TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Arvutiteaduse instituut
Võrgutarkvara õppetool

# Analysis and development of a social networking mobile application for members of motorcycle clubs

bakalaureusetöö

Üliõpilane: Alexander Gvozdkov
Üliõpilaskood: 093495IAPB
Juhendaja: Ago Luberg

Tallinn
2015

# Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

(*kuupäev*) (*allkiri*)

# Annotatsioon

Töö eesmärk on analüüsida ja arendada mobiilne rakendus mootorrattaklubide liikmete jaoks. Antud rakendus peab olema funktsionaalsusega, mis on sarnane teiste sotsiaalvõrgustikega, võimaldama kontakteeruda teiste klubide liikmetega ja küsida tee peal abi. Rakenduse sees peab olema integreeritud kaart, mille abil saab mootorrattaklubid üles leida.

Mobiilsete rakenduste hulgas hetkel puudub spetsialiseeritud tarkvara mootorrattaga reisijate ja mootorrattaklubide liikmete jaoks. Ükski rakendus ei anna võimalust reisides mootorrattaklubide asukohti vaadata, küsida tee peal abi, kui mootorratas läheb katki või kütus saab otsa. Välismaal on ratturitel keeruline orienteeruda ja nad vajavad kaarti teiste klubide ülesleidmiseks.

Antud töös on teostatud mootorrattaklubide jaoks spetsialiseerunud mobiilsete rakenduste turuanalüüs. Analüüsist järeldub, et antud valdkonnas puudub vajalik tarkvara kasutajate jaoks. Töö tulemusena on arendatud mobiilne rakendus sotsiaalse võrgustikuga, mis annab võimaluse välismaal reisides mootorrattaklubid üles leida, küsida abi teistelt kasutajatelt ja suhelda teiste klubide liikmetega.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 45 leheküljel, 8 peatükki.

# Abstract

The goal of the work is to analyse and develop a mobile application for members of motorcycle clubs. This application should have functionality similar to other social networks; give the ability to get in contact with members of other clubs and to ask for assistance on the road. The application should have an integrated map which allows finding motorcycle clubs.

At the present period of time there is a lack of software specialized for motorcycle travellers and members of motorcycle clubs in the mobile devices applications market. None of the current available applications provide the ability to find and view motorcycle clubs on the map while traveling, to ask for assistance when the motorcycle breaks down or runs out of fuel. For motorcyclists it is difficult to navigate abroad in foreign countries and they require a map to find other clubs in the area.

The market of mobile applications specialized on motorcycle clubs was analysed. The conclusion was made that in this area there was a lack of required software. As a result, a mobile social networking application was developed which allows finding motorcycle clubs, ask for assistance on the road and communicate with the members of other clubs.

The thesis is in English and contains 45 pages of text, 8 chapters.

# Glossary of Terms and Abbreviations

| | |
|---|---|
| **Sotsiaalne võrgustik** | ***Social network***<br>Suhete võrgustik, mis ühel inimesel on teiste inimestega ja nendest inimestest moodustunud gruppidega. |
| **Android** | ***Android***<br>Avatud lähtekoodiga mobiilne operatsioonisüsteem. |
| **Android Tarkvara Arenduse Komplekt** | ***Android Software Development Kit (SDK)***<br>Tarkvara arenduse komplekt Androidi jaoks. |
| **Java** | ***Java***<br>Platvormist sõltumatu objektorienteeritud programmeerimiskeel. |
| **HTML** | ***HyperText Markup Language (HTML)***<br>Keel, milles märgendatakse veebilehti. |
| **JavaScript** | ***JavaScript***<br>Objektorienteeritud programmeerimiskeel, mida kasutatakse peamiselt veebilehtede skriptimiseks. |
| **PostgreSQL** | ***PostgreSQL***<br>Objekt-relatsiooniline andmebaasi juhtimissüsteem. |
| **Node.js** | ***Node.js***<br>Tarkvaraplatvorm, mis võimaldab programmeerida serveripoolseid rakendusi kasutades JavaScript'i. |
| **Klient** | ***Client***<br>Protsess või arvuti, mis tarbib teise protsessi või arvuti teenuseid. |
| **Server** | ***Server***<br>Arvutisüsteem või selles töötav tarkvara, mis pakub teatud infoteenust sellega ühendatavatele klientidele. |

# Table of Contents

# 1. Introduction

The goal of the work is to analyse and develop a mobile application for members of motorcycle clubs. The application should have social networking functionality, provide the ability to contact other club members and request for assistance on the road. A map should be integrated into the application to help users find motorcycle clubs.

## 1.1 Background and problem

At the present time, there is a lack of specialized software for motorcycle travellers and members of motorcycle clubs on mobile devices. Right now there are no applications which allow viewing the locations of motorcycle clubs on the map while traveling and request for assistance on the road if the motorcycle breaks down or runs out of fuel. Motorcyclists who travel abroad in foreign countries require a map to navigate to other clubs.

The main target user base are members of motorcycle clubs who travel abroad in different countries.

The application is developed in Tallinn, Estonia. The distribution of the application will be done through Google Play Market and available for download for international users.

The application is under continuous development from November 2014 for the members of motorcycle clubs. The development team consists of one person, the author of the thesis.

## 1.2 Goal establishment

Analyse the mobile application market specialized for members of motorcycle clubs. Find out the needs and requirements of the users in this market. Develop a social networking mobile application for members of motorcycle clubs fitting these requirements.

## 1.3 Methodology

Research of the available software on the market was carried out on the internet using search engines and Google Play Market. The requirements of the users were found out by talking to

members of one of the motorcycle clubs in Tallinn, Estonia which requested to stay anonymous. The client side of the application was developed on Android using Android Development Kit on Java. The server was created using node.js and PostgreSQL for the database back-end. The maps with club locations were integrated into the application using Google Maps.

## 1.4 Work overview

The work has been divided into five main parts: market analysis, goal establishment, technology research, project development and results:

- Market analysis has been done to find available applications on the market and users have been interviewed to find out their needs and requirements.
- Goals were established to develop a mobile application for members of motorcycle clubs with the required features.
- Available technologies for mobile applications have been reviewed. It was decided to develop for Android using Android Studio.
- The project has been developed and development process has been described in the paper.
- As a result, the application has been published for testing, users were asked to leave their feedback and a conclusion has been written.

# 2. Market analysis

## 2.1 Estonian motorcycle statistics

In Estonia, according to the Estonian Road Administration's statistics as of 01.12.2014, there are a total of 795024 registered motor vehicles out of which 42174 are motor vehicles of category L (Mopeds, Motorcycles, Motor Tricycles and Quadricycles) [1].

Motorcyclists and members of motorcycle clubs travel on Motorcycles without a sidecar (category L3e), Motorcycles with a sidecar (category L4e) and Motor Tricycles (category L5e) as they have a maximum design speed of more than 45 km/h.

According to the statistics mentioned above, the total number of registered motor vehicles of categories L3e, L4e and L5e in Estonia is 24238 which is a significant number, considering that the length of the motorcycling season in Estonia is quite short due to the cold climate.

According to biker.ee, there are 41 listed motorcycle clubs in Estonia. However, it is important to note that some clubs do not have their own website and are not listed.

## 2.2 OS Platform Statistics and Trends

According to statistics collected from W3Schools's log-files since 2003 [2], the usage of mobile operating systems has steadily grown since 2011 from 0.7% to 4.5% in November 2014.

The most popular mobile platform is Android at 2.72%, followed by iOS at 1.12% in November 2014 [3].

Considering that the usage of the iOS platform has slowly been on the decline since January 2014 from 1.23% to 1.12% and that the popularity of Android has risen by 0.79% during the same time period, Android has a better standing on the mobile device market.

## 2.3 Internet search engine based analysis

The general search terms for possible mobile applications for motorcyclists for Android devices would be „android, navigation, motorcycle, travel, weather, map, coordinates, moto, road, motor club, motorcycle clubs, road help, assistance, emergency, app and application“.

Using the above search terms, these are the most popular applications found using internet search engines for navigation:

- Google Maps navigation app with 220 comprehensive, accurate maps in 220 countries and territories, voice-guided GPS navigation, live traffic conditions [4].
- Map Factor: GPS navigation which is a turn-by-turn GPS navigation app using OpenStreetMap data, does not require an internet connection while traveling [5].
- Waze community-based traffic and navigation app where drivers share real-time traffic and road information [6].

Applications useful for data collection:

- Diablo Superbiker app which tracks laps, speed, lean angle and lap times using smartphone’s GPS and accelerometers [7].
- EatSleepRide to track rides, create groups, check progress of routes, auto detects motorcycle crashes [8].

For communication:

- Facebook which allows to keep up with friends, share updates, photos and videos [9].
- WhatsApp cross-platform mobile messaging app [10].

Applications for requesting assistance:

- AAA Mobile provides roadside assistance for members of the American Automobile Association [11].
- Good Sam Roadside Assistance offering roadside services in America. [12].

Other useful applications:

- Minutecast by Accuweather for weather forecasts [13].
- Yelp app to find local businesses such as restaurants, bars, health and medical institutions, automotive repair shops and other services [14].

## 2.4 User based analysis

For this paper, members of one of the motorcycle clubs in Tallinn, Estonia were questioned about what functionality they would like to see in a mobile application in order to find out the needs and requirements of the users. One of the authors' friends is a member of one of the clubs and helped to conduct the research. The club has requested to keep anonymity.

### 2.4.1 User needs and requirements

According to the interviews, the perfect mobile application would have the functionality to track routes, distance travelled, fuel consumption, speed, includes maps with turn-by-turn navigation. The map should have locations of local businesses such as food places, gas stations and repair shops for motorcyclists with the ability to review them. Members of motorcycle clubs would like to able to see other clubs on the map and to be able to request roadside assistance in case of an accident or an emergency. The application should have a friends list, a motorcycle club page and a profile page for communication.

## 2.5 Current problems on the market

The market analysis has shown that there is a wide variety of applications for the Android platform on mobile devices for navigation, communication, data collection, weather forecasts, roadside assistance and other purposes which could be used by motorcyclists while traveling.

However, there was a lack of software dedicated to motorcycle clubs. Few applications were designed to be used specifically by motorcyclists and very few results were found using the search term „motor club app". None of the available applications had the full set of the required functionality to display locations of motorcycle clubs, allow users to request roadside assistance, keep a list of friends and have your own motorcycle club page.

In addition, some of the software was only designed to be used in the US and required either a membership fee or had a fixed price to download.

# 3. Goal establishment

## 3.1 Requirements

The application should work on mobile devices. Users must be able to communicate with each other. Users are able to edit their profile and create a motorcycle club page. Users have the ability to request roadside assistance. Users are able to see motorcycle clubs on the map.

The application requires internet access for authentication and communication. It requires a database to store user information as well as a server to handle client requests.

## 3.2 Work goals

Develop a social networking mobile application for members of motorcycle clubs. Create the main page for authentication. Develop the user home screen menu with access to friends list, profile page, motorcycle club page and roadside assistance menu. Integrate maps into the application. Create a database with user information. Add motorcycle club locations to the database. Setup a server to handle client requests.

# 4. Technology research

Different development frameworks and technologies for mobile application development have been analysed. The required technologies have been divided into three parts: client, server and database.

According to the market analysis conducted in this paper (see paragraph 2, section 2.2), the most popular mobile operating system is currently Android. As a result, different tools were researched for the development on this specific platform.

## 4.1 Client side

The author has never developed for Android before and has decided to review available development options.

For Android, it is possible to develop mobile applications using HTML5, HTML5 packaged in native shell, C# compiled into native Android applications, C++ or Java. Android has Linux underneath so it is possible to run many languages on it.

Development using HTML5 does not require any specific tools. It is quick and simple, applications update instantly, it works on all platforms. However, the user interface is never as responsive and smooth as on native applications, some device capabilities are limited or unavailable.

HTML5 packaged in native shell is possible using tools such as PhoneGap and Icenium. These tools have a HTML5 and Javascript reusable code base which is not specific to any platform, support more device functionality than pure HTML5. However, the performance of the applications is still limited compared to other solutions.

Xamarin platform allows to target iOS, Android and Windows with a single, shared C# codebase [15]. The apps are built with standard, native user interface controls. As a result, applications are responsive and have a native look and feel. On the other side, the platform requires a subscription paid monthly or annually and requires expensive packages to use the full functionality such as Visual Studio support. The standard free edition is limited and restricts applications from invoking third-party libraries.

Qt is a cross-platform application and user interface framework for developers using C++ or QML, a CSS and JavaScript like language [16]. Even though the user interface is created with QML, the applications have a strong C++ backend. This allows direct native access to OpenGL graphics. As a result, the developers are able to create powerful applications using QML with high levels of abstraction which simplifies the development process as QML is easier to use than C#, C++ or Java. However, the framework is still relatively new and is under constant changes and development, searching for code examples is difficult, commercial licensing requires a monthly subscription.

The official development framework for Android is Android Studio which uses the Android Software Development Kit [17]. The development is done using Java. The applications are responsive, have native look and feel. The framework does not require monthly payments. There are a lot of available tutorials and code examples. On the other side, cross-platform development is not available and there is currently no easy way to fully convert an application developed with Android SDK to other platforms such as iOS.

The author of the paper has decided to target the Android platform and does not require a framework with cross-platform development functionality. Considering that the budget is limited, tools that do not require a subscription fee were preferable. Native look and feel with good performance are necessary. As a result, the Android Software Development Kit was chosen for this project.

## 4.2 Server side

According to W3Techs' Web Technology surveys, the most popular web server is Apache at 58.9% [18] and the most popular server-side programming language is PHP at 82% on 19 December 2014 [19].

Apache used together with PHP is an old and stable combination. Many popular development frameworks such as Yii [20], Laravel [21], Code Igniter [22], Cake PHP [23] and content management systems such as Wordpress [24], Drupal [25] and Joomla [26] use PHP which, in my opinion, is an easy language for beginners and has a lot of development support material available.

However, the project does not need to serve static web content. One of the goals of the development process is to allow communication between members of motorcycle clubs, the server must be able to handle multiple concurrent client requests, web socket connections and scale well in case of an increased user base.

Considering these requirements, it was decided to use Node.js instead which is a platform built on the V8 Javascript Engine written in C++ and used in Google Chrome, the open source browser from Google [27]. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices [28]. Compared to Apache with PHP, it is asynchronous which allows it to handle many concurrent requests at the same time and requires fewer resources for real-time communication over the internet [29]. In addition, Node.js has a Node Package Manager which allows quickly installing additional tools such as the Express.js framework [39] with a robust set of features for web and mobile applications and Socket.io which is a reliable real-time engine that enables bidirectional event-based communication [30].

## 4.3 Database back-end

Database Management Systems are currently divided into Relational Database Management Systems and NoSQL Database Systems. The relational model fully supports ACID (Atomicity, Consistency, Isolation, Durability) which is a set of properties that guarantee the database transactions are processed reliably. This data model is specific and organized. The NoSQL approach eliminates strict relations, has the capability to store unstructured, semi-structured or structured data, allows working with the data efficiently for specific use cases such as cloud computing and provides horizontal scalability to spread databases among multiple servers [31].

Considering that the relational model is more robust and mature, and that the author is more familiar with it, it was decided to use PostgreSQL which is an open source object-relational database management system [32]. It has more than 15 years of active development, is proven to be reliable, does not have licensing fees compared to commercial solutions from Oracle and Microsoft, and strongly conforms to the ANSI-SQL: 2008 standard. It has an unlimited maximum database size and an unlimited number of rows per table. The maximum table size is 32 TB, maximum row size of 1.6 TB and 250 – 1600 columns per table depending on

column types [33]. Furthermore, there is a plugin available for the Node.js server to communicate with the PostgreSQL database [34].

# 5. Application development

## 5.1 Workstation setup

The workstation is a laptop running Windows 8.1 Professional. Considering the project requirements, Android Studio, Node.js server and PostgreSQL for the Windows platform have been installed.

### 5.1.1 Android Studio

The latest version of Android Studio for Windows has been installed. Android Studio includes the Android Studio IDE, Android SDK tools and has support for Android 5.0 (Lollipop) Platform and has the Android 5.0 emulator system image with Google APIs.

#### 5.1.1.1 Download and installation

Android Studio has been downloaded and installed following the developer documentation [35].

To set up Android Studio on Windows:

1. Launch the `.exe` file you just downloaded.
2. Follow the setup wizard to install Android Studio and any necessary SDK tools.

   On some Windows systems, the launcher script does not find where Java is installed. If you encounter this problem, you need to set an environment variable indicating the correct location.

   Select **Start menu > Computer > System Properties > Advanced System Properties**. Then open **Advanced tab > Environment Variables** and add a new system variable `JAVA_HOME` that points to your JDK folder, for example `C:\Program Files\Java\jdk1.7.0_21`.

The individual tools and other SDK packages are saved outside the Android Studio application directory. If you need to access the tools directly, use a terminal to navigate to the location where they are installed. For example:

```
\Users\<user>\sdk\
```

In this paper, the project is being developed and tested on a real Android device. The device being used is the Samsung Galaxy S3 mobile phone, model GT-I9300, Android version 4.1.2.

Each time the application is compiled it is automatically installed directly on the device from Android Studio for testing and debugging.

Debugging over USB has been enabled on the device in the Developer Options.

The device requires an Android USB driver for Windows. The Android Google USB driver has been installed using the Android SDK Manager Tool that is included with the Android SDK. The Google USB Driver is located in `<sdk>\extras\google\usb_driver\`

When plugged in over USB, it is possible to verify that the device is connected by executing `adb devices` from the SDK `platform-tools/` directory. If connected, the device name will be listed as a "device."

#### 5.1.1.2 Project setup and configuration

Android Application Modules are the modules that eventually get built into the `.apk` files based on the project build settings. They contain things such as application source code and resource files. Most code and resource files are generated for the developer by default, while others should be created if required. The following directories and files comprise an Android application module:

`build/`
: Contains build folders for the specified build variants. Stored in the main application module.

`libs/`
: Contains private libraries. Stored in the main application module.

`src/`
: Contains the stub Activity file, which is stored at `src/main/java//ActivityName>.java`. All other source code files (such as `.java` or `.aidl` files) go here as well.

`androidTest/`
: Contains the instrumentation tests.

`main/java/com.>project<.>app<`
: Contains Java code source for the app activities.

`main/jni/`
: Contains native code using the Java Native Interface (JNI).

`main/gen/`
: Contains the Java files generated by Android Studio, such as the `R.java` file and interfaces created from AIDL files.

`main/assets/`
: This is empty. Can be used to store raw asset files. Files that are saved here are compiled into an `.apk` file as-is, and the original filename is preserved. It is possible navigate this directory in the same way as a typical file system using URIs and read

files as a stream of bytes using the `AssetManager`. For example, this is a good location for textures and game data.

`main/res/`

Contains application resources, such as drawable files, layout files, and string values in the following directories.

`anim/`

For XML files that are compiled into animation objects.

`color/`

For XML files that describe colors.

`drawable/`

For bitmap files (PNG, JPEG, or GIF), 9-Patch image files, and XML files that describe Drawable shapes or Drawable objects that contain multiple states (normal, pressed, or focused).

`layout/`

XML files that are compiled into screen layouts (or part of a screen).

`menu/`

For XML files that define application menus.

`raw/`

For arbitrary raw asset files. Saving asset files here is essentially the same as saving them in the `assets/` directory. The only difference is how you access them. These files are processed by aapt and must be referenced from the application using a resource identifier in the `R` class. For example, this is a good place for media, such as MP3 or Ogg files.

`values/`

For XML files that define resources by XML element type. Unlike other resources in the `res/` directory, resources written to XML files in this folder are not referenced by the file name. Instead, the XML element type controls how the resources defined within the XML files are placed into the `R` class.

`xml/`

For miscellaneous XML files that configure application components.

`AndroidManifest.xml`

The control file that describes the nature of the application and each of its components. For instance, it describes: certain qualities about the activities, services, intent receivers, and content providers; what permissions are requested; what external libraries are needed; what device features are required, what API Levels are supported or required; and others.

`.gitignore/`

Specifies the untracked files ignored by git.

`app.iml/`

IntelliJ IDEA module

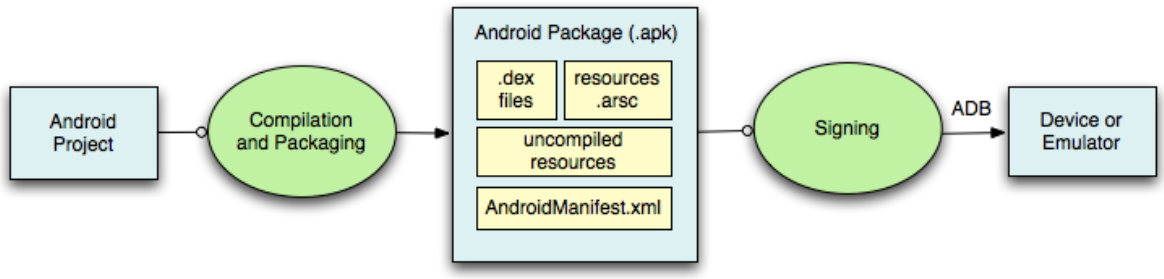
`build.gradle`

Customizable properties for the build system. It is possible to edit this file to override default build settings used by the manifest file and also set the location of the keystore and key alias so that the build tools can sign the application when building in release mode. This file is integral to the project, should be maintained in a source revision control system.

`proguard-rules.pro`

ProGuard settings file.

The following diagram depicts the components involved in building and running an application:




The Android build process provides project and module build settings so that the Android modules are compiled and packaged into `.apk` files, the containers for the application binaries, based on project build settings. The apk file for each app contains all of the information necessary to run the application on a device or emulator, such as compiled `.dex` files (`.class` files converted to Dalvik byte code), a binary version of the `AndroidManifest.xml` file, compiled resources (`resources.arsc`) and uncompiled resource files for the application.

To run an application on an emulator or device, the application must be signed using debug or release mode. The developer typically wants to sign the application in debug mode when developing and testing the application, because the build system uses a debug key with a known password so the developer does not have to enter it every time the application is built. When the developer is ready to release the application to Google Play, he must sign the application in release mode, using his own private key.

In this paper, the project has been manually signed with the debug key using the Key and Certificate Management Tool keytool from Oracle.

### 5.1.2 Node.js server

The server side of the project uses Node.js combined with Express.js which is a minimalist web framework for Node.js.

#### 5.1.2.1 Download and installation

The Node.js platform has been downloaded from the official website [36] and installed which enables the use of the Node.js package manager npm tool.

A directory has been created for the server application:

```
$ mkdir motoclubserver
```

In this directory a package.json file has been created with the npm init command:

```
$ npm init
```

Express.js has been installed and saved to the dependencies list:

```
$ npm install express --save
```

The package.json file describes the project dependencies and the modules are stored in the application's node_modules folder.

#### 5.1.2.2 Server setup and configuration

The server application functions are stored in the application folder's JavaScript file server.js.

Example of a basic Express application can be found in Appendix 2.

In this project, modules for password hashing [37], token generation [38], communication with the PostgreSQL database [34] and HTTP communication [39] have been added.

### 5.1.3 PostgreSQL database

The database back-end of the project is using the PostgreSQL database.

#### 5.1.3.1 Download and installation

The software has been downloaded from and installed for Windows from the official website [40].

The graphical installer for PostgreSQL includes the PostgreSQL server, pgAdmin III; a graphical tool for managing and developing databases, and StackBuilder; a package manager that can be used to download and install additional PostgreSQL applications and drivers.

#### 5.1.3.2 Database setup and configuration

The database server version is PostgreSQL 9.3 which was configured to run on the default port 5432 to communicate with the Node.js server. User password was set for authentication.

## 5.2 Android client development

The Android client has been developed using the Android Studio framework.

### 5.2.1 Android client structure

The project inside of the development framework has been created and called 'ee.motoabi'.The main parts of the Android client are the folders app/manifests, app/java/ee.motoabi, app/res/drawable, app/res/layout, app/res/values and build.gradle scripts.

It is important to note that the latest version of Android Studio uses Gradle which combines the flexibility of Ant with the dependency management and conventions of Maven.

The AndroidManifest.xml file inside of the app/manifests folder describes the nature of the application and each of its components. The required application permissions are set using the <uses-permission> tags:

```
<uses-permission android:name="android.permission.INTERNET" />
```

In this project, the application requests permissions for internet connectivity, accessing the network state, accessing user location data and writing to storage.

The file also holds the value of the API key required to access Google Maps:

```
<meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="key"/>
```

The app/res/drawable folder contains the application's logo seen on the home screen as well as the launcher icons and gradients used for backgrounds.

The app/java/ee.motoabi folder contains the main logic of the client application. It consists of multiple Java classes, most of which are Fragments which display the user interface.

The app/res/layouts folder contains XML files that are compiled into screen layouts for the Fragments.

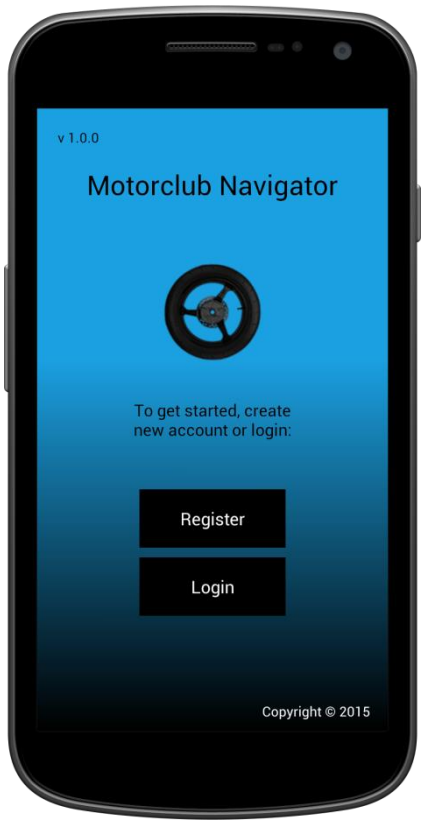
#### 5.2.1.1 Activity and fragments

The Activity class takes care of creating a window for which the developer can place the user interface. In this project, the MainActivity file hosts all of the required Fragments by using the FragmentManager class.

When the application is first loaded, the MainActivity loads up the SplashFragment which contains the interface for the home screen.

If required, the fragments are added to the back stack which allows the user to load the previous fragment by pressing the back button on the mobile device.

The fragments use the layouts taken from the app/res/layout directory. For example, the SplashFragment uses the splash.xml file which contains the user interface with a rotating logo, the version of the application, a welcome message, login and registration buttons:



### 5.2.2 Client authentication

The home screen of the application contains a login button and a registration button. Each button has a click listener defined in the SplashFragment class. The Authenticator class has been created for client authentication which communicates with server.

#### 5.2.2.1 User registration

In order to create a new account, the user taps the Register button. The RegistrationFragment is shown which contains input for the user's email and password. The user has to enter the password two times to prevent errors:


When the user taps the Confirm button, the Authenticator class sends the credentials to the server using name and value pairs. The server receives the email and password values, performs 2^12 = 4096 rounds of generating salt and hashing the password:

```
var salt = bcrypt.genSaltSync(12);

var hash = bcrypt.hashSync(post['password'], salt);
```

The module used for password hashing is called bcrypt and has been added to the server using the Node.js package manager.

Here is an example of the generated password hash:

```
$2a$12$KoBZ2T72C6NPndOP3PZgJuzCTSlztX/ESPQcW7Sp.GUaIw.z6UsIC
```

If registration has been successful, the RegistrationSuccessFragment is displayed:


If the users taps the Return Home button he is returned back to the main screen where he is able to login.

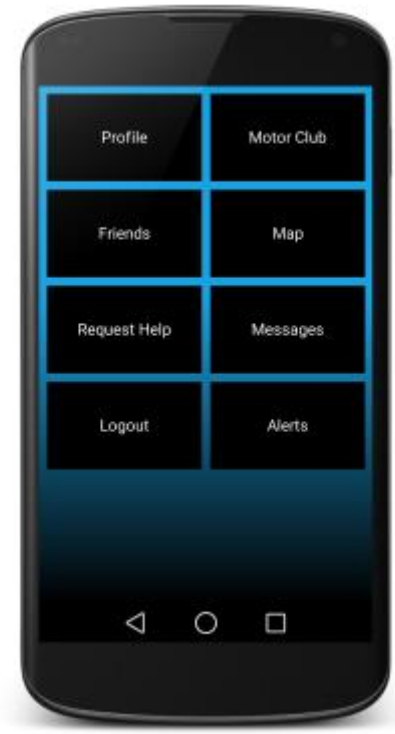
#### 5.2.2.2 User Login

When the user taps the Login button, the Authenticator checks if the user has a saved authentication token. If the user logins for the first time or has previously logged out, he does not have a saved token and has to enter his credentials in the LoginFragment:


The user login process is described in the Appendix 1 section, login diagram.

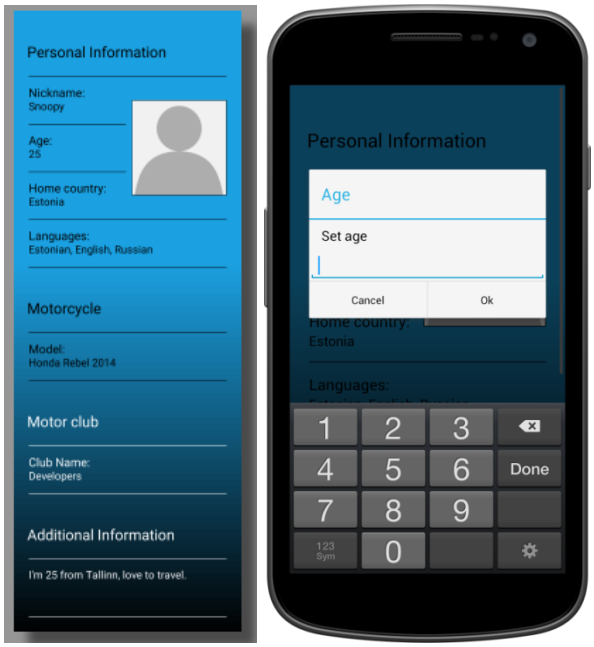
### 5.2.3 Main menu

The main menu is displayed using the SelectionFragment which consists of multiple buttons in a grid. The selection menu allows the user to access his profile, friends list, the map, the motorcycle club page, messages, alerts and to logout:



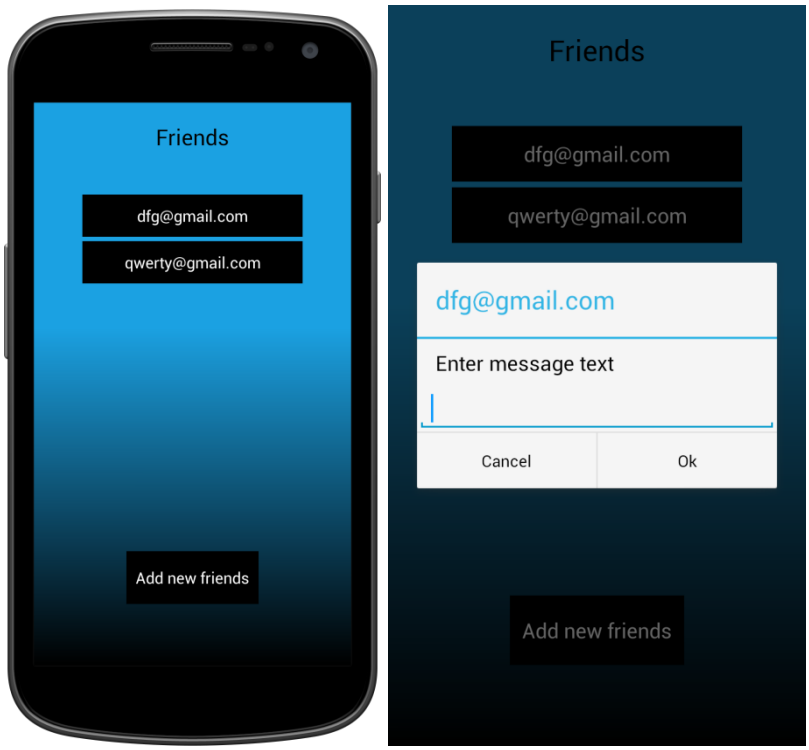
### 5.2.4 Profile menu

The profile page allows the user to edit his personal information, motorcycle model and other information. The page is displayed using the ProfileFragmentConsidering that the profile page contains a lot of information, the window has been designed to be scrollable. Each parameter on the profile page is editable. When the value is tapped an alert dialog is shown which allows to set a new value for the parameter:



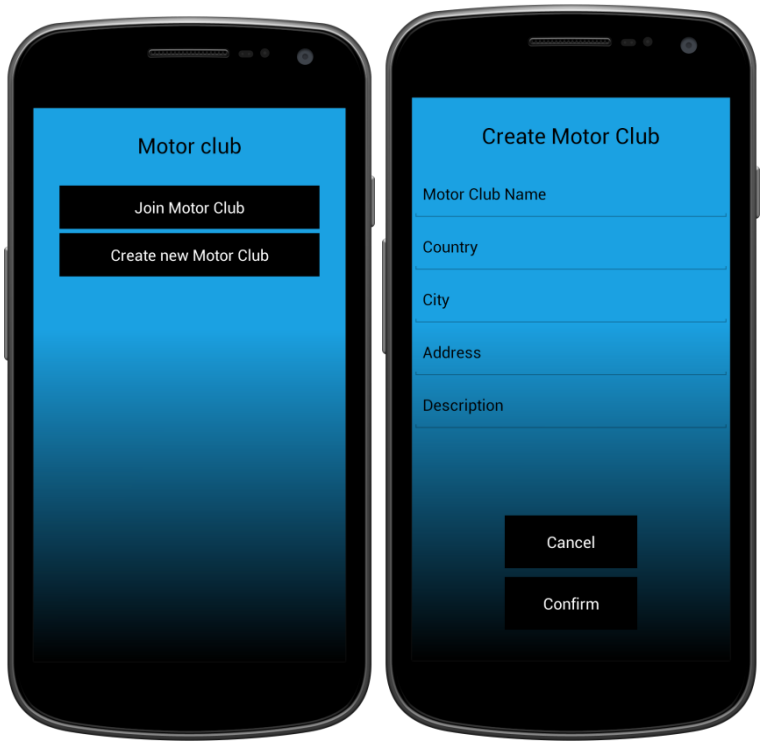

### 5.2.5 Friends list

The friends page is displayed using the FriendsFragment, it contains a list of friends. At the bottom of the page is the button to add new friends to the list. The list consists of clickable buttons. When the button is pressed, an alert dialog shows up which allows the user to send a message:

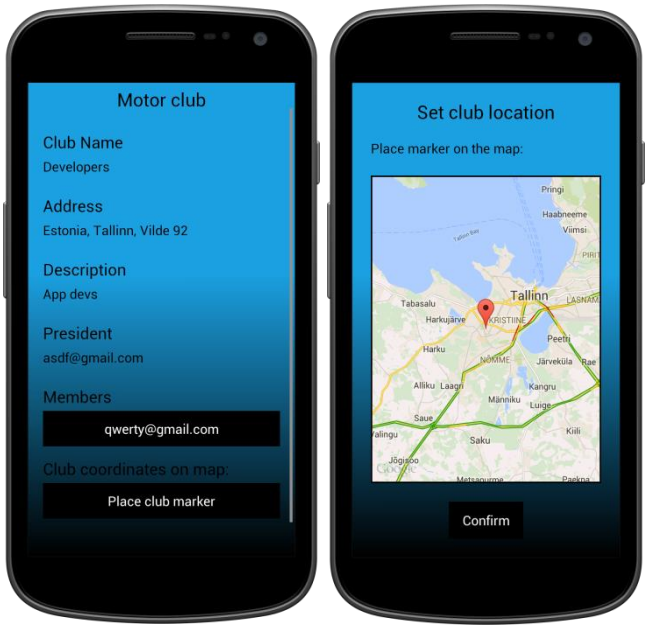



### 5.2.6 Motorcycle club page

If the user is not a member of any motorcycle clubs, he has the option to join a new club or create a new one:

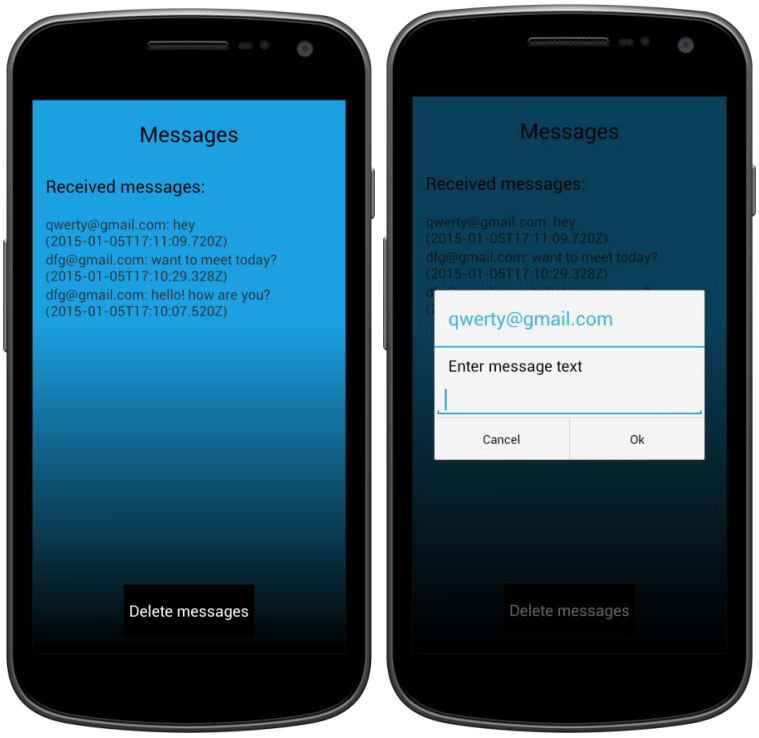


If the user is a president of the club, he has the button to place the club location marker on the map:



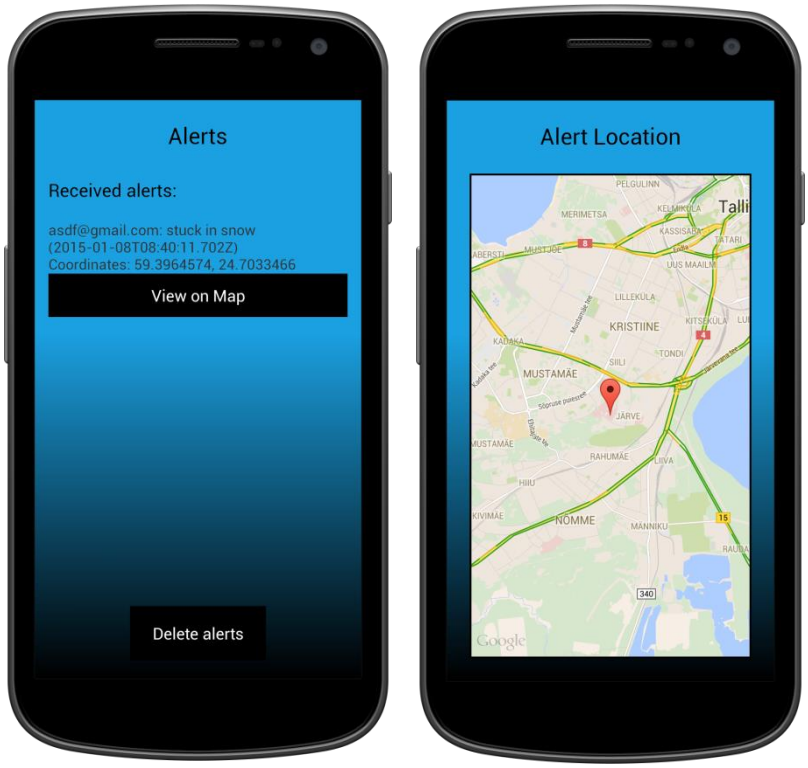

### 5.2.7 Messages page

The user is able to view his message in the MessagesFragment:



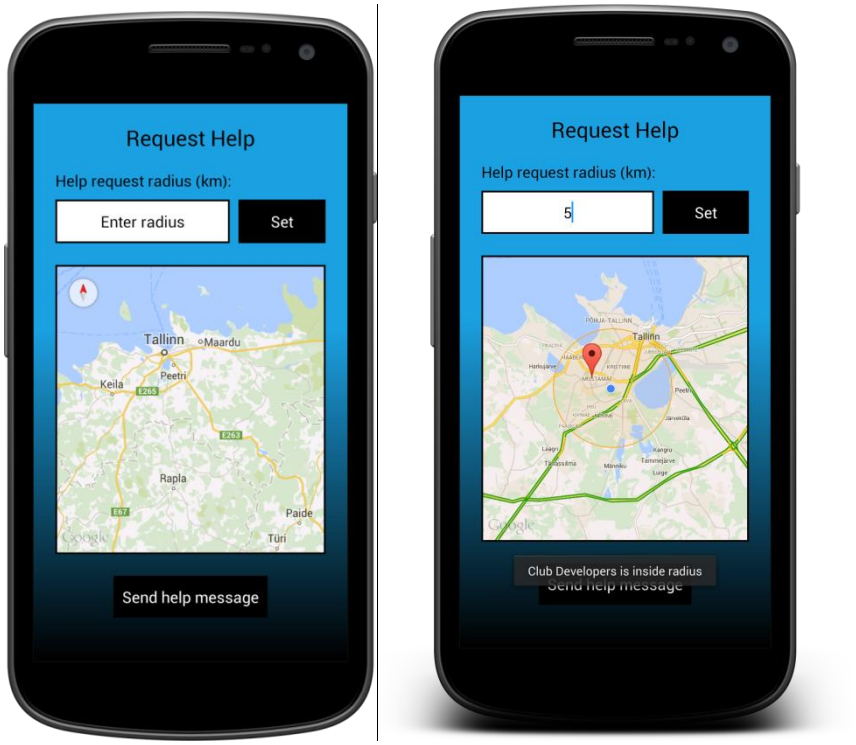
### 5.2.8 Alerts page

All alerts are shown in the AlertsFragment. The alert consists of a message and coordinates. Alerts are sent by users who request roadside assistance. By clicking the view on map button, the user is able to see the coordinates of the alert on the map:



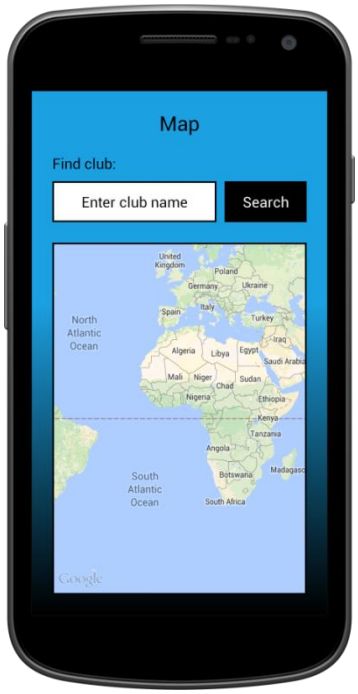

### 5.2.9 Request help page

The user is able to request roadside assistance on the Request help page displayed using the EmergencyFragment. The application automatically finds the user's location and displays it on the map. The user is then able to set the radius which determines which clubs get the help request message:




### 5.2.10 Map page

On the map page, the user can find other clubs by using the search field. If the club is found, the map zooms in on the club's marker position:



### 5.2.11 Logout

When user taps the logout button, his token removed from SharedPreferences and he is returned to the home screen.

## 5.3 Node.js server development

The application code is saved in the server.js file. The application is run using the following command:

```
$ node server.js
```

Additional modules for the plugins such as random token generation have been installed using the Node.js package manager with commands of the following structure:

```
npm install rand-token --save
```

Modules inside of the application have been added using the require() function:

```
var randtoken = require('rand-token');
```

In this project, modules for password hashing, token generation, communication with the PostgreSQL database and HTTP communication have been added.

### 5.3.1 Server side structure

The motorclubsever directory was created which contains all of the server files.

The controller that handles all of the posts from the client is inside of the server.js file. The file contains all of the required functionality to generate tokens, hash passwords, communicate with the PostgreSQL database and send responses to the client.

The additional modules are installed by using the Node.js package manager tool and stored in the node_modules directory.

### 5.3.2 Server functions

All of the functions that handle data posted from the client application are stored in the server.js file. The server typically receives key-value pairs, sends a SQL query to the database, converts the result into JSON format and sends back a response.

## 5.4 PostgreSQL database back-end development

The database has been created using the pgAdmin III tool. A password has been set to connect to the database on port 5432. The following SQL was executed to create the database called 'motoabi' with UTF8 encoding:

```
CREATE DATABASE motoabi

  WITH OWNER = postgres

       ENCODING = 'UTF8'

       TABLESPACE = pg_default

       LC_COLLATE = 'English_United States.1252'

       LC_CTYPE = 'English_United States.1252'

       CONNECTION LIMIT = -1;
```

### 5.4.1 Database structure

The database motoabi has a public Schema which has a total of 7 tables: moto_user, motorclub, motorclub_members, user_alert, user_friend, user_message, user_profile.

### 5.4.2 Database tables and descriptions

Table moto_user holds users' emails, hashed passwords and generated authentication tokens. The motorclub table holds each motorcycle club's information such as club name, address and location coordinates. Motorcycle club's list of members is stored inside of the motorclub_members table. Table user_alerts holds all alert information such as sender and receiver emails, timestamp and alert coordinates. Table user_message has the messages, receiver and sender email addresses. Each user's personal information is stored in the user_profile table. User's list of friends is kept inside of the user_friend table.

# 6. Switching from development to release

A new virtual private server has been rented from DigitalOcean [41] which allows deploying an SSD cloud server for developers in a short amount of time. The server is located in Amsterdam and provides a static IP address for communication between the client application and the server. Ubuntu was installed on the server, SSH access has been configured. Node.js and PostgreSQL for the operating system have been installed. The database from the Windows development laptop has been backed up and restored on the remote server.

After the server and database have been moved to the server in Amsterdam, the mobile application in Android Studio had to be signed with a release key. A new keystore has been generated, the configuration of the project has been updated from debug mode to release mode and the application's apk package has been built. SHA1 has been retrieved from the keystore for the release key and added in the Google Developer Console [42], under Google Maps API Key.

A developer account has been registered on Google Play Market [43], the apk package has been uploaded. Necessary screenshots and icons were added.

The application is currently distributed in Estonia and is available for Alpha Testers.

# 7. Alpha Testing

The project has been published on Google Play Market for Alpha Testing. The application is distributed in Estonia and available to members of a group of testers that has been created on Google Groups.

Alpha Testers were invited to the group and were asked to download and install the application and leave their feedback.

## 7.1 User feedback

The users were asked to leave feedback on the following aspects of the application:

- Do you like the design of the application? What did you like the most? What did you dislike?
- Does the application feel responsive on your device?
- Is the application stable? Did you have any crashes or issues?
- What functionality would you add?
- Which pages in the application do you feel need changing?

The answers have been documented and can be found in Appendix 3, alpha testing feedback.

# 8. Summary

The goal of the project was to analyse and develop a mobile application for members of motorcycle clubs. The market had to be researched and the user's needs and requirements had to be found out. The application was required to provide the functionality to create a motorcycle club, to communicate with people from other clubs, to have a map for navigation and to request for help on the road.

The application has been successfully created and has the required functionality. The user is able edit his profile, add friends, send messages to other users, find clubs on the map, request roadside assistance and create his own motorcycle club and place it on the map. The project was published on Google Play Market and is currently under Alpha Testing.

Certain improvements have already been planned for the future. The user profile has to be redesigned. The author plans to add tabs where the user can switch between personal information, club information and motorcycle information pages. The club pages currently do not have a chat room for the members and the members do not have a posting wall like on Facebook. Custom icons should be added for each club and displayed on the map instead of the default ones. When a user receives an alert, a vibration notification should be sent to the receiver. Other businesses which are convenient for travelling motorcyclists should be added and displayed on the map such as hotels, stores and gas stations.

# Used sources index


1. Maanteeamet. November 2014 aasta statistika. [WWW] http://www.mnt.ee/file.php?26931 (01.12.2014) (Excel document)
2. W3Schools. OS Platforms Statistics and Trends. [WWW] http://www.w3schools.com/browsers/browsers_os.asp (01.12.2014) (web page)
3. W3Schools. Mobile Devices Statistics and Trends. [WWW] http://www.w3schools.com/browsers/browsers_mobile.asp (01.12.2014) (web page)
4. Google Maps. [WWW] https://play.google.com/store/apps/details?id=com.google.android.apps.maps&hl=en (01.12.2014) (web page)
5. MapFactor: GPS Navigation. [WWW] https://play.google.com/store/apps/details?id=com.mapfactor.navigator&hl=en (01.12.2014) (web page)
6. Waze. [WWW] https://www.waze.com/ (01.12.2014) (web page)
7. Pirelli Diablo Superbiker. [WWW] http://www.pirelli.com/diabloapp/ (01.12.2014) (web page)
8. EatSleepRide. [WWW] https://eatsleepride.com/ (01.12.2014) (veebilehekülg)
9. Facebook. [WWW] https://play.google.com/store/apps/details?id=com.facebook.katana&hl=en (01.12.2014) (web page)
10. WhatsApp. [WWW] https://www.whatsapp.com/ (01.12.2014) (web page)
11. AAA Mobile. [WWW] https://play.google.com/store/apps/details?id=com.aaa.android.discounts&hl=en (01.12.2014) (web page)

12. Good Sam Roadside Assistance. [WWW] https://play.google.com/store/apps/details?id=com.allstate.goodsam&hl=en (01.12.2014) (web page)

13. MinuteCast by AccuWeather. [WWW] https://play.google.com/store/apps/details?id=com.skymotion.skymotion&hl=en (01.12.2014) (web page)

14. Yelp. [WWW] https://play.google.com/store/apps/details?id=com.yelp.android&hl=en (01.12.2014) (web page)

15. Xamarin. [WWW] http://xamarin.com/ (01.12.2014) (web page)

16. Qt. [WWW] http://qt-project.org/ (01.12.2014) (web page)

17. Android Studio. [WWW] http://developer.android.com/sdk/index.html (01.12.2014) (web page)

18. W3Techs. Usage of web servers for websites. [WWW] http://w3techs.com/technologies/overview/web_server/all (19.12.2014) (web page)

19. W3Techs. Usage of server-side programming languages for websites. [WWW] http://w3techs.com/technologies/overview/programming_language/all (19.12.2014) (web page)

20. Yii Framework. [WWW] http://www.yiiframework.com/ (19.12.2014) (web page)

21. Laravel. [WWW] http://laravel.com/ (19.12.2014) (web page)

22. Code Igniter. [WWW] http://www.codeigniter.com/ (19.12.2014) (web page)

23. Cake PHP. [WWW] http://cakephp.org/ (19.12.2014) (web page)

24. Wordpress. [WWW] https://wordpress.com/ (19.12.2014) (web page)

25. Drupal. [WWW] https://www.drupal.org/ (19.12.2014) (web page)

26. Joomla. [WWW] http://www.joomla.org/ (19.12.2014) (web page)

27. Node.js. [WWW] http://nodejs.org/ (19.12.2014) (web page)

28. Azat Mardan. PHP vs. Node.js. [WWW] http://webapplog.com/php-vs-node-js/ (19.12.2014) (web page)

29. Computer Language Benchmarks. [WWW] http://benchmarksgame.alioth.debian.org/u32/compare.php?lang=v8&lang2=php (19.12.2014) (web page)

30. Socket.io. [WWW] http://socket.io/ (19.12.2014) (web page)

31. Mohamed A. Mohamed, Obay G. Altrafi, Mohammed O. Ismail. Relational vs. NoSQL Databases: A survey. [WWW] http://www.ijcit.com/archives/volume3/issue3/Paper030320.pdf (19.12.2014) (pdf article)

32. PostgreSQL. [WWW] http://www.postgresql.org/ (19.12.2014) (web page)

33. PostgreSQL About. [WWW] http://www.postgresql.org/about/ (19.12.2014) (web page)

34. Node-postgres. [WWW] https://github.com/brianc/node-postgres (19.12.2014) (web page)

35. Android Getting Started. [WWW] http://developer.android.com/training/index.html (19.12.2014) (web page)

36. Node.js Downloads. [WWW] http://nodejs.org/download/ (19.12.2014) (web page)

37. Bcrypt-nodejs. [WWW] https://www.npmjs.com/package/bcrypt-nodejs (19.12.2014) (web page)

38. Rand-token. [WWW] https://www.npmjs.com/package/rand-token (19.12.2014) (web page)

39. Express.js. [WWW] http://expressjs.com/ (19.12.2014) (web page)

40. PostgreSQL Downloads. [WWW] http://www.postgresql.org/download/ (19.12.2014) (web page).

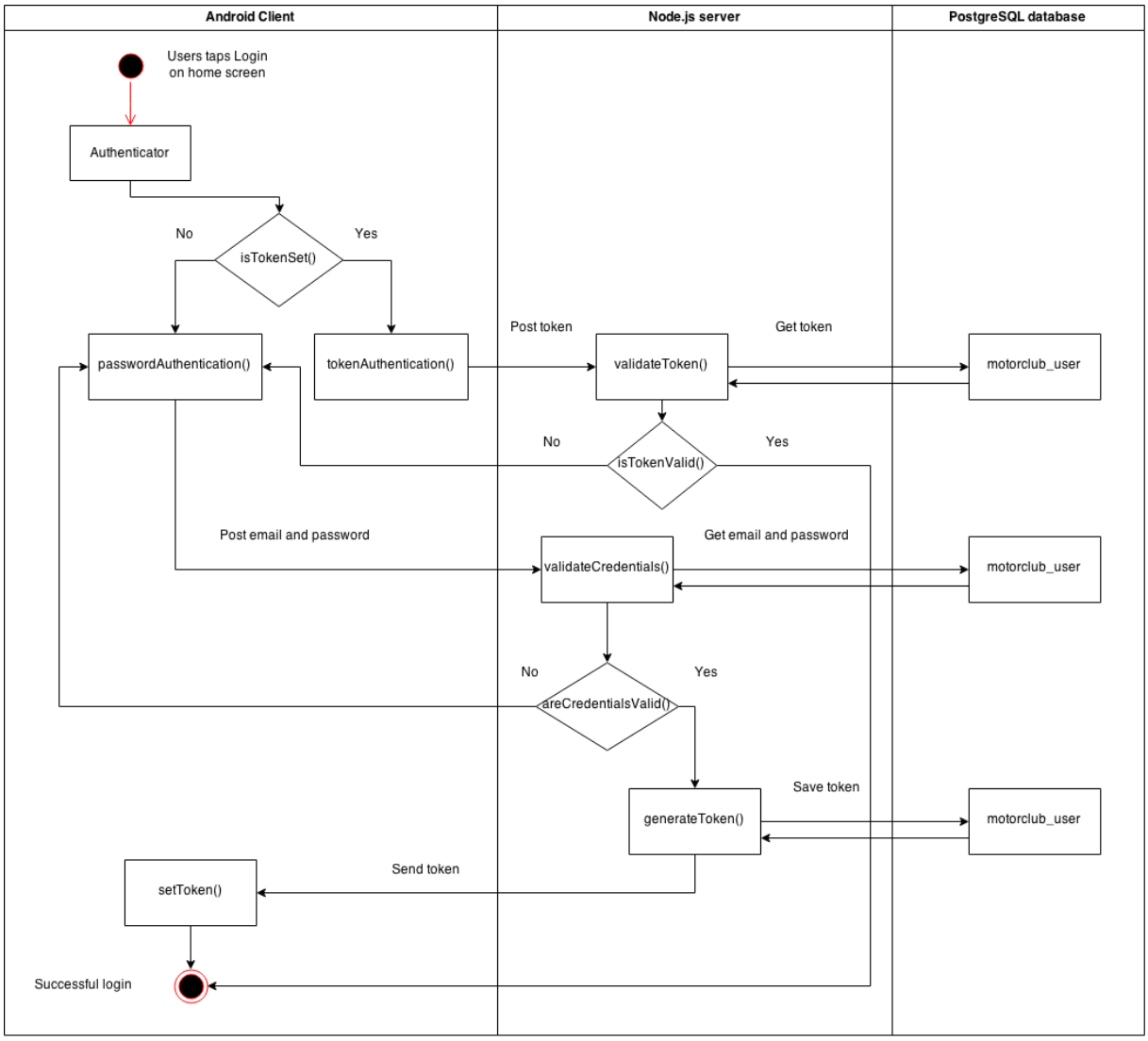
41. DigitalOcean. [WWW] https://www.digitalocean.com/ (19.12.2014) (web page)

42. Google Developer Console. [WWW] https://play.google.com/apps/publish/ (19.12.2014) (web page)

43. Google Play Market. [WWW] https://play.google.com (19.12.2014) (web page)

# Appendix 1

Login diagram




The passwordAuthentication() function is called from the Authenticator class, the credentials are posted to the server, the server checks the login credentials:

```
var check = bcrypt.compareSync(post['password'], result.rows[0].password);
```

If the password is correct, a random 64-char token is generated on the server using the randtoken module. Here is an example of the generated token:

```
gQqSsTts1QIFYTkZ1uoLbJSYX29pMTJzzjSxXAxGakpbRwTTRvVeOEDyG7HhO0Zf
```

The token is then stored into the database and sent to the client in JSON format:

```
[{"token":"
gQqSsTts1QIFYTkZ1uoLbJSYX29pMTJzzjSxXAxGakpbRwTTRvVeOEDyG7HhO0Zf"}]
```

The client application then stores the token into SharedPreferences. The token is stored until the user logs out.

If the user has a token stored in SharedPreferences, when he taps the Login Button, the Authenticator class performs authentication based on the token. The user does not have to enter his email or password. If the token is not valid, the user is sent to the LoginFragment and password based authentication is performed instead. However, if the token is valid, the main menu is displayed.

# Appendix 2

Example Express.js application

Here is an example of a basic Express application:

```
var express = require('express')
var app = express()

app.get('/', function (req, res) {
  res.send('Hello World!')
})

var server = app.listen(3000, function () {

  var host = server.address().address
  var port = server.address().port

  console.log('Example app listening at http://%s:%s', host, port)

})
```

The application starts a server and listens on port 3000 for connection. It will respond with "Hello World!" for requests to the homepage. For every other path, it will respond with a 404 Not Found.

In this project, a similar application structure is being used. The application code is saved in the server.js file. The application is run using the following command:

```
$ node server.js
```

Additional modules for the plugins such as random token generation have been installed using the Node.js package manager with commands of the following structure:

```
npm install rand-token --save
```

Modules inside of the application have been added using the require() function:

```
var randtoken = require('rand-token');
```

# Appendix 3.

Alpha Testing feedback.

| User feedback nr | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| What is the model of your device? | Samsung S3 | Samsung S3 | Google Nexus 7 | LG L70 | Cat B15 |
| Did you have any problems installing or launching the app? | no | no | no | no | no |
| Does the app user interface fit on your device screen? | yes | yes | yes | screen too small for the UI | UI a bit too large |
| Was the app performance slow anyhwere? | no | login felt a bit slow | no | map on first load | opening the map first time |
| Did the app crash anywhere? | no | no | no | no | no |
| Would you change anything in the design? | animate buttons when clicked, change color of alerts button to red when new alerts arrive | did not know you could change profile values on click, not intuitive | messages should be displayed on the left and replies on the right like in other messengers | smaller UI for lower resolution screens | give buttons different background or texture, different profile design |
| What functionality did you like the most? | requesting help, viewing alert locations | finding clubs on map and setting own club location | sending alerts, messages | request help | home screen, login was simple |
| Which page did you like least? Why? | club creation, too many fields to fill | profile not intuitive | friends list, needs more information | did not like messages, should be shown separate for each user | profile, should be split into different categories with tabs |
| What functionality should be done differently? | other users should be displayed on the map when requesting help | show loading screen when waiting for the answer from server | add notification on phone on new alerts | on friends list nickname and picture should be shown, when requesting help others should be able to see your status and chat with you in the same room | joining clubs should be done by sending and accepting an invitiation or with a pin-code |
| What functionality would you add? | tracking users on the map, message of the day for clubs | custom club markers or logos, club chat room, calender for club events | gas stations, hotels on the map, personal message wall which others can see | create travel groups which you can track on map, track travelled routes, distance travelled | find users on the map, open chat room by clicking on them |