**Politechnika Krakowska**

*im. Tadeusza Kościuszki*

**Wydział Inżynierii Elektrycznej i Komputerowej**

**Instytut Elektrotechniki i Informatyki**

Arkadiusz Junik

**Aplikacja dla fanów piłki nożnej umożliwiająca rywalizację poprzez typowanie wyników wydarzeń sportowych.**

Praca dyplomowa inżynierska

Praca wykonana pod kierunkiem

dr inż. Radosława Czarneckiego

**Kraków 2017**

Spis treści

[O pracy inżynierskiej 4](#_Toc469516071)

[Wykorzystane technologie 4](#_Toc469516072)

[Model MVC 4](#_Toc469516073)

[Model 5](#_Toc469516074)

[PostgreSQL 5](#_Toc469516075)

[Hibernate 5](#_Toc469516076)

[Widok 6](#_Toc469516077)

[HTML, CSS, Javascript 6](#_Toc469516078)

[Thymeleaf 6](#_Toc469516079)

[Bootstrap 6](#_Toc469516080)

[D3.js 6](#_Toc469516081)

[Kontroler 7](#_Toc469516082)

[Java 7](#_Toc469516083)

[Spring 8](#_Toc469516084)

[Serwer aplikacji 9](#_Toc469516085)

[Charakterystyka użytkowników 9](#_Toc469516086)

[Wymagania 10](#_Toc469516087)

[Wymagania funkcjonalne 10](#_Toc469516088)

[Wymagania niefunkcjonalne 10](#_Toc469516089)

[Scenariusze testowe 10](#_Toc469516090)

[Diagramy UML 10](#_Toc469516091)

[Diagram przypadków użycia 10](#_Toc469516092)

[Diagram STD 10](#_Toc469516093)

[Diagram ETL 10](#_Toc469516094)

[Baza danych 10](#_Toc469516095)

[Opis bazy danych 10](#_Toc469516096)

[Diagram ERD 10](#_Toc469516097)

[Instrukcje 10](#_Toc469516098)

[Instrukcja uruchomienia 10](#_Toc469516099)

[Instrukcja obsługi aplikacji 10](#_Toc469516100)

[Słownik 10](#_Toc469516101)

[Bibliografia 10](#_Toc469516102)

[O autorze 10](#_Toc469516103)

# O pracy inżynierskiej

Celem niniejszej pracy inżynierskiej było stworzenie aplikacji internetowej, której główną funkcjonalnością będzie możliwość rywalizacji użytkowników poprzez typowanie wyników wydarzeń sportowych.

Realizacja projektu była możliwa dzięki naukom i doświadczeniom, które autor wyniósł podczas trwania jego edukacji na kierunki Informatyka na Politechnice Krakowskiej. W czasie prac nad projektem pojawiały się jednak także inne, nieznane dotąd problemy, którym autor musiał sprostać jednocześnie zdobywając nowe umiejętności.

Aplikacja posiada przyjazny dla użytkownika interfejs graficzny dzięki czemu korzystanie z niej jest intuicyjne i zrozumiałe. Z aplikacji może korzystać wiele użytkowników równocześnie. Jest to aplikacja internetowa więc użytkownik posiadając jedynie przeglądarkę internetową oraz połączenie z internetem nie ma potrzeby instalowania instalowania żadnego dodatkowego oprogramowania, aby w pełni z niej korzystać.

Logika aplikacji jest zaimplementowana w języku programowania Java. Szkielet programu bazuje na wykorzystaniu trójwarstwowego wzorca projektowego Model-Widok-Kontroler (ang. *Model-View-Controller).* Został on zastosowany przy użyciu szkieletu wytwarzania aplikacji (ang. *Framework*) jakim jest Spring. Dane przechowywane są w bazie danych wykorzystując open-sourcowy system zarządzania relacyjnymi bazami danych PostgreSQL. Dostęp do danych zapewnia framework Hibernate. Widoki dostarczane są użytkownikowi przy wykorzystaniu silnika szablonów Thymeleaf. Wykorzystywane są również takie technologie jak Bootstrap, HTML, Javascript oraz CSS. Jako serwer umożliwiający uruchomienie aplikacji służył Apache Tomcat.

Tematyka aplikacji jest ściśle związana z piłką nożną oraz zakładami bukmacherskimi. O wyborze tematu aplikacji zdecydowały osobiste preferencje autora, którego sport jest największą, obok programowania, życiową pasją. Wybór ten jest także spowodowany niszą, która jest na rynku w tej dziedzinie, ponieważ poza profesjonalnymi firmami oferującymi obstawianie wydarzeń za pieniądze, brakuje aplikacji w której użytkownicy mogą rywalizować i sprawdzać się jedynie dla zabawy i własnej satysfakcji.

Nazwa aplikacji to „akoBet”. Jest to połączenie dwóch terminów: ako oraz Bet. Bet jest to angielskie wyrażenie oznaczające zakład lub postawienie na coś, jest to związane z główną funkcjonalnością programu jakim jest typowania wyników spotkań piłkarskich. Natomiast ako jest to termin funkcjonujący w zakładach bukmacherskich bardzo dobrze znany przez owe środowisko. Oznacza on skumulowany kurs wszystkich zdarzeń na kuponie gracza. Dodatkowo, ako jest to także pseudonim autora projektu pod którym jest on identyfikowany w społeczności internetowej od wielu lat. Dzięki temu aplikacja będzie w większy sposób utożsamiana z jej autorem.

# Wykorzystane technologie

## Model MVC

Wzorce projektowe to uniwersalne rozwiązania mające zastosowania do powtarzalnych problemów. Christoph Alexander stwierdził: „Każdy wzorzec opisuje problem powtarzający się w danym środowisku i istotę rozwiązania tego problemu w taki sposób, że można wykorzystać określone rozwiązanie milion razy i nigdy nie zrobić tego tak samo”. Alexander wypowiadał te słowa w kontekście architektury i budownictwa, jednak jego słowa idealnie odzwierciedlają także pojęcie wzorca w informatyce.

Stosowanie wzorców sprawia, że kod staje się znacznie czytelniejszy i bardziej przejrzysty. Sprzyja on także dalszym modyfikacjom i rozwijaniu go niekoniecznie przez tą samą osobę, ponieważ staje się on łatwiejszy do zrozumienia przez innych programistów. Wzorce są ściśle powiązane z programowaniem obiektowym.

Wzorce projektowe możemy podzielić na trzy kategorie:

* Wzorce konstrukcyjne – są to wzorce opisujące proces powstawania nowych obiektów, ich inicjalizację, konfigurację itp. Przykładami takich wzorców są „Budowniczy” (ang. *Builder*), „Singleton” (ang. *Singleton*), „Fabryka abstrakcyjny” (ang. *Abstract Factory*) czy też „Prototyp” (ang. *Prototype*).
* Wzorce strukturalne – opisują struktury powiązanych ze sobą obiektów. Wzorcami strukturalnymi są m.in. „Adapter” (ang. *Adapter*), „Fasada” (ang. *Fasada*), „Dekorator” (ang. *Decorator*), Pełnomocnik (ang. *Proxy*) oraz „Kompozyt” (ang. *Composite*)
* Wzorce operacyjne – opisują zachowanie obiektów. Ich przykłady to „Iterator” (ang. *Iterator*), „Obserwator” (ang. *Observer*), „Odwiedzający” (ang. *Visitor*) lub „Strategia” (ang. *Strategy*)

Jednym z najczęściej stosowanych wzorców w informatyce jest MVC. Głównym założeniem tego wzorca jest podzielenie kodu aplikacji na 3 moduły:

* Model
* Widok
* Kontroler

Najważniejszą zaletą wzorca MVC jest hermetyzacja modelu. Dzięki jego zastosowaniu możliwy jest jasny podział prac w zespołach projektowych. Osoba, która jest odpowiedzialna za tworzenie widoków nie musi praktycznie nic wiedzieć na temat działania logiki programu, ponieważ do widoku przekazywane są jedynie dane które należy odpowiednio obsłużyć. Natomiast osoba programująca kontroler nie musi wiedzieć jak wygląda kod modelu jeżeli wie co robią udostępnione mu metody, nie musi także wiedzieć jak będą wyglądały dane wyświetlane przez widok.

## Model

Model reprezentuje dane. Odpowiada on za pobieranie danych z określonego źródła. Najczęściej jest to baza danych, choć mogą być to także pliki, w których przechowywane są dane. Zazwyczaj każda część aplikacji posiada osobny model. W omawianym projekcie przykładami modelu są: użytkownicy, rozgrywki, artykuły.

### PostgreSQL

PostgreSQL jest obiektowo-relacyjnym systemem zarządzania bazami danych. Do najważniejszych jego zalet należą wysoka stabilność i wydajność. Względem MySQL, swojego największego konkurenta, PostgreSQL jest bardziej rozbudowany i obsługuje więcej dodatkowych funkcji języka SQL., posiada olbrzymie możliwości konfiguracji i rozszerzeń bazy, jak np. 11 serwerowych języków proceduralnych. PostgreSQL jest projektem o otwartych źródłach (ang. *Open Source*).

### Hibernate

Hibernate jest to technologia napisana w języku Java umożliwiająca mapowanie obiektowo relacyjne i zapewniająca łatwy dostęp do danych znajdujących się w bazie danych. Umożliwia ona tworzenie aplikacji bazodanowych w języku Java, w których programiści nie wykonują operacji bezpośrednio na bazie danych w języku SQL, lecz jedynie operują na klasach i obiektach języka Java, które są automatycznie tłumaczone na operacje bazodanowe na podstawie zdefiniowanych zależności. Stosowanie Hibernate poprawia wydajność i stabilność aplikacji. Jest to oprogramowanie na licencji Open Source.

## Widok

Widok służy prezentacji danych użytkownikowi końcowemu. Jest on odpowiedzialny za prezentację wyników działania logiki biznesowej. Dzięki oddzieleniu od logiki można bez konfliktów zmieniać wygląd strony w dowolnym momencie.

W aplikacji „akoBet” za widoki odpowiada silnik szablonów Thymeleaf rozszerzony o klasy CSS dostarczone poprzez blibliotekę Bootstrap. Wykresy tworzone są za pomocą biblioteki D3.js.

### HTML, CSS, Javascript

HTML, CSS oraz JavaScript są to podstawowe języki służące do tworzenia stron internetowych.

HTML (ang. *HyperText Markup Language*) jest to język opisu strony. Dokument HTML jest plikiem tekstowym, który oprócz tekstu zawiera także specjalne znaczniki interpretowane przez przeglądarke. Mimo standaryzacji niektóre przeglądarki inaczej interpretują te same polecenia.

CSS (ang. *Cascading Style Sheet*) jest to język służący do formatowania wyglądu elementów HTML. Kaskadowe arkusze stylów odpowiadają za rozmieszczenie na stronie poszczególnych elementów oraz formatowania i stylizowania wszelkich elementów na stronie. CSS ustawia parametry czcionki, kolorów, marginesów, linii, wysokości, szerokości, obrazków tła i wielu innych rzeczy.

JavaScript to język programowania, który odpowiedzialny jest za zwiększenie interaktywności strony internetowej poprzez reagowanie na działania użytkownika np. może odpowiadać za walidację formularzy. Pomimo tego, że nazwa języka sugeruje jego związek z Javą, nie mają one ze sobą zbyt wiele wspólnego.

jQuery jest to biblioteka języka JavaScript napisana w celu łatwiejszego posługiwania się tym językiem. Upraszcza ona wykorzystanie selektorów CSS oraz manipulację wszystkich elementów drzewa DOM.

### Thymeleaf

Thymeleaf jest silnikiem szablonów dokumentów XML, XHTML oraz HTML5. Biblioteka ta jest przystosowana do pełnienia warstwy widoku aplikacji, lecz można ją także wykorzystywać do generowania plików XML. Twórcy biblioteki zapewniają integrację ze Spring oraz Spring Security.

### Bootstrap

Bootstrap jest to framework służący do tworzenia responsywnych stron internetowych. W bootstrapie cały interfejs użytkownika jest reprezentowany jako siatka podzielona na dwanaście jednostek. Za pomocą odpowiednich klas CSS możemy określać ile wielkości jednostek potrzebuje dany element.

### D3.js

**D3.js** jest biblioteką przeznaczoną do tworzenia zaawansowanych wizualizacji danych w rozmaitych aplikacjach internetowych. Biblioteka ta umożliwia generowanie wykresów, ale pozwala również na tworzenie map, interaktywnych diagramów, paneli kontrolnych dla danych, raportów i wielu innych animowanych elementów.

## Kontroler

Kontroler obsługuje żadania użytkownika. Kontroler odpowiada m.in. za przetwarzanie danych pobranych za pomocą modelu i przekazanie ich użytkownikowi oraz zapisanie danych przez niego podanych. W kontrolerze odbywają się wszystkie konieczne obliczenia i podejmowane są odpowiednie akcje w zależności od działań użytkownika. Podsumowując, kontroler zajmuje się sterowaniem całą aplikacją i jest jej najważniejszym elementem.

### Java

Java jest to obiektowy język programowania wymyślony w 1991 r. Zaskakiwać może fakt, że oryginalnym celem powstania języka Java nie był internet lecz wymyślenie języka niezależnego od platform sprzętowej, który mógłby służyć do tworzenia oprogramowania dla różnych urządzeń domowego użytku.

Kluczem pozwalającym Javie rozwiązywać problemy bezpieczeństwa i przenośności jest fakt, że wyjściem generowanym przez kompilator języka Java nie jest kod wykonywalny, ale kod bajtowy.

Java to język który można scharakteryzować jedenastoma cechami, zdefiniowanymi przez autorów tego języka. Są to:

* Prosty - Składnia Javy rzeczywiście jest oczyszczoną wersją składni języka C++. Nie ma potrzeby dołączania plików nagłówkowych, posługiwania się arytmetyką wskaźnikową (a nawet składnią wskaźnikową), strukturami, uniami, przeciążaniem operatorów, wirtualnymi klasami bazowymi itd. Twórcy Javy o prostocie języka: „Naszym celem było zbudowanie takiego systemu, który można zaprogramować bez ukończenia tajemnych szkoleń, a który podtrzymywałby obecne standardowe praktyki. W związku z tym — mimo że w naszym przekonaniu język C++ nie nadawał się do tego celu — Java pod względem projektowym jest do niego podobna, jak to tylko możliwe. Dzięki temu nasz system jest bardziej zrozumiały. Java jest pozbawiona wielu rzadko używanych, słabo poznanych i wywołujących zamieszanie funkcji, które zgodnie z naszymi doświadczeniami przynoszą więcej złego niż dobrego.”
* Obiektowy – Programowanie obiektowe stanowi rdzeń języka Java. W zasadzie wszystkie programy napisane w tym języku są programami obiektowymi. Mechanizmami pomagającymi zaimplementować model obiektowy są: polimorfizm, dziedziczenie oraz hermetyzacja.
* Sieciowy - Java ma bogatą bibliotekę procedur wspomagających pracę z takimi protokołami TCP/IP jak HTTP i FTP. Funkcje sieciowe Javy są solidne oraz łatwe w użyciu.
* Niezawodny - Java została stworzona do pisania programów, które muszą być niezawodne w rozmaitych sytuacjach. Dużo uwagi poświęcono wczesnemu sprawdzaniu możliwości wystąpienia ewentualnych problemów, późniejszemu sprawdzaniu dynamicznemu (w trakcie działania programu) oraz wyeliminowaniu sytuacji, w których łatwo popełnić błąd. Kompilator Javy wykrywa wiele błędów, które w innych językach ujawniłyby się dopiero po uruchomieniu programu.
* Bezpieczny - Gdy korzysta się z przeglądarki WWW obsługującej Javę, można bezpiecznie pobierać aplety Javy i nie martwić się o możliwość zawirusowania komputera. System wykonawczy Javy otacza środowisko uruchomionego apletu od innych części komputera. Bezpieczeństwa zapewniane przez Javę jest jedną z jej najważniejszych zalet, gdyż niebezpieczne programy po prostu nie mają dostępu do istotnych elementów systemu komputerowego.
* Niezależny od architektury - Kompilator generuje kod bajtowy niemający nic wspólnego z żadnym konkretnym procesorem. W zamian kod ten jest tak konstruowany, aby był łatwy do interpretacji na każdym urządzeniu i aby można go było z łatwością przetłumaczyć na kod maszynowy w locie. Założenie projektantów Javy brzmi następująco: „Napisz raz, uruchom gdziekolwiek, kiedykolwiek, wiecznie.”
* Przenośny - W przeciwieństwie do języków C i C++ Java nie jest w żaden sposób uzależniona od implementacji. Rozmiary podstawowych typów danych są określone, podobnie jak wykonywane na nich działania arytmetyczne.
* Interpretowany - Interpreter Javy może wykonać każdy kod bajtowy Javy bezpośrednio na urządzeniu, na którym interpreter ten zainstalowano.
* Wysokowydajny - Mimo że wydajność interpretowanego kodu bajtowego jest zazwyczaj więcej niż wystarczająca, zdarzają się sytuacje, w których potrzebna jest większa wydajność. Kod bajtowy może być tłumaczony w trakcie działania programu na kod maszynowy przeznaczony dla określonego procesora, na którym działa aplikacja.
* Wielowątkowy - Java została zaprojektowana tak, by spełniać wymagania związanie z tworzeniem interaktywnych, sieciowych aplikacji. W tym celu w język zostały wbudowane mechanizmy programowania wielowątkowego, aby program mógł wykonywać wiele zadań jednocześnie.
* Dynamiczny - Java jest bardziej dynamicznym językiem niż C i C++ pod wieloma względami. Została zaprojektowana tak, aby dostosowywać się do ewoluującego środowiska. Do bibliotek można bez przeszkód dodawać nowe metody i zmienne egzemplarzy, nie wywierając żadnego wpływu na klienta.

### Spring

Spring jest to framework webowy języka Java dający programistom mnóstwo nowych możliwości. Architektura Spring to wielowarstwowy szkielet aplikacyjny dla aplikacji pisanych w Javie, Java EE oraz .NET. W skład szkieletu wchodzą:

* Kontener zarządzający komponentami JavaBean oraz klasami Java POJO.
* Warstwa zarządzania transakcjami zgodna ze standardami takimi jak JDBC, JDO, Hibernate.
* Gotowy szablon obsługujący interfejs JDBC.
* Moduły w pełni integrujące się z narzędziami zarządzania trwałością danych takimi jak Hibernate, JDO, IBatis SQL Maps.
* Kontener umożliwiający wykorzystanie programowania aspektowego.
* Kompletny szkielet aplikacyjny MVC
* Narzędzia do łatwej integracji z technologiami szkieletowymi warstwy prezentacji.

Podstawowym mechanizmem wykorzystywanym w architekturze Spring jest mechanizm wstrzykiwania zależności (ang. *dependency injection*), zwany także zasadą odwrócenia sterowania (ang. *inversion of control*). Drugą, obok wstrzykiwania zależności, podstawową techniką wykorzystywaną w architekturze Spring jest programowanie aspektowe. Ponadto Spring udostępnia wiele gotowych mechanizmów takich jak:

* Spring Security - gotowy mechanizm autentykacji i autoryzacji użytkowników
* Spring Data – mechanizm do łączenia się z bazami danych. Spring Data JPA łączy się z wieloma różnymi bazami danych. Może bez problemu działać na m.in. MySQL, Oracle oraz na bazach typu embedded jak h2 i wielu innych.

## Serwer aplikacji

Serwerem aplikacji stworzonej w ramach pracy inżynierskiej jest Apache Tomcat. Serwer Tomcat jest serwerem aplikacji na licencji Apache Software License pozwalającym uruchomić nam aplikacje webowe napisane w JAVA. Spełnia specyfikację firmy SUN jeśli chodzi o Java Servlets oraz Java Server Pages. Oraz jest w całości napisany w JAVA stąd jest dostępny na wszystkie platformy.

# Charakterystyka użytkowników

W dzisiejszych czasach sport jest jedną z najbardziej popularnych form zapewniania ludziom rozrywki, a piłka nożna jest bez wątpienia najbardziej popularnym sportem na świecie. Ludzie na całym świecie zbierają się, by śledzić rozgrywki piłkarskie niezależnie od wieku, koloru skóry, pochodzenia czy też stanu majątkowego. Futbol w obecnych czasach jest tematem codziennych rozmów ludzi z każdego kontynentu, zarówno tych najbiedniejszych jak i tych na najwyższych szczeblach. Największe piłkarskie wydarzenia są oglądane na żywo przez ponad 100 mln ludzi na całym świecie.

Z roku na rok rośnie również popularność zakładów bukmacherskich i ludzi mających ochotę spróbować swoich sił obstawiając wyniki wydarzeń sportowych. Nie zawsze jednak ludzie chcą rywalizować ryzykując utratę własnych pieniędzy, czasem wolą sprawdzić się dla własnej satysfakcji oraz spędzić wolny czas w ten sposób, ponieważ ludzie mają naturalną potrzebę rywalizowania z innymi.

Zestawiając powyższe dane z faktem, że w dzisiejszych czasach mało który człowiek na świecie nie ma dostępu do internetu, omawiana aplikacja ma szansę stać się niezwykle popularna.

Autor przewiduje, że największą popularnością aplikacja będzie cieszyła się wśród osób płci męskiej w przedziale wiekowym 12-25 lat. Jest to spowodowane tym, że mężczyzni są o wiele większymi fanatykami piłki nożnej, osoby płci żeńskiej z reguły szukają innych rozrywek w sieci. Docelowy wiek użytkowników jest spowodowany tym, że w młodym wieku osoby mają o wiele więcej czasu na wszelkiego typu rozrywki takie jak omawiana aplikacja. Autor przewiduje także popularność aplikacji wśród osób poniżej 18 roku życia z racji tego, że obstawianie wydarzeń za pieniądze jest legalne tylko dla pełnoletnich osób więc osoby niepełnoletnie będą bardziej skore do sprawdzenia się jedynie dla dobrej zabawy. Osoby starsze nieco bardziej sceptycznie podchodzą do takich form spędzania wolnego czasu. Nie oznacza to jednak że osoby o innej charakterystyce zupełnie nie będą zainteresowane aplikacją. Każda osoba mająca dostęp do internetu może bez przeszkód zacząć z niej korzystać.

# Wymagania

Wymaganie jest pojedynczą, udokumentowaną potrzebą określonego produktu lub usługi, albo sposobu ich działania. Cechy dobrze zdefiniowanych wymagań:

* Spójność – Jedno wymaganie powinno być opisane tylko w jednym miejscu aby zapobiec sprzecznościom.
* Kompletność - specyfikacja powinna wymieniać wszystkie wymagania funkcjonalne i niefunkcjonalne, które muszą być spełnione.
* Jednoznaczność – Wszystkie wymagania powinny być napisane tak, aby dało się je zinterpretować tylko w jeden możliwy sposób.
* Wykonalność – Wymaganie musi być możliwe do zrealizowania zgodnie z ograniczeniami.
* Obowiązkowość – Każde wymaganie musi być zrealizowane. W innym wypadku projekt będzie niekompletny.
* Weryfikowalność – Opis wymagania powinien być tak sformułowany, aby dało się jasno zweryfikować czy jest ono spełnione.

## Wymagania funkcjonalne

Wymagania funkcjonalne opisują funkcjonalność systemu, opisują co ma oferować system, jak ma reagować na określone dane wejściowe oraz w jaki sposób na zachowywać się w określonych sytuacjach.

Rejestracja

Logowanie

Poprawność danych

Przeglądanie aktualności

Komentowanie

Przeglądanie statystyk

Zapisanie się do rozgrywek

Uczestnictwo w rozgrywkach

Przeglądanie mapy z punktami bukmacherskimi

Dodatkowe wymagania funkcjonalne dla osób z uprawnieniami administratora:

Usuwanie użytkowników

Wybór meczów dla użytkowników.

## Wymagania niefunkcjonalne

Wymagania niefunkcjonalne opisują kryteria oceny systemu. Wymagania niefunkcjonalne nie mają bezpośredniego wpływu na funkcjonalność systemu. Nakładają one jednak ograniczenie na sposób, w jaki wymagania funkcjonalne będę realizowane.

Bezpieczeństwo

Aplikacja kliencka nie ma bezpośredniego dostępu do bazy danych, dzięki czemu dane w niej umieszczone są bezpieczne i nienarażone na przypadkowe zmiany.

Niezawodność

Aplikacja kliencka może być włączona bez ograniczeń czasowych.

Użyteczność

Aplikacja posiada prosty, przyjazny użytkownikowi interfejs. Dzięki temu korzystanie z niej jest intuicyjne i nie wymaga specjalnych umiejętności komputerowych. Jednak w razie problemów dołączona będzie instrukcja obsługi.

Dostępność

Aplikacja jest dostępna dla każdego użytkownika, który ma połączenie z internetem i zainstalowaną dowolną przeglądarkę internetową. Nie potrzebne jest instalowanie żadnego dodatkowego oprogramowania.

Odporność na błędy

Aplikacja jest odporna na błędy oraz na niestandardowe zachowania użytkownika. Po akcji programu zakończonej błędem zostaje wyświetlona specjalna strona z komunikatem informującym użytkownika co się stało.

Przenośność

Strona na telefonie blabla

Wiarygodność

Wrażliwe dane takie jak hasła użytkownika przechowywane w bazie danych są odpowiednio szyfrowane.

Rozszerzalność

Czas oczekiwania na potwierdzenie rejestracji

Email z potwierdzeniem rejestracji powinien być dostarczony w czasie nie dłuższym niż jedna godzina.

# Scenariusze testowe

# Diagramy UML

## Diagram przypadków użycia

## Diagram STD

## Diagram ETL

# Baza danych

## Opis bazy danych

## Diagram ERD

# Instrukcje

## Instrukcja uruchomienia

## Instrukcja obsługi aplikacji

# Słownik

Słownik użytych wyrażeń, które mogą być niejasne dla czytelnika.

Aplikacja kliencka – program pozwalający na wybór auta według określonych kryteriów.

Aplikacja serwera – program obsługujący żądania przychodzące od aplikacji klienckiej i odsyłający odpowiednie komunikaty.

Klient – patrz „Aplikacja kliencka”.

Serwer – patrz „Aplikacja serwera”.

Baza – baza danych, zawierająca dane użytkowników, klubów i rozgrywek.

Użytkownik – osoba korzystająca z aplikacji klienckiej.

Administrator - osoba korzystająca z aplikacji serwera.

„akoBet” – nazwa aplikacji

Framework – szkielet definiujący strukturę aplikacji oraz ogólny mechanizm jej działania.

Open source – otwarte oprogramowanie, z którego można korzystać za darmo.

# Bibliografia

„Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku”. – E.Gamma, R.Helm, R.Johnson, J.Vlissides

„Spring w akcji” – C.Walls

„JAVA. Podstawy. Wydanie IX” – Cay S. Hortsmann, Gary Cornell

„Java. Kompendium Programisty” – Herbert Schildt

# O autorze

Praca inżynierska została napisana przez Arkadiusza Junik. Studenta Politechniki Krakowskiej na kierunku Informatyka na Wydziale Inżynierii Elektrycznej i Komputerowej.

Opiekunem merytorycznym pracy był dr inż. Radosław Czarnecki.