POLITECHNIKA KRAKOWSKA

IM. TADEUSZA KOŚCIUSZKI

WYDZIAŁ FIZYKI MATEMATYKI I INFORMATYKI

KIERUNEK INFORMATYKA

JAROSŁAW ROLA

**SYSTEM DO ZARZĄDZANIA PLANEM TRENINGOWYM SPORTOWCA**

**A SYSTEM FOR MANAGING A TRAINING PLAN**

PRACA MAGISTERSKA

STUDIA STACJONARNE

Promotor: dr Agnieszka Krok

Kraków 2014

# Spis treści

[1](#_Toc397721655)

[1. Wstęp 1](#_Toc397721656)

[1.1 Rozwój i historia aplikacji mobilnych 2](#_Toc397721657)

[1.2 Rozwój i historia aplikacji internetowych 4](#_Toc397721658)

[1.3 Przegląd konkurencyjnych aplikacji 5](#_Toc397721659)

[1.4 Cele, założenia i zakres pracy 11](#_Toc397721660)

[2. Projekt aplikacji 13](#_Toc397721661)

[2.1 Wymagania funkcjonalne 13](#_Toc397721662)

[2.2 Wymagania niefunkcjonalne 16](#_Toc397721663)

[2.3 Architektura systemu 18](#_Toc397721664)

[2.4 Struktura bazy danych 19](#_Toc397721665)

[2.5 Diagramy języka UML 20](#_Toc397721666)

[2.5.1 Diagramy przypadków użycia 20](#_Toc397721667)

[2.5.2 Diagram encji 24](#_Toc397721668)

[2.6 Zastosowane technologie i narzędzia wspomagające 24](#_Toc397721669)

[3. Implementacja i wdrożenie 26](#_Toc397721670)

[3.1 Aplikacja mobilna 26](#_Toc397721671)

[3.2 Aplikacja internetowa 39](#_Toc397721672)

[3.3 Komunikacja 43](#_Toc397721673)

[3.4 Opis wdrożenia 45](#_Toc397721674)

[4. Testy aplikacji 46](#_Toc397721675)

[4.1 Testy jednostkowe 47](#_Toc397721676)

[4.2 Przypadki testowe 47](#_Toc397721677)

[4.3 Testy interfejsu 49](#_Toc397721678)

[5. Użytkowanie aplikacji 50](#_Toc397721679)

[5.1 Aplikacja mobilna 50](#_Toc397721680)

[5.2 Aplikacja internetowa 56](#_Toc397721681)

[6. Wnioski 60](#_Toc397721682)

[7. Bibliografia 61](#_Toc397721683)

[Dodatek A. Uruchomienie aplikacji 63](#_Toc397721684)

[Dodatek B. Zawartość płyty CD 64](#_Toc397721685)

# Wstęp

Zrealizowany projekt został wykonany na potrzeby pracy magisterskiej na kierunku Informatyka na Politechnice Krakowskiej. Tematem pracy jest zbudowanie systemu, który umożliwi sportowcom planowanie treningu. System będzie się składać  
z dwóch aplikacji: mobilnej, przeznaczonej na telefony komórkowe i inne urządzenia   
z systemem Android oraz aplikacji internetowej. Zebrane poprzez aplikacje mobilną dane będzie można umieścić w sieci, korzystając z aplikacji internetowej, tym samym prezentując je pozostałym użytkownikom.

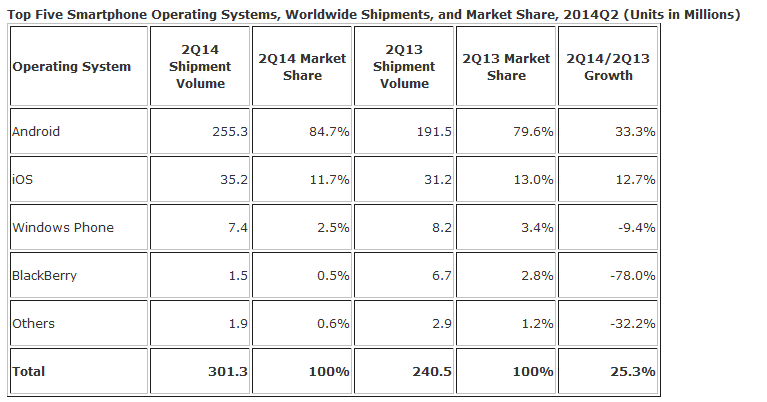
W obecnych czasach można dostrzec pewną modę na uprawianie sportu. Wynika to z faktu, że ludzie spędzający wiele godzin dziennie w szkole, bądź też w pracy, pracując umysłowo, odczuwają potrzebę relaksu poprzez aktywność fizyczną. To w połączeniu z ogólnorozumianym rozwojem techniki, wszechobecnymi w życiu każdego człowieka urządzeniami mobilnymi z dostępem do sieci i portalami społecznościowymi powoduje nieustanną popularyzację trendu aktywności fizycznej w Internecie.

## Rozwój i historia aplikacji mobilnych

Rozwój Internetu pociągnął za sobą powstanie urządzeń przenośnych, które umożliwiały dostęp do niego z dowolnego miejsca. Wraz ze wzrostem popularności telefonów komórkowych i innych urządzeń mobilnych rośnie także ilość dostępnych na nie aplikacji. Przez pojęcie aplikacji mobilnej (z ang. *mobile software, mobile application*) należy rozumieć oprogramowanie działające na urządzeniach przenośnych, takich jak telefony komórkowe, smartfony, palmtopy czy tablety, które budowane są w oparciu o różne platformy i języki programowania [1]. Logiczną konsekwencją wzrostu popularności urządzeń mobilnych jest wzrost zapotrzebowania na różnego typu aplikacje mobilne. Zgodnie z badaniem wykonanym przez TNS Polska w ciągu ostatnich dwóch lat liczba użytkowników smartfonów wzrosła o 1/3 [2]. Obecnie aż 44% Polaków posiada smartfona. Jeśli ten trend się utrzyma, to na początku 2015 Polska powinna osiągnąć poziom bliski 60%.

Swój dynamiczny rozwój aplikacje mobilne rozpoczęły pod koniec XX wieku [3]. Początkowo były to bardzo proste aplikacje, takie jak gry zręcznościowe, edytory tekstowe, kalkulatory itp. dostosowane do technicznych możliwości urządzeń przenośnych z tamtego okresu. Wraz ze wzrostem możliwości sprzętowych urządzeń, rosły także możliwości oferowane przez aplikacje. W 1983 roku pojawił się pierwszy dostępny w sprzedaży telefon. Była Motorola DynaTAC 800X o wymiarach 330 (z anteną)   
x 88,9 x 44,5 mm oraz wadze blisko 2 kg. Telefon ten umożliwiał wykonywanie połączeń oraz posiadał prostą aplikację do zarządzania kontaktami.

Wraz ze spadkiem cen, polepszeniem jakości akumulatorów i parametrów technicznych urządzeń, rynek poszerzał się i telefony komórkowe przestały być tylko nowinką technologiczną. Przełomem w rozwoju tych urządzeń przenośnych okazało się zapewnienie im dostępu do Internetu. Ówczesne telefony posiadały bardzo małe, monochromatyczne wyświetlacze oraz bardzo ograniczone pamięci. Również koszt połączenia z Internetem był wówczas bardzo wysoki. Rozwojowi aplikacji pomogło wprowadzenie standardu WAP (z ang. *Wireless Application Protocol*) [4]. Umożliwiał on dostęp do usług internetowych uwzględniając ograniczenia urządzeń mobilnych. Protokół ten nie spełniał jednak w pełni pokładanych w nim oczekiwań, głównie ze względu na szybkość działania. Korzystanie z niego powodowało spore koszty, a co za tym idzie ogólne niezadowolenie użytkowników.

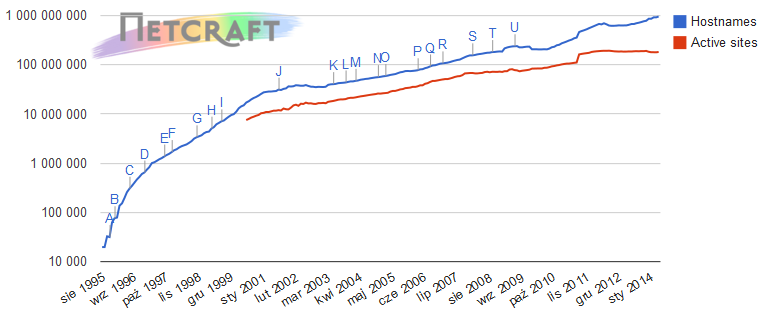
W pierwszej dekadzie XXI w. w urządzeniach mobilnych zaczęły pojawiać się kompaktowe wersje popularnych systemów operacyjnych, jak Linux czy Windows. Powodowało to zaangażowanie się w powstawanie nowego oprogramowania programistów znających te systemy. Oprogramowanie związane z urządzeniami mobilnymi stawało się więc coraz bardziej dostępne dla coraz szerszej grupy programistów. Rosła popularność Javy w wersji Micro Edition [5], ale także Symbian OS [6], który był

Rys. 1.1

Udział w rynku najpopularniejszych systemów na urządzenia mobilne w drugim kwartale 2014 roku [7].

wykorzystywany w urządzeniach takich firm jak Nokia, Sony Ericsson, Motorola, czy Samsung. W 2007 roku zaprezentowany został system iPhone iOS firmy Apple. Rok później Google przedstawiło pierwsze urządzenie z systemem Android. Systemy te są obecnie najpopularniejsze i aplikacje do nich napisane stanowią największą część rynku aplikacji mobilnych (Rys. 1.1) [7].

## Rozwój i historia aplikacji internetowych

Wraz z rozwojem Internetu nieustannie rośnie popularność aplikacji internetowych (potocznie zwanych aplikacjami *webowymi*) [8]. Aplikacje te wypierają systemy

Rys. 1.2

Wzrost liczby nazw hostingowych oraz aktywnych stron w latach 1995-2014 według ankiety przeprowadzonej przez agencję *Netcraft* [8]*.*

pisane w klasycznym modelu klient-serwer, który dominował w ostatnich dziesięcioleciach. Model ten polegał – w dużym uproszczeniu – na podziale zadań pomiędzy serwer, który zapewniał usługi dla klientów, którzy z kolei zgłaszali do niego żądania. Aplikacje napisane w tym modelu miały sporo wad. Do najważniejszych można zaliczyć z pewnością konieczność posiadania aplikacji klienckiej na każdej maszynie, która chciała się komunikować z serwerem. Również aktualizacje aplikacji klienckiej były problematyczne. Aby zachować zgodność z oprogramowaniem serwera, aktualizacja musiała zostać przeprowadzona we wszystkich aplikacjach klienckich. Powodowało to sporo trudności, zarówno dla programisty, gdy ten chciał zapewnić wsparcie dla aplikacji stworzonej w  modelu klient-serwer, jak i dla użytkownika końcowego aplikacji, który musiał te aktualizacje wykonywać.

Wymienione wyżej wady nie mają miejsca, gdy mówimy o aplikacjach internetowych. Aplikacje te wykorzystują dokumenty internetowe do przesyłania informacji. Dokumenty te są odczytywane przez przeglądarki internetowe, które obecnie można spotkać w każdym systemie operacyjnym. Aby dokonać aktualizacji oprogramowania wystarczy to zrobić jedynie na serwerze. Aplikacje internetowe można w pewnym sensie sklasyfikować jako pewien rodzaj architektury klient-serwer, gdzie przeglądarka internetowa pełni rolę cienkiego klienta (z ang. *thin client*) [9].

Dynamiczny rozwój aplikacji internetowych rozpoczął się w 1995 roku. Wtedy to firma *Netscape* wprowadziła język *JavaScript* [10]. Język ten zapewniał większą interaktywność aplikacjom na działania użytkownika. Przed tym wydarzeniem strony internetowe dostarczały jedynie statyczną, czyli taką samą dla każdego klienta, treść. Kolejnym ważnym wydarzeniem było wprowadzenie technologii *Flash* [11] przez firmę *Macromedia* (1996 r., obecnie zarządzana przez firmę *Adobe*). Umożliwiała ona dodawanie multimedialnych treści na strony www. Istotne z punktu widzenia rozwoju aplikacji internetowych okazały się także: wprowadzenie koncepcji *servletów* [12]do języka Java, wprowadzenie technologii *AJAX* [13], czy też przedstawienie standardu *HTML5* [14].

Aplikacje webowe to obecnie jedna z najszybciej rozwijających się dziedzin tworzenia oprogramowania. Badania przeprowadzone przez agencję   
*Netcraft* [8]w kwietniu 2014 roku pokazują, że w Internecie było wtedy 958 919 789 nazw hostingowych (Rys. 1.2). Rosnący trend z roku na rok się utrzymuje i nic nie wskazuje na to, by w najbliższych latach miało się to zmienić.

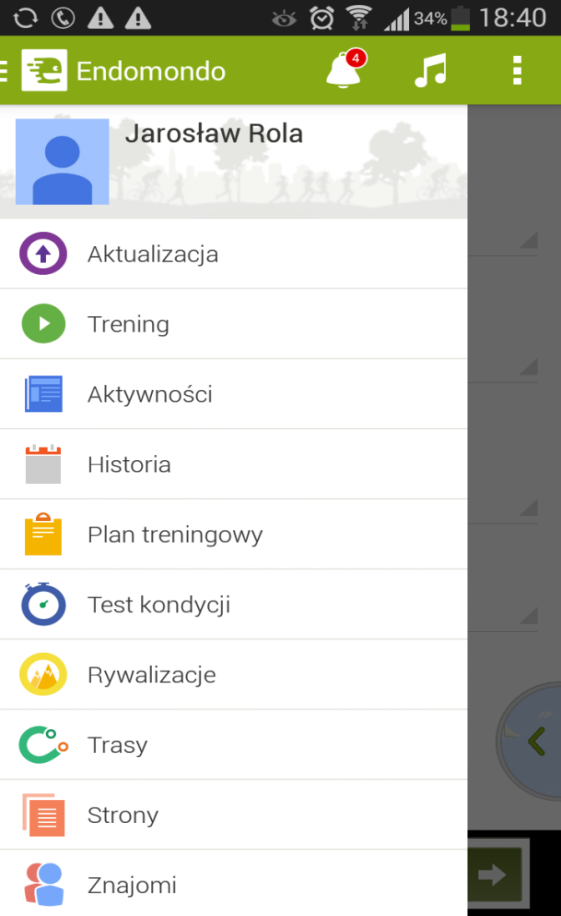
## Przegląd konkurencyjnych aplikacji

Na rynku funkcjonuje wiele serwisów internetowych i aplikacji mobilnych związanych z planowaniem treningu sportowców. Poniżej przedstawiona zostanie charakterystyka najbardziej popularnych i dostępnych za darmo aplikacji.

#### Endomondo

Platforma *Endomondo* [15] składa się z dwóch części: aplikacji mobilnej *Endomondo Sports Tracker*, która umożliwia zbieranie danych z przeprowadzonych treningów oraz aplikacji internetowej, w której użytkownicy mogą umieszczać dane zebrane przez aplikacje mobilną. Nie jest to typowa aplikacja do planowania treningu. Jej głównym przeznaczeniem jest rejestracja aktywności fizycznej w czasie rzeczywistym. Aplikacja daje też możliwość dodania treningu wcześniej odbytego. Pierwsza wersja na urządzenia przenośne została wydana w listopadzie 2007 roku i od tego czasu wersja tej aplikacji na system Android została pobrana z *Google Play* (oficjalnego internetowego sklepu z aplikacjami dla systemu Android) ponad 10 milionów razy. Aplikacja mobilna posiada dwie wersje: bezpłatną i płatną. Najważniejsze funkcje bezpłatnej wersji to:

* rejestracja prędkości, spalonych kalorii i czasu podczas uprawiania dowolnego sportu na zewnątrz wykorzystując moduł GPS,
* możliwość wprowadzenia treningu ręcznie, po jego odbyciu, nawet gdy brak zasięgu dla GPS,
* wskazówki głosowe podczas wykonywania ćwiczeń,
* umożliwienie wysyłania wiadomości pomiędzy użytkownikami,

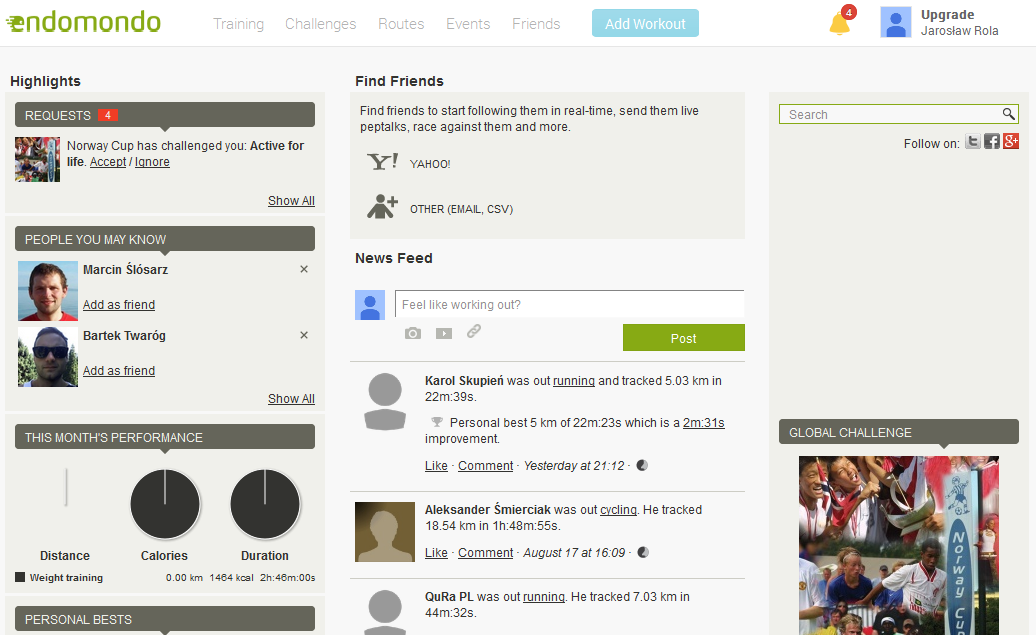
Rys. 1.3

Aplikacja mobilna *Endomondo Sports Tracker*.

* możliwość obejrzenia trasy swojego treningu na mapie,
* rejestracja i kontrola tętna (możliwość wykorzystania pulsometrów),
* możliwość utrzymywania historii treningów i analizowania uzyskanych wyników,
* możliwość definiowania celów i korzystanie z informacji głosowych aby je zrealizować,
* synchronizacja treningów na różnych platformach (np. wpis poprzez stronę internetową),
* możliwość publikowania uzyskanych rezultatów na portalach społecznościowych (np. *Facebook*, *Twitter*),
* możliwość śledzenia treningów znajomych,
* możliwość rywalizacji ze znajomymi, porównywanie osiągniętych rezultatów,
* definiowanie tras i rywalizacja z innymi użytkownikami o uzyskanie jak najlepszego wyniku na danej trasie,
* możliwość śledzenia bieżącej aktywności znajomych,
* historia słuchanych utworów podczas treningów,
* możliwość korzystania z mapy w celu orientacji w terenie,
* możliwość zatrzymania treningu i późniejszego wznowienia go,
* połączenie z rowerowym czujnikiem prędkości oraz pulsometrem,
* aplikacja dostępna jest na urządzenia mobilne z systemami: Android, iOS, Windows Phone, Symbian.

Aplikacja mobilna posiada jeszcze wiele innych funkcji, które dostępne są w opcji płatnej. Idealnie sprawdza się jako rejestrator aktywności fizycznej, gdy użytkownik wykonuje trening związany z przebyciem dystansu. Nie sprawdzi się natomiast, gdy uprawiany sport jest bardziej statyczny (np. siłownia, fitness, piłka nożna, tenis).

Aplikacja internetowa platformy *Endomondo* służy do prezentacji zebranych danych. Jeśli użytkownik dokona rejestracji, wszystkie dane z aplikacji mobilnej będą automatycznie przesyłane na portal internetowy. Aplikacja ta daje większe możliwości analizowania i porównywania wyników, komunikacji ze znajomymi oraz nawet śledzenia ich treningów w czasie rzeczywistym.



Rys. 1.4

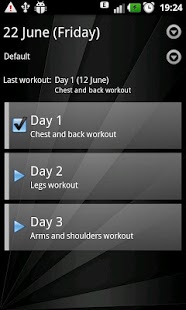
Aplikacja internetowa *Endomondo*.

#### Gym Book: training notebook

Aplikacja *Gym Book: training notebook* [16] jest przeznaczona dla użytkowników siłowni, którzy chcą monitorować postępy swojego treningu. Od momentu pojawienia się na rynku aplikacji dla systemu Android pobrało ją ponad 100 000 osób. Najważniejsze jej funkcjonalności to:

* możliwość monitorowania treningów następujących dyscyplin: lekkoatletyka, kulturystyka, trójbój siłowy,
* możliwość śledzenia osiągnięć,
* tworzenie programu szkoleniowego na każdy dzień treningowy,
* możliwość stworzenia cykli treningowych,
* podgląd statystyk po zakończonym treningu,
* przechowywanie statystyk i historii treningów,
* podgląd zmian w treningu siłowym w formie wykresu,
* import i eksport historii treningu,
* instrukcje jak wykonywać dane ćwiczenie.

Aplikacja jest łatwa w obsłudze i posiada intuicyjny interfejs użytkownika. Przeznaczona dla osób, które preferują tworzenie planów treningowych przed udaniem się na trening.

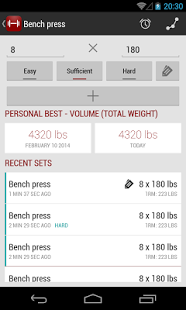
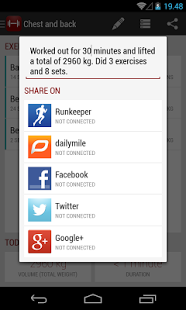
Rys. 1.5

Aplikacja mobilna *Gym Book: training notebook*.

#### Redy Gym Log - Workout Tracker

*Redy Gym Log - Workout Tracker* [17] jest aplikacją przeznaczoną dla osób preferujących trening siłowy, kulturystykę lub podnoszenie ciężarów. Do tej pory aplikacja została pobrana ponad 10 000 razy. Najważniejsze cechy darmowej wersji aplikacji to:

* możliwość zdefiniowania ulubionych ćwiczeń,
* możliwość zdefiniowania własnych planów treningowych,
* duży wybór ćwiczeń,
* możliwość publikowania treningów na portalach Runkeeper, Dailymile, Facebook, Twitter i Google+,
* dodawanie notatek, tworzenie dzienników ćwiczeń,
* zegar obliczający przerwy pomiędzy ćwiczeniami, okresy odpoczynku,
* szczegółowa historia poprzednich treningów,
* tworzenie kopii zapasowych i przywracanie danych dziennika za pośrednictwem poczty, lokalnego przechowywania lub usług w chmurze.

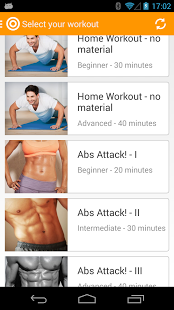
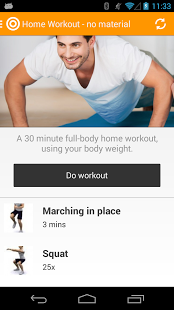
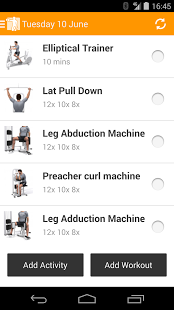
    
Rys. 1.6

Aplikacja mobilna *Redy Gym Log - Workout Tracker*.

#### Virtuagym Fitness Home & Gym

Aplikacja *Virtuagym Fitness Home & Gym* [18] przeznaczona jest dla osób, które podczas wykonywania ćwiczeń skupiają się przede wszystkim na poprawnej technice. Do tej pory została pobrana ponad 1 000 000 razy. Swoim użytkownikom daje dostęp do:

* planów treningowych przeznaczonych na siłownię oraz do ćwiczeń w domu,
* demonstracji poprawnego wykonania ćwiczenia poprzez trójwymiarowe animacje, które są przydatne szczególnie dla nowych użytkowników,
* szczegółowe instrukcje wykonywania ćwiczenia,
* moduł umożliwiający pomiar pracy serca,
* ponad 800 ćwiczeń,
* ponad 100 planów treningowych,
* synchronizacja aplikacji z portalem internetowym,
* darmowych planów żywieniowych,
* możliwość zdefiniowania celów treningu.

Rys. 1.7

Aplikacja mobilna *Virtuagym Fitness Home & Gym*.

## Cele, założenia i zakres pracy

Celem pracy jest projekt i implementacja systemu o nazwie *Vigym*, który składać się będzie z dwóch aplikacji: aplikacji mobilnej *vigyMobile* przeznaczonej na urządzenia przenośne z systemem Android oraz aplikacji internetowej *vigym.com*. Aplikacje działając razem będą tworzyć system, który będzie umożliwiał sportowcom zaplanowanie treningu. Przeznaczony będzie głównie dla sportowców trenujących na siłowni.   
W intuicyjny sposób zapewni funkcje takie jak planowanie treningu, pamiętanie historii ćwiczeń i monitorowanie postępów. System będzie umożliwiał wzajemną komunikację obu aplikacji. Przepływ danych będzie następował od aplikacji mobilnej, na której dane dotyczące treningów użytkowników będą zbierane, do aplikacji internetowej, która będzie te dane prezentować do ogółu użytkowników Internetu. System będzie łączyć cechy konkurencyjnych aplikacji związanych z planowaniem treningu. Będzie umożliwiał nie tylko rejestrowanie treningów w czasie rzeczywistym, ale również monitorowanie postępów i porównanie ich z innymi użytkownikami aplikacji webowej.

Rozdział pierwszy zawiera wstęp do pracy, określono w nim główne cele, założenia i zakres pracy. W tym rozdziale przedstawiona została historia aplikacji internetowych i mobilnych i ich rozwój na przestrzeni ostatnich dziesięcioleci. Dokonany został także przegląd konkurencyjnych aplikacji o podobnej tematyce, oferujące porównywalną funkcjonalność. W drugim rozdziale omówiono projekt aplikacji w oparciu   
o elementy inżynierii oprogramowania wraz z zastosowaniem odpowiednich diagramów UML. Ponadto scharakteryzowano użyte technologie oraz narzędzia wspomagające. W trzecim rozdziale dokonano omówienia szczegółów implementacyjnych poszczególnych elementów systemu oraz w jaki sposób w implementacji pomocne okazały się wykorzystane narzędzia i technologie. Dodatkowo opisany został proces wdrożenia aplikacji do środowiska produkcyjnego. Rozdział czwarty opisuje sposób testowania   
i weryfikacji systemu. W piątym rozdziale szczegółowo omówione zostały funkcjonalności   
z punktu widzenia użytkownika końcowego systemu. Podsumowanie zawiera rozdział szósty. W Dodatku A opisano sposób uruchomienia aplikacji, a Dodatek B przedstawia zawartość płyty CD dołączonej do pracy.

# Projekt aplikacji

Celem procesu projektowania aplikacji informatycznych jest przygotowanie ta-kiego projektu aplikacji, który będzie użyteczny i wykonalny. Wynikiem projektu jest powstanie dokumentacji. Niniejszy rozdział przedstawia projekt systemu *Vigym*.

## Wymagania funkcjonalne

Wymagania funkcjonalne względem aplikacji mobilnej:

* będzie możliwość utworzenia konta podając:
  + nazwę użytkownika (wymagane),
  + hasło (opcjonalne),
  + adres e-mail (opcjonalne),
* będzie możliwe zalogowanie się do aplikacji za pomocą nazwy użytkownika i hasła,
* aplikacja umożliwi zmianę hasła podanego przy rejestracji,
* będzie możliwe wylogowanie użytkownika z aplikacji,
* gdy użytkownik zapomni hasła, będzie możliwe przypomnienie hasła korzystając z adresu e-mail, jeśli został on podany przy rejestracji,
* aplikacja będzie umożliwiała korzystania z niej wielu użytkownikom na jednym urządzeniu mobilnym,
* jednocześnie zalogowany może być tylko jeden użytkownik.
* będzie możliwa synchronizacja danych z aplikacją internetową,
* użytkownik będzie mógł definiować charakteryzujące go cechy (wzrost, waga, obwód w pasie itp.),
* użytkownik będzie mógł edytować zdefiniowane wcześniej charakteryzujące go cechy,
* użytkownik będzie mógł usuwać zdefiniowane wcześniej charakteryzujące go cechy,
* do zdefiniowanej wcześniej cechy użytkownik będzie mógł dodawać pomiary określając: wartość, datę oraz jednostkę miary,
* użytkownik będzie miał możliwość edytowania lub usunięcia dodanych wcześniej pomiarów,
* aplikacja będzie dawać możliwość podglądu wszystkich dokonanych pomiarów wraz z szczegółami pomiaru (wartość, data oraz jednostka miary),
* aplikacja będzie wyświetlać zmianę cechy, która będzie różnicą ostatniego   
  i pierwszego pomiaru,
* aplikacja będzie posiadać zdefiniowaną pulę ćwiczeń wraz z opisami,
* użytkownik będzie mógł dodać trening wybierając ćwiczenie z listy, podając datę treningu, ilość powtórzeń danego ćwiczenia oraz obciążenie w kilogramach,
* przy dodawaniu treningu i wyborze ćwiczenia użytkownik powinien widzieć opis ćwiczenia,
* użytkownik będzie mógł zobaczyć wszystkie treningi wykonane w ramach danego ćwiczenia,
* użytkownik będzie mógł zobaczyć statystyki w ramach ćwiczenia, które wykonane zostało przynajmniej jeden raz:
  + informacja na temat ostatniego wykonanego treningu,
  + informacja na temat maksymalnego obciążenia w pojedynczym powtórzeniu,
  + informacja na temat maksymalnego obciążenia w ramach jednej serii wykonywania ćwiczenia,
  + informacja na temat średniego obciążenia w pojedynczym powtórzeniu dla ćwiczenia,
  + informacja na temat średniego obciążenia w jednej serii dla ćwiczenia,
* aplikacja będzie umożliwiać użytkownikowi edycję treningu wykonanego wcześniej (możliwa będzie zmiana daty wykonania ćwiczenia, ilości powtórzeń oraz obciążenia),
* aplikacja będzie umożliwiać użytkownikowi usunięcie treningu wykonanego wcześniej.

Wymagania funkcjonalne względem aplikacji internetowej:

* będzie możliwe zalogowanie się do aplikacji za pomocą nazwy użytkownika i hasła,
* panel logowania będzie dostępny na każdej stronie aplikacji,
* panel logowania będzie zawierał także odnośniki do:
  + strony głównej systemu,
  + do listy wszystkich użytkowników systemu,
* po zalogowaniu się użytkownik będzie miał możliwość wylogowania się,
* przeglądanie wszystkich podstron aplikacji powinno być możliwe zarówno dla zalogowanych użytkowników jak i tych, którzy nie posiadają konta   
  w systemie,
* strona główna aplikacji będzie wyświetlać listę ostatnio wykonanych treningów przez użytkowników aplikacji podając następujące informacje:
  + osoba wykonująca trening wraz z odnośnikiem do jej profilu w systemie,
  + wykonywane ćwiczenie,
  + ilość wykonanych powtórzeń,
  + użyte obciążenie,
  + data wykonania treningu,
* lista wszystkich użytkowników będzie zawierać:
  + odnośniki do profilów użytkowników w systemie,
  + informacje o ostatnim wykonanym treningu (wykonywane ćwiczenie, data wykonania ćwiczenia, liczba powtórzeń oraz zastosowane obciążenie),
* strona z profilem użytkownika będzie zawierać następujące sekcje:
  + porównanie,
  + informacje o użytkowniku,
  + wykonane treningi,
* sekcja porównanie będzie dostępna dla zalogowanych użytkowników i wyświetli porównanie ćwiczeń, które zostały wykonane zarówno przez użytkownika zalogowanego jak i użytkownika, którego profil jest przeglądany,
* sekcja informacje o użytkowniku wyświetli szczegóły cech charakterystycznych użytkownika, które zostały przez niego zdefiniowane:
  + nazwę cechy,
  + opis cechy,
  + postęp, jako różnice ostatniego pomiaru cechy i pierwszego pomiaru wraz z datą ostatniego pomiaru,
  + listę dokonanych pomiarów,
  + wykres zmiany pomiarów w czasie,
* sekcja treningi powinna udostępniać następujące szczegółowe informacje na temat wykonanych przez użytkownika treningów, pogrupowane względem ćwiczeń:
  + maksymalne użyte obciążenie w pojedynczej próbie,
  + maksymalne użyte obciążenie w serii,
  + średnie użyte obciążenie dla danego ćwiczenia w pojedynczej próbie,
  + średnie użyte obciążenie dla danego ćwiczenie w serii,
  + data, ilość powtórzeń oraz użyte obciążenie w ostatnim treningu,
  + listę wszystkich wykonanych treningów w ramach danego ćwiczenia.

## Wymagania niefunkcjonalne

Ogólne:

* system musi być w pełni skonfigurowany, czyli przygotowany do realizacji wszystkich wymagań funkcjonalnych,
* system musi spełniać wymogi bezpieczeństwa w zakresie dostępu użytkowników do zasobów systemu, poprzez zastosowanie mechanizmów uwierzytelnienia użytkownika,
* wykorzystane licencje oprogramowania muszą pozwalać na bezterminowe użytkowanie,
* system musi być dostępny przez 99,5% czasu,
* system musi być kompatybilny wewnętrznie (poszczególne elementy systemu, czyli aplikacja internetowa i mobilna muszą poprawnie współpracować w ramach systemu),
* system musi zapewniać bezpieczeństwo komunikacji pomiędzy aplikacjami wchodzącymi w jego skład,
* wszystkie komunikaty powinny być w języku polskim.

Interfejs aplikacji:

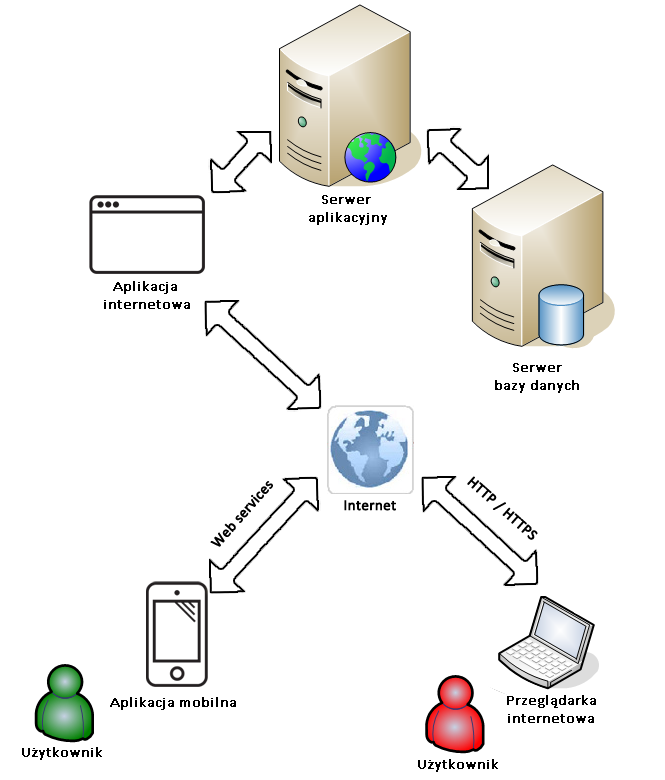
* powinien umożliwiać interakcję użytkownika z aplikacją oraz prezentować możliwości i funkcje jakie zostały zaimplementowane,
* powinien posiadać strukturę ułatwiającą korzystanie z aplikacji i możliwości, które oferuje,
* musi być przejrzysty, nie zawierać zbyt wielu grafik, umożliwiać bezproblemową nawigację,
* powinien wyświetlać podpowiedzi po umieszczeniu kursora na pewnych elementach interfejsu oraz gdzie to tylko możliwe zawierać krótkie informacje odnośnie funkcjonalności,
* musi pomagać użytkownikowi korzystać z funkcjonalności systemu w sposób intuicyjny, bez konieczności zapoznawania się z dokumentacją,
* tekst powinien być czytelny i zrozumiały, zarówno dla użytkowników korzystających z komputera stacjonarnego, laptopa, jak i dla użytkowników urządzeń mobilnych.

Techniczne:

* aplikacja internetowa, która jest integralnym składnikiem systemu musi mieć budowę trójwarstwową (serwer bazy danych, serwer aplikacyjny, aplikacja użytkownika),
* system musi pracować pod kontrolą przynajmniej dwóch różnych systemów operacyjnych, w tym przynajmniej jednego udostępnianego na zasadzie wolnego oprogramowania,
* system musi działać w technologii trójwarstwowej tzn. wykorzystywać w zakresie wszystkich oferowanych funkcji pracę poprzez graficzne przeglądarki internetowe pochodzące od minimum trzech różnych producentów, które są dostępne bezpłatnie,
* interfejs aplikacji internetowej musi być zgodny ze standardem *XHTML 1.0 Transitional* [19],
* system musi zapewnić skalowalność wydajnościową oraz funkcjonalną,
* system musi umożliwiać modyfikację sposobu działania systemu przez uprawnionych użytkowników bez konieczności ingerencji osób zewnętrznych,
* system musi współpracować z bazami danych dostępnymi bezpłatnie,
* system w zakresie czasów odpowiedzi musi spełniać kryteria:
  + średni czas odpowiedzi systemu przy transakcjach nie wprowadzających danych nie może przekraczać 5 sekund, a maksymalny 10 sekund,
  + średni czas odpowiedzi systemu przy transakcjach wprowadzających dane do systemu nie może przekraczać 5 sekund, a czas maksymalny nie może przekraczać 20 sekund,
* aplikacja internetowa wchodząca w skład systemu musi poprawnie działać dla 100 000 użytkowników w tym dla 1 000 zalogowanych jednocześnie,
* aplikacja mobilna wchodząca w skład systemu musi poprawnie działać dla 100 użytkowników,
* sesja będzie utrzymywana do momentu, gdy użytkownik wyloguje się.

## Architektura systemu

Projekt architektury systemu:



## Struktura bazy danych

Jako, że system *Vigym* składać się będzie z dwóch niezależnych aplikacji, konieczne jest posiadanie dwóch odrębnych baz danych. Jedna znajdować się będzie na maszynie, na której znajduje się serwer aplikacyjny, natomiast druga na urządzeniu mobilnym. Struktura obu baz będzie prawie identyczna. Jedyna różnica jest taka, że prawie każda tabela bazy danych aplikacji internetowej będzie posiadać dodatkową kolumnę, która będzie używana podczas synchronizacji danych z aplikacją mobilną. Baza danych systemu *Vigym* składa się z następujących tabel:

* USERS – tabela zawierająca dane zarejestrowanych użytkowników:
  + USER\_ID – identyfikator użytkownika,
  + USER\_NICKNAME – nazwa użytkownika,
  + USER\_MAIL – adres e-mail,
  + USER\_PASSWORD – hasło użytkownika zaszyfrowane za pomocą SHA1,
  + USER\_MOBILE\_ID – identyfikator używany przy synchronizacji (obecny tylko w bazie danych aplikacji internetowej),
* PROFILE\_INFORMATION – tabela zawierająca cechy charakterystyczne użytkownika:
  + PROF\_INFO\_ID – identyfikator cechy,
  + PROF\_INFO\_NAME – nazwa cechy,
  + PROF\_INFO\_DESC – opis cechy,
  + PROF\_INFO\_USER\_ID – identyfikator użytkownika powiązanego z cechą,
  + PROF\_INFO\_MOBILE\_ID – identyfikator używany przy synchronizacji (obecny tylko w bazie danych aplikacji internetowej),
* PROFILE\_INFORMATION\_VALUES – tabela zawierająca pomiary cech charakterystycznych:
  + PROF\_INFO\_VAL\_ID – identyfikator pomiaru,
  + PROF\_INFO\_VAL\_DATE – data wykonania pomiaru,
  + PROF\_INFO\_VAL\_VALUE – wartość pomiaru,
  + PROF\_INFO\_VAL\_UNIT\_ID – identyfikator jednostki pomiaru,
  + PROF\_INFO\_VAL\_INFO\_ID – identyfikator powiązanej cechy,
  + PROF\_INFO\_VAL\_MOBILE\_ID – identyfikator używany przy synchronizacji (obecny tylko w bazie danych aplikacji internetowej),
* UNITS – tabela zawierająca listę miar:
  + UNIT\_ID – identyfikator jednostki,
  + UNIT\_NAME – nazwa jednostki miary,
  + UNIT\_DESC – opis jednostki miary,
  + UNIT\_SHORT\_NAME – skrócona nazwa jednostki miary,
* TRAININGS – tabela zawierająca dane odbytych treningów:
  + TRAINING\_ID – identyfikator treningu,
  + TRAINING\_DATE – data wykonania treningu,
  + TRAINING\_VALUE – użyte obciążenie,
  + TRAINING\_COUNT – ilość wykonanych powtórzeń,
  + TRAINING\_EXERCISE\_ID – identyfikator wykonywanego ćwiczenia,
  + TRAINING\_USER\_ID – identyfikator użytkownika wykonującego ćwiczenie,
  + TRAINING\_MOBILE\_ID - identyfikator używany przy synchronizacji (obecny tylko w bazie danych aplikacji internetowej),
* EXERCISES – tabela zawierająca listę ćwiczeń:
  + EXERCISE\_ID – identyfikator ćwiczenia,
  + EXERCISE\_NAME – nazwa ćwiczenia,
  + EXERCISE\_DESC – opis ćwiczenia.

Diagram encji przedstawiający zależności pomiędzy tabelami przedstawiony został w rozdziale 2.5.2.

## Diagramy języka UML

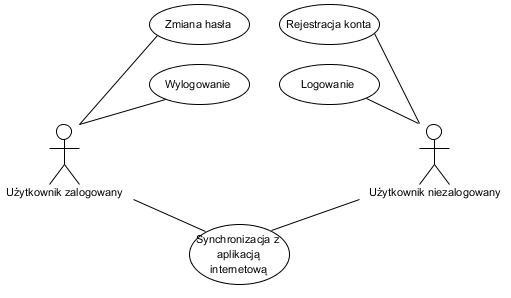
W tym rozdziale przedstawiono diagramy języka UML [20] związane z przed-stawionym systemem.

### Diagramy przypadków użycia

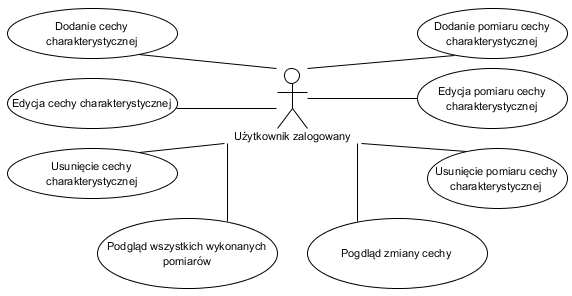
Diagramy przypadków użycia stosuje się w celu opisania wymagań tworzonego systemu informatycznego. Przypadki użycia określają wymagania funkcjonalne systemu.

Aplikacja mobilna:

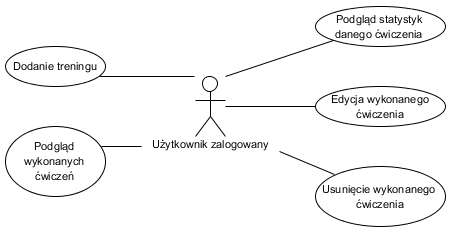
* funkcjonalność ogólna:



* funkcjonalność modułu *Cechy profilu*:

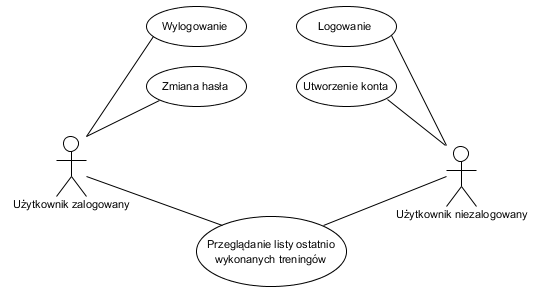


* funkcjonalność modułu *Trening*:

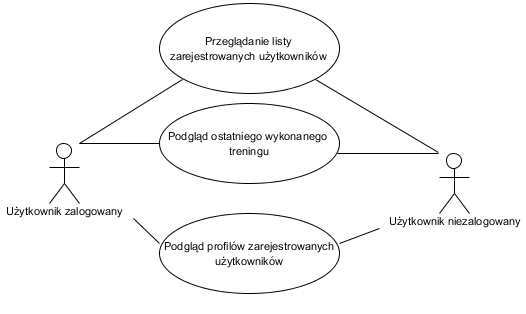


Aplikacja internetowa:

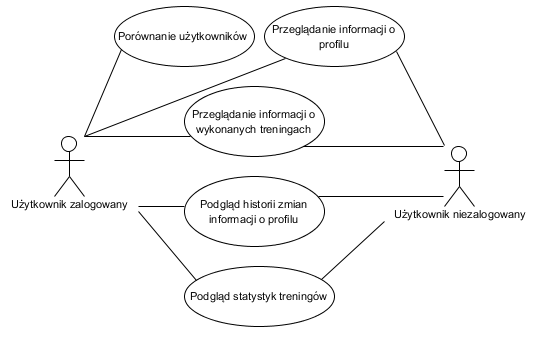
* funkcjonalność ogólna:



* funkcjonalność modułu *Lista użytkowników*:



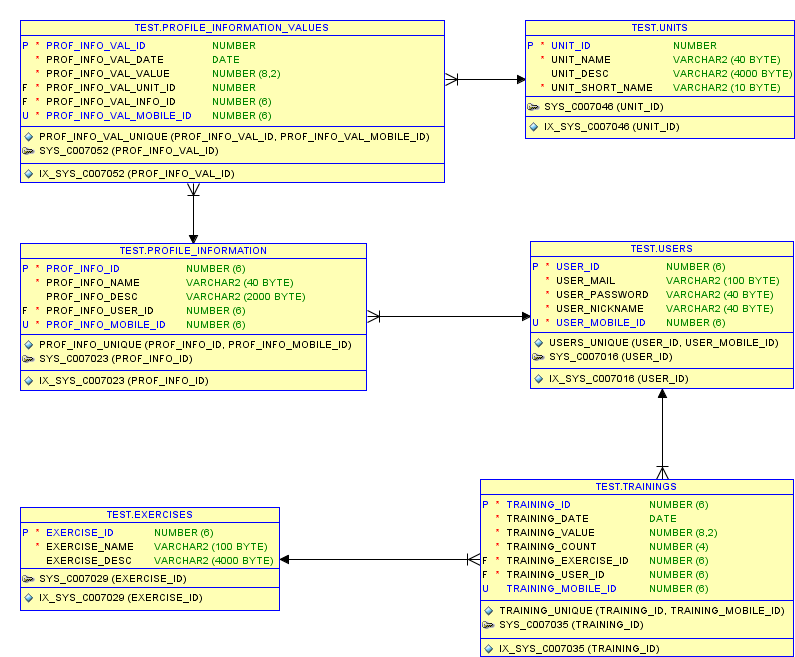
* funkcjonalność modułu *Profil*:



### 

### Diagram encji

Diagram encji przedstawiający powiązania pomiędzy tabelami znajdującymi się w bazie danych:



## Zastosowane technologie i narzędzia wspomagające

Jedną z najważniejszych, ale i najbardziej problematycznych decyzji na etapie projektowania systemu informatycznego, jest wybór technologii, które zostaną wykorzystane. Błąd na tym etapie może znacząco odbić się na dalszych fazach budowy oprogramowania. Do realizacji niniejszej pracy wykorzystane zostały następujące technologie i narzędzia:

* aplikacja mobilna:
  + system: *Android 4.4* [21],
  + baza danych: *SQLite 3.7.11* [22],
  + narzędzia wspomagające:
    - środowisko programistyczne - *Android Developer Tools v.22.3.0-887826* [23],
    - edytor dla bazy SQLite: *SQLite Database Browser 3.1.0* [24],
* aplikacja internetowa:
  + język programowania: *Java 1.7* [25],
  + technologie wspomagające:
    - *JavaServer Faces 2.2.7* [26]– biblioteka do budowania interfejsu użytkownika,
    - *Hibernate 4.3.6* [27] – biblioteka do mapowania obiektowo-relacyjnego,
    - *JUnit 4.11* [28] – biblioteka do testowania jednostkowego,
    - *Primefaces 5.0* [29]– biblioteka komponentów dla *JavaServer Faces*,
    - *Log4j 1.2.17* [30] – biblioteka do logowania,
  + baza danych: *Oracle Express Edition 11g* [31],
  + serwer aplikacyjny: *WebLogic 12c* [32],
  + narzędzia wspomagające:
    - środowisko programistyczne - *Eclipse Luna 4.4 EE for Web Developers* [33],
    - środowisko programistyczne dla baz Oracle - *SQL Developer 3.0.04* [34],
    - narzędzie automatyzujące budowę oprogramowania: *Apache* *Maven 3.2.2* [35],
* system zarządzania wersją - *Git 1.9.0* [36],
* narzędzie do projektowania diagramów - *Visual Paradigm for UML CE 10.2* [37].

Technologie które miały zostać wykorzystane do budowy systemu *Vigym* próbowano dobrać tak, aby zachować jednolitość języka programowania w obrębie obu aplikacji wchodzących w jego skład. Dodatkowo wzięto pod uwagę cechy bibliotek dostępnych dla języka, która umożliwią w łatwy sposób realizację postawionych wymagań projektowych. Te ograniczenia spowodowały, że wybór języka programowania musiał być dokonany pomiędzy platformami: *Java* oraz *.Net* [38]. Wybrana została Java, ze względu na jej lepszą znajomość, większą popularność oraz przenośność.

# Implementacja i wdrożenie

Rozdział ten opisuje proces implementacji systemu *Vigym*. Opisane zostały wykorzystane możliwości wybranych technologii.

## Aplikacja mobilna

#### Struktura projektu

Aplikacja mobilna składa się z dwóch projektów budowanych w środowisku *Eclipse*:

* *vigyMobile* – projekt aplikacji,
* *vigyMobileTests* – projekt zawierający testy jednostkowe, posiadający taką samą strukturę jak projekt aplikacji.

Projekt aplikacji i testów posiada następującą strukturę katalogową:

|  |  |
| --- | --- |
| Katalog | Zawartość |
| .settings | Katalog z ustawieniami środowiska programistycznego. Nie obecny w projekcie  z testami. |
| assets | Zasoby aplikacji. Pliki umieszczone tutaj zostaną umieszczone w pliku APK. |
| bin | Zbudowana aplikacja. Znajduje się tam plik APK do umieszczenia na urządzeniu mobilnym. |
| gen | Pliki generowane automatycznie przez środowisko programistyczne. |
| libs | Biblioteki zewnętrzne dołączone do projektu. |
| res | Zasoby aplikacji (obrazy, pliki XML). |
| src | Kod źródłowy. W przypadku aplikacji do testów jednostkowych kod zawiera testy. |
| .classpath | Plik konfiguracyjny ścieżki dla projektu. |
| .project | Plik konfiguracyjny projektu. |
| AndroidManifest.xml | Plik konfiguracyjny każdej aplikacji dla systemu Android |
| local.properties | Ustawienia projektu specyficzne dla maszyny, na której się znajduje. Nie obecny w projekcie z testami. |

Każdemu ekranowi aplikacji w systemie Android odpowiada klasa, która musi dziedziczyć z klasy *android.app.Activity* [21] – tzw. *Aktywność* (Aktywność, czyli klasa *Activity* zostanie opisana w późniejszym podrozdziale).Z tego faktu wynika podział struktury źródeł projektu aplikacji:

|  |  |
| --- | --- |
| Pakiet (zawarty w *pl.jrola.java.android.vigym.vigymobile.activities*) | Zawartość |
| ACTIVITY\_NAME | Klasy, które wchodzą w skład Aktywności*,* zazwyczaj elementy interfejsu użytkownika. *ACTIVITY\_NAME* to nazwa klasy, która dziedziczy po klasie  *android.app.Activity*. |
| ACTIVITY\_NAME.tasks | Klasy wątków. |
| ACTIVITY\_NAME.validators | Klasy sprawdzające zawartość danych w formularzu. |

W projekcie występują następujące klasy dziedziczące po klasie *Activity*: *LoginActivity, RegisterActivity, ImportDataActivity, MainActivity,* *ProfileManagementActivity*   
i *TrainingAcitivty*.

Pozostałe pakiety w źródłach:

|  |  |
| --- | --- |
| Pakiet | Zawartość |
| pl.jrola.java.android.vigym.vigymobile | Klasy ogólnego użytku. |
| pl.jrola.java.android.vigym.vigymobile.db.dao | Klasy dostępu do bazy danych. |
| pl.jrola.java.android.vigym.vigymobile.dao.to | Klasy zawierające informacje pochodzące z bazy danych. |
| pl.jrola.java.android.vigym.vigymobile.tasks | Klasy ogólne dla wątków. |
| pl.jrola.java.android.vigym.vigymobile.ui | Klasy ogólne dla interfejsu użytkownika. |
| pl.jrola.java.android.vigym.vigymobile.utils | Klasy zawierające funkcje, które są używane w całym projekcie. |
| pl.jrola.java.android.vigym.vigymobile.validator | Klasy ogólne sprawdzające poprawność formularzy. |

#### Komponenty interfejsu użytkownika

Każda klasa, która jest główną częścią interfejsu użytkownika (czyli taką, która zawiera inne elementy interfejsu) powinna implementować interfejs   
VigymUIComponent:

**public** **interface** VigymUIComponent {

**void** refresh();

**void** refresh(Collection<TransferObject> to);

LoadingSpinner getLoadingSpinner();

DbHelper getDatabaseHelper();

String getStringRes(**int** resId);

Context getContext();

}

Opis funkcji zawartych w interfejsie:

|  |  |
| --- | --- |
| Funkcja | Opis |
| **void** refresh(); | Odświeża komponent. |
| **void** refresh(Collection<TransferObject> to); | Odświeża komponent w oparciu o dane przekazane jako argument wywołania funkcji. |
| LoadingSpinner getLoadingSpinner(); | Zwraca pasek postępu wczytywania danych. |
| DbHelper getDatabaseHelper(); | Zwraca obiekt umożliwiający wykonywanie operacji bazodanowych. |
| String getStringRes(**int** resId); | Zwraca ciąg znaków powiązany z identyfikatorem przekazanym jako argument wywołania funkcji. |
| Context getContext(); | Zwraca obiekt aktualnego kontekstu aplikacji. |

#### Aktywność

Aktywność (z ang. *Activity*) jest to podstawowy komponent systemu Android. Klasa ta dziedziczy z klasy android.app.Activity i odpowiedzialna jest m.in. za interakcję z użytkownikiem i uruchamianie okna aplikacji. Zazwyczaj jest tak, że jedna Aktywność odpowiada jednemu oknu. Każda musi być także zadeklarowana w pliku AndroidManifest.xml. Wszystkie Aktywnościw systemie Android znajdują się na wspólnym stosie. Gdy następuje uruchomienie Aktywności, umieszczana jest ona na jego szczycie. W razie problemów z pamięcią, system podejmuje decyzje, które z nich usunąć na podstawie pozycji na stosie – te znajdujące się najniżej są usuwane jako pierwsze. Bardzo ważne jest zrozumienie cyklu życia obiektów tej klasy (Rys. 2.1).   
Z cyklu życia wynika jakie metody są uruchomiane podczas działania aplikacji i innych zdarzeń zewnętrznych, które to działanie przerywają (np. rozmowa przychodząca   
w trakcie korzystania z aplikacji). Najważniejsze metody znajdujące się w klasie Activity:

|  |  |
| --- | --- |
| Funkcja | Opis |
| onCreate | Metoda jest wywoływana, gdy Aktywność jest tworzona. Jest najlepszym miejscem do inicjalizacji komponentu. |
| onRestart | Funkcja jest uruchamiana, gdy Aktywność została wcześniej zatrzymana. Po tej funkcji zawsze jest uruchamiana funkcja onStart. |
| onStart | Metoda uruchamiana jest zanim Aktywność staje się widoczna dla użytkownika. |
| onResume | Funkcja uruchamiana jest zanim Aktywność będzie gotowa do interakcji  z użytkownikiem. |
| onPause | Metoda jest wywoływana, gdy system zaczyna rozpoczynać inną Aktywność. |
| onStop | Funkcja jest wywoływana, gdy Aktywność przestaje być widoczna dla użytkownika. |
| onDestroy | Metoda jest wywołana przed usunięciem Aktywności z pamięci. Jest najlepszym miejscem dla zwolnienia zasobów. |

W aplikacji mobilnej *vigyMobile* każda Aktywność powinna rozszerzać klasę VigymActivity. Skorzystanie z tej klasy:

* ułatwi dostęp do obiektu klasy DbHelper, który:
* umożliwia korzystanie z bazy danych,
* otwiera połączenie z bazą, gdy Aktywność jest tworzona – programista nie musi tego robić,
* zamyka połączenie z bazą danych, gdy Aktywność jest usuwana z pamięci,
* daje dostęp do komponentu klasy LoadingSpinner, który jest paskiem postępu. Możliwe jest dzięki temu włączanie i wyłączanie paska korzystając z tego obiektu w obrębie Aktywności.



Rys. 2.1

Cykl życia Aktywności.

Źródło: http://developer.android.com/reference/android/app/Activity.html.

#### Fragment

*Fragment* w systemie Android jest modularną częścią Aktywności, posiadającą swój własny cykl życia. Fragment jest zawsze zagnieżdżony w Aktywności i cykl jego życia, zależy od cyklu życia Aktywności. Cykl życia Fragmentu został przedstawiony na Rys. 2.2.



Rys. 2.2

Cykl życia Fragmentu.

Źródło: http://developer.android.com/guide/components/fragments.html.

W aplikacji *vigyMobile* każdy Fragment powinien rozszerzać klasę   
VigymFragment. Ułatwi to między innymi dostęp do połączenia bazodanowego.

#### Okna dialogowe

Okno dialogowe jest to małe okno, które jest wyświetlane, gdy potrzebne jest potwierdzenie użytkownika do wykonania operacji lub też uzupełnienie dodatkowej informacji. W aplikacji *vigyMobile* wszystkie okna dialogowe pochodzą od klasy bazowej

VigymDialog. Zapewni to możliwość korzystania z połączenia bazodanowego i paska ładowania w oknie dialogowym.

#### Layout

*Layout* w systemie Android definiuje rozmieszczenie komponentów graficznych, czyli ich układ. W systemie Android można zadeklarować układ komponentów na dwa sposoby:

* jako plik XML i przypisać go do Aktywności,
* zbudować układ tworząc obiekty elementów interfejsu użytkownika w trakcie działania aplikacji.

System Android zapewnia elastyczność w doborze metody budowania interfejsu użytkownika. Możliwe jest użycie obu metod jednocześnie. Zaletą deklarowania układu   
w pliku XML jest oddzielenie warstwy prezentacji aplikacji od kodu. Daje to możliwość zmieniania wyglądu bez konieczności dokonywania zmian w kodzie aplikacji. Gdy layout zostanie zbudowany z użyciem XML należy go przypisać do Aktywności   
w metodzie onCreate:

public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.main\_layout);  
}

Został ustawiony przykładowy layout zawierający pole tekstowe oraz przycisk:

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
              android:layout\_width="match\_parent"  
              android:layout\_height="match\_parent"  
              android:orientation="vertical" >  
    <TextView android:id="@+id/text"  
              android:layout\_width="wrap\_content"  
              android:layout\_height="wrap\_content"  
              android:text="Hello, I am a TextView" />  
    <Button android:id="@+id/button"  
            android:layout\_width="wrap\_content"  
            android:layout\_height="wrap\_content"  
            android:text="Hello, I am a Button" />  
</LinearLayout>

#### Formularze

Każdy komponent aplikacji *vigyMobile* wprowadzający dane do systemu, czyli zawierający formularz powinien implementować interfejs VigyForm:

**public** **interface** VigymForm {

**void** clearForm();

**void** clearFormErrors();

**boolean** validate();

TransferObject submitForm();

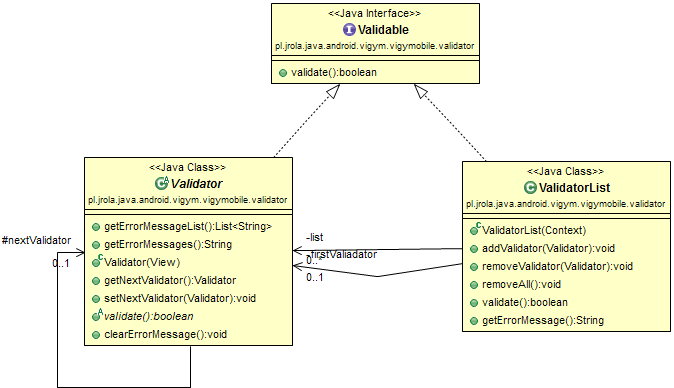
**void** initForm(TransferObject transferObject);

}

Opis funkcji zawartych w interfejsie:

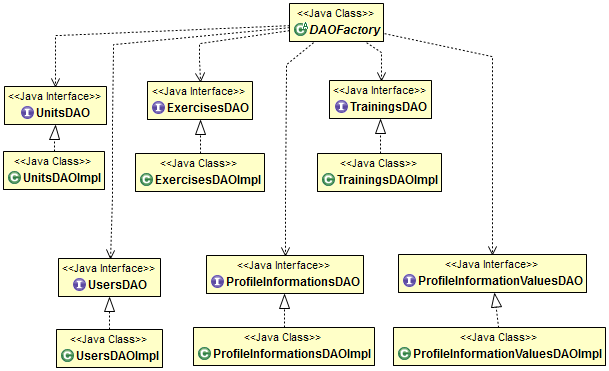
|  |  |
| --- | --- |
| Funkcja | Opis |
| **void** clearForm(); | Funkcja usuwa wszystkie wartości wpisane przez użytkownika z formularza. |
| **void** clearFormErrors(); | Funkcja usuwa wszystkie komunikaty o błędach walidacji pól formularza. |
| **boolean** validate(); | Funkcja sprawdzająca poprawność danych formularza. |
| TransferObject submitForm(); | Funkcja odczytująca dane z formularza i zwracająca je w formie obiektu. |
| **void** initForm(TransferObject transferObject); | Funkcja uzupełniająca formularz danymi z obiektu, który został przekazany jako argument wywołania. |

Sprawdzanie poprawności formularza odbywa się w oparciu o wzorzec projektowy Łańcuch zobowiązań [39, pp. 244-253], którego diagram klas został przedstawiony poniżej.



#### Dostęp do źródła danych

W aplikacji *vigyMobile* dla systemu Android jako źródło danych wykorzystana została baza danych. Dostęp do niej został zrealizowany w oparciu o wzorce projektowe: metoda wytwórcza [39, pp. 110-119], DAO (z ang. *Data Access* Object) [40] oraz obiekt transferowy (z ang. *Transfer Object*) [41]. Diagram klas przedstawiony został poniżej.

****

Zastosowanie wzorca DAO powoduje uniezależnienie warstwy danych od sposobu dostępu do nich. Zapewnia jednolity interfejs ukrywający szczegóły implementacyjne. Daje programiście elastyczność, gdy trzeba zmienić źródło danych (np. system bazodanowy) lub sposób dostępu do nich. Wzorzec Transfer Object użyty został do transportowania danych. Zastosowanie tego wzorca powoduje zmniejszenie liczby odwołań przez interfejsy, które często są zdalne, a co za tym idzie, zmniejszenie ruchu sieciowego. Przykładowo pobierając dane użytkowników z tabeli bazodanowej potrzebny będzie identyfikator użytkownika oraz jego login. Najkorzystniej będzie zaprojektować klasę obiektu transferowego posiadającej dwa atrybuty odpowiadające identyfikatorowi i hasłu. Przy żądaniu danych ze źródła danych tworzony jest obiekt transferowy i zostaje on wypełniony danymi. Możliwy jest teraz odczyt właściwości z instancji obiektu transferowego za pomocą metod dostępu bez pośrednictwa zdalnego interfejsu.

#### Wielowątkowość

Podczas wykonywania czasochłonnych operacji w systemie Android optymalnie jest wykorzystanie możliwości oferowanych przez ten system w zakresie wielowątkowości. Pozwoli to na wykonanie w obrębie jednego procesu kilku   
zadań. Twórcy systemu oddali do dyspozycji programistów klasę   
android.os.AsyncTask<Params, Progress, Result>, która ułatwi korzystanie   
z możliwości jakie daje wielowątkowość. Najważniejsze metody tej klasy:

|  |  |
| --- | --- |
| Funkcja | Opis |
| onPreExecute | Metoda uruchamiana przed wykonaniem wątku. |
| doInBackground | Główna metoda wykonania wątku.  W niej powinien zostać umieszczony kod, który zajmuje najwięcej czasu. |
| onProgressUpdate | Funkcja używana do przekazywania parametrów postępu do wątku interfejsu użytkownika. |
| onPostExecute | Metoda ta jest wywoływana po zakończeniu działania wątku. |

Obiekt stworzony na podstawie klasy AsyncTask może być wykonany tylko jeden raz.

W aplikacji *vigyMobile*, każda klasa wątku powinna rozszerzać klasę:   
VigymAsyncTask. Spowoduje to:

* automatyczne pokazanie się paska ładowania, gdy wątek zacznie się wykonywać,
* automatyczne ukrycie paska ładowania, gdy wątek zakończy swoje działanie,
* aktualizacja komponentu interfejsu użytkownika, z którego wątek został wykonany.

#### Internacjonalizacja

Do dobrych praktyk budowania aplikacji mobilnych w systemie Android należy umieszczanie ciągów znaków w osobnym pliku. Umożliwi to przetłumaczenie aplikacji na inne języki. Aby tego dokonać wystarczy w katalogu res projektu utworzyć strukturę katalogową jak w poniższym przykładzie:

MyProject/

res/

values/

strings.xml

values-es/

strings.xml

values-fr/

strings.xml.

W katalogu values powinny znajdować się pliki XML dla domyślnej wersji językowej, w values-es dla wersji hiszpańskiej, a w values-fr dla francuskiej. System Android automatycznie zlokalizuje z której wersji powinien skorzystać bazując na ustawieniach lokalizacji urządzenia. Przykładowa zawartość pliku dla wersji angielskiej:

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
    <string name="title">My Application</string>  
    <string name="hello\_world">Hello World!</string>  
</resources>

#### Kompatybilność z różnymi urządzeniami

System Android jest dostępny dla wielu urządzeń przenośnych różniącymi się znacznie parametrami technicznymi, takimi jak rozmiar wyświetlacza, szybkość procesora, rozmiar pamięci. Sporym wyzwaniem jest dla programistów aplikacji sprawienie, żeby aplikacja na każdym z urządzeń działała poprawnie. System Android posiada wiele ułatwień, które pomagają w realizacji tego celu:

* plik konfiguracyjny AndroidManifest.xml – plik ten zawiera informacje niezbędne dla systemu Android, dotyczące:
  + dla jakich wersji systemu Android aplikacja jest przeznaczona,
  + z jakich zasobów systemu aplikacja będzie korzystać i do jakich zasobów będzie potrzebować uprawnień,
* różne rozmiary obrazów – twórca aplikacji może umieścić obrazy, z których korzysta aplikacji w różnych rozmiarach, system automatycznie wybierze ten, który jest odpowiedni bazując na rozmiarze wyświetlacza,
* różne rozmiary layoutów – programista może zaprojektować kilka layoutów,   
  a następnie system, w zależności od urządzenia na którym działa wczyta odpowiedni,
* biblioteki zgodności – są to biblioteki, które zapewniają dostęp do nowych funkcji systemu w starszych urządzeniach.

## Aplikacja internetowa

#### Struktura projektu

Aplikacja internetowa została zbudowana w środowisku *Eclipse* z użyciem narzędzia *Apache Maven*. Projekt aplikacji posiada następującą strukturę katalogową:

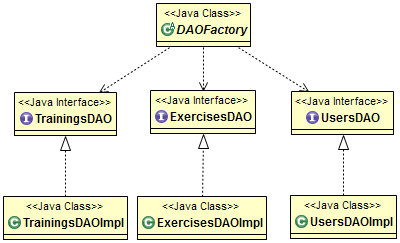
|  |  |
| --- | --- |
| Katalog | Zawartość |
| .git | Katalog z ustawieniami repozytorium. |
| .settings | Katalog z ustawieniami środowiska  programistycznego. |
| src/main/java | Kod źródłowy. |
| src/main/resources | Pliki konfiguracyjne: plik z konfiguracją biblioteki Hibernate oraz pliki językowe. |
| src/main/webapp | Dokumenty xHTML i inne. |
| src/test/java | Testy jednostkowe. |
| target | Katalog ze zbudowaną wersją aplikacji. |
| .classpath | Plik z konfiguracją ścieżki dla projektu. |
| .gitignore | Plik konfiguracyjny repozytorium. Zawiera informacje na temat plików, które nie mają się znaleźć w repozytorium. |
| pom.xml | Plik konfiguracyjny Apache Maven. |

Struktura kodu źródłowego:

|  |  |
| --- | --- |
| Pakiet | Zawartość |
| pl.jrola.java.www.vigym.model | Katalog z ustawieniami repozytorium. |
| pl.jrola.java.www.vigym.model.dao | Warstwa dostępu do danych. |
| pl.jrola.java.www.vigym.model.dao.exceptions | Wyjątki. |
| pl.jrola.java.www.vigym.model.entities | Encje, czyli klasy, na podstawie których są tworzone obiekty odpowiadające rekordom z tabeli bazodanowej. |
| pl.jrola.java.www.vigym.viewcontroller | Komponenty interfejsu użytkownika. |

#### Dostęp do bazy danych

W aplikacji internetowej *vigym.com* jako źródło danych wykorzystana została baza danych. Dostęp do niej został zrealizowany analogicznie jak w aplikacji mobilnej - w oparciu o wzorce projektowe: metoda wytwórcza [39, pp. 110-119], DAO (z ang. *Data Access* Object) [40] oraz obiekt transferowy (z ang. *Transfer Object*) [41]. Diagram klas przedstawiony został poniżej.

****

W warstwie dostępu do danych wykorzystana została biblioteka Hibernate. Zapewnia ona mapowanie danych zawartych w relacyjnej bazie danych na obiekty języka Java. Klasy, z których mają być tworzone obiekty zawierające dane pochodzące z tabel bazodanowych muszą posiadać specjalne adnotacje. Dodatkowo Hibernate zwiększa wydajność operacji na bazie danych dzięki optymalizacji zapytań.

#### Interfejs użytkownika

*JavaServer Faces* [26] jest biblioteką dostępną w języku Java, która upraszcza tworzenie interfejsu użytkownika. Jest częścią specyfikacji *Java Enterprise Edition* [42]. Jest oparta o komponenty, które działają po stronie serwera. Zakres funkcjonalny JSF obejmuje:

* budowę stanowego interfejsu użytkownika opartego na komponentach zarządzalnych,
* obsługę nawigacji pomiędzy stronami,
* walidację danych,
* internacjonalizację, czyli wsparcie dla aplikacji wielojęzycznych.

*Primefaces* [29] jest biblioteką, która zapewnia dodatkowy zestaw komponentów dla aplikacji internetowych pisanych z użyciem JavaServer Faces. Zapewnia bogaty zestaw komponentów interfejsu użytkownika, które wykorzystują technologię AJAX.

#### Zarządzalne komponenty

*Zarządzalne komponenty* lub *zarządzalne ziarna* (z ang. *managed beans)* są to klasy spełniające reguły *JavaBeans* [43, pp. 8-10], czyli:

* bezargumentowy konstruktor,
* brak publicznych pól,
* dostęp do pól przez metody get i set,

i zarejestrowane w pliku konfiguracyjnym lub poprzez adnotacje. Ziarna w aplikacji internetowej służą przede wszystkim do przechowywania danych dynamicznych. Przy rejestracji nadawana jest nazwa oraz zakres w jakim działać będą utworzone na podstawie zarejestrowanych klas obiekty. Po zarejestrowaniu programista może korzystać z obiektów na stronach JSF. Biblioteka ta oferuje następujące zakresy ważności komponentów:

|  |  |
| --- | --- |
| Zakres | Opis |
| ApplicationScope | Obiekt o takim czasie życia funkcjonuje przez cały cykl życia aplikacji. Jest tworzony w momencie uruchamiania aplikacji  i niszczony w momencie zatrzymywania jej działania. |
| SessionScope | Komponenty o takim czasie życia dostępne są tworzone w momencie utworzenia sesji i żyją tak długo, jak długo trwa sesja. |
| ViewScope | Zasięg życia obiektu jest ograniczony do jednego widoku. |
| RequestScope | Tak utworzony obiekt jest dostępny od momentu rozpoczęcia przetwarzania żądania wyświetlenia strony aż do momentu jej całkowitego utworzenia. Po wysłaniu strony do klienta obiekt jest niszczony. |
| NoneScope | Obiekt nie posiadający żadnego zasięgu. |

#### Internacjonalizacja

Aby zwiększyć potencjalne grono odbiorców aplikacji internetowej należy uczynić ją dostępną dla użytkowników, którzy porozumiewają się w różnych językach. Dobrą praktyką jest umieszczanie ciągów znaków w osobnym pliku. Umożliwi to łatwe przetłumaczenie aplikacji na inne języki. Aby tego dokonać wystarczy w katalogu src/main/res utworzyć następującą strukturę:

src/main/res

lang.properties

lang-en.properties

lang-es.properties

i zarejestrować ją w pliku konfiguracyjnym:

<application>

<locale-config>

<default-locale>pl</default-locale>

</locale-config>

<resource-bundle>

<base-name>lang</base-name>

<var>msg</var>

</resource-bundle>

</application>

Pliki zawierające ciągi znaków wykorzystywane w aplikacji mają następującą strukturę:

date=Data

desc=Opis

exercise=\u0106wiczenie

history=Historia

history\_chart=Wykres zmiany

Aplikacja korzystająca z bibliotek JSF automatycznie, na podstawie lokalizacji użytkownika wykryje, którą wersję językową wybrać.

## Komunikacja

Aplikacje działające w ramach systemu *Vigym* komunikują się ze sobą za pośrednictwem Internetu. Dane zebrane na aplikacji mobilnej są przesyłane na aplikację internetową na żądanie użytkownika. Aplikacja mobilna nie potrzebuje więc ciągłego połączenia z Internetem do poprawnego działania. Rozwiązanie takie powoduje, że użytkownik łączy się z siecią tylko wtedy, gdy chce zsynchronizować dane. Transfer nie jest zbędnie zużywany.

Algorytm synchronizacji danych jest następujący:

1. Użytkownik aplikacji mobilnej wykonuje synchronizację.
2. Jeżeli użytkownik wykonywał już wcześniej synchronizację, należy przejść do punktu 3. W przeciwnym przypadku - jeżeli jest to pierwsza synchronizacja:
   1. użytkownik proszony jest o podanie:
      1. nazwy użytkownika,
      2. adresu e-mail,
      3. hasła,

jeżeli dane te zostały podane przy rejestracji, nie ma potrzeby ich ponownego wprowadzania,

* 1. sprawdzana jest unikalność nazwy użytkownika i adresu e-mail:
     1. jeżeli dane są unikalne, na podany adres e-mail wysyłany jest klucz aktywacyjny,
     2. jeżeli dane nie są poprawne, użytkownik proszony jest o poprawne podanie danych,
  2. po otrzymaniu odnośnika z kluczem aktywacyjnym i uruchomieniu go   
     w przeglądarce konto jest w pełni aktywowane,
  3. następuje umieszczenie wszystkich danych z aplikacji mobilnej na koncie użytkownika w aplikacji webowej,

1. Jeżeli użytkownik wykonywał już wcześniej synchronizacje:
   1. system dodaje brakujące dane w bazie danych aplikacji internetowej,
   2. system edytuje dane zmienione w aplikacji internetowej, a które nie zmieniły się w
   3. system usuwa dane, które zostały usunięte z aplikacji mobilnej, a istnieją w aplikacji internetowej.

Wykorzystywana jest kombinacja kluczy głównych tabel. Poniżej został przedstawiony przykładowy proces synchronizacji dla tabeli fragmentu tabeli TRAININGS, które zawiera informacje na temat treningów odbytych przez użytkownika. Zawartość tabeli bazodanowej na urządzeniu mobilnym:

|  |  |  |  |
| --- | --- | --- | --- |
| TRAINING\_ID  (klucz główny) | TRAINING\_DATE | TRAINING\_USER\_ID  (klucz obcy z tabeli USERS) | TRAINING\_COUNT |
| 1 | 2014/08/02 | 1 | 10 |
| 2 | 2014/08/06 | 1 | 12 |
| 3 | 2014/08/09 | 1 | 2 |
| 7 | 2014/08/22 | 1 | 12 |

Zawartość tabeli TRAININGS w bazie danych aplikacji internetowej dla danego użytkownika:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TRAINING\_ID  (klucz główny) | TRAINING\_DATE | TRAINING\_USER\_ID  (klucz obcy z tabeli USERS) | TRAINING\_COUNT | TRAINING\_MOBILE\_ID |
| 12601 | 2014/08/02 | 551 | 10 | 1 |
| 40642 | 2014/08/06 | 551 | 11 | 2 |
| 60993 | 2014/08/09 | 511 | 2 | 3 |
| 71214 | 2014/08/20 | 511 | 4 | 4 |

Każda z tabel posiada odrębną numerację klucza głównego. Również identyfikator tego samego użytkownika w obu bazach danych jest różny. Znaczenia ma tutaj kolumna TRAINING\_MOBILE\_ID, która znajduje się w bazie danych aplikacji internetowej. Na jej podstawie system podejmuje decyzję dotyczące zmian w tabeli aplikacji internetowej:

* pozostawienie wierszy o identyfikatorach 12601 oraz 60993, gdyż nie uległy one zmianie,
* aktualizacja wiersza o identyfikatorze 40642, gdyż został on zmieniony na urządzeniu mobilnym,
* usunięcie wiersza o identyfikatorze 71214, gdyż został on także usunięty z urządzenia mobilnego,
* dodanie wiersza o identyfikatorze 7, gdyż nie istnieje on w bazie danych aplikacji internetowej.

Do wymiany danych pomiędzy urządzeniem mobilnym oraz aplikacją internetową został wykorzystany format *JSON* [44]. Główną jego zaletą jest uniezależnienie od języka programowania, czy też technologii i łatwość implementacji.

## Opis wdrożenia

#### Aplikacja mobilna

Aplikacja mobilna *vigyMobile* została zaprojektowana tak, aby poprawnie działać z urządzeniami mobilnymi posiadającymi wersję systemu Android 2.2 i nowszą. Aplikacja wykorzystuje bazę danych wbudowaną w system Android. Ponadto aplikacja musi posiadać uprawnienia do zapisu pamięci zewnętrznej i połączenia z Internetem.   
O przydzielenie tych uprawnień aplikacja zapyta przy instalacji.

Aby korzystać z aplikacji wystarczy skopiować plik *vigyMobile.apk* znajdujący się w katalogu bin projektu zbudowanego w środowisku programistycznymna urządzenie przenośne z systemem Android. Po uruchomieniu aplikacja jest gotowa do użytku.

#### Aplikacja internetowa

Aplikacja internetowa *vigym.com* do poprawnego działania wymaga serwera aplikacyjnego dla języka Java 1.6 lub nowszą. Aplikacja wykorzystuje bazę danych, stąd istnieje konieczność poprawnego skonfigurowania dostępu do niej przed pierwszym połączeniem. Aby to zrobić należy edytować fragment pliku   
hibernate.cfg.xml, który znajduje się w katalogu src/main/resources:

<property name="hibernate.connection.driver\_class">oracle.jdbc.driver.OracleDriver</property>

<property name="hibernate.connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>

<property name="hibernate.connection.username">test</property>

<property name="hibernate.connection.password">test</property>

<property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>

Przedstawiony powyżej fragment jest prawidłową konfiguracją dla bazy danych Oracle Express Edition 11g.

Aplikacja *vigym.com* korzysta z wielu bibliotek wspomagających. Do poprawnego działania należy zagwarantować ich obecność na serwerze aplikacyjnym. Można to zrobić na dwa sposoby:

* umieszczając biblioteki na serwerze w taki sposób, aby były one dostępne dla wszystkich aplikacji działających w obrębie danego serwera aplikacyjnego,
* umieszczając biblioteki w pliku vigym.war, który jest wynikowym plikiem kompilacji.

Następnie można umieścić plik vigym.war na serwerze aplikacyjnym. Dla serwera aplikacyjnego Oracle Weblogic 12c można skorzystać z narzędzia *Enterprise Manager*, które jest integralną jego częścią.

# Testy aplikacji

Chcąc zapewnić wysoką jakość wytworzonego oprogramowania konieczne jest wykonanie testów. Mają one na celu:

* weryfikację oprogramowania, czyli sprawdzenie, czy realizuje ono przyjęte założenia, które zostały zawarte w specyfikacji,
* walidację oprogramowania, czyli sprawdzenie, czy działanie aplikacji jest poprawne, czyli czy daje spodziewane wyniki w określonych sytuacjach.

W tym rozdziale opisany został proces testowania systemu *Vigym*.

## Testy jednostkowe

Testy jednostkowe (z ang. *unit test*) są niskopoziomową techniką weryfikacji działania pojedynczych jednostek programu. Fragment kodu, który zostaje poddany testowi jest wykonywany, a wynik jego działania jest porównywany z wartością oczekiwaną.

Niektóre języki posiadają biblioteki, które ułatwiają wykonywanie testów jednostkowych. W języku Java najpopularniejszą biblioteką służącą do testowania jednostkowego jest JUnit.

Każdy fragment logiki biznesowej systemu *Vigym* został przetestowany jednostkowo pod kątem poprawności działania.

Budując system *Vigym* wykorzystano technikę zwaną TDD. Programowanie sterowane testami (z ang. *Test Driven Development, TDD)* [45]jest techniką tworzenia oprogramowania zaliczaną do metodyk zwinnych [46]. Polega na powtarzaniu następujących kroków:

* napisanie testu jednostkowego sprawdzającego funkcjonalność. Test w tym momencie nie powinien się udać,
* implementacja funkcjonalności. W tym momencie wcześniej napisany test powinien się udać,
* dokonanie refaktoryzacji napisanego kodu, aby spełniał on przyjęte standardy.

## Przypadki testowe

Przypadek testowy jest zbiorem danych wejściowych, wstępnych warunków wykonania, oczekiwanych rezultatów i końcowych warunków wykonania. Został utworzony w celu wykonania pewnej ścieżki programu lub zweryfikowania zgodności   
z konkretnym wymaganiem. Szablon przypadku użycia składa się z:

* unikalnego identyfikatora,
* nazwy testu,
* opisu środowiska, w którym przypadek testowy jest wykonywany,
* warunków wstępnych, czyli takich, które muszą być spełnione przed wykonaniem testu,
* kroków koniecznych wykonania,
* oczekiwanego rezultatu,
* warunków końcowych, czyli takich, które muszą być spełnione po wykonaniu testu.

Przykładowe przypadki testowe dla systemu *Vigym*:

* aplikacja mobilna:

|  |  |  |
| --- | --- | --- |
| 1.1 | *Nazwa testu* | Prawidłowe dodanie nowego treningu. |
| *Środowisko* | Urządzenie mobilne z systemem Android z wersją 2.2 lub nowszą. |
| *Warunek wstępny* | Otworzone okno z listą treningów. Użytkownik aplikacji musi być zalogowany. |
| *Kroki do wykonania* | * naciśnij przycisk *Dodaj trening,* * wypełnij formularz:   + wybierając ćwiczenie z listy,   + podając ilość wykonanych powtórzeń,   + podając użyte obciążenie,   + wybierając datę wykonania ćwiczenia, * naciśnij przycisk *OK*. |
| *Oczekiwany rezultat* | Dodanie nowego treningu i przypisanie go do aktualnie zalogowanego użytkownika. Zaktualizowanie statystyk dostępnych na liście z wykonanymi ćwiczeniami. |
| *Warunki końcowe* | Dodano nowy trening. Zaktualizowano statystyki. |

* aplikacja internetowa:

|  |  |  |
| --- | --- | --- |
| 1.1 | *Nazwa testu* | Prawidłowe logowanie. |
| *Środowisko* | Dowolny system operacyjny, dowolna przeglądarka internetowa z włączoną obsługą *JavaScript*. |
| *Warunek wstępny* | Włączona przeglądarka na dowolnej stronie systemu. Żaden użytkownik nie może być zalogowany. |
| *Kroki do wykonania* | * wypełnij formularz logowania, który dostępny jest na każdej stronie aplikacji, podając:   + nazwę użytkownika,   + hasło, * naciśnij przycisk *Loguj.* |
| *Oczekiwany rezultat* | Zalogowanie użytkownika. |
| *Warunki końcowe* | Użytkownik jest zalogowany. W panelu użytkownika widoczny jest adres e-mail, jaki użytkownik podał przy rejestracji. |

## Testy interfejsu

Z punktu widzenia użytkowników systemu *Vigym* interfejs użytkownika jest najważniejszą warstwą systemu. Jest warstwą z którą osoby korzystające z systemu mają bezpośredni kontakt. Dlatego też bardzo ważne jest wykonanie odpowiednich testów interfejsu użytkownika. Spowoduje to, że aplikacja będzie użyteczna dla korzystających z niej osób.

Testowanie użyteczności interfejsu systemu *Vigym* polegało na zebraniu grupy użytkowników z różnych przedziałów wiekowych i wykonanie przez nich określonych operacji, określonych przez diagramy przypadków użycia. W testach uczestniczyło:

* 5 osób z przedziału wiekowego 15-25,
* 5 osób z przedziału wiekowego powyżej 25 lat.

Korzystanie z żadnej funkcjonalności systemu nie okazało się dla osób testujących problematyczne. Na tej podstawie można więc stwierdzić, że interfejs systemu jest użyteczny.

# Użytkowanie aplikacji

System *Vigym* umożliwia tworzenie planów treningowych. Poniższy rozdział opisuje użytkowanie aplikacji wchodzących w skład systemu: mobilnej oraz internetowej.

## Aplikacja mobilna

#### Ekran startowy

Po uruchomieniu aplikacji mobilnej ukazuje się ekran zawierający trzy przyciski (Rys. 5.1):

* *Logowanie* – powoduje przejście do ekranu logowania (Rys. 5.3),
* *Stwórz profil* – powoduje utworzenie nowego konta użytkownika w aplikacji (Rys. 5.2),
* *Wyjście* – powoduje zamknięcie aplikacji.

|  |  |
| --- | --- |
| C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-02-23-01-01.png  Rys. 5.1  Główna strona aplikacji dla użytkownika niezalogowanego. | C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-03-00-16-23.png  Rys. 5.2  Formularz umożliwiający założenie nowego konta. |

#### Tworzenie nowego konta

Aby uzyskać dostęp do funkcjonalności oferowanych przez aplikację *vigyMobile* wymagane jest posiadanie konta. Aby to zrobić należy poprawnie wypełnić formularz rejestracyjny:

* nazwa użytkownika – maksymalnie 30 znaków,
* hasło (może pozostać puste),
* adres e-mail (opcjonalnie) – maksymalnie 75 znaków.

Następnie należy nacisnąć przycisk *Rejestruj*. Jeśli wszystko zostało wypełnione poprawnie to konto zostanie utworzone. Następnie użytkownik zostanie automatycznie zalogowany do systemu i przekierowany do głównego menu aplikacji (Rys. 5.4).

|  |  |
| --- | --- |
| C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-03-00-00-19.png  Rys. 5.3  Formularz logowania. | C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-02-23-01-15.png  Rys. 5.4  Główna strona aplikacji dla użytkownika zalogowanego. |

#### Ekran logowania

Aby zalogować się do systemu należy podać następujące dane:

* nazwę użytkownika,
* hasło użytkownika.

Jeśli powyższe dane zostaną poprawnie podane to nastąpi przekierowanie do menu głównego aplikacji.

#### Menu główne

Menu główne zawiera następujące pozycje:

* *Trening* – naciśnięcie powoduje przejście do modułu trening,
* *Zarządzenie profilem* – naciśnięcie powoduje przejście do ustawień, gdzie użytkownik może zmienić hasło lub wprowadzić swoje cechy personalne (Rys. 5.5),
* *Synchronizacja / Eksport­ –* synchronizacja danych z aplikacją internetową,
* *Wyloguj ­*– wylogowanie użytkownika z aplikacji.

|  |  |
| --- | --- |
| C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-02-23-13-00.png  Rys. 5.5  Ekran cech personalnych zdefiniowanych przez użytkownika. | C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-03-00-47-32.png  Rys. 5.6  Formularz do zarządzania pomiarami cech personalnych. |

#### Zarządzanie profilem

Po wybraniu z menu pozycji *Zarządzanie profilem* użytkownik ma do wyboru dwie karty (Rys. 5.5):

* *Konto –* umożliwia zmianę hasła podanego przy rejestracji,
* *Personalne –* umożliwia zdefiniowanie cech charakterystycznych dla użytkownika i dokonywanie pomiarów.

#### Zarządzanie cechami personalnymi profilu

Użytkownik aplikacji *vigyMobile* ma możliwość zarządzania cechami personalnymi (Rys.5.5). Decyzja jakie cechy monitorować należy do użytkownika. Aby zdefiniować nową cechę wystarczy nacisnąć przycisk *Dodaj własność* – spowoduje to otworzenie okna dialogowego, w którym użytkownik powinien podać:

* nazwę własności (pole obowiązkowe),
* opis własności (pole opcjonalne),

i zatwierdzić przyciskiem *Dodaj*. Po poprawnym zdefiniowaniu własności użytkownik może edytować własność (), usunąć ją () lub dodać pomiar (). Aby dodać nowy pomiar konieczne podanie jest podanie następujących informacji w nowo otwartym oknie dialogowym:

* wartości pomiaru (liczba),
* jednostki pomiaru (wybór z listy),
* daty pomiaru,

i zatwierdzić naciskając przycisk . Dodane pomiary można edytować () oraz usuwać (). Każde dodanie, edycja lub usunięcie pomiaru powoduje zaktualizowanie informacji dostępnych na liście własności: daty ostatniego pomiaru, postępu oraz aktualnej wartości.

**Zarządzanie treningiem**

Po wybraniu z menu pozycji *Treningu* użytkownik przekierowany zostanie do modułu umożliwiającego zarządzenie treningami (Rys. 5.7). Aby dodać nowy trening należy nacisnąć przycisk *Dodaj trening.* Następnie w nowo otwartym oknie dialogowym (Rys. 5.8) należy podać:

* ćwiczenie (pole obowiązkowe), którego dotyczy pomiar, wybierając z listy dostępnych – wybranie ćwiczenia spowoduje wyświetlenie się jego opisu,
* ilość powtórzeń (pole obowiązkowe, liczba całkowita),
* użyte obciążenie (pole obowiązkowe, liczba),
* data pomiaru (pole obowiązkowe).

|  |  |
| --- | --- |
| C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-02-23-04-57.png  Rys. 5.7  Ekran cech personalnych zdefiniowanych przez użytkownika. | C:\Users\JrQ-\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_2014-09-02-23-03-25.png  Rys. 5.8  Formularz do zarządzania pomiarami cech personalnych. |

Każde dodanie, edycja () lub usunięcie () treningu powoduje aktualizację statystyk dla danego ćwiczenia dostępnych na liście ćwiczeń (Rys. 5.7):

* *maksymalnie* – maksymalna wartość obciążenia w pojedynczej próbie,
* *seria* – maksymalna wartość obciążenia w serii,
* *średnia* – średnia wartość obciążenia w pojedynczej próbie,
* *średnio w serii* – średnia wartość obciążenia w serii,
* *ostatnio –* informacje na temat ostatnio wykonanego treningu.

## Aplikacja internetowa

#### Ekran logowania

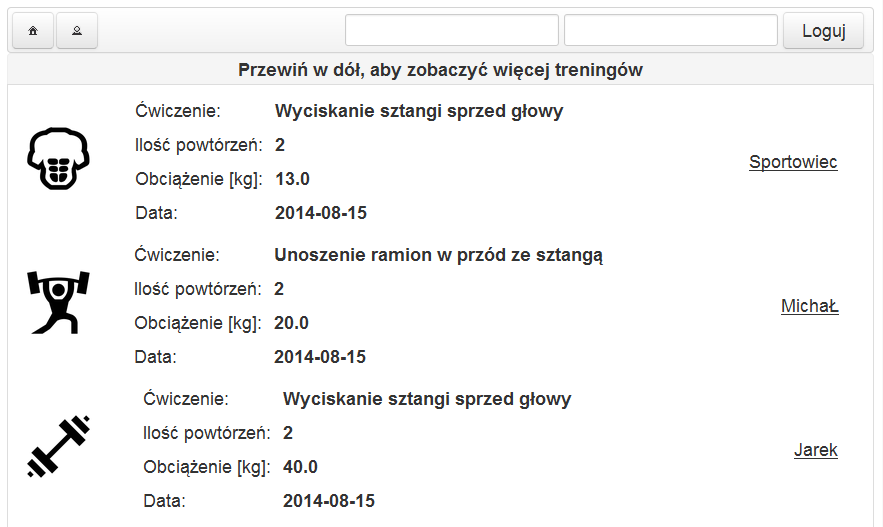
Aby zalogować się do systemu należy podać następujące dane:

* nazwę użytkownika,
* hasło użytkownika.

Panel logowania jest dostępny na każdej stronie systemu. Jeśli powyższe dane zostaną poprawnie podane panel logowania przestanie być widoczny. Jego miejsce zajmie panel użytkownika, w którym użytkownik zobaczy swój adres e-mail podany przy rejestracji konta w systemie oraz otrzyma możliwość wylogowania się z systemu.

**Strona główna**

Na stronie głównej aplikacji internetowej wyświetlana jest lista ostatnich treningów wykonanych przez wszystkich użytkowników systemu (Rys. 5.9). Treningi posortowane są malejąco według daty wykonania treningu. Oprócz daty wykonania treningu znajdują się tu informacje na temat rodzaju wykonywanego ćwiczenia, ilości powtórzeń i użytego obciążenia. Z prawej strony dostępny jest także odnośnik do profilu użytkownika w aplikacji internetowej, który wykonał trening. Domyślnie wyświetlana lista treningów zawiera dziesięć pozycji. Jeśli użytkownik przewinie listę do końca, załadowane zostanie kolejne dziesięć elementów.

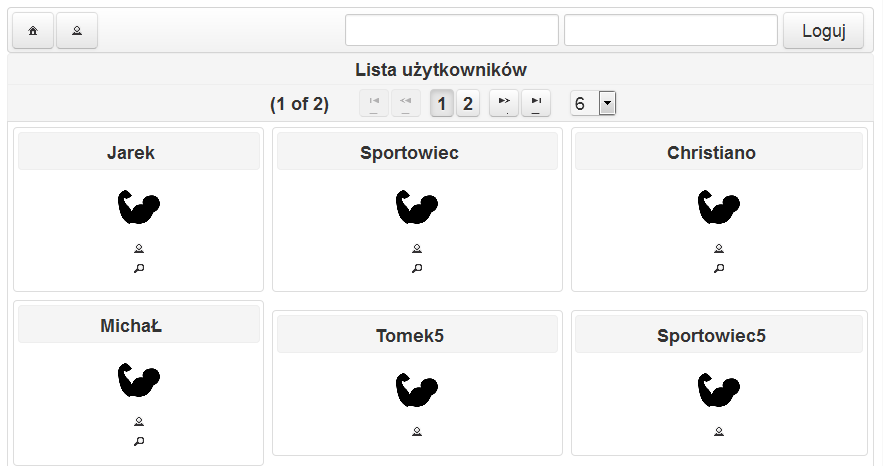
****

Rys. 5.9

Fragment strony głównej aplikacji *vigym.com.*

**Lista użytkowników**

Lista użytkowników zawiera informacje na temat wszystkich użytkowników systemu (Rys. 5.10). Strona ta zawiera tabelę z odnośnikami do profili zarejestrowanych użytkowników (), a także informację na temat ostatniego wykonanego przez nich treningu (dostępną po kliknięciu  w otwartym oknie dialogowym).

****

Rys. 5. 10

Lista zarejestrowanych użytkowników.

**Profil użytkownika**

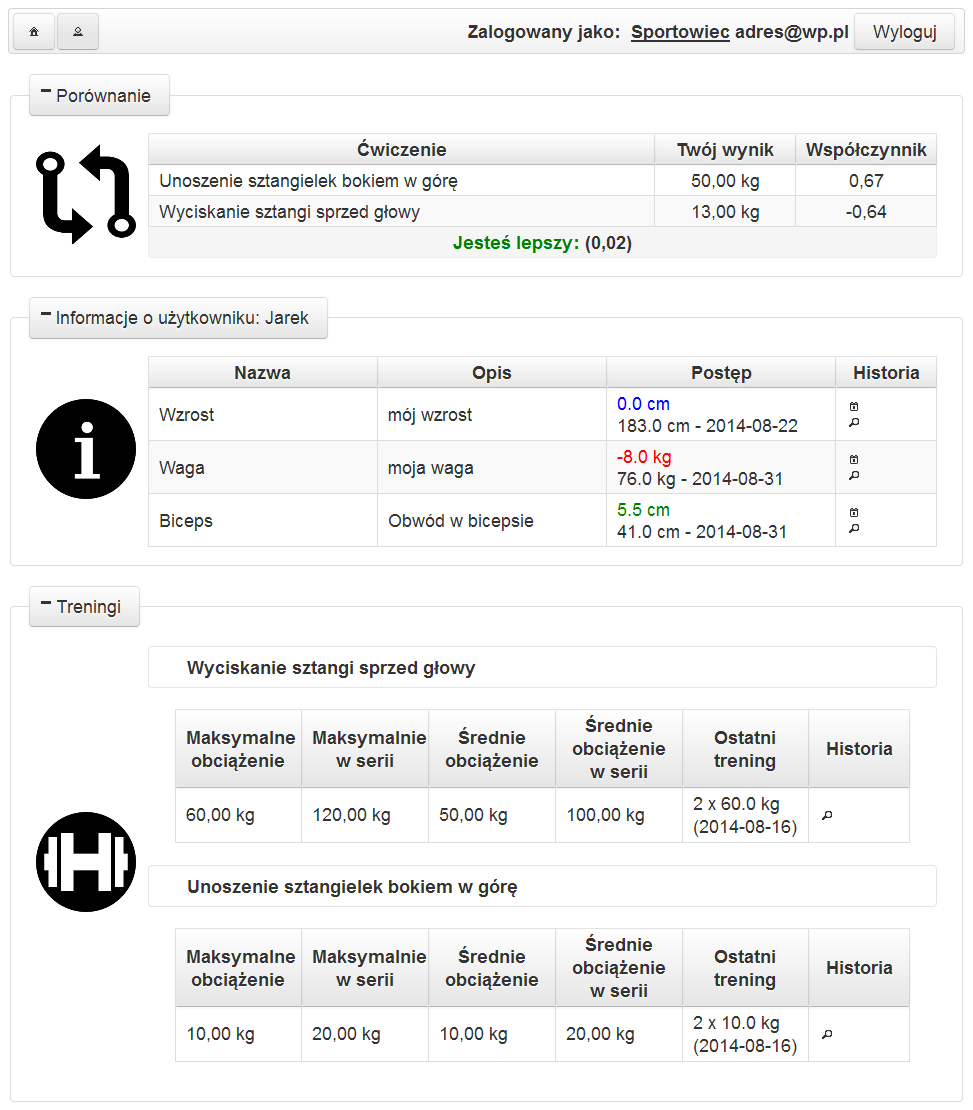
Każdy zarejestrowany w systemie użytkownik posiada własną podstronę (Rys. 5.11), na której zamieszczone są informacje na temat:

* cech personalnych zdefiniowanych przez użytkownika,
* treningów wykonanych przez użytkownika.

Dodatkowo jeśli strona jest przeglądana przez zalogowanego użytkownika wyświetlana jest sekcja zawierająca porównanie użytkownika przeglądającego profil i użytkownika do którego dany profil należy.

Profil użytkownika jest podzielony na następujące sekcje:

* *Porównanie* – dostępny jedynie dla zalogowanych użytkowników. Zawiera szczegółowe porównanie ćwiczeń, które wykonał użytkownik, którego profil jest przeglądany oraz użytkownik przeglądający. Zawiera także zbiorcze podsumowanie, wyświetlające informację, który użytkownik osiągnął sumarycznie lepszy rezultat.
* *Informacje o użytkowniku* – zawiera informacje personalne, które użytkownik do którego profil należy zdefiniował. Na podstawie pomiarów, które zostały dokonane wyświetlany jest postęp użytkownika, historia wykonanych pomiarów () oraz wykres postępu ().
* *Treningi –* sekcja ta zawiera informacje na temat treningów wykonanych przez użytkownika. Treningi zostały pogrupowane względem ćwiczenia.



Rys. 5.11

Przykładowy profil użytkownika aplikacji.

Dostępne są szczegółowe informacje na temat statystyk dotyczących ćwiczeń:

* *Maksymalne obciążenie* – maksymalna wartość obciążenia w pojedynczej próbie,
* *Maksymalnie w serii* – maksymalna wartość obciążenia w serii,
* *Średnie obciążenie* – średnia wartość obciążenia w pojedynczej próbie,
* *Średnie obciążenie w serii* – średnia wartość obciążenia w serii,
* *Ostatni trening –* informacje na temat ostatnio wykonanego treningu,
* *Historia –* podgląd dotychczas wykonanych ćwiczeń wyświetlane w oknie dialogowym.

# Wnioski

Celem niniejszej pracy magisterskiej było stworzenie systemu, który umożliwi sportowcom planowanie treningu. System składać się miał z dwóch aplikacji: mobilnej, przeznaczonej na urządzenia z systemem Android oraz internetowej. Aplikacje powinny działać niezależnie. System zapewniać miał możliwość komunikacji aplikacji, potrzebną w celu synchronizacji danych. Wszystkie założenia projektowe zostały w pełni zrealizowane. Zbudowanie aplikacji okazało się sporym wyzwaniem. Problemy integracji aplikacji wykorzystujących różne technologie należą do najczęstszych i najbardziej skomplikowanych problemów budowania systemów informatycznych.

Pomimo, że wszystkie zakładane funkcje systemu zostały w pełni zrealizowane, posiada on spore możliwości rozwoju. Jako przykładowe rozszerzenia dla aplikacji mobilnej można wymienić: dodanie możliwości tworzenia zestawów ćwiczeń, planowania czasu treningu i powiadomienia o zaplanowanym treningu, czy też komunikację użytkowników w obrębie aplikacji. Aplikację internetową można rozbudować poprzez dodanie możliwości tworzenia planów treningowych, które obecnie jest dostępne jedynie dla aplikacji mobilnej. Kierunek dalszego rozwoju aplikacji będzie wynikać z żądań jej użytkowników.

# Bibliografia

|  |  |
| --- | --- |
| [1] | „Aplikacja mobilna,” 2014. [Online]. Available: http://pl.wikipedia.org/wiki/Aplikacja\_mobilna. |
| [2] | „Mamy coraz więcej smartfonów i jesteśmy tego coraz bardziej świadomi,” 4 2 2014. [Online]. Available: http://www.tnsglobal.pl/coslychac/2014/02/04/mamy-coraz-wiecej-smartfonow-i-jestesmy-tego-coraz-bardziej-swiadomi/. |
| [3] | P. J. F. Clark, „History of Mobile Applications,” [Online]. Available: http://www.uky.edu/~jclark/mas490apps/History%20of%20Mobile%20Apps.pdf. |
| [4] | „WAP,” [Online]. Available: http://technical.openmobilealliance.org/Technical/technical-information/material-from-affiliates/wap-forum. |
| [5] | „Java ME,” [Online]. Available: http://www.oracle.com/technetwork/java/embedded/javame/index.html. |
| [6] | „Symbian OS,” 26 7 2014. [Online]. Available: http://developer.nokia.com/community/wiki/Symbian\_OS. |
| [7] | „Android and iOS Devices Account for 96% of the Global Market,” 14 8 2014. [Online]. Available: http://www.idc.com/getdoc.jsp?containerId=prUS25037214. |
| [8] | „April 2014 Web Server Survey,” 7 2014. [Online]. Available: http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html. |
| [9] | „Cienki klient,” [Online]. Available: http://pl.wikipedia.org/wiki/Cienki\_klient. |
| [10] | „JavaScript,” [Online]. Available: http://www.w3.org/standards/techs/js. |
| [11] | „Adobe Flash,” [Online]. Available: http://www.adobe.com/pl/products/flash.html. |
| [12] | „Package javax.servlet,” [Online]. Available: http://docs.oracle.com/javaee/7/api/javax/servlet/package-summary.html. |
| [13] | „AJAX,” [Online]. Available: http://www.w3.org/standards/webdesign/script. |
| [14] | „HTML5,” [Online]. Available: http://www.w3schools.com/html/html5\_intro.asp. |
| [15] | „Endomondo,” [Online]. Available: https://www.endomondo.com. |
| [16] | „Gym Book: training notebook,” 18 7 2013. [Online]. Available: https://play.google.com/store/apps/details?id=com.rk.gymstat\_limited. |
| [17] | „Redy Gym Log - Workout Tracker,” 12 8 2014. [Online]. Available: https://play.google.com/store/apps/details?id=com.redyApps.redyGymLog. |
| [18] | „Virtuagym Fitness Home & Gym,” 16 7 2014. [Online]. Available: https://play.google.com/store/apps/details?id=digifit.virtuagym.client.android. |
| [19] | „XHTML 1.0 The Extensible HyperText Markup Language,” [Online]. Available: http://www.w3.org/TR/xhtml1/. |
| [20] | B. M. K. W. Stanisław Wrycza, Język UML 2.0 w modelowaniu systemów informatycznych, Gliwice : Helion, 2006. |
| [21] | „Android for developers - documentation,” 2014. [Online]. Available: https://developer.android.com/index.html. |
| [22] | „SQLite documentation,” 2014. [Online]. Available: http://www.sqlite.org/docs.html. |
| [23] | „Android Developer Tools,” [Online]. Available: http://developer.android.com/tools/index.html. |
| [24] | „Database Browser for SQLite,” 28 8 2014. [Online]. Available: http://sourceforge.net/projects/sqlitebrowser/. |
| [25] | „Java,” [Online]. Available: https://www.java.com/pl/. |
| [26] | „JavaServer Faces documentation,” 2014. [Online]. Available: https://javaserverfaces.java.net/docs/2.2/. |
| [27] | „Hibernate ORM documentation,” 2014. [Online]. Available: http://hibernate.org/orm/documentation/. |
| [28] | „JUnit,” [Online]. Available: http://junit.org/. |
| [29] | „Primefaces,” [Online]. Available: http://primefaces.org/. |
| [30] | „Log4j,” [Online]. Available: http://logging.apache.org/log4j. |
| [31] | „Oracle Database Express Edition Documentation,” [Online]. Available: http://docs.oracle.com/cd/E17781\_01/index.htm. |
| [32] | „Oracle WebLogic Server documentation,” [Online]. Available: http://www.oracle.com/technetwork/middleware/weblogic/documentation/index.html. |
| [33] | „Eclipse,” [Online]. Available: https://www.eclipse.org. |
| [34] | „Oracle SQL Developer,” [Online]. Available: http://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/index.html. |
| [35] | „Apache Maven,” [Online]. Available: http://maven.apache.org/. |
| [36] | „Git,” [Online]. Available: http://git-scm.com/. |
| [37] | „Visual Paradigm,” [Online]. Available: http://www.visual-paradigm.com/. |
| [38] | „.Net,” [Online]. Available: http://www.microsoft.com/net. |
| [39] | R. H. R. J. J. M. V. Erich Gamma, Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku, Gliwice: Helion, 2010. |
| [40] | „Core J2EE Patterns - Data Access Object,” Oracle, 2001. [Online]. Available: http://www.oracle.com/technetwork/java/dataaccessobject-138824.html. |
| [41] | „Core J2EE Patterns - Transfer Object,” Oracle, 2001. [Online]. Available: http://www.oracle.com/technetwork/java/transferobject-139757.html. |
| [42] | „Java EE,” [Online]. Available: https://www.oracle.com/java/technologies/java-ee.html. |
| [43] | B. B. K. Sierra, Sun Certified Programmer for Java 6 Study Guide, McGraw-Hill, 2008. |
| [44] | „Format JSON,” [Online]. Available: http://www.json.org/json-pl.html. |
| [45] | „Test Driven Development,” [Online]. Available: http://msdn.microsoft.com/library/test-driven-development.aspx. |
| [46] | „Manifest Zwinnego Tworzenia Oprogramowania,” [Online]. Available: http://agilemanifesto.org/iso/pl/. |
| [47] | R. C. Martin, Czysty Kod. Podręcznik dobrego programisty, Gliwice: Helion, 2010. |

# Uruchomienie aplikacji

#### Aplikacja mobilna

Aplikację mobilną *vigyMobile* można uruchomić na dwa sposoby:

1. Jako projekt w środowisku programistycznym *Android Developer Tools*. Aby to zrobić należy:
   * + - uruchomić środowisko programistyczne,
       - wybrać z menu *File > Import*,
       - wybrać *General* > *Existing Project Into Workspace*,
       - wskazać lokalizację projektu,
       - nacisnąć *Finish*,
       - następnie klikając na zaimportowanym projekcie wybrać *Run as > Android Application*. Należy pamiętać, że urządzenie (lub emulator) powinno być podpięte do komputera przez port USB.
2. Jako aplikacja na urządzeniu mobilnym. Aby to zrobić wystarczy skopiować plik *vigyMobile.apk* na urządzenie mobilne i uruchomić.

#### Aplikacja internetowa

Aplikację internetową *vigym.com* można uruchomić na dwa sposoby:

1. Jako projekt w środowisku programistycznym *Eclipse*. Aby to zrobić należy:
   * + - uruchomić środowisko programistyczne,
       - wybrać z menu *File > Import*,
       - wybrać *General* > *Existing Project Into Workspace*,
       - wskazać lokalizację projektu,
       - nacisnąć *Finish*,
       - następnie klikając na zaimportowanym projekcie wybrać *Run as > Run on server*.
2. Jako archiwum na serwerze aplikacyjnym. Aby to zrobić, wystarczy skopiować plik *vigym.com.war* na serwer aplikacyjny i uruchomić z przeglądarki internetowej.

# Zawartość płyty CD

|  |  |  |  |
| --- | --- | --- | --- |
| .. |  |  |  |
| /doc |  |  |  |
|  | Praca.docx |  | *Niniejsza praca w formacie docx.* |
|  | Praca.pdf |  | *Niniejsza praca w formacie pdf.* |
| /src |  |  | *Katalog z kodem źródłowym aplikacji.* |
|  | /vigyMobile |  | *Kod źródłowy aplikacji mobilnej.* |
|  | /vigym.com |  | *Kod źródłowy aplikacji internetowej.* |