TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Department of Software Science

Valeria Furmanets 142419 IABB

MOBILE APPLICATION PROTOTYPE BY THE EXAMPLE OF THE TALLINN UNIVERSITY OF TECHNOLOGY CAFETERIA

Bachelor's thesis

Supervisor: Inna Švartsman

MSc

Lecturer

Tallinn 2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Valeria Furmanets 142419 IABB

MOBIILIRAKENDUSE PROTOTÜÜP TALLINNA TEHNIKAÜLIKOOLI SÖÖKLATE NÄITEL

Bakalaureusetöö

Juhendaja: Inna Švartsman
Magistrikraad
Lektor

Tallinn 2017

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: Valeria Furmanets

21.03.2017

Abstract

The aim of this thesis is to create mobile application prototype for customers, which deals with Tallinn University of Technology cafeterias.

To achieve this aim author has analysed people interest about this idea of application, analysed existing university cafeterias mobile application by describing its functionality, compiled functional and non-functional requirement for TUT cafeteria prototype, analysed types of mobile applications and frameworks for development.

Based on this collected data, author has chosen the most appropriate framework for development. After prototype creating, author test it with people using usability testing questionnaire.

The thesis is in English and contains 46 pages of text, 7 chapters, 20 figures, 5 tables.

Annotatsioon

Mobiilirakenduse prototüüp Tallinna Tehnikaülikooli sööklate näitel

Käesoleva lõputöö eesmärgiks on luua mobiilirakenduse prototüüp Tallinna Tehnikaülikooli sööklate näitel.

Selle eesmärgi saavutamiseks on analüüsitud inimesi huvi selle mobiilirakenduse idee vastu, analüüsitud olemasolev mobiilirakendus, mis puudutab ülikooli sööklaid, kirjeldades selle rakenduse funktsionaalsust, koostatud funktsionaalse ja mittefunktsionaalse nõuded käesoleva prototüüpi kohta, analüüsitud erinevad mobiilirakenduse tüübid ja valitud sobilik raamistik arenguks.

Tuginedes kogutud andmetele on valitud kõige sobivam raamistik arenguks. Pärast prototüübi loomist testitakse seda inimestega - kasutatavuse testimise küsimustikuga.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 46 lehekülge, 7 peatükki, 20 joonist, 5 tabelit.

List of abbreviations and terms

Angular 2	Development platform for building mobile and desktop web application.
API	*Application programming interface*
Bootstrap	Mobile first front-end framework for faster and easier web development.
CLI	*Command-line interface*
CMD	*Command Prompt*
Cordova/PhoneGap	A platform to build Native Mobile Applications using HTML5, CSS and JavaScript.
Cross-platform	Able to be used on different types of OS.
CSS	*Cascading Style Sheets* Style sheet language used for describing the presentation of a document written in a mark-up language.
HTML	*HyperText Markup Language* Describes and defines the content of a webpage.
JavaScript	High-level, dynamic, untyped, and interpreted run-time language.
JSX	Statically typed, object-oriented programming language designed to run on modern web browsers.
Mockup	Illustrated connection between pages.
Objective-C	General-purpose, object-oriented programming language.
OS	*Operating system*
Prototype	Early sample, model, or release of a product built to test a concept or process or to act as a thing to be replicated or learned from.
SDK	*Software Development Kit* Set of software development tools that allows the creation of applications for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform.

SUS	*System Usability Scale*
TUT	*Tallinn University of Technology*
TypeScript	Free and open-source programming language developed and maintained by Microsoft.
User Story	Informal, natural language description of one or more features of a software system.

Table of contents

List of figures

List of tables

1 Introduction

The goal of this thesis is to create a mobile application prototype for customers by the example of cafeterias of the Tallinn University of Technology campus area.

For today, the Tallinn Technological University has 9 cafeterias [1]. Many of cafeterias do not have menu in digital form. It means, that customers can get information about the menu only when arrive to the cafeteria. Also, there are some cafeterias that put the menu on the Internet every day. Each of these cafeterias use different sources. For comparison, customer has to spend time to get information about the menu in advance, moving from one source to another.

1.1 Problem

Tallinn University of Technology has many cafeterias. Each cafeteria differs from another by price, menu, location, opening hours, campaigns. Furthermore, cafeterias are at the different distances from each other. The Tallinn University of Technology is a large educational institution, and it will take time to get from one cafeteria to another.

Usually, time is limited. It applies primarily to university students and workers. It is logical that if time is limited, then you will choose close place for eating. Sometimes happens that the menu does not suit, but lunch plays a significant role in our lives. Looking for the suitable menu takes time, especially when you have to move from one cafeteria to another.

The purpose of this thesis is to create a prototype application for customers, where will be collected information about the menu of all cafeterias on the campus area of Tallinn University of Technology. Customer will get information about menu in advance and choose the most suitable cafeteria for lunch.

1.2 Objective

The purpose of this application is to collect the necessary information in one place. "Necessary information" means the menu of all the cafeterias of the Tallinn University of Technology.

The main objectives are:

- Analyse customer's interests of idea to create mobile application for Tallinn University of Technology cafeterias.
- Find and analyse existing mobile application, which deals with university cafeterias.
- Explore the various platforms of mobile applications and choose the most appropriate for creating cafeteria application prototype for customers.

1.3 Methodology

To obtain the useful information that will help in the implementation of the set objectives, author explores:

- Contact the cafeterias service providers and provide survey among Tallinn University of Technology students and workers to get their opinion about this idea for creation cafeterias mobile application.
- Analyse Tallinn University of Technology cafeterias today by exploring how cafeterias demonstrate menu, which sources they use for that.
- To write out functionality of existing mobile application, which deals with university cafeterias. Based on this information create functional and non-functional requirements for Tallinn University of Technology cafeterias application prototype.
- Analyse types of mobile application. Based on functional and non-functional requirement choose appropriate type and framework for implementation.

The expected result is prototype for customers.

2 Analysis of the need for TUT cafeteria application

2.1 TUT cafeterias today

Tallinn University of Technology has 9 cafeterias in campus area. There are three service providers [1]:

- Baltic Restaurants Estonia AS
- Rahva Toit
- TTÜ Sport OÜ

Some of these service providers have several cafeterias in campus area. However, every cafeteria menu, opening hours differ from another, despite the fact that cafeteria has one service provider.

Every cafeteria demonstrates menu in different way. Some of them write menu on the wall, use Facebook group, campaign advertisements, screens, where customer can get information about menu.

Baltic Restaurants Estonia AS has their own website, where customer can find all Daily [2] cafeterias and see the menu of selected dining hall (Figure 1). Customer can see the menu up to one week in advance.

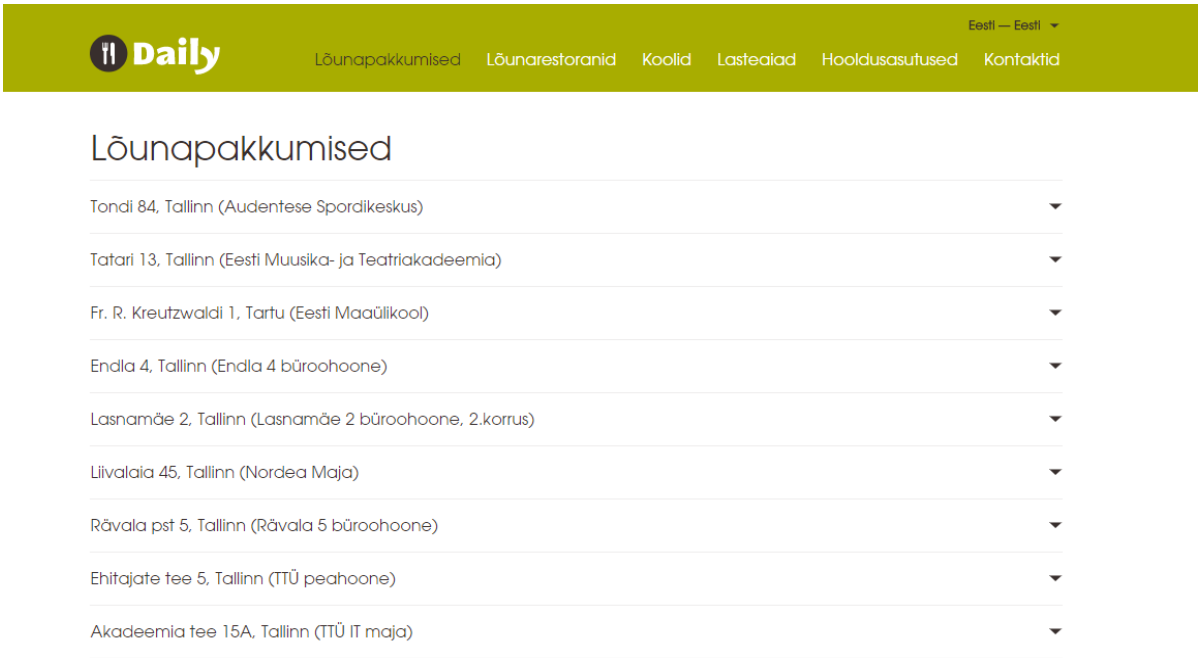

Figure 1. Daily cafeterias [2]

If customer want to get information about menu of all cafeterias, he should find it from different recourses. For instance, if cafeteria demonstrates menu in Facebook, customer should have Facebook account, find this cafeteria and follow it. If cafeteria has website, customer should know about it and every time open the browser to see the menu.

Moreover, some cafeterias do not add the menu on the Internet. Customer can see the menu when arrive to the cafeteria.

2.2 Consumer's opinion

Author has provided interview with Daily cafeteria. As a result of the interview were formed functionality for application like see the menu of current day, cafeteria location on the map, possibility to see special offers.

On the whole, Daily cafeteria supported the idea of creating mobile application for Tallinn University of Technology cafeterias, especially if it is need for customers. Therefore, author carried out a survey [3] among people, who deals with Tallinn University of Technology. Customer's opinion is important of creating something in this kind.

For each product is certain target group of consumers. The survey helps understand people interest for this idea of application.

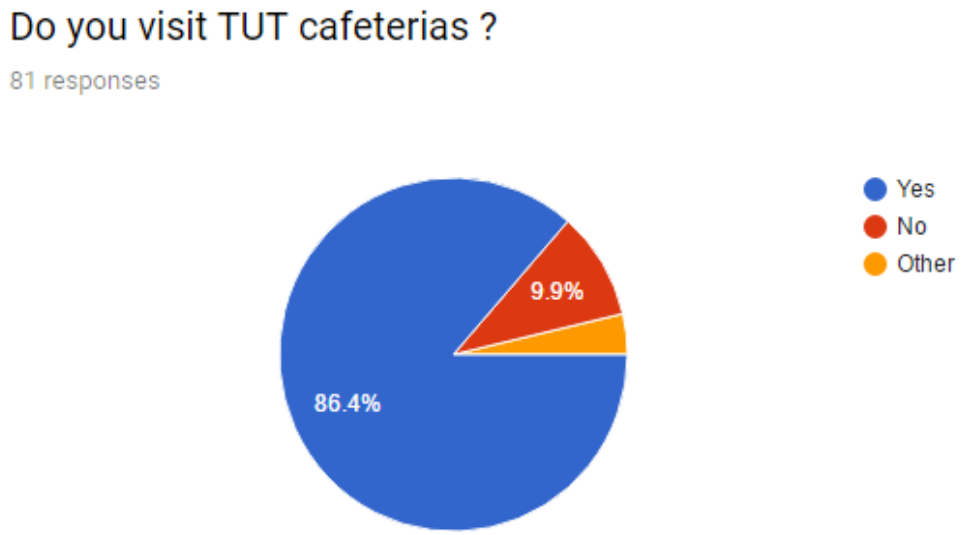

Figure 2. Answer to question "Do you visit TUT cafeterias?"

On this diagram (Figure 2) we can see, that the major part of respondents visit Tallinn University of Technology cafeterias. It is good result, because there are probability that part of them would like to use application, where is possibility to see menu in advance.

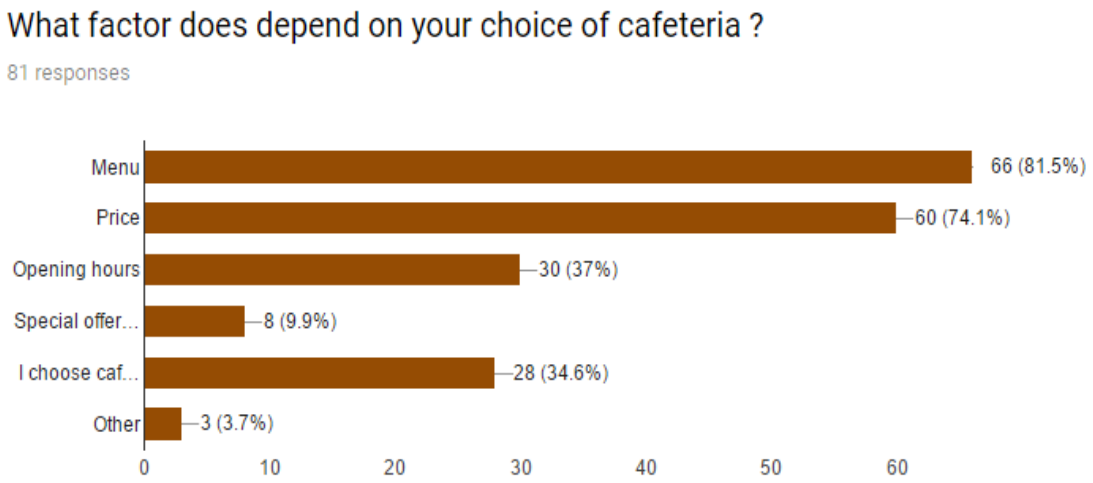

Figure 3. Answer to question "What factor does depend on your choice of cafeteria?"

The main factors that depend on cafeteria choice are menu and price. This information customer will find in TUT cafeteria mobile application in advance (Figure 3).

It was useful to know, in which way customers get information about menu for today (Figure 4).

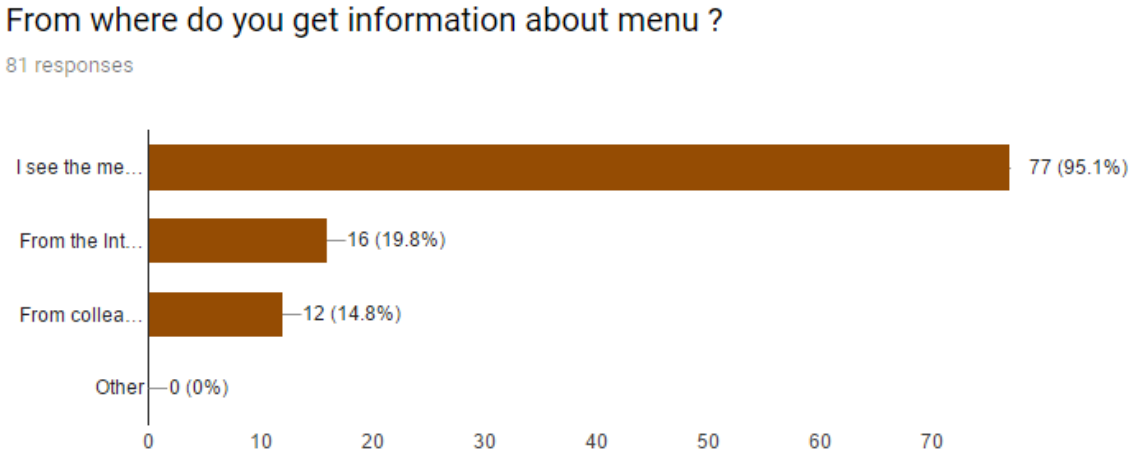

Figure 4. Answer to question "From where do you get information about menu"

The major part of respondents get information about menu when arrive to the cafeteria. It can be caused be factors like they do not about this possibility to know menu on the internet or they choose the cafeteria that is close to them. Because of that, author need the information about general interest of customer to use this application (Figure 5).

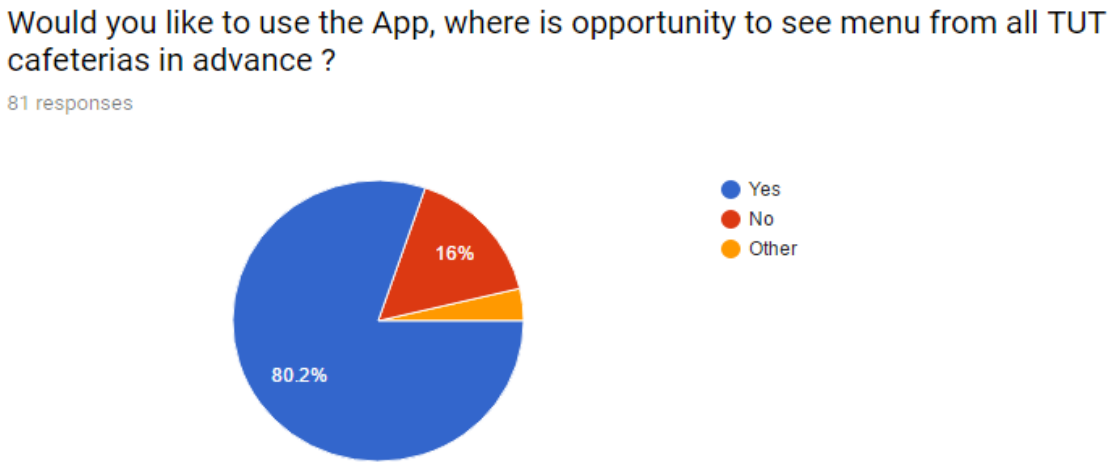

Figure 5. Answer to question "Would you like to use the App, where is opportunity to see menu from all TUT cafeterias in advance?"

We can see that the major part of respondents would like to use mobile application, where is opportunity to see menu from all Tallinn University of Technology cafeterias in advance. It is very good result, because people really have interest for it.

In addition, it is useful to know respondents mobile OS. This information is needed for application development (Figure 5).

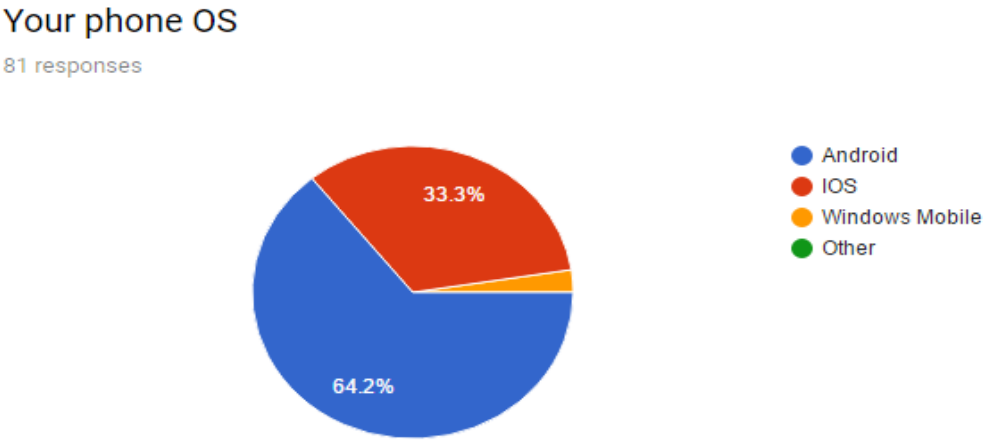

Figure 6. Answer to question "Your phone OS"

The popular mobile OS is Android and iOS. It means that for development is need to use tool that support both Android and iOS platforms.

On the whole, customers would like to use this application. The main factors like menu and price they will find quickly in the application.

3 Analysis of cafeteria mobile applications

Before creating new application, it is good idea to analyse something similar. It can help you in future, for instance, with functional requirement. You will take something from existing application and add something new and useful or do something better.

Goal of this thesis – create prototype of university cafeterias. It means that author is interested in existing applications, which deal with university cafeterias.

Author analysis something similar in Estonia and do not find anything, which deal with university cafeteria mobile application. Of course, some cafeterias, like Daily [2], have their own website and customer can choose cafeteria and view the menu, but it is not mobile application.

Author finds cafeteria mobile application for University of Illinois at Urbana-Champaign [4], which name is UI Dining [5]. On the following paragraph author will analyse this application.

3.1 Analysis of existing mobile application

UI Dining is brought by Administrative Information Technology Services (AITS) [6] and University Housing at the University of Illinois at Urbana-Champaign [7]. UI Dining is the must-have application for anyone who dines at University Housing's award-winning facilities.

Idea of this application is to check out dining information on tablet or other mobile devices of University of Illinois at Urbana-Champaign cafeterias and specialty restaurants. User can get this application from App Store [8] for IOS and Google Play [9] for Android.

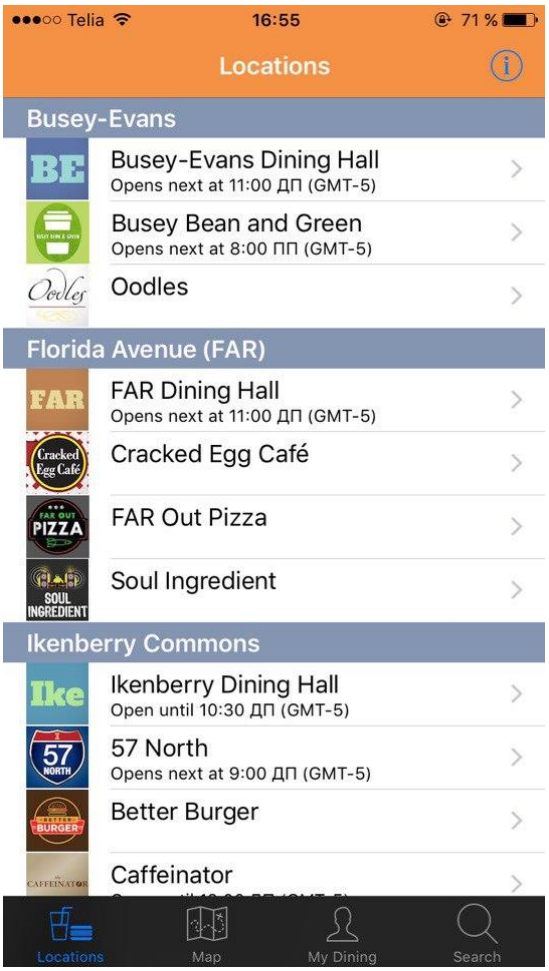

Figure 7. Homepage of existing UI Dining mobile application.

Author downloaded UI Dining application, used it and describes functionality of this application for customers using use case diagram (Figure 8).

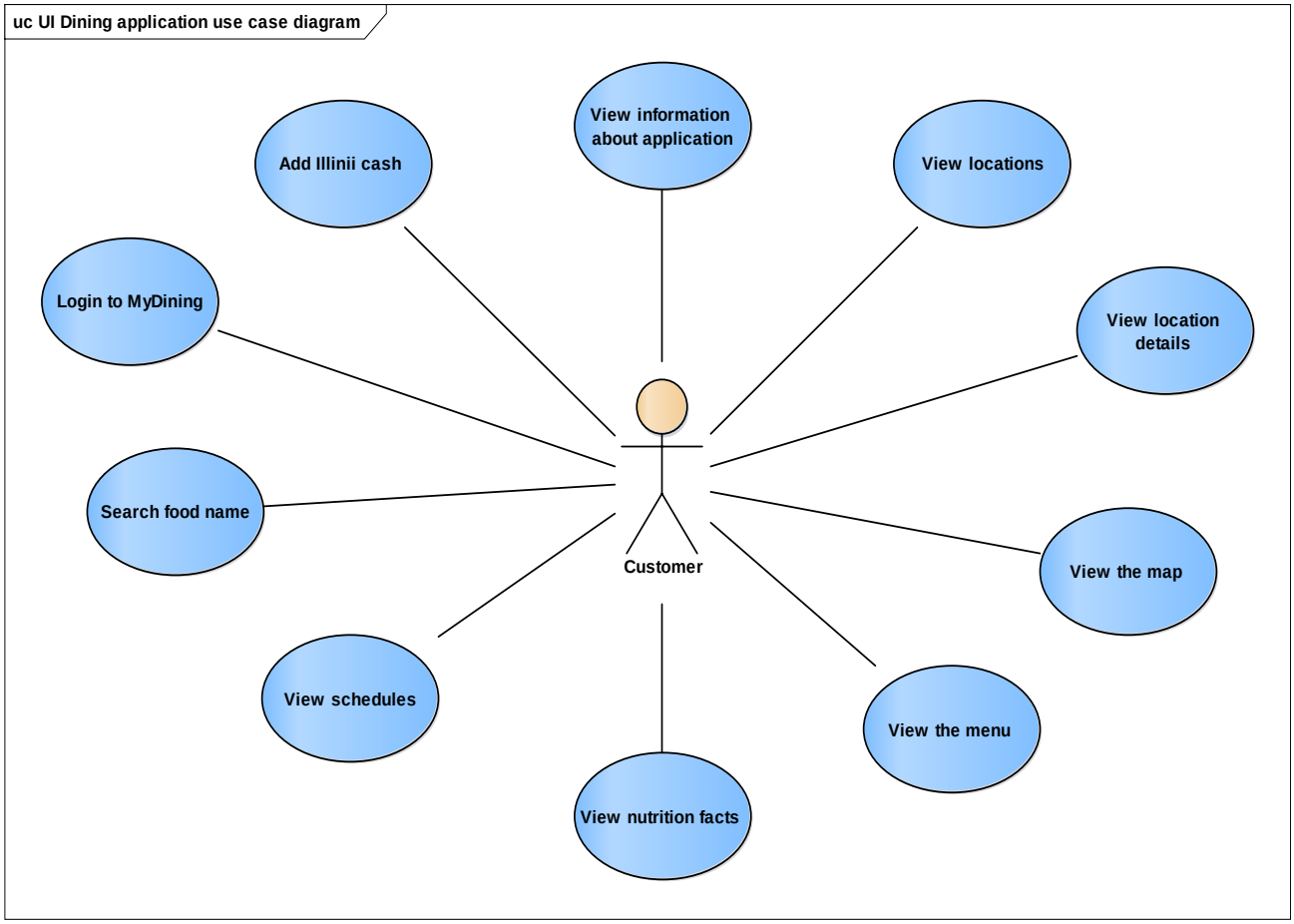

Figure 8. Functionality of existing UI Dining application.

Customer, who uses UI Dining application, has following possibilities:

Table 1. UI Dining mobile application functionality for customer.

Functionality	Description
View information about application	Customer can see information about UI Dining, which includes description of this application, useful links like university website, social pages.
View locations	Customer can see list of University of Illinois at Urbana-Champaign cafeterias and specialty restaurants, which includes in this application.

View location details	Customer can see additional information about location: floor, type of meal, payment method.
View the map	Customer can use map for quickly find the dining hall closest to him.
View the menu	Customer can see the menu of selected dining hall. Menu is separated for section: Breakfast, Lunch and Dinner. Each sector lasts definite time.
View nutrition facts	Customer can see complete nutritional information (for instance calories, total fat, serving size, protein), includes ingredients.
View schedules	Customer can see the menu up to two weeks in advance.
Search food name	Customer can search dining hall by food name. System displays dining halls and date, when customer can visit it.
Login to MyDining	Customer can login to MyDining by *Net ID* and *password*. Login is not required for watching menu. Login give for customer additional possibilities like make meal plan, quantity of eaten meal, view credit balance.
Add Illini cash	Customer can add Illini cash to his i-card. It is quick and convenient way to purchase items like food, printing, books, supplies

	and computer equipment at various locations across campus [10].

UI Dining mobile application has a lot functionality for customers. TUT cafeteria application will have less functionality, because the main goal of this application is view cafeterias menu of current day. As a result of the interview with cafeteria "Daily" [2] cafeteria and survey among customers were formed part of main functionality for customers. In the next paragraph author describes TUT cafeteria functionality in details.

3.2 Analysis of TUT cafeteria mobile application

In the preceding paragraph author analysis UI Dining mobile application generally and describes functionality for customers.

UI Dining application is similar to idea of TUT cafeteria application, because it deals with dining halls and menu. Users have opportunity to check out dining information in advance. UI Dining application consist of cafeterias that located at university campus area. It means that customers get useful information using this application.

Idea of creating application for TUT cafeteria will be simpler than UI Dining. Customer's goal is just to see menu of current day and get useful information of cafeteria.

3.2.1 Functional requirements of TUT cafeteria application

In consequence of the analysis of existing UI Dining mobile application, author defines main functionality for TUT cafeteria application.

Author describes functional requirement of TUT cafeteria application for customers using User Story:

- As a customer, I want to view information about application.
- As a customer, I want to view locations.
- As a customer, I want to view information about cafeteria.
- As a customer, I want to view campaign.
- As a customer, I want to view the map.
- As a customer, I want to change the language.

- As a customer, I want to view the menu.

Based on User Story, author creates Use Case diagram (Figure 9).

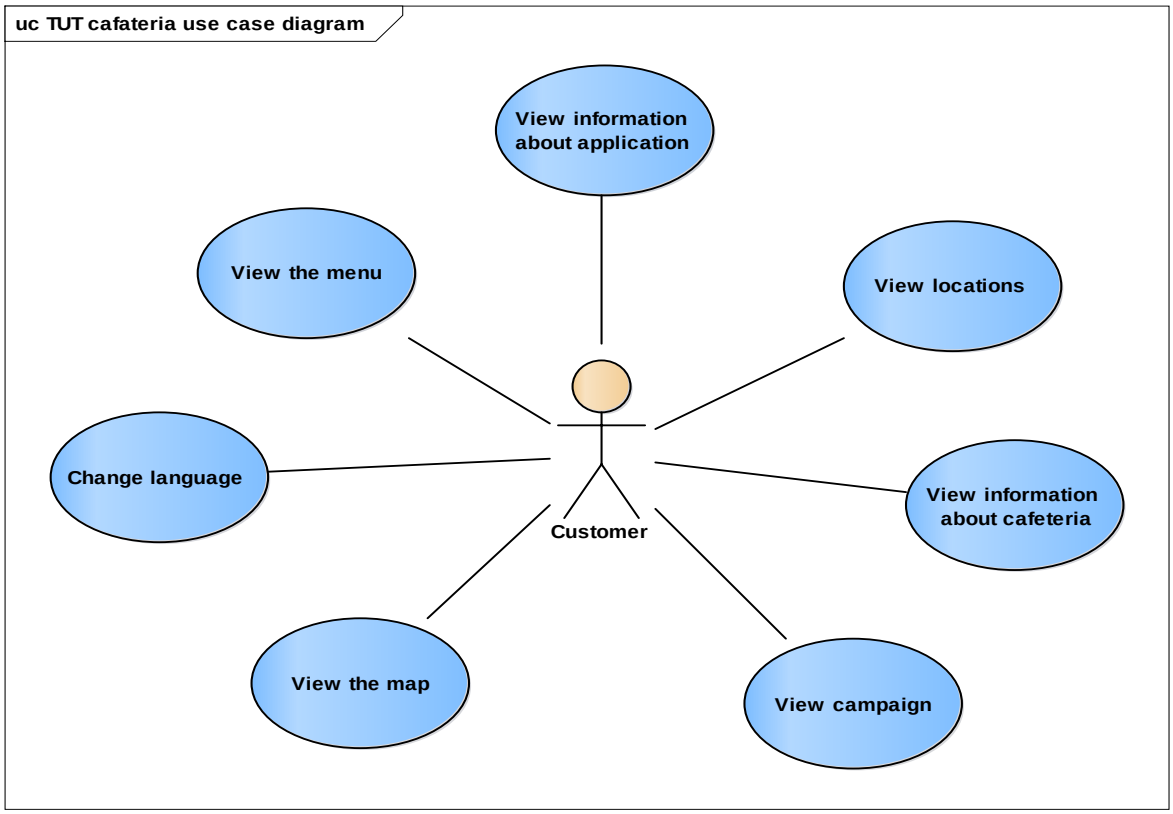

Figure 9. Use case diagram of TUT cafeteria application.

Use case: View information about application

Actor: Customer

Description: Customer can see information about TUT cafeteria, which includes description of this application, useful links like university website, social pages.

Use case: View locations

Actor: Customer

Description: Customer can see list of buildings, where is cafeterias. Customer should select definite location to see the menu.

Use case: View information about cafeteria

Actor: Customer

Description: Customer can see additional information about cafeteria like opening hours, address.

Use case: View campaign

Actor: Customer

Description: Customer can see campaign from different cafeterias.

Use case: View the map

Actor: Customer

Description: Customer can see cafeteria location by using Google Maps.

Use case: Change language

Actor: Customer

Description: Customer can use this application in English or in Estonia languages.

Use case: View the menu

Actor: Customer

Description: Customer can see menu of selected cafeteria.

3.2.2 Non-functional requirements of TUT cafeteria application

Author describes non-functional requirement of TUT cafeteria application for customers (Table 2).

Table 2. Non-functional requirements of TUT cafeteria application

Category	**Non-functional requirements**
Implementation	- Cafeteria application should run on any mobile operating systems (for instance IOS, Android). - Cafeteria application should be available for download on mobile phone or tablet.
Interface	- Cafeteria application should be nice designed on mobile phones and tablets.
Performance	- Cafeteria application do not need fast performance. It will depend on the internet speed.
Supportability	- Support will not be costly.
Security	- Customers do not need to create account for checking menu. Security does not play important role.

Functional and non-functional requirements are ready for prototype of TUT cafeteria application. In the next chapter author describes implementation possibilities for this application.

4 Analysis of implementation possibilities

4.1 Types of mobile applications

There are three basic types of creating application for mobile devices:

- Native
- Web
- Hybrid

Every type of mobile application development platform has benefits and limitations, advantages and disadvantages.

Author will research every type of mobile application and decide which type is more appropriate for cafeteria application.

4.1.1 Native applications

Native applications are specific to a given mobile platform using the development tools and language that the respective platform supports. It means that mobile application is coded in a specific programming language. For instance, Objective-C on IOS, Java on Android [11].

The native mobile application provides fast performance and a high degree of reliability. They also have access to phone's various devices, such as camera. The user can also use the application without internet connection [12]. Developers can use APIs and frameworks, but the code can not be ported to another platform. Also, support and updates are costly, because there are longer development and testing cycles, after which the consumer typically must log into a store and download a new version to get the latest fix.

These applications also have the best appearances and functionality. This type of application is expensive to develop, because it is tied to one type of operating systems. These applications are compiled into machine code, which gives the best performance you can get from the mobile phone.

4.1.2 Web applications

Mobile web applications are software that requires web browsers to run and are developed in a browser-supported language like HTML, CSS, JavaScript. We can say that it is the website that is optimized by mobile phone. Web application have easy compatibility on several operating systems. This kind of application has easy support, you have just to add features and deployed it for all users [11].

There is no native code used as a must and the applications qualify as mobile websites. Generally, though, a mobile website is read-only, while a mobile web application is read-write, enabling the user to interact with the application.

4.1.3 Hybrid applications

Hybrid or cross-platform mobile applications are those that have features of the native application, but run a web application in the way that they can function on two or more platforms. A single codebase and framework are used to write the code for the application, after which the application can run on a number of platforms [11].

Hybrid apps are native applications in functionality and downloadability, but access the internet for user interaction.

4.1.4 Summary of mobile applications types

Author have considered three types of mobile applications – native, web and hybrid. The choice of mobile application depend on goals. We can not say that one type is better than another. All depend on application functionality.

Author has wrote out advantages and disadvantages of these three types of mobile application (Table 3).

Table 3. Advantages and disadvantages between types of mobile applications.

Type of mobile application	Advantages	Disadvantages
Native	- Fast performance and a high degree of reliability - Can use without internet connection - Access to phone's various devices (e.g. camera)	- Works only with one platform - Expensive to develop - Developer should have knowledge of a specific programming environment - Support and updates are costly
Web	- Cross-platform - Easy creation and support	- Need connection to the internet - Can only run in a browser - Slower performance - Do not have push notifications
Hybrid	- Cross-platform - Have features of the native application - Integration of the web and native application features - Faster and cheaper to develop than native	- Need connection to the internet - Slower performance - Updates have to be re-approved

Based on this information, author choose hybrid type of mobile application for cafeteria application. This choice is done because of next reasons:

- Cafeteria application do not need fast performance (it is not, for example game or geolocation services). Also, security does not play important role.
- Possibility to use for IOS and Android (cross-platform).
- Possibility to use push notifications. In future, it will be good tool for cafeteria like advertisement.

4.2 Types of hybrid mobile application frameworks

In this paragraph author describes two hybrid mobile applications frameworks:

- Ionic 2
- React Native

Goal is to research the differences between these two frameworks, to define each type advantages, disadvantages and to understand, which framework suits better for cafeteria application development.

Author chooses these kinds of framework due to their massive popularity.

4.2.1 Ionic 2

Ionic 2 [13] is an HTML5 mobile application development framework targeted at building hybrid mobile applications. It uses web technologies to write and render the application, and requires PhoneGap/Cordova plugins to access native features. It will try to reproduce native behaviours to provide the best user experience.

Ionic is fully cross-platform, because it is capable to build progressive web and native mobile apps for every major application store, with one codebase. It means that Ionic works and looks beautiful wherever it runs. However, Ionic will adapt a few of its behaviour according to the platform. For instance, if you use tabs, they will be displayed just as recommended by the platform - at the bottom of the screen in iOS, and at top in Android (Figure 10).

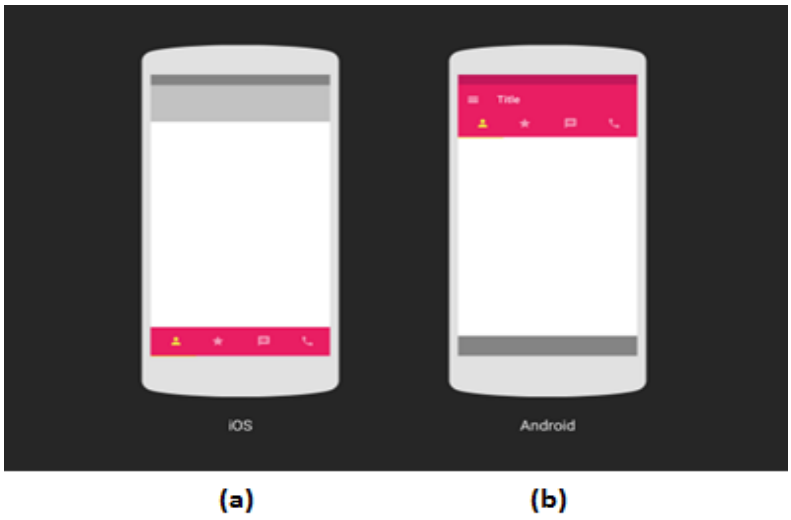

Figure 10. Tabs in iOS (a) and Android (b) [14]

Ionic 2 is a framework based on Angular 2, a Javascript framework. It uses HTML templates for its views.

It is very important to get immediate feedback, when you develop an application. Ionic can instantly preview your application in your browser or mobile devices. It instantly

refreshes as you make changes in code. Moreover, you can see in real time how application will look like in Android, iOS platforms [15] (Figure 11).

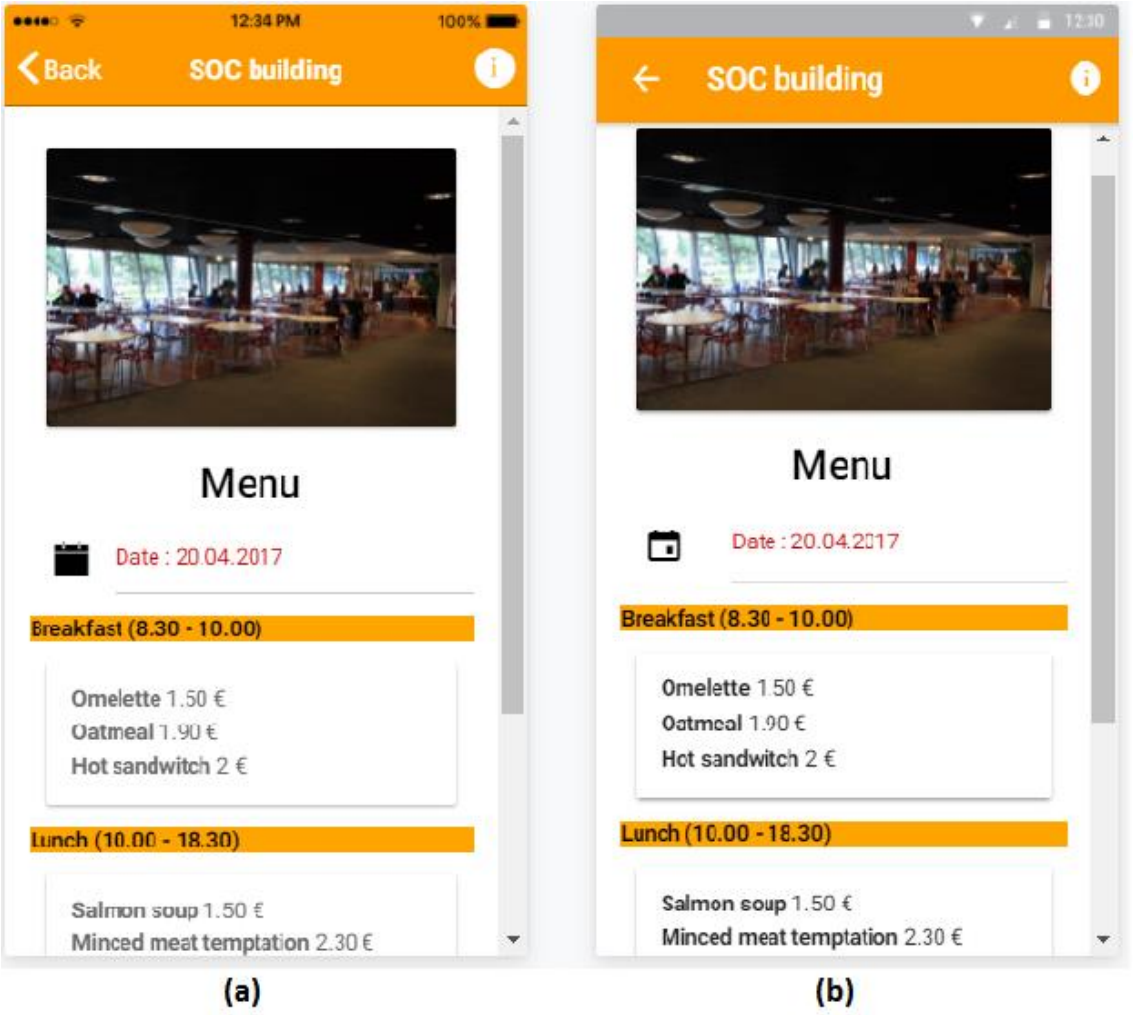

Figure 11. Feedback in: (a) iOS, (b) Android.

Working with Ionic is a bit like working with a CSS framework like Bootstrap [16]. It comes with ready-made components, typography, and an attractive base theme that adapts to each platform.

Ionic 2 supports Android, IOS and Windows Universal Platforms. Moreover, Ionic framework is a 100% free and open source project.

4.2.2 React Native

Developing in React Native is primarily done with Javascript. It means that most of the code you need to get started can be shared across platforms. However, if Ionic render using HTML and CSS, React Native will render using native components. It means that

the user experience will generally be closer to other native applications as they will follow the patterns imposed by the operating system.

The goal of React Native is not to provide a way to write once, and run everywhere compared to Ionic. Developers of React Native should use the components, which best follow the native behaviours of the platform [15].

React Native is based on the Javascript framework React [17] and uses Javascript code that resembles HTML, but essentially is not. It is JSX [18]. With Ionic you will be in more confident, because of classic HTML and CSS. With React Native, you will have to learn how to style and create your user interfaces using their own HTML-like components (Figure 12).

Figure 12. Snippet of : (a) React's JSX, (b) Ionic 2 template on the right [15]

React Native have his own rendering. You do not need to test application in browser. The result is instant in an emulator or a real device. Both React Native and Ionic support Android and iOS [15].

4.2.3 Summary of frameworks

In previous paragraph author has described two types of frameworks – Ionic and Native React. We can not say that the first is better that another. All depends on our project, our goals, requirements, and skills.

Author has created table for visualisation. It helps to understand which framework suits better (table).

Table 4. Summary of Ionic and React Native platforms.

	Ionic	**React Native**
The language stack	HTML, CSS, Angular, JavaScript	JavaScript, JSX
Testing during development	Can get immediate feedback	Can get immediate feedback
Supported platforms	Android, iOS, Windows Universal Platform	Android, iOs
Write once, run everywhere	Yes	No

Based on this information, author choose Ionic framework. This choice is done because of the next reasons:

- Knowledge in HTML, CSS, JavaScript. JSX is something new for author. It will take more time to learn it.
- Important advantage of Ionic is the possibility to write once and run everywhere. It will make work easier and comfortable.
- Time is constrained. Fast application development is needed.

5 Implementation of TUT cafeteria prototype

From the previous chapter author has chosen Ionic 2 for implementation TUT cafeteria prototype. Ionic is the beautiful, free and open source mobile SDK for developing native and progressive web applications with ease.

Prototype is created for customers. Goal of this prototype is demonstrate functionality for customers (Appendix 1 – Link to prototype files).

Prototype based on functional requirement, which is described in *Functional requirements of TUT cafeteria application* (page 23) chapter.

5.1 Ionic 2 installation

Ionic applications are created and developed primarily through the Ionic command line utility (the "CLI"), and use Cordova [19] to build and deploy as a native application. It means that it is need to install a few utilities to get developing.

First of all, it is need to install Node.js [20]. Then install the Ionic CLI and Cordova for native application development in terminal (cmd):

```
$ npm install -g ionic cordova
```

To create an Ionic application it is need to use one of ready-made application templates, or a blank one to start fresh (Figure 13).

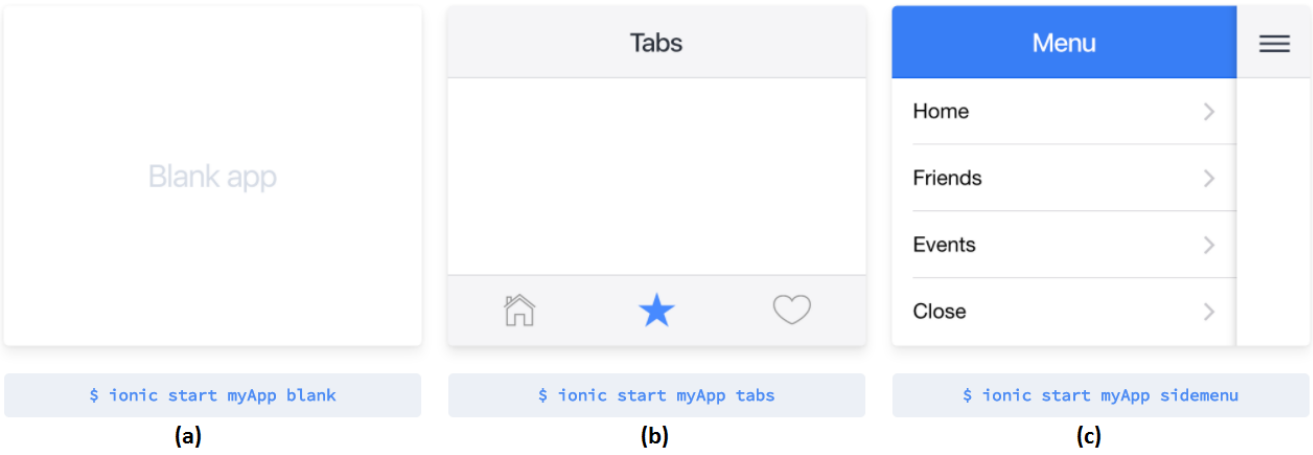

Figure 13. Start Ionic project: (a) fresh project, (b) and (c) ready-made application templates [21]

Author uses blank type:

```
$ ionic start Cafeteria blank
```

Author has running and testing application by using:

```
$ ionic serve --lab
```

With this code, application opens in browser and you can see how application is look like in iOS and Android operating systems (Figure 11).

5.2 Ionic 2 components

Ionic applications are made of high-level building blocks, which called components. Components allow to quickly constructing an interface for application. For instance, for creating Tallinn University of Technology cafeterias list, author uses “List in card” component (Figure 14). It is a card, which can contain a list of items.

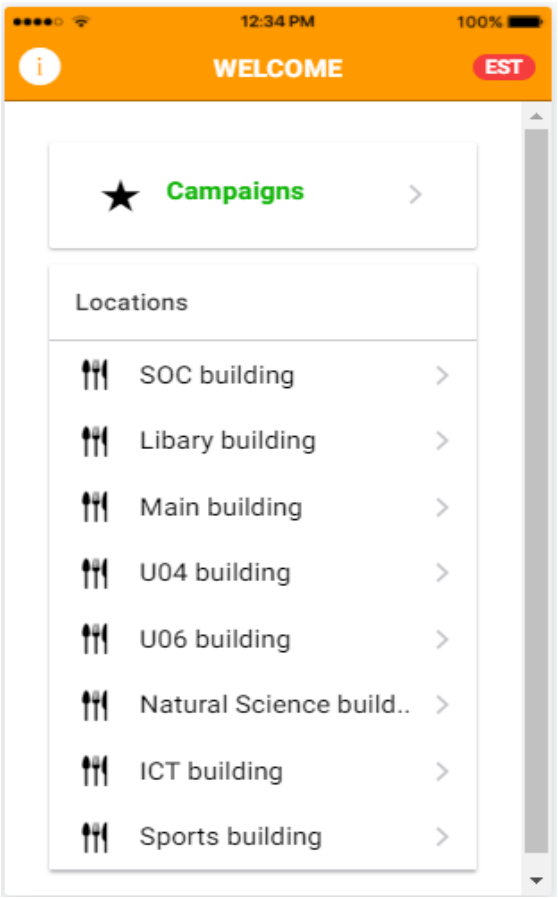

Figure 14. “List in card” component for creating list of cafeterias.

5.3 Google maps integration

The Google Maps JavaScript API [22] can be embedded and used on any website for free. Since it can be embedded in websites, it can also be embedded in mobile applications [23].

Author have added possibility for customer to view cafeteria location by Google Map. To add Google Map to existing project author have done following steps:

- Add the JavaScript SDK

Author has included the library to make the JavaScript API available within application. The following script (Figure 15) is needed to load in *../src/index.html* file.

```
34      <script src="https://maps.googleapis.com/maps/api/js"></script>
```

Figure 15. Google maps API script.

- Load the Map

  When Google Maps Javascript API is available within application author has created and load a map.

  First, it is need to modify file *../pages/soc-building-map/soc-building-map.html* (Figure 16).

```
1   <ion-header>
2     <ion-navbar color="orange">
3       <ion-title>
4         Google Map
5       </ion-title>
6     </ion-navbar>
7   </ion-header>
8
9   <ion-content>
10    <div #map id="map"></div>
11  </ion-content>
```

Figure 16. Modifying .html file.

Author has created *placeholder * to act as a container for the map. *#map* is local variable that used for grabbing reference in TypeScript with @ViewChild [24]. *id="map"* is used for style.

The next piece of code (Figure 17) author has modified in file *soc-building-map.ts* .

```
12  export class SocBuildingMap {
13    @ViewChild('map') mapElement;
14    map: any;
15    constructor(public navCtrl: NavController, public navParams: NavParams) {
16      }
17
18    ionViewDidLoad(){
19        this.loadMap();
20    }
21
22    loadMap(){
23      let latLng = new google.maps.LatLng(59.396945, 24.670000);
24      let mapOptions = {
25        center: latLng,
26        zoom: 15,
27        mapTypeId: google.maps.MapTypeId.ROADMAP
28      }
29
30    this.map = new google.maps.Map(this.mapElement.nativeElement, mapOptions);
31
32      let marker = new google.maps.Marker({
33        map: this.map,
34        animation: google.maps.Animation.DROP,
35        position: this.map.getCenter()
36      })
37    }
38  }
```

Figure 17. Modifying .ts file.

Author describes below main functions and objects (Table 5).

Table 5. Description of main functions and objects.

Name	Description
ionViewDidLoad()	Function, which will run, when the view is loaded to call the *loadMap()* function.

loadMap()	Function, which will handle creating a new map and loading it into map .
latLng	Object, which is provided by the Google Maps API. It is used to represent the location that we want to center the map on. In this case, address is *Akadeemia tee 3* and coordinates (59.396945, 24.670000) [2].
mapOptions	object, which allows to define some options for our map. The center – coordinates that author has created, zoom – the initial zoom level, mapTypeId – type of the map. In this case, author is using the road map style.

Also, author has added *declare var google;* in the beginning. It is needed to prevent any warnings from TypeScript about the google object that the Google Maps SDK makes available.

In the result of integrating Google map, customers can see the cafeteria location choosing road map (by default) or satellite imagery (Figure 18).

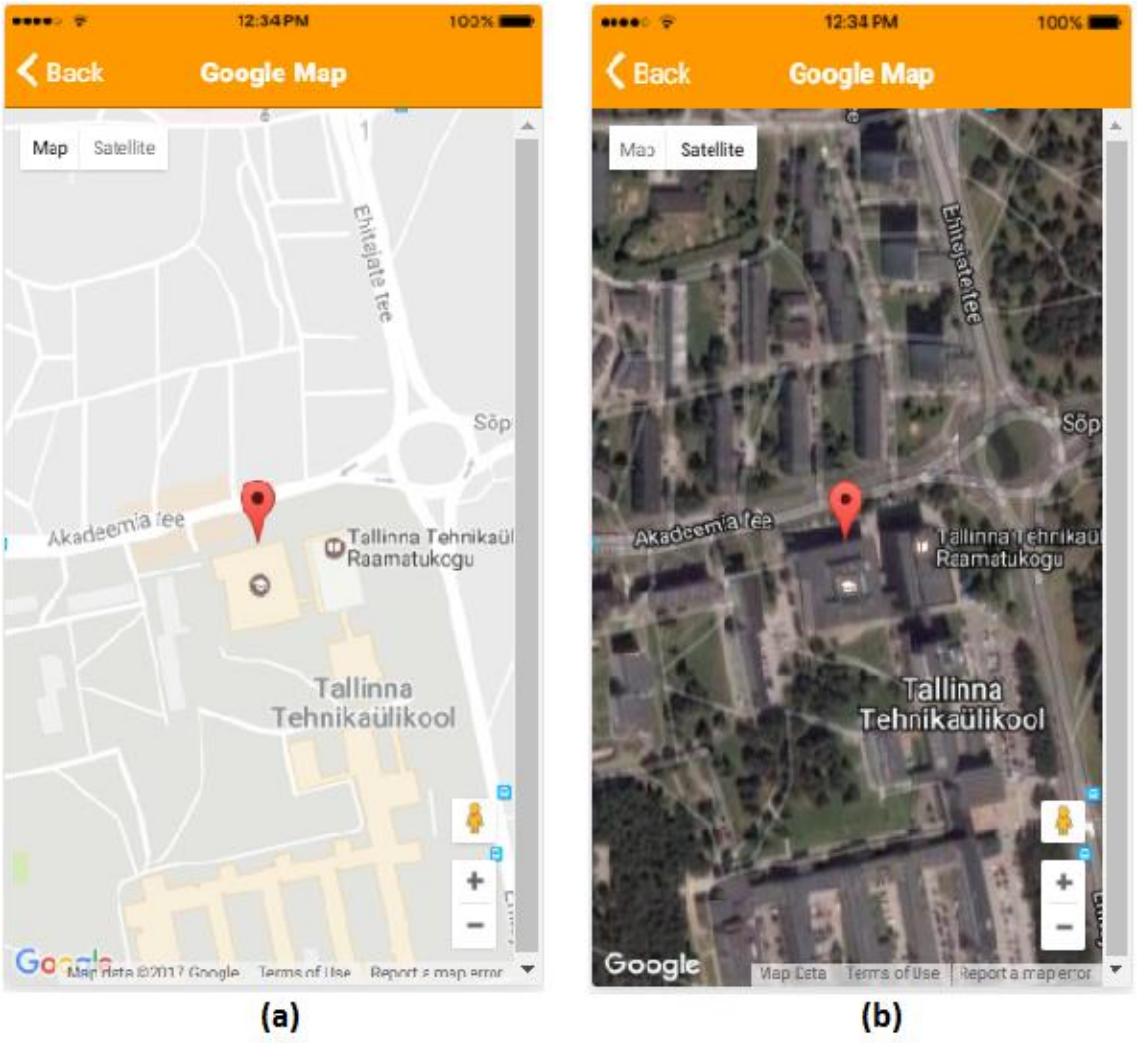

Figure 18. Cafeteria location: (a) road map, (b) satellite map.

5.4 Mockup of TUT cafeteria application prototype

Mockup illustrated connection between pages and how prototype generally looks like.

Author provides mockup of TUT cafeteria application (Figure 19).

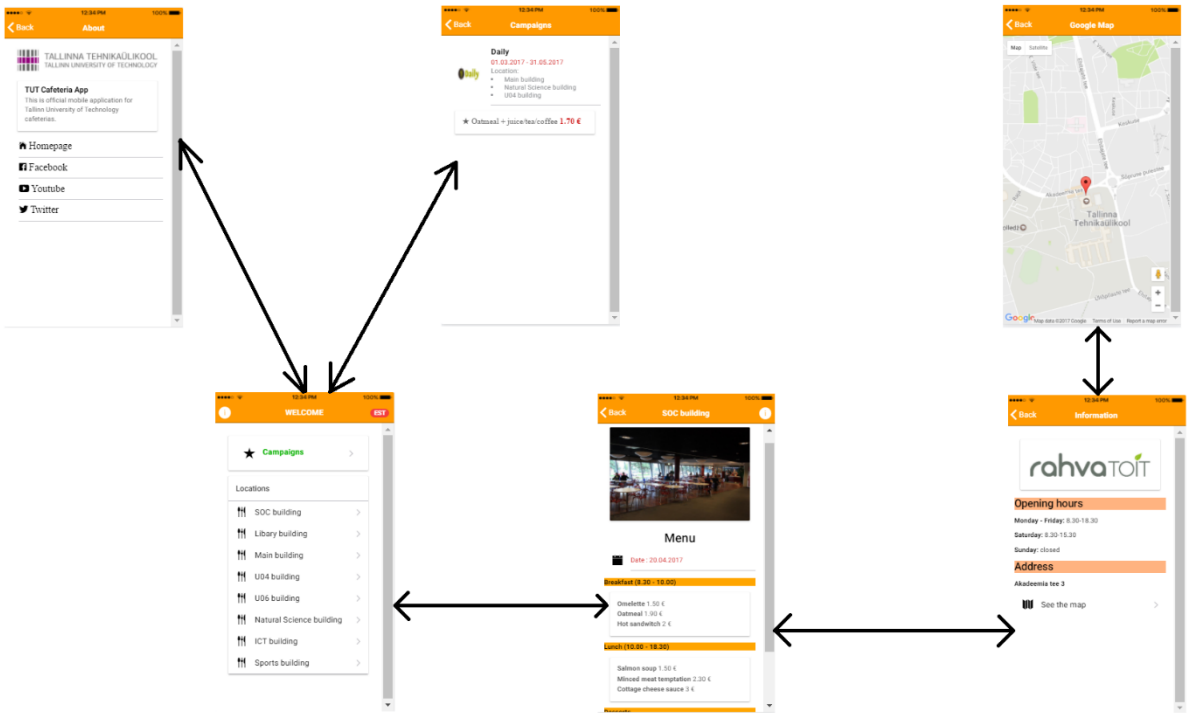

Figure 19. Mockup of TUT cafeteria application prototype.

On the whole prototype consist of 6 pages. From the homepage customer can get information about application, campaigns and chose interesting cafeteria from the list. When customer choses the cafeteria, he will see the menu of current day. In addition, customer can see information about cafeteria. It consist of service provider, opening hours, address. Also, customer can see cafeteria location by using Google map.

6 Usability testing

For testing TUT cafeteria prototype was used Ionic View App [25]. Ionic View allows to quickly and easily load, view and test the applications you build with Ionic Framework on a device. It is need just upload this application on your device from App Store for iOS or Play Market for Android. Author tests TUT cafeteria prototype on IPhone 6 and Sony Z2 mobile devices.

Author creates survey [26] for feedbacks from users, who tests this prototype. Survey is based on „ System Usability Scale (SUS) “, which was originally created by John Brooke in 1986.

SUS provides a quick and reliable tool for measuring the usability. It consists of a 10 template questions with five response options for respondents (1- strongly agree, 5 - strongly disagree):

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system [27].

Usability testing and survey were provided among 5 persons. Testing with 5 people is enough, because it lets find almost as many usability problems as find using many more test participants.

The result of SUS survey is illustrated on diagram (Figure 20). It is need to find SUS score by:

- For each of the odd numbered questions, subtract 1 from the score.

- For each of the even numbered questions, subtract their value from 5.
- Take these new values, which you have found, and add up the total score. Then multiply this by 2.5.

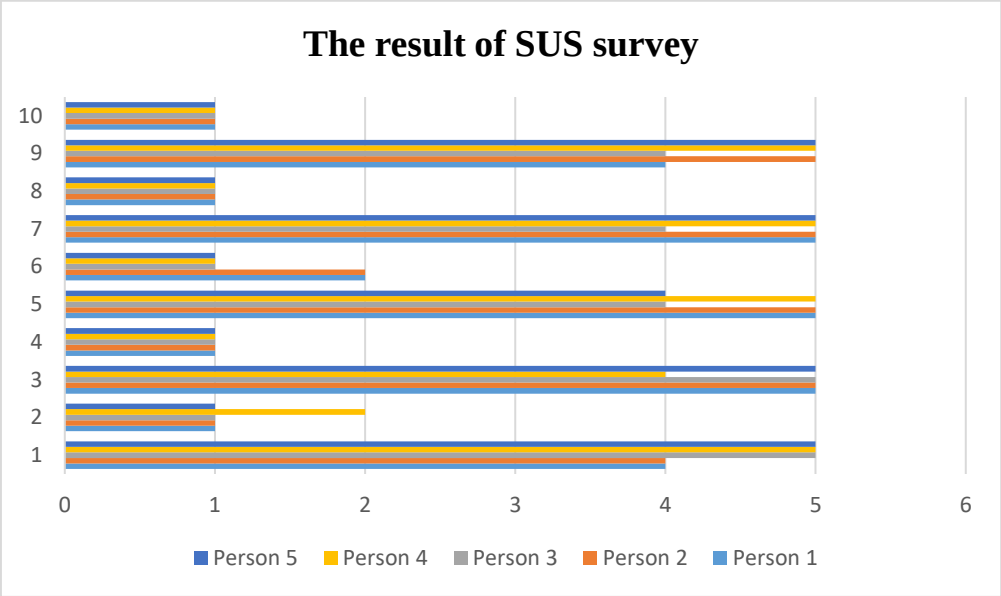

Figure 20. The results of SUS survey.

After calculating, author gets score of 94.5 and it refers to “A”. It means that result is very good.

7 Summary

The goal of this thesis was to create a mobile application prototype for customers by the example of cafeterias of the Tallinn University of Technology campus area. The goal is achieved, prototype for customers is done and people have tested it.

Author has provided survey among Tallinn University of Technology students and workers. Also, interview was provided with “Daily” cafeteria. On the whole, this idea of creating application has interest among people.

Author has find existing application, which deals with university cafeterias. There was done functionality analysis. Based on this existing mobile application and interview with “Daily” cafeteria, author formed functional and non-functional requirements for prototype.

Author has analysed different mobile application types and chosen hybrid type, because it is cross-platform, does not need fast performance. Then, author analysis frameworks, which deals with hybrid application. The prototype is done by Ionic 2 framework. Ionic is the beautiful, free and open source mobile SDK for developing native and progressive web applications with ease.

In addition, author has provided usability testing and got very good result for prototype.

References

[1] "Cafeterias at Tallinn University of Technology," [Online]. Available: https://www.ttu.ee/tudengile/teenused-tudengile/toitlustus/toitlustus-ttu-s/ (14.04.2017)

[2] "List of "Daily" cafeterias," [Online]. Available: http://www.daily.ee/ee/lunch-offers/ (24.04.2017)

[3] "Google Forms: TUT Cafeteria App," [Online]. Available: https://docs.google.com/forms/d/e/1FAIpQLSfO4mSheLzu-TgfYgoxRt0Wit-pi8AuoTkiZ-oCDljKX7ZutQ/viewform?usp=sf_link (17.05.2017)

[4] "University of Illinois at Urbana-Champaign homepage," [Online]. Available: http://illinois.edu/ (24.04.2017)

[5] "UI Dining application page," [Online]. Available: http://www.housing.illinois.edu/dining/menus/dining-halls (24.04.2017)

[6] "Administrative Information Technology Services," [Online]. Available: https://www.aits.uillinois.edu/ (24.04.2017)

[7] "University of Illinois at Urbana-Champaign housing," [Online]. Available: http://www.housing.illinois.edu/ (24.04.2017)

[8] "UI Dininng at App store," [Online]. Available: https://itunes.apple.com/us/app/ui-dining-university-illinois/id575565670 (24.04.2017)

[9] "UI Dining at Google Play," [Online]. Available: https://play.google.com/store/apps/details?id=edu.uillinois.aits.uidining&feature=search_result#?t=W251bGwsMSwyLDEsImVkdS51aWxsaW5vaXMuYWl0cy51aWRpbmluZyJd (24.04.2017)

[10] "Illini cash," [Online]. Available: https://web.housing.illinois.edu/MyBalances/ (24.04.2017)

[11] M. Korf and E. Oksman, "Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options," [Online]. Available: https://developer.salesforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options (28.04.2017)

[12] "Types of Mobile App," [Online]. Available: http://www.socialhunt.net/blog/types-of-mobile-app/ (28.04.2017)

[13] “Ionic framework,” [Online]. Available: https://ionicframework.com/ (04.05.2017)

[14] “Picture of tabs in iOS and Android,” [Online]. Available: https://www.sitepoint.com/android-design-anti-patterns-pitfalls/ (04.05.2017)

[15] F. Massart, “React Native vs Ionic: A Side-by-Side Comparison,” [Online]. Available: https://www.codementor.io/fmcorz/react-native-vs-ionic-du1087rsw (07.05.2017)

[16] “Bootstrap,” [Online]. Available: http://getbootstrap.com/ (07.05.2017)

[17] “React,” [Online]. Available: https://code.facebook.com/projects/176988925806765/react/ (07.05.2017)

[18] “Introducing JSX,” [Online]. Available: https://facebook.github.io/react/docs/introducing-jsx.html (07.05.2017)

[19] “Cordova,” [Online]. Available: https://cordova.apache.org/ (07.05.2017)

[20] “NodeJS,” [Online]. Available: https://nodejs.org/en/ (07.05.2017)

[21] “Get started with Ionic Framework,” [Online]. Available: https://ionicframework.com/getting-started/ (10.05.2017)

[22] “Google Maps JavaScript API,” [Online]. Available: https://developers.google.com/maps/documentation/javascript/ (08.05.2017)

[23] J. Morony, “Ionic 2 & 3: How to Use Google Maps & Geolocation,” [Online]. Available: https://www.joshmorony.com/ionic-2-how-to-use-google-maps-geolocation-video-tutorial/ (08.05.2017)

[24] “ViewChild component,” [Online]. Available: http://learnangular2.com/viewChild/ (08.05.2017)

[25] “Ionic View,” [Online]. Available: https://docs.ionic.io/tools/view/ (11.05.2017)

[26] “Google forms: Measuring the usability of the "TUT Cafeteria" prototype,” [Online]. Available: https://docs.google.com/forms/d/e/1FAIpQLSccNOeBmztL6p5qYddQytFXUQEjfCIidhXVojN-DQcagfh9iw/viewform?usp=sf_link (13.05.2017)

[27] N. Thomas, “How To Use The System Usability Scale (SUS) To Evaluate The Usability Of Your Website,” [Online]. Available: http://usabilitygeek.com/how-to-use-the-system-usability-scale-sus-to-evaluate-the-usability-of-your-website/ (13.05.2017)

Appendix 1 – Link to prototype files

- Prototype files link on GitHub: https://github.com/ValeriaFur/Cafeteria