TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Juri Gurjev 154875 IAPB

INFOHUNT EVENT SHARING MOBILE APPLICATION DEVELOPMENT

Bachelor's thesis

Supervisor: Gert Kanter
PhD

Tallinn 2020

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Juri Gurjev 154875 IAPB

INFOHUNDIGA LÄBIVIIDUD ÜRITUSTE INFO EDASTAMISEKS JA VAHENDAMISEKS MOBIILIRAKENDUSE LOOMINE

Bakalaureusetöö

Juhendaja: Gert Kanter
PhD

Tallinn 2020

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Juri Gurjev

14.05.2021

Abstract

The purpose of this thesis is to create a hybrid mobile application, that provides quick and easy information sharing about events held by Infohunt organization which is an information portal for youngsters and professionals working with them. The mobile application user should be able to get information about events taking place around user's current location, filter events list according to the user's interests, register for participation, as well as further confirm user's participation in the registered event and receive bonus points that can be exchanged for gifts in the future. For more convenient personalized use and access to full application functionality, users can register an account and log in to gain access to their personal account data and general statistics on visited or registered events.

The main task of this work is the connection of the application with the backend through API requests and data synchronization since the application uses the same server and database as the existing web application.

The goal of the work is an application with previously described functionality for both iOS and Android operating systems and its publication for general use on such services as App Store and Google Play Store.

This thesis is written in English and is 37 pages long, including 6 chapters, 13 figures and 1 tables.

Annotatsioon

Infohundiga läbiviidud ürituste info edastamiseks ja vahendamiseks mobiilirakenduse loomine

Antud lõputöö eesmärgiks on luua hübriidne mobiilirakendus, mis võimaldab Infohunti organisatsiooniga korraldatud ürituste kohta informatsiooni jagamist. Infohunt on üle-eestiline teabeportaal noortele ja nendega töötavatele spetsialistidele. Mobiilirakenduse kasutajatel peaks olema võimalus saada ülevaadet enda ümber toimuvatest sündmustest vastavalt kasutajaga valitud huvide valdkondadele, registreeruda nende ürituste osalemiseks ning lisaks kinnitada oma osalemist ja saada boonuspunkte, mida saab tulevikus kingituste vastu vahetada. Mugavamaks isikupärastatud kasutamiseks ja juurdepääsu rakenduse täielikule funktsionaalsusele saavad kasutajad registreerida konto ja selle kaudu sisse logida, et pääseda juurde oma isikliku konto andmetele ja külastatud või registreeritud sündmuste üldisele statistikale.

Töö põhiülesandeks on rakenduse ühendamine taustaprogrammiga API-päringute kaudu ja sellega andmete sünkimine, kuna rakendus kasutab sama serverit ja andmebaasi kui olemasolev veebirakendus.

Töö tulemuseks on mobiilirakendus, mis vastab tellija funktsionaalsetele nõuetele. Rakendus peab töötama samamoodi iOS ja Android seadmetel ning peab olema avalikustatud üldiseks kasutamiseks sellistel platvormidel nagu App Store ja Google Play Store.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 37 leheküljel, 6 peatükki, 13 joonist, 1 tabelit.

List of abbreviations and terms

API	Application Programming Interface Interface that defines interactions between multiple software applications
CSS	*Cascading Style Sheets*
Expo	Open-source platform for making universal native apps with JavaScript and React
HTTP	*Hypertext Transfer Protocol*
HTML	*HyperText Markup Language*
iOS	Mobile *operating system* created by Apple
JSON	*JavaScript Object Notation* Easy to read data-interchange format
QR	*Quick Response* (Code) Two-dimensional barcode that contains data like URL or another encoded data type
REST	*Representational State Transfer* Software architectural style based on HTTP requests through the APIs
RAM	*Random Access Memory*
URL	*Uniform Resource Locator* Unique address of the resource on the Web

Table of contents

1 Introduction 11
1.1 Problem and background 11
1.2 Goals 11
1.3 Methodology 12
1.4 Thesis overview 13
2 Technology 14
2.1 Drupal backend 14
2.2 Analysis of mobile application types 14
2.2.1 Native applications 15
2.2.2 Web applications 15
2.2.3 Hybrid applications 16
2.3 Comparison of mobile application types 16
2.4 React Native and Expo 18
3 Analysis of existing apps 19
3.1 Bolt Scooters 19
3.2 Eventbrite 20
4 Mobile application development 23
4.1 User authentication and secure data exchange 23
4.2 Translations and language selection 24
4.3 Front end views 24
4.3.1 Login Screen 24
4.3.2 Register Screen 25
4.3.3 Profile Screen 26
4.3.4 Navigation 27
4.3.5 Events Screen 28
4.3.6 Map Directions 29
5 Validation and publication 31
5.1 Usability Testing 31
5.2 The application publication 31

6 Summary 33
References 34
Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis 37

List of figures

Figure 1 Global market share held by the leading smartphone operating systems [8]... 15
Figure 2 Bolt application that shows users location and scooters nearby [13]. 20
Figure 3 Eventbrite application events list (a) and detailed view (b) screens. 21
Figure 4 Selected location block on top of the list (a) and a window to change the location (b)..... 22
Figure 5 Abstract Protocol Flow [5]..... 23
Figure 6 Login screen (a), login screen with validation (b), password recovery screen (c)..... 25
Figure 7 Multistep registration screens: Location choice (a), interests choice (b) and user personal data form (c). 26
Figure 8 User profile screen (a), visited events statistics below main profile area (b) and user personal data update screen (c). 27
Figure 9 Pseudocode that shows how were drawer and stack navigations combined.... 27
Figure 10 Hamburger navigation menu for logged in users (a) and guests (b). 28
Figure 11 Screen with listed events (a) and a Google map screen (b) indicating event's location. 29
Figure 12 Map component with user and event locations connected by the closest connecting path line..... 30
Figure 13 Infohunt application published to the Google Play Store (a) and App Store (b). 32

List of tables

Table 1 Comparison of different mobile application type parameters [6], [7]..... 17

1 Introduction

1.1 Problem and background

Infohunt organization, which is an information portal for youngsters and professionals working with them [1], contracted the company at which the author is employed [2] to develop a mobile application which will allow to spread information about the events that they hold. The mobile application must provide user with the information about events occurring nearby, sort events list according to user's interests, register for participation and confirm user's event participation by scanning the code to receive bonus points that can be exchanged for gifts in the future. The mobile application must be developed for both Android and iOS platforms and translated into Estonian and English to ensure application accessibility for most users.

According to the client's specification the mobile application must use the same event and user databases as the existing web application that uses Drupal [3] for the backend and therefore should communicate with the backend by using API requests.

1.2 Goals

The main goal of the work is to develop and publish a hybrid mobile application that meets the following functional requirements:

1. The user should be able to create an account as well as log into the application using it. The profile view should provide the ability to customize information associated with a user account such as personal information, profile photo, interests and location. Additionally, this view should contain statistics of events visited by the user.
2. Depending on the permitted / not permitted processing of geolocation data in the application, the user should see a list of events at a distance of 50 km from his current location or select the location manually.

3. In the case of non-virtual events, it should be possible to see the location at which they will take place on the map. Virtual events should instead contain a link to the event-related web page.
4. The application must be available in both Estonian and English languages between which the user can easily switch.

Additional requirements that were not developed by the author but by other member of the development team:

1. The user should be able to register for the desired event and later confirm their participation by scanning the QR code, as well as leave feedback about the event.
2. For a successfully validated visit to the event, the user receives bonus points that can later be used to choose a gift.

1.3 Methodology

In order to fulfill all the goals, initially the author will have to study modern technologies and methods of developing mobile applications and choose the most appropriate one. While making the technology choice, it must be considered that the application should be developed with the same functionality for both iOS and Android platforms, as well as make the development process as cheap and fast as possible to meet the client's budget and deadlines.

Additionally, the author will analyse applications that are similar in functionality to get some ideas that could help making better decisions in further development and simplify its process.

Since the application uses an already existing backend server there should be configured backend API URLs which can be accessed from the application to exchange data with it.

For all sorts of additional functionality such as translations or a view with the location of the event on the map, third-party libraries may be used.

1.4 Thesis overview

The first part of the thesis covers technologies that were used for developing the mobile application and comparing them to possible alternatives.

The second part of the thesis provides an analysis of applications with similar functionality and practises that are worth taking into consideration for achieving set up goals.

The third part of the thesis covers main development processes and ideas behind them that gives a general overview about how core functionality was achieved.

The last part of the thesis describes how the application was tested, validated and published for the public use.

2 Technology

2.1 Drupal backend

Since one of the conditions is that the application must use the same data as the existing web application, the author is not able to use another technology for the server and must work with the existing one. Fortunately, Drupal is great for this kind of REST architecture by exchanging data via API requests and has enough tools to implement this functionality [4], as well as the ability to extend it using a library of additional plugins.

Drupal also contains an OAuth 2.0 module for authentication that allows making protected API requests in order to securely exchange information and resources between the mobile application and the server using the generated user access token [4], [5].

2.2 Analysis of mobile application types

Mobile application development is divided into three categories, depending on what functionality is needed to achieve the set goals, as well as the available amount of resources, budget and time for development [6], [7]. The types of mobile apps -

1. **Web applications**: Applications that run via a web browser on the mobile device and do not need to be downloaded [6].
2. **Native applications**: Applications that are native to one platform – Android or iOS. This type of applications can truly utilize all the possibilities provided by the operating system of choice [6].
3. **Hybrid applications**: A combination of web and native application features that has multi-platform support [6].

While keeping development costs low it is very important to reach as many users of the target audience as possible and despite the fact that the average part of Android devices (as of 2018) is approximately equal to 88% shown in Figure 1 [8], the second most popular operating system – iOS, can not be ignored.

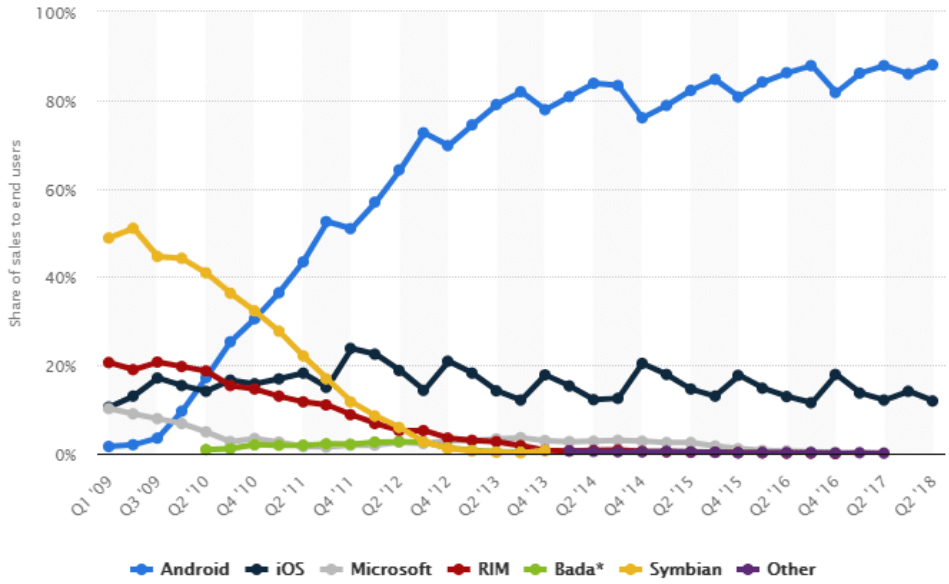


Figure 1 Global market share held by the leading smartphone operating systems [8]

Since it is very important to choose the optimal method of developing an application from the very beginning, the author is going to view each one in detail and compare them with each other.

2.2.1 Native applications

Native applications are designed and developed for a particular operating system therefore can only work or function on the devices for which they are built [6]. Often such applications are developed using a programming language specific to this system, which makes development more expensive and is one of the reasons why applications should be created separately for each platform [7].

The main advantage of native applications is that they have better performance and security compared to other types of applications while also can take full advantage of the hardware and software capabilities of the device for which they were developed, since there are no abstraction levels that grants easy access to all the hardware features and the code is as short and efficient as possible [7].

2.2.2 Web applications

Web applications are built using technologies such as JavaScript, CSS and HTML5 which is usually much quicker, easier, and cheaper to build compared to native

applications. Unlike native applications web-based applications are accessible via the internet browser and do not need to be downloaded or installed [9]. The good part about web applications is that they do not take much of the device's resources, such as RAM and storage but on the other hand they also do not have access to the device's hardware functionality and always require internet connection to work properly [6], [9].

2.2.3 Hybrid applications

Hybrid applications are sometimes referred to as cross-platform mobile applications and it was achieved by combining both native and web application features. The user can download and install them just like usual native applications but on the other hand, these type applications use a browser embedded inside the app itself to render the data [6].

Most parts of the app are written in a common programming language that gets converted into a native form to run on each operating system which also means that developers have access to the native hardware and software features of the device [7]. The fact that multi-platform applications can be built on a single code base using some popular common languages like JavaScript means that such development takes much less time and resources compared to native applications [6].

Hybrid applications lack a little bit of performance and may have some features missing on certain devices as well as design issues that do not support both devices as hybrid applications are developed for multiple platforms and are required to adjust to them simultaneously [6].

There are several platforms that allow for cross-platform development including Xamarin, React Native, Titanium, Flutter and Ionic. One of the good examples is Instagram that is a popular photo-sharing application built with React Native showing that the amount of code shared between iOS and Android devices was over 90% [8].

2.3 Comparison of mobile application types

As a result of the analysis of various types of mobile applications, the author compiled a table (Table 1) comparing the main parameters which are important for choosing the most appropriate method to fulfill objectives of this work.

Parameters	Native	Web	Hybrid
Performance	Fast and smooth No abstraction Efficient coding with access to native features	Slower speed Dependant on internet connection Do not run well on older browsers	Slightly slower speed compared to native Do not run well on older devices
Hardware features access	Easy access to all native features through the APIs	Can not access the capabilities of a device	Can access the native features but some may be missing on certain devices
Development Cost	Native developers are more expensive Separate applications may use different programming languages	Affordable development costs as customizing to a specific platform is not needed Easy distribution and instant updates	Easier to develop as it partially uses standard web technologies and has single code base
Development Time	To cover multiple platforms will have to develop and maintain separate applications	Quick and easy as it does not require knowledge about different operating systems specifics	Some features may take time to work on several platforms simultaneously Single code base saves time compared to native
Multiple platform	Separate application for each platform	All devices with browser and internet connection	Multi-platform support

Table 1 Comparison of different mobile application type parameters [6], [7].

Based on this table, the author concluded that the most appropriate method is to develop a hybrid application, since according to the set goals application requires access to the user's camera and geolocation that can not be achieved with web applications. Hybrid type of application also provides access from devices with various operating systems and single code base keeps it relatively simple, fast, and cheap to develop and maintain which satisfies all the requirements.

2.4 React Native and Expo

From several platforms for the development of hybrid applications, the author chose React Native as he already had experience in developing web applications using React and is partially familiar with the architecture and the main principles of development. It is also using the well-known JavaScript programming language which helps to learn mobile specific features quick and easy so that even developers without previous mobile development experience could maintain it in the future.

React Native was released and is maintained by Facebook since 2015 and has a large number of contributors for GitHub repositories [10] as well as big community and public resources which provide a large variety of learning materials and libraries.

To set up a React Native project, run and debug an application during development it is considered a good practice to use Expo [11]. Expo is a framework and a platform for hybrid React applications. It is a set of tools and services built around React Native and native platforms that helps to develop, build, deploy, and quickly iterate on iOS, Android, and web apps from the same JavaScript/TypeScript codebase [12].

3 Analysis of existing apps

The author conducted an analysis of applications with similar functionality to find ideas and practices that will help to find better solutions during development of the application as well as provide better user experience.

The author will consider following applications:

- Bolt, featuring the process of renting electric scooters.
- Eventbrite, focusing on the user experience.

3.1 Bolt Scooters

One of the problems that had to be solved was the confirmation of the user's participation in the event. since for participation, users receive bonus points that can be exchanged for prizes, it is necessary to come up with a system that the users cannot easily abuse and, for example, scan a code from a photo without being present at the event in person, or so that one user can confirm participation by logging in using several accounts.

The author was inspired by Bolt application that offers electrical scooters renting [13] and supposed that since the application constantly monitors the user's geolocation (Figure 2), it considers that the code can only be scanned while being directly next to the scooter. And in order that several users could not simultaneously occupy the same scooter, the codes are unique.

The author concluded that it is possible to track the user’s location in a similar way in the background while the user is using the application and allow the user to confirm the participation only if the used mobile device is near the event. In order to make sure that the user can confirm each participation only once per device, a unique installation number that is provided by Expo application metadata can be used [14].

The technical solution to the problem of secure user participation confirmation was proposed by the author to the client and subsequently approved, however, the development of this logic was carried out by another developer.

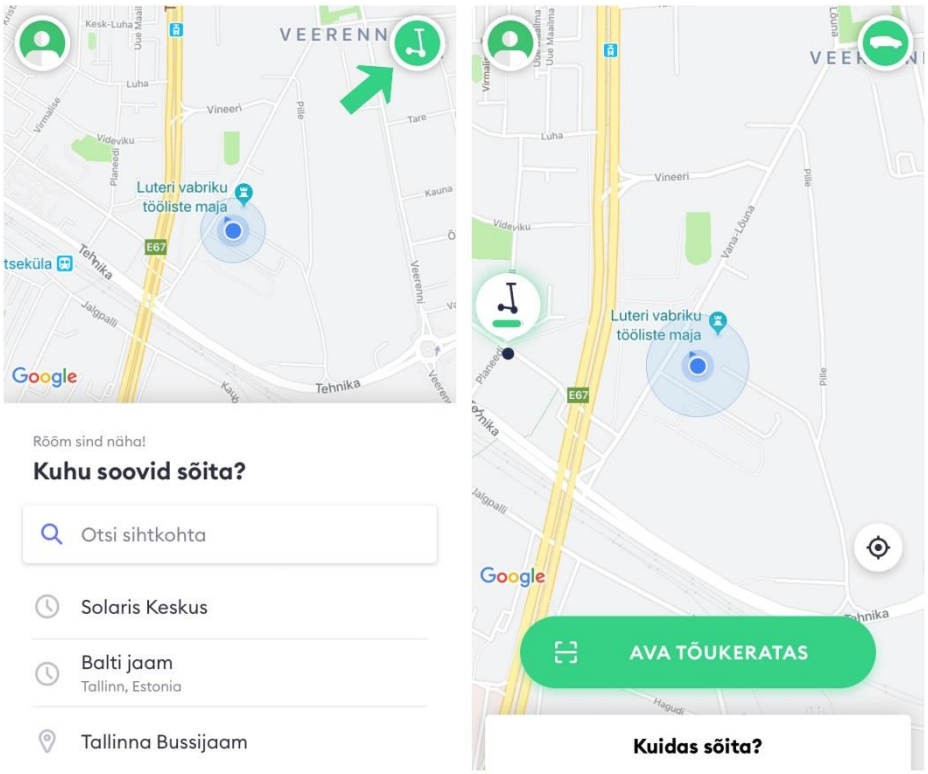


Figure 2 Bolt application that shows users location and scooters nearby [13].

3.2 Eventbrite

Another reviewed application was Eventbrite [15], which shows a list of events that are taking place next to the user's location. As this application has very similar theme with infohunt's application, it has been a great reference in terms of the development of the user interface, the display of all the necessary event information, as well as the general user experience.

Eventbrite does an excellent job of showing as much information as possible in the list of events (Figure 3 a), but our client insisted that there should be as many rows as possible on the screen without scrolling therefore only the most important information may be displayed.

A detailed view of the event attracts the user's attention with a large colorful illustration, gives a detailed description and information about the time and place of the event, as well as a button for registering participation in the event (Figure 3 b).

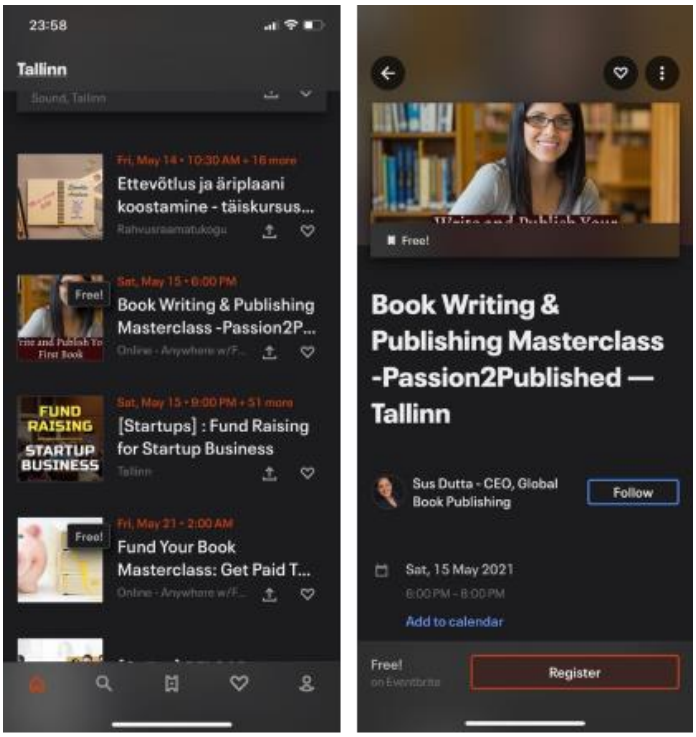


Figure 3 Eventbrite application events list (a) and detailed view (b) screens.

The lower bar with menu items is very convenient for navigation (Figure 3 a), since it is very easy for the users to reach with their finger while holding the device in one hand, however, since in our case the menu will contain more than 5 items, it has been decided to use the hamburger menu instead.

Everbright has an excellent solution to show the user for which particular location the events are shown in the list at the moment (Figure 4 a). If the user clicks on the name of the current location, a window pops up that allows user to specify a different location from the proposed list (Figure 4 b) or use the current user's geolocation to show the events nearby.

Unfortunately, this kind of development was not included in the client's scope and the deadlines did not allow for additional improvements, there is no convenient way to see the current selected user location in the recent version of Infohunt's application, however this may change in the future as it provides a great boost in terms of the user experience.

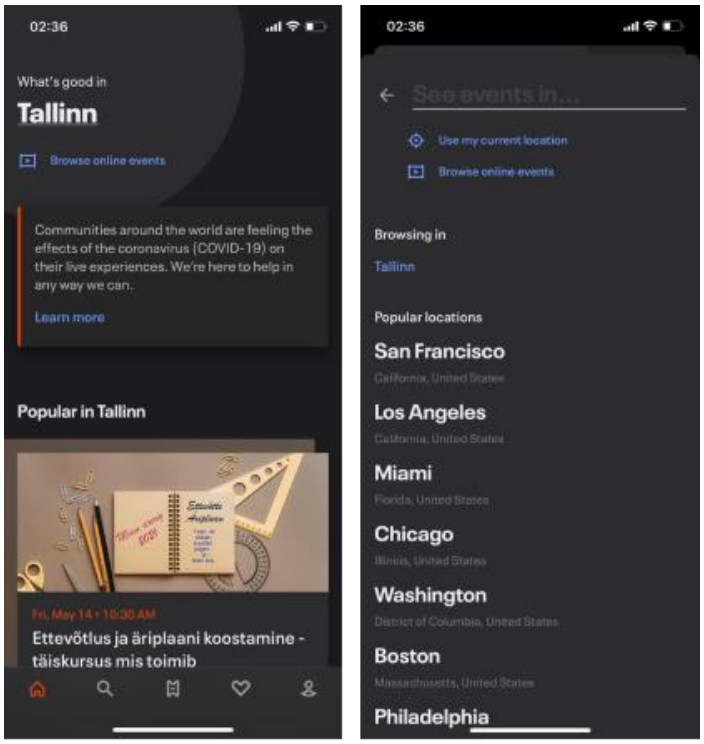


Figure 4 Selected location block on top of the list (a) and a window to change the location (b).

4 Mobile application development

This thesis presents and discusses in detail the screens and functionality directly developed by the author of the work. To exchange data between the application and the Drupal server, rest-api is used to which the application makes requests using the *axios* library. Axios is a set of tools with convenient and clean code that helps to handle HTTP requests and automatically transforms responses into JSON format [16].

The application can be used as a guest, but in this case, you can only get information about events taking place nearby without the possibility of registration. To register, confirm participation, receive gifts and personalized convenient use of the application, the user should create an account and log in.

4.1 User authentication and secure data exchange

When the user logs in by entering his nickname and password, an authentication request is made. If successful, another request occurs that generates a session token, which will be further required for the secure exchange of information and resources associated with the user between the application and the server (Figure 5). If the server successfully generates access token, it is saved to the application's local storage for further use.

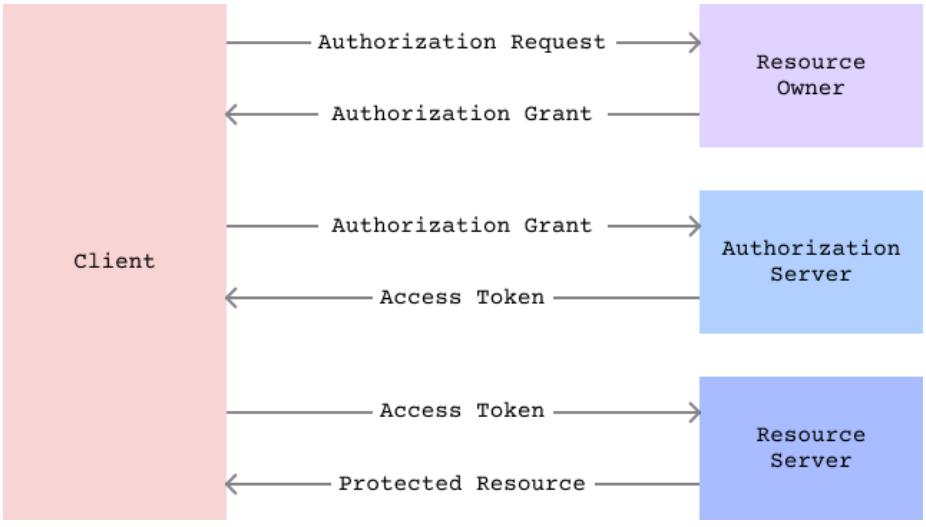


Figure 5 Abstract Protocol Flow [5].

To store the data (including the user access token) locally on the device, the Expo module called *SecureStore* is used [17]. This is a persistent storage which means that it holds data even if the application is closed, which means if the user reopens the application and his access token is still valid, he will be automatically logged in.

4.2 Translations and language selection

All application texts visible to the user were changed to aliases and translated using the i18n-js library [18]. It provides a very convenient way to store and group all strings in Json format, which makes it easy to scale and, if necessary, to translate the whole application relatively quickly into another language. When the application is first launched, the expo localization module [19] determines the default language of the user's device, if the language is Estonian, the application language automatically changes to Estonian, in any other case, English is selected. The next time the user changes the language, selected value is saved to the local storage and will be used with priority on subsequent launches of the application. When changing the language, the application will warn the user that the application will be restarted for the change to take effect. This is necessary due to the peculiarities of how the react application render methods work. All data is fetched only once when every screen component is loaded and the render function called out. To re-request data in another language, component’s render functions must be called again and the easiest way to do it would be to rerun the application.

4.3 Front end views

4.3.1 Login Screen

The login screen consists of a form with fields for the user's nickname and password (Figure 6 a, b) with appropriate validation to make sure that the user has entered the data and to prevent unnecessary requests to the server. The screen also contains a link to the password recovery screen (Figure 6 c), where the user can recover the password by entering the username or email, and a button to begin the registration process (Figure 7) for new account creation.

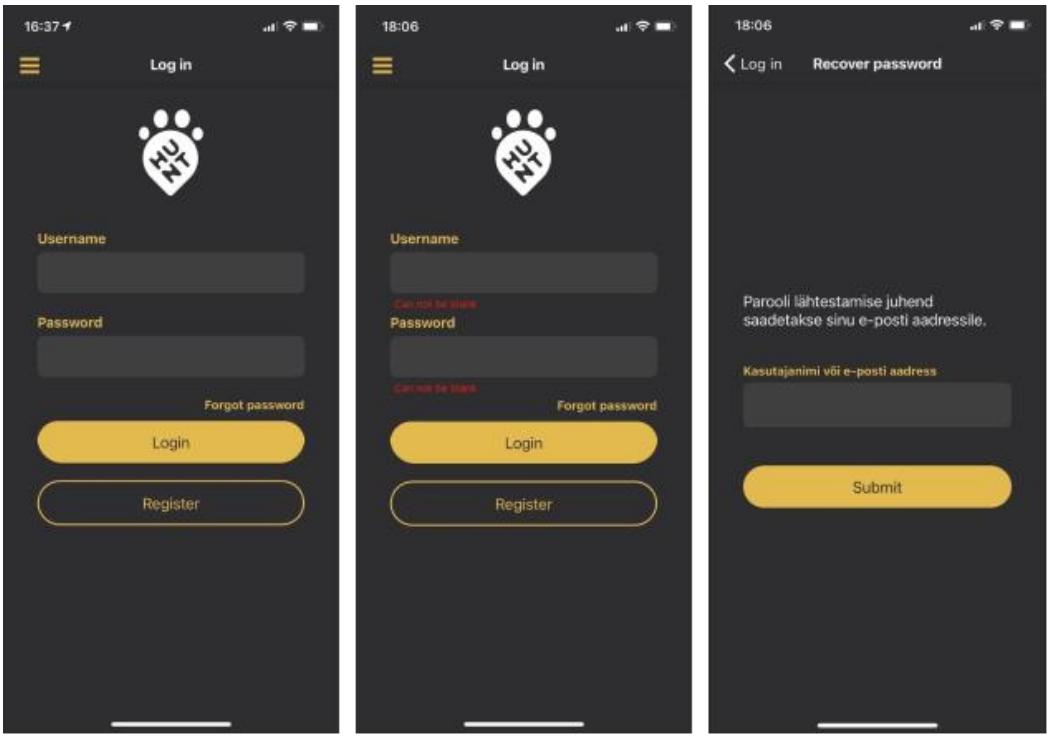


Figure 6 Login screen (a), login screen with validation (b), password recovery screen (c).

4.3.2 Register Screen

The process of registering a new user account consists of three steps:

1. From the proposed list, the user selects the place that is closest to him or in the region from where he would like to receive information about the events (Figure 7 a).

2. The user selects suggested topics of interest for events (Figure 7 b).

3. Finally, the user fills out a form with personal data (Figure 7 c) and confirms the registration by agreeing with terms and conditions [20] as well as privacy policy [21].

After validating the registration fields, whether they are filled in and are in a correct format, all the data is packed together and an API request for registration is sent to the server. If the server responds that all the data is correct and a new account was successfully created, the user automatically logs in and gets redirected to the profile screen (Figure 8).

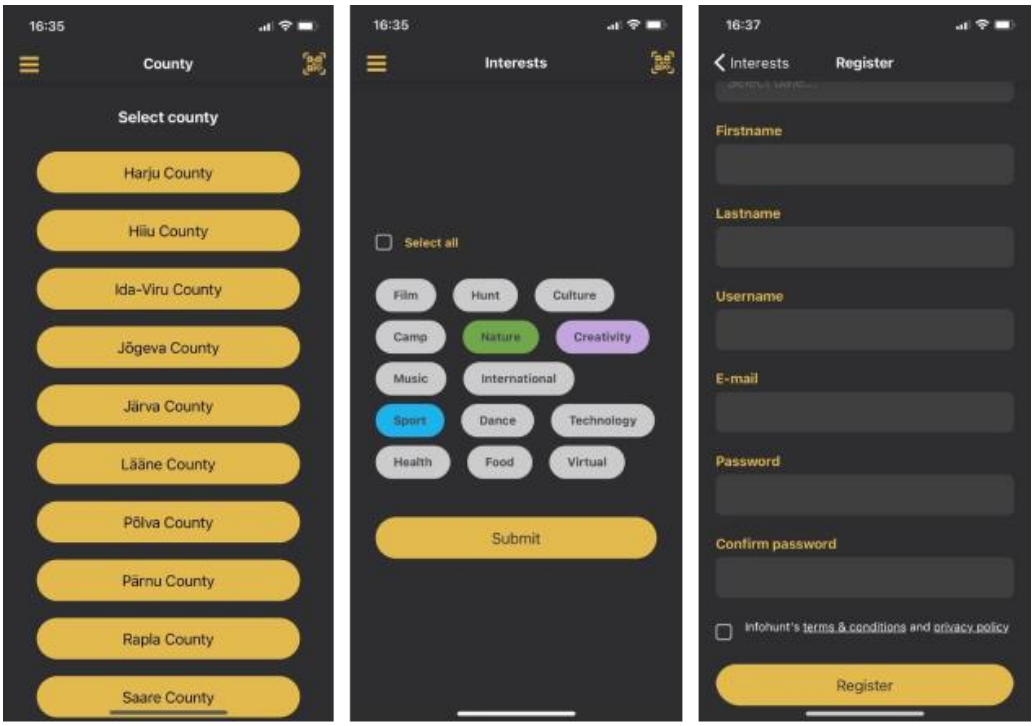


Figure 7 Multistep registration screens: Location choice (a), interests choice (b) and user personal data form (c).

4.3.3 Profile Screen

The profile screen displays the user's personal information, the ability to set or change profile picture, the number of bonus points for attended events (Figure 8 a), as well as general statistics of visited places and the ratio of visited events by interests (Figure 8 b).

The user can also change the personal data associated with his account by going to the data change screen (Figure 8 c) by clicking on the corresponding button. To change the interests and location associated with the profile, the user can follow links in the hamburger menu (Figure 10). The screens for changing user location and interests are identical to those that were used at the registration stage (Figure 7 a, b).

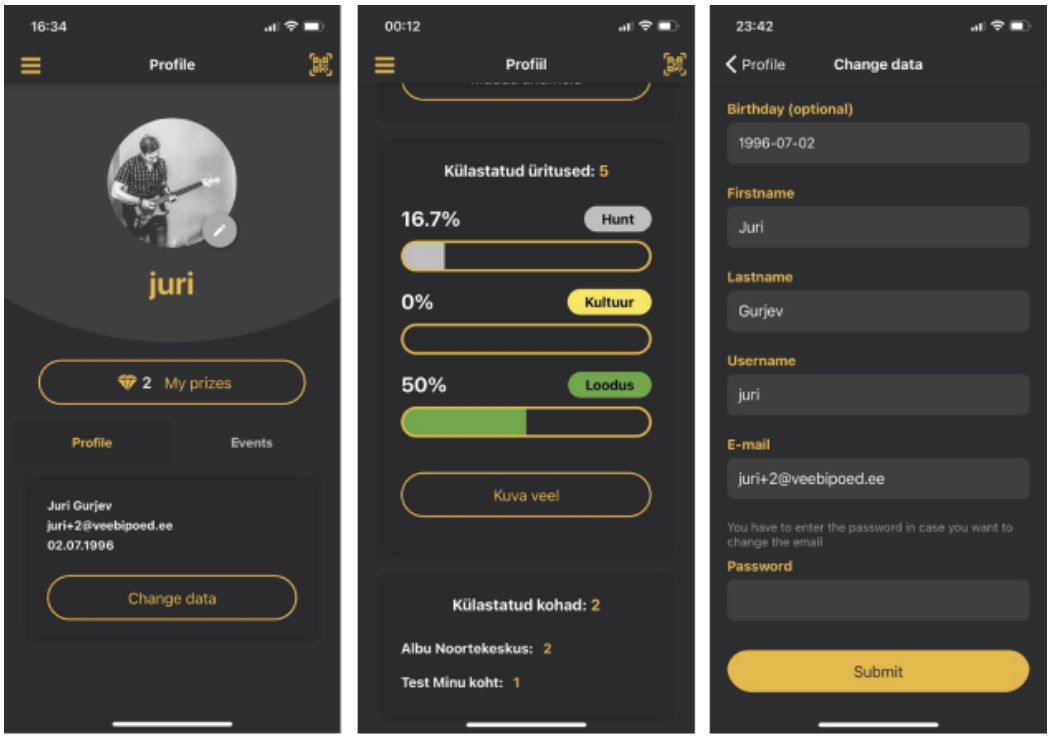


Figure 8 User profile screen (a), visited events statistics below main profile area (b) and user personal data update screen (c).

4.3.4 Navigation

For convenient movement between the screens of the application, the links located in the hamburger menu are used. Menu also provides quick access to the application language change block and login or logout buttons (Figure 10).

```
<Drawer.Navigator>
  <Drawer.Screen name="Events">
    <StackNavigation name="Events">
      <Stack.Screen name="GoogleMap"/>
      <Stack.Screen name="EventDetailView"/>
      ...
    </StackNavigation>
  <Drawer.Screen>

  <Drawer.Screen/>
  <Drawer.Screen/>
</Drawer.Navigator>
```

Figure 9 Pseudocode that shows how were drawer and stack navigations combined.

For routing and navigation between screens, as well as the hamburger menu, a third-party react navigation library was used [22]. Since, by default, the drawer navigation does not provide a screen header that shows current screen name and does not provide a link to quickly return to the previous screen, the author had to combine two types of navigation (drawer and stack) in order to use the functionality of both at the same time (Figure 9).

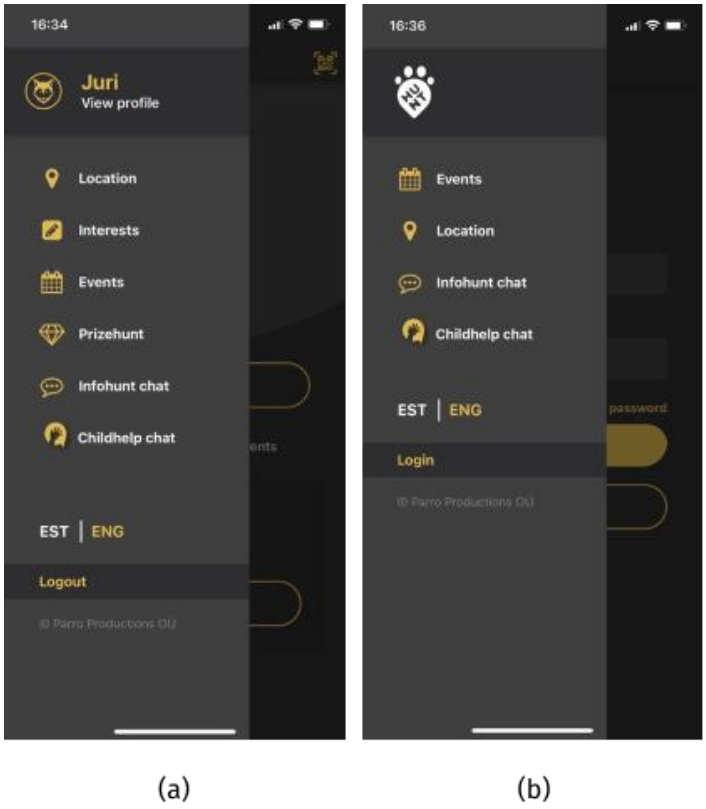


Figure 10 Hamburger navigation menu for logged in users (a) and guests (b).

4.3.5 Events Screen

On the events screen there is a list of events sorted by the date those are held on (Figure 11 a). The user can additionally filter the list by interests or, if he is already logged in, use the interests specified during account registration. In case of virtual events, corresponding button will transfer the user to the mobile browser using the link associated with this event, but if the event occurs in some specific physical place, the user will be transferred to the map screen (Figure 11 b) where user can see his own location and the point where the event takes place.

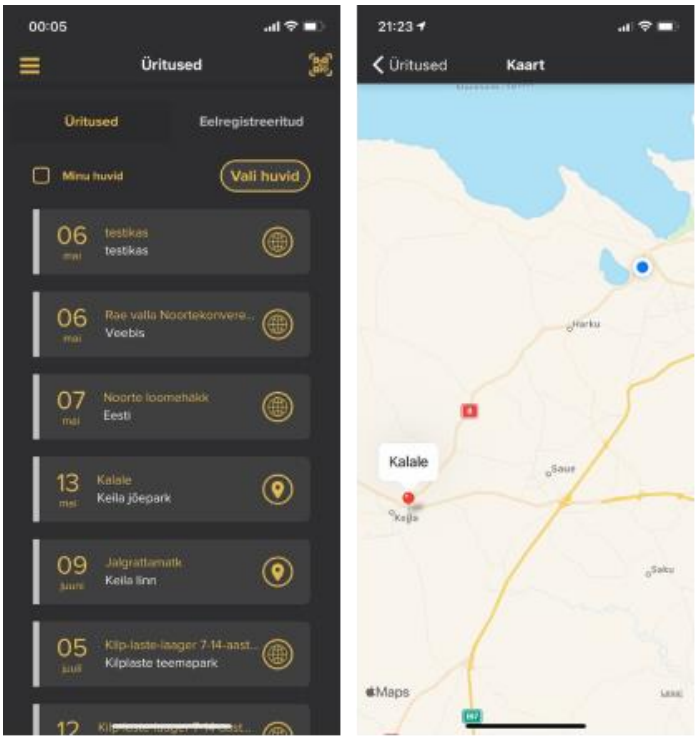


Figure 11 Screen with listed events (a) and a Google map screen (b) indicating event's location.

The map including the user's and event's geolocation markings were developed using the third-party library called *React Native Maps* [23] which is based on the Google Maps API [24]. To dynamically track the user's location and pass it to the map component an additional *Expo Location* module was used [25].

4.3.6 Map Directions

The author has developed additional functionality for the map, which is that the map shows not only the user's current location and the point where the event is held, but also a line that indicates the shortest path between these two points (Figure 12). For this development, the author used React Native third-party library called *Directions Component* [26] and integrated it with the existing map component using dynamically changing user geolocation data collected by the application.

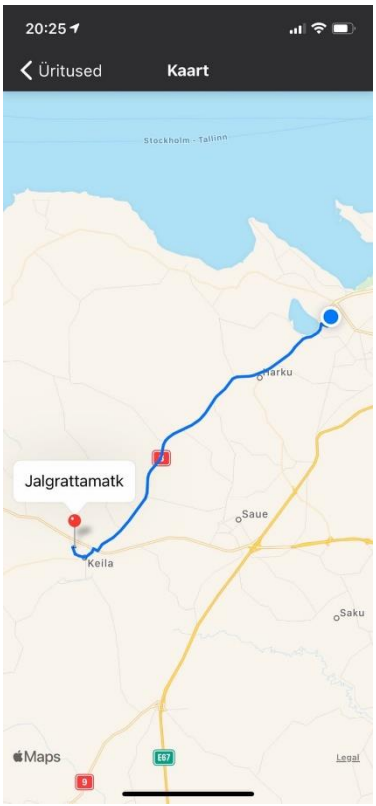


Figure 12 Map component with user and event locations connected by the closest connecting path line.

However, the previously described closest path functionality does not work in the latest version of the application, which was published for public use. There were difficulties with the fact that the Google Directions API key [27], required for directions library to work properly, is restricted by the link of the web application for security purposes and the author had no time left to figure out how to use the restricted key within the hybrid application.

5 Validation and publication

5.1 Usability Testing

During the development cycle, the application was sent to the client at least twice for review, where the target group of users, mainly consisting of teenagers, tested the main functions of the application and gave a report on the found bugs or proposed some user experience improvements. Based on the feedback received, a series of bug fixes and improvements were developed.

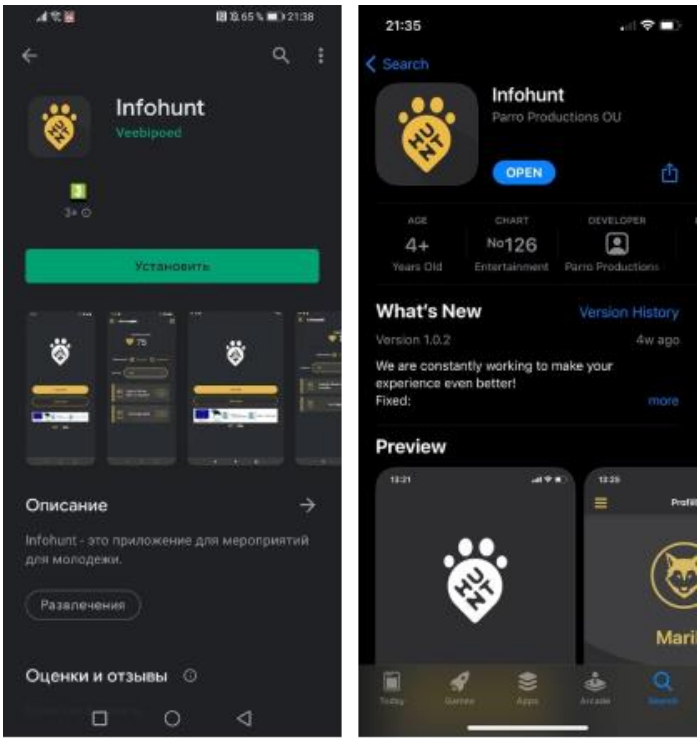
This kind of testing was carried out several times during the development period and proved to be effective to detect major problems early enough in development, making it easier and cheaper to fix them. As a result, when all the errors found during testing were corrected, the main functionality was finalized and validated, the client confirmed that he was satisfied with the solution and confirmed the possibility of uploading the application for public use on the App Store and Google Play Store services.

5.2 The application publication

The application publication to the iOS App Store and Google Play Store based on the code developed using the *Expo* environment, required to generate an apk (Android) or ipa (iOS) files and upload them on the corresponding platform for further testing and submission process [28].

Both platforms initially rejected the application, in both cases due to the conditions of user personal data usage. In case of Google Play Store, the application was rejected because it tracked the user's geolocation on the background although it was completely unnecessary. The solution was simply to limit the collected personal information to the amount that is required for core functionality. In the case of App Store, the problem was that the user was not promptly informed about the use of his geolocation and the application missed some metadata to explain why the user's geolocation is being collected and in which way it is processed. The problem was solved by adding a notification on the very first screen when starting the application and adding comments to the configuration files explaining for what purposes the collected user's personal information is used.

After all the requirements above were fulfilled, the application was successfully confirmed and added for public use on the Google Play Store (Figure 13 a) and App Store services (Figure 13 b).



(a) (b)

Figure 13 Infohunt application published to the Google Play Store (a) and App Store (b).

6 Summary

The purpose of the work is to develop a mobile application for sharing information about events held by Infohunt organization. The application must be available on both iOS and Android devices and published on App Store and Google Play Store services for public use.

The author studied the available technologies for developing mobile applications, compared them with each other and chose a hybrid development method since it is optimal, especially if it is important that the application works simultaneously on both platforms. The author reviewed some existing applications with a similar theme and received some ideas for solving functional problems and some improvements to the user interface.

As a result, the author has developed a mobile application that meets the list of client's functional requirements and put it together with the developments carried out by another member of the development team to prepare the application for publication.

During development, the application was tested several times by users from the target audience and, as a result, was validated by the client who confirmed that all the conditions were met and was satisfied with how the application fulfills its purpose.

As a result, the goal of the work was achieved, and the application was successfully published on the App Store and Google Play Store services.

After the initial development, the client addressed the company at which the author is employed with the desire to make small changes and improvements, as well as the desire to implement the ability to login using social networks such as Google and Facebook, therefore the author periodically continues to supplement and maintain the application.

References

[1] “Infohunt homepage”, [WWW]. Available: https://infohunt.ee/et (29.04.2021)

[2] “Veebipoed Homepage”, [WWW]. Available: https://veebipoed.ee/ (14.05.2021)

[3] “Drupal – Open Source CMS”, [WWW]. Available: https://www.drupal.org/ (01.05.2021)

[4] “REST APIs in Drupal”, [WWW]. Available: https://opensenselabs.com/blog/articles/rest-apis-drupal (01.05.2021)

[5] “The OAuth 2.0 Authorization Framework”, [WWW]. Available: https://tools.ietf.org/html/rfc6749 (01.05.2021)

[6] “What Are The Different Types of Mobile Apps?”, [WWW]. Available: https://www.spaceo.ca/types-of-mobile-apps/ (01.05.2021)

[7] “Comparing Different Types of Mobile Appliaction Development”, [WWW]. Available: https://blog.vsoftconsulting.com/blog/comparing-different-types-of-mobile-application-development (01.05.2021)

[8] “Native, Web or Hybrid Apps?”, [WWW]. Available: https://www.mobiloud.com/blog/native-web-or-hybrid-apps (01.05.2021)

[9] “Mobile Apps vs. Web Apps”, [WWW]. Available: https://careerfoundry.com/en/blog/web-development/what-is-the-difference-between-a-mobile-app-and-a-web-app/#:~:text=Native%20mobile%20apps%20are%20built,run%20on%20the%20device%20itself (03.05.2021)

[10] “React Native Homepage”, [WWW]. Available: https://reactnative.dev/ (04.05.2021)

[11] “Stack Overflow: Difference between Expo and React Native”, [WWW]. Available: https://stackoverflow.com/questions/39170622/what-is-the-difference-between-expo-and-react-native (04.05.2021)

[12] “Expo Documentation – Quick Start”, [WWW]. Available: https://docs.expo.io/ (04.05.2021)

[13] “How to use Bolt Scooters?”, [WWW]. Available: https://blog.bolt.eu/et/bolt-elektritoukeratas/ (07.05.2021)

[14] “Constants – Expo Documentation”, [WWW]. Available: https://docs.expo.io/versions/latest/sdk/constants/ (07.05.2021)

[15] “Eventbrite – Discover Great Events”, [WWW]. Available: https://www.eventbrite.com/ (12.05.2021)

[16] “Axios Docs – What is Axios?”, [WWW]. Available: https://axios-http.com/docs/intro (05.05.2021)

[17] “SecureStore – Expo Documentation”, [WWW]. Available: https://docs.expo.io/versions/latest/sdk/securestore/ (05.05.2021)

[18] “React Native Localize and i18n-js”, [WWW]. Available: https://medium.com/swlh/translating-your-react-native-app-with-i18n-js-expo-localization-4f6866090e5a (06.05.2021)

[19] “Localization – Expo Documentation”, [WWW]. Available: https://docs.expo.io/versions/latest/sdk/localization/#localizationlocale (06.05.2021)

[20] “Infohunt’s terms & conditions”, [WWW]. Available: https://infohunt.ee/et/node/41 (05.05.2021)

[21] “Infohunt’s privacy policy”, [WWW]. Available: https://infohunt.ee/et/node/40 (05.05.2021)

[22] “Routing and navigation for React Native apps ”, [WWW]. Available: https://reactnavigation.org/docs/drawer-based-navigation/ (06.05.2021)

[23] “GitHub React native Mapview”, [WWW]. Available: https://github.com/react-native-maps/react-native-maps (06.05.2021)

[24] “Google Maps Platform API”, [WWW]. Available: https://developers.google.com/maps (06.05.2021)

[25] “Location – Expo Documentation”, [WWW]. Available: https://docs.expo.io/versions/latest/sdk/location/ (07.05.2021)

[26] “GitHub Directions Component”, [WWW]. Available: https://github.com/bramus/react-native-maps-directions (18.05.2021)

[27] “The Directions API – Google Developers”, [WWW]. Available: https://developers.google.com/maps/premium/apikey/directions-apikey (18.05.2021)

[28] “Publishing and Deploying – Expo Documentation”, [WWW]. Available: https://docs.expo.io/workflow/publishing/ (12.05.2021)

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I Juri Gurjev

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “Infohunt Event Sharing Mobile Application Development”, supervised by Gert Kanter
   1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
   1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

14.05.2021

1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.