

Gliwice, 12.09.2018

# Menager piłkarski z bazą danych

Prowadzący: Krzysztof Pasterak

Autor:

Krzysztof Werner

## Temat

Tematem projektu jest stworzenie menagera piłkarskiego opartego o bazę danych.

## Analiza tematu

Podczas analizy tematu musiałem podjąć decyzję jakiej bazy danych użyję, oraz jaką strukturę będzie miał mój program.

Pierwszą decyzją był... wybór bazy. Zastanawiałem się między relacyjną i nierelacyjną bazą danych. Nierelacyjna baza danych pozwoliłaby mi na trzymanie informacji o różnych typach piłkarzy w jednej tabeli bez wielu niepotrzebnych kolumn. Problemem okazał się brak bibliotek, oraz mojego doświadczenia w bazach danych tego typu. Ze względu na niewielkie skomplikowanie modelu zdecydowałem się na sqlite – co oznacza po prostu trzymanie danych w jednym pliku i wyciąganie ich za pomocą sqlowych zapytań.

Co do struktury programu – chciałem, aby mój program był przejrzysty i skalowalny. Postanowiłem go więc podzielić na model – w którym trzymam informację o klasach bazodanowych, Controller – kontroler zarządzający programem, Helperzy – czyli wydzielone minikontrolery odpowiedzialne za różne funkcje w programie oraz Dao – czyli warstwę komunikacji z bazą danych. Struktura programu wzięłem z projektów w których pracowałem – dlatego struktura jak i nazewnictwo są bardzo zbliżone do projektów w Javie.

## Specyfikacja zewnętrzna

Instrukcja obsługi jest dostępna w programie pod opcja menu Instructions.

Program posiada interfejs tekstowy dostępny w konsoli. Na razie program oferuje tylko jeden tryb gry. Aby rozpocząć działanie z programem – gdy pojawi się menu należy wybrać opcję 3, gdy program był już wcześniej uruchamiany – jak nie, ten krok można pominąć. Program wtedy czyści bazę danych. Następnie należy wybrać opcję 2 – setup database, która tworzy odpowiednie tabele w bazie danych i wypełnia je początkowymi danymi.

Gdy dane zostaną dodane do bazy można wybrać opcję 1 – new game. Na początku tworzona jest lista meczów na cały sezon. Następnie użytkownik dostaje do wyboru klub. Gdy poprawnie do wybierze – można rozpocząć grę.

Przed każdym meczem użytkownik jest pytany o taktykę zespołu na dany mecz. Do wyboru jest: ultradefensywna, defensywna, zbalansowana, ofensywna i ultraofensywna. Wybór taktyki zmienia algorytm wyliczający szanse zespołu w danym meczu, oraz strzelone i stracone gole.

Po rozegranym meczu wyświetlane są wyniki w danej kolejce. Następnie użytkownik może ustawić plan treningowy dla każdego z zawodników w swoim zespole. Domyślnie jest wybrany trening zbalansowany. W ramach tego treningu zawodnik MOŻE otrzymać +1 do 3 różnych statystyk. Niestety przy takim treningu – użytkownik nie ma wpływu jakie statystyki zostaną poprawione – może się

okazać, że napastnikowi podniosły się umiejętności bramkarskie. Użytkownik może wybrać trening pozycyjny - GK dla bramkarza, CB dla obrońcy, MF dla pomocnika czy ST dla napastnika. Oczywiście można przypisać trening dla napastnika nawet obrońcy. W ramach takiego treningu zawodnikowi MOŻE się podnieść jedna umiejętność w ramach danej gupy umiejętności o +2.

Użytkownik rozgrywa mecze, oraz ustala treningi i taktykę aż do połowy sezonu. Wtedy pojawia się okienko transferowe. W ramach okienka użytkownik ma możliwość zamiany jednego ze swoich zawodników na takiego dostępnego na liście transferowej. W zależności czy transferowany zawodnik będzie miał mniejsze czy większe umiejętności od wytransferowanego zawodnika z konta klubu ubędzie lub przybędzie pieniędzy w budżecie.

Następnie następuje kolejna połowa sezonu zakończona okienkiem transferowym i oficjalnymi wynikami. Użytkownik jest pytany czy chce rozpocząć następny sezon i gdy tak się stanie – kolejny sezon jest tworzony.

Poniżej – kilka zrzutów z działania aplikacji:

The screenshot shows a window titled "C:\Users\User\Documents\stidia\PK\Menager2019\Test2\bin\Debug\Test2.exe". The interface displays player statistics for three players: 58 Christian Eriksen25 MF, 69 Timo Werner6 ST, and 77 Timo Werner14 ST. Each player's section shows their current training type (A) and a table of skills (DIV, HAN, KIC, REF, TAC, HEA, MAR, ITC, VIS, CROS, LPAS, SPAS, FIN, POW, VOL, LSHOT, FKIC, POS) with values ranging from 54 to 95. At the bottom, there is a prompt: "If you want to change training type for any of Your players please type in its id, if not - type in 0: 77 Now type in training type for Your player (GK, CB, MF, ST, or anything different for general training: ST If you want to change training type for any of Your players please type in its id, if not - type in 0:".

Wybór treningu

```
C:\Users\User\Documents\studia\PK\Menager2019\Test2\bin\Debug\Test2.exe
Matchweek: 2
MATCHWEEK | HOME | AWAY | RESULT | H_GOALS | A_GOALS
2 | Piast Gliwice | Wisla Krakow | NOT PLAYED | 0 | 0
2 | Legia Warszawa | Gornik Zabrze | NOT PLAYED | 0 | 0
2 | Lech Poznan | Lechia Gdansk | NOT PLAYED | 0 | 0
Current team tactics: 3 balanced
Please choose new tactics for You club (1 - 5: 1 - ultradefensive, 5 - ultraoffensive), or type in 0 to skip this step:
5
Results:
1 vs 5
3 : 3
2 vs 4
2 : 1
6 vs 3
2 : 1
Legia Warszawa - position: 1 - points: 6 - goals scored: 4 - goals lost: 2 -W:D:L: 2:0:0 - budget: 5100000 - tactics balanced
Piast Gliwice - position: 1 - points: 4 - goals scored: 5 - goals lost: 4 -W:D:L: 1:1:0 - budget: 5070000 - tactics ultraoffensive M
Y CLUB
Wisla Krakow - position: 1 - points: 4 - goals scored: 5 - goals lost: 4 -W:D:L: 1:1:0 - budget: 5070000 - tactics balanced
Lech Poznan - position: 2 - points: 3 - goals scored: 3 - goals lost: 3 -W:D:L: 1:0:1 - budget: 5060000 - tactics balanced
Lechia Gdansk - position: 2 - points: 0 - goals scored: 2 - goals lost: 4 -W:D:L: 0:0:2 - budget: 5020000 - tactics balanced
Gornik Zabrze - position: 2 - points: 0 - goals scored: 2 - goals lost: 4 -W:D:L: 0:0:2 - budget: 5020000 - tactics balanced
Press any key to continue . . .
```

Mecz

```
C:\Users\User\Documents\studia\PK\Menager2019\Test2\bin\Debug\Test2.exe
| | FKIC 92 |
| | POS 95 |
77 Timo Werner14 ST
Current training type: ST
GK | CB | MF | ST
DIV 10 | TAC 56 | VIS 75 | FIN 87
HAN 10 | HEA 66 | CROS 65 | POW 76
KIC 10 | MAR 45 | SPAS 90 | VOL 68
REF 10 | ITC 54 | LPAS 87 | LSHOT 85
| | FKIC 92 |
| | POS 95 |
If you want to change training type for any of Your players please type in its id, if not - type in 0:
0
Team players list
1 David De Gea GK OVERALL SCORE: 89 PRICE: 4009456
12 Diego Godin CB OVERALL SCORE: 82 PRICE: 9971176
32 Kalidou Koulibaly19 CB OVERALL SCORE: 77 PRICE: 9568168
33 Kalidou Koulibaly20 CB OVERALL SCORE: 77 PRICE: 9568168
34 Kalidou Koulibaly21 CB OVERALL SCORE: 77 PRICE: 9568168
35 Christian Eriksen MF OVERALL SCORE: 88 PRICE: 12086890
49 Christian Eriksen16 MF OVERALL SCORE: 88 PRICE: 12086890
51 Christian Eriksen18 MF OVERALL SCORE: 88 PRICE: 12086890
58 Christian Eriksen25 MF OVERALL SCORE: 88 PRICE: 12086890
69 Timo Werner6 ST OVERALL SCORE: 83 PRICE: 27349350
77 Timo Werner14 ST OVERALL SCORE: 83 PRICE: 27349350
Transfer list
3 Jan Oblak GK OVERALL SCORE: 86 PRICE: 3778889
4 Kevin Trapp GK OVERALL SCORE: 78 PRICE: 3097606
5 Yann Sommer GK OVERALL SCORE: 82 PRICE: 3377889
14 Kalidou Koulibaly CB OVERALL SCORE: 77 PRICE: 9568168
39 Christian Eriksen5 MF OVERALL SCORE: 88 PRICE: 12086890
40 Christian Eriksen6 MF OVERALL SCORE: 88 PRICE: 12086890
46 Christian Eriksen12 MF OVERALL SCORE: 88 PRICE: 12086890
47 Christian Eriksen13 MF OVERALL SCORE: 88 PRICE: 12086890
63 Christian Eriksen30 MF OVERALL SCORE: 88 PRICE: 12086890
66 Timo Werner3 ST OVERALL SCORE: 83 PRICE: 27349350
68 Timo Werner5 ST OVERALL SCORE: 83 PRICE: 27349350
75 Timo Werner12 ST OVERALL SCORE: 83 PRICE: 27349350
78 Timo Werner15 ST OVERALL SCORE: 83 PRICE: 27349350
Please type in player id from Your team to trade, or type 0 to exit:
```

Lista transferowa

## Specyfikacja wewnętrzna

Ze względu na dużą ilość klas osobno prezentuję model i osobno pozostałą część aplikacji (przepraszam za jakość, ale nie potrafiłem użyć programu do tworzenia UMLa).

Model:



Reszta klas:



W ramach tematów zajęć wykorzystano kontenery STL, algorytmy i iteratory STL, wątki (niestety nie `std::thread`, tylko `boost::thread` i `mutex`, ponieważ kompilator `gcc` nie współpracuje za dobrze z `std::thread`), oraz wyjątki. Kontenery STL, oraz algorytmy i iteratory zostały wykorzystane do trzymania danych – takich jak lista zawodników, lista klubów, lista meczów oraz wiele, wiele innych. Są obecne niemal w każdej funkcji w programie. Wątki wykorzystano do tego, aby zrównoleglić obliczenia dotyczące na przykład meczów – gdzie zrównoległono obliczenie siły zawodników dla całego zespołu. Zastosowałem także wyjątki do obsługi nieprawidłowych danych wejściowych użytkownika, oraz obsługi błędów bazy danych.

## Testowanie i uruchamianie

Testy programu były prawdziwą próbą charakteru – zwłaszcza pod koniec, gdy podczas testowania, czy kolejne sezony generują się poprawnie trzeba było rozegrać poprzednie. Przede wszystkim zmuszenie do działania bibliotek zewnętrznych – `sqlite3` oraz `boost` wymagało wielu testów, ze względu na brak `Visual Studio` i używania kompilatora `gcc`, który nie zawsze chciał współpracować. A i ja nie miałem z nim doświadczenia. Bardzo dużo początkowych błędów w programie dotyczyło tego, że przy wielokrotnym odczycie danych z bazy czasami przy kopiowaniu obiektów, czy ich pokazywaniu – podczas zapisu pojawiały się dziwne obiekty o id idącym w miliony, podczas gdy powinny mieć mniej niż 100. Na szczęście dzięki inteligentnemu zarządzaniu połączeniami z bazą udało się te błędy pokonać. Dużo pracy kosztowało także zbalansowanie algorytmu meczowego, aby dawał prawdopodobne wyniki, uwzględniając szczęście czy przewagę własnego stadionu. W tym pomogły funkcje randomizujące z nowego C++, które są oparte o rozkład normalny. W obecnej wersji program działa stabilnie (piszę obecnej, bo może go będę rozbudowywał – to fajny projekt).

## Uwagi i wnioski

Przy tworzeniu tego projektu patrzyłem z perspektywy programisty Javy i zdziwiło mnie, że C++ to język, w którym można tworzyć duże, skalowalne i czytelne aplikacje. Gdybym jeszcze znał obowiązujące wzorce projektowe w C++, myślę że mógłbym w nim tworzyć naprawdę fajne aplikacje, nie tylko projekty na studia. C++ kojarzył mi się z niezrozumiałą składnią i wskaźnikami na referencje czy innych rzeczy, których nie rozumie chyba nawet ich twórca, ale ze zdziwieniem odkryłem, że dzięki odpowiednim bibliotekom i uporządkowanej strukturze jest to naprawdę czytelne. Podczas pisania projektu bardzo mnie smucił brak szerokiego wsparcia dla baz danych (zwłaszcza tych sieciowych – np. PostgreSQL, czy MySQLa), oraz brak ORMa, ale zdaję sobie sprawę, że C++ nie jest zwykle wykorzystywany w ten sposób, aby powstanie ORMa było kluczowe.