**WOJSKOWA AKADEMIA TECHNICZNA**

**im. Jarosława Dąbrowskiego**

**w Warszawie**



TECHNOLOGIE INTERNETOWE I MOBILNE

Sprawozdanie z projektu:

**Aplikacja zliczająca kroki „Movester”.**

**Prowadzący:**

ppor. mgr inż. Damian Frąszczak

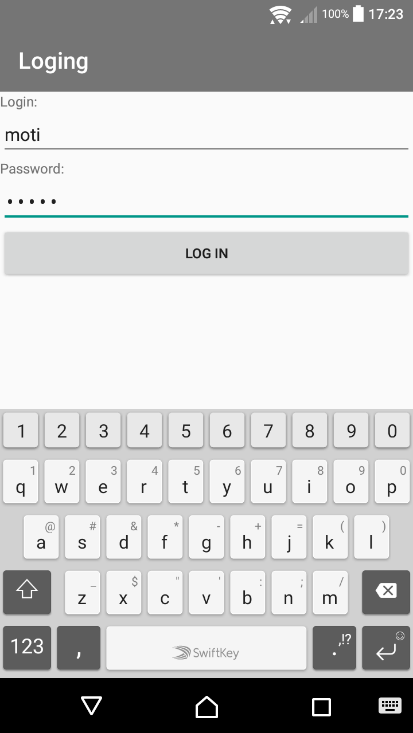
grupa dziekańska I6B1S1

1. **Aplikacja dedykowana na smartfony z systemem Android**

Wszystkie zrzuty ekranu aplikacji pochodzą z telefonu Sony Xperia E5, na którym zostały przeprowadzone testy oprogramowania.

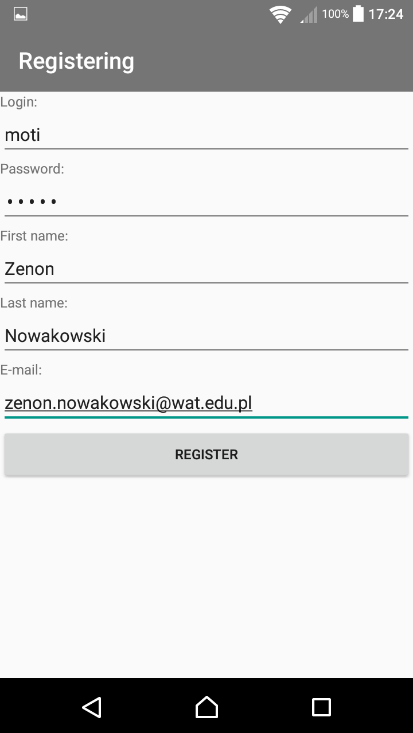
* „Namacalne” metody

Część metod jest związana bezpośrednio z przyciskami znajdującymi się w layoutach odpowiadających aktywnościom. Poniżej ich krótki opis wraz z prezentacją wizualną aplikacji mobilnej w praktyce.



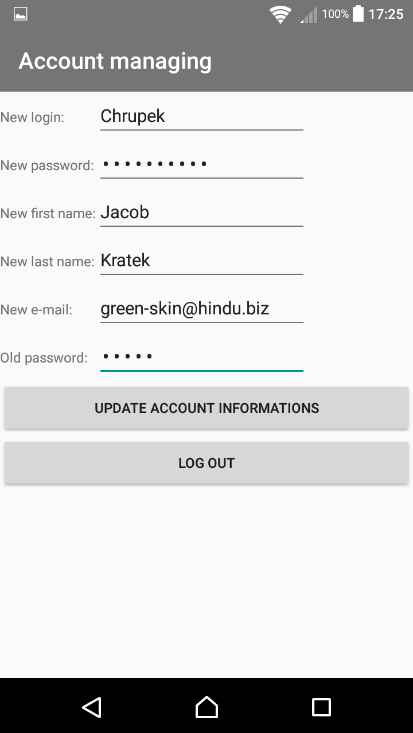
*void onLogInClick()*

Wysyła na zewnątrz aplikacji login z hasłem, a „w zamian” pobiera token oraz numer identyfikacyjny użytkownika.



*void RegisterClick()*

Wysyła dane z formularza, na podstawie których zostanie utworzony nowy użytkownik.

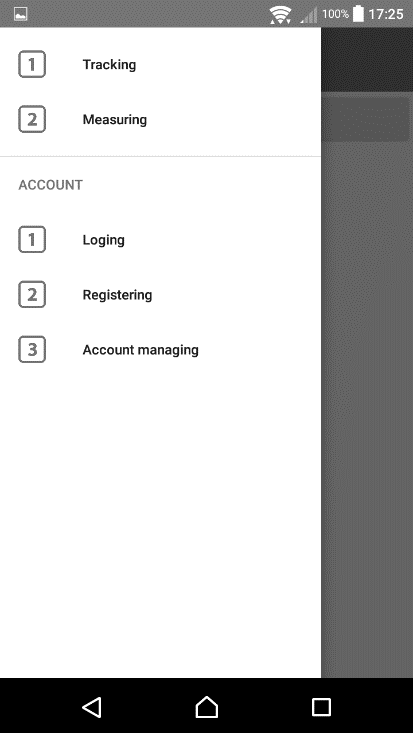


*void onUpdateAccountInformationsClick()*

Wysyła dane z formularza, które są wykorzystane do edycji danych użytkownika. Odpowiednik rejestracji, który nie tworzy nowego użytkownika w bazie danych.

*void onLogOutClick()*

Czyści informacje na temat przechowywanego tokena i numeru identyfikacyjnego użytkownika, nie mamy więc po wylogowaniu dostępu do metod przeznaczonych dla zalogowanych użytkowników.



*void onStartMeasuringClick()*

Rozpoczyna pomiar.



*void onEndClick()*

Kończy pomiar, wysyła o nim informacje zbiorcze.

* Pozostałe ważne metody

*JSONObject toJson(String)*

Pozwala na łatwiejszą komunikację z serwerem, tworzy obiekt typu JSON z tekstu zawartego w HTML’u.

*JSONArray toJson(String)*

Analogiczna do powyższej metody. Tworzy tablicę typu JSONArray.

*void createLocationRequest()*

Nawiązuje połączenie z sensorem GPS.

*void startLocationUpdates()*

Zezwala aplikacji (za zgodą użytkownika) na korzystanie z sensora GPS.

*void onLocationChanged()*

Metoda wywoływana przez sensor – zapisuje aktualne położenie i czas.

*void addMarker()*

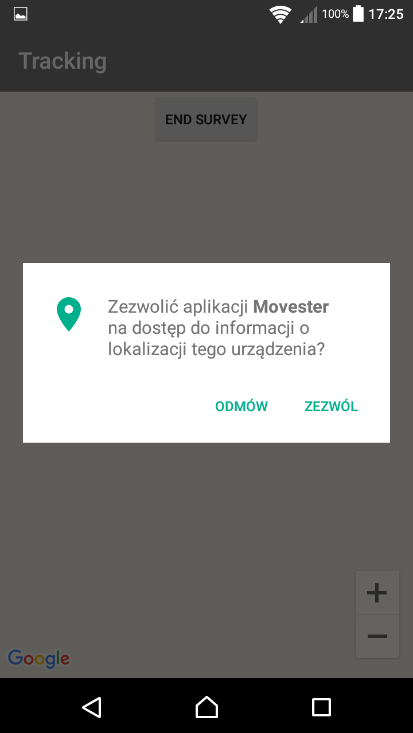
Tworzy na mapie znacznik. Ta metoda wywołuje także inne, służące do komunikacji z serwerem, w celu przesyłania danych na bieżąco - także w tym momencie wykonywane są lokalne obliczenia.

*void onSensorChanged()*

Metoda korzystająca z sensora, wykonująca się tylko na niektórych telefonach – zlicza kroki. W przypadku, gdy telefon nie posiada odpowiedniego sensora, liczba kroków jest szacowana.

*double distance(LatLng, LatLng)*

Oblicza odległość między dwoma punktami na mapie w metrach, zapisanymi w domyślnym dla API formacie LatLng.



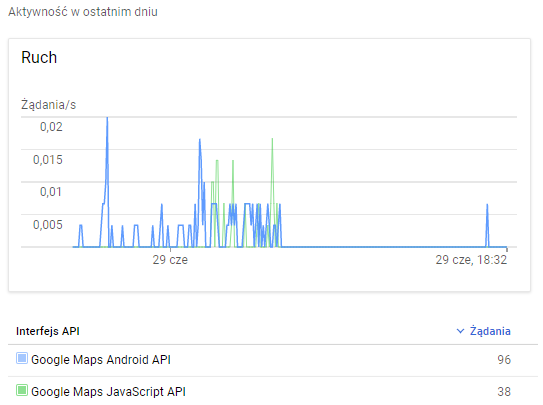
Opisane wcześniej żądanie dostępu do sensora GPS.

* Wykorzystane technologie (wybrane)

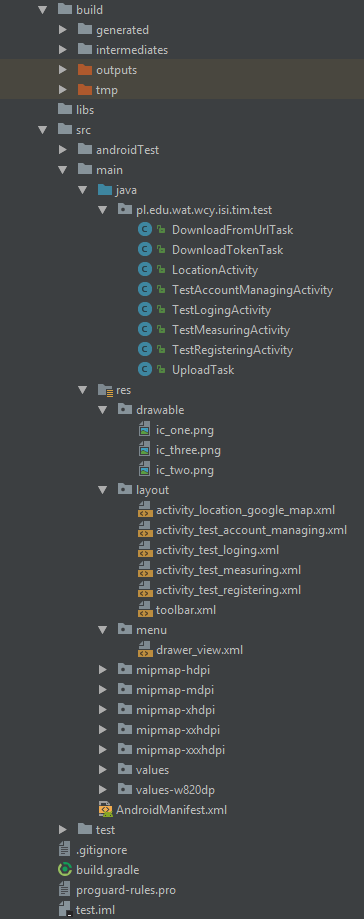
*AsyncTask* – wykorzystywane, kiedy wymieniamy informacje z serwerem. Dzięki nim unikamy „zawieszania” aplikacji, gdyż wszystkie operacje wykonywane są w tle. W naszej aplikacji utworzono trzy podobne AsyncTaski, wykorzystywane naprzemiennie w zależności od sytuacji.

*NavigationView* – biblioteka, która pozwoliła na utworzenie menu wysuwanego z lewej strony ekranu. Oparta, tak jak layouty, o plik .xml, znajduje swoje odbicie w każdej aktywności aplikacji.

*Google Maps Android API* – pozwala na dodanie do aplikacji mapy, a także udostępnia szereg metod, dzięki którym łatwiejsze staje się wykonywanie na nich obliczeń.



Wykres przedstawiający wykorzystanie interfejsów API w platformie Google Cloud.



Tak prezentuje się cały moduł, na zrzucie ekranu widać rozwinięte tylko kluczowe foldery.

1. **Motysek**

Blabla – wiem, że formatowanie jest słabe, ale niech już będzie jednolite.

1. **Lew**

Blabla – ok?

1. **Praca zespołowa**

Komunikowaliśmy się korzystając z platformy GitHub…

Lew, napiszesz też coś o tym? Masz 5, a my biedni nie, musimy się uczyć do egzaminu ☹