



Fail-fast in the Design Process of an Interactive Voice Response System

Elise Eborn & Sara Lindgren

DEPARTMENT OF DESIGN SCIENCES
FACULTY OF ENGINEERING LTH | LUND UNIVERSITY
2019

MASTER THESIS

TELAVOX



Fail-fast in the Design Process of an Interactive Voice Response System

Elise Eborn & Sara Lindgren



LUNDS
UNIVERSITET

June 7, 2019

Fail-fast in the Design Process of an Interactive Voice Response System

©Copyright 2019 Elise Eborn, Sara Lindgren

Published by
The Department of Design Sciences
Faculty of Engineering, Lund University
Box 118, 221 00 Lund

Degree Project in Interaction Design (MAMM01)
Examinator: Johanna Persson
Supervisor: Kirsten Rasmus-Gröhn
Supervisor Telavox AB: Frida Bredberg

Abstract

This paper contains a report of the master thesis, *Fail-fast in the design process of an Interactive Voice Response System*.

The world of creating new software is currently going more and more towards being as fast and agile as possible to be able to adapt to changing customer demands. The design process of new software should reflect that, which doesn't leave much time for scheduling user tests in person. How can feedback from users be gathered in a scalable way, and the design of a product be made using a fail-fast method?

To answer these questions we chose to develop a new design of an existing product. The design was to be for a mobile-first way of viewing and editing an interactive voice response system in a web application. The old design of an interactable tree would not be optimal for a smaller screen. Two iterations of design, implementation and evaluation were performed, before a final evaluation of the whole project. The evaluations were performed by releasing the product to Telavox and one of their customers.

The application improved greatly from one iteration to the next. Between the two iterations the usability went from poor to acceptable. The different methods of gathering feedback gave different, but all helpful, results. Gathering feedback was difficult, but in the end the design was deemed usable.

Keywords: fail-fast, minimum viable product, minimum viable user experience, design process, interactive voice response

Sammanfattning

Detta dokument innehåller en rapport av examensarbetet, *Fail-fast i designprocessen för ett talsvarssystem*.

Utvecklingen av ny mjukvara rör sig mer och mer mot att vara så snabb och rörlig som möjligt för att kunna anpassa sig efter ändringar i kundens behov. Designprocessen för ny mjukvara behöver vara på liknande sätt, vilket inte lämnar mycket tid till att boka in användartest. Hur kan återkoppling från användarna samlas in på ett skalbart sätt, och designen av produkten göras med en fail-fast metod?

För att svara på detta gjordes en ny design av en redan existerande produkt. Designen var för ett mobilvänligt sätt att kolla på och redigera ett talsvarssystem i en webbapplikation. Den gamla designen med ett interagerbart träd var inte optimal för en mindre skärm. Två iterationer av design, implementation och utvärdering utfördes, innan en sista utvärdering av hela projektet gjordes. Utvärderingarna utfördes genom att släppa produkten till Telavox och en av deras kunder.

Applikationen förbättrades mycket efter bara en iteration. Användbarheten gick från en undermålig till en acceptabel nivå mellan de två iterationerna. De olika testade metoderna av att samla återkoppling gav olika sorters resultat, men alla var användbara för att vidareutveckla applikationen. Att samla återkoppling var svårt, men designen i slutändan bedömdes vara användbar.

Nyckelord: fail-fast, minimum viable product, minimum viable user experience, designprocess, talsvar

Acknowledgements

We would like to thank our supervisor at LTH, Kirsten Rasmus-Gröhn for all her help and support.

We would also like to thank Telavox, and especially Frida Bredberg and Zsolt Demeter for all their help during the course of the entire project. Also thanks to Sofie Eliasson, Magnus Sillén and the rest of the Webapp team for reviewing our code and coming with suggestions during this project. Thank you to Henrik Thorvinger and Magnus Lorentzon for valuable UX input.

A last thank you to all the people who took the time to test our solution. We really appreciate it!

Lund, May 2019

Elise Eborn & Sara Lindgren

Table of contents

Acronyms & abbreviations	8
1 Introduction	9
1.1 Background	9
1.1.1 Telavox	9
1.1.2 Telavox's IVR system	10
1.2 Purpose	10
1.3 Research questions	11
1.4 Method	12
1.5 Terminology	13
1.6 Delimitations	13
1.7 Division of work	13
2 Theoretical background	14
2.1 User experience	14
2.2 Iterative design	15
2.3 Evolutionary prototyping	15
2.4 Fail-fast	15
2.4.1 Minimum viable product	16
2.4.2 Minimum viable user experience	16
2.5 React.js	17
2.6 Usability Evaluation	17
2.6.1 Heuristic evaluation	17
2.6.2 Catastrophe insurance	18
2.6.3 Google Analytics	18
2.6.4 System Usability Scale	18
2.7 Brainstorming	19
3 Iteration 1: a simple settings menu	21
3.1 Design phase	21
3.1.1 Method	21
3.1.2 Result	22
3.2 Implementation phase	23

3.2.1	Method	23
3.2.2	Result	24
3.3	Evaluation phase	24
3.3.1	Method	25
3.3.2	Result	27
3.4	Discussion	28
4	Iteration 2: a keypad menu	30
4.1	Design phase	30
4.1.1	Method	31
4.1.2	Result	31
4.2	Implementation phase	32
4.2.1	Method	32
4.2.2	Result	33
4.3	Evaluation phase	33
4.3.1	Method	34
4.3.2	Result	37
4.4	Discussion	39
5	Discussion	40
5.1	Iteration 1 vs Iteration 2	40
5.2	Fail-fast	41
5.3	Gathering feedback	42
5.3.1	Heuristic evaluation	42
5.3.2	Catastrophe insurance	43
5.3.3	Google Analytics	43
5.3.4	Survey	44
5.4	Further improvements	45
5.4.1	Sounds	45
5.4.2	More than one submenu	46
5.4.3	More than one profile	46
5.4.4	Creating a new IVR	46
6	Conclusion	47
A	Evaluation form	52
B	Google Analytics Data Iteration 1	54
C	Google Analytics Data Iteration 2	57

Acronyms & abbreviations

- IVR** *Interactive Voice Response.* A system where a caller reaches a prerecorded message which gives the caller alternatives to be redirected to by pressing a certain digit on the callers phone.
- UX** *User eXperience.* A person's perceptions and responses that result from the use or anticipated use of a product, system or service[1]. See Section 2.1
- MVP** *Minimum Viable Product.* A product which has been developed in a fast manner where some functionality might be missing. See Section 2.4.1
- MVUX** *Minimum Viable User eXperience.* A version of MVP where the user experience is produced in a fast manner with the intention of testing it on people at an early stage. See Section 2.4.1.
- SUS** *System Usability Scale.* A scale which is formed by 10 statements which are answered with a Likert scale. A score is produced from which assumptions on the products usability can be made. See Section 2.6.4.

Chapter 1

Introduction

This chapter will describe the background for this master thesis. It will also describe the purpose, research questions, method, terminology, delimitations and the division of work. The background will include a description of the IVR system we worked on and information about the company who hosts the product, Telavox AB.

1.1 Background

An interactive voice response (IVR) system is a system where a caller reaches a prerecorded message which gives them alternatives to be redirected to by pressing a certain digit on the callers phone. This in turn can either give the caller additional choices, or it can redirect them to the right area or phone number. IVR systems are used in telephony all over the world.

1.1.1 Telavox

Telavox AB is a Swedish IT company that was founded in 2003 as a startup. Since then, they have grown significantly and can now call over 15000 companies their clients. Their specialty is combining telephony, Private Branch Exchange, video and chat in a cloud-based solution in a way that expands the way a business communicates.

1.1.2 Telavox's IVR system

The IVR system in question is already implemented in a different environment than the environment worked on in this thesis project, and can be seen in Figure 1.1. It does not scale well to the smaller screen of a mobile phone, as seen in Figure 1.2. Three out of four users log in to Telavox's services via a mobile phone, and a portion of them only have access to the features provided to phones. Administrating the IVR-system is today one of those features which is not reachable from a mobile phone. Therefore our solution should be written mobile-first, so that it can be responsive first and foremost and reachable by all of Telavox's customers. Telavox also wanted to evaluate the use of fail-fast in the design process. Fail-fast is a method where the distance between idea and release is as short as possible to gather user feedback early while spending as little resources as possible. Read more about fail-fast in Section 2.4.

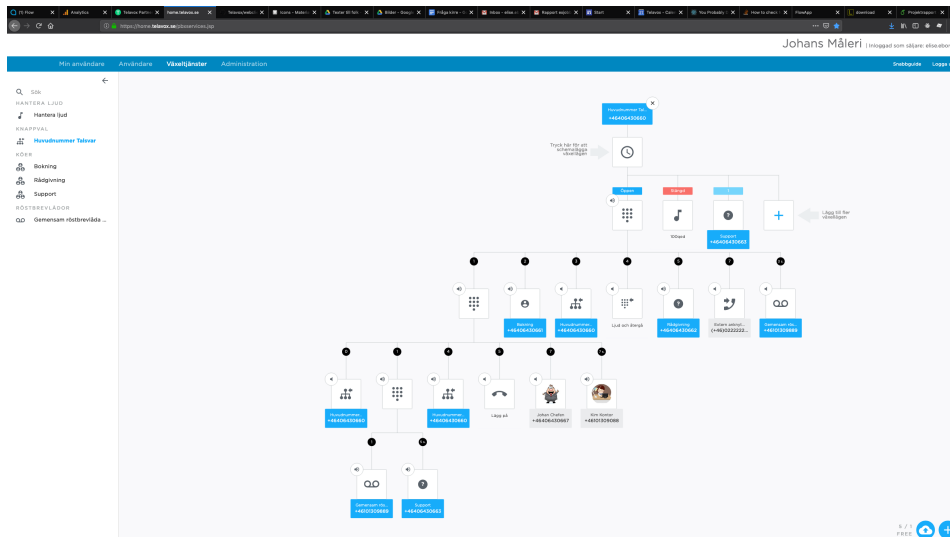


Figure 1.1: The current IVR system for a desktop environment

1.2 Purpose

The primary purpose of this master thesis is to investigate the difficulties of using a fail-fast method in the design phase of a project. More specifically, the aim is to compare methods of collecting feedback, which is necessary when continuing to develop a prototype. As Telavox is a rather large company, the focus will be on scalable methods of collecting feedback. Scalable is used here to mean methods of collecting feedback from a large amount of users, above 100 and maybe from all over the world, that can not easily be invited in to do

a user study.

This design method is evaluated by developing an IVR solution focused on working on a smaller screen, for example a tablet or a phone. One problem facing an IVR solution which is specific to a smaller screen is how it should show the user where they are in the tree structure. Another goal was to put the product into production and test it on real customers.

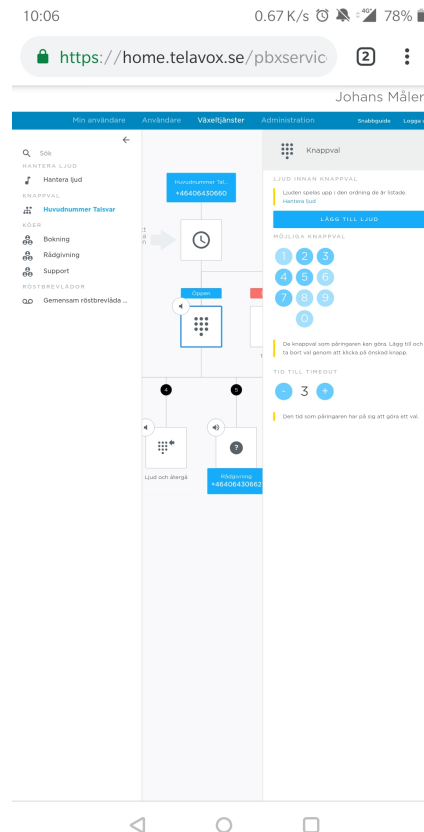


Figure 1.2: The current IVR system on a phone

1.3 Research questions

These are the research questions we intend to answer in this master thesis:

- How can feedback be collected in a scalable way when developing a mobile-first application?
- What are the challenges when working with a fail-fast design process when developing a mobile-first application?

1.4 Method

The process of this thesis work was as shown in Figure 1.3. We started off with a research phase where we deepened our knowledge of the fail-fast process and other suitable methods which could be useful in the thesis work. Afterwards we began the practical part, which consisted of two iterations which both consisted of a design-, an implementation- and an evaluation phase. The design phases were intentionally kept very short due to the fail-fast method and the implementation phase included implementing the design into a high fidelity prototype using React.js. From the beginning, the goal was to release a working product at least internally after seven weeks, to test the fail-fast method. The evaluation phases included the evaluation of the high fidelity prototypes in a realistic environment. A selection of more or less scalable evaluation methods were chosen to test the application. These were heuristic evaluation, catastrophe insurance, Google Analytics and System Usability Scale. Read more about these methods in Section 2.6. After the evaluation phase of the second iteration, a final product was presented.

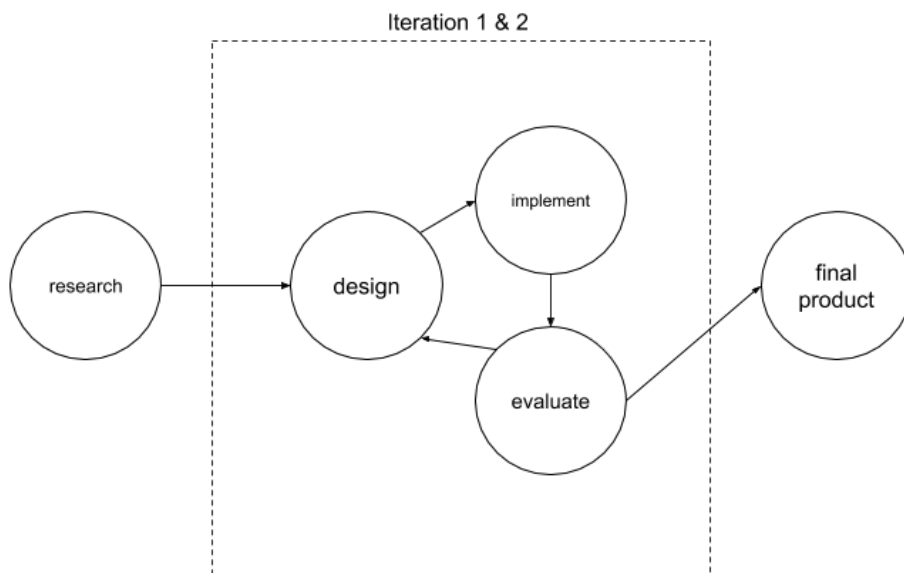


Figure 1.3: Our process

1.5 Terminology

Some terminology to describe an IVR system is used repeatedly in this work. As the system is visualised as a tree, most terminology is borrowed from graph theory. The end points of the system, or the leaves of the tree graph are referred to as nodes. The intermediary nodes in the tree are referred to as submenus. The entrance point to the system is called the root menu. Callers enter the tree at the root menu, and press digits until they reach one of the nodes. Then the action specified by that node happens. A visual representation of such a tree is shown in Figure 1.4.

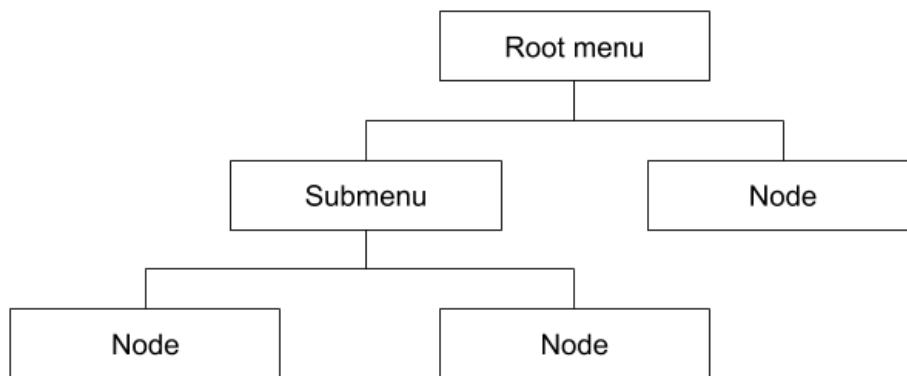


Figure 1.4: Terminology in the IVR system

1.6 Delimitations

Fail-fast is a method with many names, like *lean startup* which is used amongst startups [2] and *think it, build it, ship it, tweak it* which is used at Spotify [3]. Fail-fast can also have different meanings in a development environment. We chose to use the one described in Section 2.4.

1.7 Division of work

Sara has led the work in the design phases and evaluation phases and Elise has led the work in the implementation phases. Both have worked together during the whole project.

Chapter 2

Theoretical background

This chapter will explain more in depth the theoretical background on which this master thesis is based. It will shortly explain concepts in designing, implementing and evaluating software products.

2.1 User experience

Users interact with products in a myriad of ways, and many factors influence how the product is received. When Donald Norman, author of well-known book *The design of everyday things* [4], wanted to cover all the ways a user interacted with a product he coined the term user experience (UX), as confirmed in his 2007 interview with Peter Merholz [5]. UX includes industrial design, graphics, user interface, physical interaction and manuals. Norman has however also said that the term has lost some of its meaning in being used so widely as it is. One way to describe UX is:

Designing things simple enough for people to understand and delightful enough for people to want to use. [6]

Arvola [7] writes that a general guideline for a program to be easy to use is that it should be broad and shallow, as opposed to being narrow and deep. He also writes that when the components in the interface follow a pattern it is restful for the eye, and when something sticks out from this pattern it is more immediately recognizable.

2.2 Iterative design

Iterative design is one of the important parts of human-centered design [7]. It is when the design is improved through different iterations.

Preece et al. [8] write that the project stage most likely to cause project failure is the requirement definition stage. They also write that the factor most mentioned to cause successful projects was clear and detailed requirements.

In Norman's book *The Design of Everyday Things* [4] he recounts a conversation with a designer who says that it takes five to six attempts to get a product right. Also that when introducing a new product to market you only get two or three attempts to introduce it before no one is interested anymore. This paradoxical statement means that all new and revolutionary products are almost guaranteed to fail. Even Norman didn't get his book right the first time, as it was released with the title *The Psychology of Everyday Things*, which did not interest an as broad reader base as the book has today.

2.3 Evolutionary prototyping

Evolutionary prototyping is when a prototype is evolved into the final product. A first prototype is created based on the best understood requirement and after receiving feedback from the customer, the prototype is improved in iterations. This method of prototyping is often used in agile development, where each development iteration brings the product closer to its final form [7, 8].

2.4 Fail-fast

In today's society, it is important for IT companies to get their products on the market before the competition does. This means that the time between idea and a released product should be as short as possible, while still retaining the quality. To facilitate this, a company could use the fail-fast method [9]. It is based on the idea that the faster a bad idea can be invalidated, the faster it can be restructured into a good idea. This method includes a shorter planning- and implementation phase in favour of getting a product on the market as soon as possible. The product will then be evaluated by users and changed through a series of iterations. It is called fail-fast due to the fact that a product gets produced in a fast manner, where some functionality might be missing, and is then put in front of real customers in a real environment. The customers can

then decide in which areas the product needs improvement or if the product is wanted at all.

Fail-fast is a popular method among startup companies, where resources are scarce and the time to market has to be fast. However, resources spent and time to market should be something that larger companies should care about too.

The first product which is produced during the fail-fast method can be called a minimum viable product, or minimum viable user experience depending on the nature of the product.

2.4.1 Minimum viable product

The minimum viable product (MVP) [9] technique involves creating a product that is as simplistic as possible, but still is able to be put on the market. It lets the users try the product in a realistic environment and it lets the company know if it is a product that the users actually want. MVP is based on the *learn-measure-build cycle* in Figure 2.1, which represents the fast changes that can occur in a product in its early stage.

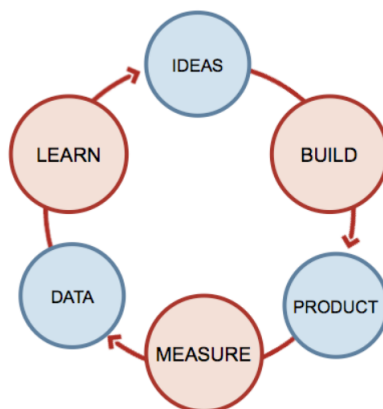


Figure 2.1: The learn-measure-build cycle [10] of which MVP is based on

Minimum viable product is very similar to the principles of eXtreme Programming, for example “Do the simplest thing that could possibly work” [11].

2.4.2 Minimum viable user experience

There is also a quite new term which is minimum viable user experience (MVUX) [12]. This method is like the MVP method except for the fact that

it focuses on the user experience of a product instead of the core functionality of the product. Instead of creating a low fidelity prototype of the product, the developers will produce a realistic and functioning interface without much input from the end user. This interface can then be tested on end users directly in a more realistic environment to then be further developed according to the *learn-measure-build cycle* in Figure 2.1.

2.5 React.js

React.js is “a JavaScript library for building user interfaces” [13]. JavaScript[14] is a scripting language often used in the creation of web applications. It was released in 1995 and there are several frameworks and libraries built on top of it. React was released in 2013 and is often used for developing front-end applications due to its modularity and quickness to fetch data when loading for example a web page. It is modular due to each web page being built up by components, in a way reminiscent of object-oriented programming. An important concept in React.js is props and state. Props, short for properties, are read-only values that are passed from a parent component to its child. State on the other hand is managed by each component, and can be changed asynchronously. Simple components have no state, but may get values from their parents state as props.

2.6 Usability Evaluation

There are many different ways to evaluate a product, and many different things to focus on. Below, some evaluation methods used in this master thesis are described.

2.6.1 Heuristic evaluation

A heuristic evaluation [15] is when a usability expert with no or little involvement in the project performs a review of the product with the intended user and usability standards in mind. This evaluation can be done individually or in groups. It is a cheap alternative [16] and is easy to organize due to the lack of constraints. However, the success of the evaluation is quite dependent on the expert who is doing the evaluation, which should be kept in mind while analyzing the results. A person can be considered a “double expert” if they

are well versed in both human factors and the domain area of the product in question.

2.6.2 Catastrophe insurance

Catastrophe insurance [15] is a form of validation testing where the product is sent out on an internal system. This will lead to coworkers being able to test the system and find bugs the developer might have missed before customers have a risk of finding them. In doing this, the developer have a chance to fix minor bugs before the product is released to the customers. If the errors are too large to fix directly, the developers have a chance to warn the support team which complaints might be incoming when the product has been released.

2.6.3 Google Analytics

One way to collect quantitative data is with the help of Google Analytics [17]. The way it works is that different points of measurements are added to the code of the product. When a user performs a task connected to these measurement points, it is triggered and data is sent to a Google website. This can then be viewed on different setups and can be analyzed.

2.6.4 System Usability Scale

A System Usability Scale (SUS) [18] is where 10 statements are assessed on a Likert scale from 1 - do not agree to 5 - agree fully. The result is then compiled into a score from 0 to 100 which can indicate how usable a product is, where a higher number is better.

Tullis and Stetson concluded that a sample size of 8-12 users are sufficient to reach a reliable result using the SUS questionnaire [19].

One popular way of interpreting the SUS score is to divide them into percentiles, where it has been found that the score of 68 is the average score, and therefore the 50th percentile [20]. Bangor et al. described a SUS score of 70 as passable [21], where high 70s to upper 80s would describe a better product. They also said that a product with a score of less than 70 should be a subject of scrutiny and rethought. A product with a score of less than 50 should be considered a failure and be deemed unacceptable.

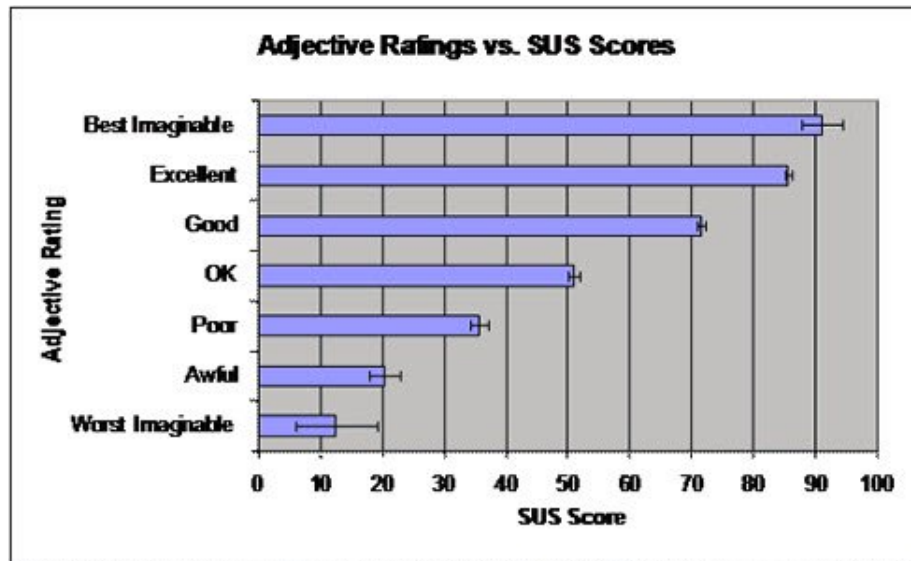


Figure 2.2: SUS translated to adjective ratings [22]

Another way of interpreting the SUS score is like Bangor et al. who translated the SUS scale to adjective ratings. This was done in their article Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale [22] as seen in Figure 2.2. We will use these adjective ratings while discussing the SUS result.

2.7 Brainstorming

A brainstorming session [23] is where group of people gather in an open and constructive environment to solve problems or create ideas. Firstly, an individual brainstorming is done with the intention of creating and gathering individual ideas. These ideas are written down and brought to the joint brainstorming session which is held afterwards. At this session, the participants read through the ideas from the individual sessions and when discussing the possibilities improve the existing ideas and create new ones. Every new idea is written down.

In some cases the joint brainstorming session is held firstly followed by the individual one, and both can be claimed to work better depending on different studies. [24, 25]

According to Osborn [26] there are three rules during a brainstorming session:

1. Criticism is forbidden

2. Quantity is wanted
3. Freewheeling is welcome

These rules are meant to encourage creativity and get the participating members comfortable enough to mention any idea that comes to their mind. The idea is that every idea is a good idea, it just need some work sometimes.

Chapter 3

Iteration 1: a simple settings menu

This chapter will describe the first iteration of two. The iteration consists of a design-, an implementation- and an evaluation phase. The design was based on the settings menus already in the existing web application. This design was then implemented in React.js and the functioning solution was then evaluated with an heuristic evaluation, a catastrophe insurance, Google Analytics and a Google form.

3.1 Design phase

Due to the fail-fast method we intended to follow, the design phase was purposely kept as short as possible, about a week. The thought was to spend less resources on the design phase to be able to spend more resources on the subsequent implementation- and evaluation phase.

3.1.1 Method

It became known to us that the UX department at Telavox had some ideas for potential solutions to the design. They had made some sketches which can be seen in Figure 3.1. After evaluating this design, what we wanted to add was a way for the user to know where in the tree structure they are.

As a next step we held a meeting with people from the UX department at Telavox. At this meeting we discussed possible design solutions and in the end established three different design alternatives, hereafter called A, B and

C. A, as seen in Figure 3.2, was a simplistic settings menu that already existed in some form in the system, but put together in a way which suited the IVR system. B was the design from the UX team mentioned above. C was a tree structure similar to the one in the desktop environment seen in Figure 1.1. These alternatives were then discussed in terms of pros and cons of both user impact and implementation cost.

3.1.2 Result

Design C was eliminated due to it working poorly on a mobile platform. It was decided that we would start implementing design A as there was some precedent for it in the application already, so it would both be easier to implement and perhaps intuitive for users. What differed between A and B was the layout element, so implementing A would make it easier to implement B at a later time.

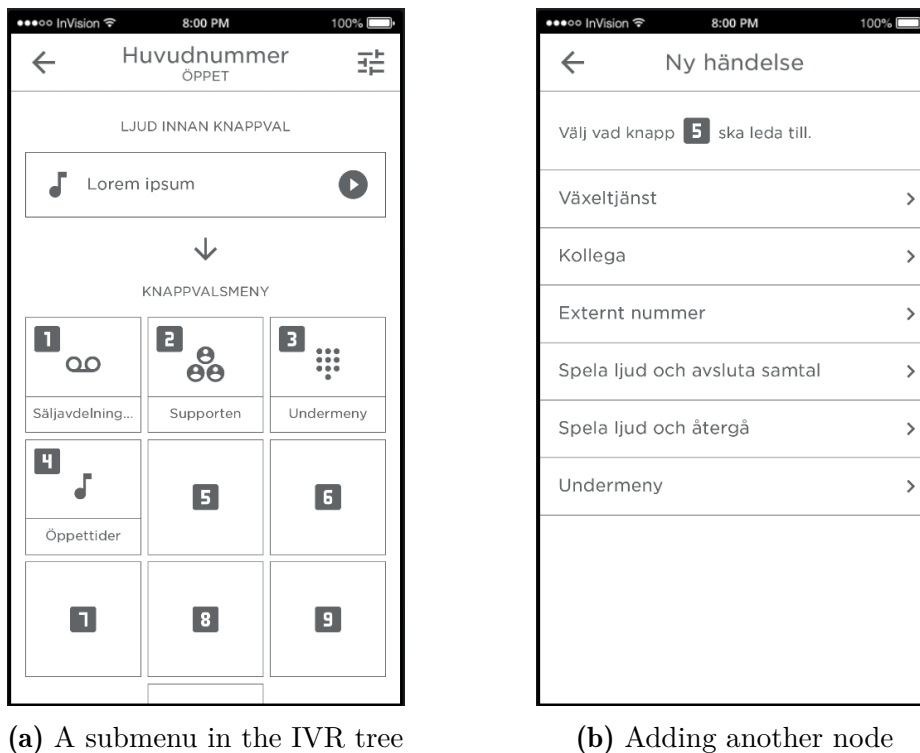


Figure 3.1: Potential prototype B

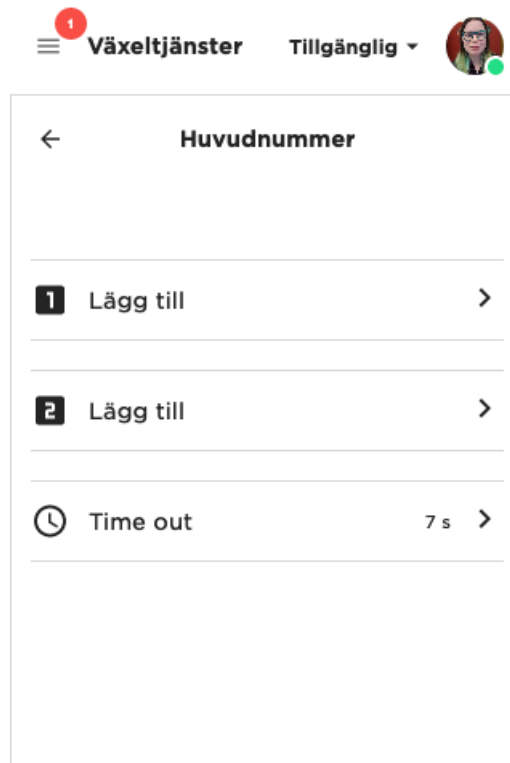


Figure 3.2: Potential prototype A

3.2 Implementation phase

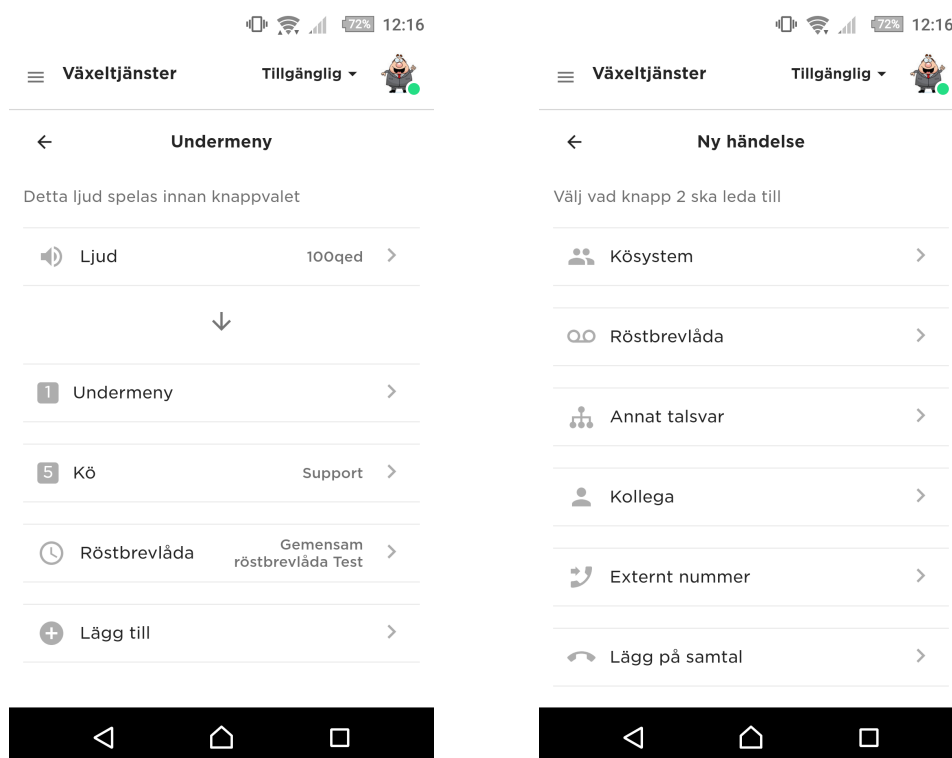
After the design phase we implemented our chosen design. We chose to implement it in React.js, described in Section 2.5, as it was important for us to get test data of the customers using the application with their real data. It was also chosen as the rest of the web application is written in React.js.

3.2.1 Method

First we mocked up some layouts by building the user interface as intended but using placeholder data. When this was done, real data was read from the database and inserted in the correct places. The last step of the implementation was to be able to make changes in the database from our application. Measurement points were also put in the code to produce quantitative data via Google Analytics. During the implementation we got feedback from the UX department on design choices as well as from the web application team on best practices during coding.

3.2.2 Result

The final version of the application for this phase can be seen in Figure 3.3. The final product was a fully functional IVR solution with the capability to alter the database. The features implemented were adding new nodes and submenus, removing nodes and changing existing nodes. The features that had not been implemented yet were handling sound in existing nodes and submenus, and removing submenus.



(a) A submenu in the IVR tree

(b) Adding another node

Figure 3.3: Prototype A implemented

3.3 Evaluation phase

The evaluation phase after the first implementation phase is an especially important phase while working on an MVP. This is the first time the end user sees the product and it is important to evaluate it properly to be able to make further developments on the product to the users' liking.

3.3.1 Method

At first, the solution was released under a “secret” URL which could be reached by navigating to a specific point on the website and adding “/ivr” to the end of the URL. A new release was made after the heuristic evaluation and catastrophe insurance which meant that the solution could be reached via a button for some users and via the “secret” URL for the rest.

Heuristic evaluation

As a first step to evaluate the solution, we called in an expert in interaction design to do a heuristic evaluation. This evaluation was performed with some guidelines. At first we asked the evaluator to perform some basic tasks like “remove a node” or “add a submenu”. After that we let the evaluator roam free and point out weaknesses. These weaknesses, which are mentioned in Section 3.3.2, were improved before the next step in the evaluation process.

Catastrophe insurance

After the heuristic evaluation was performed and the improvements were made, we did an internal release. This was done in order to perform a catastrophe insurance. In our catastrophe insurance, we sent out our product under a secret URL on the beta branch in the web app. This meant that the coworkers had to log in on the beta version of the website, which is a version the customers cannot reach. From this, we gained some insight in bugs that could appear in our solution. These bugs are mentioned in Section 3.3.2.

Google Analytics

We used the measurement points put in the code to collect quantitative data of the users behaviours.

The questions we wanted to answer with Google Analytics are:

- How far down the tree does the user go?
- Does the user look to see the possibility to add a new node?
- Are changes made? Which ones?
- Is it used mainly via mobile?
- What is the engagement on mobile versus desktop?

To answer these questions we set a few goals.

- To measure that the users fully explored the tree, we thought that at least 50% of users that went to level 0 should go on to level 1 if it existed.
- We thought that if less than 10% of users performed actual changes (such as adding or deleting a node, or changing a node's sound or action) it would be a good measure of if something needed to be done to clarify what could be changed in the IVR tree.
- We wanted a higher percentage of users to use our solution via a mobile phone than the web application had otherwise. This would mean more than 5% of users.
- We wanted 50% of users to be looking at adding new nodes. We also wanted to look at how many users were actually making changes on mobile platforms.

Google Form

We developed a form to get more information from the people testing. Both quantitative and qualitative data can be gathered from this form. The quantitative data is collected from a System Usability Scale (SUS). We also added some qualitative questions to gather some more specific feedback to considered in further development.

The qualitative questions in the form are meant to answer the following questions:

- How well do users understand the symbols and icons?
- Is the response time a cause of user frustration or errors?

We chose to leave the questions in the form open-ended, even though we were looking for specific information, so as not to guide the users' answers. If no one commented anything about response times or the symbols and icons, it could be assumed that they were clear enough as to not draw unwanted attention. The form can be seen in its entirety in appendix A.

This form was sent out to the advisors and lead technicians at Telavox and one customer. Advisors are the ones who are in direct contact with the customers and help them with support. Lead technicians are the ones who facilitate communication between the development teams and the advisors. The customer in question was a medium sized Swedish company that had expressed interest in testing our application.

3.3.2 Result

When performing the heuristic evaluation, the evaluator uncovered some minor details which could be changed to improve the user experience. It was mostly naming of buttons and the color of an unclickable item that was mentioned.

The developers in the catastrophe insurance were able to find some bugs in the solution. Some of the bugs that arose were already known bugs, but that we had forgotten to fix, for example some functionality which was not included in the release but not removed fully from the code, and some crashes in the system. We were able to solve these before the release to the customers. However, we were not able to fix other bugs that came up for example problems with reloading certain pages and some bugs which we could not reproduce.

From the qualitative questions on the form, we found that users found the lack of visualization of the tree confusing. They did not find it easy enough to understand and would like some element which would improve the visual experience. Some also mentioned certain features they found to be missing, for example the lack of ability to interact with sounds. Neither the icons nor the response time were mentioned, but this could be because the other issues were found to be more important.

The SUS form got 8 answers. The score was calculated individually for the people who answered and then an average score was comprised. The final SUS score was 51.875 , which according to Bangor et al. [21] is below 68 and therefore is not good. However, it is over 50 which would have been terrible. According to the Adjective Rating Scale [22] the score is barely over “OK”. We also calculated a confidence interval with a confidence level of 95% which gave us a error margin of ± 15.844 .

When a measurement point from Google Analytics is triggered, it is presented as an event representing that specific measurement point.

Of the 72 events that were users entering the first level of a tree, 50 of them were on a tree with only one level, as can be seen in Table 3.1. This leaves 22 events entering trees with more than one level. 24 events were users entering the second level of the tree. Therefore, 109% of users went on to a lower level from the first level, assuming all the unknown trees have more than one level. We did not measure when users backed to a previous page, which could explain the strange percentage. However, an above 50% result can still probably be assumed, as users backing to a previous page shouldn't exceed the number of users going forward, due to the linear nature of the application.

Tree	1	2	3	4	5	6	7	8	9	10	11
number of levels in tree	1	1	2	1	2	1	3	2	2	?	?
nbr. of events entering this tree	29	11	8	6	5	4	3	2	2	1	1

Table 3.1: Table of the trees visited in Iteration 1

26.5% of events were navigation events for creating a new node. 4.7% events were making changes. But, it should be taken into account that 40% of the events entering a tree were on “Tree 1” which was Telavox’s main number. Users would probably be very apprehensive of even trying to make changes there. With adjustments for this information, the navigation events for creating a new node were 44.4% and the changes were 7.9%. The changes made can be seen in Table 3.2.

5.6% events were on mobile, during the same period it was 5.3% on the web application at large. No changes or attempts at changes were made on mobile.

Action	nbr of events	% of total
Saved a change in an existing node	12	60%
Deleted a node	5	25%
Created a new node	3	15%

Table 3.2: Table of the actions making changes in Iteration 1

The results can be seen in their entirety in Appendix B.

3.4 Discussion

Some of the people testing noted certain features missing which they deemed necessary. These were mostly features we chose to postpone to a later release due to the time restrictions of the fail-fast method. At least one person was very negative to the design in this iteration, saying that it was a step back from the current design. It may not have been communicated well enough that this was to be a mobile-first application, but this dissatisfaction was also reflected in the SUS score.

The SUS score is not good, but it could be a lot worse. This showed us that some major improvements had to be made, both to the design, and to the

overall feel of the solution. Bangor et al. [22] mention this when discussing adjective ratings, that “OK” should not be taken to mean that the product is not in need of improvement. We are aiming for at least a SUS score of 68 but preferably the rating of “Excellent” which would mean a SUS score of 85 or higher.

About half of the Google Analytics measurements reached their set goals, and the rest made it pretty close. Therefore improvements such as the process of making changes should be made clearer in the next iteration. We did get more engagement on mobile on our part of the application than on the application in general. Even so, not a lot of people use the web application on phones. However pages from the web application are also used in the Android and iPhone apps, so it is possible that the mobile-first approach will still be useful. All users with the possibility to do so navigated to a lower level. Therefore, the submenu button does not have to be changed.

Chapter 4

Iteration 2: a keypad menu

This chapter will describe the second iteration of two. The iteration consist of a design-, an implementation- and an evaluation phase. The design was based on the keypad of a phone. This design was developed based on the feedback from the evaluation phase in the previous iteration and brainstorming sessions. This design was then implemented in React.js and the functioning solution was then evaluated with an heuristic evaluation, an catastrophe insurance, Google Analytics and a Google form.

4.1 Design phase

When the results from the evaluation phase, described in Section 3.3.2, had been analyzed and ideas of which areas of the first prototype needed improvement was formed, it was time to realize them.

After evaluating the results of the first iteration we gathered some main goals of improvement to be worked on in this iteration. These were:

1. Improving the overall usability
2. Finding a way to let the user know where in the tree they are located
3. Showing the users the progress when creating a new node

This design phase was kept to just a couple of days, while still finding solutions for the problems above.

4.1.1 Method

As a first step in the second design phase, we held a brainstorming session ourselves. We focused on the main goals of improvements for this iteration and brainstormed solutions for these topics.

Afterwards a second brainstorming session was held containing four people, the authors and two employees from Telavox with insight into the project.

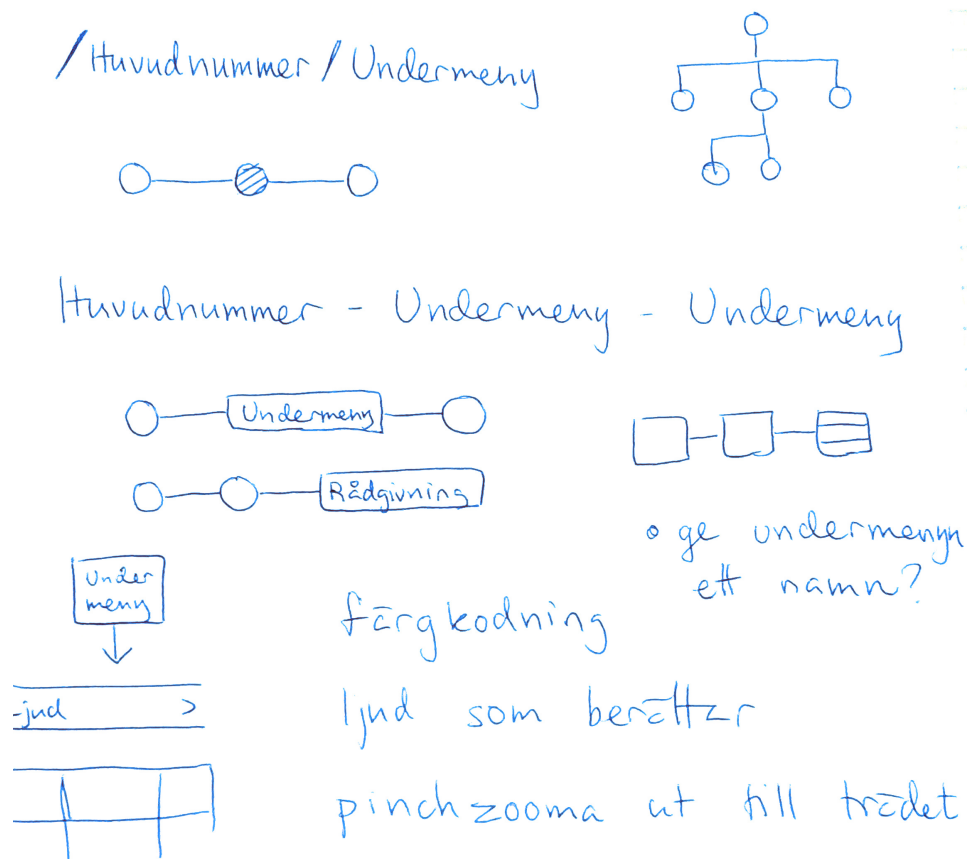


Figure 4.1: The results of the second brainstorming session

4.1.2 Result

To improve the overall usability of the solution, the qualitative questions from the Google form were considered. From this it was found that design B from Section 3.1 would be a suitable improvement to the solution. This was considered the main solution to the first goal of improvement *Improving the overall usability*, but the rest of the solutions could also be considered a solution to this goal.

In the brainstorming sessions, we found that multiple people suggested a progress bar above the existing function for creating a new node. This to solve the third goal of improvement, that making changes should be clearer. Since a progress bar was an easy and clear way to solve this problem, we decided to implement it.

Another solution which was conceived during the brainstorming sessions was how to show the user where in the tree they were located at. Many ideas emerged, as seen in Figure 4.1 and in the end we chose the simplistic, but clear, solutions of having circles on the top of the screen represent submenus. The idea was to fill in the circle representing the submenu the user was located at with a color. It was chosen due to the fact that it was best suitable to be scaled depending on the screen. It was also the option which gave the most information to the user without cluttering the screen. This solution was meant to solve the second goal of improvement for this iteration, to let the users know where in the tree they were located.

As another solution to the second goal of improvement, to let the users know where in the tree they were located, we wanted the users to be able to show the tree in its entirety. We did not want a tree on every page and therefore wanted a button the user could press to show it. We found on other places on the website that a button was used in the upper right corner for settings. Since this was not used in our solution, we found that this was the best place for the button to show the tree.

4.2 Implementation phase

The second implementation phase was done by reworking the existing solution from the first iteration in this thesis work. The rework was done in React.js.

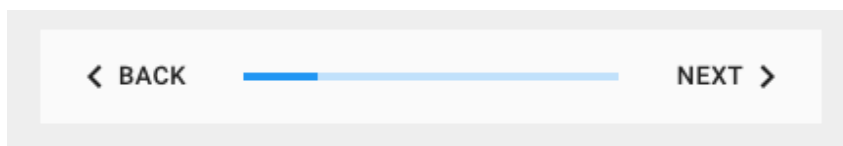


Figure 4.2: A Material UI mobile stepper [27]

4.2.1 Method

This time the implementation involved a lot of writing CSS to make custom components and tweak Material UI components to fit our needs. We created the “phone button” component completely from scratch, and used Material

UI steppers for the progress bar when creating a new node, as seen in Figure 4.2, and to indicate which level of the tree the user was, as seen in Figure 4.3. Right away when we added the stepper for level indication, we noticed that we were often trying to click on the different levels to navigate. After some research, we found that making the icons clickable was possible, and therefore added that as a sort of shortcut for experienced users of large trees. This is also explained by Shneiderman [28] where experienced users and beginners have different needs for good usability. Beginners need more explanations, while experienced users like shortcuts. Also, icons were increased in size for the desktop view so that the application would look balanced.

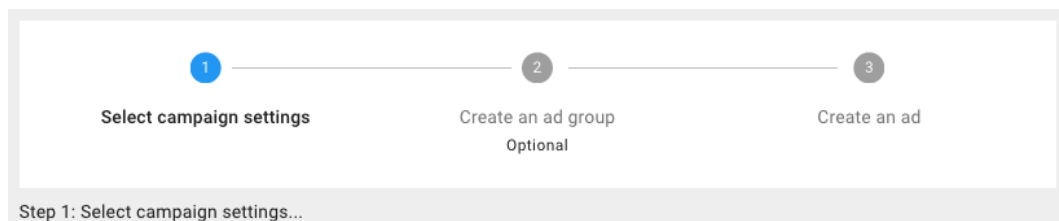


Figure 4.3: A Material UI stepper [27]

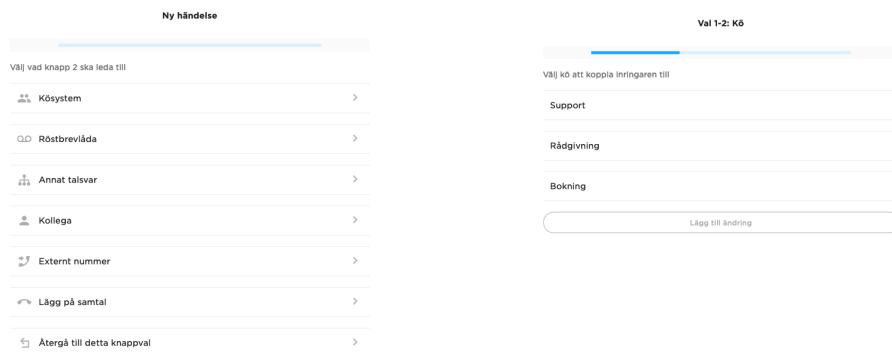
Some features that had not been implemented in the first version were also added, such as handling sounds and deleting submenus. Updating state and properties was tweaked. A button leading to a view of the tree was added.

4.2.2 Result

The final result of the implementation was made to look like prototype B (Figure 3.1), with the addition of a navigation bar of sorts at the top of the page. The menu pages of the application can be seen in Figure 4.5 and the pages to create a new node can be seen in Figure 4.4. We also added a way to visualize the tree. How this was done can be seen in Figure 4.6.

4.3 Evaluation phase

The final product from this iteration was evaluated in the same manner as the result from the previous iteration.



(a) Choosing node type

(b) Choosing a queue



(c) Reviewing and finalizing node creation

Figure 4.4: Creating a new node with progress bar

4.3.1 Method

We wanted to make this evaluation phase similar to the previous evaluation phase described in Section 3.3. This to make the comparison between the two implementations as fair as possible.

Heuristic evaluation

As a first step in the evaluation phase, we contacted an expert to evaluate the IVR solution. This time it was the UX department at Telavox who evaluated it. They can be considered experts in human factors as well as in the domain area, which makes them a so called “double expert”. The test was performed by letting the UX department test the IVR solution freely. There was no more scheduled implementation time when the heuristic evaluation was completed, but since we found the feedback to be sensible and would give our solution a

real boost in usability, we added a few days extra of implementation to resolve it before the test was sent out to the advisors and the lead technicians of the company.



Figure 4.5: Prototype B implemented

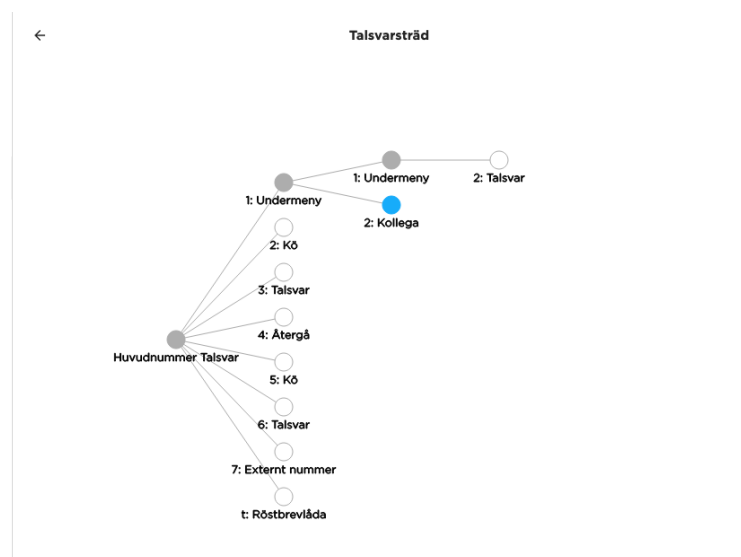


Figure 4.6: Tree View

Catastrophe insurance

At the same time as the heuristic evaluation was being performed, we held a catastrophe insurance. This to, with the help from the developers at Telavox, find hidden bugs before the solution reached the customers. It was sent out on the beta branch of the web app. The test was done as described in Section 3.3.1. This test resulted in some feedback which we improved on before the next step in the evaluation phase.

Google Analytics

Some additional measurement points were added to the code before any of the testing was done. These measurement points represented the new features which were added this iteration to see if the features were used.

The questions we want to answer with Google Analytics this iteration are:

- How far down the tree does the user go?
- Does the user look to see the possibility to add a new node?
- Are changes made? Which ones?
- Is it used mainly via mobile?
- What is the engagement on mobile versus desktop?
- Are the shortcuts being used?
- Is the tree view shown?

To answer these questions, we again set up some goals. The percentages in the goals were decided on by the authors. For the first four items in the list, we used the same goals as in Section 3.3.1.

- To evaluate if shortcuts were being used sufficiently, we wanted at least 25% of the users on trees with more than one level to use them.
- We wanted 25% of users to show the tree view to see if the button leading to it was visible enough or not.

Google form

A Google form was sent out to the testers of the solution to gather the feedback that can not be gathered from Google Analytics. This form consisted of a SUS questionnaire and some additional qualitative questions. As stated previously, we wanted continuity between the evaluation phases of the two iterations. This

meant that the questionnaire was identical to the form in Section 3.3. It can be seen in its entirety in Appendix A. This also meant that we wanted to answer the same questions as in Section 3.3.1. This to make it easy to evaluate the design against each other for a sort of comparison testing.

The form was again sent out to the advisors and the lead technicians at Telavox.

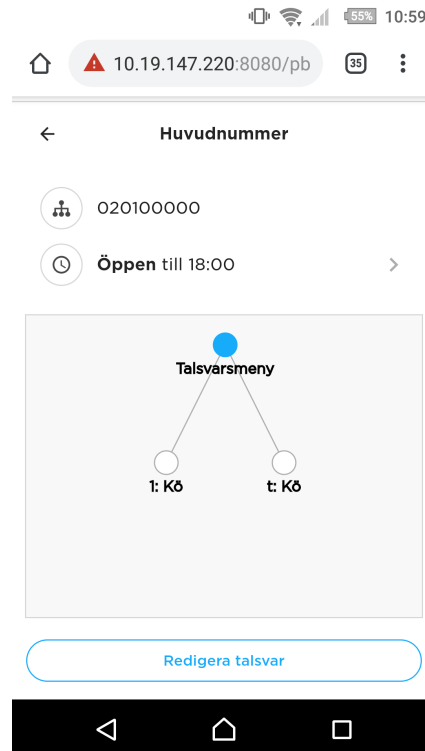


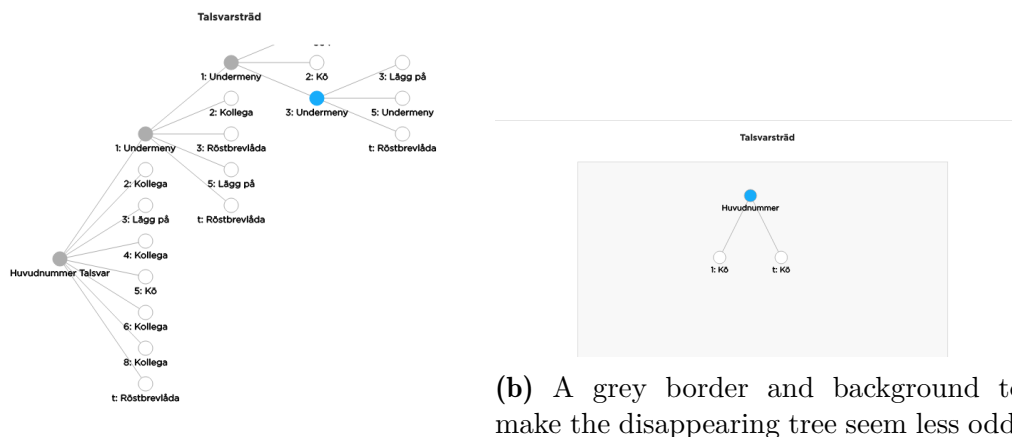
Figure 4.7: The landing page on a phone

4.3.2 Result

From the heuristic evaluation, we got the feedback to add a landing page containing the customers IVR tree before the page where the IVR can be edited. This can be seen in Figure 4.7. We also got the feedback to use auto-back on selection when e.g. editing a node. This to make the IVR solution more consistent to the rest of the application. Some feedback was also received on formulations on our help texts. All of this feedback was incorporated into the IVR solution.

From the developers in the catastrophe insurance testing, we got some feedback that the tree view button was a bit hard to find. We also got feedback that it was confusing when the tree view disappeared because of the boundaries of the content pane in the application as in Figure 4.8a. This was rectified by

adding a light grey background and border, as in Figure 4.8b.



(a) Tree disappearing in the top of the image.

(b) A grey border and background to make the disappearing tree seem less odd.

Figure 4.8: The problem and solution of the disappearing tree.

From the qualitative questions on the form, we got some feedback. One thing that was mentioned was that we had missed changing one helptext which referred to a node. This was seen as confusing. One person also wanted to be able to search for a certain colleague, or the phone number of a colleague, when adding them to a node. Otherwise the feedback was really positive.

The SUS form got 8 answers. The score was then calculated individually for the people who answered and then an average score was comprised. The final SUS score was *68.1*. Also according to Bangor et al. [21] is over 68 and can therefore be considered a usable product and to the Adjective Rating Scale [22] is right under “Good”. We also calculated a confidence interval with a confidence level of 95% which gave us a error margin of ± 4.699 .

When a measurement point from Google Analytics is triggered, it is presented as an event representing that specific measurement point.

The percentage of mobile events was lower this time, only 5.3%, perhaps due to less time. 84% looked at creating a new node, this rose to 94% after removing the Telavox number, which was in use and should not be edited unnecessarily. 52% made changes, 59% without Telavox number. 26% went to a lower level, 42% when only counting the trees with more than one level. 26% showed the tree view from parts of the tree. 37% used the quick navigation, or 58% when only counting the trees with a lower level. No changes or attempts at changes were made on mobile.

Tree	1	2	3	4	5	6	7	8	9	10
nbr. of levels in tree	2	1	?	1	1	2	?	?	2	1
nbr. of events entering this tree	3	2	2	2	2	2	2	2	1	1

Table 4.1: Table of the trees visited in Iteration 2

The results can be seen in their entirety in Appendix C.

4.4 Discussion

We did not get that much feedback from the qualitative questions during this iterations, but the feedback we received was useful. If we would have done another iteration this feedback would have been considered.

The SUS score was over 68, which indicates that the improvements made during this iteration made an improvement on the overall usability of the solution. The score did not reach the internal goal of the adjective rating scale [22] of “Excellent”, which would have been a SUS score of 85 or higher. However it can still be considered an acceptable score.

According to the Google Analytics the number of users that viewed the tree view was just above 25%, but to be able to use it as a good way to get to know exactly where in the tree the user is it should probably have a more visible button or be located in another place.

The changes were way over the set goal, which hopefully means that the stepper and throw away changes button helped the testers feel sure of what they were doing. Looking at creating a new node was a good bit over the goal, which compared to last iteration is a great improvement. No one used the shortcuts in the beginning, but this could be due to most of the tested trees being only one level, and therefore the shortcuts being hidden. However, some of the testers in the end of the testing period used the shortcuts so this measurement also reached its goal.

Chapter 5

Discussion

This chapter will discuss the implications of the results of this thesis work. It will discuss the advantages and disadvantages of the methods used and also what we have learned by using them. Suggestions on future work will also be made.

5.1 Iteration 1 vs Iteration 2

The difference between iteration 1 and 2 is the design of the submenus, the two different kinds of steppers added in iteration 2, as well as some functionality. Iteration 2 kept a lot of the features from iteration 1, but with a lot of rework on the design.

From the qualitative questions on the survey, we found that iteration 1 was very lacking when it came to usability. Iteration 2 did not get nearly as much constructive criticism as iteration 1. It was also a lot less extensive, from which we drew the conclusion that the second iteration did not have as many frustrating or confusing aspects as iteration 1.

The SUS score was very different depending on the iteration. It was significantly higher in the second iteration compared to the first, rising to an acceptable score. It went from *51.875* in the first iteration to *68.1* in the second iteration, which is quite a large improvement. This indicates that the IVR solution went from not usable to usable in one iteration.

As for Google Analytics, iteration 2 did not improve all the goals. Especially the goals of how many were using the application on mobile as well as how many were going to a lower level went down. The engagement on mobile

would hopefully improve when including the application into the mobile applications. As for going to a lower level, this could be due to different trees being visited.

A goal that was greatly improved by iteration 2 was if the users were making changes. This measurement went up by a factor ten. The goal of looking into creating a new node also made a small improvement, as it more than doubled. Goal 3 of this iteration, showing the users the progress when creating a new node, seems to have had an effect.

5.2 Fail-fast

While using the fail-fast method we found some advantages and some disadvantages.

One advantage was the time we saved on not inviting people for focus groups or going to the clients office and observing. When working with a more classic design process these costs would have been guaranteed, while fail-fast has a low cost due to more guess-work. We also did not need to spend time and energy in creating and testing low- and high fidelity prototypes. Another advantage was that once a working product was produced, it was not as costly as we predicted to rework the design. This was in fact done quite smoothly.

The disadvantages are that without design experience or an understanding of the clients needs, the first design could be way off mark, which could lead to more work down the line. Since we chose to work closely with the UX department at Telavox, we got a understanding of the clients via their experiences. Companies with established customers might not be affected by this disadvantage as much as startup companies, but it might not be the case for all companies.

Another disadvantage that can occur is when choosing one design and doing evolutionary prototyping on it can lead to the design reaching a local maximum of optimal design. A more open brainstorming process could find a possible global maximum, as mentioned in Henrik Kniberg's presentation about how Spotify builds products [3].

Since the first iteration had a strong focus on making a working solution and releasing it, the design did not become the most beautiful. If more focus was put on the design and not making something that was also easy to implement, a better design may have been found.

5.3 Gathering feedback

We focused on a scalable way of gathering feedback. This to see if it was a viable way for a company to work or if there were some drawbacks.

One aspect which was difficult when gathering feedback remotely was the instructions. It was quite difficult to give directions on an appropriate level. The text had to be detailed enough to not assume the people testing knew too much and at the same time not full of a lot of explanatory text which would be skipped leading to important information being lost. For example, to make the secret URL more clear in the second iteration, we included an example URL where some characters, marked as XXXXX, were to be replaced with the ID number of the IVR. But this led to some testers going to the URL without replacing the X:es.

Another aspect which had to be considered was when the instructions should be sent out to maximize the feedback. Variables which had to be considered was which weekday, which time of day and if the people were likely to be on holiday or not. We noticed in Google Analytics that if the testers did not have time to test when they received the e-mail, they were unlikely to test at all without further prompting. Therefore, some thought should be put into when potential testers are likely to have time to test the application.

5.3.1 Heuristic evaluation

The heuristic evaluation is the one evaluation method we used which is not scalable. This due to the fact that if too many experts are used, they will eventually not discover any new usability problems [29]. It is therefore not beneficial for the project to use this method in a scalable manner. However, since it meant a one time occurrence with the expert and since it produced such qualitative and usable results, we considered it a viable evaluation method for us to use even though it was not scalable.

We found that the result from the evaluation differed a lot depending on who the evaluator was. The way the produced feedback differed was the area the evaluators focused on. The first evaluator was new to the system and produced feedback which focused on how new users see the solution, while the second evaluation was done by people who was experienced with the rest of the system, but not our solution. This produced feedback which focused on making our solution a part of the rest of the system. Based on this information, it could have been useful to have both kinds of evaluators evaluating both iterations, to be able to receive both kinds of feedback for each iteration. However, the difference in the feedback could also have been a result of the solution appearing less finished in the first iteration and in need of more feedback which

would help the user. This was then not as much of a problem in iteration 2 which gave the evaluators more freedom to notice other improvements.

One important thing we noticed when doing the heuristic evaluation was to have some time scheduled afterwards to implement the feedback which came from the evaluation. Almost every comment which were brought up in the evaluation were, from our experience, worth implementing. We found that the feedback from the evaluators were things we agreed with them on, and in some cases were things we had thought of but forgotten. Many comments were also things we had missed due to the fact that we were used to our own system, but when pointed out it became apparent that it needed improvement. This was namely naming of items and help texts, which needed improvements in both iterations.

5.3.2 Catastrophe insurance

We found the catastrophe insurance to give us some peace of mind before sending the solution to customers, or in our case testers. What was mentioned in Section 2.6.2 *Catastrophe insurance* was correct, namely that the test would find bugs which the developers might have missed. We were then fortunate to be able to fix the bugs that came up, but if this were not the case we would at least have known of the bugs existence. Even if the people performing the catastrophe insurance did not find all of the bugs in the IVR solution, they found more bugs than the developers alone. We also sent out an survey with the catastrophe insurance where we asked for feedback on the design. In the first iteration, the people performing the test did not give any design feedback, they only found bugs. However, in iteration 2 the people performing the test came with some feedback regarding the design of the solution. This might have been due to the fact that less bugs appeared in the second iteration and that gave the people performing the test more room to evaluate the design. It could also be that the people performing the test was more used to the IVR solution in the second iteration and felt more comfortable giving feedback.

5.3.3 Google Analytics

We found from the Google Analytics measurement points that there were a large number of people testing our solution who did not give feedback via the form. Without those measurement points, we would not have known that. The measurement points gave us the possibility to measure every step the user took, which made it easy to collect exactly the data we needed to evaluate the solution. However, the fact that we could measure everything made it harder to decide exactly what data we needed to collect. To decide this,

we formulated some questions we wanted to answer with the test, as seen in Section 3.3.1. From these questions we calculated which measurement points that would answer them and implemented those. Since everything was possible to collect with these measurement points, the Google Analytics website was at times very complex with a large amount of features we did not use. This made the tool a bit harder to use at first, but after some practice it got easier. Someone with a lot of experience in using Google Analytics may use it to get even more advanced information than we did, making it a very handy tool to gather feedback.

5.3.4 Survey

We chose to have a SUS survey and some additional qualitative questions in the same survey due to the fact that we only had three additional qualitative questions and because the SUS survey itself is quite short. We also did not want the people testing to have to navigate multiple ways to give their feedback. Even with this, there was not a lot of answers to the survey and it was good that we had multiple ways to gather user data.

We tried to make the qualitative questions very open as to not steer the users answers, but that also meant that we didn't get any answers about the more specific questions we were asking about, the response times and the icons. Therefore, we had to assume that since they were not mentioned they were acceptable. The questions asked "Did you experience anything as confusing" and "Did you experience anything as frustrating" gave us a lot of useful feedback which we would not have gotten via the SUS score alone. Even though the questions were quite similar, they provided differing feedback. They also both provided the appropriate feedback to make further developments to the solution.

When evaluating the SUS score we relied on two different evaluation techniques. One was the more common "over 68 rule", which stated that if a product got a SUS score of over 68, it was probably a usable product, and if it got a SUS score of under 50, it was probably a very unusable product. However, we did not think this method gave a very nuanced picture of the product, and therefore we chose to complement this technique with a second technique, the "adjective ratings scale". Together, these techniques gave an indicator of how to best interpret the SUS score.

One thing that stood out in the SUS surveys were that a lot of the testers, even if they were otherwise positive, gave very low scores on the first statement of the survey. This statement, "I think that I would like to use this system frequently", might have caused a slightly lower SUS scores for our application. The application's main feature, editing an IVR system, is not something that is supposed to be used frequently.

Other forms of surveys

Another scalable method of gathering feedback is having a small survey or rating directly in the application, so that after a certain amount of time the user is prompted to review the application. One kind of rating method which was considered was having a thumbs up or down. This would give the user a quick and easy way to indicate if they liked the solution or not. However, this would not have given us the qualitative feedback we needed to make improvements to the design which means that there would still be necessary to have some sort of survey. We considered having both thumbs and a survey for a while but after some research we found that too many feedback options most likely would give the testers respondent fatigue [30], a phenomenon when the testers are tired of giving feedback and therefore do not give as good feedback. To avoid this, we decided to gather the qualitative and quantitative feedback in the same Google Form to make sure the testers only had one place for giving feedback.

5.4 Further improvements

An IVR solution has the possibility to have a lot of and very complex features. Therefore the thesis work had to narrow down the amount of features to be able to keep the time plan of 20 weeks. In this section we will discuss some features that would have been implemented and general improvements that would have been made if the time frame was different.

5.4.1 Sounds

At the moment a customer can choose to add a sound to be played before a certain submenu or a certain node. However, some features are missing which could lead to large improvements to the product.

Adding new sounds

When choosing a sound to be played, the customer can choose from a list of sounds which are connected to this specific user. However, in the future it would be added that the customer could add a new sound to this list. Since the platform is mobile first, it could also be added that the customer could record their own sound using the microphone on their mobile phone.

Rearranging sounds

In the final implementation, the customer is able to add multiple sounds to the list of sounds playing before the submenu or node. They can also remove a specific sound from the list of sounds playing. As a further development of this, the customer should be able to rearrange the order of specific sounds in this list. This would lower the amount of effort significantly for the customer if, for example they accidentally added the sounds in an incorrect order.

5.4.2 More than one submenu

One thing we would have changed if we got to redo this thesis work is the way we handle submenus. At the moment when a submenu is chosen, we search the list of possible choices for the current level in the tree of which the user is located and find a submenu to navigate to. This means that there can only be one submenu at each tree level.

If more time were given, the code could be rewritten to incorporate a depth first search to more fully map the tree. Depth first search is a class of algorithms that allow for complete traversal of a graph, and as a tree is just a special case of graph, it would work in this case [31].

5.4.3 More than one profile

As a standard, a customer at Telavox with an IVR receives two profiles, open and closed. This can be expanded into more profiles at the customers request. This regulates which actions shall be taken when a caller reaches the IVR. A customer can set during which days in the week and during which time frame a certain profile shall be active. The “Open” profile is the one we have implemented and consists of a tree. However the other profiles proved to be a lot more complex due to the fact that they did not consist of a tree. For example, when the “Closed” profile is active the caller will be sent to an end point directly without them pressing a button on their phone. This could seem straight forward, but due to the fact that the profiles was constructed a lot differently in the database it would have taken a lot of time to solve.

5.4.4 Creating a new IVR

At this point, a customer can only view and edit an already existing IVR system. As a further development, this could be extended to creating a new IVR system. This feature would give the customer more liberties in the product and make them less bound to the desktop application or the advisors at Telavox.

Chapter 6

Conclusion

This chapter will draw conclusions based on the research questions stated in Section 1.3.

After two iterations of designing, developing and evaluating an application for viewing and editing an IVR-system, a lot of insights into gathering feedback and working with a fail-fast method in design have been reached. The evaluation phase in both iterations included scalable methods of collecting feedback. Each method were intended to gather different kinds of feedback to help with the further development of the product.

How can feedback be collected in a scalable way when developing a mobile-first application?

We found it quite challenging to get feedback from real customers, especially with surveys. This due to the fact that a sent out survey is less personal, which yields less incentive to actually give feedback. In the end there was only a small percentage who actually gave their feedback. However, if this had been done in a larger scale, the small percentage would still have yielded a lot of useful feedback to the company. We also did not have a lot of people testing via their mobile phone. This might have been a higher percentage if we sent out a direct link to their phone instead of via email.

The scalable ways we collected feedback were all useful in their own way. Every method brought their own sort of feedback which were all useful.

What are the challenges when working with a fail-fast design process?

One challenge is how to not reach a local max instead of a global max. It is quite difficult to know if the beginning is too narrow and therefore will not reach the best product after some development.

Another challenge is knowing when to stop. When producing a MVP, it can be hard to know if the implementation is enough to produce results and at the same time not implementing it fully.

If the company does not know which customers to focus on, or their customers' needs, they might want to do some sort of pilot study before the start of the implementation. This to have some indication of where to start. If a company already have established customers, this might not be necessary. This due to the fact that they already have knowledge of what their customers want. We found that we had a lot of knowledge from Telavox about their customers to support our starting designs, which meant that a pilot study was not necessary.

Bibliography

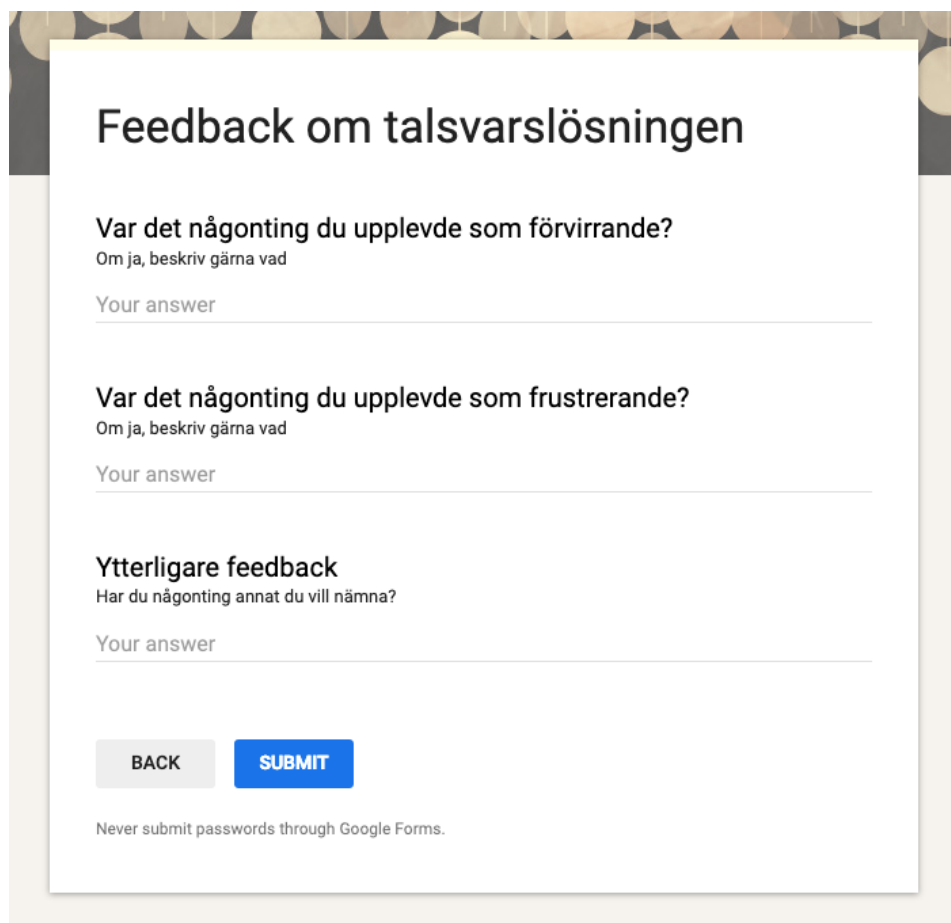
- [1] International Organization for Standardization. *Ergonomics of Human-system Interaction: Part 210: Human-centred Design for Interactive Systems*. ISO, 2010.
- [2] Laura Hokkanen, Kati Kuusinen, and Kaisa Väänänen. Early product design in startups: Towards a ux strategy. In *Product-Focused Software Process Improvement*, pages 217–224. Springer International Publishing, 2015. ISBN 978-3-319-26844-6.
- [3] Henrik Kniberg. How spotify builds products, 2013. https://www.dcc.fc.up.pt/~mcoimbra/lectures/SIM_1415/SIM_1415_T2.3_HowSpotifyBuildsProducts.pdf. Fetched 2019-01-16.
- [4] Donald A. Norman. *The Design of Everyday Things*. The MIT press, Cambridge, Massachusetts, 2013.
- [5] Peter Merholz. Peter in conversation with don norman about ux & innovation, 2007. <https://www.adaptivepath.com/ideas/e000862/>. Fetched 2019-04-16.
- [6] Laura Klein. *UX for Lean Startups: Faster, Smarter User Experience Research and Design*. O'Reilly Media, Inc., 2013.
- [7] Mattias Arvola. *Interaktionsdesign & UX: Om att skapa en god användarupplevelse*. Studentlitteratur AB, 2014.
- [8] Jennifer Preece, Yvonne Rogers, and Helen Sharp. *Interaction Design: Beyond Human-Computer Interaction*. John Wiley and Sons, third edition, 2002.
- [9] Martina Maria Keitsch. Design driven innovation - minimum viable products for local entrepreneurship in nepal, 2015.
- [10] Siddharth. How to go from a minimum viable product to a minimum marketable product. <https://www.snyxius.com/how-to-go-from-a-minimum-viable-product-mvp-to-a-minimum-marketable-product-mmp/>. Fetched 2019-01-30.

- [11] M O'Reilly. *Extreme Programming Pocket Guide*. O'Reilly Media Inc, 2003.
- [12] Laura Hokkanen, Kati Kuusinen, and Kaisa Väänänen. Minimum viable user experience: A framework for supporting product design in startups. In *Agile Processes, in Software Engineering, and Extreme Programming*, pages 66–78. Springer International Publishing, 2016. ISBN 978-3-319-33515-5.
- [13] React - a javascript library for building user interfaces.
- [14] Javascript. *Collins English Dictionary – Complete & Unabridged 2012 Digital Edition.*, 2012. <https://www.dictionary.com/browse/javascript>. Fetched 2019-04-29.
- [15] Dana Chisnell and Jeff Rubin. *Handbook of Usability Testing: How to plan, design and conduct effective tests*. Wiley Publishing, Inc., 2008.
- [16] Rolf Molich and Jakob Nielsen. *Heuristic evaluation of user interfaces*, 1990.
- [17] Google. Analytics. <https://marketingplatform.google.com/about/analytics/>, 2019. Fetched 2019-03-14.
- [18] John Brooke. Sus - a quick and dirty usability scale. in jordan, pw, thomas, b., weerdmeester, ba & mccllland, al (eds.) *usability evaluation in industry*, 1996.
- [19] Thomas S Tullis and Jacqueline N Stetson. A comparison of questionnaires for assessing website usability. In *Usability professional association conference*, volume 1. Minneapolis, USA, 2004.
- [20] Jeff Sauro. *A practical guide to the system usability scale: Background, benchmarks & best practices*. Measuring Usability LLC Denver, CO, 2011.
- [21] Aaron Bangor, Philip Kortum, and James Miller. An empirical evaluation of the system usability scale. *International Journal of Human-Computer Interaction*, 24(6):574 – 594, 2008. ISSN 10447318.
- [22] Aaron Bangor, Philip Kortum, and James Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of usability studies*, 4(3):114–123, 2009.
- [23] How to facilitate a brainstorming session: The effect of idea generation techniques and of group brainstorm after individual brainstorm. *Creative Industries Journal*, 11(3):263 – 277, 2018. ISSN 17510694.

- [24] Paul B Paulus and Huei-Chuan Yang. Idea generation in groups: A basis for creativity in organizations. *Organizational behavior and human decision processes*, 82(1):76–87, 2000.
- [25] Jonali Baruah and Paul B Paulus. Effects of training on idea generation in groups. *Small Group Research*, 39(5):523–541, 2008.
- [26] Alex Osborn Faickney. Applied imagination. *New York.: Scribner*, 1957.
- [27] Matt Brookes. Steppers, 2018. <https://material-ui.com/demos/steppers/>. Fetched 2019-04-16.
- [28] Ben Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition, 1998.
- [29] Jakob Nielsen. Finding usability problems through heuristic evaluation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 373–380, 1992.
- [30] Pazit Ben-Nun. Respondent fatigue. *Encyclopedia of survey research methods*, 2:742–743, 2008.
- [31] Shimon Even and Guy Even. *Graph Algorithms*. Cambridge University Press, 2011. URL <https://books.google.se/books?id=m3QTSMYm5rkC>.

Appendix A

Evaluation form



Feedback om talsvarslösningen

Var det någonting du upplevde som förvirrande?
Om ja, beskriv gärna vad

Your answer

Var det någonting du upplevde som frustrerande?
Om ja, beskriv gärna vad

Your answer

Ytterligare feedback
Har du någonting annat du vill nämna?

Your answer

BACK **SUBMIT**

Never submit passwords through Google Forms.

Figure A.1: Qualitative questions

Can y... Inbox... Analyt... Exjobb... SUS... Fe X... hrz.m... Saker... Exjobb... Telav... Goog... Proj... Telav... Fil... Does... +

https://docs.google.com/forms/d/e/1FAIpQLSckGZQvKsrBenBkxHV6NhTema...

* Required

1. Jag tror att jag skulle vilja använda appen regelbundet. *

1 2 3 4 5

Håller inte med alls Håller med helt

2. Jag tycker att appen är mer komplicerad än vad den behöver vara. *

1 2 3 4 5

Håller inte med alls Håller med helt

3. Jag tycker att appen är lätt att använda. *

1 2 3 4 5

Håller inte med alls Håller med helt

4. Jag tror att jag skulle behöva personlig teknisk support för att kunna använda appen. *

1 2 3 4 5

Håller inte med alls Håller med helt

5. Jag tycker att de olika funktionerna i appen fungerar väl tillsammans. *

1 2 3 4 5

Håller inte med alls Håller med helt

6. Jag tycker att det finns många saker som inte är konsekventa i appen. *

1 2 3 4 5

Håller inte med alls Håller med helt

7. Jag tror att de flesta skulle kunna lära sig den här appen snabbt. *

1 2 3 4 5

Håller inte med alls Håller med helt

8. Jag tycker att den här appen är besvärlig att använda. *

1 2 3 4 5

Håller inte med alls Håller med helt

9. Jag känner mig väldigt säker och trygg (på vad jag gör) när jag använder appen. *

1 2 3 4 5

Håller inte med alls Håller med helt

10. Jag behöver lära mig ganska mycket innan jag kan börja använda appen. *

1 2 3 4 5

Håller inte med alls Håller med helt

Figure A.2: SUS questions

Appendix B

Google Analytics Data Iteration 1

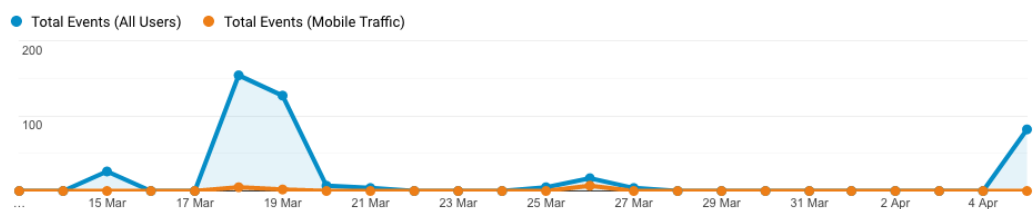


Figure B.1: The usage of the application during the first evaluation

Table B.1: Google Analytics data from the first evaluation

Beginning of Table			
Event Label	Segment	Total Events	Unique Events
Went to Node on Level 0	All Users	80	17
	Mobile Traffic	2	1
Clicked on IVR	All Users	61	27
	Mobile Traffic	6	4
Went Back When Creating New Node	All Users	44	12
	Mobile Traffic	0	0
Started Creating New Node on Level 0	All Users	37	13
	Mobile Traffic	0	0
Showed Sound in Submenu	All Users	30	13
	Mobile Traffic	1	1

Continuation of Table B.1			
Event Label	Segment	Total Events	Unique Events
Showed Ivr Tree in Refer With Key Refer-21878	All Users	29	17
	Mobile Traffic	5	3
Went to Parent on Level 1	All Users	24	6
	Mobile Traffic	0	0
Went to Next Step in Creating New Node	All Users	23	9
	Mobile Traffic	0	0
Showed Actions in Node	All Users	12	6
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1259423	All Users	11	1
	Mobile Traffic	0	0
Showed Sound in Node	All Users	10	6
	Mobile Traffic	0	0
Saved a Change on an Existing Node on Level 0	All Users	8	4
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1620303	All Users	8	1
	Mobile Traffic	0	0
Started Creating New Node on Level 1	All Users	7	3
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-2088065	All Users	6	2
	Mobile Traffic	0	0
Deleted a Node on Level 0	All Users	5	4
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1298062	All Users	5	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-2141030	All Users	4	1
	Mobile Traffic	0	0

Continuation of Table B.1			
Event Label	Segment	Total Events	Unique Events
Showed Ivr Tree in Refer With Key Refer-500241	All Users	3	3
	Mobile Traffic	0	0
Went to Node on Level 1	All Users	3	2
	Mobile Traffic	0	0
Created a New Node on Level 0	All Users	2	2
	Mobile Traffic	0	0
Saved a Change on an Existing Node on Level 1	All Users	2	2
	Mobile Traffic	0	0
Saved a Change on an Existing Node on Level 2	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1231433	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1620275	All Users	2	1
	Mobile Traffic	0	0
Started Creating New Node on Level 2	All Users	2	1
	Mobile Traffic	0	0
Created a New Node on Level 1	All Users	1	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1198275	All Users	1	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1585990	All Users	1	1
	Mobile Traffic	0	0
Went to Parent on Level 2	All Users	1	1
	Mobile Traffic	0	0
End of Table			

Appendix C

Google Analytics Data Iteration 2

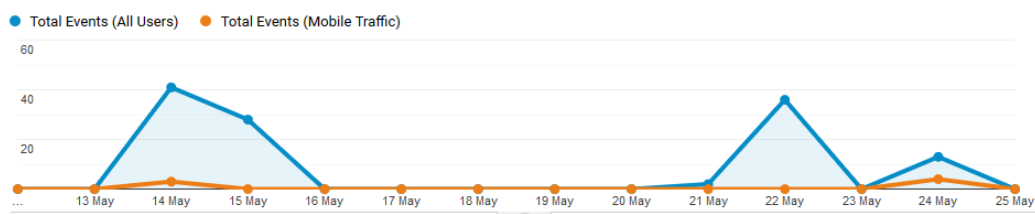


Figure C.1: The usage of the application during the second evaluation

Table C.1: Google Analytics data from the second evaluation

Beginning of Table			
Event Label	Segment	Total Events	Unique Events
Clicked on IVR	All Users	15	10
	Mobile Traffic	3	1
Went to Node on Level 0	All Users	15	8
	Mobile Traffic	1	1
Went to Next Step in Creating New Node	All Users	13	5
	Mobile Traffic	0	0
Started Creating New Node on Level 0	All Users	8	6
	Mobile Traffic	0	0
Showed Actions in Node	All Users	7	3
	Mobile Traffic	1	1

Continuation of Table C.1			
Event Label	Segment	Total Events	Unique Events
Started Creating New Node on Level 3	All Users	6	1
	Mobile Traffic	0	0
Went Back When Creating New Node	All Users	4	3
	Mobile Traffic	0	0
Showed Treemap From Submenu	All Users	3	3
	Mobile Traffic	0	0
Went to Parent on Level 1	All Users	3	3
	Mobile Traffic	0	0
Created a New Node on Level 0	All Users	2	2
	Mobile Traffic	0	0
Deleted a Node on Level 0	All Users	2	2
	Mobile Traffic	0	0
Saved a Change on an Existing Node on Level 3	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1193233	All Users	2	1
	Mobile Traffic	2	1
Showed Ivr Tree in Refer With Key Refer-1907065	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1922919	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-2000805	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-2056978	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-2180964	All Users	2	1
	Mobile Traffic	0	0

Continuation of Table C.1			
Event Label	Segment	Total Events	Unique Events
Showed Ivr Tree in Refer With Key Refer-2273675	All Users	2	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-500241	All Users	2	1
	Mobile Traffic	0	0
Showed Treemap From Node	All Users	2	1
	Mobile Traffic	0	0
Used Shortcut to Go to Parent on Level 0	All Users	2	2
	Mobile Traffic	0	0
Used Shortcut to Go to Parent on Level 1	All Users	2	1
	Mobile Traffic	0	0
Used Shortcut to Go to Parent on Level 2	All Users	2	1
	Mobile Traffic	0	0
Backed to Root of Refer With Key Refer-1907065	All Users	1	1
	Mobile Traffic	0	0
Created a New Node on Level 1	All Users	1	1
	Mobile Traffic	0	0
Created a New Node on Level 2	All Users	1	1
	Mobile Traffic	0	0
Created a New Node on Level 3	All Users	1	1
	Mobile Traffic	0	0
Saved a Change on an Existing Node on Level 0	All Users	1	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-1839897	All Users	1	1
	Mobile Traffic	0	0
Showed Ivr Tree in Refer With Key Refer-21878	All Users	1	1
	Mobile Traffic	0	0

Continuation of Table C.1			
Event Label	Segment	Total Events	Unique Events
Started Creating New Node on Level 1	All Users	1	1
	Mobile Traffic	0	0
Started Creating New Node on Level 2	All Users	1	1
	Mobile Traffic	0	0
Used Shortcut to Go to Parent on Level 3	All Users	1	1
	Mobile Traffic	0	0
Went Back to Tree Without Saving New Node	All Users	1	1
	Mobile Traffic	0	0
Went to Parent on Level 2	All Users	1	1
	Mobile Traffic	0	0
Went to Parent on Level 3	All Users	1	1
	Mobile Traffic	0	0
End of Table			