POLITECHNIKA ŁÓDZKA

Wydział Elektrotechniki, Elektroniki,

Informatyki i Automatyki

|  |
| --- |
| Praca dyplomowa inżynierska **Opracowanie aplikacji wspierającej proces decyzyjny przy zakładach bukmacherskich z wykorzystaniem sztucznych sieci neuronowych** Artur Widziński |

# Nr albumu: 195085

Opiekun pracy:

**Dr inż. Wojciech Zabierowski**

## Łódź, 2018

**Streszczenie**

Blablabla….

**Abstract**

Blablabla…

**Spis treści**

1. Wstęp 3

1.1. Użyte technologie i frameworki 4

1.2. Skąd taki pomysł 6

2. Aplikacja webowa 8

2.1. Wzorzec MVC 8

2.2. Opis serwisu 9

2.3. Schemat bazy danych 11

2.4. Wygląd i działanie aplikacji 14

2.4.1. Główne menu 14

2.4.2. Panel użytkownika 18

2.4.3. Przygotowywanie meczów 20

3. Spis rysunków 22

4. Spis tabel 23

5. Bibliografia 24

1. Wstęp

W mojej pracy inżynierskiej zdecydowałem się na zrobienie aplikacji webowej, jaką jest serwis bukmacherski. W tej aplikacji postanowiłem wykorzystać realne drużyny z niemieckiej ligi rozgrywek piłkarskich. Ten projekt powstał w mojej głowie ponad pół roku temu. Stwierdziłem, że taka aplikacja może być przydatna dla ludzi, którzy się zastanawiają, czy warto zaczynać swoją przygodę z zakładami bukmacherskimi. Taka rozrywka jest jedną z najpopularniejszych form hazardu w dzisiejszych czasach. Hazard wiąże się z ryzykowaniem własnymi pieniędzmi. Zanim ktoś postawi swój pierwszy zakład w życiu, długo się zastanawia, jakie mecze obstawić, czy zaznaczyć mecze, w których szansa na wygranie jest duża, ale zysk niewielki, czy jednak te bardziej ryzykowne, które oferują większy możliwy profit. Różnorodność opcji, jak zakład może wyglądać jest tak wielka, że początkujący hazardzista będzie miał dużo problemów z tworzeniem opłacalnych kuponów, ponieważ będzie się bał zaryzykować postawienie swoich pieniędzy na zwycięstwo jednej z drużyn w wyrównanym meczu. Moja aplikacja może być dużą pomocą dla takich osób. Użytkownicy nie używają własnych pieniędzy. Ten element sprawia, że ten serwis bukmacherski nie powoduje żadnej presji na klientach.

Celem tego projektu nie jest przekonanie użytkownika, że zakłady bukmacherskie to najmniej ryzykowna inwestycja pieniężna i nie musi się ich bać, ale pokazanie mu, czy jego przewidywanie jest na tyle trafne, że może spokojnie odwiedzać stoiska bukmacherskie, albo wręcz przeciwnie, wyperswadowanie mu tego pomysłu z głowy.

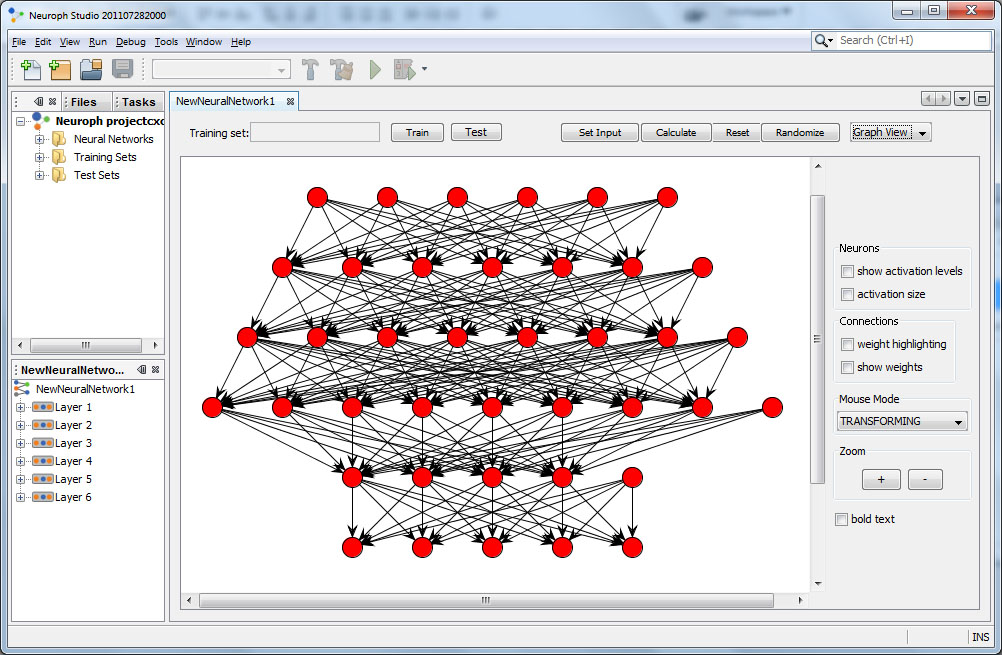
* 1. Użyte technologie i frameworki

**Spring Framework** jest szkieletem dla aplikacji napisanych w języku **JAVA EE** (Enterprise Edition). Ten projekt był moją pierwszą stycznością z tym frameworkiem. Dlatego zacząłem szukać różnych poradników, tutoriali w Internecie, jak stworzyć przykładową aplikację napisaną za pomocą wyżej wymienionej technologii. W trakcie tych poszukiwań znalazłem informacje o **Spring Boot**. Jest to rozwiązanie oferowane przez twórców technologii Spring, oparte na tym frameworku, w którym konfiguracja jest znacząco ułatwiona w porównaniu do konfiguracji aplikacji napisanych w „czystym” Springu. Ta zaleta przekłada się na szybkość pisania kodu. Dlatego zdecydowałem się stworzyć ten projekt właśnie w tej technologii.

Podczas nauki tego Framework korzystałem z poradnika znalezionego z Internecie, jak stworzyć podstawową aplikację w tym frameworku[[1]](#footnote-2). Dzięki temu tutorialowi nauczyłem się również korzystać z szablonów **Thymeleaf** oraz frameworka **Spring Security**. Wspomniane szablony generują kod HTML5, za pomocą którego wyświetlam dane zaciągnięte z bazy danych. Jest to możliwe, ponieważ twórcy tej biblioteki zapewnili możliwość integrowania tych szablonów z frameworkiem Spring. Spring Security udostępnia możliwość autoryzacji i autentykacji użytkowników korzystających z tego serwisu. Autoryzacja polega na sprawdzeniu, czy dany użytkownik posiada uprawnienia do wykonania określonych operacji. Celem autentykacji jest sprawdzenie, czy użytkownik, za którego się uważamy rzeczywiście istnieje w systemie (bądź bazie danych). W moim projekcie zapewniłem dostęp do większości stron tylko dla użytkowników, którzy mają określone role (administrator, klient). Każda rola posiada określone uprawnienia, które są weryfikowane przez Framework Spring Security.

Zastanawiając się, którą bazę danych wybrać kierowałem się ilością wpisów w Internecie, które pomagały czy to w rozwijaniu aplikacji, czy to w rozwiązywaniu problemów[[2]](#footnote-3). W taki sposób zdecydowałem się na bazę **MySQL**.

Do zaimplementowania sztucznej inteligencji użyłem biblioteki **Neuroph**. Jest to Framework do tworzenia sieci neuronowych. Został on stworzony przez twórców IDE Netbeans. Początkowo istniało osobne IDE do tworzenia różnych architektur sieci neuronowych, zwane Neuroph Studio. Posiada ono graficzny interfejs użytkownika, za pomocą którego można ustawić wszelkie parametry sieci.



Rysunek 1: Widok Neuroph Studio**[[3]](#footnote-4)**

* 1. Skąd taki pomysł

Od najmłodszych lat interesowałem się piłką nożną. Kiedy na studiach informatycznych dostaliśmy dowolność wyboru tematu projektu, który mamy stworzyć na dany przedmiot, nie zastanawiałem się długo i zdecydowałem się napisać program związany z piłką nożną. Na początku to były zwykłe programy, w których wpisywało się wynik meczu, a aplikacja pokazywała jak wygląda tabela ligi lub drabinka turniejowa. Wtedy błędnie nazywałem te projekty „symulatorami”. Dopiero na jednych zajęciach na naszej uczelni zrozumiałem znaczenie tego słowa. Doktor prowadzący przedmiot powiedział mi, że taki program piłkarski, który jest symulatorem powinien sam ustalać wynik meczu. Pierwszą moją myślą wtedy było zwykłe losowanie liczby z jakiegoś przedziału, ale szybko odrzuciłem ten pomysł. Wyniki meczu uzyskane w taki sposób nie miałyby za wiele wspólnego z rzeczywistością. Ten sam Doktor poradził mi zapoznanie się z tematem sieci neuronowych. One mogłyby się nauczyć przewidywać wynik meczu, jeśli dostałyby odpowiednie dobrane dane dotyczące poprzednich gier.

Na jednych z przedmiotów mieliśmy stworzyć aplikację internetową (z użyciem technologii Spring) z relacyjną bazą danych. Wiedziałem, że chcę stworzyć aplikację związaną z piłką nożną, jednak nie chciałem po raz kolejny robić tej samej aplikacji, chciałem ją jakoś rozwinąć, dodać element biznesowy. Wtedy przyszedł mi do głowy pomysł serwisu bukmacherskiego.

Wiele osób w moim otoczeniu reagowało negatywnie na mój pomysł, bo obstawianie zakładów bukmacherskich to hazard, a to słowo jest odbierane na ogół negatywnie. Ja jednak nie patrzyłem na ten pomysł w ten sposób.

Podczas pisania tego projektu polubiłem technologię Spring Boot, i szkoda mi było zostawiać ten projekt. Dlatego chciałem go użyć w pracy inżynierskiej. Kiedy się zastanawiałem, jak go rozwinąć, przypomniał mi się temat sieci neuronowych. Sam byłem bardzo ciekawy dwóch wyników tego projektu:

* Czy da się przewidzieć za pomocą sztucznej inteligencji rezultat meczu?
* Jaki dochód można uzyskać prowadząc taki serwis?

Każdy z tych problemów opisałem w dalszej części tej pracy. Wiedziałem, że aby sztuczna inteligencja mogła jak najlepiej przewidzieć wynik meczu, musiałem znacznie rozbudować istniejącą bazę danych. Jak się później okazało, był to jednak duży problem.

Na początku szukałem darmowego dostępnego API, które oferuje wszystkie potrzebne dla mojego projektu dane. Niestety, nie znalazłem żadnej strony, która oferowałaby pożądane informacje bez żadnych opłat. Dlatego postanowiłem zebrać dane w inny sposób. Z użyciem program **WebHarvy** można odczytać dane ze strony (inaczej mówiąc „zescrappować”) i zapisać je w dowolnym formacie (np. XML, JSON, CSV). Do tej operacji skorzystałem ze strony transfermarkt[[4]](#footnote-5). Pomimo tego, że na początku wydawało się to dobrym rozwiązaniem, to równie szybko okazało się, że nie będę w stanie zapisać wszystkich danych przed końcem semestru. Dlatego pozostałe dane pobrałem z płatnego API[[5]](#footnote-6), które zawierało potrzebne informacje.

Kiedy już miałem wszystkie niezbędne pliki zapisane, kolejnym problemem okazał się alfabet. Dane, które pobrałem z API nie posiadały wszystkich znaków specjalnych, przez co piłkarz, który nazywał się *„Jérôme Boateng”* był innym zawodnikiem niż *„Jerome Boateng”*. Poprawienie tych nieścisłości również zajęło trochę czasu.

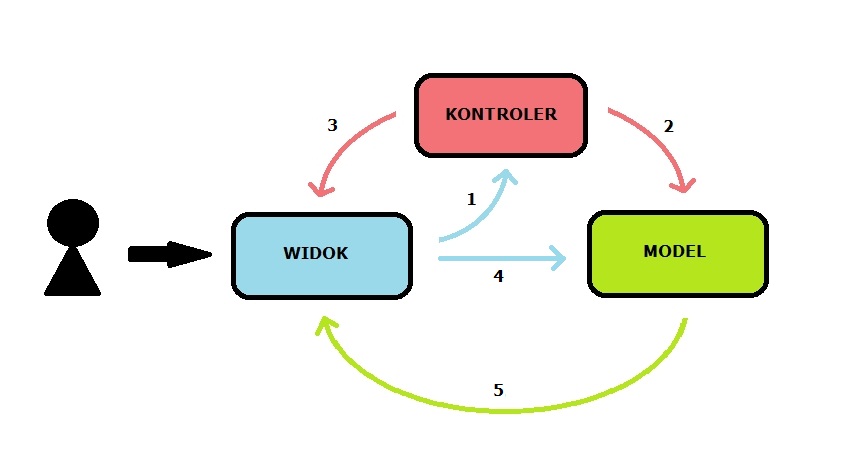
1. Aplikacja webowa

W tym rozdziale zostanie opisana aplikacja webowa, jej funkcjonalności dla klienta i administratora oraz sposób działania.

* 1. Wzorzec MVC

Projekt serwisu bukmacherskiego został zaprojektowany zgodnie z wzorcem projektowym Model-View-Controller (MVC). To rozwiązanie można podzielić na 3 osobne części:

* Model – element logiczny aplikacji
* Widok – miejsce prezentacji danych za pomocą graficznego interfejsu
* Kontroler – element odpowiadający za wykonywanie operacji na danych wejściowych, które wywołuje użytkownik dzięki graficznemu interfejsowi, i następne przesłanie zaktualizowanych modelów do warstwy widokowej



Rysunek : Schemat wzorca projektowego MVC[[6]](#footnote-7)

Taki wzorzec projektowy jest jedną z najczęściej wykorzystywanych koncepcji podczas tworzenia aplikacji webowych.

* 1. Opis serwisu

Strona bukmacherska jest aplikacją skierowaną dla ludzi interesujących się rozgrywkami sportowymi. Wszystkich użytkowników aplikacji można podzielić na kilka grup, którym przypisane są poniższe role:

* Administrator (inaczej opiekun serwisu)
* Zalogowany użytkownik
* Niezalogowany użytkownik

Administrator jest główną osobą, która czuwa nad poprawnością działania serwisu oraz wprowadza dane związane z nadchodzącymi meczami. Zalogowany użytkownik tworzy kupony, za pomocą których zakłada przewidywany rezultat meczu. Większość usług jest dostępnych dla wszystkich zalogowanych osób, jednak jako użytkownik niezalogowany mamy również kilka możliwości.

Oto najważniejsze funkcjonalności przeznaczone dla administratora:

* Stworzenie meczu
* Dodanie składu drużyny do meczu
* Rozegranie meczu

Serwis bukmacherski jest stroną, która musi mieć dostęp do dużej ilości danych na temat rozgrywanych meczów. Jest to ważne z punktu widzenia klienta, który decyduje się, czy postawić zakład na daną drużynę. Chętnie sprawdzi np. jakie wyniki klub osiągał w przeciągu kilku ostatnich gier, jakich piłkarzy posiada czy też na którym miejscu w lidze się znajduje. Dlatego aplikacja oferuje dużą ilość różnych możliwości skierowanych głównie dla klienta. Oto niektóre z nich:

* Sprawdzenie tabeli ligi
* Sprawdzenie danych drużyny
* Sprawdzenie poprzednich wyników klubu
* Zobaczenie detali meczu

Z materialnego punktu spojrzenia udostępniono w aplikacji funkcjonalności odpowiadające za zarządzanie kuponami. Klient może sprawdzić jak wygląda jego sytuacja finansowa, a administrator może sprawdzić stan konta każdego zalogowanego użytkownika. Takie usługi są dostępne w tym celu:

* Doładowanie konta
* Stworzenie kuponu
* Sprawdzenie stanu konta
  1. Schemat bazy danych

W bazie danych tej aplikacji znajdują się tabele, w których aplikacja przechowuje dane, a także dodaje, modyfikuje i usuwa kolejne rekordy. Występują tam również tabele, które przechowują dane, ale aplikacja z nich nie korzysta. Związane jest to z tym, że aplikacja będzie jeszcze rozwijana.

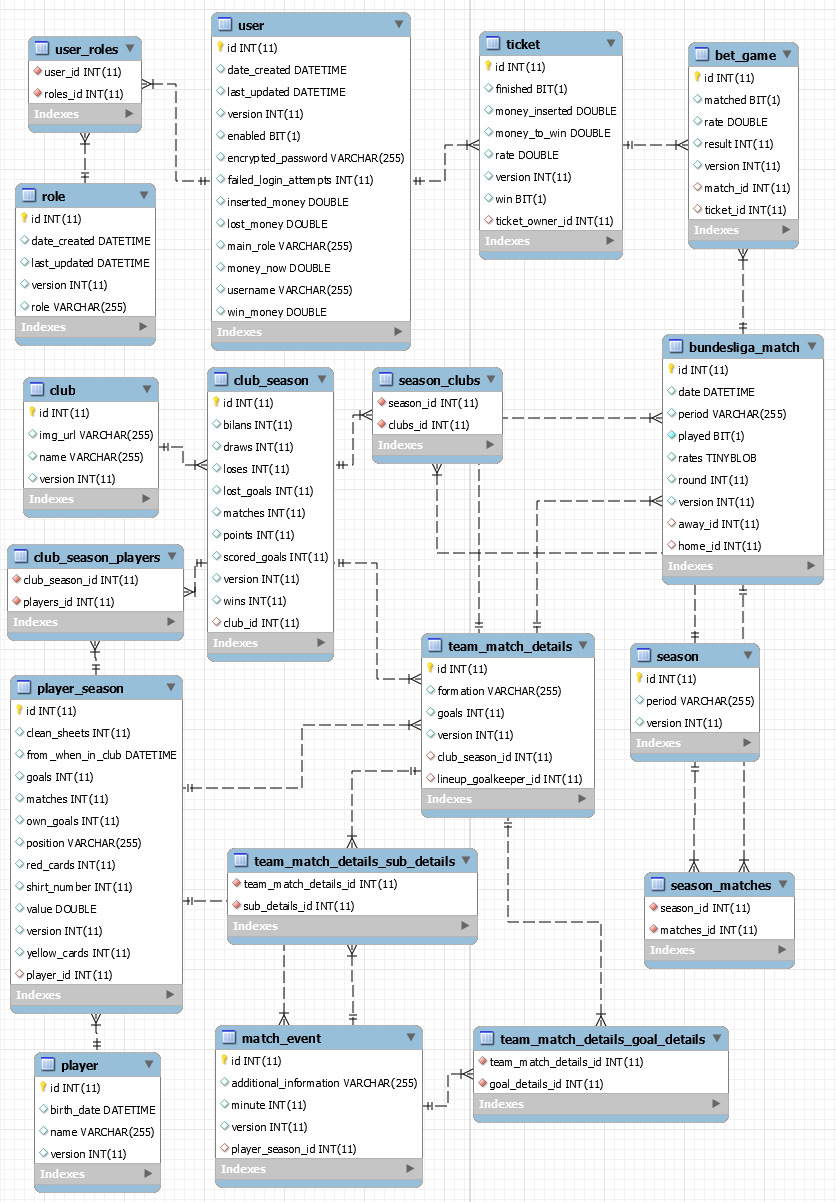
Podstawową tabelą w tej aplikacji jest tabela **USER**. Przechowuje ona dane użytkownika, takie jak nazwa użytkownika, zaszyfrowane hasło, ilość pieniędzy na koncie czy rolę użytkownika. Domyślnie po rejestracji każdy użytkownik posiada rolę *„user”*.

Każdy użytkownik ma przypisaną listę kuponów, które kiedyś stworzył. Trzymane są one w tabeli **TICKET**. Każdy kupon przechowuje informacje między innymi o tym ile użytkownik postawił pieniędzy, ile może wygrać oraz czy dany kupon został już rozstrzygnięty. Dodatkowo do każdego z nich jest przypisanych kolekcja tzw. singli, inaczej mówiąc zakładów pojedynczych, singlowych. Jest to zdarzenie, które dotyczy pojedynczego meczu. W zdarzeniu zapisane są informacje o tym jaki rezultat użytkownik przewidział, i czy się zgadzał z rozegraną już grą. Te dane znajdują się w tabeli **BET\_GAME**.

Najbardziej zewnętrzną tabelą trzymającą informacje o rozgrywkach jest tabela **SEASON**. Każdy rekord przechowywał lata trwania, mecze wtedy rozegrane oraz kluby, które w nim uczestniczyły. Rekordy z tabeli **CLUB\_SEASON** przechowują dane o tym jak drużyny spisywały się w danym sezonie (ilość wygranych, przegranych, remisów, strzelonych i straconych bramek). Dodatkowo do każdej encji z tej tabeli przypisana jest lista zawodników, którzy wtedy należeli do danego klubu. Każdy z nich jest zapisany w tabeli **PLAYER\_SEASON** i dodatkowo zawiera zarówno informacje personalne piłkarza (imię i nazwisko, data urodzenia) jak i dane piłkarskie (numer na koszulce, pozycja, wartość rynkowa, ilość meczów, strzelonych bramek oraz meczów bez traconej bramki bramkarzy).

Tabela **BUNDESLIGA\_MATCH** przechowuje encje wszystkich meczy, które się już odbyły oraz tych, które zostały już zaplanowane. Można tutaj znaleźć datę danej rozgrywki, wyliczone kursy na dany mecz oraz dane składów obu drużyn, które są trzymane w tabeli **TEAM\_MATCH\_DETAILS**. Każdy rekord zawiera informacje dotyczące piłkarzy składu wyjściowego oraz ławki rezerwowych, ilość strzelonych goli, a także zdarzenia w meczu dotyczące drużyny. Te ostatnie znajdziemy w tabeli **MATCH\_EVENT**. Jest tam zapisana minuta wydarzenia, piłkarz, którego ono dotyczy oraz dodatkowa krótka informacja, która dokładnie opisuje co to za rodzaj zdarzenia (np. gol, zmiana, samobój).

Bardziej szczegółowy schemat tej bazy danych przedstawia rysunek na następnej stronie.



Rysunek 3: Schemat bazy danych

* 1. Wygląd i działanie aplikacji

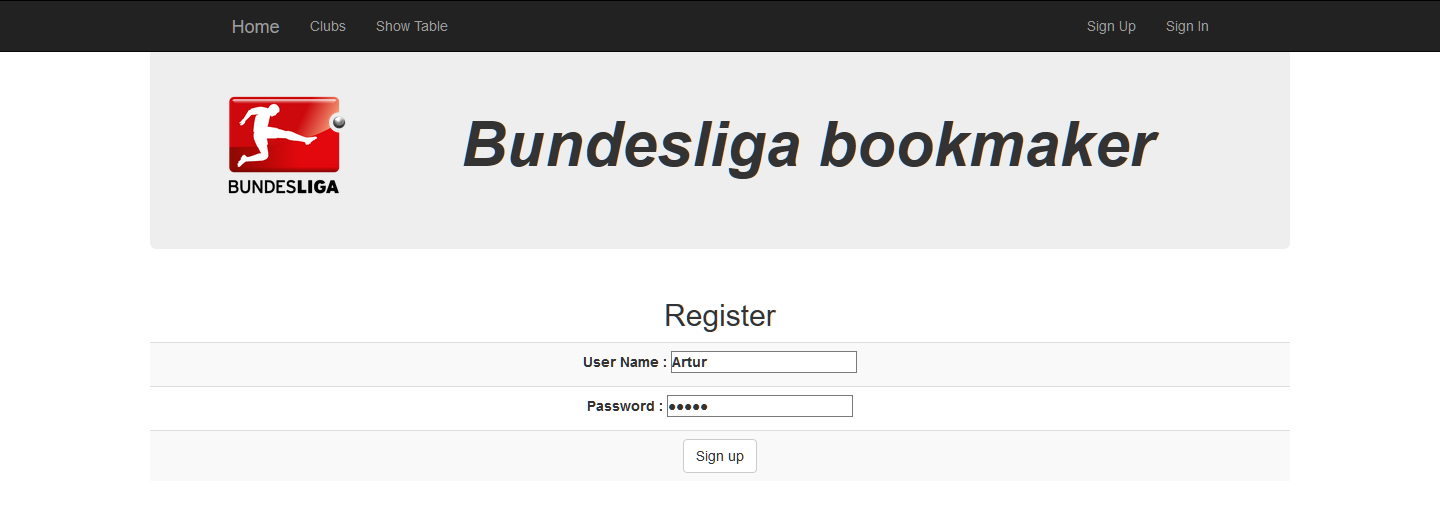
W tym podrozdziale zostaną przedstawione oraz omówione poszczególne strony aplikacji.

* + 1. Główne menu

Po uruchomieniu serwisu zostaje pokazana główna strona serwisu.

Rysunek : Strona główna

W tym momencie użytkownik nie jest zalogowany. Dla serwisu oznacza to, że dany użytkownik nie jest zautoryzowany, czyli nie ma dostępu do większości usług. Aby to zmienić, klient musi się zarejestrować, a następnie zalogować.

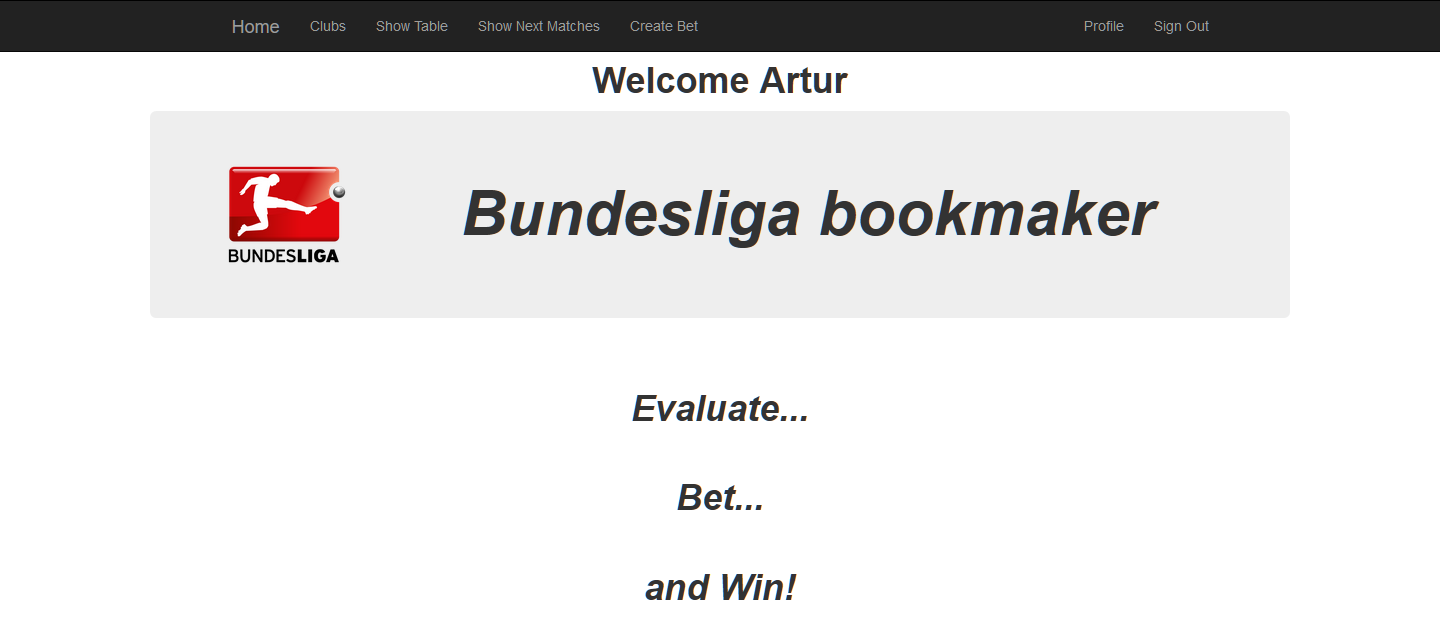


Rysunek : Ekran rejestracji

##### Logowanie.png

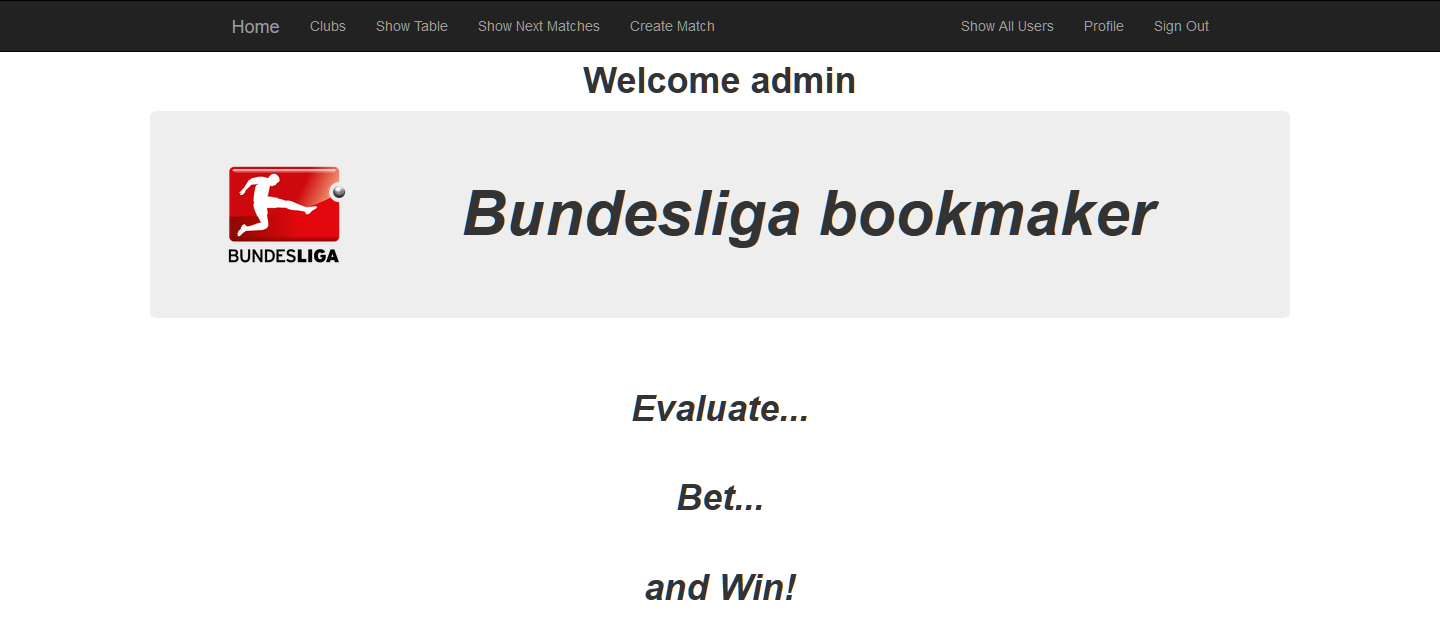
Rysunek : Ekran logowania

Po poprawnym zalogowaniu przed użytkownikiem ponownie pojawia się główna strona. Jednak tym razem system rozpoznaje użytkownika jako osobę uwierzytelnioną, dlatego udostępnia mu więcej możliwości.



Rysunek : Ekran główny zwykłego użytkownika po zalogowaniu się

Zmiany można zauważyć w górnym pasku menu. Uwierzytelniony użytkownik może teraz skorzystać z takich zakładek jak „Show next matches”, „Create bet” czy „Profile”. Nowy użytkownik domyślnie należy do grupy klientów z rolą *„user”*. Natomiast administrator systemu jest jedyną osobą z rolą *„admin”*. Jego ekran główny posiada kilka funkcji, do których zwykły użytkownik nie ma dostępu.



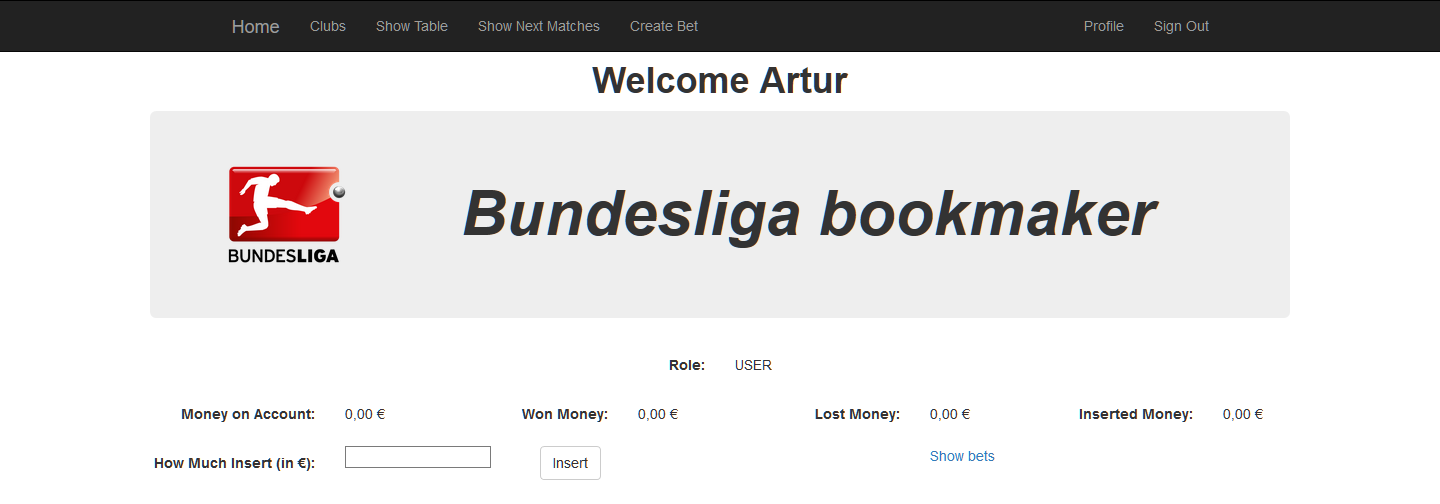
Rysunek : Ekran główny administratora po zalogowaniu się

Na powyższym rysunku można zauważyć, że administrator systemu ma kilka dodatkowych funkcji, np. „Create match” czy też „Show all users”. Taką różnicę w dostępności usług w serwisie uzyskano dzięki autoryzacji. Proces ten pozwala na rozpoznanie roli danego użytkownika i na tej podstawie decyduje, z których możliwości strony może on skorzystać.

Uwierzytelnianie i autoryzacje użytkowników można uzyskać dzięki bibliotece Spring Security. W projekcie serwisu bukmacherskiego te dwa procesy są wykorzystywane na każdej stronie. Ten framework zabezpiecza serwis również przed wchodzeniem na strony niedostępne dla użytkownika o niższej w hierarchii serwisu roli. Strona bukmacherska posiada ochronę w sytuacji, w której jeden użytkownik próbuje wcielić się w drugiego.

* + 1. Panel użytkownika

Zalogowany klient ma dostęp do wglądu w swój profil. Po naciśnięciu przycisku *„Profile”* GUI (graficzny interfejs użytkownika) przekazuje do kontrolera żądanie wykonania usługi. W tym celu kontroler pobiera z bazy danych informacje o tym użytkowniku, a następnie przekazuje je do warstwy widokowej, która wyświetla wszystko w określony sposób (Rysunek 9). W taki sposób wygląda komunikacja klient-serwer w tej aplikacji.



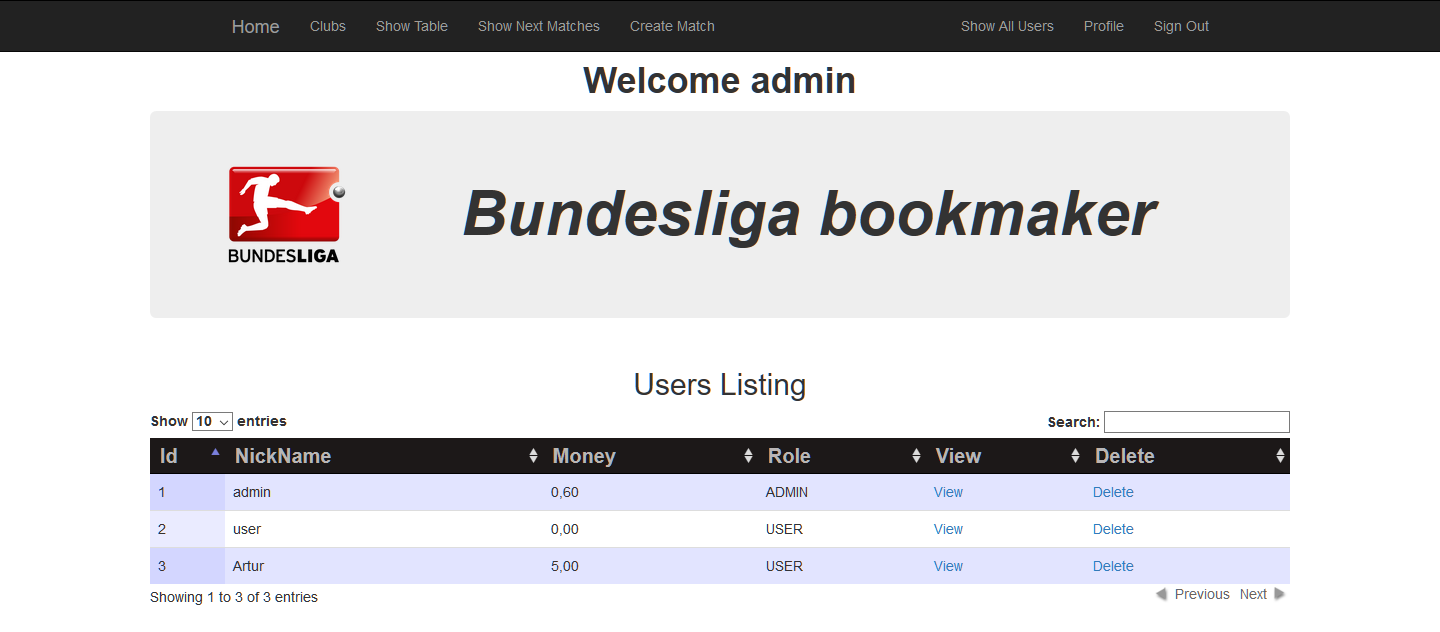
Rysunek : Profil użytkownika

Tutaj znajdują się informacje o danym kliencie. Może on zobaczyć do jakiej grupy użytkowników należy (czyli jakie ma uprawnienia), sprawdzić stan swojego konta pieniężnego oraz zobaczyć historię własnych kuponów.

W tym serwisie bukmacherskim nie używa się własnych pieniędzy, zgodnie z tym, co autor wcześniej napisał. Ta aplikacja jest dla klientów sprawdzianem swoich umiejętności, próbą generalną przed postawieniem pierwszego zakładu bukmacherskiego za pomocą finansów z własnego konta bankowego. Aby użytkownik mógł doładować swoje konto, wystarczy że wpiszę w pole tekstowe liczbę, i naciśnie przycisk *„Insert”*. Po tej operacji profil klienta zostanie zaktualizowany, zwiększy się ilość pieniędzy wyświetlana przy tekstach *„Money on Account”* oraz *„Inserted Money”*.

Nie każdego stać, żeby na każdy kupon poświęcać duże kwoty pieniężne. Dlatego ciągłe doładowywanie konta tysiącami złotych może odebrać realizm z zakładów bukmacherskich. Takie zachowanie będzie jednak złe tylko dla klienta. Jeśli nawet w tym serwisie uda mu się zarobić więcej pieniędzy niż wpłacił za pomocą jednego kuponu o wpisowej kwocie np. milion złotych, to po rozpoczęciu przygody z realnymi zakładami szybko zobaczy, że takiego samego zysku prawie na pewno nie osiągnie.

Jako administrator mamy dostęp do profili wszystkich użytkowników.

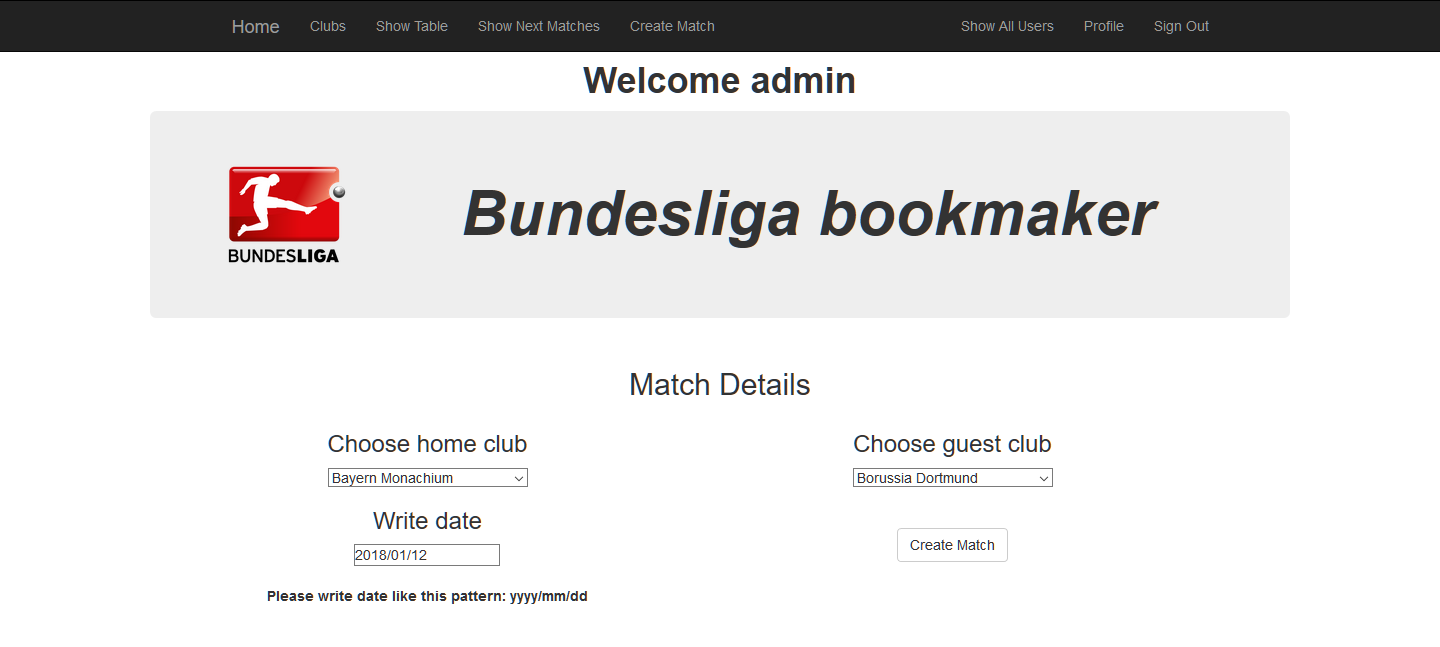


Rysunek : Widok wszystkich użytkowników

W tym panelu można zobaczyć ilość aktualnych pieniędzy na koncie klientów. Istnieje tutaj możliwość usunięcia użytkownika, a także sprawdzenia tych samych danych, które były pokazane na poprzednim rysunku (Rysunek 9). Jedyna różnica w wyświetleniu tej strony polega na tym, iż administrator nie może doładować konta żadnego z użytkowników.

* + 1. Przygotowywanie meczów

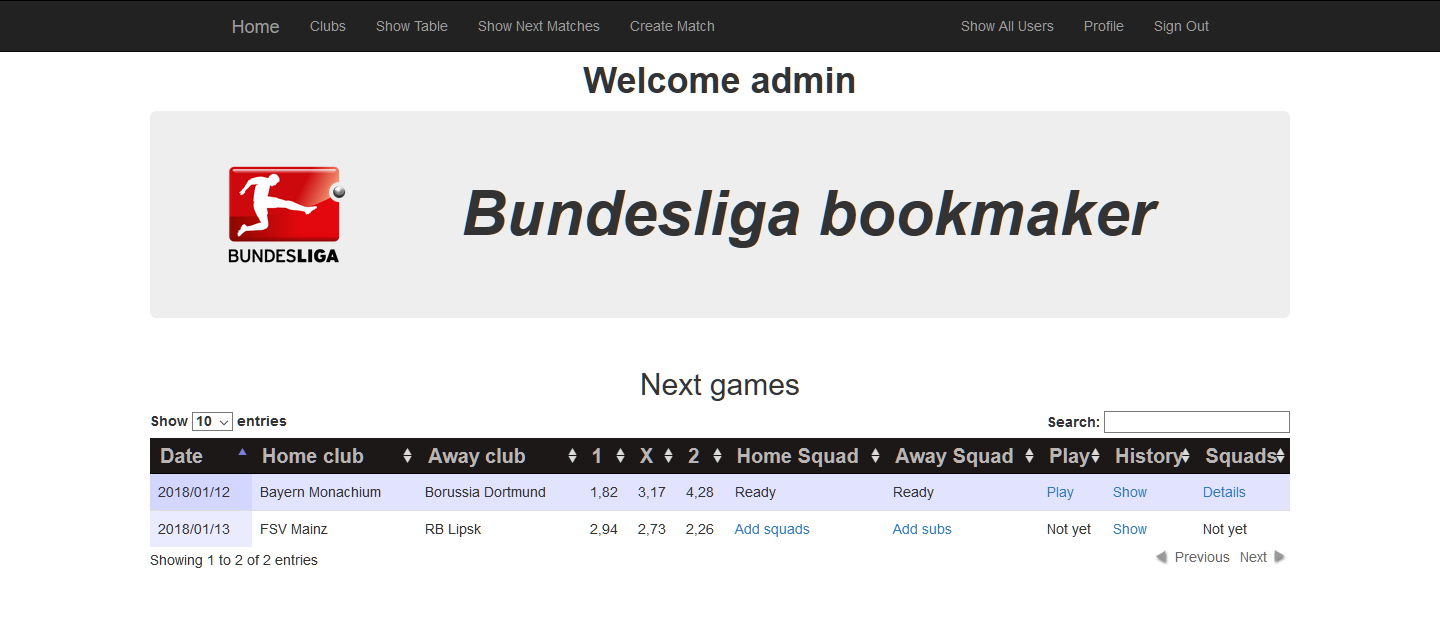
Kiedy po raz pierwszy zostanie naciśnięty przycisk *„Show next matches”*, pojawi się nowe okno, na którym jest napisana informacja o braku w bazie danych meczów nierozegranych. Aby to zmienić, musi zostać ogłoszona następna gra. W tym serwisie bukmacherskim tylko administrator posiada taką możliwość. Po wybraniu przycisku *„Create match”* z górnego menu strona wyświetla taką zawartość, jak w rysunku poniżej (Rysunek 11).



Rysunek : Tworzenie meczu

W tym panelu opiekun serwisu wybiera które drużyny zagrają ze sobą w tym meczu. Nie jest obojętne który klub zostanie wybrany pod polem *„Choose home club”* i *„Choose away club”*, ponieważ algorytmy liczące kursy na dany mecz i symulujące daną grę operują między innymi na tym, która drużyna gra na swoim stadionie (ich szczegółowy opis zostanie opisany w kolejnych rozdziałach).

Po wpisaniu daty i naciśnięciu przycisku *„Create match”* aplikacja sprawdzi poprawność przesłanych danych (np. czy data jest podana w odpowiednim formacie, bądź klub nie gra sam ze sobą), następnie algorytm wyliczy kursy na ten mecz, a na końcu serwis przekieruje użytkownika do strony z nierozegranymi meczami (Rysunek 12).

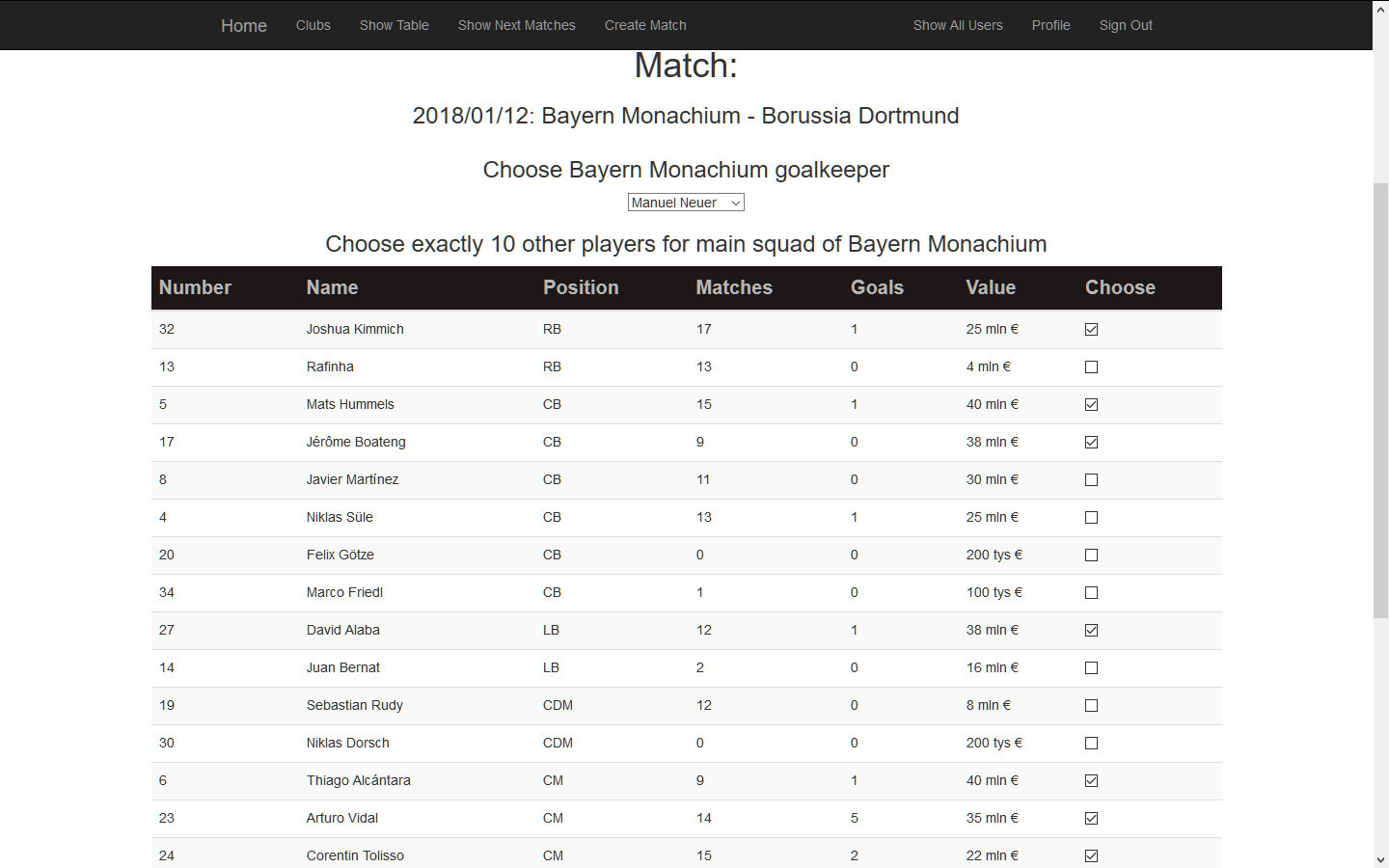


Rysunek : Następne mecze

Na tej stronie widać tabelę z dużą ilością kolumn. Pierwsze trzy z lewej zostały wybrane w poprzednim kroku (Rysunek 11). Pozostałe z nich:

* 1 – kolumna pokazująca kurs na wygranie drużyny gospodarzy
* X – kolumna pokazująca kurs na remis w tym spotkaniu
* 2 – kolumna pokazująca kurs na wygranie drużyny gości
* Home Squad – kolumna widoczna tylko dla administratora, znajduje się tutaj informacja, czy skład wyjściowy i rezerwowy drużyny gospodarzy został już dodany i zapisany w bazie danych. Może tutaj się znaleźć jedna z opcji:
  + Add squads – link do dodania piłkarzy składu wyjściowego
  + Add subs – link do dodania piłkarzy z ławki rezerwowych
  + Ready – informacja potwierdzająca zapisanie wszystkich wymaganych informacji na temat drużyny gospodarzy w tym meczu
* Away Squad – analogicznie do kolumny „Home Squad”, dotyczy drużyny gości
* Play – kolumna widoczna tylko dla administratora, mówiąca czy można już zasymulować wynik tego meczu. Ta kolumna może zawierać:
  + Not yet – co najmniej jedna z dwóch drużyn nie jest jeszcze gotowa do rozegrania meczu (nie ma w kolumnie „Home/Away Squad” napisane „Ready”)
  + Play – link do strony rozgrywającej ten mecz
* History – kolumna zawierająca link, który prowadzi do strony wyświetlającej historię meczów rozgrywanych pomiędzy tymi dwoma drużynami
* Squads – kolumna pokazująca dostępność zobaczenia składów wyjściowych oraz ławek rezerwowych. Zawiera ona pole:
  + Not yet – tak samo jak pole „Not yet” w kolumnie „Play”
  + Details – link do strony pokazującej składy obu drużyn na to spotkanie

Kolejnym etapem w przygotowywaniu meczu jest dodanie piłkarzy, którzy zagrają od początku spotkania na boisku. Do tego używa się przycisku *„Add Squad”*.



Rysunek : Wybieranie graczy składu wyjściowego

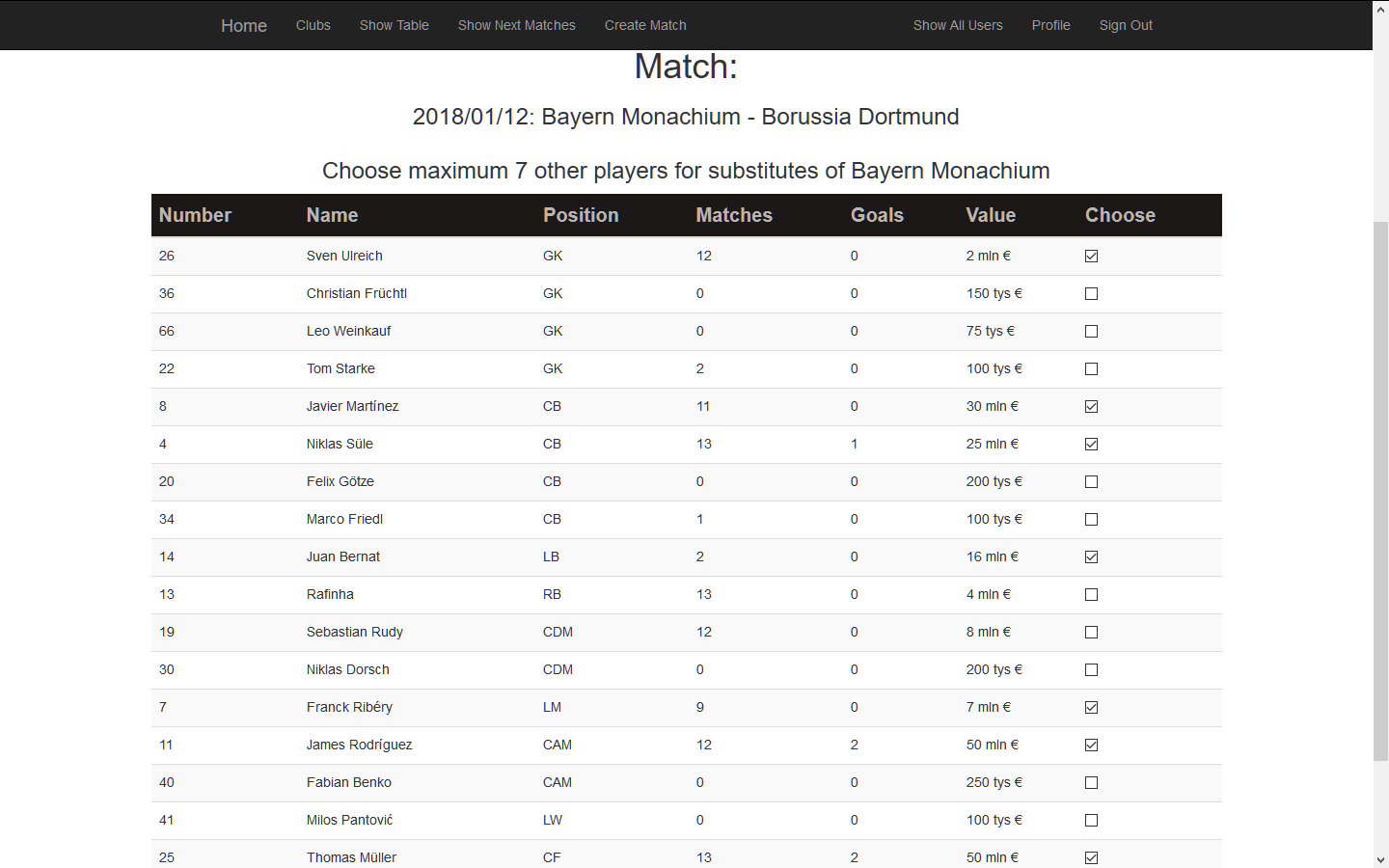
Na ekranie pojawili się gracze z drużyny, którą opiekun wybrał w poprzednim oknie. Aplikacja wyciągając ich z bazy danych podzieliła ich na dwie grupy: piłkarze grający na pozycji bramkarza i pozostali zawodnicy. Ten zabieg zapewnił to, iż w każdym podstawowym składzie będzie co najmniej jeden bramkarz. Z listy rozwijanej administrator wybiera zawodnika z tej pozycji. Niżej znajdują się wszyscy pozostali piłkarze tej drużyny. W tej tabeli znajdują się kolejno numer na koszulce, imię i nazwisko zawodnika, podstawowa pozycja na boisku, ilość rozegranych meczy w sezonie, ilość goli w sezonie oraz wartość rynkowa tego gracza. Pozycja każdego z nich jest zapisana w skrócie, którego rozwinięcie można znaleźć w poniższej tabeli.

Tabela : Rozwinięcie skrótów pozycji na boisku

|  |  |  |
| --- | --- | --- |
| Skrót | Angielskie rozwinięcie | Polskie znaczenie |
| GK | Goalkeeper | Bramkarz |
| RB | Right Back | Prawy obrońca |
| CB | Center Back | Środkowy obrońca |
| LB | Left Back | Lewy obrońca |
| CDM | Center Defensive Midfield | Defensywny pomocnik |
| RM | Right Midfield | Prawy pomocnik |
| CM | Center Midfield | Środkowy pomocnik |
| LM | Left Midfield | Lewy pomocnik |
| CAM | Center Attacking Midfield | Ofensywny pomocnik |
| RW | Right Wing | Prawy skrzydłowy |
| LW | Left Wing | Lewy skrzydłowy |
| CF | Center Forward | Cofnięty napastnik |
| ST | Striker | Środkowy napastnik |

Spośród wszystkich wypisanych piłkarzy administrator wybiera dziesięciu, którzy rozpoczną mecz zaznaczając okienko po prawej stronie, a na końcu wybiera przycisk *„Choose”*. Wtedy aplikacja zapisuje wybranego bramkarza oraz zaznaczonych piłkarzy w bazie, przekierowywuje do poprzedniej strony, a tam, gdzie wcześniej opiekun kliknął przycisk *„Add Squad”* zmieniła się pozycja na *„Add subs”*.

Następnym etapem przygotowania drużyn do meczu jest wybranie piłkarzy, którzy usiądą na ławce rezerwowych.



Rysunek : Wybieranie graczy do ławki rezerwowych

Ten etap wygląda prawie tak samo jak wybór graczy podstawowego składu, z jedną różnicą. Tutaj aplikacja wyciągając piłkarzy z bazy danych odrzuciła wszystkich tych, którzy już są dodani do tego meczu, i nie robi podziału na bramkarzy i pozostałych zawodników. Stało się tak, ponieważ według zasad rozgrywek piłki nożnej w Niemczech nie ma ustalonej liczby ile maksymalnie bramkarzy może być wybranych do rezerwy.

Jedyne ograniczenie jakie nakładają przepisy, to łączna liczba piłkarzy na ławce. Maksymalna ilość takich zawodników nie może być większa niż siedmiu.

Po wybraniu odpowiednich piłkarzy administrator zatwierdza swój wybór, a aplikacja go przenosi do strony z nierozegranymi meczami. W kolumnie, gdzie opiekun wcześniej nacisnął przycisk widnieje teraz napis *„Ready”*. Oznacza to, że ta drużyna jest już gotowa do rozegrania meczu. Po wykonaniu tych samych operacji dla drugiej drużyny w kolumnach „Play” oraz „Squads” aplikacja odblokuje możliwość rozegrania meczu oraz zobaczenia składów obu drużyn. Przebieg pierwszej z nich zostanie omówiony w jednym z kolejnych podrozdziałów.

1. Spis rysunków

*Rysunek 1: Widok Neuroph Studio 5*

*Rysunek 2: Schemat wzorca projektowego MVC 8*

*Rysunek 3: Schemat bazy danych 13*

*Rysunek 4: Strona główna 14*

*Rysunek 5: Ekran rejestracji 15*

*Rysunek 6: Ekran logowania 15*

*Rysunek 7: Ekran główny zwykłego użytkownika po zalogowaniu się 16*

*Rysunek 8: Ekran główny administratora po zalogowaniu się 16*

*Rysunek 9: Profil użytkownika 18*

*Rysunek 10: Widok wszystkich użytkowników 19*

*Rysunek 11: Tworzenie meczu 20*

1. Spis tabel
2. Bibliografia
3. Poradnik jak stworzyć podstawową aplikację za pomocą frameworka Spring Boot <https://springframework.guru/spring-boot-web-application-part-1-spring-initializr/> maj 2017
4. Konfiguracja aplikacji napisanej w Springu pod bazę danych MySQL <https://spring.io/guides/gs/accessing-data-mysql/>, październik 2017
5. Widok IDE Neuroph Studio <http://neuroph.sourceforge.net/images/screenshot-big.jpg> grudzień 2017
6. <https://www.transfermartk.pl/> listopad 2017
7. Strona z API <http://www.xmlsoccer.com/> styczeń 2018
8. Schemat wzorca projektowego <http://blog.karolak.it/wp-content/uploads/2010/11/cakephp_mvc.jpg> styczeń 2018

1. Poradnik jak stworzyć podstawową aplikację za pomocą frameworka Spring Boot <https://springframework.guru/spring-boot-web-application-part-1-spring-initializr/> maj 2017 [↑](#footnote-ref-2)
2. Konfiguracja aplikacji napisanej w Springu pod bazę danych MySQL: <https://spring.io/guides/gs/accessing-data-mysql/>, maj 2017 [↑](#footnote-ref-3)
3. Widok IDE Neuroph Studio <http://neuroph.sourceforge.net/images/screenshot-big.jpg> grudzień 2017 [↑](#footnote-ref-4)
4. <https://www.transfermarkt.pl/> listopad 2017 [↑](#footnote-ref-5)
5. Strona z API <http://www.xmlsoccer.com/> styczeń 2018 [↑](#footnote-ref-6)
6. Schemat wzorca projektowego MVC <http://blog.karolak.it/wp-content/uploads/2010/11/cakephp_mvc.jpg> styczeń 2017 [↑](#footnote-ref-7)