create or replace procedure** *f\_change\_match\_status*(f\_match\_id **integer**)  
**language** plpgsql  
**as**  
 **$$**  
 **BEGIN**  
 **update**  
 **referee\_match**  
 **set**  
 **match\_status\_id = (**  
 **select next\_status from referee\_matchstatus where id = (select match\_status\_id from referee\_match where id = f\_match\_id)**  
 **)**  
 **where**  
 **id = f\_match\_id;**  
  
 **commit;**  
 **end;**  
 **$$**;

## Transakcje

### Dodanie nowego meczu