LiU-ITN-TEK-A--21/010--SE

# A tracking mobile application for school buses

Martin Tran

2021-03-22


**Department of Science and Technology**
**Linköping University**
**SE-601 74 Norrköping, Sweden**

**Institutionen för teknik och naturvetenskap**
**Linköpings universitet**
**601 74 Norrköping**

LiU-ITN-TEK-A--21/010--SE

# A tracking mobile application for school buses

The thesis work carried out in Medieteknik
at Tekniska högskolan at
Linköpings universitet

Martin Tran

Norrköping 2021-03-22


**Department of Science and Technology**
**Linköping University**
**SE-601 74 Norrköping, Sweden**

**Institutionen för teknik och naturvetenskap**
**Linköpings universitet**
**601 74 Norrköping**

Linköping University | Department of Science and Technology
Master thesis, 30 ECTS | Medieteknik
202021 | LIU-ITN/LITH-EX-A--2021/001--SE

# A tracking application

– to keep track of a school bus and its students with the ability to plot the route to the target destination

**Martin Tran**

Supervisor : Pierangelo Dell'acqua
Examiner : Camilla Forsell

External supervisor : Christofer Rydvall

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

## Upphovsrätt



## Copyright

# Abstract

Driving kids to school is a straightforward task but in a world where technology is constantly growing the demand to digitalize every daily process has been normalized. From bus drivers, school administrators to parents there have been demands to track the driven route of the bus and to keep track of the children.

In this master thesis the development of a mobile application to keep track of a school bus and its pupils was carried out as well as the development of a toolkit to visualize the collected data. The aim of the application was to help the client collect data of a driven route. The clients will use the application for resource optimization. Methods within interaction design and user experience has been applied to design an application that works best for the targeted group. Interviews and a user study were carried out to evaluate if the application can be used as intended.

The result has all of the functionalities that satisfy the needs of the clients and it's customers. In case of accident the application is able to pinpoint the exact location of the bus and keep track of the pupils that have been checked in by the bus driver. The application will also log the geographical coordinates of the driven route when being used. A toolkit with a visual map has been implemented to show the driven route and information about the pupils. However the application can be improved by design to help the bus drivers perform actions on the application faster to save time while logging pupils statuses for each station.

# Acknowledgments

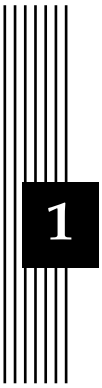
I would like to thank my supervisor Pierangelo Dell'acqua, examiner Camilla Forsell and my external supervisor Christofer Rydvall for giving valuable inputs and important feedback throughout the entire thesis work.

On a more personal note, I would like to thank my family and my partner Sara Lindström for always being supportive during the most stressful times and giving me motivation. I would also like to thank my great friends Johan Jinwoo Yu and Adam Tellia who helped me on my journey to become a software developer. Without any of you this wouldn't be possible.

*Martin Tran*
*Norrköping, July 2018*

# Contents

# List of Figures



# 1 Introduction

This is a master thesis written at the Media Technology and Engineering - Master of Science in Engineering in Linköping university and developed in cooperation with Optiplan AB. This section will describe the background of the thesis work.

## 1.1 Background

In Sweden each municipality in the country are responsible for its residents and their education by assigning school and school buses for the needing pupils. The demand for developing tools to help with the given task has increased over the years. With correct data each municipality can find out which pupil needs transportation more than the other, to be able to assign transportation to the most needing pupils.

Optiplan AB based in Norrköping, Sweden is a company that specializes in optimization for school transportation's. Optiplan AB has developed an algorithm that calculates the best possible route and stations to pick up pupils and transport them to school according to distance. Optiplan AB also helps their clients to decide which pupil needs transportation the most by calculating the distance from home to school for each pupil.

## 1.2 Purpose

The clients of Optiplan AB, Skövde kommun and Norrköping kommun have asked for an application that is able to provide necessary information of each route for a bus driver to deliver pupils to school and at the same time track the driven route. The idéa was for bus drivers to use the mobile application as a tool to help the driver while collecting information of the driven route. The collected data have to be visualized to help a user optimize a new route for the upcoming periods since not all calculated route and stations are optimal since the algorithm only takes distance in consideration. By having a visualization it's easier to see where the bus stops and if it can be possible to merge some stations with each other. The desired application needed to be developed in Android and was going to be used by bus drivers to check-in and check-out pupils. The client also wished for the mobile application to

display the calculated route for the bus drivers. Another feature that was discussed was for the application to log the geographical coordinates of the bus while beings used in real time. The collected data would be used to visualize the driven route compared with the planned route.

The discussed available actions for a school bus driver looks accordingly:

- Sign in with a unique id
- Choose which route to track
- See which station is the next stop
- See for each station which pupils shall be picked up and which station they are planned to get off at
- To be able to check in and out pupils when necessary

Requirements that were important to achieve to help the end users were to develop a user friendly interface to make it possible for the bus driver to check-in and check-out pupils without unnecessary time being spilled. While using the application the bus's GPS position has to be logged in real-time while the application is active to minimize the tasks for a user. Each bus driver will also be responsible for their own identification which consequently required a safe authentication.

The discussed available actions for a customer to examine the collected data looks accordingly:

- Choose a driven route that has been logged with the application.
- Choose which options to visualize on the map
- Examine the tables with check-in and check-out data

The solution to visualize the collected data required a function to compare the planned route with the driven route. The toolkit also needed a function to compare multiple routes with each other for the customers to be able to analyze unexpected behaviours. By comparing multiple driven routes of the same route any systematical error can be analyzed.

Optiplan AB has contributed to this thesis by giving access to the required data and by sharing their knowledge and expertise in software development.

## 1.3 Problem formulation

1. How should the user interface be designed to help a user to check-in and check-out pupils the fastest way and help the users stay focused on their driving?
2. How should the structure for database be designed to easily separate data for each customer?
3. How should the application produce a better user experience compared to similar applications
4. How should the toolkit be implemented to best visualize the collected data for its customers?

## 1.4 Limitations

Optiplan AB is developing with .Net framework and the programming language C# therefore the application was restricted to the mentioned framework and language. Another limitation that arose during the thesis work was the limited API choices that were suited with the selected framework.

# 2 Theory

This section describes the theoretical background of the thesis work.

## 2.1 Related Work

Similar applications have already been developed and consequently a study should be carried out before the development. This section will go through and compare existing applications.

### 2.1.1 Versatrans My Stop

Versatrans My Stop has features similar to the app Optiplan AB wishes to develop. The app is available for mobile devices and the user can track the bus exact position having its location being updated every five seconds. Versatrans My Stop provides information to the user by notifications resulting in decrease incoming phone calls to the transportation company [1]. The app does not provide any solution for attendance tracking of the pupils and only track the bus current position. In Figure 2.1 [2] the app is showcased. The app displays the bus position and by selecting a specific pupil it is possible to see the time of arrival for that pupils station.

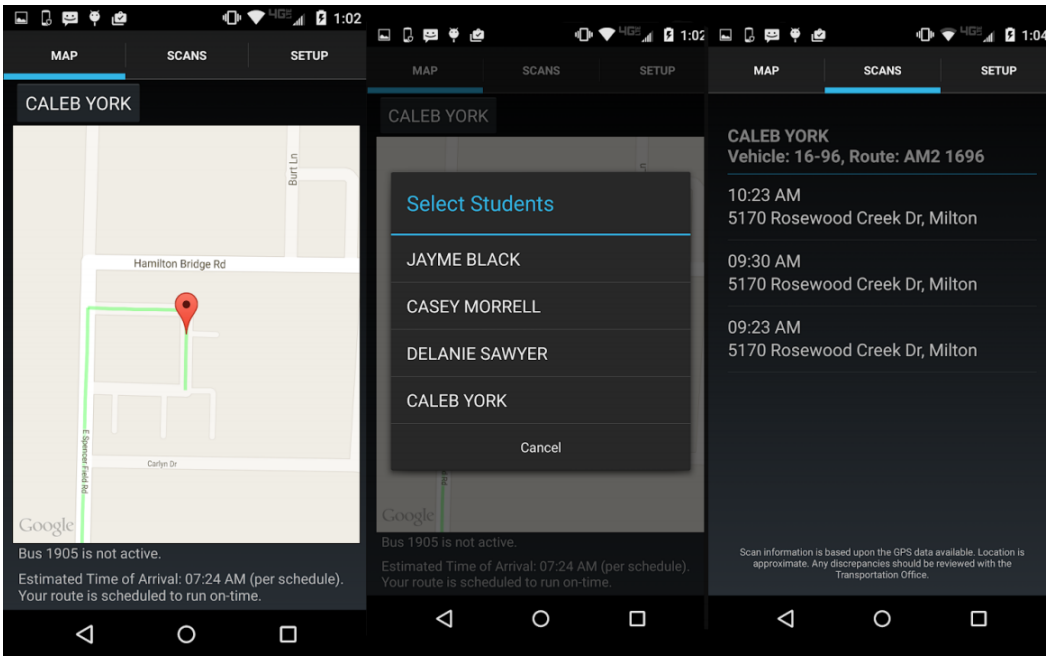



Figure 2.1: The user interface for the app Versatrans My Stop

### 2.1.2 UbicaBus

UbicaBus is mainly developed for usage by website. Parents can with UbicaBus track buses in real-time to see where their children are at the moment. Other features in UbicaBus are mainly used by school administrators. As a school administrator it is possible to fully monitor buses, pupils and drivers [3]. Some of the features UbicaBus have are features that will be adapted and improved in this master thesis. The feature that will be added into the developed mobile application are ways to monitor the driven route. According to reviews on app stores the mobile app from UbicaBus does not have a user friendly interface and needs improvements [4].

### 2.1.3 Here Comes the Bus

Here Comes the Bus is an app where the parents as users can check the bus location in real-time. The app has two main functions. The first one lets the parents see their children's location in real-time by displaying their position on a map and also notifies the parents about the bus's arrival and departure time. The second main function is called pupil Ridership and lets the parents know at which stations their children entered and left the bus [5].

Here Comes the Bus has features Optiplan AB wishes to implement. By examining the user review of the app it indicates that the features are useful but the execution can be improved. There are two main complaints about Here Comes the Bus which will be taken into consideration when implementing similar features in this master thesis. First complaint is that the app has a lot of crashes and bugs when there is a substitute bus. The app doesn't show the bus location or their children's statuses. The problem arises because Here Comes the Bus uses RFID (Radio-Frequency IDentification) readers or Mobile Data terminal that are tied to specific buses. This means when the pupil enters the bus they scan using ID cards in which the GPS device pinpoint the location and notifies the parents. If the substitute bus doesn't have RFID reader installed the information about the children will never be registered. This

indicates that the developed application for the master thesis needs to have a solution that doesn't require extra hardware. The other complain is about the user experience and interface of the app. According to user reviews the app gives too much information which confuses the user [6]. In Figure 2.2 an overview of the apps user interface is showcased [7].

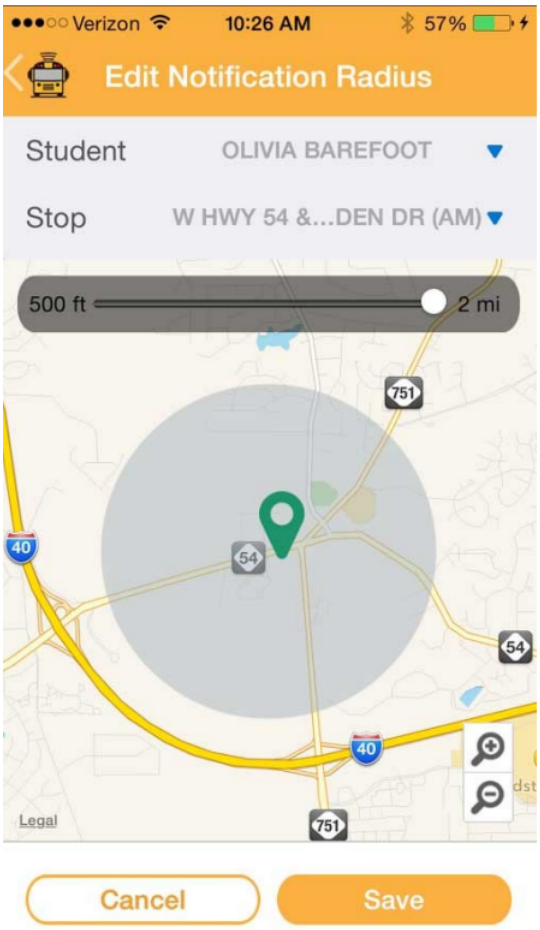



Figure 2.2: User interface for the app Here Comes the Bus

## 2.2 Designing a mobile application

In order for an application to be successful it needs to be designed with User Interface (UI) and User Experience (UX) in mind. A bad user experience is one of the most common issue as to why the majority mobile app users delete their apps [8] [9] [10]. Design is the main key component of innovation. Bad designs often lead to frustrated users and forces the developers to change the application which is costly for the development. Before developing an application the process of designing the applications interface with UI- and UX-design are important.

### 2.2.1 User interface

In principle the user interface is easy to implement. The idéa is to design the app for the user to see and change the data in the system and allow the user to perform any command

across the technical interfaces. A system with such functionality is useful but not necessary easy to use. If it's complicated the user will lose interest in the application. According to Euphemia Wong a UX researcher, designer and Strategist at Interaction Design Foundation [11], Soren Lauesen the author of User Interface Design a Software Engineering Perspective [12], Scott W. Ambler the author of User Interface Design: Tips and Techniques published by Cambridge University Press [13] and Wilbert O. Galitz the author of The Essential Guide to User Interface Design the second edition [14] there are numerous of rules to follow when designing a user interface. The rules that have been listed and occurred in all of the mentioned books and article to increase the apps usability, utility and desirability are consistency.

An application needs consistency or the user will experience confusion. Consistency is achieved by using the same terminology and procedures in the whole system. If the application has multiple pages and a function is re-used then the interface needs to look the same for the said function. The application should also be self-explanatory and interactive with the user. Experience users and new users alike should be able to navigate the application without further instructions. If an application contains of numerous buttons the user should know what each button does without clicking the button. Further when a user clicks a button a feedback should be returned. A good user interface tells the user what their actions have led to and constantly updates accordingly to the users action. A human's attention is restricted and a design with a lot of information may confuse the user. The application should have a good mapping and only display the information user is working with. With the right amount of visualisation a user will be able to calculate what is next and act accordingly.

According to Caroline White, a UX analyst and writer simplicity is another key component for user experience and is referred to one of the most fundamental principles of having a user friendly design [15]. In order to implement simplicity the designer needs to define "simple" for the application that will be developed. Simplicity is a concept which varies depending on the features an application are built from. The core concept behind simplicity is the number of component a design has and should only be the exact number of components that are needed to perform the task. By having less components the user wont get confused.

### 2.2.2 User experience

The aim of user experience is to learn about the end users for the application in development. The process of user experience is illustrated in Figure 2.3 according to UX Mastery [16].

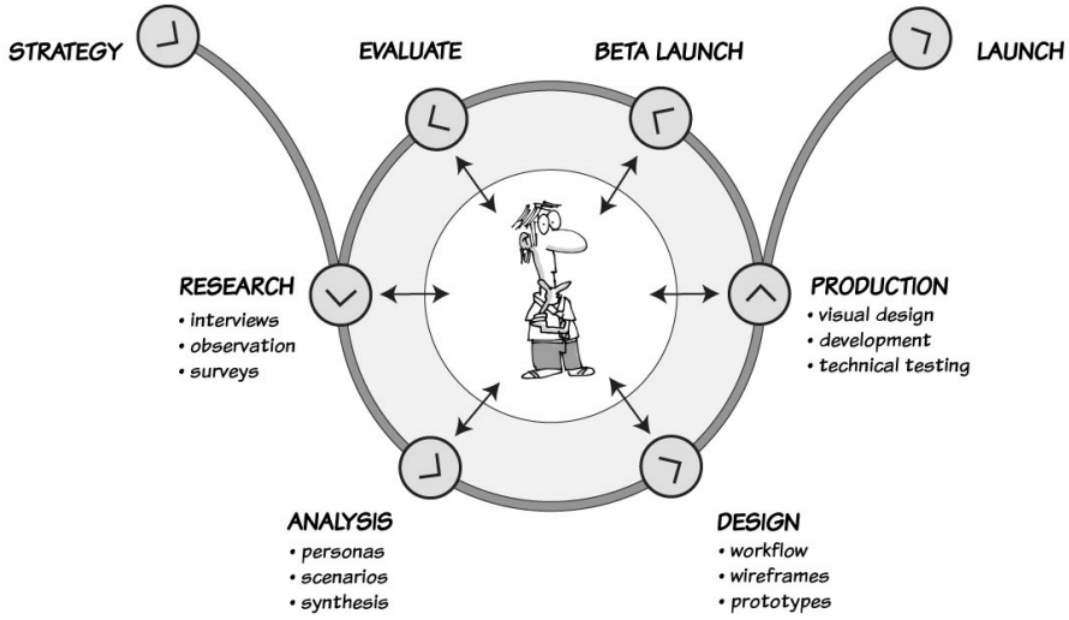



Figure 2.3: The process of user experience

#### 2.2.2.1 Users

An application may have different kind of users. Further the users can be categorized in different groups. The users with similar attributes are divided in the same group and are expected to react to the application in similar manners. For an app to be successful the app have to be designed with the end users in mind. Knowing the end users mindset will help shape the app to be interactive and useful for the users. A thorough research is important to collect the necessary data. To collect the data it requires customer interview, surveys, ethnographic research and competitive reviews [17]. The collected data is the key to create an informed user experience for the end user. An analysis of the data is performed to create personas and scenarios for the application and applying it to the design.

#### 2.2.2.2 Design and Prototype

The beginning of design phase is to sketch multiple solutions. It's recommended by Google's Design Team to sketch the solutions on paper instead of wireframes for the reason that it's quicker and that changes are more easily adjusted [18]. The objective of sketching is to collect as many idéas as possible and at the end of the phase combine the best solutions and decide the design of the application. Budget, users, technology capacity and business drives are important factors for the decision. If possible accumulating idéas from the masses and input from the user at this stage is of importance. A prototype of the chosen design has to be created and later tested. The prototype has to give the test-user the experience of look and feel of the product. It's about taking the structure and visual of the design into real life simulation. There are multiple type of prototyping: paper prototyping, digital prototyping and fully programmed prototypes [19].

#### 2.2.2.3 Production and Testing

Last phase is validating the prototype by testing the product on users. There are multiple factors to take into consideration when evaluating the result from the test since the prototype will decide the outcome for the end product. The prototype helps the developers to learn if the system is usable, easy to handle for end-users, provide the desired solution to the user's problem [20]. If the prototype fulfills the requirements the development of the application may start or the process of user experience needs to get repeated as seen in Figure 2.3.

### 2.2.3 Effect mapping

Before development a description of the end product is given but mostly very faintly and it's rare to have a defined aim on how to achieve it. This often leads to an unfinished product and dissatisfaction from the customers and users which leads to delays and rework of the product. The effect map is a method to eliminate these shortcomings under a development process, as described by Ingrid Ottersten and Mijo Balic in Effect Managing IT [21]. The effect model describes how to achieve the data for a better end result without the shortcomings. The effect model depicts multiple questions that must be answered to collect the needed data. Finding out why the product is being developed, finding out the targeted group and how they interact with the product and how to design the product with the previous points in mind are the key points of the effect model.

The aim should be formulated with one sentence to have a clear objective. The aim should describe what benefit a user can get from the product or how to develop a product that attract the users. To achieve the aim there can be multiple measurements, all that can be attest to. There can be multiple usage goals, also known as effect goals depending on the target groups to achieve the aim. Commonly the target groups are divided in different categories. This depends on that people are different and the idéa is to map a group of users with similar behaviours and same patterns when using an application. Each group have different expectations from the product and different experiences when using the exact same application. By finding the targeted groups it's much easier to develop a product that suits all of its users by setting an effect goal for each group. This will lead to developing a product that meets the needs and expectations for the most highly prioritized users.

## 2.3 Development methods

Optiplan AB exercises agile methods in their development process. As a consequent the agile method was adapted into the thesis work and will be explained in this section.

### 2.3.1 Agile methods and Scrum

The foundation in agile method is a combination of iterative and incrementally work with the idéa of working close to the customers. Agile method is a collective name for system development methods of the same character. As a result of the iterative and incrementally working methods in agile the development process is adaptive to changes during the work. There are multiple types of agile development, one which will be used in the thesis work is Scrum, described in The Scrum Guide by K. Schwaber and J. Sutherland [22].

A team that works according to Scrum divides the work in visible, iterative and incrementally phases, called sprints. A sprint can last from one week up to one month, the workload is decided in each sprint to distinguish what should be developed in the time period. A sprint starts with a team meeting, contact is being made with the customer and the developers in the team divide the work between each other. A sprint ends with a corresponding meeting

to evaluate the work done and to discuss improvement possibilities. An accomplished sprint means that a milestone has been achieved and a deliverable product can be delivered according to plan. In iterative work a deliverable product has to be ready after each sprint but there are variations in Scrum where it's enough to deliver small fraction of the product each sprint. Generally for Scrum a short daily meeting is held for the developers to discuss what has been done and what has to be done. These meetings are called stand-up and is held with the participator standing up to keep the meeting short and effective. With the arrangement described the developers have as much contact as possible amongst the team to keep all the members up to date with each step in the development and changes that has been made. The developed applications requirements, customers wishes and priorities are stored in a backlog to keep track of every functionalities that is needed.

Agile methods such as scrum has been developed to work for a small team. In a large team the developers will lose insight in each others work which is the concept of scrum. Scrum can be applied to one man team with planning, iterative work and dividing the development into phases. One flaw with Scrum is that it's hard to plan sprints at the start of the project. One solution to minimize the flaw is called sprint 0. In sprint 0 the projects architecture, requirements and preliminary sprints are summed up to get an overview of the whole project. With the help from the overview meetings with customers can be planned and the developers can research what's necessary and test any technique they find unfamiliar.

## 2.4 Software architecture

The accessed data, consisting of pupils and the planned route during the thesis work was provided by Optiplan AB. The new data consisting of the driven route and statuses of the pupils were saved and retrieved using web services with the architectural style "representational state transfer" shorten for REST.

### 2.4.1 Representational State Transfer

REST is an architecture style which is based on web standards and HTTP protocol. This style was initially described by Roy Fielding in 2000, same author of the principal of HTTP specifications [23]. REST relies on a stateless Client-Server communication protocol and REST is operating using HTTP [24]. To understand REST the user needs to understand the HTTP protocol. When the client enters a URL in a web browser the client is sending a request to the web server. The web server responds with a resource and delivers the request to the web browser illustrated in Figure 2.4.

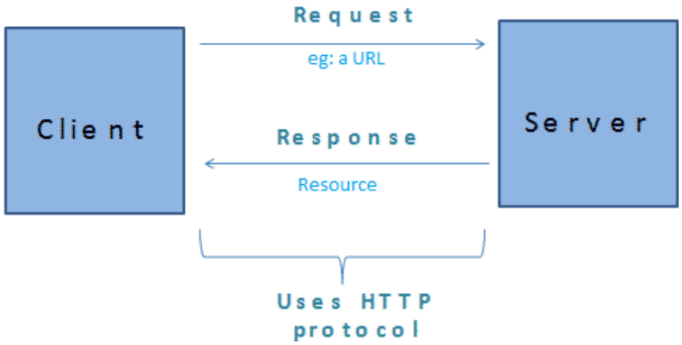



Figure 2.4: Illustrating HTTP protocol

REST service makes use of the HTTP specification with a URL structure. The URL of the request contains information that the client needs meaning instead of a URL representing a directory structure and a specific file in a specific folder, in a RESTful service it is pointed at the RESTful state of the URL that represents the data.

The concept of REST is that the client passes necessary information and key to the server and the fundamental about REST service is that it is stateless. It does not maintain the state on the server that the client passes so no information is being retained by either sender or receiver. With the given information and key the server performs an action and accepts the request. The server then returns to the client with the action completed, making REST resource based. The resources are identified by URIs and individual resources are identified in each request. In addition the representations are separate from the resources. The term representations refer to how resources get manipulated in a RESTful API architecture. The resources represent part of the resource state and that representation gets transferred between a client and a server. When a client holds a representation of a resource it is attached to the request and with enough information the resource can be modified or deleted on the server given that it has permission. A RESTful service uses hypermedia response such as XML or JSON web services notation. It uses the full range of the HTTP properties and then it uses XML or JSON to return back to the client to be further processed.

REST was made to treat objects on the server side as resources that can be created, updated and destroyed or deleted. REST makes it possible to provide standards between computer systems on the web, making it easier for systems to communicate with each other [25]. The REST architectural style is composed of five constraints. The five constraints are following: Client-Server, Uniform Interface, Stateless, Cacheable and Layered System. Without any of the constraints the REST architecture would fall apart.

### 2.4.2 Client-Server

With the help from Client-Server architecture it is possible to improve scalability of the application by simplifying the server components. This is done by separating the application into concerns and that is the principle behind Client-Server constraint [26]. An application may have a user interface and a data storage and by dividing the user interface into one concern and data storage to another concern the separation is complete. In this constraint the information flow goes through the client who sends a request to the server. The server processes the request and delivers the response. In this way the portability of the user interface is improved and can easily be combined with separate and different data storages when the client makes the request.

### 2.4.3 Stateless

Stateless means that the server doesn't save any context from the client. To make a request, each individual request has to contain the necessary state to handle the transfer. This constraint guarantees that any given response from the server is independent. By removing the server side state the complexity is reduced and induces the properties of visibility, reliability and scalability [23].

### 2.4.4 Uniform Interface

This constraint guarantees that the request is the same regardless of the client [27]. For the request to be approved by the server a request must have a resource identifier. If the request

is approved a response from the server will return all the information the client needs in order to understand the response.

### 2.4.5 Cacheable

The data sent from the server contains information if the data is cacheable. If the data is cacheable this constraint helps to improve network efficiency. This constraint guarantees that the client will not accept the same requests when the data has already been returned. The client will send a new request to the server when the current data has been expired [28]. By implementing cacheable constraint the risk of having duplicate data is terminated. An illustration of a cache flow chart is presented in Figure 2.5 [29].

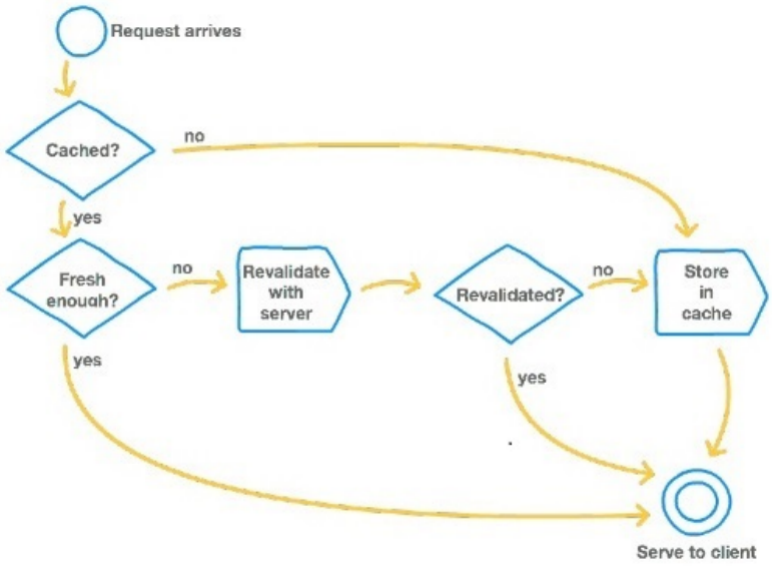



Figure 2.5: Flow chart of cache-system by Luis Cipriani, Rocket Internet SE

### 2.4.6 Layered System

In a request and response operation there might be a number of servers in the middle. To improve the internet-scale properties the layered system is added. This constraint is composed by multiple layers in a hierarchy and guarantees that no layer will interfere with the active layer interacting to the request and response. This means that the REST is restricted to a single layer and the overall complexity will be reduced. The constraint is illustrated in Figure 2.6.

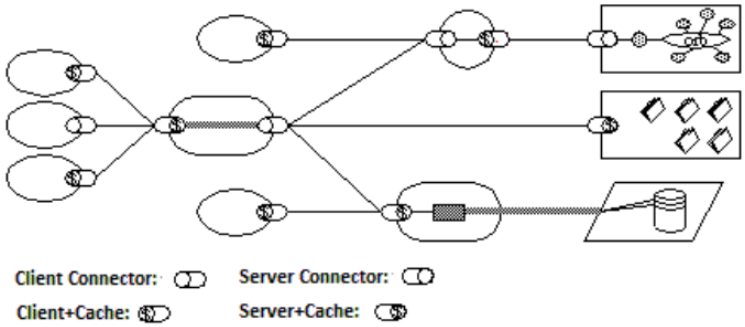



Figure 2.6: Uniform-Layered-Client-Cache-Stateless-Server illustrated by Roy Fielding

## 2.5 MongoDB

MongoDB is a non relational open source database (NoSQL) [30] and was utilized under the thesis to store and retrieve the collected data with the help of REST services, mentioned in section 2.1. This section will present the research and explain why MongoDB was chosen as a suitable database for the mobile application.

### 2.5.1 MongoDB vs SQL

According to DB-Engines the top ten databases were either relational databases or document type databases. DB-Engines is a website created by solid IT, an Austrian IT consulting company that specializes in software development for database-centric applications [31]. The rankings of databases are updated monthly and in July, 2019 the top four were relational databases. MongoDB, a document type database was placed in a fifth place as the most popular [32].

Relational databases are the most common and has a lot of perks. The data is organized as sets of tables consisting of columns and rows. The objects are stored in the tables and are represented as information in the database. One of the beneficial function is that the data can be accessed in different ways without reorganizing the database table. Each row in the table can be assigned with a primary key where each primary key holds a unique identity. The tables also consists of foreign keys which makes it possible to connect a relation with one or multiple tables. The columns hold information related to each row and can be accessed and retrieved all at the same time. Another benefit with relational databases are the use of Structured Query Language, shorten for SQL. With SQL one can manage all aspects of the database such as adding, updating, deleting rows of data and retrieving information from the tables [33]. Using SQL the developer may access many objects with one single command without specifying how to reach the index.

Unlike a relational database MongoDB consists of collections and documents instead of tables and rows. None relational databases such as document type databases are using a single architecture for data consistency. A document in MongoDB has the format Binary JavaScript Object Notation, shorten for BSON. BSON format represents data structure, name-value pairs, arrays and fundamental types. A collection consists of one or multiple documents and can store any type of data. Similar to relational databases a unique primary key has to be assigned to each documents. A collection is the counterpart of a relational database structure.

### 2.5.2 Why MongoDB

MongoDB was chosen because the stored data in a collection doesn't need any relation with other objects. According to official documentation from MongoDB the best practice in designing data models for document type database applications revolves around the relations from object to object. With document type database it's possible to store related data within a single document [34].

MongoDB satisfies the condition to store the data effectively. All collected data for the thesis will be stored in a single collection. Each driven route will be stored as each document for data consistency. Each documents are independent to each other and the data don't have outer relations hence document type database are more suitable for the developed application for the thesis.

## 2.6 ASP.NET Web API

The data which is essential for the mobile application to function will be provided from Optiplan's server. Optiplan's web service was developed with ASP.NET framework and consequently ASP.NET Web API with Transport Layer Security protocol shorten for TLS are a part of the mobile applications authenticate flow.

### 2.6.1 ASP.NET Authentication

To access data from a server an authentication and authorization process are being made in the host. A web request is made when authenticating a user from the mobile application into Optiplan's server. Optiplan's server is hosted through Internet Information Service shorten for IIS. An IIS host uses HTTP modules which consequently forces the request to be HTTP-request type. The authentication process are handled in the host. The host creates an object that represent a security context under the authentication process. The object is called a principal and is attached to the request thread. According to official documentation from Microsoft the developers of ASP.NET Web API [35] the principal contains an associated identity object that holds user information. The host decides if the user is authenticated and returns a represented value for the authorization process to address. The authorization process grants which information the requested user gets access to from the host's server.

### 2.6.2 ASP.NET security protocol

The web service developed by Opitplan runs on .NET Framework 4.6 with TLS 1.2. According to Josh Lake a writer and specialist in security, privacy and encryption for CompariTech TLS is one of the most important security protocol [36]. TLS protects the transmitted data from a request and secure the information through a HTTPS-request. TLS main role is to authenticate a clients request in a connection, check the integrity of data and provide encrypted protection [36]. With TLS protocol the receiving server ensures that the requested client corresponds to the clients integrity. TLS also ensures that the given information has not been modified by the time the server received the information from when the request was sent from a client. TLS also prevent outside sources from accessing the data.

TLS uses handshake method to fulfill the protocols objective. There are nine steps in a TLS handshake according to Cloudfare an American web-infrastructure and website-security company [37].

The first step in a TLC handshake is initiated by the client by sending an initial request message to the server. The request from the client contains information about TLS version from

client side, supported cipher suite and a client random which is composed by a string of random bytes. The second step occurs when the server receives the initial message from the client. In response the server returns a message consisting of SSL certificate, the server's chosen cipher suite and the server random which corresponds to the client random. In step three the protocol confirms if the client is interacting with the owner of the domain by receiving the SSL certificate from the server and verifies the certificate that it corresponds to the issued certificate. In step four after confirmation the client sends a new string of bytes consisting of random characters. The string is encrypted with a public key from the SSL certificate which the server decrypts in step five using a private key that only the server has access to. In step six a session key is generated from both server and client with the given parameters given in the previous steps. The server and client should generate a session key with the same result. In step seven and eight the client and server respectively sends an encrypted message with session key to each other. The handshakes is completed in the last step and communication between server and client proceeds with the help of the session keys [38].

## 2.7 Software overview

Several software development kits (SDK) were used during the development. The main SDKs will be introduced together with a description of how they were used in the project.

### 2.7.1 Xamarin

Xamarin is a free open-source SDK that runs under the MIT license and was used under the project as the core and base of the developed app. Xamarin was the natural choice for the project since it operates on the IDE Visual Studio and delivers native Android, iOS and Windows apps with a single shared .NET code base using C#. With Xamarin the developer have full access to the underlying platform and device, including platform-specific capabilities like ARKit and Android Multi-Window mode [39].

Xamarin is built by multiple APIs which are Xamarin.Forms, Xamarin.Android, Xamarin Mac iOS and Mono. With Xamarin approximately 75+% of the app are shared code with the same language, APIs and data structures across all mobile development platforms. Xamarin.Forms is a UI toolkit for cross-platform development while Xamarin.Android and Xamarin Mac iOS are specific for their platforms. By combining the APIs an application for cross-platform can be developed.

### 2.7.2 Google Maps

With the Google Maps SDK, a visual interactive map based on google maps can be implemented into the application. Handling the access for google map servers, data downloading, map display and response to map gestures are done via API [40]. The SDK also makes it possible to navigate from a given a position to the next. Each navigation makes a call to the Google Maps API which is limited without charge.

It is possible to customize the Google Maps API by implementing a custom renderer for the map control in Xamarin.Forms [41]. The map integration for the app has been implemented using Google Maps SDK and with custom rendering. With a custom rendered map it is possible to navigate from A to B without making API calls using known latitude- and longitude-coordinates. With the given coordinates a designated route with polylines can be drawn into the map visualising how to navigate to the desired destination for the bus driver without making multiple API calls.

### 2.7.3 Oauth2

Oauth2 was used in the thesis to implement safe authentication for the app. Oauth2 enables applications to protect resources from unauthorized access. Google and Microsoft were used as third-party authorization services. The basic flow when using Oauth2 is illustrated in Figure 2.7 [42].

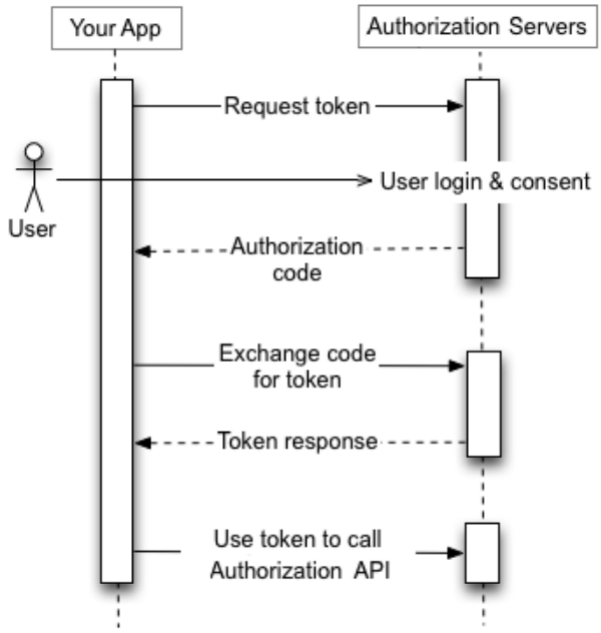



Figure 2.7: Oauth2 flow between client and service

The basic idéa of the Oauth2 is that an authorization starts when a client, the user of the app sends a request to the authorization server. The authorization server may be Google, Twitter, Microsoft, Facebook and etc depending on which API the developer chooses to implement for the user to call. The authorization server receives the request which contains the necessary information of a user. Depending on the information sent the authorization server either validates the credentials or denies the request. With a validated credential the authorization server sends back an authorization code for the user and client with a client secret. The client will use the authorization code to communicate with the authorization server once again trading the client secret with an access token. The process of the grant flow has been divided into two step authorization. By splitting the process the access token does not flow through the application. With the access token the client makes a last call to the authorization server by passing the key within the token.The authorization server holds the same key as the token and verifies the user. The secret will never be exposed to the client or user since the credentials and authorization is being entirely processed by the authorization server [43].

# 3 Implementation

This section describes the development process of the application from user research, prototype development, user test to implementation of the final product. The relevant architecture/solutions that were used for the implementation was carried out in the pre-research described in chapter 2.

## 3.1 Specification of requirements

Before the development it was important to have functionalities planned out to ensure that key components will be implemented. The work carried out in the pilot survey and data mapping laid the foundation for the minimum requirement needed to build the mobile application and it's toolkit to work as intended. The requirements that were appointed after the research and user study were a safe authentication for users to access correct data for the mobile application. The data a user has contained information of pupils and their whereabouts. The information was confidential and can not be displayed to users without permission. Other requirements that were appointed were to display the route for the driver and display which and where a pupil needs to be picked up and dropped off for the mobile application. Another requirement appointed was that the application had to track the bus's location and store the pupils information in real time. The toolkit needed to receive and store the collected information from the application. The toolkit also needed to visualize the driven route compared with the planned route on a map. Further functionalities that were appointed as a minimum requirements were to make it possible for a user to analyze the map by adding information such as where pupils were checked in and out.

## 3.2 Pilot survey and data mapping

This section describes the steps that were carried out to retrieve the relevant data mapped to the users. These steps were essential to build a user friendly experience application. The result from user mapping can be seen in section 4.1.

### 3.2.1 Customer meeting and interview

Before developing the user interface for the mobile application interviews were carried out on representative users. The main users were bus drivers which could be mapped in a few categories. The categories separates the end users in different groups that handles the application differently or has different experiences. By mapping users in different categories the result for user interface and user experience will be more adapted to a wider range of users. Thirteen interviews were carried out on bus drivers with the help of Norrköpings kommun. The users were grouped by age, how accustomed they were with technology and the experience of driving their assigned route. Questions were asked under the interview to gain understanding of the users own perception and skills while driving the route and thoughts about using the application on the field. The questions from the interview can be seen in Figure 3.1.

Frågor till användare

Optiplan AB, Användare #___

**Ålder:**

_____

**Skattningsskalor:**

1. Hur van är du med smartphone och användning av teknik?

Inte alls 1 2 3 4 5 Väldigt

2. Tycker du att det är stressigt vid på-/avstigning vid hållplatser?

Inte alls 1 2 3 4 5 Väldigt

3. Känner du igen alla elever per hållplats?

Inte alls 1 2 3 4 5 Väldigt

4. Det går snabbt och effektiv vid på-/avstigning?

Inte alls 1 2 3 4 5 Väldigt

5. Det är en bra idé med en app som visar rutt och vilka elever som ska på/av vid respektive hållplatser.

Inte alls 1 2 3 4 5 Väldigt

**Frågor:**

1. Hur länge stannar ni max per hållplats?
2. Hur länge väntar ni i en hållplats om inga elever står där?
3. Vet du hur många elever som ska av/på per hållplats?
4. Hur gör ni med kompisåkning (om en person ni inte känner till ska på bussen)?

Avslutande kommentarer/frågor:

Figure 3.1: Questions asked before designing user interface

Beside interviewing users minor questions were asked to municipalities, customers of Optiplan that were interested on the idéa of the application, to receive information on what data should be stored and to be visualized on a toolkit that comes along with the mobile applica-

tion. No user study was made to develop the toolkit since the core of the product was the mobile application.

### 3.2.2 Field study

A field study was carried out to observe the environment of the user for the mobile application. The objective was to increase apprehension of the situation in the field when using the application. By acquiring the perception on how the bus driver worked in the field helped the process of designing the user interface by experiencing different situations that may emerge and therefore see how the user may act.

The field study was executed in two rounds consisting of driving different routes. The field study was conducted with observation and taking notes when the questions were asked in the interview. The routes are grouped into morning route and afternoon route. The two routes differs such that the task in one route was to pick up pupils at respective stations and drop the pupils off at school. The other route was picking pupils up at school and dropping them at respective stations.

The two routes were driven by the same driver. The driver has been in an interview and was mapped into the group called "Uninterested users", see Figure 4.5 in section 4.1.1. The group which the driver was mapped into was the hardest to design. The users in this group were negative to the application and was slightly accustomed with technology and this was taken into consideration while doing the study. The field study indicated that the application needed different solutions regarding the task of check in and check out. The morning route was less stressful checking in small number of pupils at each station and dropping large number of pupils at school. The user had better control in the morning at respective station as to which pupils to pick up. The afternoon route was more stressful when the driver checked in a large number of pupils at school and couldn't analyze which pupils were missing or not.

### 3.2.3 Comparison with similar apps

The idea of mapping similar applications were to summarize a list of do and don'ts. Since there are similar applications the user reviews from those applications can be used to optimize the development of the app developed in this thesis work. The comparison can be read in section 2.1. Three different apps were found and used for comparison. The comparison was carried out by downloading the applications to test them out. All similar functions and each user interface were carefully studied. Reading user reviews were also a part of finding out what users preferred and not. The result from the test and the reviews were being taking into consideration when the design for the user experience was made.

In this master thesis a toolkit was also developed and integrated into Optiplan's website to display and analyze the collected data from the mobile application. The developed toolkit of visualising the collected data was the key to develop an application that differed from other similar existing applications at the time. The idéa of the toolkit and it's functionality was shaped by reviews from the existing applications. Functions that were highly requested was to be able to visualize the collected data from the web. Other idéas for the toolkit were wishes from Optiplan's customers, for example to be able to use the data for resource optimization. A comparison of the driven route with a planned route, the amount of driven km, where pupils are being picked up and dropped of were examples that were requested.

## 3.3 Prototype development for the mobile application

Several prototypes were designed before developing the final product. The process of development followed the pattern of user experience described in section 2.2. The core of designing the interface for this application followed the concept of simplicity described in section 2.2.1. By designing a prototype and having users evaluate the interface the development time was greatly reduced. Before developing the application a design was already tested and ensured that the user understood the navigation behind the application. This chapter focuses on three core functionalities for the application and the design behind the user interface to manage these functionalities. The core functionalities were displaying information about the route, check-in pupils on the bus and check-out pupils from the bus.

### 3.3.1 Prototype 1

The idéa behind the first prototype was to keep the interface simple and have all of the components on a single page for the user to manage. The result for the first prototype is illustrated in figure 3.2. In the first prototype the components were divided by three components. The first component consisted of a map indicating the bus's current position and the positions for respective stations. A list of pupils for each station was designed next to the map. The pupils were listed under their respective stops and next to the pupils were checkboxes for the user to manually decide which pupil to check-in and check-out. The last component consisted of a label positioned under the map displaying which station was next and the estimated time of arrival for the given station.

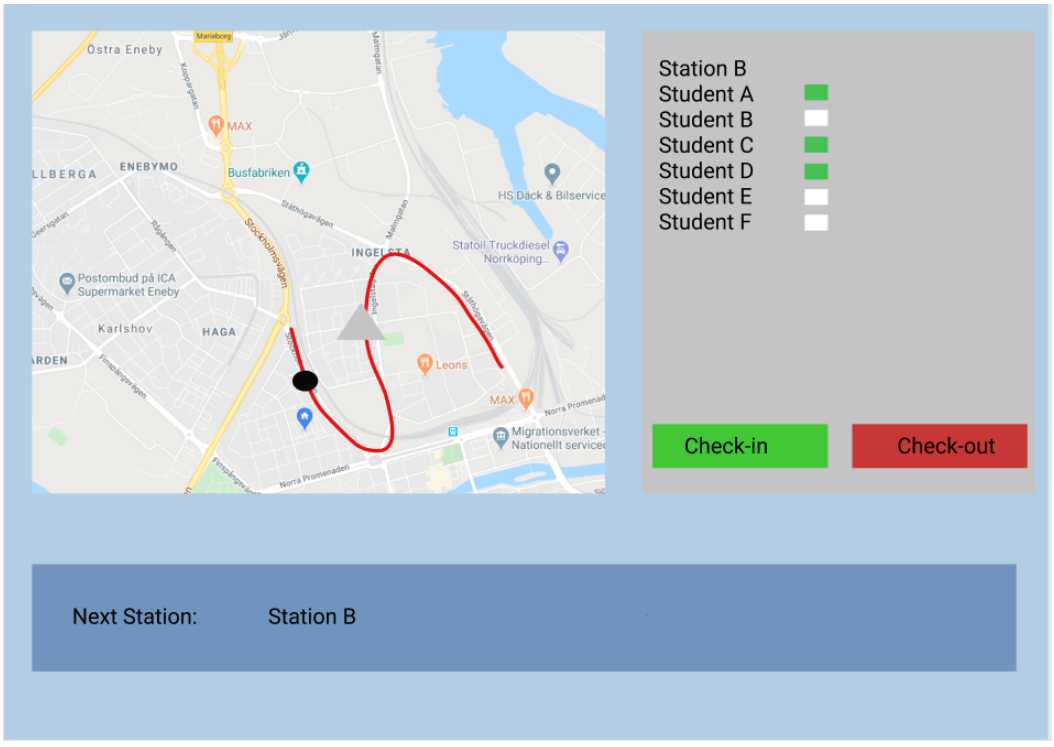



Figure 3.2: First prototype for the application

### 3.3.2 Prototype 2

The idéa behind the second prototype was similar to the first prototype. The solution and it's components should reside on a single page. The difference between the first and second prototype were that in the second prototype the design was taking advantage of the whole screen and not leaving any white spaces. The result for the second prototype is illustrated in figure 3.3. The map was enlarged and the list consisting of pupils were positioned horizontally with the list consisting of stations under the map to make use of the space screen offered. In the second prototype the checkboxes were removed and was instead replaced by clickable labels displaying the pupils names. The main idéa was to minimize the number of actions it took for a users to check in or check out the pupils. The logic behind should know if the pupil has been picked up or dropped off and store the information accordingly.

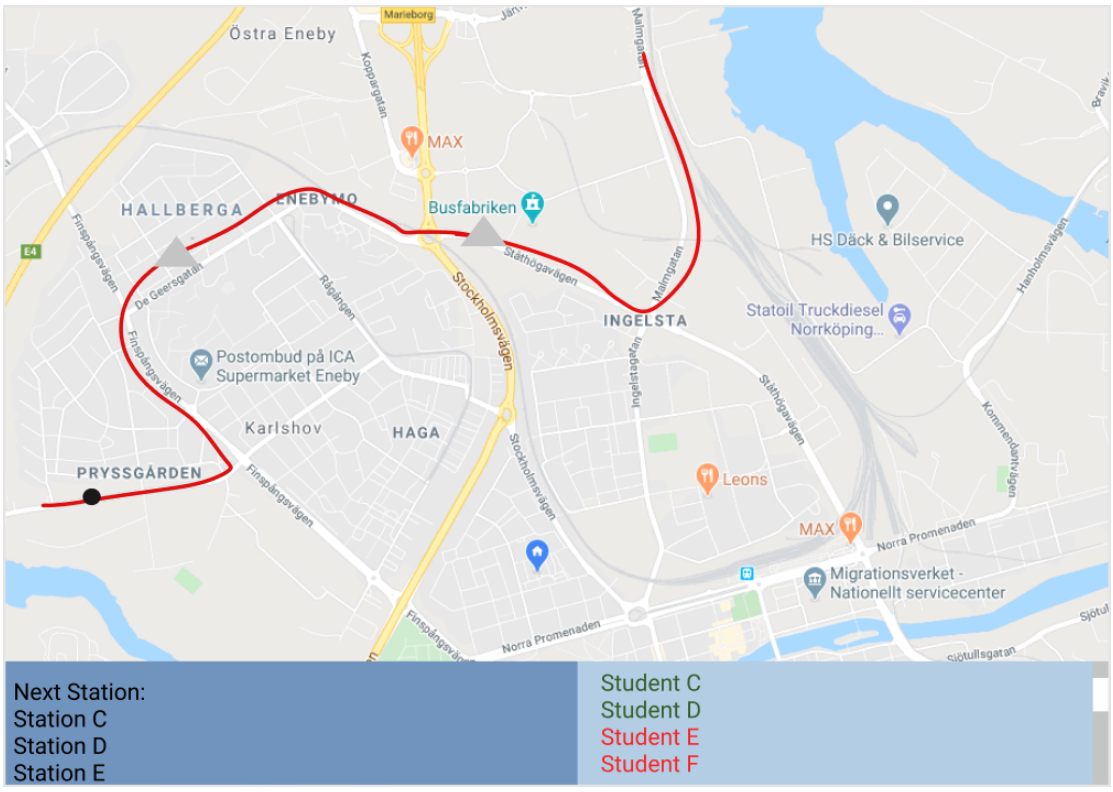



Figure 3.3: Second prototype for the application

### 3.3.3 Prototype 3

The last prototype had a different approach from the first two prototypes. In prototype three the pages were built by tabs and each components represented each tab-page. The reason behind the design for the last prototype was to enlarge check-in and check-out components to make it easier to handle. All three components were important for the application and should be easy to use. By dividing each component into each tab-page reduced the risk of getting confused by the applications different features. The page visualizing the route only consisted of relevant objects representing the selected route. The map page was designed to have a map and a polyline to indicate the planned route and information about where the bus current position. The tab-page that was named "map" was also designed with a table that displayed a list of stations and information about the time of arrival for the route. The concept was applied to other tab-pages for example the page named "check-in" displayed information

about which stations to pick up pupils and which pupils belong to which stations. With the information given the user was able to take actions to check-in pupils in "check-in" page. The page named "check-out" main objective was to highlight which pupils have been checked in. The result is illustrated in figure 3.4.

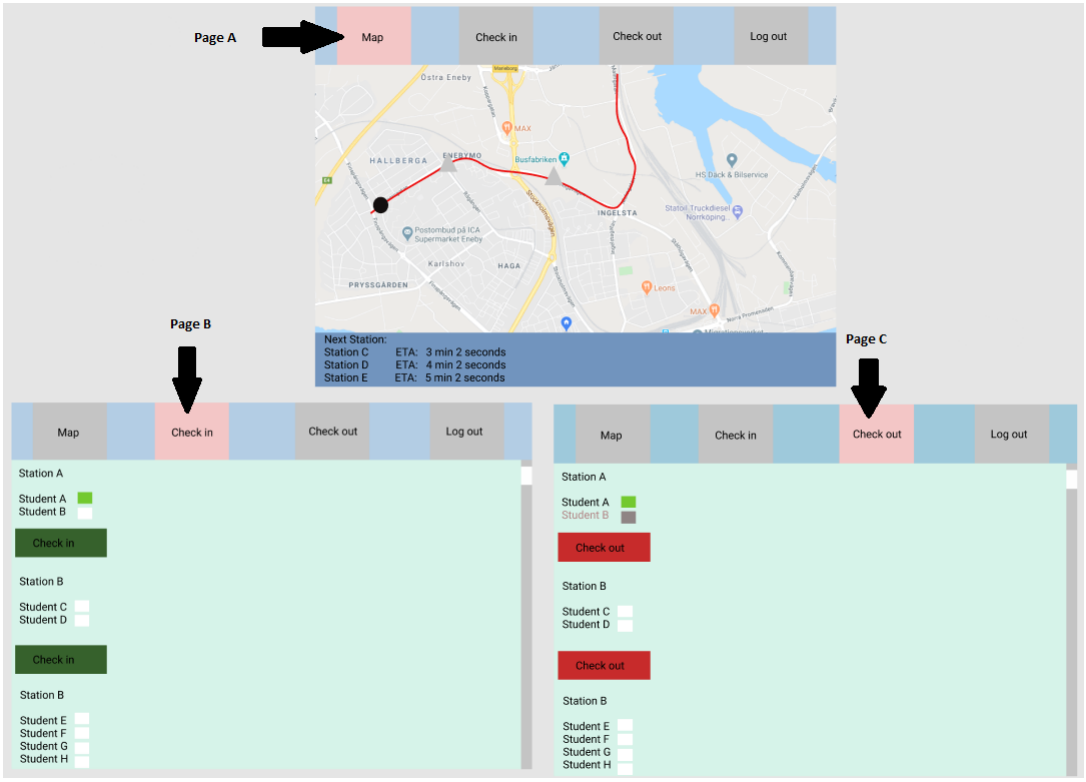



Figure 3.4: Third prototype for the application with multiple tabs where each tab is a whole a page.

## 3.4 Development of the mobile application

### 3.4.1 Database structure

The mobile application had two main features which is to display data from the selected route and store the driven data. The data for each route was given by Optiplan. To fetch the data from Optiplan a REST-request was made to the server. The server received a request and if the authentication was complete the server would authorize the user and respond by passing the correct data to the mobile application. The data consisted of a combination of four collections in Optiplan's database.

One of the collection which responded to the REST-request held information about the authorized user. The user collection held information about which municipality a user had access to and further connects to other collections and an array of objects consisted of vehicles can be retrieved. The vehicle collection held an array of route ids and with each route id the vehicle could connect to which route belonged to the given vehicle. The route collections held information about passengers consisted of pupil objects. A pupil object held information about pupil name, attending school, station to pick up among other information. The route object also consisted of arrays of stations and coordinates. The combined data from the four collections were the building block for the mobile application to function properly.

A new collection was created to store the collected data from the mobile application. Each document in the new collection represented a driven route. The main elements in a document consisted of arrays of pupil objects, location objects, station objects, tracked station objects, information about the user and the loaded route object fetched from the server. The idéa of storing all data in one collection and separate each driven route as each document was to keep data consistency as explained in section 2.5.2. By having all data for a driven route stored in a document, visualizing information or retrieving data from the route resulted in a single query call to fetch the needed data without involving outer collections. An overview of the database structure is represented in figure 3.5.

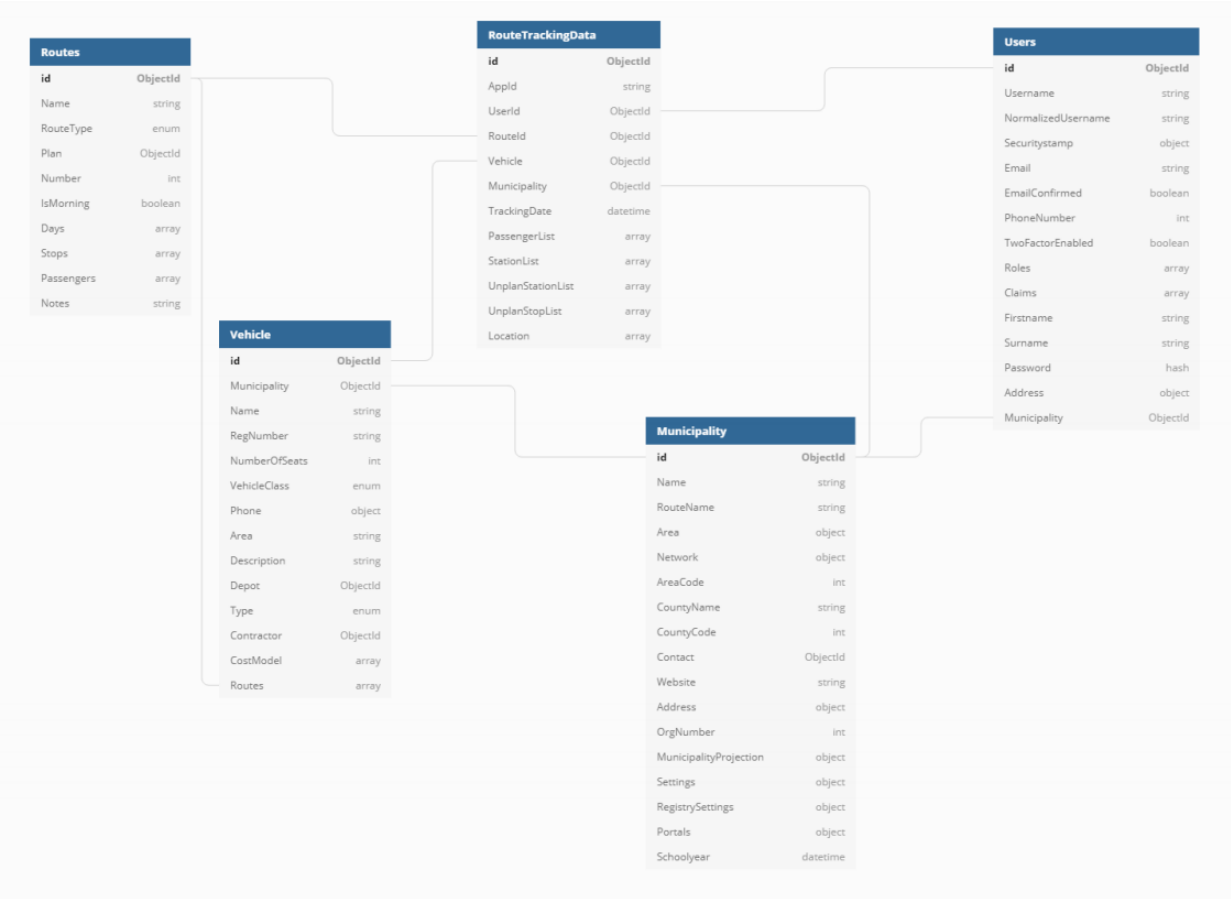



Figure 3.5: Diagram illustrating bindings for RouteTrackingData collection with other collections

### 3.4.2 Login

Two options were implemented to authenticate a user. Each option had it's own security protocol and structure. Option one was using OAuth to authenticate through a third party provider as described in section 2.7.3. Option two was integrating Optiplan's protocol to authenticate a user. Optiplan used ASP.NET core security protocol described in chapter 2.6.1. To communicate with Optiplan's server the application was using REST service as described in section 2.4.1. With REST service the mobile application passed the information entered by the user to Optiplan's server. Optiplan's server retrieved the given data and used the information to authenticate with ASP.NET authorization protocol. If authentication was successful the server returned data which the authenticated user had access to.

The option to authenticate through Optiplan's server was required and targeted users in the category "bus drivers". The targeted bus drivers worked for the municipality that were customers of Optiplan and the information of each bus driver was already registered in Optiplan's server. By authenticating through Optiplan's authentication solution it was easier to implement an authorisation process. Another reason why a solution with authentication from Optiplan's server was required was because not all users were technical and not every user in the targeted group had an email from Google or Microsoft to authenticate through third party provider services.

The second solution to authenticate a user was through a third party provider. Implementing multiple solutions for authentication was also to prepare the application for future development that didn't cover this thesis. The plan was to extend the mobile application to also target the guardians of the pupils. The plan from Optiplan for future development for the mobile application was to display the whereabouts of the pupils for their respective guardians during the bus ride. The users in the group "guardians" would have resulted in a larger number of users than the group "bus drivers", consequently by implementing authentication solution with third party providers the user could login with Google and Microsoft email account. By having multiple authentication solution the application didn't force a user to create an account in Optiplan and the application was made more user friendly.

The login page had one purpose which was to authenticate a user. Following the rules described in section 2.2.1 the components for login page were designed to only consist of three elements. One text field to enter a username, one text field to enter password and one button to confirm the entered information. The users only needed these three components to handle the task given for the page. In this thesis the application also offered a solution for a user to authenticate through a third party provider which consequently forced the interface to have an extra button.

### 3.4.3 Map

Google maps API was chosen to integrate a 2D map into the application. A custom renderer was implemented into the google maps SDK as described in chapter 2.7.2. With the implementation of a custom renderer for google maps API it was possible to draw a polyline on the map indicating the planned route for a user. A polyline is a number of line segments connected next to each other to create a chain and to draw the polyline into the map the application retrieved the route information form Optiplan's server by REST-service. The route object held an array of location for the route. The location object consisted of arc-restrictions, longitudinal and latitudinal coordinates. By placing each element in the location array on google maps a polyline was created with the help of the given information. Using the same logic pins could be added with the help of custom renderer to the map. A pin was used to display the bus current position and by placing multiple pins on specific coordinates it represented respective stations incorporated into the route.

By using google maps SDK it was possible to retrieve the bus current position with the help of the device's GPS. The mobile application was implemented to retrieve the current position of the device every five seconds and store the information parsed into coordinates. Every minute a request was sent to Optiplan's server to store the data consisted of coordinates into the database. By fetching information from Optiplan's database and by iterating to the last stored coordinates from the list it was possible to use the information to decide the bus's position with one minute delay. Further the whole list was used to create a line segment to draw a polyline that represented the route the bus has driven.

### 3.4.4 Check-in

The solution for checking in pupils was implemented for bus drivers to carry out. The reason behind the decision was the possibility of pupils cheating by checking in a friend if they had access to the mobile application, therefore only bus drivers should be able to manage the check in function. Consequently a key component to implement in check-in page was to develop a user interface to help the bus drivers check-in the pupils swiftly and accurately. The bus drivers had a strict schedule to maintain therefore the users can't spend minutes at each station to check-in the pupils.

The check-in page consisted of a station list and a pupil list. The station list had information about which stations belong to the route, their names and locations. The pupil list contained information about pupils for the route, pupils names and which stop the pupils are stationed at. By sorting the pupils by their respective stations and displaying the stations in ascending order for check-in page it made it easier for the user to maintain control over the situation. If the bus currently was positioned in station B that comes after station A, the user would already have gone through the list of pupils displayed in station A. This removed the need to scroll through the application to find the station the bus currently is located. The pupils were sorted by alphabetical order to help the user navigate to the desired pupil without needing to look randomly. A checkbox was positioned horizontally to each pupil and was being used to decide which pupil to check in.

At the end of every station displayed in the application, two buttons were implemented. One button was being used to store the pupils check in statuses at the given station. The other button implemented was to cancel the action and is disabled by default. In order to display the information of pupils statuses in real time it was implemented to send a request to Opiplan's server after the confirmation button was pressed. The request added the information to "RouteTrackingData" collection in Optiplan's database as mentioned in chapter 3.4.1 and could be fetched to display the information. The cancellation button was enabled when the confirmation button has been pressed. The cancellation was implemented to correct human error that may occur when using the application. By pressing cancellation button a request was sent to Optiplan's server to undo the actions made with the confirmation button. An example is if a pupil has been wrongly checked in or a pupil has been forgotten the user may correct the mistake. To reduce the number of request to the server the pupils statuses are being updated in batches, for each station.

An alternative solution to speed up the process was to ignore the checkboxes and check in the pupils directly by pressing the pupil names as designed in chapter 3.3.2. The idéa was not implemented because it was tested and decided that prototype two has flaws and not user friendly for color blinds. The amount of time saved without the checkboxes were not great to ignore the user interface experience. Lastly a checkbox was implemented for each station and placed horizontally to the station name to help the user check in pupils faster. The function implemented checked all pupils for the given station in an instant. If a station has 14 pupils and the driver has estimated that the number of pupil matches the station the bus driver can check all pupil in two clicks compared to 15 clicks.

### 3.4.5 Check-out

The concept for the check-in page was incorporated into the check-out page. Components and logic for solutions in check-in page was reused for check-out page. By reusing components and by designing the check-out page similar to check-in page the risk of confusing the user was reduced.

The advantage with check-out page was the possibility for the user to only focus on pupils that has been checked in. By taking the advantage into consideration it was possible to speed up the process for different actions in the page. Similar to check-in page the stations were listed and sorted by the planned route in ascending order. Each station displayed which pupils belonged to which stations. All pupils in check-out page are grayed out and disabled by default. These pupils will be enabled only if they have been checked-in and consequently it was possible to implement a confirmation button to check out all pupils on the list for the given station in an instant. This removed the need to select which pupil to check-out and by default the pupils will be positioned to check out at their planned station. To decide which pupils has been checked in a request will be sent to Optiplan's server to retrieve the stored

data. The data decides which pupil is currently on the bus and eliminates human error by forcing the user to manually choose which pupil to check-out.

The check-out page required another solution. When driving pupils home from school the pupil have a choice to leave the bus at a friends station. Consequently the user needed a solution to check out pupils correctly and not by the default value. A new button was added for each station. By pressing the button a list of pupil names will appear for the user. The list of pupil consisted of pupils that has been checked in and pupils that are not planned for the clicked station. The user can then select which pupil to add to the selected station. The user interface is dynamically implemented and the pupil name will be added to the new station. The pupil name in the original station will be grayed out and a text will appear next to the pupil indicating which station the pupil was selected to be checked out of. Pupils that have been added to another station will have a button horizontally to their name. This button will remove the pupil from the selected station and re-add the pupil to it's original station.

## 3.5 Development of the toolkit to visualize the collected data

To easily access the stored data from the mobile application a single page application was developed and integrated in Optiplan's website. The implementation was written in .Net MVC framework with C# the same language and framework as the website Optiplan has developed. To analyze a desired route the user needs to indicate which data to be displayed. A selection screen was implemented for the purpose. Each collected route has a vehicle and date that connects with a specific route. The selection screen was implemented with three drop-downs to help the user select the desired route. The first drop-down is populated with a list of vehicles. Depending on the selected vehicle the second drop-down displays which routes belong to the vehicle. The selected route generates a list for the third drop-down which consists of date and time for each driven route.

When a route has been selected the user will be redirected to the single page application. The purpose of the toolkit was to give the user an overview of the driven route. This was solved by having an interactive map as a central part of the single page application where most of the stored data was drawn on the map. The interface for the page was designed with having the map embedded in a content container with a header and footer. The header and footer was needed to display information that couldn't fit in the map. The route name, date and the driven path in kilometers for the selected route is displayed in the header. To display the map Microsoft Bing Maps API was used. The planned route, the driven path and icons representing pupils geographical position at pick up and drop of stations were implemented to be drawn on the map to visualize the stored data for the user. Displaying too much data can sometimes confuse a user, this was solved by giving users free choice by having checkboxes implemented in the footer. The checkboxes controls which data to be displayed and are all check by default.

The drawn polyline for the planned route is visualized by a red thick line and the drawn polyline for the driven path is visualized by a green thin line. This was made to highlight an overlap if the planned route and the driven path are the same. To indicate pupils an icon represented by a figure with a head and half a body was used. A text is displayed under the icon representing number of pupils that were picked up or dropped of at the given position.

A table and a content container were also implemented in the single page application. The table was needed to display information in text format that wouldn't have been a good visual representation in the map. The content container was implemented for the user to manage

the route. Features that could be accessed with the content container are exporting the driven route to geojson and deleting the selected route.

## 3.6 User test

A user test was performed on the mobile application. The user test was performed to collect feedback to improve the product. Bus drivers are the end users for the mobile application and consequently the most important group to test the application on. Ten bus drivers tested the application in the field and each bus driver tested the application during the period of two weeks time. The testers were in the age span of thirty to sixty years old. An interview with the testers was carried out after the test period. The questions asked in the interview can be seen in the questionnaire illustrated in figure 3.6. The result from the user test is represented in chapter 4.4.

Enkät för användartest av första prototyp

Optiplan AB, Testperson #___

**Skattningsskalor:**

1. Det var lätt att förstå hur appen fungerar

Håller inte med 1 2 3 4 5 Håller fullständigt med

2. Det var enkelt att lära sig

Håller inte med 1 2 3 4 5 Håller fullständigt med

3. Det går snabbt att checka på/av elever

Håller inte med 1 2 3 4 5 Håller fullständigt med

**Frågor:**

1. Vilken storlek på surfplattan är optimal för att det ska vara lätt att hantera appen?
2. Vilket alternativ är bättre?
   a) Markera elever och sen bekräfta att de har checkats in/ut?
   b) Markera elever och de checkas in direkt?
   Motivera:
3. Får barnen hoppa av bussen innan de har anlänt till skolan? Om ja, är du beredd på att checka av eleverna manuellt vid varje station och hade det varit möjligt?
4. På hemvägen får barnen hoppar av stationer innan deras respektive station? Om ja, är du beredd på att checka av eleverna manuellt vid varje station och hade det varit möjligt?
5. Vad var bra och vad var mindre bra?
6. Vad kan förbättras? (vad som saknas av innehåll och funktioner)

Avslutande kommentarer/frågor:

Figure 3.6: Questionnaire for user test.

# 4 Results

This chapter will present the result for the mobile application, toolkit for visualisation and user test.

## 4.1 Pilot survey and data mapping

This section will present a summary of pilot survey and data mapping that were carried out before the development. The result from pilot survey and data mapping were a crucial part of designing the user interface.

### 4.1.1 Customer meeting and interview

The goal for customer meeting and interview were to collect information from the customers and potential users to construct the mobile application accordingly. A part of the result can be seen in figure 4.1 where the answers for the questions with a rating scale are illustrated in graphs.

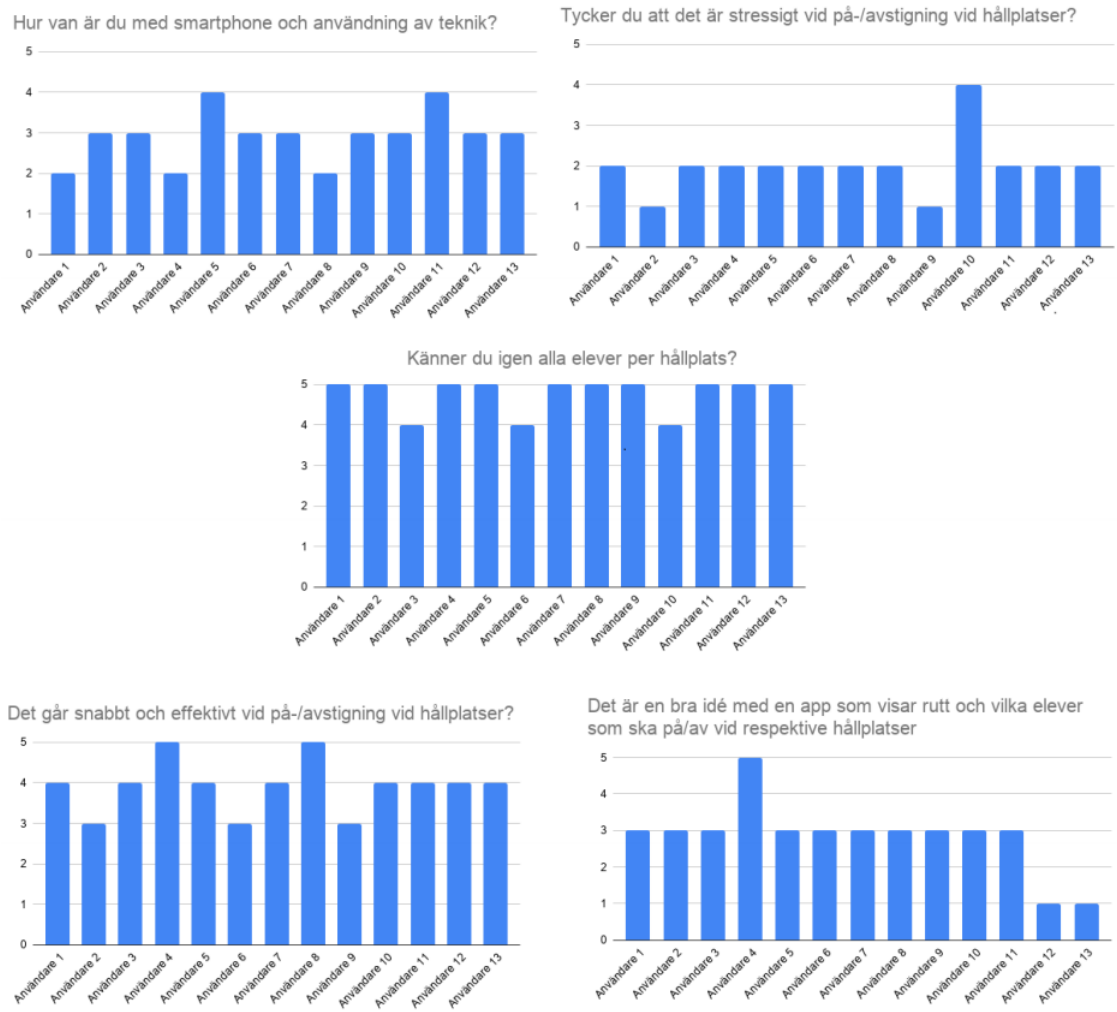



Figure 4.1: Result from the interview illustrated in graphs.

The questionnaire also consisted of questions with free answers and each questions were summarized with the most common answers from the 13 interviewed bus drivers. For question one, "For how long does the bus stay at a station" the most common answer was one minute. It was also stated that depending on the number of pupils at a given station it could take up to two minutes. The second question, "For how long do you wait for a pupil in each station" the answers were mixed but every interviewed bus driver agreed on the maximum waiting time is around three minutes. The third question was asked to find out about if the bus drivers would be able to take advantage of the mobile application if the application displayed a list of pupils for respective stations. The result was that most bus drivers didn't know the exact number of pupil at a given station and already uses a list of pupils printed on a paper. They believed a digital list would be helpful to be able to gather all information in one place. Last question was about how the bus drivers handle situation where a pupil brings a friend along that are not registered to be transported by the bus. The common answer was that the drivers trusts the pupil and no validations were made. The feature to display an unknown pupil for the bus driver was a planned feature to implement for the master thesis and answers confirmed that the feature will be helpful.
With the result from the interview and customer meeting an effect map was constructed. Effect map was constructed to map the users to find out the goal and reason why the mobile

application was necessary as mentioned in section 2.2.3. The result for effect goal for the application was collected mostly from the customers wishes and finding out features that would be helpful for the users. The result for the goal can be seen in figure 4.2.

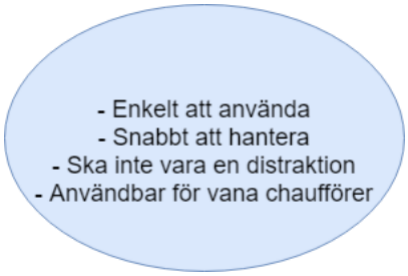



Figure 4.2: The mobile application mapped into effectgoal.

The first group of users were called "inexperienced users". The result for group one is illustrated in figure 4.3. This group was created for users that are not experienced with smart phones and mobile applications in general. The task was to create an application that will work for the given group. To achieve the goal the idéa was to create a user friendly application in which the users can learn and use without a manual or help. To improve the user experience further the application needed to respond quickly, for example if a user triggers an action the loading time has to be short to not confuse the user with waiting time.

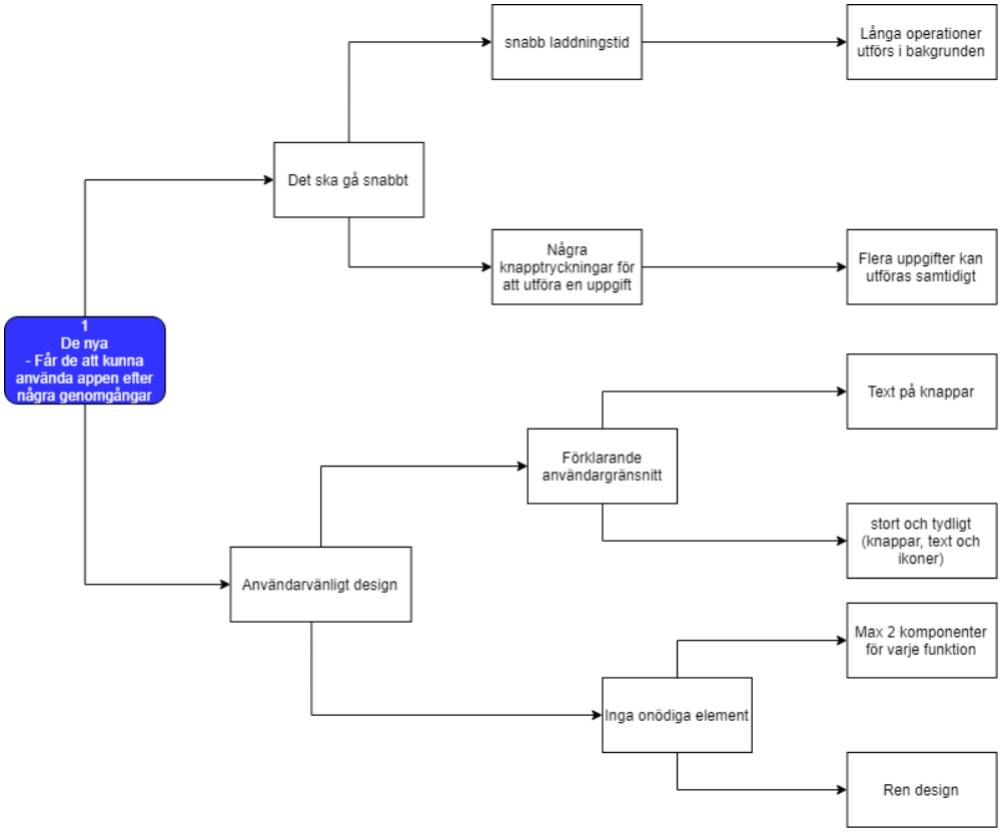



Figure 4.3: First group mapped as "New users".

The second group of users were called "experienced users". The result for group two is illustrated in figure 4.4. The second group is well known to technology and uses smart phone and mobile application on a daily basis. The challenge for this group was to create an attractive application since users in this group naturally have a higher standard from using other mobile applications. To achieve the goal two important factors needed to be included. The first factor is a well design user interface. The feel and look of a mobile application is big part of the total impression and might be the deciding factor if a user prefers the application over a similar application. Another crucial factor is that the app should follow modern architecture and have modern components since the application needs to be competitive with eventually new similar applications.

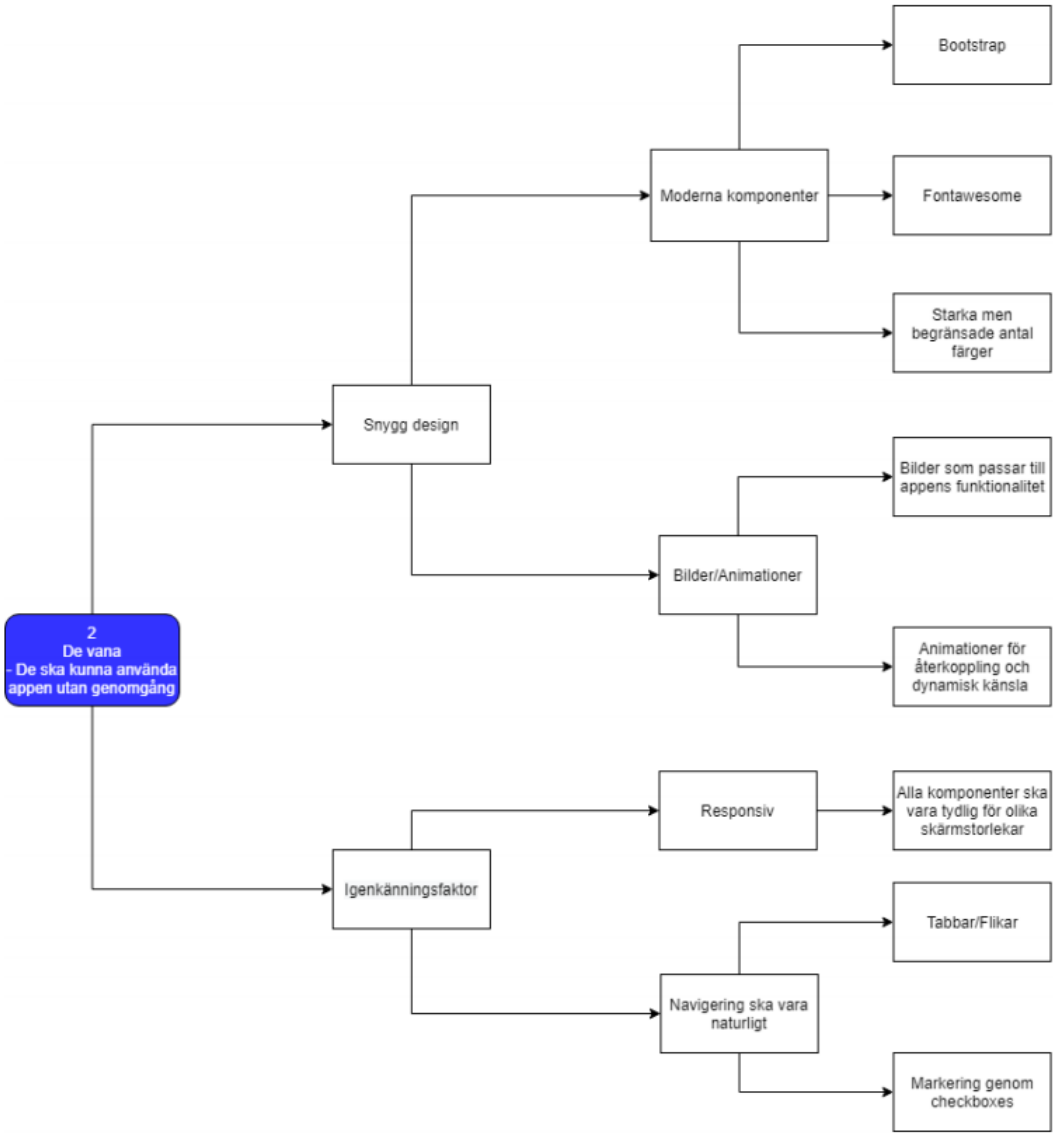



Figure 4.4: Second group mapped as "Experienced users".

The last group of users were called "uninterested users". The result for group three is illustrated in figure 4.5. This group consisted of users that wasn't interested on having a mobile application to check in and check out pupils. The main reason was because the users didn't feel that the application was necessary and couldn't see why it could be useful. The challenge for this group was to implement useful features for a group of users that have fully mastered their driven route and recognizes all pupils. To solve the problem every important data will be displayed on the mobile application to give the user a visual representation. No matter knowledge a visual impression will enhance the experience and triggers memories when needed. The application will visualize the bus's current position in a map, visualize the number of pupils for respective station and order them accordingly to the planned route.

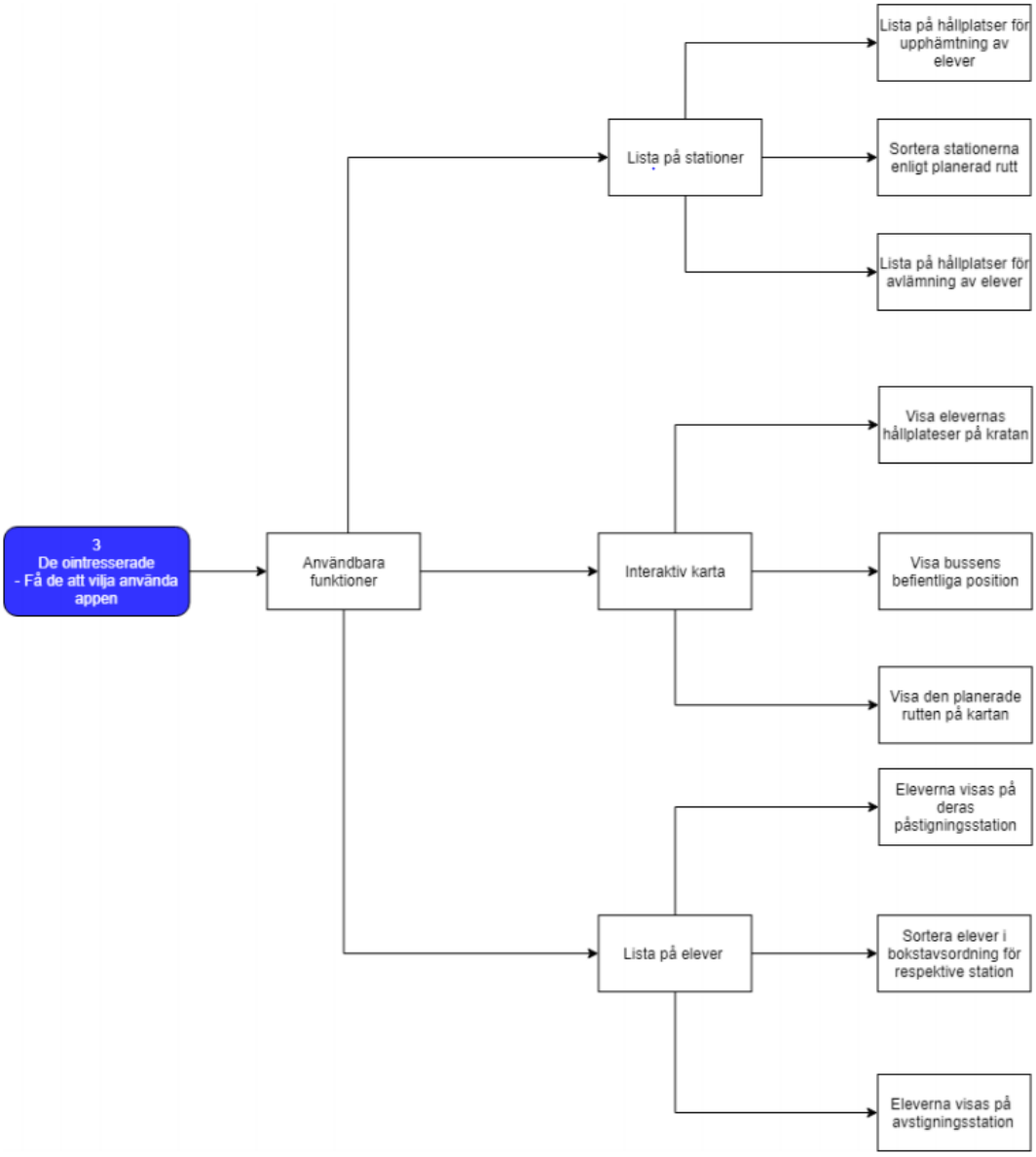



Figure 4.5: Third group mapped as "Uninterested users".

### 4.1.2 Field study

The result made from the observation in the field study was that the bus driver which is the user for application had couple of minutes at each station to check in pupils in a safe environment. The result could slightly differ depending on what type of route the bus drove. The routes are divided into morning and afternoon routes as mentioned in section 3.2.2. The difference between the two types of routes were number of pupils the bus driver had to check in at the same time for a given station. The number of pupils were highest at school in afternoon route and consequently the afternoon route was hardest to find a solution to quickly check in pupils. The result from field study showed that the users didn't have much time at each station and needed a solution to check in all of the pupils quickly and accurately.

### 4.1.3 Comparison with similar apps

By comparing similar applications a list of pros and cons were made. The list was useful for designing the prototypes. A couple of elements that were good were taken and modified for the user interface. User reviews from the similar applications were taken into consideration and elements where the users complained were avoided.

Both "Versatrans My Stop" mentioned in section 2.1.1 and "Here comes the bus" mentioned in section 2.1.3 have a check in solution where the user needs to select each pupil one at a time. The complaints were that the task is time consuming in a station with a lot of pupils. This will be avoided to help the bus driver speed up the process when using mobile application developed in this master thesis. Some of the application had tab-page navigation to separate the different tasks and therefore divide the application into different parts. By dividing the task into different pages the user have lesser risk of getting confused. The choice of colors were also mentioned in user review. In a user perspective too many colors in a single page will ruin the experience for the user by taking away the attention from the application and it's features, this will be avoided in the design.

## 4.2 Prototype

Three prototypes were designed with the mock-up model as presented in section 3.3. The prototypes have no real functionalities and were designed to give an overall visual of the product. The prototypes were designed with the targeted groups in mind. Prototype 1, described in section 3.3.1 and prototype 2, described in section 3.3.2 were composed very similarly with the same components in one page but arranged differently. Prototype 3, described in section 3.3.3 was designed differently, the components were divided in multiple pages instead of one page.

Before choosing which prototype to implement for the mobile application a none formal user test was performed. The user test was performed by friends and employees at Optiplan. Each testers got to see the prototypes and got a brief explanation of the mobile application. The testers would analyze the user interface and explain what they thought each button and information displayed on the application meant. The feedback from testers were summarized for each prototype. A comparison were then made to see which prototype had the most user friendly interface.

The result was unanimous that both prototype 1 and 2 described in section 3.3.1 and 3.3.2 had too much information in one page. A direct consequence with too much information in one page would be to fit all information in a small screen such as a tablet or a smartphone which the application was intended to be developed for. The third prototype described in section 3.3.3 was chosen because most users thought it was less confusing interface and had better

design overall. The third prototype was implemented and the result is presented in chapter 4.3.

## 4.3 Development of final product

This section will present the final product developed in the master thesis.

### 4.3.1 Login

Opening the mobile application the user is presented with a log in screen as seen in figure 4.6. Without an account the user won't be able to advance. The application does not offer a sign up page and a user is created by Optiplan in their system. The login page offers everything a user needs to login and emphasis which company the product belongs to with a big logo in the center.


Figure 4.6: Entry page for a user

Entering a username and password a request is made to Optiplan's server. The authentication process is explained in chapter 3.4.2. If the authentication fails a popup with a message will be displayed for the user. If the user has entered correct information the user will be redirected to selection page, presented in figure 4.7.

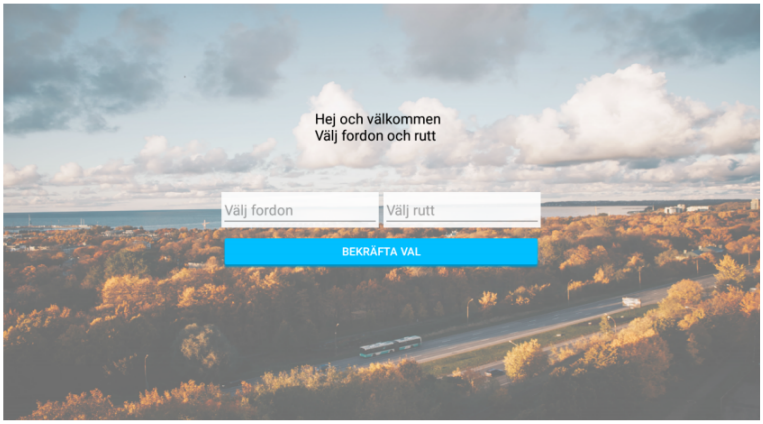



Figure 4.7: Selection page

In the selection page the user is presented with two drop down lists, a confirm button and a background image representing the theme for the application. The first drop down list is populated with different vehicles the user has access to. The second drop down list is by default empty. By selecting a vehicle the second drop down will be populated with routes relevant for the chosen vehicle. The confirm button will process the given information in the drop down list. If the user has not selected a vehicle or route a error message will be prompted telling the user to select a vehicle and a route. A successful confirm will process the given information and redirect the user to the main page displaying the data for the selected route.

### 4.3.2 Map

The main page is built by tab bars and the map page is by default the first view. The map page is presented in figure 4.8.

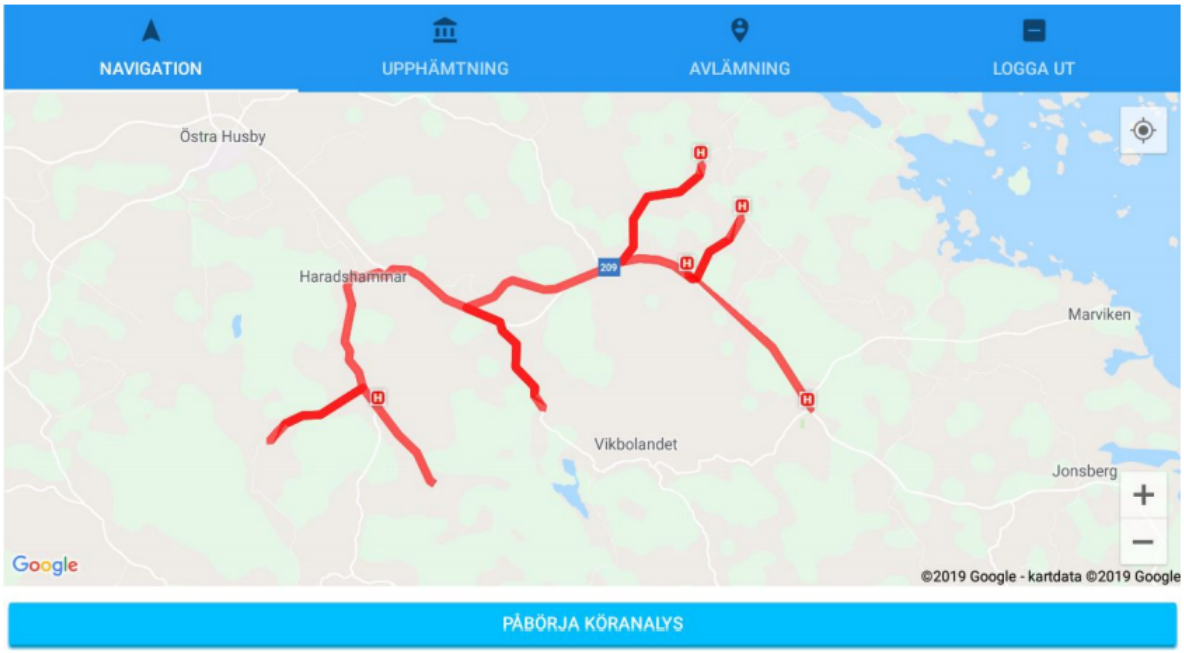



Figure 4.8: Map page

The map page is built by two components, consisting of an interactive google map and a button. The map is zoom-able and movable. The map also displays a drawn path illustrating the planned route for the chosen route. Icons with the letter H are drawn in the map to indicate the planned positions for respective stations. Pressing the button in the page a text-box appears asking if the user wishes to start analyzing the chosen route. If the user presses yes the button will disappear and the map will be enlarged to cover the whole page, saving only the header. An icon will also appear in the map representing the bus current location. When the button has disappeared the mobile application will log the bus's current position every five seconds. For the mobile application to work the user needs to accept and activate location service on the device. The first time a popup will appear asking for permission and if the location service is turned off the application will prompt a message telling the user to turn on location service.

### 4.3.3 Check-in

The second tab bar is the check-in solution page. The check-in page is by default blank with a message telling the user to activate the analyze as described in section 4.3.2 to start all

functions.

After activation a list of stations and pupils will be loaded and drawn in the page. The stations will be listed and sorted by the routes planned order. Next to each station a checkbox is implemented. By checking the box all pupils listed for the station will be checked. The list of pupils are ordered in alphabetical order for their respective stations. Next to each pupil name a checkbox is implemented to check in pupils separately. Color schemes are being used to give the user feedback. A pupil is by default not checked in and the name is colored red. If the checkbox is checked the color will be changed to green indicating the pupil has been chosen to be checked in. A grayed out pupil is disabled and a text next to the pupil name is displayed to notify that the pupil has been cancelled for toddy's trip.

At the bottom of each station list two buttons are drawn horizontally to each other. The first button is blue and has a text explaining to the user if pressed the selected pupils will be checked in for the station. The second button is red and has a text explaining if pressed the checked in pupils for the station will be reverted. The second button is by default disabled and is enabled when the first button is clicked. When the first button has been clicked it will be disabled. When the second button is clicked it will enable the first button and disable itself. The result for check-in page can be seen in figure 4.9.

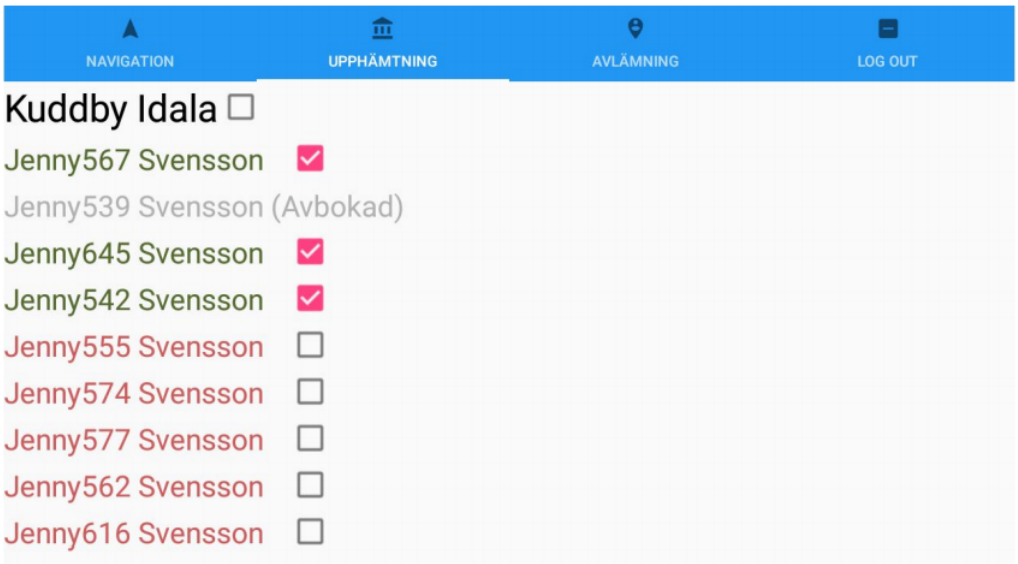



Figure 4.9: Check-in pupil page

### 4.3.4 Check-out

The check-out page is similar built and designed as the check-in page. Identically to check-in page the page is by default blank and needs to be activated to display the content. Similar to check-in page, check-out page has a list of station and pupils sorted by the routes order. Check out page has two buttons for each station. One button confirms the checkout statuses for pupils, and a button to regret the decision if human error occurs. The result for check-out solution is shown in figure 4.10.

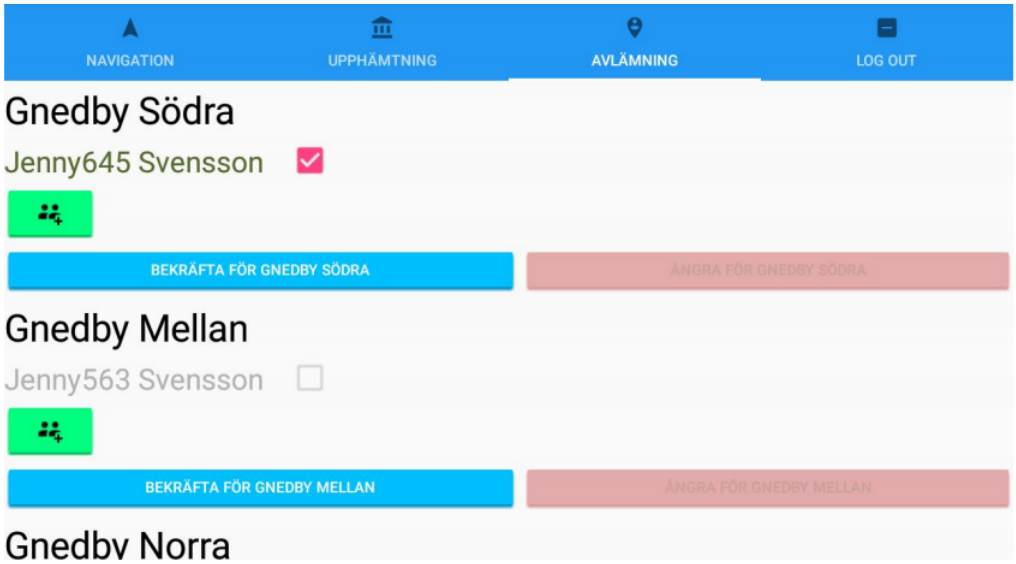



Figure 4.10: Check-out pupils page

The check-out page is dependent to the check-in page. By default the pupils listed on check-out page are disabled and grayed out. The pupils will be enabled to handle in check out page once the user checks in and confirms the pupil in check-in page. Checked in pupils will be displayed with a green color in check-out page and the checkbox will be checked. This is to save time for the user because the pupils are confirmed to be on the bus and they are ready to be checked out at their respective stations. The difference in check-out page compared to check-in is that the checkbox next to station name are not implemented. An extra button is implemented at the bottom of each station before the confirm and regret button. The button has an icon representing pupils, as seen in figure 4.11.

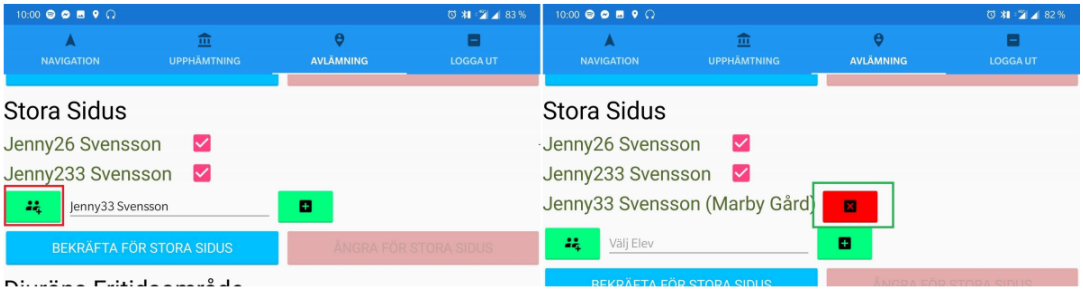



Figure 4.11: Check out solution for pupil in a different station

By pressing the button the user activates the function to check out pupils at another station. If the button has been clicked an input field and a extra button is drawn on the application. This situations happens and is needed if a pupil follows another pupil home. The main function for the checkboxes next to each pupil is to keep a pupil in the bus. If the check out station for a pupil is after the planned station the user can select to keep pupil on the bus. By pressing the input field a pop up appears with a list of pupils that has been checked in and not currently being listed on the station. The user can pick a pupil from the list and by pressing the plus button the application changes the pupil's station. The mobile application will draw the selected pupil for the given station and adds a text to indicate the pupil's original planned station. Next to the pupil a red button will be drawn to remove the selected pupil from the new station if human error occurs. The function is illustrated in figure 4.11.

### 4.3.5 Logout

The last page in tab-bar navigation layout is the logout page. The page offers two alternative for the user. The user can logout from the application and consequently ends the analyze for the route. Option two is to change current route to a new route without needing to login and logout. This will be helpful if the user has selected the wrong route or has another route upcoming right after the current route. The result for the page can be seen in figure 4.12.

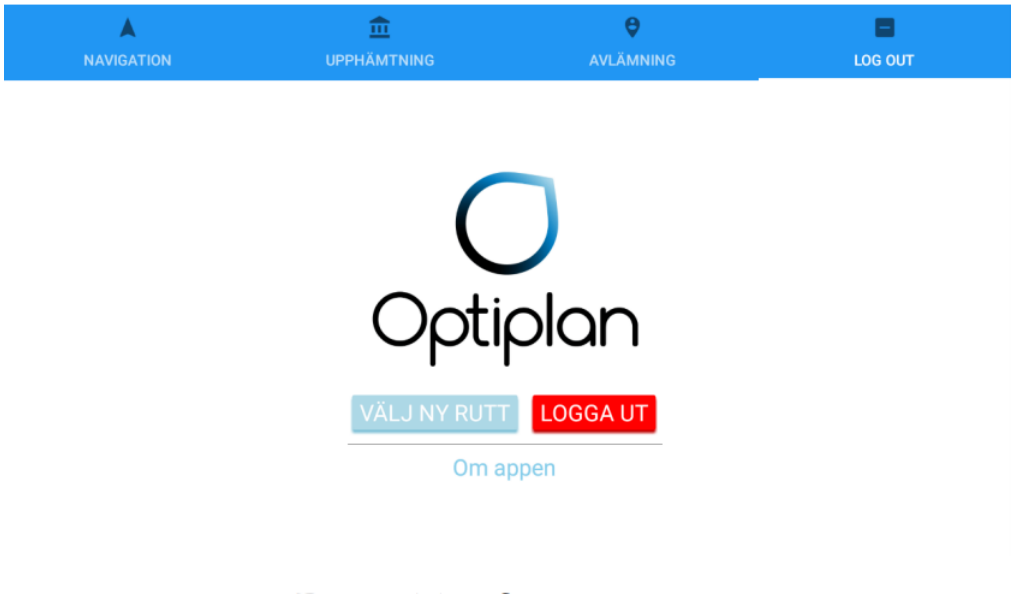



Figure 4.12: Sign out page.

### 4.3.6 Toolkit to visualize the collected data

A toolkit to visualize the collected data is integrated into Optiplan's website. The data is displayed with the help of a map and a table. One of the main feature with the mobile application was to use the data for resource optimization. One optimization was to find out the number of pupils that received the service to be taken by bus to school utilizes the service. The toolkit displays raw data collected from the mobile application printed in a table for a chosen route. The result for the table is shown in figure 4.13. The table content consists of each pupil's statuses for the driven route. Information shown for each pupil are their names, which station they were picked up and dropped off from and at what time. The table is interactive and the header content is clickable to sort the pupils accordingly to the selected field.

This gives the user the opportunity to sort the table after pupil name or check-in time. A pupil that was supposed to attend but didn't is marked with a hyphen in the table.

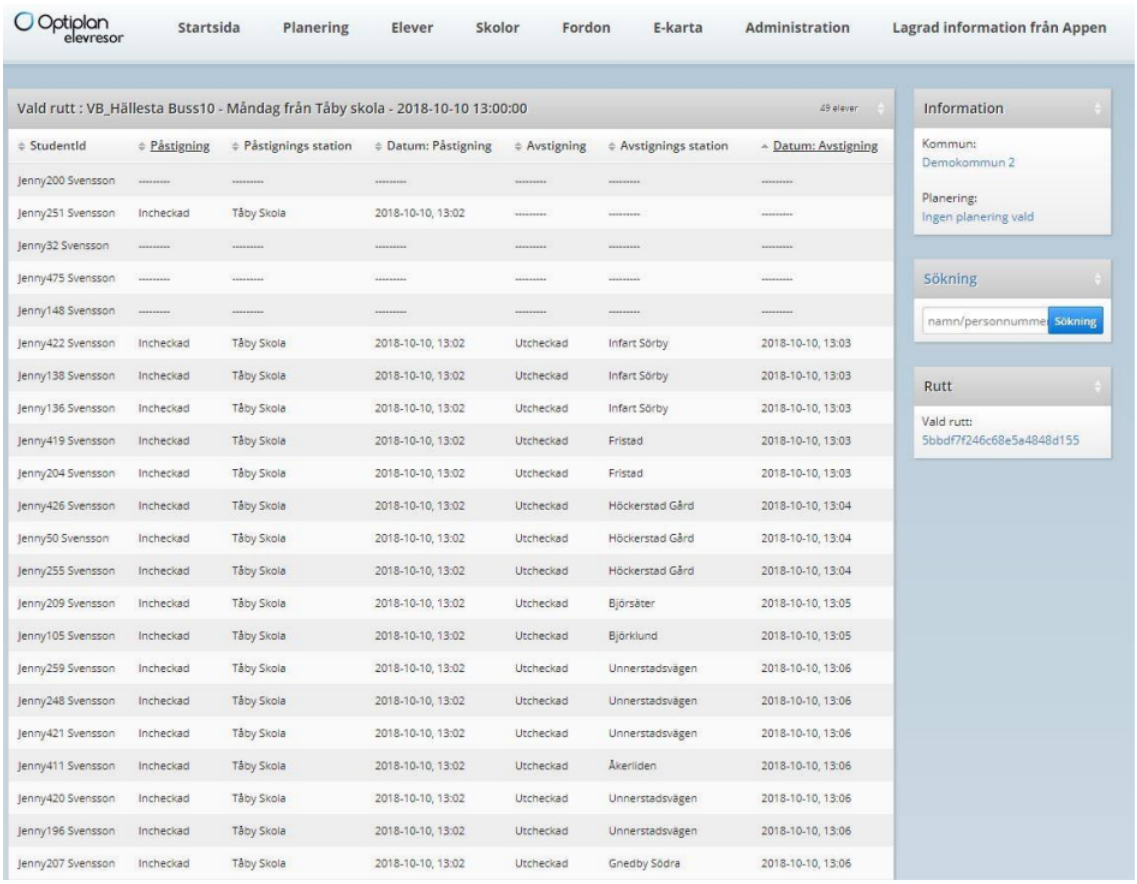



Figure 4.13: A list of pupils regarding their status

The second main feature was to confirm if the bus drivers drove the route according to the planned path and if the pupils were picked up at the planned stations. The map integrated into the toolkit displays the data visually to help a user understand the driven route without interpreting the collected raw data. The toolkit for the map has checkboxes a user can choose from. Each checkbox has a label indicating what information will be drawn on the map if the checkbox is ticked. The options are polyline of the planned route, polyline of the logged driven route, icons of planned stations and icons of check-in and check-out locations of the pupils. By selecting multiple checkboxes the user can compare the collected data in a visualization. If both checkboxes for displaying the planned and the logged route are checked a direct comparison can be made to see if the driver has driven the route accordingly to the planned path or if the driver has been driving another path. The polyline for the planned route is a thick red line and the logged driven path is a thin green line. The planned and driven route have different colors and thickness to help the user compare if both polylines crossed path. By ticking the checkbox to display stations, icons will be drawn on the map at the location where the stations were planned. The icons for stations are visualized by capital letter H. By ticking the checkbox to display pupils picked up and dropped of locations, icons visualized by a pupil head will be drawn at respective positions on the map. Since the driver can check in and check out multiple pupils at the same location the map will only draw one icon at the given place indicating the spot where the pupils were picked up. The icon has a label of the station name and a number which represents the number of pupils that were

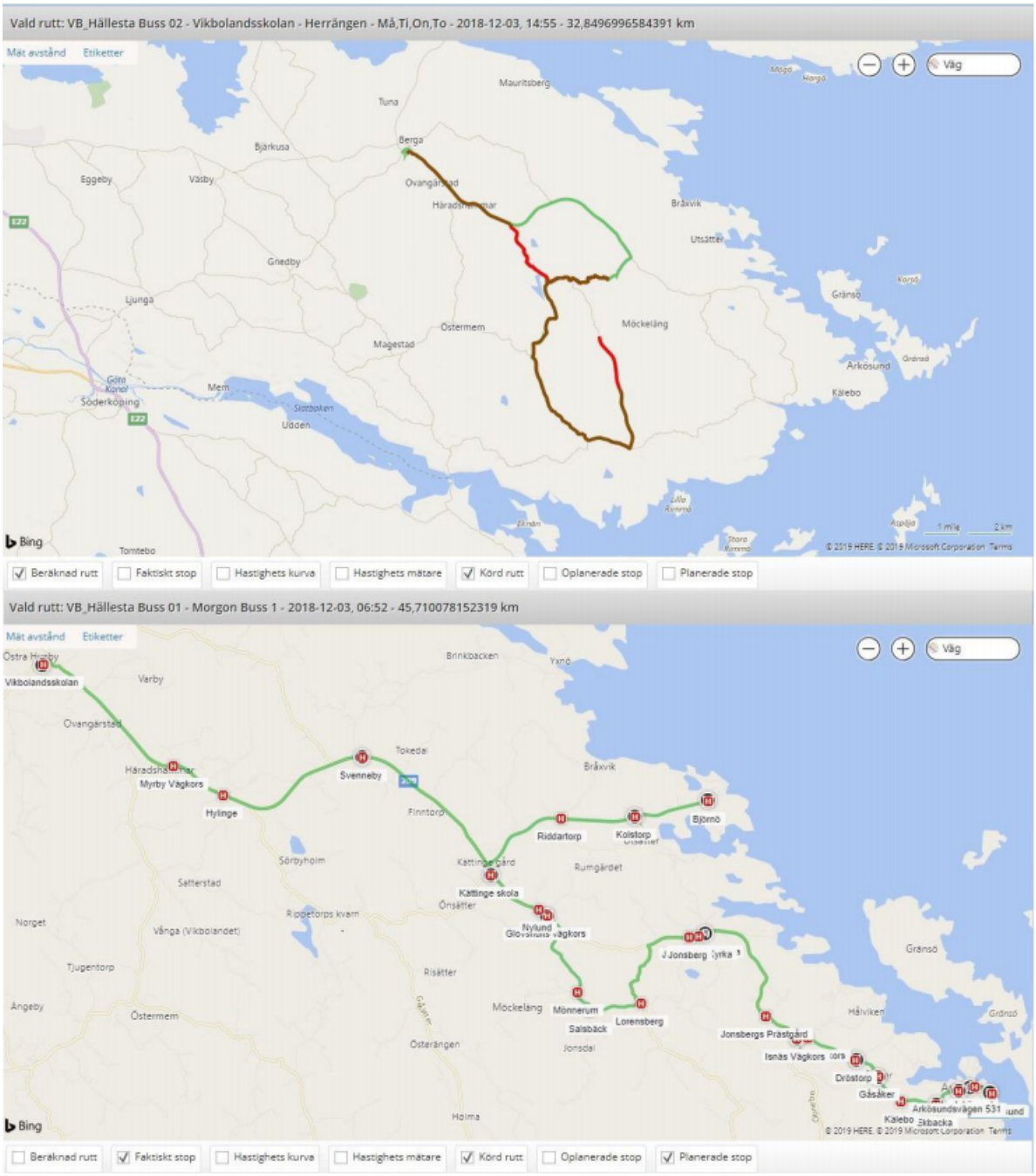



Figure 4.14: A map visualizing route comparison

picked up or dropped of at the given place. A direct comparison can me made by displaying the planned stations and pupils actual picked up or dropped off locations. Part of the result is presented in figure 4.14.

## 4.4 User test

Ten bus drivers tested the mobile application on field in two weeks time and answered a questionnaire seen in figure 3.6 after the end of the test period. The result for the questions

with a rating scale is illustrated in a graph in figure 4.15.

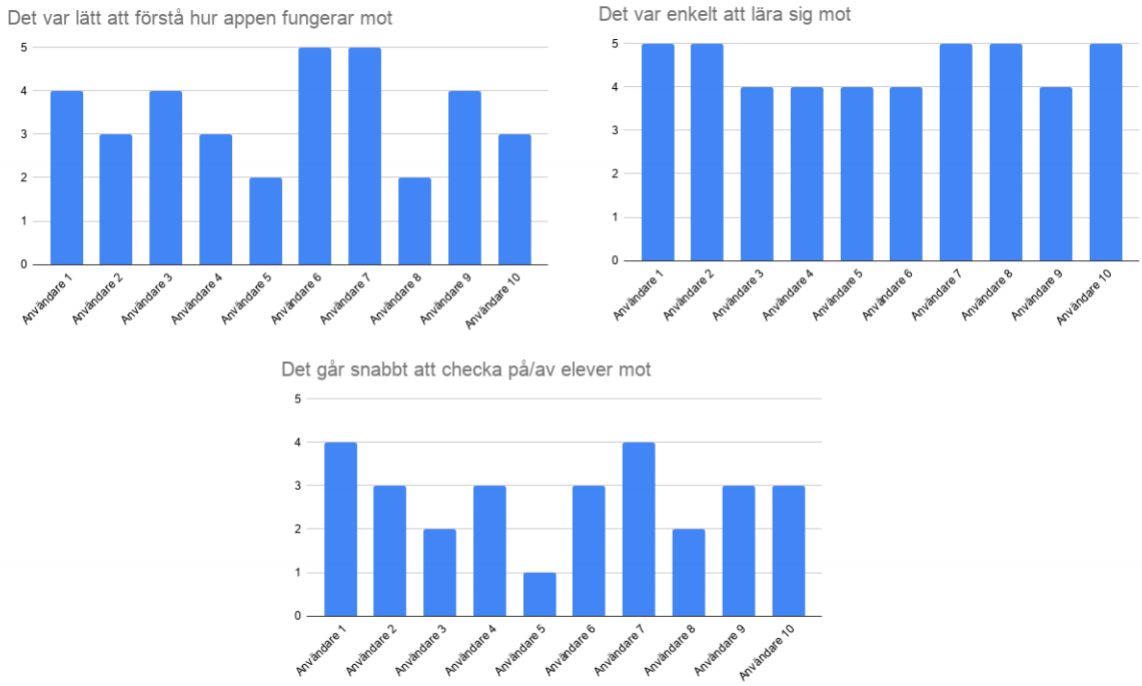



Figure 4.15: Result from the user test.

Two test person gave a grade below average for the question "is the application easy to understand" which indicated that the mobile application lacked self explaining components and that the design can be improved. The response for the follow up question "it was easy to learn" was more positive. In a scale from one to five the answers only consisted of grades of four and above. The higher the grade the easier the test person found the application easy to learn. Comparing the result for question one with question two showed that the mobile application is well designed for users to adapt and learn. The third question with a rating scale had mixed response with both positive and negative feedback. The result indicated that the mobile application needed a better solution to check-in and check-out pupils.
A summary was made from the questions with free answers from the same questionnaire. A unanimous response could be deducted that the users feel that the mobile application is better suited on tablets with larger screens. Since the mobile application has a lot of moments in which the user needs to manually take action on it is easier to press correctly the bigger the screen size. The majority also feel it's safer with a design that forces the user to confirm which pupils have been checked in with an extra button instead of only by checkboxes. The reason is human error exists and it happens frequently that a user accidentally checked in wrong pupil. Only a few complained about having an extra confirm button with the reason that it took extra time. For question three the answer showed that the application doesn't need a solution to check out pupil on another station if the pupils are being taken to school. A lot of testers find it time consuming to check out pupils at different station according to question four and consequently a solution to quickly change a pupil station is required for the application to be attractive.

The intention with question five was to collect the test users honest opinion about the mobile application. The response was positive but also highlighted possibilities for improvements.

Among positive responses the test users find the mobile application to be useful overall with good functionalities. The users also mentioned that the application had a dynamic user interface which was a good feedback for users with little experience with mobile applications. Majority of test users complained about check-in solution taking too much time and leading to frustration. The test users had couple of idéas to improve the functionality which they answered on question six. To save time the recommendation is to implement a check-in solution that forces the pupils to check in and remove the task from the bus drivers. The top idéa is for the pupils to use near-field communication, NFC tag or similar technology. Last recommendation from the test users is to enlarge the components in the application to enhance user experience by toning up the readability and improve the users actions.

# 5 Discussion

This chapter will discuss the result of the thesis and a reflection of the chosen method to achieve the goal. A summary of the final product and it's use will also be discussed in this chapter.

## 5.1 Result

The final product has all functionalities that were needed to meet the requirement of the master thesis as described in chapter 1.2. The mobile application can be used by multiple users with each unique account. To access the data the user needs to verify by authenticating the users information in Optiplan's server. By implementing Optiplan as the authentication host instead of using third party providers a more secure outcome is guaranteed since Optiplan holds all information a user needs. When the mobile application is being used the user will collect information of the device's location in real-time just by starting the application. This process will help the bus driver to focus on driving since the application will collect the driven coordinates without the help of the user when started. When the bus driver has stopped to pick up or drop off pupils the user may manage the application by confirming the station. This process will also not disturb the bus drivers focus on the road. This makes it secure to use the mobile application without any functionalities needed to be managed manually while driving.

The main task for the mobile application was to collect data of the passengers for a driven route. The idéa of the product has been implemented before and there exist applications with similar functionalities. The main difference between the developed application for the master thesis compared with existing similar applications is the collected data. The developed application stores more information and has a toolkit to visualize the information for resource optimization purposes and not only display a passengers whereabouts. With the collected data from the mobile application there are many possibilities to visualize the driven route. Each route has a planned driven path which the bus driver has to follow. With the collected information an administrator for a municipality can analyze with the help of the developed toolkit to see if the planned path has been followed. The toolkit will make it possible to compare multiple instances for example the planned path versus the driven path, the planned stop stations versus the actual stop location, the passengers planned to attend

the bus versus the actual passengers attending the bus. With the collected data a new optimal route can be planned with new paths, new passengers and new stations.

By doing research on similar applications before the development a pros and cons list were achieved. The list was used to design multiple idéas for the user interface. The components which were liked by many users were re-utilized with a new design and the components that were disliked were avoided. A user evaluation was made with a group of bus drivers in which they were presented with multiple designs. The design with the best feedback where then chosen to be implemented and tested on field. The result in chapter 4.4 shows that the users for the mobile application which are the bus drivers were pleased. The result for the user test of the final product was similar to user evaluation of the design. The majority of bus drivers agreed that each page handling each operation was user friendly. The feedback also displayed that the user interface can be improved but overall most users could learn how to use the application after a couple of times.

The result indicates that user test and user evaluation before development were important to design a user friendly interface. Without a user evaluation another design may have been selected and the satisfaction from the users would have been different. Researching similar applications also helped the end product to be unique and have better functionalities than already existing applications.

## 5.2 Method

Scrum methodology was applied during the thesis work as mentioned in section 2.3.1. The thesis work started with customer meetings and researching subjects related to the work. By applying sprint 0 it was made clear what needed to be done to achieve the end product. This made it easier to plan different sprints ahead. By investigating the market if similar applications existed a comparison could be carried out.

Different subjects were researched in sprint 0 such as different frameworks, theory about user interface and different database hosting. By researching at the start of the project a lot of information was gained but a lot of time was also wasted since the choice of framework and other technical choices could change under the thesis work. This meant that a lot of time spent under sprint 0 was not necessary.

Before the implementation a study was also carried out to map the users for the application. With a mapped user profile different designs were made. The designs were then evaluated by test subjects that matched the mapped user profile. The result from the test evaluation was utilized to create the user interface for the mobile application. By designing different prototype and having them tested it was possible to combine the different solution to a common user interface with all of the important features. Creating a user profile proved to be important to design a user friendly application for the targeted group.

Apart from sprint 0 Scrum was utilized throughout the thesis work with meetings in different intervals to gain a direct feedback. By having many iterations different parties could make sure that the thesis followed the right track. Different iterations also helped to correct mistakes and further improved specification in early stages. When a prototype was done in the development process it was tested on field by end users. The user test was carried out up to two weeks time where the user would test the prototype and treat it as the end product. After the test period an evaluation was made with the testers. The result from the evaluation was crucial for the last part of development. With the help of the result the prototype could

be improved and be developed into the end product.

Working according to Scrum had both upside and downsides. If Scrum wasn't applied it might have ended up with a lot of modifications to the product at the end of the project. With scrum big adjustments were avoided under the thesis work since sprint 0 helped with defining the end product early on. The biggest advantage of scrum couldn't be utilized since the thesis work was done by one person. This meant that Scrum may have not been the best agile method to apply for the thesis work since scrum is meant for a small team. Overall Scrum methodology was rewarding and helped the work progressed accordingly to plan and always made sure the product was on track by having meetings and making small adjustments between the sprints.

## 5.3 The work in a wider context

In a time where mobile applications are getting more popular there are demands on mobile applications for every subjects. This study shows some options that are possible by collecting data with a mobile application. By collecting information and visualizing the data analyzes can be made to impact different fields to help the society as it is today. One very important subject is the climate change. The climate has been changing in a rapid speed and different solutions are needed to help stop the greenhouse effect. One way to help is to reduce the carbon dioxide emissions. School buses have existed for a long time and using the service is needed for the society to keep going. By optimizing the route with the help of the collected data it can help lower carbon dioxide emissions by not wasting more than needed. This is true for transportation's of every kind, by collecting data to optimize routes for post delivery to truck delivery would give the same effect.

# 6 Conclusion and future work

A conclusion for the master thesis will be discussed in this chapter and thoughts for future work will be mentioned.

## 6.1 Conclusion

This section will discuss the conclusion made from the thesis work and answer the problem formulation presented in section 1.3.

### 6.1.1 How should the user interface be designed to help a user to check-in and check-out pupils the fastest way and help the users stay focused on their driving?

Designing a user interface for this thesis work required a lot of research and user mapping since the end users were a mixed group of different ages and had different experiences regarding the use of mobile applications. The user test displayed that a familiar interface increased a users time management and a simple, clean interface helped the users to keep focus on the tasks. To increase the check-in and check-out time the interface needed minimum number of actions to perform a task and the mobile application was designed accordingly. The check-in page and check-out page were able to perform the given task with two button presses at minimum but number of actions could be increased depending on situations occurring on each station. The user test showed that the end users were mostly satisfied with the user interface. Most users learned how to use the application after a couple of times and didn't find the application as a disturbance while driving because the user only needed to take actions when the bus wasn't moving.

### 6.1.2 How should the structure for database be designed to easily separate data for each customer?

Multiple databases can co-exist in the same server at the same time with mongoDB. To separate customers a database is created for each customer. Each database has all relevant collections to store the collected data. The collections created for the mobile applications are a collection of the driven route, a collections of user information, a collection of student data, a

collection of vehicle and a collection of customer information. The collections are structured to store each document as it's own entity, for example a document in driven route collection represents a driven route. This way the data for each driven route can't be mixed.

### 6.1.3 How should the application produce a better user experience compared to similar applications?

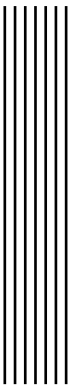
Three similar applications were found during the thesis work. The applications interface were carefully examined and user reviews were taken into consideration. Components that had good user review were copied. Components that were needed but had a lot of negative reviews were taken but reworked to be more user friendly. Researching similar applications was not enough to produce a user friendly experience, a lot of work were needed to understand the users for the developed application. Key points were to sort the users in different groups, composing an effectmap and designing different prototypes with the targeted groups in mind. By having close iterations and user test tweaks could be made to adapt to the feedback resulting in a customized user experience.

### 6.1.4 How should the toolkit be implemented to best visualize the collected data for its customers?

The main purpose of the toolkit is to help a user find out information regarding the driven route. The collected data can be used to optimize the route, optimize which passengers should attend the school bus service and control if the drivers are following the planned route. A map is needed to best visualize the driven path. A direct comparison is possible for the user, the planned path and the driven path can both be drawn on the map at the same time. The map is interactive and the user can move for interactions. Pupils check-in and check-out positions versus the planned positions can be displayed. By displaying the locations a user can see the difference without needing to compare the stored locations in raw data. A user will be able to select what kind of information to be displayed on the map to minimize confusion. Too much information on screen will give the opposite effect and the idéa is to keep the map clean with options to display information when needed.

## 6.2 Future work

The application is collecting valuable data from a driven vehicle. It would be interesting to further utilize the collected data by improving the visualizing toolkit and add more features. For this particular case where Optiplan has data of the planned route and may collect the actual driven route an algorithm can be added to combine and calculate a new route, optimized from both data. The procedure of optimizing the route as given in the specification was planned to be processed by a school administrator which will easily results in human error. An algorithm calculating would be a good solution to utilize the data correctly.



# Bibliography


[1] Tyler Technologies. *Versatrans My Stop*. URL: https://www.tylertech.com/solutions-products/versatrans-product-suite/mobile-school-bus-stop-time-look-up. Accessed: 07.25.2018.

[2] Google Play Store. *Versatrans My Stop*. URL: https://play.google.com/store/apps/details?id=com.tyler.versatrans.mystop. Accessed: 07.25.2018.

[3] UbicaBus. *The easy way to optimize your school's fleet*. URL: https://www.ubicabus.com/schools. Accessed: 07.25.2018.

[4] Google Play Store. *UbicaBus Parent*. URL: https://play.google.com/store/apps/details?id=com.artech.busev3.sdmonitoreo&hl=en_US&showAllReviews=true. Accessed: 07.25.2018.

[5] Here Comes the Bus. *Here Comes the Bus: Right Bus. Right Stop. Right Time.* URL: https://herecomesthebus.com. Accessed: 07.25.2018.

[6] Multiple Authors. *Here Comes the Bus review*. URL: https://www.facebook.com/SSHereComestheBus/reviews/. Accessed: 07.25.2018.

[7] Stephen Schroeder. *https://turtler.io/news/top-school-bus-gps-tracking-systems*. URL: https://cdn.turtler.io/photos/1/School-Bus-Systems/Here-Comes-The-Bus-School-Bus-Tracking-App.jpg. Accessed: 07.25.2018.

[8] APPNWEB Technologies LLP. *Top 7 Reasons Why People Uninstall Your Apps*. URL: https://www.appnwebtechnologies.com/blog/top-7-reasons-people-uninstall-app/. Accessed: 07.18.2018.

[9] Şeyhmus Ölker. *10 Reasons for Why People Uninstall Your Mobile App*. URL: https://appsamurai.com/10-reasons-for-why-people-uninstall-your-mobile-app/. Accessed: 07.18.2018.

[10] Capita. *Why do people uninstall apps?* URL: https://www.capitatranslationinterpreting.com/why-do-people-uninstall-apps/. Accessed: 07.18.2018.

[11] EUPHEMIA WONG. *User Interface Design Guidelines: 10 Rules of Thumb*. URL: https://www.interaction-design.org/literature/article/user-interface-design-guidelines-10-rules-of-thumb. Accessed: 07.18.2018.

[12] Soren Lausen. *User Interface Design, A Software Enginnering Perspective*. Pearson, Addison Wesley, 2005.

[13] Scott W. Ambler. *User Interface Design: Tips and Techniques*. President, Ronin International, 2000.

[14] Wilbert O. Galitz. *The Essential Guide to User Interface Design, An Introduction to GUI Design Principles and Techniques*. Wiley Computer Publishing, 2002.

[15] Caroline White. *Why Simplicity Is So Incredibly Important In UX Design*. URL: https://careerfoundry.com/en/blog/ux-design/how-important-is-simplicity-in-ux-design/. Accessed: 02.22.2020.

[16] UX Mastery. *What does a UX process look like?* URL: https://uxmastery.com/resources/process/. Accessed: 07.20.2018.

[17] MailChimp. *The UX Reader*. URL: http://www.serra-abouzeid.com/wp-content/uploads/2015/11/ux-reader.pdf. Accessed: 07.20.2018.

[18] Interaction Design Foundation. *Make Your UX Design Process Agile Using Google's Methodology*. URL: https://www.interaction-design.org/literature/article/make-your-ux-design-process-agile-using-google-s-methodology. Accessed: 07.20.2018.

[19] Jerry Cao. *What Is a Prototype: A Guide to Functional UX*. URL: https://www.uxpin.com/studio/blog/what-is-a-prototype-a-guide-to-functional-ux/. Accessed: 07.20.2018.

[20] Saadia Minhas. *User Experience Design Process, Overview of Stakeholders and Activities involved in each stage*. URL: https://uxplanet.org/user-experience-design-process-d91df1a45916. Accessed: 07.20.2018.

[21] Ingrid Ottersten and Mijo Balic. *Effect Managing IT*. URL: https://books.google.se/books?id=mXV_ukM3A8gC&printsec=frontcover&source=gbs_atb&redir_esc=y#v=onepage&q&f=false. Accessed: 10.01.2020.

[22] K. Schwaber J. Sutherland. *The Scrum Guide*. URL: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100. Accessed: 10.01.2020.

[23] Roy Thomas Fielding. *Representational State Transfer (REST)*. URL: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm. Accessed: 07.17.2018.

[24] Roy. Fielding, UC Irvine, J. Gettys, J. C. Mogul, Compaq, H. Frystyk, W3C/MIT, L. Masinter, Xerox, P. Leach, Microsoft, T. Berners-Lee, and W3C/MIT. *Hypertext Transfer Protocol – HTTP/1.1*. URL: https://www.w3.org/Protocols/HTTP/1.1/rfc2616.pdf. Accessed: 07.17.2018.

[25] Code Academy. *What is REST?* URL: https://www.codecademy.com/articles/what-is-rest. Accessed: 07.17.2018.

[26] Sagar Mane. *Understanding REST (Representational State Transfer*. URL: https://medium.com/@sagar.mane006/understanding-rest-representational-state-transfer-85256b9424aar. Accessed: 07.17.2018.

[27] Shif Ben Avraham. *What is REST—A Simple Explanation for Beginners, Part 2: REST Constraints*. URL: https://medium.com/extend/what-is-rest-a-simple-explanation-for-beginners-part-2-rest-constraints-129a4b69a582. Accessed: 07.17.2018.

[28] Alejandro Gomez. *RESTful API know-how*. URL: https://gist.github.com/alexserver/2fcc26f7e1ebcfc9f6d8. Accessed: 07.17.2018.

[29] Luis Cipriani. *API Caching, why your server needs some rest*. URL: https://www.slideshare.net/lfcipriani/api-caching-why-your-server. Accessed: 07.17.2018.

[30] Bridget Botelho Jack Vaughan Margaret Rouse. *What is MongoDB?* URL: https://searchdatamanagement.techtarget.com/definition/MongoDB. Accessed: 07.22.2019.

[31] solid IT. *About Us*. URL: https://db-engines.com/en/about. Accessed: 07.22.2019.

[32] solid IT. *DB-Engines Ranking*. URL: https://db-engines.com/en/ranking. Accessed: 07.22.2019.

[33] Amazon Web Services. *What is a Relational Database?* URL: https://aws.amazon.com/relational-database/. Accessed: 07.26.2019.

[34] Inc MongoDB. *Data Modeling Introduction*. URL: https://docs.mongodb.com/manual/core/data-modeling-introduction/. Accessed: 03.29.2020.

[35] Mike Wasson. *Authentication and Authorization in ASP.NET Web API*. URL: https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/authentication-and-authorization-in-aspnet-web-api. Accessed: 03.28.2020.

[36] Josh Lake. *What is TLS and how does it work?* URL: https://www.comparitech.com/blog/information-security/tls-encryption/. Accessed: 04.05.2020.

[37] Inc Cloudfare. *About Cloudfare*. URL: https://www.cloudflare.com/about-overview/. Accessed: 04.05.2020.

[38] Inc Cloudfare. *What Happens in a TLS Handshake?* URL: https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/. Accessed: 04.05.2020.

[39] Microsoft. *Visual Studio Tools for Xamarin*. URL: https://visualstudio.microsoft.com/xamarin/. Accessed: 07.25.2018.

[40] Google. *Maps SDK for Android, Overview*. URL: https://developers.google.com/maps/documentation/android-sdk/intro. Accessed: 07.25.2018.

[41] Microsoft. *Customizing a Xamarin.Forms Map*. URL: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/app-fundamentals/custom-renderer/map/. Accessed: 07.25.2018.

[42] Jimmy Bogard. *Mobile authentication with Xamarin.Auth and refresh tokens*. URL: https://lostechies.com/jimmybogard/2014/11/13/mobile-authentication-with-xamarin-auth-and-refresh-tokens/. Accessed: 07.25.2018.

[43] Skip Hovsmith. *Adding OAuth2 to Mobile Android and iOS Clients Using the AppAuth SDK*. URL: https://hackernoon.com/adding-oauth2-to-mobile-android-and-ios-clients-using-the-appauth-sdk-f8562f90ecff. Accessed: 07.25.2018.