BSc in computer science

Talbankinn

Students
Bjarki Hrafn Axelsson
Leifur Pálsson
Smári Freyr Guðmundsson
Snorri Arinbjarnar

Instructor
Guðný Ragna Jónsdóttir
Examiner
Ásgeir Freyr Kristinsson

June 2019

Table of contents

1 Introduction

Talbankinn was created as a final project for a Bachelor's of Science degree at Reykjavík University by a team of four students that decided to work together because they have known each other since the first semester and have very good experience working on large scale projects together.

Talbankinn is a voice command service which was created for Landsbankinn. Almost everyone should be familiar with services such as Landsbankinn's Call Service Center. You call in to get basic information when you do not want to wait in line in a bank or wait on hold for the human service desk. This service could be much better. The phone service works like this: you call and have to wait for the friendly automated voice, who goes by the name of Karl reading up available services:

> *"Press 1 to hear about the exchange rates,*
>
> *press 2 to check your balance…*
>
> *press 3 for…"*

What if, instead of all of these endless delays you could just use a mobile app and say exactly what you want to know or do, a speech bot would find what you are looking for and answer you straight away. Landsbankinn and the development team believe that many people will jump on that boat because it will make banking quicker, more accessible and even enjoyable. They also believe this will be a good alternative to the Internet bank for visually and touch impaired people.

Say "No" to waiting, tell Talbankinn what you want and it will provide.

1.1 Landsbankinn

Landsbankinn is Iceland's largest financial institution. It has a software development department with around 100 employees. All their software, such as their Internet bank website and mobile app are developed internally.

1.2 The Problem

Landsbankinn has requested a solution for their customers that allows them to access the bank's services through voice commands instead of a website or a phone call. This will require the voice to be converted to a string which the application will process through AI technology and various APIs. When a request has been processed the string gets processed back through a text-to-speech software to inform the user of the software's response. This will open new

doors for disabled users. For example, blind users will have a much easier access to the bank's services using voice commands.

1.3 Overview

This final report will cover the project planning, the project analysis, the design, how the development progress went, the testing that was done and all the research that had to be done in order to create Talbankinn.

2 Project Planning

This chapter covers how the project was planned. It covers the methodology used when creating the project, how sprints were organized, the schedule of the sprints, when meetings were held, the project visualization, the different workspaces the project was worked on, rules regarding coding and the technical environment of Talbankinn.

2.1 Methodology

Agile methodology was chosen and the use of the Scrum framework. The decision was made because Scrum is simple, maintains discipline. The development team has had good experience using it and because it is the most popular framework for software development in Iceland [1].

2.1.1 Roles

Product owner: Landsbankinn hf
- **Contact:** Albert G. Jónsson

Scrum master: Smári Freyr Guðmundsson

Team members: Bjarki Hrafn Axelsson,
Leifur Pálsson,
Smári Freyr Guðmundsson,
Snorri Arinbjarnar

2.1.2 Communication Protocols

Since the team members were all still studying at Reykjavik University and their schedules did not differ too much, they were able to create a fixed schedule where all team members could meet up and work together, face to face. Since situations could occur where not all members

were able to attend a meeting, the person not attending would be given a task to work on outside the meeting.

2.1.3 Moral Principles

1. Every team member should show up at all meetings on the set time.
2. Every team member should work the set amount of hours.
3. No team member should ever be idle. There is always something to work on.
4. Negative influence will not be tolerated. If a team member is visually upset, he should leave the work space to cool off.
5. Breaks should be taken regularly, involve all team members and should be taken outside of the work environment.

2.2 Sprint Organization

Nine sprints were planned, including sprint zero. The first three sprints were two week long. During the exams it was assumed that not all members would be able to attend the scheduled meetings and because of that the fourth sprint was three weeks long. After the exams members were be able to focus fully on the project which is why the last four sprints were one week long sprints with six days of work, except the last one. The eighth and last sprint lasted 12 days. It was originally two sprints but was merged into one large sprint. Each sprint was started with a planning session, included daily scrum meetings and were always concluded with a retrospective meeting to go over the results of the sprint.

2.2.1 Velocity

The team's capacity was calculated as follows. Each workday is eight hours, two days a week, two weeks for each sprint, and with a four man team. That makes 8 * 2 * 2 * 4 = 128. To take into account meetings, breaks, etc. That number is multiplied by 0.75 which gives 96 hours per sprint or circa 24 hours per day.

To calculate the velocity three general stories with common weights, 3, 5 and 8 story points were divided into tasks. Afterwards, each tasks duration was estimated in hours. The total amount of hours to finish the tasks was estimated to be 56 hours. That is 56 hours for total of 16 story points. That results in 56h / 16p = 3.5 hours to complete one story point.

The initial velocity was calculated to be 96 hours / 3.5 hours per point = 27.4 points per sprint. The final sprints, after the exams, had more capacity. Each was one week long and had a capacity of 41.1 story point per sprint. That is (8 * 6 * 1 * 4) * 0.75.

The final sprint, sprint #8 was originally intended to be the second to last sprint, with sprint #9 as the last one. In the final days of the project, it was decided to merge sprint #8 and #9 into one large sprint #8 where the main emphasis was on optimizing already implemented actions and finishing documentation. Its capacity was calculated in similar manner to be 62 story points.

The initial plan was to work on the project for three days per week but in the beginning of sprint #2, it was decided to decrease it to two days per week. The reason was obligations in other courses. Thus, during the first two sprints the velocity was 41.1 story point per sprint. In sprint #2, although the initial velocity was 41.1, the team worked as if the velocity was 27.4 story points per sprint. After the exams, there was more time. Thus, in each one week sprint it was planned that 41.1 story point should be finished. In the final sprint, which was longer than the others, the planned amount of story points to be finished was 62 story points.

The table below describes the duration of each sprint along with which stories were worked on in each one. The description column contains short descriptions of what was done in the sprints. Stories that were not finished in sprint *x* were usually continued on sprint *x + 1*. Scrapped stories are marked bold. Although they were scrapped, they might have been finished, see further in progress overview chapter (ch. 8).

#	Duration	Length	Stories	Description
0	27.01 - 02.02	1 week	n/a	Sprint zero. General planning and project analysis.
1	03.02 - 16.02	2 weeks	1,3,4,6,16,17	Research of various methods available to implement product.
2	17.02 - 02.03	2 weeks	6, **28**, **29**, 30, **31**, 32, **13**	Attempted implementation of Talbankinn as Google Assistant Action.
3	03.03 - 16.03	2 weeks	6, 32, 33, 34, 10, 11, 14	Serious implementation of final API design begins.
4	17.03 - 13.04	3 weeks	32, 33, 34, 11, 14, 35	(Wrap-up of other courses + Exam period)
5	14.04 - 19.04	1 week	32, 33, 11, 9, 12, 15	Continued implementation of product. Materialization of first user stories. Slow start after exams.
6	20.04 - 27.04	1 week	32, 8, 18, 35, 36, 19	Materialization of multistep user stories, user testing and implementation of multiple user support.
7	28.04 - 04.05	1 week	8, 18, 35, 19, 42, 37, 38, 39, 40	First full speed post-exam sprint. A few small researches as well as continued implementation of mostly multistep user stories. More user testing.
8	05.05 - 16.05	1.5 week	35, 37, 43, 44, 45	Feature freeze effective immediately. Product refinement and increase of robustness. No new user stories added. Clean up. Code freeze in end of 14.05.

2.3 Sprint Schedule

The fixed schedule working hours 27.01 - 14.02 (sprint #0 - #1):

- Sundays: 12:00 - 20:00
- Mondays: 12:00 - 20:00
- Thursdays: 12:00 - 20:00

The fixed schedule working hours 17.02 - 12.04 (sprint #2 - #4)

- Sundays: 12:00 - 20:00
- Mo
- Language: English for documentation,
  Icelandic for LUIS and the names of icelandic declensionsndays:
  12:00 - 20:00
- Thursdays: Attendance if deemed necessary

  Sprint #4 was during exams and there was no strict schedule.

The fixed schedule working hours 14.04 - 16.05 (sprint #5 - #8):

- Monday - Friday: 10:00 - 18:00
- Saturday 12:00 - 20:00
- Sunday Attendance if deemed necessary

2.4 Meetings

There were regular meetings every week:

- Landsbankinn: Mondays 15:30 - 16:30
- Instructor: Mondays 12:00 - 12:30

2.5 Project Visualization

Scrum was used to visualize the project. For each sprint, there was a board that consisted of four states: *Stories*, *Tasks*, *In-progress* and *Done* was created. Stories contained the story description while Tasks contained the stories' tasks. Trello was used as a tool to create the

boards. A product backlog that consisted of all the stories arranged by their priority was also created. See chapter 3.3.

2.6 Workspace

Two different workspaces were used for the development of Talbankinn. The first one was in the master students' work area within the department of computer science at Reykjavík University. There, the team was granted access to a large table where everyone could sit together, a whiteboard and some conference rooms which could be reserved for few hours a day. The latter workspace was not available until the project was halfway done. That workspace was at Fjártækniklasinn at Laugavegur 77. There, the team had access to four tables next to each other in an open space where everyone could sit together. There was also a comfortable shared conference room.

2.7 Code rules

To prevent style discrepancy and poor code quality the following rules were set:

- Casing
  - Variable casings: camelCase
  - Function casings: camelCase
  - Class casing: PascalCase
- Indentation
  - JS: 2 spaces
- Curly braces
  - JS: K&R
- Comments
  - ```
    /*
     * multi
     * line
     * comment
     */
    ```
  - `// single line comment & commenting out code`
- Other
  - File names: camelCase
  - Folder names: PascalCase
    ```

2.8 Technical Environment

Talbankinn API and Mock Data API were written in NodeJS using Visual Studio Code and were deployed to Microsoft's Azure cloud. Talbankinn API speaks to multiple other APIs such as Google's speech-to-text API, see more about API connections in chapter 5: Design. The mobile app is written in JavaScript using the React Native library. Visual Studio Code source-code editor, along with Android Studio were the main development tools for the mobile app.

3 Project Description

Landsbankinn's intent was to develop a speech bot which enables users to have verbal communication to perform simple actions within the bank's infrastructure. Such actions range from asking for share rates to transferring funds from one account to another.

3.1 Main Goal - Minimum Viable Product

The goal was to have a mobile app which enables users to ask for simple information such as share rates as one step queries with no data mutations. The second step is multi-step conversations with data mutations. For example, the user should be able to transfer money from one account to the other.

3.2 Secondary Goals

When everything had started to come together, different ways to implement authentication and security for the service was researched. The plan was also to look into Cordova plugins in order to integrate the solution to Landsbankinn's mobile app but due to lack of time it was not done.

3.3 Stories

The following table is the project backlog and displays all the stories which were created by analysing and discussing the project description set by Landsbankinn. It contains all stories that were made throughout the project, both scrapped ones and not. The stories' weights are measured in story points. A planning poker session was held to determine the number of story points for each story in a concise and accurate manner [2]. The numbers inside the parentheses in the Status column denote in which sprint the story gained the status in question. There were three types of stories, *Research, Developer* and *User* stories. Note that this backlog was a subject to regular changes and revisions during the project, mainly between sprints.

#	Priority	Status	Type	Description	Pts.
1	**A**	**Done(1)**	Research	As a developer, I want find out whether RU AI or Google AI is more suitable for the project.	3
28	**A**	**Scrapped(2)**	Developer	As a developer I want to set up Talbankinn Webhook fulfillment service	8
29	**A**	**Scrapped(2)**	Developer	As a developer I want to deploy Talbankinn Webhook fulfillment service	3
30	**A**	**Done(2)**	Developer	As a developer I want to create mock database + data	5
31	**A**	**Scrapped(2)**	Developer	As a developer I want to bypass Dialogflow	3
32	**A**	**Done(6)**	Developer	As a developer I want to receive sound from AWS Polly and route it to app	5
3	**A**	**Done (1)**	Research	As a developer I want to research the best suitable Language Understanding platform (Luis ui, Botengine etc.)	3
4	**A**	**Done(1)**	Research	As a developer I want to research the best suitable Text-to-Speech tool: AWS Polly - Karl vs Dóra, Blindrafélagið's Jóhanna	5
6	**A**	**Done(3)**	Research	As a developer I want to research how to create a skill on Amazon or Google	5
8	**A**	**Done(7)**	User	As a user I want to transfer money from my account to a known recipient	5
9	**A**	**Done(5)**	User	As a user I want to get overview of unpaid bills	3
10	**A**	**Done(3)**	User	As a user I want to know exchange rates	3
11	**A**	**Done(5)**	User	As a user I want to know the share rates	3
12	**A**	**Done(5)**	User	As a user I want to get information about a specific unpaid bill	3
13	**A**	**Scrapped(2)**	User	As a user I want to connect to Talbankinn through Google Home	13
14	**A**	**Done(4)**	User	As a user I want to know the balance of my accounts	3
15	**A**	**Done(5)**	User	As a user I want to know the status of my credit card (funds available + balance)	3
16	**A**	**Done(1)**	Developer	As a developer I want to be able to deploy changes directly to a cloud service	8
17	**A**	**Done(1)**	Developer	As a developer I want to set up an environment to enable	3

				testing to be possible	
18	**A**	**Done(7)**	User	As a user I want to transfer money between two of my accounts	8
34	**A**	**Done(4)**	Developer	As a developer I want to be able to transfer sound from app to api	8
35	**A**	**Done(8)**	Research	As a developer I want to research how users phrase specific requests	12
37	**A**	**Done(8)**	Research	As a developer I want to research and report on how certain dialogs are structured by the API	5
38	**A**	**Done(7)**	Research	As a developer I want to write a report on the process it took to record and send sound	3
39	**A**	**Done(7)**	Research	As a developer I want to add details about deployment features to the design chapter	2
40	**A**	**Done(7)**	Research	As a developer I want to make a short report about authentication possibilities	5
41	**A**	**Done(7)**	Research	As a developer I want to report on the difficulties that were encountered with Icelandic declension	3
43	**A**	**Done(8)**	Developer	As a developer I want to write a user instruction manual	1
44	**A**	**Done(8)**	Developer	As a developer I want to write a operations manual	3
45	**A**	**Done(8)**	Developer	As a developer I want to optimize actions and make sure they perform well	54
36	**B**	**Done(6)**	Developer	As a developer I want Talbankinn API to distinct between multiple users and keep some state	12
33	**B**	**Done(5)**	Developer	As a developer I want to set up a Landsbanki mock API	5
19	**B**	**Done(7)**	User	As a user I want to pay my bills	8
20	**B**	**Not started**	User	As a user I want to create a new account	8
21	**B**	**Not started**	User	As a user I want to create a new known recipient	8
22	**B**	**Not started**	User	As a user I want get overview of my loans	5
23	**B**	**Not started**	User	As a user I want to get details about a specific loan	5
24	**B**	**Done(6)**	Developer	As a developer I want to have logs of failed AI text interpretations	8

42	**B**	**Done(7)**	User	As a user I want to convert one currency to another	5
25	**C**	**Not started**	User	As a user I want to check my account statements to monitor my finance	5
26	**C**	**Not started**	User	As a user I want to get details on a specific transaction in my account statements	5
27	**C**	**Not started**	User	As a user I want to transfer money from my account to an unknown Icelandic account	13
7	**C**	**Not started**	Research	As a developer I want to research how to implement Cordova plugins to add our product to Landsbankinn's app	3

4 Risk Analysis

This chapter assesses the various risks that were predicted to come up during the development of Talbankinn. In total there were nine risks which were analyzed. For each risk the following factors were analysed: the *impact* the risk would have on the project, possible *mitigation* tactics to prevent a risk from happening, how to *solve* the problem that follows a materialized risk along with damage, probability and severity of each risk. Finally, each risk was assigned to one or more team member which would assess the situation of each risk

.

Severity Table

Damage				
	High	B	A	A
	Medium	C	B	A
	Low	C	C	B
		Low	Medium	High

Probability

#	Description	Impact	Mitigation	Solution	Damage, Probability, Severity			Assigned to
1	Team member is sick or taking time off because of surgery	Project might get delayed since not all members are present	Get enough sleep, eat healthy, exercise	Daily communications with the sick individual so he can work at home	med	low	C	Bjarki, Leifur, Smári, Snorri
1.1	Team member worked at home when possible and communicated through messenger							
2	Team members have big assignments in other courses	Project might get delayed since not all members are present	Plan ahead of time so that planned workdays are free	Try to split workdays between projects if possible	high	low	B	Bjarki, Leifur, Smári, Snorri
2.1	There came gaps when everyone was studying for the final exam and when members had some big assignments but after the final exams team members worked every day of the week instead of 3 times a week to make up for it							
3	Unable to make Product work with smart speaker	Product Owner gets disappointed, solution might work better with it	Try to research about this as much as possible	Discuss with product owner about next steps	high	low	B	Smári
3.1	This seemed to be the case, even though Google speech-to-text supported Icelandic, Google home did not. For more details please refer to the Smart speaker plugin research (chapter 7.4)							
4	Unable to connect to Google Cloud's Speech-To-Text API	One of the milestones. If not working, project itself won't work	Try to research about this as much as possible	Team members try to figure it out together	high	low	B	Snorri
5	Team is unable to finish a story in sprint	Team might get behind original plan	Stories for each sprint will be chosen more wisely	That story must be assigned to the next sprint	low	high	B	Bjarki, Leifur, Smári, Snorri
5.1	This was a rather common occurrence but it never caused any big problems and most stories were finished in the next sprint. More about that in chapter 7.5.							

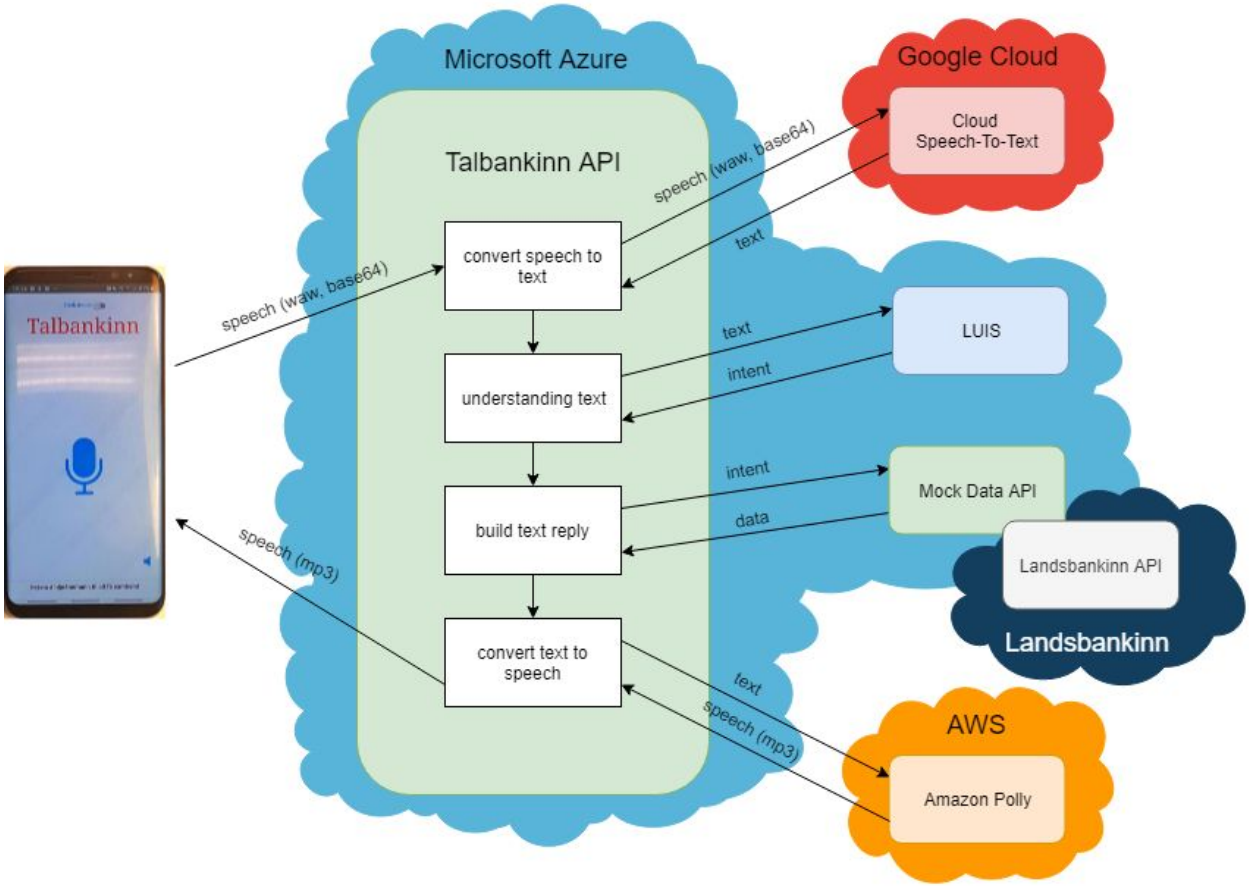
6	Speech-to-text API is unable to understand complex numbers (5566 kr)	It would be very bad since this is a project for a bank where numbers are important	Research this, making sure that Google can handle this	There might come up need to switch to RU Speech-to-text	med	low	C	Snorri, Bjarki
7	Turns out it is impossible to make our product work because of external reasons (force majere)	Team members fall into despair	Think about this every step of the way, if the product is going in the direction as expected	Only time can fix this problem. Time is the best healer	high	low	B	Bjarki, Leifur, Smári, Snorri
8	Unable to connect to AWS Text-to-speech API	One of the milestones. If not working, project itself won't work	Research this, make sure it is possible with current system design	Team members try to figure it out together	high	low	B	Leifur
9	AWS Text-to-speech can not pronounce a complicated sentence	Delayment due to having to write a new script	Try and keep the scripts simple and concise	Come up with a new script, possibly with assistance from Product owner	low	med	C	Leifur

By the end of sprint #2 it was clear that it was not possible to implement Talbankinn as smart speaker plugin. This maps to risk #3. It was assessed properly on a meeting with Landsbankinn. See further in smart speaker research chapter (ch. 7.4)

5 Design

Talbankinn contains three main components. First, the Talbankinn API, which is the main API. Second, the Mock Data API which contains fake data which the Talbanki API works with. Finally, there is a mobile app which is used to communicate with the Talbanki API. It is essentially an interface.

Below is a diagram which shows visually how these components are connected as well as the flow of data between them. A detailed description follows below the diagram.



These are the steps the user goes about using the system:

1. User records his voice in the app.
2. The resulting audio file is sent to Talbankinn API.

3. Talbankinn API forwards the audio data to Google Cloud's Speech-To-Text and receives it back as text.

4. The text is sent to Microsoft's LUIS, a natural language understanding processor which interprets the meaning of the request and returns the result to Talbankinn API in the form of a JSON object containing information about the users' intent.

5. Using the interpretation data, the Talbanki API performs the requested action. E.g. transferring money between accounts or fetching data about share rates.

6. The Talbanki API puts together text which includes feedback for the requesting user, with data gathered from either Landsbankinn's API or the Mock Data API.

7. The string is sent to Amazon Polly which is hosted on Amazon Web Service (AWS). Polly returns a corresponding sound file with the spoken words represented by the string.

8. The resulting sound file is sent back to the mobile app and played for the user.

5.1 Deployment Features

Microsoft Azure was used to host both Talbankinn API and Mock Data API because of the relatively easy setup and connection to Visual Studio Code. Azure was also used for continuous deployment through GitHub. To achieve this comfortably the main API and the Mock Data API as well as the mobile app were set up on three different repositories. This also makes it easier switching parts of the system out, like replacing the Mock Data API out for the actual Landsbanki data API.

5.2 Dialog Flow

The dialog flow of Talbankinn was designed according to Microsoft Azure's bot service design documentation. This chapter contains information about how Talbankinn meets Microsoft's principles of a good speech service, how the conversation for each financial assistance works and how the dialog could be improved in the future.

5.2.1 Current design

The first impressions of the application is very important. When the Talbankinn mobile app is opened, the user is greeted with the question "Hæ, hvernig get ég aðstoðað þig með fjármálin þín í dag?" which translates to "Hi, how can I assist you with your finances today?". Since it is generally not recommended to start with an open ended question such as "How can I help you?", it was decided to add "with your finances" to give a better indication what kind of

assistance Talbankinn offers. A help request was also implemented to give information about what kind of financial assistance Talbankinn offers.

How interrupts are handled is very important. It is therefore possible to cancel all multi step operations such as financial transactions. It is however, not possible to ask about things that do not relate to what assistance is currently being provided. So, if the users asks about something unrelated he is asked to cancel or continue with the current process.

To make sure Talbankinn does not seem clueless, every response that informs the user it does not understand him, was implemented to be very informative.

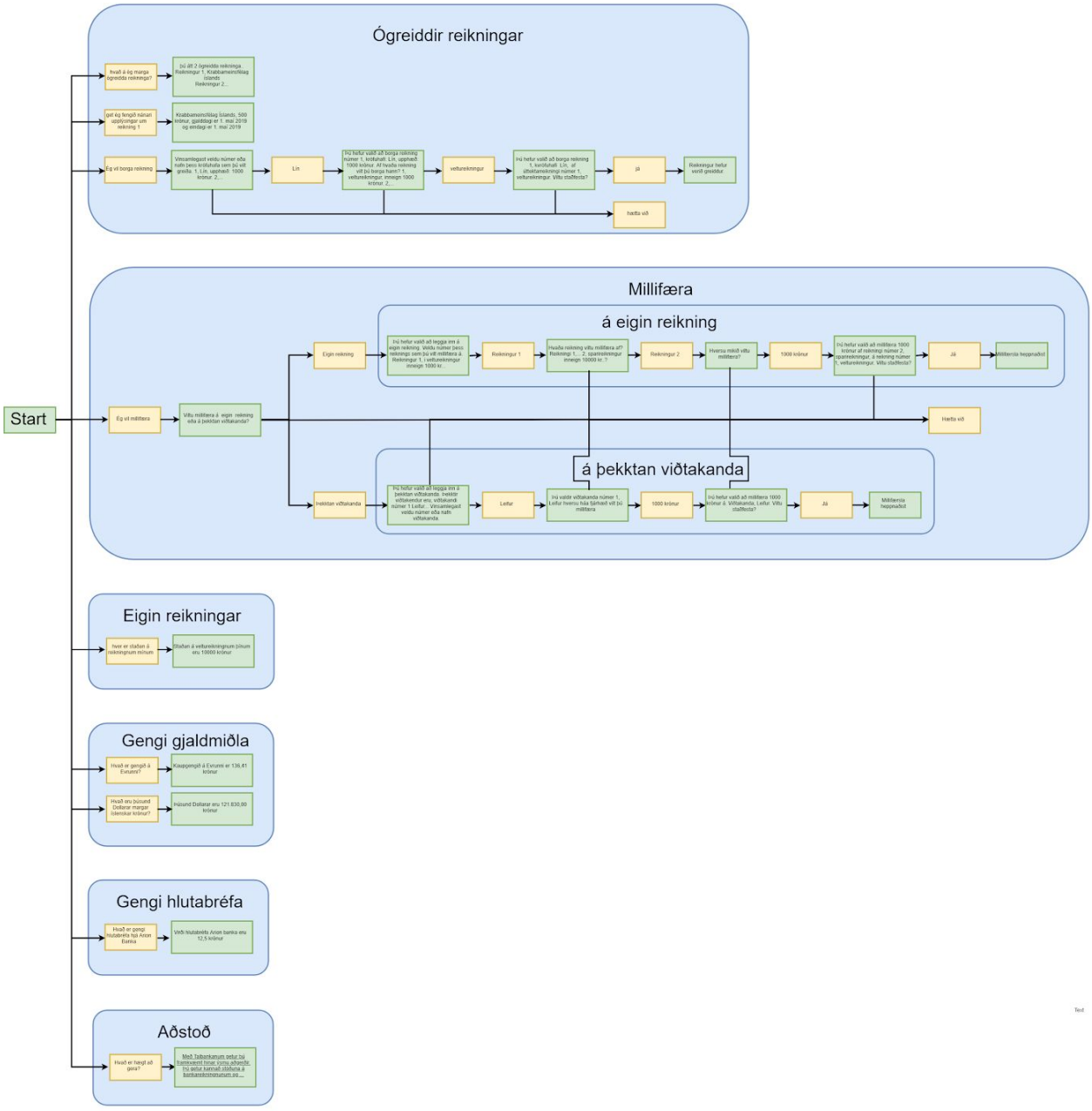
Talbankinn must immediately acknowledge requests, especially since it is offering assistance with finances. That is why every request with the user is met with a response, e.g. after the user confirms that all transaction information is correct and when he wants to transfer money, he gets a response informing him that the transaction was successful. Visual indicators such as spinners were also implemented for the mobile app to show the user that something is going on while he waits for the reply.

It is also important not to offer the same information too often and therefore Talbankinn does not inform the user what he is doing with every step he takes in a multi step operation but sufficient reassurance that it understands him correctly is provided. For example, Talbankinn does not remind the user that he is performing a transaction with every step of the process, nor does it make the user confirm that Talbankinn has understood him correctly with every step. Instead Talbankinn just asks for the next information it needs and then, at the very end, Talbankinn asks for confirmation whether it has gathered all the correct information.

Talbankinn should not keep information about what the user was doing for too long. Originally, it was intended to implement a feature that resets the user's state when he closes the mobile app. This is so that if the user closes the mobile app and opens it the next day, he is not asked to continue with what he was doing last time but is instead met with the initial greeting. This feature is something that should be included with future improvements [3].

5.2.2 Dialog Tree

On the next page is the current dialog tree for Talbankinn. It shows all of the actions it offers and all the steps required to reach a specific goal. The soft blue rectangles represent different parts of the system, the yellow boxes represent what the user says and the green boxes represent what the Talbankinn says.



5.2.3 Future Improvements

One of the big problems working with the Icelandic language were the various declensions. Both when determining the intents of the user and when creating replies for the user. With future development it will be very important to improve how declensions are handled so that users can

give more natural responses and understanding their responses better. Currently when the user is asked to pick something from a list, whether it is known recipients, accounts or bill he is required to give the number of the bill or the name in the correct declension when it is listed for him. The use of numbers give fewer declensions to work with and the response to the user is phrased so the user is forced to give the name in the correct declension.

Another improvement for future development will be handling interruptions more naturally. One important aspect that has to be tackled is allowing the user to check the status of his accounts while performing various transactions. Currently he is required to cancel transactions if he wants to check account status and starting the transactions again from the start.

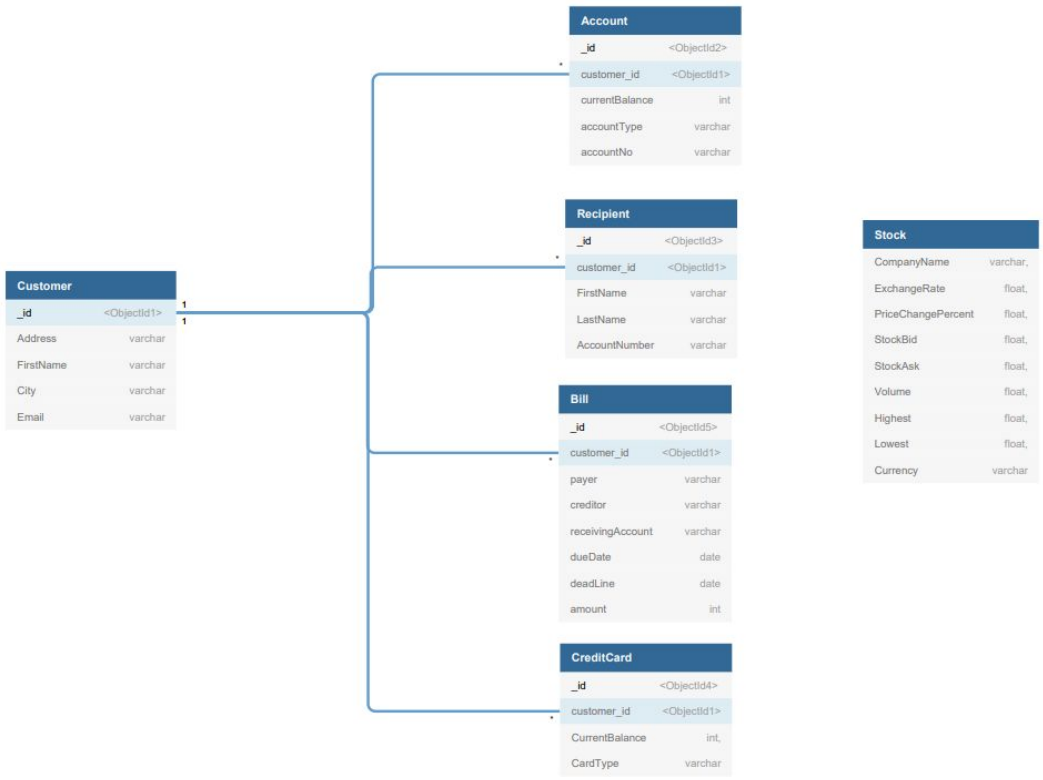
5.3 Mock Database

Landsbankinn API had to be substituted because, obviously, Landsbankinn did not provide access to actual data from real customers. To build answers with relevant information for the user, that relevant information had to be fetched from another data source.

5.3.1 Database chosen

MongoDB with Mongoose was chosen because it is very comfortable to use and works well with NodeJS. MongoDB was also well suitable since complex relations are not required for a minimum viable product.

5.3.2 Table schema



This is the table schema which was designed for Talbankinn. Each customer has his own identifier (id). In MongoDB and Mongoose id's are generated automatically and are a type of string called ObjectId. In the picture above there is a customer and he can have multiple accounts, known recipients, bills and credit cards. Stock just lives in its own collection with no relations to the customer table. It would, though, have made sense to connect it to the customer table because a customer can own shares in a company. Note that MongoDB and Mongoose use strings but not *varchar*. The figure above includes varchar rather than strings as it was made with software which aims at creating table schemas for SQL.

5.3.3 Endpoints

Access to the data is available through endpoints in the Mock Data API. There is not a complete CRUD operations for all of the endpoints. For a minimum viable product, the possibility to create a new customer, new bill, new stock or a new credit card was not implemented. Those are all actions that are not something customers can do by themselves but have to happen at the banks end. Operations that are supported but were not implemented in the final project are that the customer could request to create a new account and a new recipient. To see details about

all of the endpoints a special swagger site was created: https://talbankinnmock.azurewebsites.net/api-docs/

6 Testing

A lot more testing was intended than was actually done. Originally, everything was going to be test-driven and multiple user tests on the system were going to be conducted. With more time it would have been part of the project. Although not everything was test-driven and unit tested, some parts of the system were implemented in a test-driven manner and a user test with multiple different users was conducted. This chapter contains information on the testing that was done.

6.1 Unit testing

A little emphasis was put on unit testing due to the lack of stand alone functions that could be tested. However all stand-alone functions used for the purpose of string building were tested.

6.2 User testing: Wording

There are many different things that affect the way people speak. Such as age, gender, ethnicity occupation, region, and more [4]. It is important that it feels natural for all people to speak with Talbankinn. So, the goal of this test was to answer the research question “What different ways will users go about wording their requests to Talbankinn?”.

6.2.1 Methods

A user test for *Google Home*, done by the Google Usability Study Group, was used as an inspiration for this test. One of their research questions matched this research and so the methods used were adapted to the Google one. This section covers the overview of the participants, the environment that was tested in, a task list description and information about the session format [5].

The participants

The criteria for the participants was that they have experience doing regular banking. In order to get a good variety in participants, a choice was made to take four interviews for each age group with two interviews with each sex. The four age groups were, 16-24, 25-40, 40-60 and 60+ years old, resulting in a total of 16 interviews. Information about the participants is displayed in a table on the next page.

Participant	Age	Sex	Internet bank use per month	Uses banking app	Computer knowledge on scale of 1 to 10	Experience using speech interface
P1	23	Male	1	No	10	None
P2	38	Male	30	Yes	9	None
P3	27	Female	30	Yes	9	Some experience using Samsung Bigsby and Siri
P4	62	Female	4	No	5	None
P5	27	Male	4	Yes	8	Good experience with Amazon Alexa and Google Assistant
P6	46	Male	3	No (uses mobile site)	10	A lot of experience from researching and developing speech services and interfaces.
P7	17	Male	1	No	8	Some experience using Siri
P8	35	Female	6	Yes	8	None
P9	59	Male	5	Yes	5	None
P10	23	Female	4	Yes	7	None
P11	67	Female	1	No	4-5	None
P12	22	Female	7	No	8	A little experience using Siri
P13	66	Male	0	No	2	None
P14	54	Female	3-4 a week	Yes	5	None
P15	59	Female	30	Yes	6	None
P16	69	Male	20	No	6	Yes but a long time ago. Not regular.

The tasks set for the participants correlate to the user stories shown in the following table.

Task number	Story number	Description
1	**14**	As a user I want to know the balance of my accounts
2	**10**	As a user I want to know exchange rates
3	**15**	As a user I want to know the status of my credit card (funds available + balance)
4	**25**	As a user I want to check my account statements to monitor my finance
5	**9**	As a user I want to get overview of unpaid bills

6	**12**	As a user I want to get information about a specific unpaid bill
7	**22**	As a user I want get overview of my loans
8	**27**	As a user I want to transfer money from my account to an unknown Icelandic account
9	**21**	As a user I want to create a new known recipient
10	**8**	As a user I want to transfer money from my account to a known recipient
11	**26**	As a user I want to get details on a specific transaction in my account statements
12	**18**	As a user I want to transfer money between two of my accounts
13	**11**	As a user I want to know the share rates
14	**19**	As a user I want to pay my bills
15	**23**	As a user I want to get details about a specific loan

The tasks were provided in the form of a scenario both so that they seemed more interesting and relatable and also to try avoid influencing how the tester words his requests. The tasks were ordered from easiest to hardest where the difficulty was measured by the amount of information that had to be provided and how frequently the tasks were done in real life. The full list of tasks can be seen in Appendix 1. Each participant was given as much time as they needed to perform each task and while completing the tasks, his requests were recorded to capture his wording precisely.

The environment

It is important that the environment the tests are conducted in is the same area or as close as possible to the environment that users will actually use the service in. The goal with the service is that it can be used anywhere, so the tests were not conducted in any specific environment. Tests were conducted both at Reykjavik University and in participants' homes.

Session Format and Roles

The testing began by greeting the participants and making them feel familiar with the assignment. Before the participants started doing the tasks, they were asked some context questions about themselves. Before the participants started doing the tasks they were reminded that the goal of the research was to gather information about wording and not to measure their knowledge of the Icelandic language or banking phrases. Then the task session began with every participant reading through all of the scenarios and when they were ready, the recording of their requests began. Finally, participants were thanked for taking part in the test.

The wording test was created so that it could be conducted by a single group member at any given time. Whoever was moderating the test was responsible for recording what the user said during the task session and, afterwards, noting down how the participant worded the requests as well as the answers given to the context questions, debriefing questions and any important comments from the participant.

6.2.2 Results

Below are the results from the debriefing questions.

participant	**Myndir þú treysta því að stunda bankaviðskipti með talþjónustu eins og Talbankanum og ef ekki, af hverju?**	**Gætir þú ímyndað þér að nota þjónustu á borð við Talbankann og hvar myndir þú helst vilja nota hana og með hvaða tæki**	**Hvað myndir þú helst vilja nýta þjónustuna í að gera?**
p1	Já	Já, í símanum, nánast alstaðar	Millifæra á þekkta viðtakanda/viðtakanda sem hefur verið lagt inná nýlega. Borga alla ógreidda reikninga. Skoða nýjasta launaseðilinn.
p2	Já	Já, í símanum, í einrúmi eða þar sem ég gæti hvíslað	Millifærslur, borga reikninga og allt þetta helsta.
p3	Já	Snjallsíma, spjaldtölvu	Athuga reikningsstöðu og millifæra ef það er veitt nægilegt öryggi og staðfestingar til þess að passa að allt sé rétt
p4	Já	Já, með tölvunni	Borga reikninga, millifæra
p5	Já ef það eru nægilegar staðfestingar um að maður sé skilin rétt og ef það er góð radd þekking	Já, með símanum hvar sem er í næði.	Minni aðgerðir, til dæmis að millifæra lágar upphæðir og tékka stöðuna á reikningnum. Gott að nota samhliða appinu.
p6	Já, ef næg staðfesting er til staðar	Já, með símanum á skrifstofunni	Millifæra og borga reikninga, þar sem er mest stress og þar sem maður þarf að veita sem flestar upplýsingar
p7	Já	Já, með símanum	Borga reikninga og skoða innistöðu
p8	Já	Já, með símanum	Millifæra á þekkta viðtakanda og borga reikninga
p9	Já, með appinu til samanburðar til að byggja upp traust	Já, með símanum	Athuga stöðu reikninga, skoða og borga ógreidda reikninga, færa pening á sparnaðarreikninga, millifæra á þekkta viðtakendur
p10	Já	Já, með símanum	Millifæra á þekkta viðtakendur og skoða stöðu debetkorts

p11	Já, hlýtur að vera traustsins vert	Já, með símanum	Millifæra og kannski borga einhverja reikninga og eitthvað
p12	Já, en bara fyrir einfalda hluti eins og litlar millifærslur, en ekki hluti eins og viðskiptaráðgjöf eða lánaupplýsingar	Já ef hún er mjög klár en vil frekar tala við manneskju. Nota þjónustuna í tölvu, heima	Engin stór viðskipti eins húsakaup og bifreiðakaup. Millifæra og spyrja um vexti og þessháttar áður en farið er til ráðgjafa
p13	Já	Símanum	Staða á bankareikningnum og millifæra
p14	Já, ég myndi alveg treysta því	Símanum myndi ég nota hann	millifæra
p15	Já. Ef að allt öryggi væri vel útfært.	Já, í bílnum og heima. Ekki innan um fólk. Mundi nota síma.	Millifærslur, kanna stöðu reikninga og greiða þá.
p16	Já en mundi vilja hafa sjónrænt feedback um leið.	Já, heima, í bílnum, úti í búð og jafnvel í sumarbústað. Skiptir ekki máli þó það sé innan um annað fólk. Mundi nota borðtölvu og síma.	Greiða reikninga og skoða innistöðu á bankainnistæðum. Skoða gengi gjaldmiðla.

Here in the tables show the result on how each person would go about wording their requests.

P1

Task number	**Request wording**	**comments**
1	Opna reikning	
2	Breyta evru í íslenska krónu	
3	kreditkort	
4	Sjá heildar innistöðu á reikningnum	
5	Opna óborgaða reikninga	
6	Opna óborgaða reikninga, skoða nánar Krabbameinsfélag Íslands	
7	Skoða skuldir	
8	Opna reikning. Velja aðal reikning. Millifæra á maka litla bróðir	
9	Bæta Ásgeir við þekkta viðtakanda	
10	Opna þekkta viðtakanda, velja jóa, millifæra eitt þúsund og fimmhundruð krónur á Jóa	

11	Opna sparireikning. Sjá upplýsingar um nýjustu millifærslu	
12	Opna kortareikning. Millifæra 10000 krónur inná sparireikning eitt	
13	Opna hlutabréf	
14	Opna ógreidda reikninga. Borga alla ógreidda reikninga	
15	Skanna óverðtryggð húsnæðislán	

P2

Task number	**Request wording**	**comments**
1	Hver er reikningsstaðan mín?	
2	Hvað er ein evra margar íslenskar krónur?	
3	Hver er staðan á kreditkortinu?	
4	Hver er staðan á sparnaðarreikningnum?	
5	Á ég einhverja óborgaða reikninga?	
6	Nánari upplýsingar um reikning frá Krabbameinsfélagi Íslands	
7	Hef ég efni á x útgjöldum á árinu	
8	Millifæra á kennitölu x með x reikningsupplýsingum	
9	Bættu reikningsupplýsingum og kennitölu yfir Ásgeir við þekkta viðtakanda	
10	Millifæra á Jóa fimmtánhundruð krónur	
11	Fá nánari upplýsingar um millifærsluna	
12	Millifærðu tíu þúsund krónur af Kortareikning yfir á Sparireikning eitt.	
13	Hver er staðan á hlutabréfum í Endalausri Hamingju	
14	Borga ógreidda reikninga	
15	Fá upplýsingar um óverðtryggt húsnæðislán	

P3

Task number	**Request wording**	**comments**
1	Hversu mikinn pening á ég á debetkortinu mínu	

2	Hversu hátt er gengið á Evru. Hvað er ein evra margar krónur?	
3	Hvað á ég mikið til ráðstöfunar á kreditkortinu. Hvað á ég til ráðstöfunar á kreditkorti eitt/tvö/Gullkortinu.	
4	Hver er staðan á Sparnaðarreikningnum.	
5	Hvaða reikningar eru ógreiddir. Á ég einhverja ógreidda reikninga	
6	Hver er eindagi á reikningi frá krabbameinsfélaginu/Krabbameinsfélagi Íslands	
7	Hversu mikið skulda ég í húsnæðislán og námslán	
8	Greiða inná reikning “reikningsnúmer” kennitala. 1500 krónur.	Wanta there to be a multi step process with the response: “hversu mikil upphæð?”.
9	Bæta við þekktum viðtakanda. já/staðfesta.	Wants there to be a multi step process with the responces: “hvert er reikningsnúmerið? Hver er kennitalan? Er Ásgeir sá sem þú vilt bæta við”.
10	Greiða þekktum viðtakanda. Jóa. 1500 krónur.	Wants a multi step process with the responses: “hvaða þekkta viðtakanda viltu greiða? Hversu háa upphæð?” And ends with a verification.
11	Fá upplýsingar um millifærslu. Þúsund krónur 1. apríl	Wants a multi step process
12	Millifærsla. Milli eigin reikninga. Frá debet yfir á sparireikning. Tíu þúsund af debet reikning á sparireikning 1.	Wants a multi step process where all owned stocks are listed up.
13	Athuga hlutabréf. Endalaus hamingja. Athuga núvirði hjá Endalaus hamingja.	Wants a multi step process.
14	Hver er upphæðin í ógreiddum reikningum. Hvað skulda ég mikið í ógreiddum reikningum.	Wants information about how much money is left after the bill has been paid before paying.
15	Upplýsingar um húsnæðislán. Hvaða húsnæðislán hentar best. Húsnæðislán. Upplýsingar um óverðtryggt húsnæðislán.	

P4

Task number	wording	notes
1	Hvað á ég inná reikningunum mínum	
2	Hvert er gengið á evrunni	
3	Gefðu mér stöðuna á visakortinu	
4	Hver er inneignin á sparireikningnum mínum	
5	Á ég eftir að borga reikninga	
6	Upplýsingar um reikning frá Krabbameinsfélagi íslands	
7	Hver er skuldastaðan mín	
8	Greiða fimmþúsund á x x	
9	Setja ásgeir sem þekktan viðtakanda	
10	Borgaðu jóa fimtánhundruð krónur	
11	Vantar upplýsingar um þúsund króna greiðslu á sparireiknginginn	
12	Millifæraði tíuþúsund af launareikning yfir á sparireikning eitt	
13	Hvers virði eru hlutabréfin mín í endalaus hamingja	
14	Borgaðu ógreidda reikninga	
15	Vantar upplýsingar um óverðtryggt húsnæðislán	

P5

Task number	wording	notes
1	Athuga innistæðu	Expects a multi-step process
2	Hvað er ein evra í íslenskum krónum	
3	Athuga innistæðu á kreditkorti	
4	Athuga innistæðu	Expects a multi- step process
5	Á ég óborgaða reikninga	

6	Upplýsingar um reikning frá Krabbameinsfélagi Íslands	
7	Athuga skuldir	
8	Millifæra	Expects a multi- step process
9	Bæta við þekktum viðtakanda	Expects a multi- step process
10	Millifæra á jóa	Expects a multi- step process
11	Upplýsingar um millifærslu	Expects a multi- step process
12	Millifæra af kortareikning á sparireikning tíu þúsund krónur	
13	Athuga hlutabréfavirði endalaus hamingja	
14	Borga ógreidda reikninga	Expects a multi- step process
15	Upplýsingar um lán	

P6

Task number	**wording**	**notes**
1	Hvað á ég mikið á “vörðureikningnum”	Made up account
2	Hvað er ein evra í íslenskum krónum	
3	Hver er ráðstöfunin á kreditkortinu. Gullkortinu.	Expects a multi- step process.
4	Hvað á ég mikið á vörðureikningnum. En á vaxtareikningnum. Hvað á ég mikið í allt.	Made up accounts
5	Hvaða reikninga á ég eftir að borga. Hvað er það mikið sem ég á eftir að borga.	
6	Náðu í reikninginn frá Krabbameinsfélagi Íslands. Hvenær þarf að vera búið að borga hann.	
7	Hvað er ég að borga mikið á mánuði í lánum	
8	Millifærðu þúsund kall á kennitöluna “123” á reikning númer “123”	
9	Búðu til þekktan viðtakanda. Hann heitir Ásgeir. Kennitalan er 123, reikningurinn er 123.	Expects a multi-step process
10	Millifærðu eitt þúsund og fimmhundruð krónur á þekktan viðtakanda. Jóa	Expects a multi-step

		process
11	Finndu millifærslu á þúsund krónum inná sparireikninginn minn. Segðu mér hvað þú veist um þessa millifærslu.	
12	Millifærði tíu þúsund krónur af launareikningnum yfir á sparireikning eitt	
13	Hvers virði eru hlutabréfin mín. Finndu hlutabréf í Endalausri Hamingju. Hvers virði er Hlutabréfið	Expects a multi-step process
14	Hver er heildar upphæð ógreiddra reikninga. Er næg innistæða í vörðureikningnum. Borgaðu alla ógreidda reikninga	Expects a multi step process and the ability to check if there are enough funds on a specific account to pay the bills.
15	Segðu mér meira um þetta óverðtryggða húsnæðis lán. Hversu mikið er maður þá að borga á hverjum mánuði. Hversu mikil verður heildar upphæðin þegar uppi er staðið. Hversu hagstætt er þetta lán.	Gives good examples about information he would like to know about the lona.

P7

Task number	**wording**	**notes**
1	Hvað á ég mikinn pening	
2	Hvað er ein evra margar íslenskar krónur	
3	Hvað er staðan á kortinu mínu	
4	Hversu mikinn pening á ég á kortinu mínu	
5	Er ég búinn að fá útborgað þennan mánuð	
6	Má ég fá nánari upplýsingar um Krabbameinsfélgas reikninginn	
7	Hversu mikið skulda ég	
8	Borga Andrési sex þúsund krónur	
9	Bættu Ásgeiri í þekkta viðtakanda	
10	Borgaðu jóa eittþúsund og fimmhundruð krónur til baka	
11	Upplýsingar um millifærslu síðustu sex daga	
12	Millifærðu tíuþúsund krónur á kortareikningnum yfir á sparireikning 1	
13	Hvað eru virði hlutabréfs míns	

14	Borgaðu skuldir	
15	Mætti ég fá upplýsingar um lánið	

P8

Task number	Wording	Notes
1	Hver er staðan á tékkareikningi 12345	
2	Hvað er gengið á evrunni	
3	Hver er staðan á kreditkortinu	
4	Hver er staðan á reikningi númer	
5	Á ég einhverja óborgaða reikninga	
6	Hver er eindagi reiknings frá Krabbameinsfélagi Íslands	
7	Hver er skuldastaðan mín	
8	Legðu inná reikning Leifs, kennitala 12345678910, reikningur númer 103 13 12345	
9	Gerðu Ásgeir að þekktum viðtakanda	
10	Millifærðu eitt þúsund og fimmhundruð krónur á Jóa	
11	Gefðu mér upplýsingar um færslu þann þriðja mars, þúsund krónur.	
12	Millifærði tíu þúsund krónur af tékkareikninga yfir á sparireikning 1	
13	Hvað er gengið á Endalausri Hamingju	
14	Borgaðu ógreidda reikninga	
15	Hver eru núverandi kjör á Óverðtryggðu Húsnæðisláni	

P9

Task number	Wording	Notes
1	Hvernig er staðan á launareikningnum mínum	
2	Hvað þarf ég að borga fyrir eina evru. Hvað þarf ég að borga marga íslenskar krónur fyrir eina evru	
3	Hver er staðan á launareikningnum mínum	

4	Hver er staðan á launareikningnum	
5	Eru einhverjir ógreiddir launareikningar inná heimabankanum	
6	Hvenær er eindagi á reikningi frá krabbameinsfélaginu	
7	Hver er staðan á launareikningnum mínum	
8	Hvað þarf ég drengnum fyrir nammið	Expects to have received a transaction request from the person he is transferring money to
9	Getur þú sett Ásgeir sem þekktan viðtakanda á heimabankanum	
10	Hvað þarf ég að borga honum Jóa fyrir bíómiðann	Expects to have received a transaction request from the person he is transferring money to
11	Hvaðan kemur þessi þúsund kall á heimabankann	
12	Viltu millifæra tíu þúsund kall á sparireikning 1 af launa launareikningnum	
13	Hver er staðan á hlutabréfinu mínu hjá Endalausri Hamingju	
14	Viltu borga upphæð ógreiddra reikninga	
15	Hver er munurinn á að taka óverðtryggt lán og verðtryggt. Hver er afborgunin á miljón krónum af óverðtryggðum reikningi og verðtryggðum.	

P10

Task number	**Wording**	**Notes**
1	Hvað á ég mikinn pening	
2	Hvað er gengið á evrunni	
3	Hvað á ég mikinn pening	
4	Hvað á ég mikinn pening	
5	Á ég einhverja óborgaða reikninga	
6	Hvað er eindagið á reikningi frá Krabbameinsfélagi Íslands	
7	Hvað skulda ég mikið	
8	Legðu 123 krónur á reikning númer 123	

9	Bættu Ásgeiri við þekkta viðtakanda	
10	Leggðu inná jóa eitt þúsund og fimmhundruð krónur	
11	Nánari upplýsingar um nýjustu millifærslu á sparireikninginn minn	
12	Millifærðu tíu þúsund krónur af kortareikningi á sparireikning 1	
13	Hversu mikið eru hlutabréfin mín virði hjá Endalausri Hamingju	
14	Borgaðu alla ógreidda reikninga	
15	Gefðu mér nánari upplýsingar um óverðtryggt húsnæðislán	

P11

Task number	**Wording**	**Notes**
1	Hvað á ég mikla innistæðu/Hvað á ég mikið inn á reikningnum	
2	Hvað á ég að borga fyrir eina evru í íslenskum krónum	
3	Hvað ætli ég eigi mikið eftir inn á kreditkortinu mínu	
4	Hvað ætli ég eigi mikið inn á reikningunum mínum hjá bankanum	
5	Á ég eftir að borga einhverja reikninga?	
6	Ég ætla að vita hvort ég geti fengið upplýsingar um reikninginn minn hjá bankanum. [Innskot frá mér: Gætir verið spurð til baka hvaða reikning þú eigir við]. Hver er eindagi á reikningnum sem ég var að fá frá krabbameinsfélaginu	
7	Get ég fengið upplýsingar um hvað ég skulda mikið hjá bankanum	
8	Myndirðu vilja millifæra fyrir mig inn á reikning?	
9	Viltu bæta Ásgeir við sem þekktum viðtakanda hjá mér hjá bankanum	
10	Viltu borga honum Jóa til baka það sem ég skulda honum. [Innskot frá mér: Gætir verið spurð þá til baka hversu mikið] fimmtán hundruð krónur	
11	Geturðu gefið mér upplýsingar um hver var að leggja inn á mig / Geturðu gefið mér upplýsingar hver var að millifæra inn á reikninginn minn	
12	Viltu millifæra 10.000 krónur af launareikningnum mínum yfir á spari reikninginn minn	
13	Get ég fengið upplýsingar um hvers virði hlutabréfin mín eru í dag	
14	Borgaðu fyrir mig ógreidda reikninga	
15	Myndirðu vilja gefa mér upplýsingar um óverðtryggð húsnæðislán	

P12

Task number	Wording	Notes
1	Hversu mikinn pening á ég á reikningnum mínum	
2	Hvað er ein evra margar íslenskar krónur	
3	Hversu mikinn pening er ég með á kortinu	
4	Hversu mikinn pening á ég á heimabankanum mínum	
5	Hvað eru margir ógreiddir reikningar inná heimabankanum mínum	
6	Hvernig reikningur er þetta. Mig langar að fá meiri upplýsingar um þennan reikning	
7	Hversu mikið skulda ég af húsnæðaláninu mínu og námsláninu	
8	Get ég fengið kortaupplýsingar og kennitölur ‘blíp” þessarar manneskju.	
9	Ég vill bæta Ásgeri ‘blíp* syni sem viðtakanda	
10	Mér langar að bæta Jóni sem viðtakanda	
11	Hvaða upplýsingar hefur þú um þessa millifærslu	
12	Viltu millifæra tíuþúsund kalli yfir á sparireikning einn í ákveðinn tíma, kannski svona tvo mánuði	Wants there to be an option to automatically transfer money to savings account
13	Hversu mikið eru hlutabréfin mín virði og hverjir eru vextir þess	
14	Hversu marga ógreidda reikninga borga ég og hvað eru launin mín eftir það	
15	Gete ég fengið upplýsingar um hagstætt húsnæðislán og hvað eru vextir mínir eftir nokkur ár, sirka fimm	

P13

Task number	Wording	Notes
1	Hver er staðan á reikningnum	

2	Flaska af vatni sem kostar eina evru, hvað kostar hún í íslenskum krónum / Hvað er ein evra í íslenskum krónum	
3	Hver er staðan á kreditkortinu	
4	Hver er staðan á reikningnum	
5	Á ég óborgaða reikninga í heimabankanum	
6	Hvenær þarf ég að vera búinn að borga reikninginn frá krabbameinsfélaginu	
7	Á ég fyrir þessarri sólarlandaferð / Hvað skulda ég mikið	
8	Borgaðu <nafn> ef það er hægt / Borgaðu x kr á <kt-rknnr> / Viltu borga <kt-rknrk> x krónur / Viltu borga á <kt> <upphæð> -> reikningsnúmer [þegar bent á]	
9	Ásgeir er einn af þeim sem á að borga vikulega / Gerðu Ásgeir að þekktum viðtakanda	
10	Borgaðu Jóa eitt þúsund og fimm hundruð krónur	
11	Hvaða snerill lagði inn á mig þúsund krónur / Upplýsingar um það hver lagði inn á mig / Geturðu komist að því hver lagði inná mig þúsund krónur	
12	Viltu millifæra tíu þúsund krónur af kortareikningnum mínum á sparireikning eitt	
13	Hvers virði eru hlutabréfin mín í fyrirtækinu endalaus hamingja	
14	Borgaðu ógreidda reikninga	
15	Viltu gefa mér upplýsingar um óverðtryggð húsnæðislán / Geturðu gefið mér upplýsingar um óverðtryggð húsnæðislán	

P14

Task number	**Wording**	**Notes**
1	Hef ég efni á því að fara í sumarfrí til spánar sem kostar x krónur? /	

	Hef ég efni á að fara í spánarferð sem kostar flug og hótel hálfa milljón / Hef ég lagt nógu mikið fyrir til að komast í spánarferð sem kostar x krónur	
2	Hvað kostar vatnsflaskan í íslenskum krónum / Hvað kostar vatnsflaska sem kostar eina evru í íslenskum krónum / Hvað kostar ein evra í íslenskum krónum	
3	Hver er staðan á kortinu mínu	
4	Hef ég efni á að kaupa Mercedez Benz sem kostar fimm milljónir	
5	Er ég búinn að borga alla reikninga?	
6	Hver er eindaginn á reikning frá krabbameinsfélaginu	
7	Get ég safnað fyrir ferð til kanaríeyja / Hef ég efni á tilboði til kanaríeyja / Hef ég efni á því að fara til kanaríeyja ef tilboðsverðið er x krónur	
8	<kt> greiða þrjú þúsund krónur inn á <rknr> / <kt-hb-rknr> borga þrjú þúsund krónur	
9	Greiða <nafn> þrjú þúsund krónur (misskildi, hægt að spyrja í framhaldi etv hvort vilji setja Ásgeir sem þekktan viðtakanda)	
10	Borga/greiða Jóa (vin) eitt þúsund og fimm hundruð krónur	
11	Hver lagði inn þúsund krónur á sparireikning minn? /	
12	Ég vill greiða tíu þúsund krónur af kortareikningi mínum í sparireikning 1	
13	Hvað á ég mikinn hlut í endalausri hamingju og hvers virði er (sic) þau	
14	Greiða ógreidda reikninga / Ég vil greiða ógreidda reikninga	
15	Er hagkvæmara óverðtryggt eða verðtryggt núna / Er óverðtryggt húsnæðislán sniðugt ? / Ég vil fá upplýsingar um óverðtryggt húsnæðislán / Ég vil fá upplýsingar um verðtryggt húsnæðislán / Hver er munurinn á verðtryggðu og óverðtryggðu húsnæðisláni	

P15

Task number	**Wording**	**Notes**
1	Getur þú sagt mér hvað er inn á reikning 1280?	1280 is a made-up account number.
2	Gætirðu sagt mér hvað ein vatnsflaska kostar í íslenskum krónum?	Did not realize that it is necessary to say the amount the bottle costs.

3	Gætiru sagt mér hvað ég hef mikið til ráðstöfunar á kreditkortinu?	
4	Geturðu sagt mér stöðuna á reikningnum mínum, númer 1280?	1280 is a made-up account number.
5	Geturðu sagt mér hvaða reikningar eru ógreiddir?	
6	Gætirðu sagt mér hvenær eindagi gjalddaga er á reikningnum frá Krabbameinsfélagi Íslands?	Did not remember at first to mention which bill (Krabbameinsfélag Íslands)
7	Geturðu sagt mér hvað ég skulda í húsnæðislánum og námsláninu hjá mér?	Uses both plural and not for “lán”
8	Ég ætla að leggja inn á reikningsnúmer X, kennitala X, 2500 krónur.	X are made up values for reikningsnúmer and kennitala.
9	Ég ætla að bæta við Ásgeiri Jónssyni, kennitala X, reikningsnúmer X.	Assumed that Ásgeir was also Jónsson to use full name. Expected a multistep conversation with a confirmation step.
10	Ég ætla að leggja inná Jóa, 1500 krónur, reikningsnúmer X.	Mentioned reikningsnúmer even though Jói was a known recipient. Possible misunderstanding of task. Expected a multistep conversation with a confirmation step.
11	Viltu athuga fyrir mig nýjustu millifærsluna?	Was not sure how to specify which millifærsla.
12	Viltu millifæra yfir á sparireikning 1, 10000 krónur / Ætlarðu að millifæra fyrir mig á sparireikning 1, reikningur 185?	Forgot, the first time, to specify úttektarreikningur. We should assume some default úttektarreikningur if none is specified. In the second try, reikningur 185 is the úttektarreikningur. Forgot to mention the amount though.
13	Ætlaru að athuga fyrir mig stöðu mína á hlutabréfum hjá fyrirtækinu Endalaus Hamingja?	
14	Ætlaru að borga reikninga út af debetreikning 1980?	1980 is a made-up account number.
15	Ætlarðu að athuga stöðu mína í sambandi við húsnæðislán?	

P16

Task number	Wording	Notes
1	Hver er inneign mín? / Hvað er inneign mín há?	
2	Ég er með 200 Evrur, hvað er það mikið í íslensku? / Ég er með 200 Evrur, hvað er það mikið í íslenskum krónum?	Found it ridiculous to make effort for 1 Euro. Instead a 200 Euro camera was proposed.
3	Hver er staðan á kortinu?	
4	Hvað er mikill peningur inni hjá mér?	Assumes a default account if no account is specified.
5	Hvað eru ógreiddir reikningar? Hverjir eru þeir og hve háir?	
6	Hvenær er reikningur <Nafn/> í eindaga?	
7	Hver er staðan á bílalánunum? / Hver er staðan á íbúðarláninu?	
8	Millifærðu á reikning X, krónur <Upphæð/> og láttu fylgja með skýringu: <Skýring/>	Thought it was a must to have a skýring.
9	Vinsamlega bættu Ásgeiri við sem þekktum viðtakanda hjá mér.	Did not realize that more information was needed.
10	Greiddu Jóa bíómiða, krónur 1500 með þökk fyrir lánið.	“Með þökk fyrir lánið” is intended as skýring.
11	Hvaða koma þúsund krónurnar? Hver lagði þær inn á mig og hvers vegna?	“Hvers vegna” intended to get skýring.
12	Vinsamlegast settu inn á Sparireikning 1 10000 krónur mánaðarlega.	Wanted to make a monthly 10000 krónur transaction plan.
13	Hvers virði eru hlutabréfin í Endalaus hamingja?	
14	Vinsamlega borgaðu ógreidda reikninga. / Vinsamlega greiddu ógreidda reikninga.	If (s)he was to pay only certain bills, (s)he would enumerate those bills.
15	Hver eru vaxtakjör og tími á þessu láni?	

6.2.3 Discussion

The results show that Google Usability Study Group’s method for gathering information about different wordings was not perfect. Some participants gave requests for tasks that were very similar to how the scenario for the task was described. They were influenced by the scenario

description. Despite this, the results show many different ways of wording requests, some of which are very different from what was expected. The results also give good insight into how the users think. For example one interesting thing that the results show is that users generally do not expect to be using this type of method to transfer money to unknown recipients. When they were asked to transfer money to an individual that they did not have the name of they often gave him a name or referred to the person by the nameless description. The results also show that people that have good experience using voice interfaces expected to be performing tasks in multiple steps although no information about the steps was given. Another interesting thing that the results show is that participants expected there to be a default withdrawal account. This shows that having a default withdrawal account is important and that it is also important to lead them through the steps needed to perform the tasks that require a lot of different information.

The results from the debriefing questions show that there is a very positive outlook towards using a service like Talbankinn to manage finances. Neither age nor computer skills seem to affect this outlook. No one showed great mistrust to the system. People generally assumed the system to be safe as it was being developed by a bank.

6.2.4 Conclusion

The results gave good information regarding the research question “What different ways do users go about wording their requests to Talbankinn?”. The research also gave interesting insights into how users of the system will think and what they will expect when using a speech based banking system. They also reveal that people are positive towards the technology. Future testing should emphasize on finding even better ways of avoiding influencing how the user phrases his request.

7 Research

Before development could begin, research had to be conducted to determine the current capabilities of the enabling technologies for Talbankinn as well as which companies provide the best services for these technologies. The three main enabling technologies that had to be researched were speech-to-text, natural language comprehension, and text-to-speech. These three technologies were researched specifically with the Icelandic language in mind since Landsbankinn is an Icelandic institution and the service will be used mostly by Icelandic speaking users. The following subchapters contain the reports about each research.

7.1 Speech-to-text

The first step in our API is to transform voice recordings of the user to text. Both RU (Reykjavík University) and Google Cloud have a speech-to-text API that understands Icelandic. The question is: Which one of them is more accurate, especially in terms of numbers?

7.1.1 Methods

Both the RU's and Google Cloud's speech-to-text API had to be tested with the same texts. To test the difference, https://tal.ru.is/ was used to test the RU's API and https://www.google.com/intl/en/chrome/demos/speech.html for the Google Cloud's API. In each sentence, certain keywords were picked so that if the API got them right, even though the rest of the sentence was not completely correct, it got a pass (1) but otherwise a fail (0). Information about the sentences, used along with the keywords (marked in blue), can be seen in the Results section of this research. Sentences were chosen to try to mimic how the user could possibly ask for certain information. The accuracy of the service to understand numbers and names were of vital importance.

7.1.2 Results

The two APIs resulted in a tie at nearly every level. The first tests were basic sentences describing a request such as to transfer money between accounts and checking an account's balance. The second tests were plain numbers. The only difference between the two APIs was that the Google API converted spoken numbers to digits while the RU's API converted them to letters. The Google API did not, however, convert to numbers when "þúsund" was said, "eitt þúsund" had to be said for that. Reasons for this are unknown. The last test had basic Icelandic names where Google won because the RU's API could not identify non-Icelandic family names such as Zoega correctly.

Test 1	**Google Cloud's API**	**RU's API**
Hver er staðan á reikningnum?	Pass	Pass
Ég vil vita stöðuna á reikningnum	Pass	Pass
Ég velti fyrir mér stöðunni á reikningnum	Pass	Pass
Hver er staðan á kreditkortinu?	Pass	Pass
Ég vil vita stöðuna á kreditkortinu mínu	Pass	Pass
Ég velti fyrir mér stöðunni á kreditkortinu	Pass	Pass
Ég vit vita frekari upplýsingar um reikninginn frá símanum	Pass	Pass
Ég ætla að millifæra 1 milljón á spari reikninginn minn	Pass	Pass
Ég millifæri 3000 kr á vaxtareikninginn minn	Pass	Pass
Ég ætla að borga reikninginn frá Símanum	Pass	Pass
Ég borga kröfuna frá Símanum	Pass	Pass
Reikningurinn frá Símanum, ég ætla að borga hann	Pass	Pass

Ég ætla að millifæra 5500 kr á Ara Arason	Pass	Pass
Millifæra 10 000 kr á Agla Helgason	Fail	Fail
Hvaða reikninga á ég ógreidda?	Pass	Pass
Hverjir eru ógreiddu reikningarnir mínir?	Pass	Pass
Hvað á ég marga ógreidda reikninga?	Pass	Pass
Hvað skulda ég í ógreiddum reikningum?	Pass	Pass
Ég ætla að stofna nýjan reikning	Pass	Pass
Ég vil búa til nýjan reikning	Pass	Pass
Stofna nýjan reikning	Pass	Pass
Stofna nýjan vaxtareikning	Pass	Pass
Hvert er gengið á Bandaríkja dollar	Pass	Pass
Hvað kostar bandaríkjadollar?	Pass	Pass
Hvað eru 100 dollar margar íslenskar krónur?	Pass	Pass
Hvað eru 10 500 kr í bandaríska genginu	Pass	Pass
Hvernig standa lánin mín?	Fail, falsely interpreted “lánin” as “málin” each time	Pass
Hvað skulda ég í lánum?	Pass	Pass
Hvað á ég mörg lán útistandandi?	Pass	Pass
Hvaða lánagreiðslur eru fallnar í gjalddaga?	Pass	Pass
Frekari upplýsingar um lánið frá Símanum	Pass	Pass
Gefðu mér frekari upplýsingar um efsta lánið	Pass	Fail, could not recognize the “efsta” word correctly.
Total	30	30

Test 2: Numbers	**Google Cloud’s API**	**RU’s API**
Einn	Pass	Pass
tíu	Pass	Pass
hundrað	Pass	Pass

þúsund	Pass	Pass
Tíu þúsund	Pass	Pass
Hundrað þúsund	Pass	Pass
Hundrað og fimmtíu þúsund	Pass	Pass
Fimm þúsund og fimm hundruð	Pass	Pass
Ein milljón	Pass	Pass
Ein milljón og fimm hundruð þúsund	Pass	Pass
Total	10	10

Test 3 : Icelandic Names	**Google Cloud's API**	**RU's API**
Leifur Pálsson	Pass	Pass
Smári Freyr Guðmundsson	Pass	Pass
Snorri Arinbjarnar	Pass	Pass
Þuríður Djúpúðga	Pass	Pass
Hermóður Sigurðsson	Pass	Pass
Berglind Zoega	Pass	Fail, recognized Zoega as "sjóveika"
Sveinn Baldursson	Pass	Pass
Þórkatla Þórarinsdóttir	Pass	Pass
Total	8	7

7.1.3 Discussion

The two APIs seemed to be equal, with the exception that the Google Cloud's API converts spoken numbers to digits while the RU's API does not. They were equally good at recognizing numbers and sentences. The fact that the Google Cloud's API converts to digits is a huge plus but at the end of the day, what it really came down to was simplicity in development, learning how to use each and access to support if needed. Google has number of examples on how to use it for different platforms and/or frameworks, RU does not have as much documentation. The developers of the RU AI were, however, literally five steps away from the developers' working area at RU, so it was possible to get personal help.

7.1.4 Conclusion

After discussing this research with the Product Owner, the fact that Google converts to digits is such a big advantage over the RU that a decision was made to use Google's speech-to-text API.

7.2 Language comprehension

Language comprehension platforms offer a solution to categorize strings, that the speech-to-text API provides, to groups called intents and entities. The platforms use AI and machine learning technology to do this but they all differ from each other in how they do it. This study was intended to find out which one of them was best suited for Talbankinn.

7.2.1 Methods

To find the best language comprehension platform, material was read on each of them, features were written down and compared to others. Once a platform was deemed inferior to the others, it was removed from the study as a contender. Once every platform had been researched, tutorials were viewed for better understanding of the remaining platforms.

7.2.2 Results

After researching and finding all the features the platforms offered, the most suitable platforms were Watson assistant from IBM, and Microsoft's LUIS. The two services' features are listed below.

Watson assistant [6]

- Digressions (allows end users to change the topic and return to where they left off)
- Dialog Tracing ("When building a virtual assistant, you often want to test it out in your Try it panel. What happens when you're trying to troubleshoot a problem? We've added a feature that allows you to see exactly what dialog nodes were hit during a given utterance" - Taken from Watson's website)
- Dialog trees
- Secure and trusted (You maintain control and ownership of your data and IP)
- Intents and entities
- Utterance

Microsoft's Language Understanding Intelligence Service (LUIS) [7].

- Intents and entities
- Phrase lists (A phrase list includes a group of values (words or phrases) that belong to the same class and must be treated similarly (for example, names of cities or products). What LUIS learns about one of them is automatically applied to the others as well. This list is not a closed list entity (exact text matches) of matched words)
- Utterance

7.2.3 Discussion

In the end, there were two contenders for the project, Microsoft's LUIS, and Watson Assistant from IBM. After a discussion with the Product Owner, it was revealed that there was a flaw with Watson Assistant in the way it handled entities for Icelandic, something which LUIS had no problem with. LUIS was not perfect either, it handled entities well but had no built in feature to make dialog trees. That means that if Watson was picked, regex statements had to be made for every entity that might have to be needed. If LUIS was picked the dialog flow work would have to be implemented from scratch.

7.2.4 Conclusion

Although Watson Assistant offered more features, many of which were very convenient for the project, the work that would go into constructing the regex statements for all entities would be much greater than constructing the dialog trees from ground up. Therefore the platform chosen was Microsoft's LUIS.

7.3 Text-to-speech

This study was made to determine what would be the best text-to-speech service available for Talbankinn. The text-to-speech software is used so that the Talbankinn can speak text information for the user so that a human conversation can be imitated. The services that were tested were Amazon Web Services' (AWS), Google's and Lumenvox's text-to-speech services. The main thing that had to be tested was overall understandability.

7.3.1 Methods

All three text-to-speech services had to be tested using the same text. To test the difference, the software at *translate.google.com* was used to test Google's text-to-speech software, *https://www.lumenvox.com/products/tts/#* was used to test Lumenvox's text-to-speech software and *https://eu-west-2.console.aws.amazon.com/polly/home/SynthesizeSpeech* to test AWS's

text-to-speech service. Multiple different texts were chosen to test these services. The first one was created to mimic how the software would respond to user requests. The second one included names of multiple Icelandic places which are hard to pronounce. The third and last one included multiple Icelandic tongue twisters.

7.3.2 Results

The table below shows results from the testing of each speech-to-text software. The tests were played for the members of the group and then they graded the understandability of each software on a scale of 0 to 5. A score of 0 means completely incomprehensible and a score of 5 means perfectly clear. The average score of each is shown in the table.

		Average understandability score on a scale of 0 to 5		
Description	Text	AWS	Google	Lumenvox
Exchange rates	Bandaríkjadollari. kaup 119,39 krónur. Sala 120,12 kr. Evra kaup 136,79 krónur. Sala 137,61 kr. Pund. kaup 156,33 kónur. Sala 157,29 kr.	5	1	5
Information about unpaid bill:	Skýring Valgreiðslukrafa. Útgefandi reiknings SOS-barnaþorpin. Gjalddagi 01.12.2018. Eindagi 15.12.2018. Upphæð 1300 krónur.	5	1	5
Balance of my accounts:	Verðtryggður 36. 10000 krónur. Fjárhæðaþrep 514852 krónur. Gullkjarareikningur 32697 krónur.	5	2	5
Names of Icelandic places which are hard to pronounce:	Barkarstaðagilslægðir, Bleikkollubólstindur. Eyjafjallajökull. Fimmálnaspottalækur. Golbílduhjallaröðlar. Glúmsgilskeggjahryggur. Hvalvörðugilslækur. Kirkjubæjarblettur.	4.5	0.5	4.5
Icelandic tongue twisters	Hnoðri úr norðri verður að veðri þó síðar verði. Stebbi stóð á ströndu og var að troða strý, en strý var ekki troðið nema Stebbi træði strý. Eintreður Stebbi strý, tvítreður Stebbi strý, þrítreður Stebbi strý… Frank Zappa í svampfrakka. Rómverskur riddari réðst inn í Rómarborg, rændi þar og ruplaði radísum og rófum. Hvað eru mörg R í því? Barbara Ara bar Ara araba bara rabbabara.	5	0.5	5

7.3.3 Discussion

The Google text-to-speech software was very unclear. AWS and Lumenvox offer text-to-speech software that can both speak with a male and female voice. AWS's voices, which are called Karl and Dóra sound exactly the same as Lumenvox's Birta and Isak. Their understandability is,

therefore, exactly the same. What it really comes down to is simplicity in development, learning how to use each service and the cost of using the service. AWS's does not require its users to set up a deal with the company on how they are allowed to use the service like Lumenvox requires.

7.3.4 Conclusion

AWS's service is more accessible than Lumenvox's and because of that it was chosen. To choose between AWS's voices, Dóra and Karl, a research on whether people prefer male or female voices was conducted. A female voice won by a marginal amount on a poll conducted by Android Authority. Out of the 1648 individuals that participated, 58% prefer a female voice but only 9% preferred a male voice. The rest either preferred a gender neutral voice or had no preference [8]. Because of this, Dóra was chosen as the default voice for Talbankinn.

7.4 Smart speaker plugin research

Note: Sometimes during this report, there are references to 'the initial design'. Initial design refers to the design which was eventually implemented. For more information, see chapter 5: Design.

The smart speaker researches were the last major researches that were performed before any major implementation of the product was started. Some important facts were discovered which had been good to know before the other researches were made. It turned out that the frameworks offered by the smart speaker producers, Google and Amazon, take care of most of the heavy lifting that was originally intended for implementation. They take care of factors like speech-to-text, natural language comprehension and text-to-speech. While researching the technologies available for these factors, it was assumed that Talbankinn API, which would take care of these factors by itself, could simply be "plugged" into Google's and Amazon's services. It turned out not to be that simple which is not good as it is one of the main requirements of Talbankinn, to be accessible via smart speakers. Ways to make Talbankinn work with Amazon's *Alexa* and Google's *Assistant* were researched and these are the results.

7.4.1 Google Assistant

Google's smart speakers, such as *Google Home*, are powered by a powerful AI that goes by the name of *Google Assistant*. It listens to the user's request, connects to its cloud service and processes the request before returning an answer to the user.

Google Assistant's custom plugins are called *Google Actions*. The goal of this research was to find out if it was feasible to implement Talbankinn as a custom Action for Google Assistant [9].

There are a few key terms that Google uses in the development of such Actions. First, there is an *intent* which is a task that the user wants to do. In Talbankinn's case it might be something

like getting the balance of an account or to pay a bill. Such a task or goal is identified by Actions as a unique identifier which can be triggered by certain user utterances. Second, there is the *Action* itself. It is the interaction built for Google Assistant, which supports certain intents and has implemented a fulfillment for those intents. The third term are those *fulfillments*. Fulfillment is a service or a logic that receives the identified intents and carries out the request (action) identified by the intent [10]. For us, this is an API which receives information about the intent, performs the requested action and finally sends back an answer indicating success or failure. These terms will be used regularly in the text below.

Over 500 million devices, including smart speakers, are powered by Google Assistant [10]. That means that there is a possibility that, by making Talbankinn available as a custom Action, that it might be available through countless types of devices other than smart speakers, such as phones, cars, and smart watches.

7.4.1.1 Methods

Google Assistant Actions documentations were viewed where the steps to make a custom Action are explained thoroughly. The team did not have access to a Google Home speaker during the earlier period of the research but there was a lot of information available about its capabilities on the internet. The factors that were against the feasibility of making this happen were analyzed and, of course, the factors that implied that it is actually possible to do.

The result from this research was that in order to make all of this work for Talbankinn, certain Google restrictions would have to be bypassed, more on that in the Results chapter. To find out how to bypass these restrictions a massive research began, covering articles and discussion forums where eventually good descriptions were found along with code examples.

7.4.1.2 Results

After a lot of digging around, reading official tutorials and discussion forums the result was that there was *probably* a way to make Talbankinn work as a custom Action for Google Assistant. “Probably” because unusual paths towards the goal would have to be taken and because of uncertainty of the capability of custom Actions to use Icelandic speech-to-text.

Part I: What Google Assistant offers

The main control interface is the *Actions console* offered by Actions. From there is a way to configure, test and publish a custom Action. One simply creates an Actions project and is good to go [11]. There are very helpful instructions and support. Everything is very accessible.

Google Assistant is able to connect to custom Actions by invocation request from the user. The user can say something like *“Okay Google, let’s talk to Talbankinn”*, Google Assistant asks Actions to invoke the custom Action, then the fulfillment service for the Action, the custom fulfillment service responds with a default welcome message which is sent back to Actions and

from there to the user's device [12]. From that moment, the user is not conversing with Google Assistant anymore but the custom action which has specific responses to corresponding intents, such as checking for account balance.

Google Actions takes complete care of speech-to-text processing. For that it uses Google's own speech-to-text, machine learning enabled service, and Cloud Speech-to-Text. It recognises 120 languages and one of them happens to be Icelandic [13], as is also described in the speech-to-text research in chapter 7.1. As it turned out, Google Actions does not allow developers to use Icelandic speech-to-text. See further information about that later in this report. When a user has his voice recorded by, for instance, *Google Home*, the recording of the voice is sent to Google Actions which uses this service to obtain a string of text that corresponds to the spoken words in the voice recording.

The next step in the system is to process the string of words by a NLU (Natural Language Understanding) service, also referred to as language comprehension engine. This is the service that finds the intents of the request, so that Talbankinn API fulfillment service knows how to act. Google Actions prefers, by default, to use a NLU service called *Dialogflow* which is powered by Google's machine learning algorithm [14]. There is, however, another option where Google simply puts this responsibility into the developer's hands. Google Actions simply sends the data load, in the form of a JSON object which, among other things, contains the string of text, to the custom fulfillment service which takes care of parsing it [15].

The fulfillment service implemented would be a *Node.js* API using what Google Actions calls *conversation webhook format*. A fulfillment service is essentially just a fancy term for an API that reacts to Action requests and sends back an answer. Assuming that Google Actions were used, the API would essentially be Talbankinn. The API must be able to maintain constant conversation with Google Actions via back-and-forth HTTPS POST requests containing large JSON objects that must be parsed by both sides and both sides must act in accordance to their contents. It is the API's responsibility to react to these requests and send back corresponding answers, that is why NodeJS would be a good choice. There is a client library called *Actions on Google* available for it, which simplifies this JSON HTTPS conversation progress a lot. It wraps this conversation into functions and handles all of this for the developer behind the scenes [15]. The fulfillment service fulfills its purpose by forwarding the string of spoken words to a NLU service, receives back intents, performs some logic according to the intents before finally constructing an answer JSON object which is sent back to Google Actions.

Google Actions also takes complete care of text-to-speech. When it has received the JSON object containing the answer from the API, it uses its *Cloud Text-To-Speech* service to transform the words into audio. The audio is then sent to the user's device, like Google Home, and played for the user to hear. This text-to-speech service has a major flaw though. It is not at all suitable for Talbankinn as it does not support Icelandic language [16]. This was also the result of the text-to-speech research in chapter 7.3.

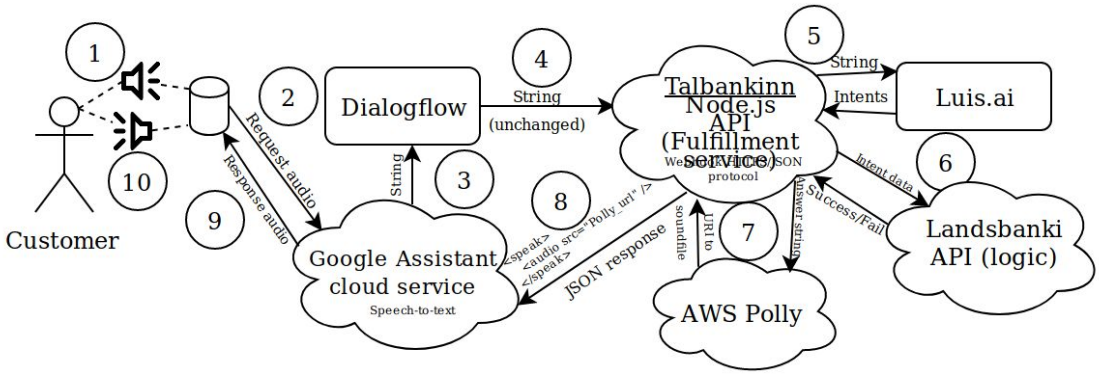
The things that Google Actions offers are all good and handy for Talbankinn except for the last factor, text-to-speech. There, however, was a way around this obstruction as will be seen in next part of this research result.

Part II: How the team hoped to make it work

Because another NLU was researched, Microsoft's *LUIS* (see chapter 7.2), it was chosen to use that service since it was good with Icelandic. At first it was believed that *Dialogflow* was mandatory to use and impossible to use something else due to Google's restrictions. The intention was to "cheat" on Google Actions by creating a single default intent on Dialogflow which would always contain the whole string of spoken text. Then, that string would be forwarded to the API which would then in turn forward it to LUIS which would then return the actual intents. However, as the topic was explored and setting up this Google Actions project had begun it was discovered that it is in fact possible to skip Dialogflow by legal means. The fulfillment services that can be connected to custom Actions can either be in *Dialogflow webhook format* where Dialogflow is always in between Google Actions and the API *or* they can be in *conversation webhook format*, as is described in part I [15]. Using that format, Google Actions does not speak to Dialogflow and sends a JSON object directly to the API containing the string of spoken words.

Obviously, Talbankinn must be able to answer Icelandic users in Icelandic language. That is why a way to bypass Google's text-to-speech engine had to be found. In the text-to-speech research, chapter 7.3, the result was that the best suitable text-to-speech service was *AWS Polly*. A way to use AWS Polly instead of Google's Cloud Text-to-Speech was found. Here is how. The fulfillment service, upon fulfilling a request, must construct an answer to be sent back to Google Actions. This is usually done using *Speech Synthesis Markup Language (SSML)*, which contains tags that wrap around strings of speech and controls how the words will be spoken by the receiving device. This can make the responses more life-like [17]. There is, however, an SSML tag which is of a particular interest, the *<audio />* tag. It contains an attribute called *src* which accepts a *URI*. This tag supports insertion of recorded audio files, so when Google Actions receives an answer containing this tag, it plays the sound file found at the URI in the receiving device [17]. The idea was to have the custom fulfillment service build an answer string, send it to AWS Polly, receive the audio file, find some way to host it and, finally, to put a URI to its location inside the src attribute of the audio tag in SSML. That way, Google Actions should ensure that the receiving device will play back a recording, of AWS Polly speaking in Icelandic.

The design



This shows how Talbankinn implementation as Google Assistant Action was visualized. It assumes that Dialogflow is “cheated” on, as is described above. See steps 4 and 5. Below is a description of the steps taken.

1. User records an audio request with Google Home.
2. Audio recording is sent to Google Assistant cloud service where it is changed to text.
3. The resulting text is sent to Dialogflow, which is a NLU service.
4. Dialogflow is configured such that it sends the whole text as a single intent to the custom Webhook fulfillment API.
5. Talbankinn API forwards the unaltered text to LUIS which returns the intents.
6. For security reasons, the bank logic would be on a separate secure API. The intents are forwarded there and the requested action is performed. A notification of success or failure is returned.
7. A text regarding a success or failure is sent to AWS Polly which turns the text to a audio file. It returns an URI to the audio file.
8. The fulfillment server returns a JSON object containing SSML which contains instructions to play the audio file when the response has reached Google Home.
9. The response is forwarded from the Google Assistant cloud service to Google Home.
10. Google home plays the audio file containing the response.

Part III: How it turned out not to work

A disappointing discovery came to light in the latter half of this research. At that stage, the team had gotten its hands on a Google Home Mini speaker and were finally able to give its features a try.

As it turned out, speech-to-text proved to be what eliminated all hope of making this implementation a reality. Even though Google has the capability to transcribe Icelandic speech, it will not allow developers to utilize it when developing a custom Action for Google Assistant. This infeasibility was also confirmed by a Google Actions support employee which was e-mailed for help. This is probably due to Google Assistant's general inability to understand and answer requests in Icelandic. Due to those reasons, it may be speculated that Google will not allow developers to transcribe Icelandic speech.

7.4.1.3 Discussion

By implementing *Talbankinn* as a custom Action for Google Assistant the initial API design would not work and different approach needed to be found. It might have been a good thing, though, as it possibly lessens the scope of the product. For instance, there would have been less programming as Google Assistant's framework, Actions, takes care of most of the things originally intended to implement from scratch, like device-to-cloud communication for instance.

In the earlier half of the research, before the problem was realized with speech-to-text, two challenges had to be solved using Google Actions. The first one was handling multiple customers using the service simultaneously. The second challenge was to come up with some kind of authorization that would enable the service to speak to Landbankinn's real API, not the mock one used for testing and development. There were some ideas on how to solve those challenges and which were applied in the final design. For the multiple user handling, the JSON package received from Google Actions at the custom fulfillment service contains data about which user and which device is communicating with the custom Action. That way there is a way to be able to distinguish between multiple incoming requests and keep the books about at which stage of dialog each user is. Regarding authorization, Google Actions supports *OAuth 2* which is the industry standard and happens to be used by Landsbankinn [18]. This is a material for a separate research, however. Authentication was not an A-requirement for Talbankinn but Landsbankinn requires at least some analysis or research on this. See chapter 7.6: Authentication methods. The first priority was to make a minimum viable product which would not take care of these challenges.

7.4.1.4 Conclusion

It is currently not possible to implement Talbankinn as a custom Action for Google Assistant. Although Google allows for great amount of freedom when creating their actions there is one step that cannot be edited by users, their speech-to-text algorithm. For reasons unknown, their algorithm that supports Icelandic is not used for Actions and therefore, the API had to be constructed in some different manner, without Google Home. If, in the future, Google decides to support Icelandic, the needed connections will be minimal since the final design of Talbankinn API is written in NodeJS which has libraries to make the communications between the systems very straightforward.

7.4.2 Amazon Alexa

Amazon Alexa is Amazon's cloud-based AI voice service equivalent to Google's Assistant. It powers tens of millions of devices, including Amazon's smart speakers, *Amazon Echo*. It also powers other devices of the Amazon's Echo family as well as third-party devices that have it built in [19]. Amazon Alexa's custom plugins are called *Skills*. As with the research performed for Google Assistant, the goal of this research was to find out if it is possible to implement Talbankinn as a custom Skill for Alexa. For the development of such Skills, Amazon offers an

environment called *Alexa Skills Kit (ASK)*. As with Google Actions, ASK is a framework which the product would need to be adapted to as it has set most of the important things up for the developers. Things like speech-to-text, natural language understanding and text-to-speech.

7.4.2.1 Methods

The documentation for ASK was read to get to know how the workflow is designed. At this stage, it was still unclear if it was possible to adapt the initial design to this environment and workflow. The research was still on how to connect a Skill to the initial API design. The online documentation were thoroughly scanned for any signs of this being possible and noted down the results. The lack of freedom available using Google Assistant was unfortunate and it was hoped that ASK would give more flexibility in using the initial API design. Thus, any indications of that were thoroughly pursuited. The research was not limited to the documentation but also various discussion forums and blogs.

7.4.2.2 Results

After some research the realization struck that implementing Talbankinn as a custom Skill for Amazon Alexa is not possible. Initially, the focus was on learning how ASK works in general, looking at the big picture to see how the workflow and its design could be adapted to the initial API design. One of the first things learned is that using a custom HTTPS server running an API to speak to ASK is a possibility. It is not unlike fulfillment services work in Google Actions. Using such a server, however, requires some deal of configuration to enable SSL and a signed digital certificate [20]. As with, Google Actions, the communication between Amazon's platform and Talbankinn API would be through repeated HTTP POST requests containing JSON objects that contain information such as device ID, user ID and, of course, the detected intents interpreted from the user's utterance. There was an alternative choice, which was to use *AWS Lambda*, which is a Amazon-governed serverless platform that already has the needed configurations as well as a signed digital certificate.That would have required the product to be running on a server owned by an outside party which was unacceptable.

The natural language understanding engine is provided and built-in to *Alexa Developer Console*, which is a part of the ASK. It offers basic features, such as defining intents and corresponding utterance examples as well as defining entities. It looked okay but research on this did not get far, so unlike research on how to bypass Google NLU engine in favour of preferred LUIS were not researched for ASK.

One of the best things about the potential of developing Talbankinn as an Alexa Skill is that, unlike Google Actions, it can use *AWS Polly*'s voices for text-to-speech when returning answers to users, without any dirty tricks as was intended to use in Google Actions [21]. Polly allows the use of Icelandic voices to speak strings of text for the users. It is perhaps not that surprising that it is possible, without any trouble, to use one Amazon product with another.

After somewhat extensive research of the big picture of ASK, the first step needed to be examined for this to be able to work in the implementation of Talbankinn. That is the speech-to-text service offered by Alexa. It turned out that Alexa's speech-to-text service does not support Icelandic and ways to swap it for some other service which transcribes Icelandic speech to strings of text were not found.

7.4.2.3 Discussion

The product owners were quite interested in Amazon Echo as a platform to connect to Talbankinn. They even bought the speaker for testing purposes but that was before the conclusion about the infeasibility of this implementation. They took it back and provided a Google Home Mini speaker instead, which eventually, also, was turned out to be of no good use either.

7.4.2.4 Conclusion

Amazon Alexa offers a very helpful and user friendly interface and a very nice and descriptive tutorials and documentation. It seems to have everything the bare minimum of Talbankinn requires except for the most crucial part which is Icelandic speech-to-text. It would be nice, in the future, if Icelandic became supported, to implement Talbankinn as a Skill for Alexa. But due to Alexa's lack of that support, it is currently not a possibility and will not be considered further.

7.4.3 Final conclusion

Due to technical restrictions and lack of freedom of implementation, it is not possible to implement Talbankinn as a plugin for a smart speaker. Two of the biggest smart speakers were examined on the market, Google Home and Amazon Echo, with their respective AIs, Google Assistant and Amazon Alexa. Neither supports Icelandic speech-to-text.

With Google Assistant, it was assumed during the research that it could turn Icelandic speech to string of text. That was for the simple reason that Google offers software called Google Cloud Speech-to-text which fully supports Icelandic, as was also concluded in the speech-to-text research. It happens to be that Google Actions does not grant developers access to use this at all. Amazon Alexa does not offer Icelandic speech-to-text either. That was a little bit more obvious as there are not many Icelandic speech-to-text engines out there and Amazon's competitor, Google, owns one of them while they do not. It is therefore, due to current technological restrictions set by Google and Amazon, it is currently infeasible to implement Talbankinn as a plugin for their smart speakers.

7.5 Recording sound with React Native

This chapter covers the technological challenges regarding sound recording in React Native mobile apps.

The goal seemed simple and so was the story which weighed two story points (which is about seven hours of work). The goal included recording audio, sending it from React Native app over to the NodeJS API (Talbankinn API), which would then send it to Google Cloud speech-to-text to get the audio transformed to text format.

7.5.1 Expo CLI

All members of the group had experience with React Native, or so they thought. The first attempt, Expo audio, worked in the mobile app. It was possible to record the audio and play it back, the quality of the playback was very good. In order to make the recorder compatible with Google’s speech-to-text there were two options when it came to encoding: Adaptive Multi-Rate Wideband and Adaptive Multi-Rate Narrow-band [22], [23]. This required some research on what audio extensions could be used with the encoder and what the output format should be, sample rate of the sound also had to be compatible with what Google expected.

After a few days, Narrow-band was successfully up and running. Google was able to detect the sound and sometimes, although rarely, gave back the correct string. It was odd how low Google’s success hit rate was and therefore, it was decided to save the file when it came to the API. The audio quality was terrible. By enabling playback in the app it was clear that the audio quality was the same. This meant that Narrow-band was ill-suited for recording audio so the

only option was Wideband. Three days went into trying to make Wideband work but Google's API would not accept the format.

After some time of installing packages that referenced files that did not exist within the project and did not work without them, it was clear that there is a big difference between React Native CLI and Expo CLI. Expo CLI is great when learning to program apps but is very limited and is not a good choice for real mobile apps. Instead of having a program within the phone, Expo uses a wrapper which it sends to the phone which makes it appear to be a real mobile app.

7.5.2 React Native CLI

Although the story had gone from the weight of originally estimated four hours into more than a week, it was important to know the differences between the two and that Expo was not a good idea when creating a real app. After one day of work a real app was up and running. After that, the options of audio recorders were not limited to Expo and it was possible to record sound of much higher quality [24].

7.5.3 Result

With React Native, a wav-recorder was used because Google supports wav-format and it ensures the audio quality that was needed [22], [23]. The *rn-fetch-blob* library was used to send the audio file over the stream because it is designed for file transfers and is a lot quicker than normal *fetch*. The audio gets converted to a *base64* string for Google speech-to-text before the transfer is emitted. The audio that is returned from the API is an MP3 file and in order to play the file it had to be created on the device first.

The solution worked about 80% of the time but sometimes a duplicated audio file would be sent over to the API. It turned out that the two files had to be either deleted in each communication cycle or given a new name in each iteration. It was decided to delete the file in each iteration in order to keep the memory consumption down. Deleting the file turned out to be problematic in an asynchronous app because, even though the action was called after a file had been created and was nested within a *promise* that first checked if the file existed, the execution of the app resulted in a failure because a file was being created that already existed. This problem was fixed when two points in the app that were bound to be synchronous were discovered. The first one happens when the user starts recording because he has to hold for more than a second to be accepted as a request. The second one is when waiting for a response from the API via a fetch request. When the user pushes the record button, the former audio recording made is deleted. When the app makes the fetch request, the former file sent by the API is deleted.

7.6 Authentication methods

This research covers different recognition types, protocols and authentication services that can be used for future development of Talbankinn. Although implementing security was not one of the requirements set by the product owner, Landsbankinn, the requirement for researching possible security methods was.

7.6.1 Voice Authentication

Voice Authentication is one of many different biometric authentication methods. Biometric authentications use unique biological characteristics to verify the identity of an individual. Voice recognition does this by analyzing the voice of the individual, just like the name implies. Biometric authentication methods have a high acceptance rate with users because they are less intrusive than conventional methods such as password authentication and they are also harder to spoof.

There are two main approaches of voice recognition, text independent method and text dependent method. The first approach authenticates the user by using any given phrase and the second verifies the user by making him speak a predetermined phrase.

The main advantage of using voice authentication for Talbankinn is that all the devices that are required to use the application are the same as the ones that are required for the authentication i.e. a microphone. The main disadvantage of voice recognition is that its reliability is low compared to other methods [25]. The problems is not that it requires a high quality recording of the users voice. That problem is shared with the application as a whole, but rather the fact that if something impacts the tone of the users voice such as over excitement or a physical ailment such as a cold can cause the voice recognition method to not recognize the users voice. This along with the concern of attackers using recordings of the users voice to gain access to the system means that voice recognition can not be a stand alone method of authentication for Talbankinn [26].

All the service providers that are used to create Talbankinn support voice recognition. Google’s service is called Voice Match, Amazon’s service is called Amazon Transcribe and Microsoft's service is called Speaker Recognition. However none of the service providers currently provide voice recognition for Icelandic [27], [28], [29].

7.6.2 Fingerprint recognition

Fingerprint recognition is another form of biometric recognition. Like the name implies, it verifies the identity of a user by their fingerprints. There are no big disadvantages of using fingerprint scanning other than perhaps the requirement for the user to have a device that supports fingerprint scanning. Most of the top mobile manufacturers include fingerprint scanners in their

phones and since Landsbankinn's App already supports this feature the code could be reused [30], [31].

None of the currently used service providers provide services regarding fingerprint scanning but there exists a npm package that can be used to enable fingerprint scanning on react native apps called react-native-biometrics [32].

7.6.3 Face recognition

Face recognition is another form of biometric recognition that is also more reliable than voice authentication. It relies on high quality facial shots to identify an individual. The authentication method is widely available for users since most, if not all, smartphones have front facing cameras for the purpose of taking high quality self-portraits. This method can also be used for laptop and desktop computers since most laptops contain front facing cameras for video calls and it is possible to get high quality cameras which can be connected to desktop computers for the same purpose.

The main disadvantage of Face recognition as the changing appearance of the user due to aging, facial hair growth, glasses and more [33].

All the service providers that are used to create Talbankinn support facial recognition. Google's service is called Vision AI, Amazon's is called Amazon Transcribe, and Microsoft's is called Computer Vision [28], [34], [35].

7.6.4 OAuth 2

OAuth 2 is the industry standard protocol for authentication. The framework works well for Talbankinn since it provides specific authorization flows which cover the system in its entirety i.e. web application such as the Talbankinn API and mobile devices such as the mobile app currently connected to the Talbankinn API. It can also cover future development since it also offers authentication flows for desktop applications, if talbankinn will be connected to Landsbankinn's website, and living room devices, for when smart speakers start supporting Icelandic [36].

A lot of information exists about how to setup OAuth, Google search gives almost 2 million result for "OAuth tutorial" and Google has tutorials on how to use OAuth 2 for their API's and Smart Speakers [11], [37], [38].

7.6.5 Rafræn Skilríki

Rafræn Skilríki is an authentication service provided by the Icelandic company Auðkenni. With rafræn skilríki, which means electric id, authentication information can be stored inside the users sim-card. This means that for a user to authenticate himself to the service he has to provide his

mobile information and then set up a pin number. After that the only thing that the user has to do to authenticate himself is use the pin number [39].

The service is widely used by Icelandic financial companies e.g. Landsbankinn where they are used to gain access their online bank. Rafræn skilríki is a good authentication solution for Talbankinn since Landsbankinn already has a service set up for using it [40].

7.6.6 Zero-Touch

Zero-Touch is a authentication method by Futurae which can be used to authenticate users without having them input any passwords or pin numbers. The authenticating works by checking whether their phone is located nearby the smart speaker and having the user speak a phrase. According to Futurae the service is meant to be used for authenticating users into services such as internet banks so it should be very safe. This could be a very nice authentication alternative for when smart speakers start supporting the Icelandic language [41].

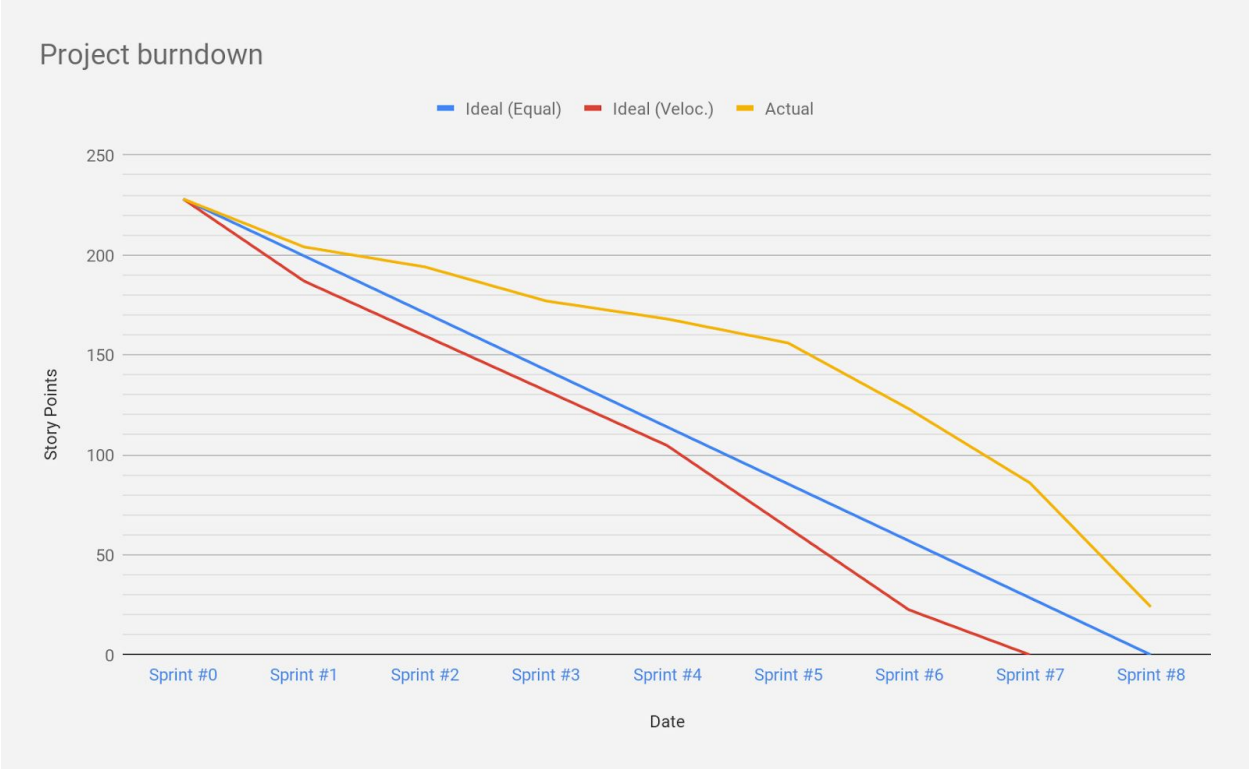
8 Progress overview

The project burndown chart is displayed on next page. It reflects the state of work by the end of the project. May 17th 2019. The total amount of story points for Talbankinn is 228. The legend for the chart is described below:

Ideal(Equal): Equal distribution velocity. The calculations assume an even distribution of story points to 8 sprints. That is 228 / 8 = 28.5 story points per sprint if scrapped stories and priority C stories are excluded. This, of course results in a straight line where all story points have been finished by the end of sprint #8.

Ideal(Veloc): Calculated capacity/velocity estimate. The calculations assume the calculated capacity, with regards to velocity (see ch. 2.2.1) for each sprint. Sprint #1 was two weeks long and had a capacity of 41.1 story points. Sprints #2 - #4 had capacity of of 27.4 story points and were two weeks long, each. Sprints #5-7 had capacity of 41.1 story points and were one week long, each. Finally, sprint #8 was 12 days long with a capacity of 61.7 story points. This results in an estimation that all story points would be finished by the end of sprint #7. This, of course, was built upon estimations and was not achieved.

Actual: The teams actual status at a given time.



As can be seen, the ideal velocity was followed during the first sprint but after sprint #1. As the time went by, the burning process grew quite a way apart from the ideal process. This suggests that the team's work capacity and velocity was overestimated in the beginning of the project.

When a little bit more than half of the time had passed, though, it was managed to finish 89% of the project. All that was left was a handful of B-stories and some C-stories.

Bear in mind that during some sprints, completed stories and their story points got scrapped. Those story points are not taken into account in this burndown as that would make it look like more story points had been completed than was the case. C-stories are not taken into account either.

8.1 Sprint 0

January 27th - February 2nd

Sprint #0 was used to do preparation work for the upcoming sprints. Deciding when to work on the project, when and where meetings should be held, creating templates for various things, such as Trello boards and research papers as well as creating user stories. Neither a burndown chart nor an hour log was used for this sprint since creating them was a part of the sprint. By the

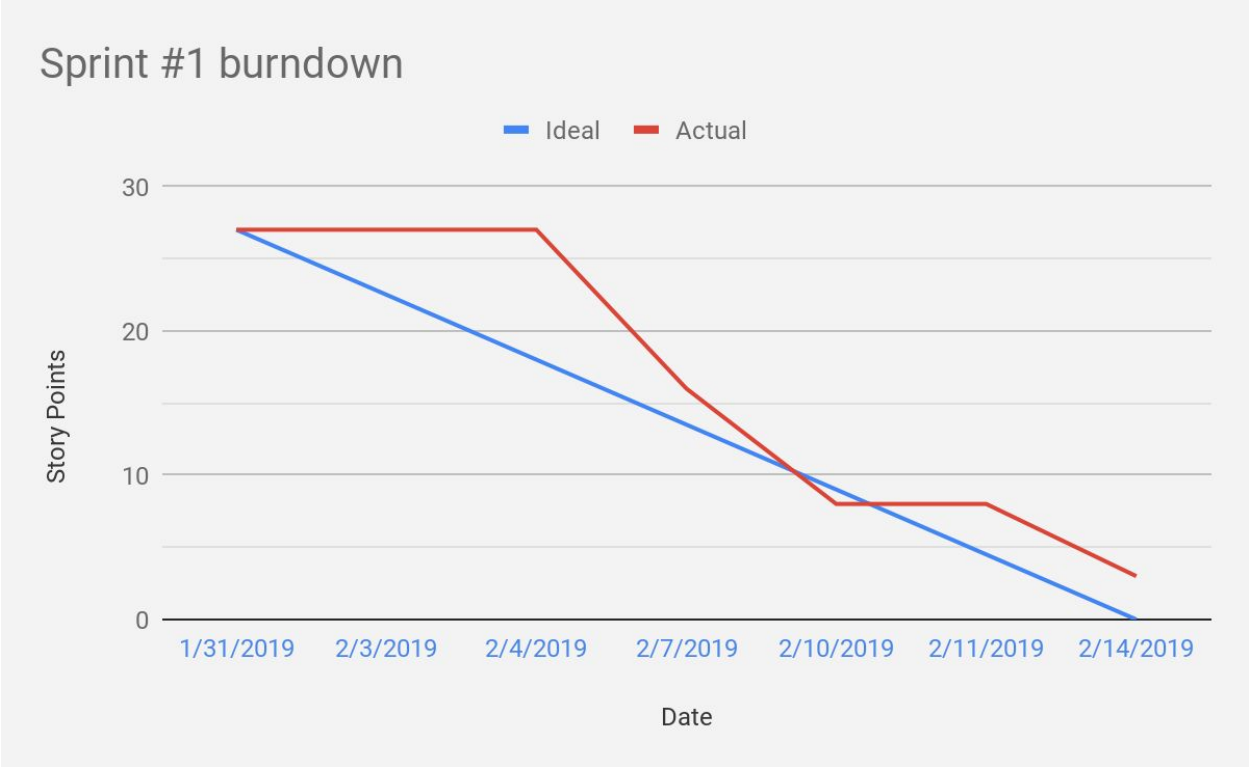
end of this sprint the groundwork for the rest of the project had been laid out and starting the implementation on the product on its way.

8.2 Sprint 1

February 3rd - February 14th

The theme of this sprint was research. A significant part of the research went into writing the reports in chapter 7 and creating a simple software for testing the research subjects. Some part of it also went into preparation for the first status meeting which intersected with this sprint (Feb. 12th). Regardless of the sprint capacity being 41.1 story points, it was decided to take 27 points for this sprint as one of members, Bjarki, had to be away due to personal reasons.

8.2.1 Sprint burndown



No story points were burned on the first three days since the research was being worked on. Then on the fourth day three out of the four researches were finished and that is the reason for the large dip.

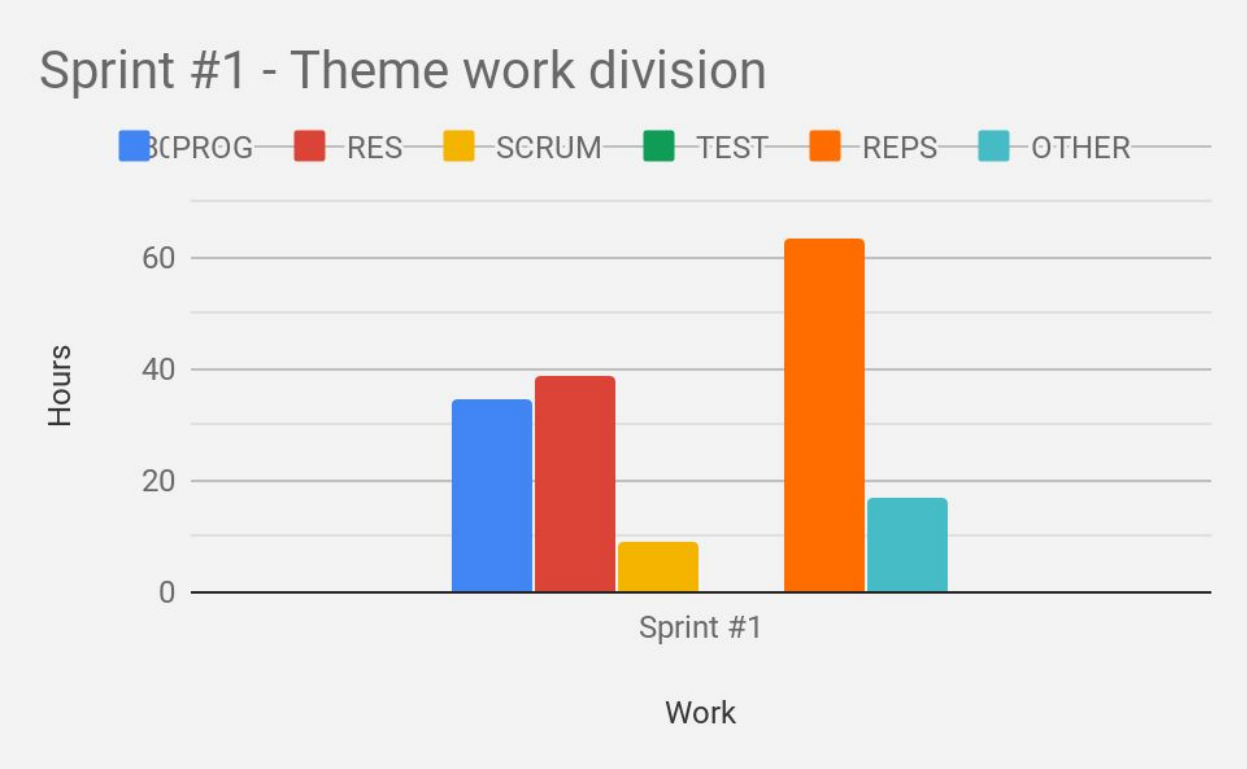
8.2.2 Sprint backlog

Partially finished stories are marked with a fraction in the *Finished* column. It is to be read this way: *story points finished / total story points of story*.
If not a single task was finished, the finished column simply contains *No*. Fully completed stories are marked with *Yes*.
In the *Pts.* column, *5(3)*, for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

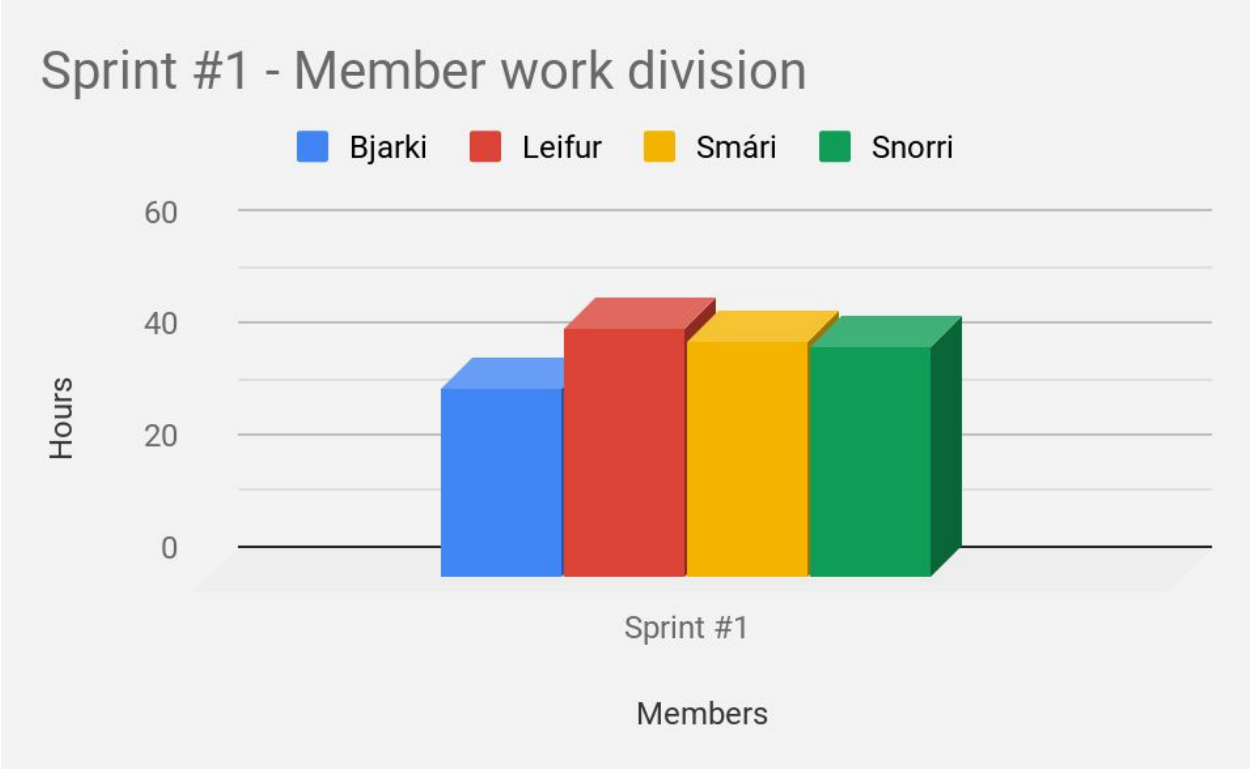
#	Priority	Type	Description	Pts.	Started	Finished
1	**A**	Research	As a developer, I want find out whether RU AI or Google AI is more suitable for the project.	3	Here	Yes
3	**A**	Research	As a developer I want to research the best suitable Language Understanding platform (Luis ui, Botengine etc.)	3	Here	Yes
4	**A**	Research	As a developer I want to research the best suitable Text-to-Speech tool: AWS Polly - Karl vs Dóra, Blindrafélagið's Jóhanna	5	Here	Yes
6	**A**	Research	As a developer I want to research how to create a skill on Amazon or Google	5	Here	No
16	**A**	Developer	As a developer I want to be able to deploy changes directly to a cloud service	8	Here	Yes
17	**A**	Developer	As a developer I want to set up an environment to enable testing to be possible	3	Here	Yes

As can be seen, most of the stories consist of research and preparation. Most of them were finished and reports written for the researches.

8.2.3 Sprint work division



The reasons these reports consumed so much of the time is that they had to be written from ground up, including setting up their structures, deciding what should be in them and much more. This includes this final report.



This sprint's work was spread quite evenly between members. The ideal error rate is a difference within 10 hours. Bjarki was away due to personal reasons for a part of this sprint, this is reflected in this bar graph. No harm was done, as the majority of what was intended was finished.

8.2.4 Sprint retrospective

This sprint went all right. The researches went well and a lot of valuable information was gathered. The status meeting went very well, the members were praised for doing a good presentation. Almost all stories were finished, the only work remaining was writing a report for the plugin research. Things that could have gone better were human communication between group members, one group member was missing due to personal reasons for more time than expected and, perhaps, too much time was spent on Scrum master activities. Things that group members planned on doing to prevent these issues from happening again were taking more breaks, which will be set at fixed times where everyone will participate. These breaks should reduce stress, improving work spirit.

8.3 Sprint 2

February 17th - February 28th

The team had already begun the work on this sprint when it was decided that too much time was being spent on this project and decided to decrease the work capacity. The amount of time spent on this project was preventing some of team members from having time to work on other courses that they were attending. Before the change, work was conducted three times a week, Sundays, Mondays and Thursdays with a capacity of 144 hours per sprint or 41 story points. It was decided to reduce that down to only Sundays and Mondays, using Thursdays as backup days which anyone could use to work if there was time. The new capacity was 96 hours or 27.4 story points. It was calculated to still be enough to finish the project on time. However, a decision was made not to change the already established sprint plan which contained 42 story points.

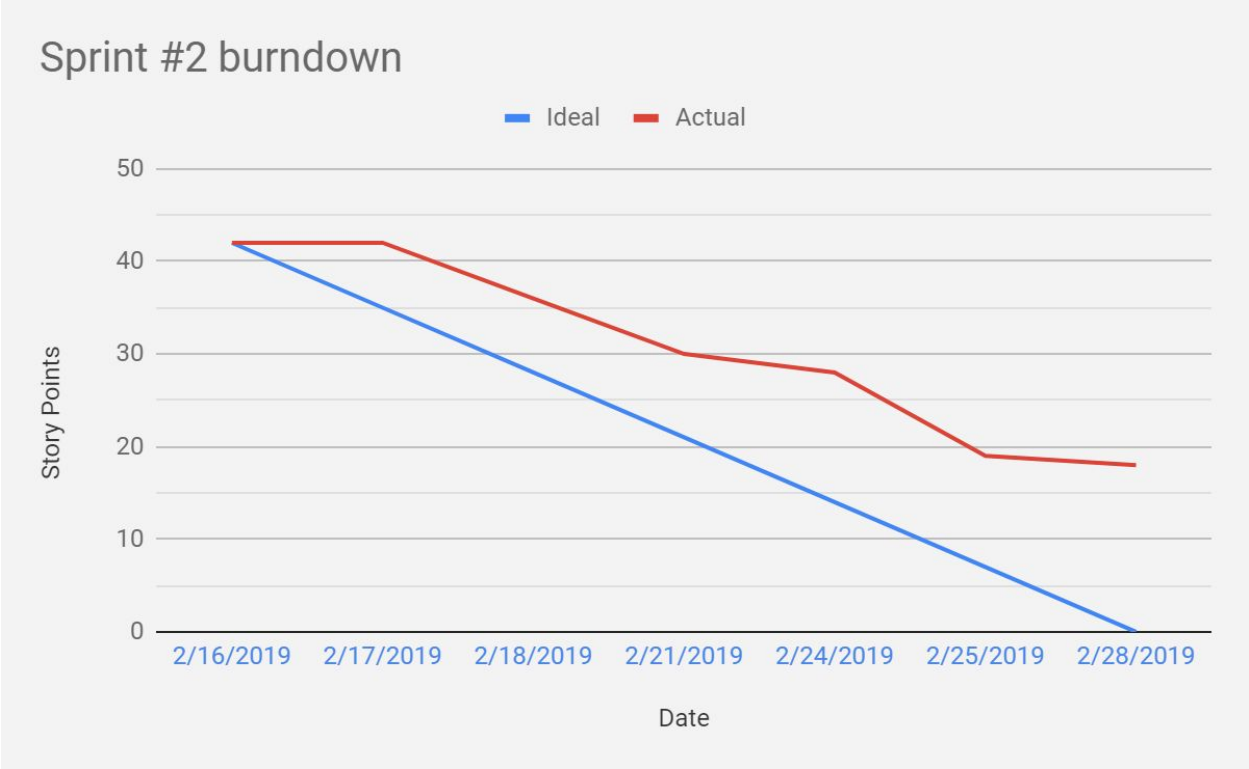
The theme of this sprint was Google Assistant custom Action implementation along with continued research on the feasibility of that method to implement Talbankinn. It was discovered that most stages of the process (Speech-to-text, natural language comprehension, text-to-speech) were possible using Google Actions in the desired way. That is, such that Icelandic could be spoken to the custom fulfillment API. It was known how LUIS could be used as the NLU of choice and how to get Polly to speak Icelandic back to the user.

It was figured out, however, towards the end of the sprint, that it was not possible to speak Icelandic to custom Actions. It was quite a disappointment. It was not until the latter half of the sprint that the team got their hands on a Google Home Mini speaker and then could try to speak to their fulfillment service. It turned out to be completely unable to transcribe Icelandic speech to text.

The mistake made was to assume that the Google Home speaker, along with Google Assistant, could perform speech-to-text for Icelandic. But this assumption was still very logical. Google's speech-to-text engine supported Icelandic. Thus, it was not illogical in assuming that the speaker was using that engine and could be configured to let the custom action use Icelandic. The assumption happened to be incorrect.

It is not a total waste of time, though, as a lot of the code from this sprint will be used in the next sprint. In sprint #3 it was returned to the original idea of an API which has one endpoint which takes in sound data, processes it, performs logic and returns some sound feedback.

8.3.1 Sprint burndown



These changes are reflected in this sprint burndown chart. The intended 42 story points could not be finished as there were fewer working hours than originally intended. The good news, though, is that 24 story points were finished which is quite near the newly calculated sprint capacity of 27 story points.
By the end of February 18th, 6 story points were burnt by finding information on the requirements of an API that speaks to Google Actions.

By the end of February 21st other 6 story points were burnt by managing to deploy and send a string to the custom Google Action fulfillment service and receiving back an answer.

By the end of February 24th, a decision was made on what kind of mock database would be used as well as setting up Nodejs API folder structure. That was 2 more story points burnt.

Quite a lot of work was finished on February 25th. The mock database was set up, the writing of the Google Action development research was completed and AWS polly was implemented to a fully working state. That was in total 9 story points burnt. On the last day of the sprint, February 28th, data was inserted into the mock database. That is 1 story point burnt.

A total of 24 story points were burned. Looking at the original capacity of 41 story points per sprint, it is barely half of the original capacity. This is however satisfying enough if the new

capacity that was calculated after the sprint is taken into account, which is 27 story points per sprint.

8.3.2 Sprint backlog

Partially finished stories are marked with a fraction in the *Finished* column. It is to be read this way: *story points finished / total story points of story*.
If not a single task was finished, the finished column simply contains *No*. Fully completed stories are marked with *Yes*.
In the *Pts.* column, *5(3)*, for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

#	Priority	Type	Description	Pts.	Started	Finished
6	**A**	Research	As a developer I want to research how to create a skill on Amazon and Google	5	Sprint #1	No
28	**A**	Developer	As a developer I want to set up Talbankinn Webhook fulfillment service	8	Here	Yes
29	**A**	Developer	As a developer I want to deploy Talbankinn Webhook fulfillment service	3	Here	Yes
30	**A**	Developer	As a developer I want to create mock database + data	5	Here	Yes
31	**A**	Developer	As a developer I want to bypass Dialogflow	3	Here	Yes
32	**A**	Developer	As a developer I want to bypass Google Text-to-speech service	5	Here	No
13	**A**	User	As a user I want to connect to Talbankinn through Google Home	13	Here	No

The majority of the stories were finished quite early. The remaining stories were not finished for the following reasons.

Story #6 is a research story that started in sprint #1. It wound up to be much larger than expected. When the sprint came towards the end, it turned out to be infeasible to implement Talbankinn as a smart speaker plugin. This must be taken into account in the research report and has to be done in next sprint.

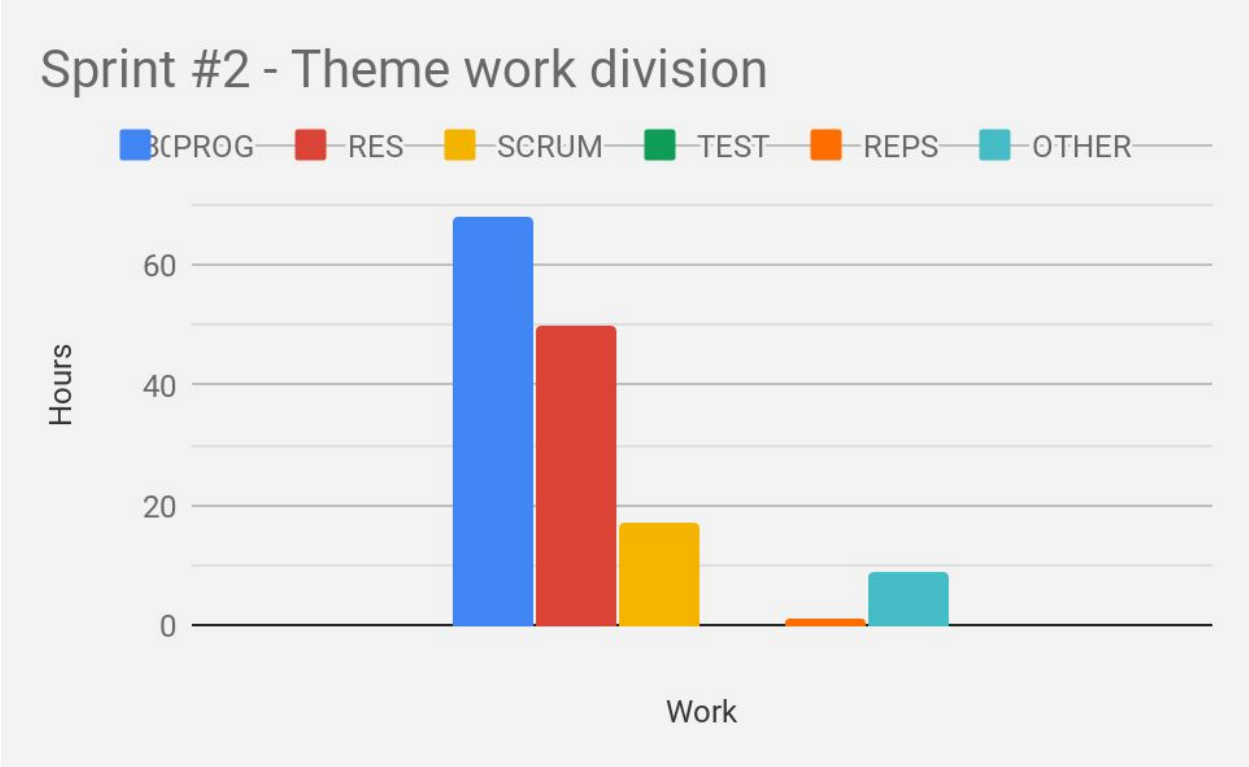
Story #32 was partially finished, 2 story points out of 5. Sound data from Polly was successfully received and hosted in the S3 cloud. The last step was to route it via SSML back to Google

Home for playback which was not finished in time. This story will continue in a different form in the next sprint. The description of this story was changed in the beginning of sprint #3 to

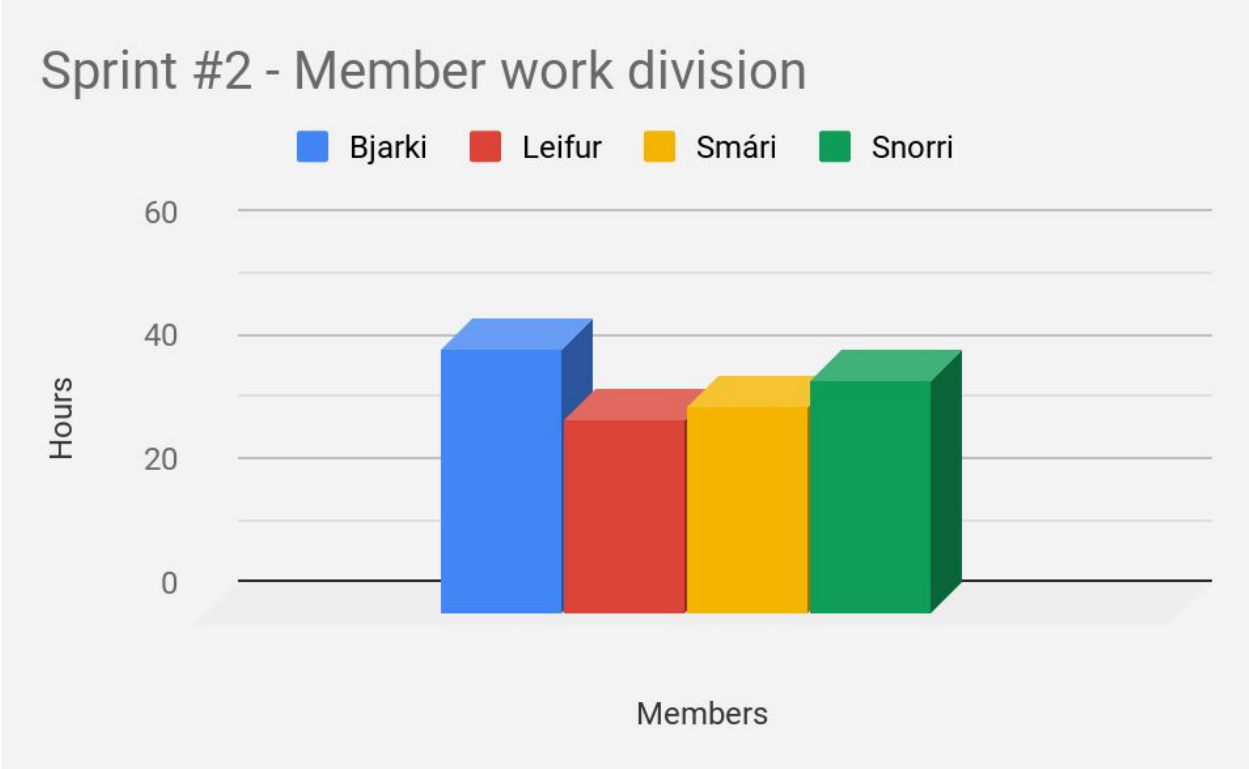
“As a developer I want to receive sound from AWS Polly and route it to app”.

Story #13 was almost done. It was successfully done on day one of sprint #3. That was right before the infeasibility of using Google Assistant was found out and that idea was scrapped.

8.3.3 Sprint work division



The theme of this sprint reflects quite well in this bar graph. Most hours were spent on programming tasks regarding Google Actions and the fulfillment service. A lot of time also went into researching Google Actions further. That is the PREP column, for preparation. At this stage, most of the reports had already been set up in sprint #1 and, thus, much less time had to be spent on them.



It is clear, from this bar graph, that Bjarki intended to make up for his absence in sprint #1. He worked quite a few more hours than other team members. Otherwise, the division was quite even.

8.3.4 Sprint retrospective

This sprint went all right. A satisfactory amount of story points were finished. Everyone was happy.

It would have been better, however if the number of workdays had been decreased before or during the sprint planning session. It was a little bit awkward updating the hour log and burndown charts that relied on team members be working on Thursdays and being able to finish 41 story points in one sprint. Hopefully a change like this will not occur again.

Another thing that could perhaps been done at the very start of the sprint was to immediately assess if it was possible to create an Action for Google Assistant which speaks Icelandic. It was Googled early on but with limited results and assumptions being relied upon, the project continued with these assumptions. It was not until the latter half of the second week that results were really concluded about this. Assumption would not again be relied upon, even if they sound logical. When it comes down to it, facts are all that matters.

8.4 Sprint 3

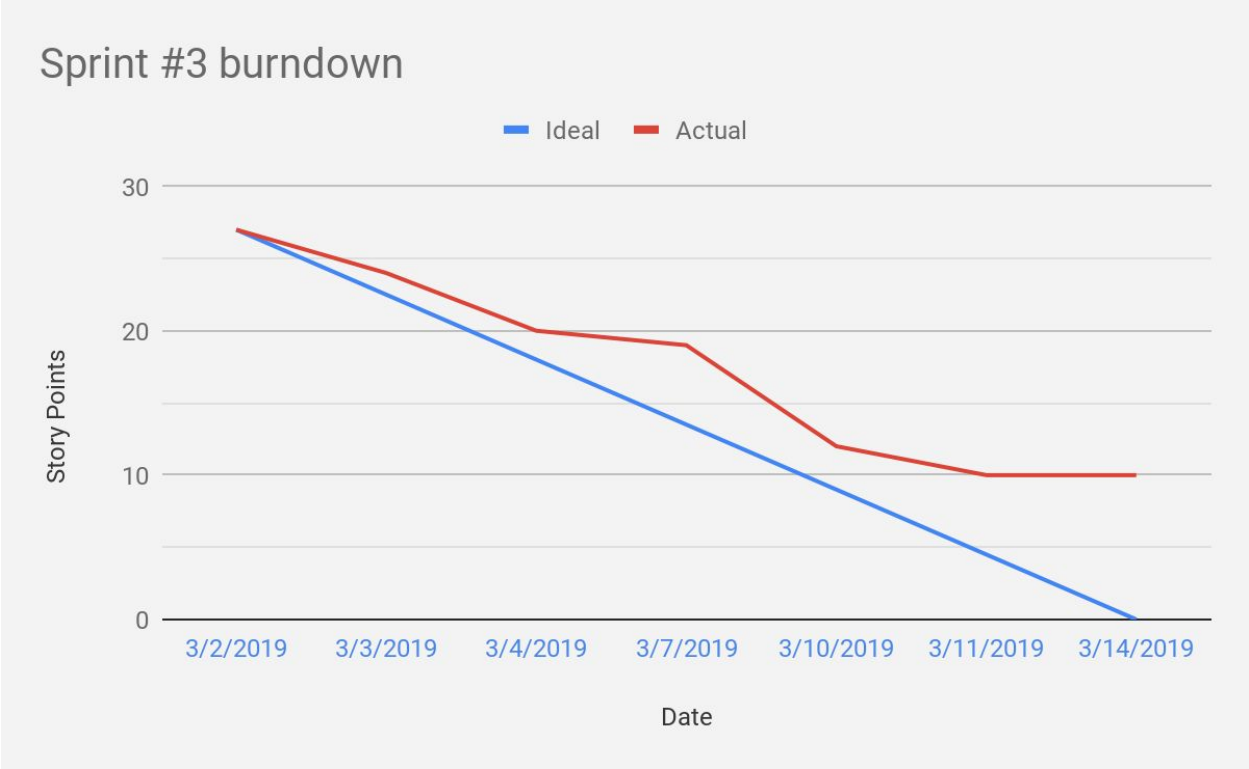
March 3rd - March 14th

The theme of this sprint was programming. This is the sprint where serious implementation of Talbankinn began. After two weeks (sprint #2) of false hopes of implementing Talbankinn as a custom smart speaker plugin, it was settled on an alternative system design which was known to be feasible. It was the initial design, as seen in the Design chapter (ch. 4).

The reason why the initial design by the end of sprint #1 was abandoned is because it was unfit to be used as a smart speaker plugin. Thus, with this design abandoned, sprint #2 was spent on a different design (see smart speaker research, ch. 7.4.1.2) until it was discovered by the end of that sprint that smart speakers are completely unable to transcribe Icelandic language.

That lead to the point of returning to the initial API design with no hope of having it capable of being used with smart speakers, at least not at the current time. In short, this design revolves around having a simple app which records user's voice, sends it to the API, which is implemented such that it speaks to the required services (Google SST, MS luis.ai, AWS Polly), performs some corresponding logic, builds an answer sound file which it sends back to the app. The app then plays the sound file and retrieves the next command from the user.

8.4.1 Sprint burndown



This is the second sprint finished after decreasing the workload by one day. That change decreased sprint capacity from 41.1 to 27.4 story points per sprint. During this sprint, however, the decreased capacity was taken into account which put a 27 story points worth of stories into the sprint backlog. This is contrary to sprint #2 where 42 story points were inserted into the sprint backlog because it was decided, *during* that sprint, to decrease the workload. That resulted in a burndown chart which looked like the team could not finish the backlog.

As can be seen on this chart, the ideal workflow was followed quite closely but ultimately could not be finished on time. Some stories were partially finished or not at all. Most of what was left unfinished were user stories who relied on developer stories to be finished first. As is quite common during the beginning of implementation of large projects, there were some minor technical hiccups. The largest one being difficulties to convert sound recordings to a certain format so that Google SST can work with them. That, of course, is vital for the user stories to be implemented. This is a task which will be continued with in sprint #4, along with the ones that were not finished.

8.4.2 Sprint backlog

Partially finished stories are marked with a fraction in the Finished column. It is to be read this way: *story points finished / total story points of story*.
If not a single task was finished, the finished column simply contains *No*. Fully completed stories are marked with yes.

In the *Pts.* column, 5(3), for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

#	Priority	Type	Description	Pts.	Started	Finished
6	**A**	Research	As a developer I want to research how to create a skill on Amazon and Google	5(2)	Sprint #1	Yes
32	**A**	Developer	As a developer I want to receive sound from AWS Polly and route it to the app	5(3)	Sprint #2	(2/3)
33	**B**	Developer	As a developer I want to set up Landsbankinn's mock API	5	Here	(3/5)
34	**A**	Developer	As a developer I want to be able to transfer sound from app to API	8	Here	(6/8)
10	**A**	User	As a user I want to know the exchange rates	3	Here	Yes
11	**A**	User	As a user I want to know the share rates	3	Here	No
14	**A**	User	As a user I want to know the balance of my accounts	3	Here	(1/3)

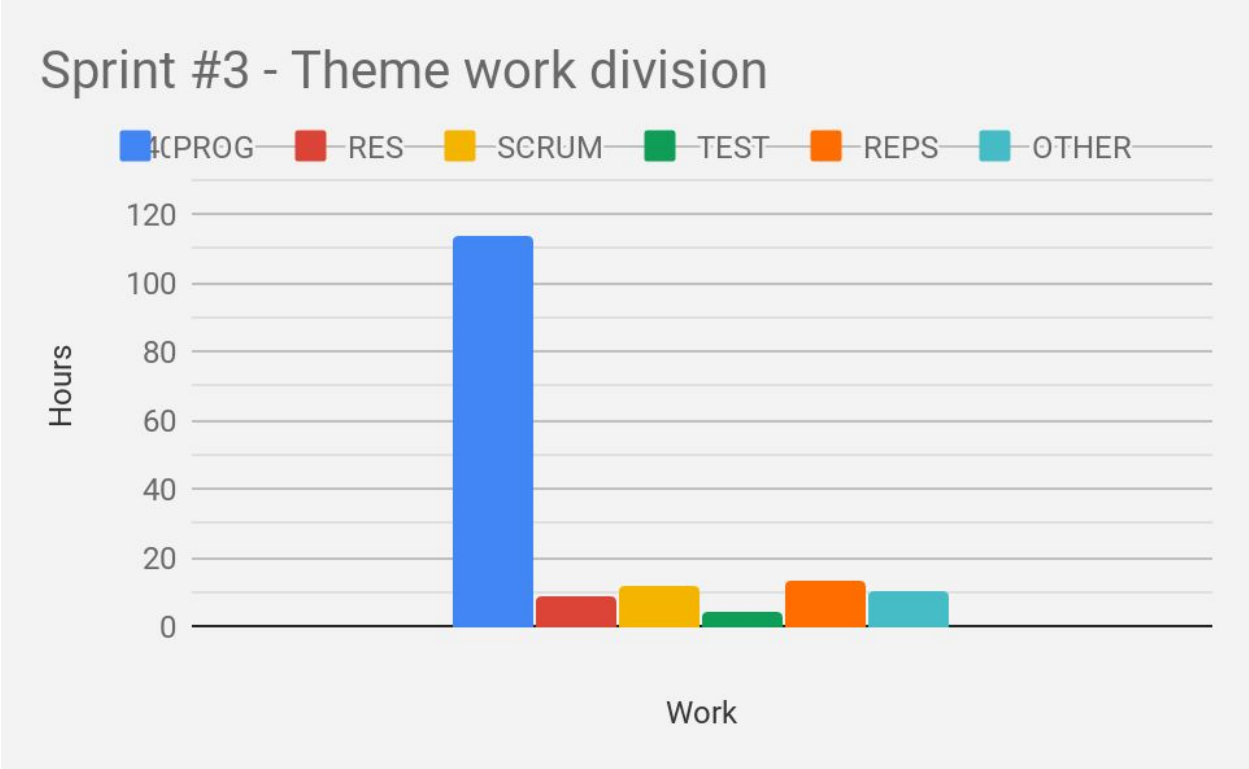
Only two stories were fully finished this sprint. Taking into account the partially finished stories, though, 17 story points were finished out of 27.

The smart speaker research turned out to be much more extensive than initially thought. The research started in sprint #1 and finished in the earlier half of this sprint.

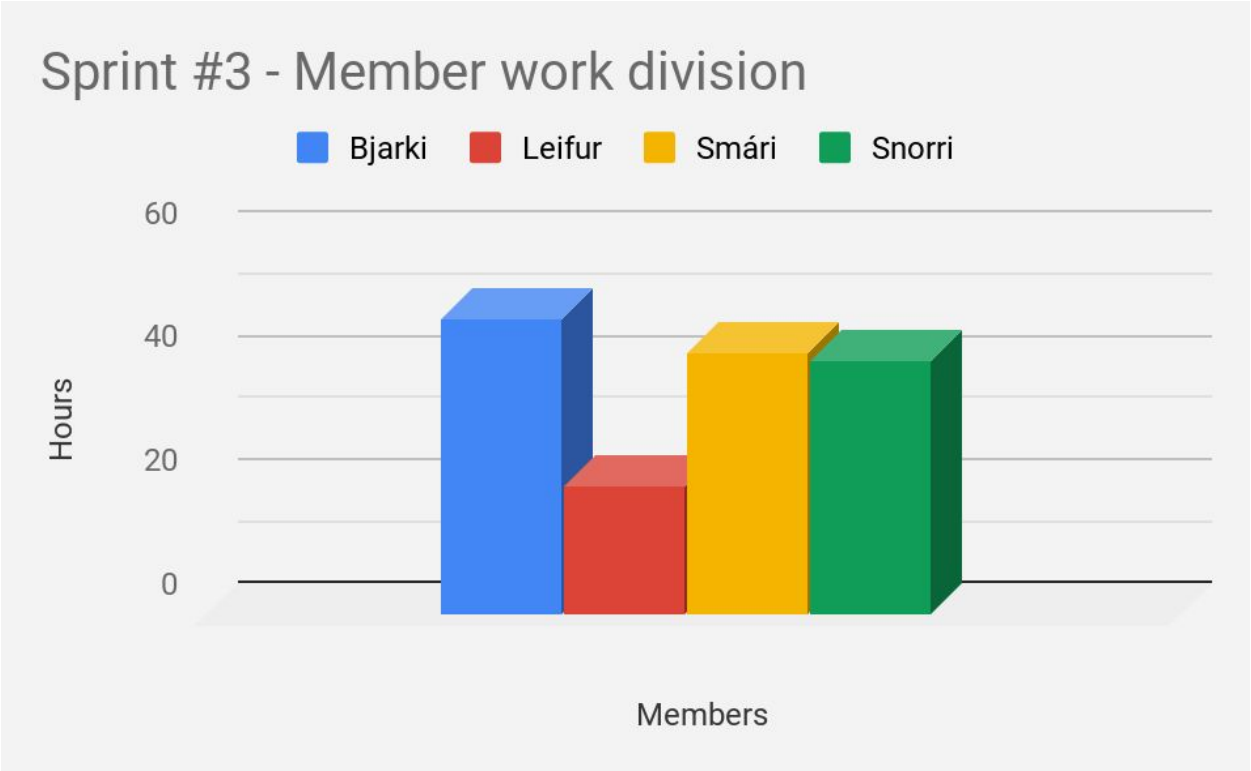
It is debatable whether story #10 is actually finished. It is three tasks were: a) Create relevant intent and entities, b) Successfully get data from developers.landsbanki API and c) Build an answer string and return to app. These were all finished. However, because not all of the required developer stories had been finished by the end of the sprint, this action was not performable.

This pointed to that a user story should be defined more carefully, such that the action in the story describes is actually performable when all tasks are done.

8.4.3 Sprint work division



This bar graph reflects, quite obviously, the theme of this sprint. Programming. 75% of the time was spent programming. This mainly contains adding voice recording features to the app, defining and setting up Talbankinn API along with the Mock Data API. The PREP column demonstrates the time spent on smart speaker research. The largest change from previous sprints is the major reduction of time spent on researches. At this stage, most of the researches had already been made and the reports written. OTHER contains meetings, mostly. The amount of time spent on meetings is quite steady and sees inconsiderable changes between sprints.



Bjarki finally managed to work in an even amount compared to other team members. The workload was surprisingly evenly distributed, which is good. Leifur was missing one day because of an exam.

8.4.4 Sprint retrospective

During this sprint, there was a lot of workload in other courses for most team members. Because of that, there was not too much time to work on the project. The team members were also not as disciplined as they wanted to and did not take regular breaks. Despite of that, this sprint went quite well. Better than expected. A lot of groundwork was covered for the product before the notorious exam sprint (#4) began. That was very valuable, to have something concrete to continue with after the exams.

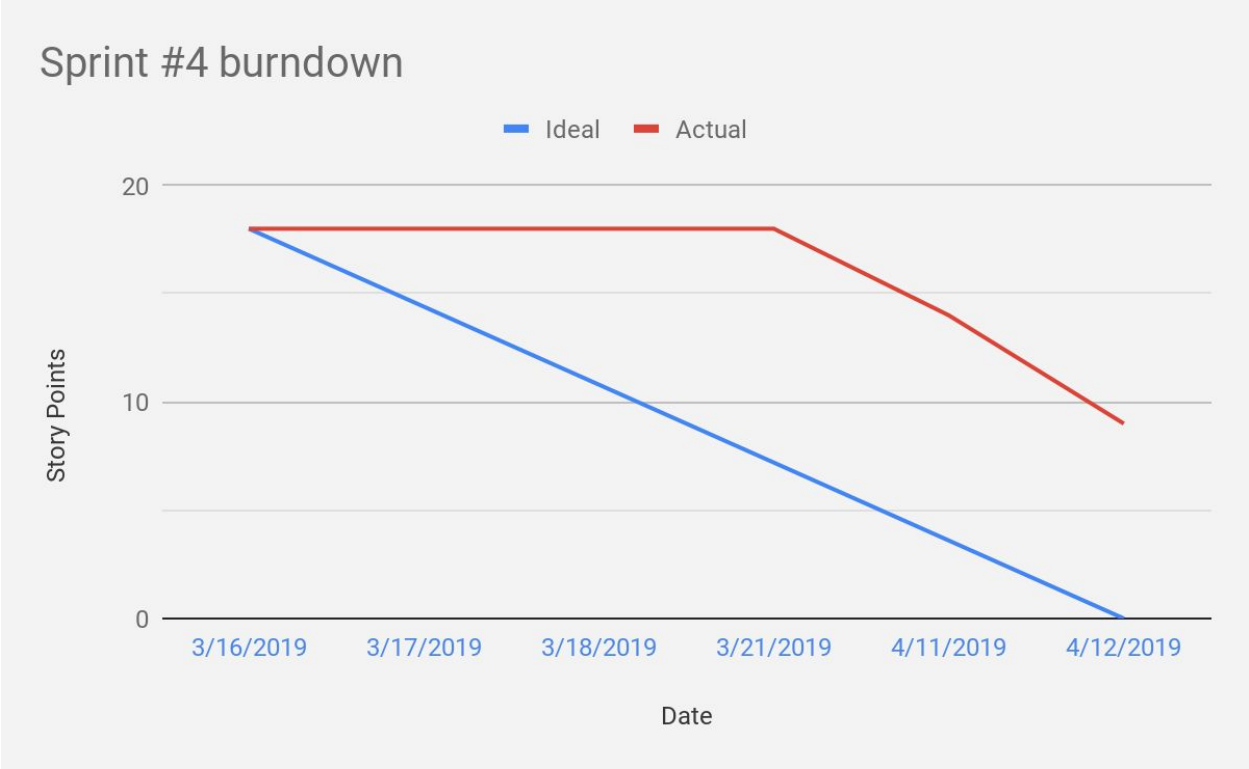
8.5 Sprint 4

March 17th - April 13th

This sprint was different than most of the other ones. First off, it was much longer than other sprints and secondly, despite of that, much less work was done than in other sprints. If there is any theme applicable for this sprint, it may be said that it was wrapping-up of other courses and exams. This kind of sprint had been foreseen from the beginning of the project, where heavy workload of other courses at the end of the semester as well as exams would consume most of the time. Consequently, a little work was done in this sprint.

Those who had time, though, continued working on the project. This sprint consisted, for the most part, of leftover tasks from stories of sprint #3. Thus, it was a continuance from sprint #3. The tasks included continued programming of Talbankinn API, implementation of mock data API and getting recorded sound to be interpreted clearly by the Google speech-to-text API.

8.5.1 Sprint burndown



Due to the exams, the sprint capacity was much lower than usual. Because every member of the team had exams of varying difficulties and at differing dates, it was very difficult to decide

the capacity. Thus, this sprint assumed the same previous sprint capacity of 27.4 story points. Because it was not expected that much work would be done in this sprint, it consisted of only 18 story points. It is quite evident, looking at the chart, that there was not much time to work on the project. As a consequence 9 story points were left by the end of it.

8.5.2 Sprint backlog

Partially finished stories are marked with a fraction in the *Finished* column. It is to be read this way: *story points finished / total story points of story*.

If not a single task was finished, the finished column simply contains *No*. Fully completed stories are marked with *Yes*.

In the *Pts.* column, *5(3)*, for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

#	Priority	Type	Description	Pts.	Started	Finished
32	**A**	Developer	As a developer I want to receive sound from AWS Polly and route it to the app	5(1)	Sprint #2	No
33	**B**	Developer	As a developer I want to set up Landsbankinn's mock API	5(2)	Sprint #3	No
34	**A**	Developer	As a developer I want to be able to transfer sound from the app to the API	8(2)	Sprint #3	Yes
11	**A**	User	As a user I want to know the share rates	3	Sprint #3	(2/3)
14	**A**	User	As a user I want to know the balance of my accounts	3(2)	Sprint #3	Yes
35	**A**	Research	As a developer I want to research how users phrase specific requests	8	Here	(3/8)

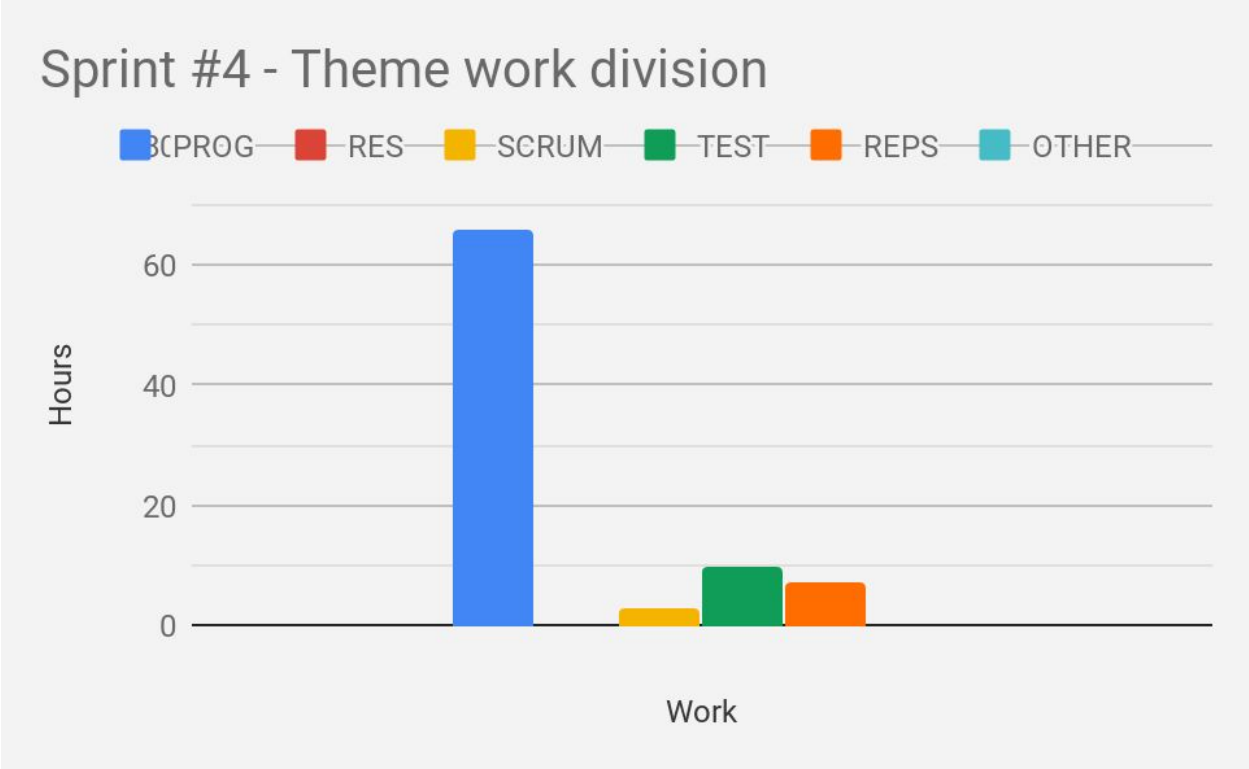
There were quite many partially finished stories in sprint #3 which followed into this sprint. Some of them were finished while others were not. Leifur added story #35, started researching it and finished one of its tasks. The biggest breakthrough was, perhaps, when Bjarki managed to find a better way to record sound, finishing the important story of #34.

Most of these stories were continued in sprint #5.

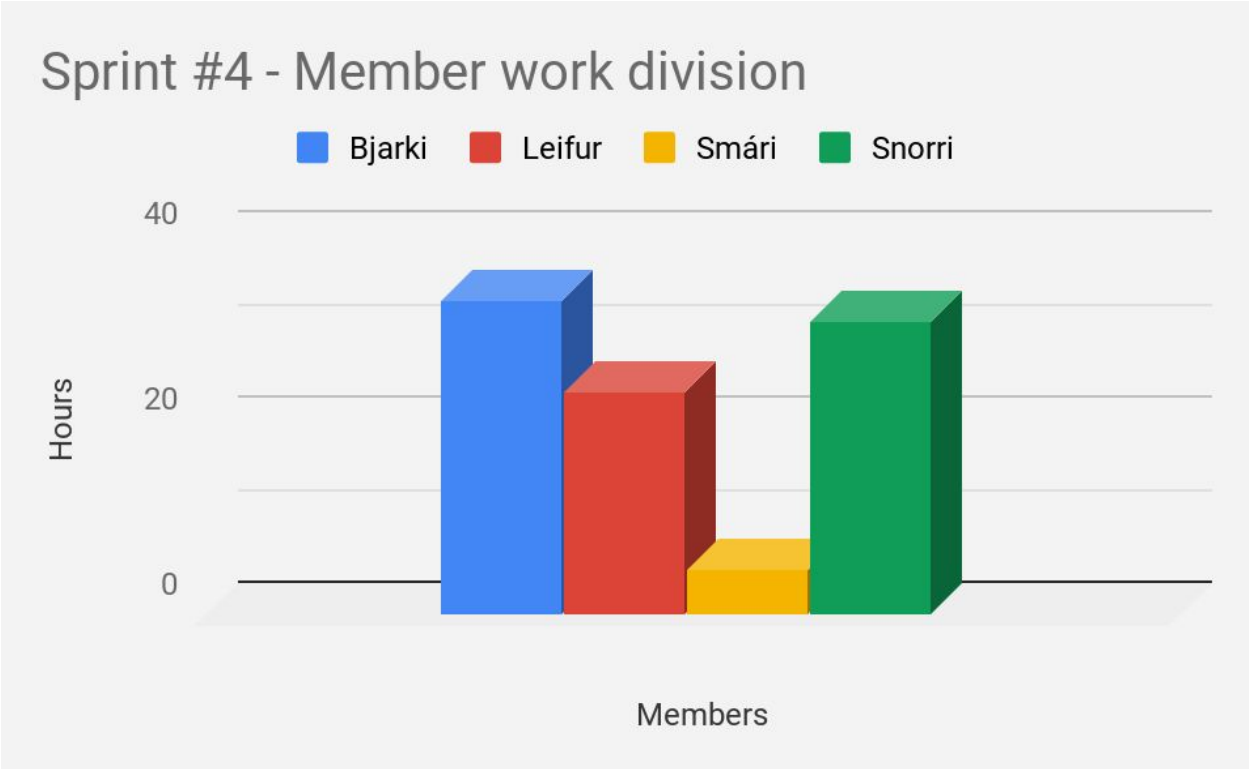
Even though, story #14 is marked as done, it does not mean that the users can perform the task. All of the tasks, regarding to that story, were successfully completed but the technical aspects of the system are still too lacking for the user to actually perform it. The same problem

occurred with story #10, see chapter 7.5.2. Nevertheless, as soon as these technical factors are taken care of, the user should immediately be able to perform the task.

8.5.3 Sprint work division



The bars do not reach as high on this graph as they do on other sprints' *Theme work division* graphs. That reflects quite clearly the “theme” of this sprint which was, as mentioned above, wrapping-up of other courses and final exams. The little time team members had was mostly spent on programming tasks. It is interesting to see the TEST bar finally raise a little as the testing process finally started here. Due to the team members being busy on other matters, such as exams, there were no meetings in this sprint.



The team members worked in mostly equally amounts. It is evident, though, that the exams took the toll on some people more than others.

8.5.4 Sprint retrospective

Due to the nature of this sprint, there was little to retrospect on. This sprint, nevertheless, went similarly as expected. The project carried somewhat forward, despite of the pressure which followed other courses' projects and the exams. Everyone agreed it was very important not to abandon the work completely during this sprint. It worked out quite well.

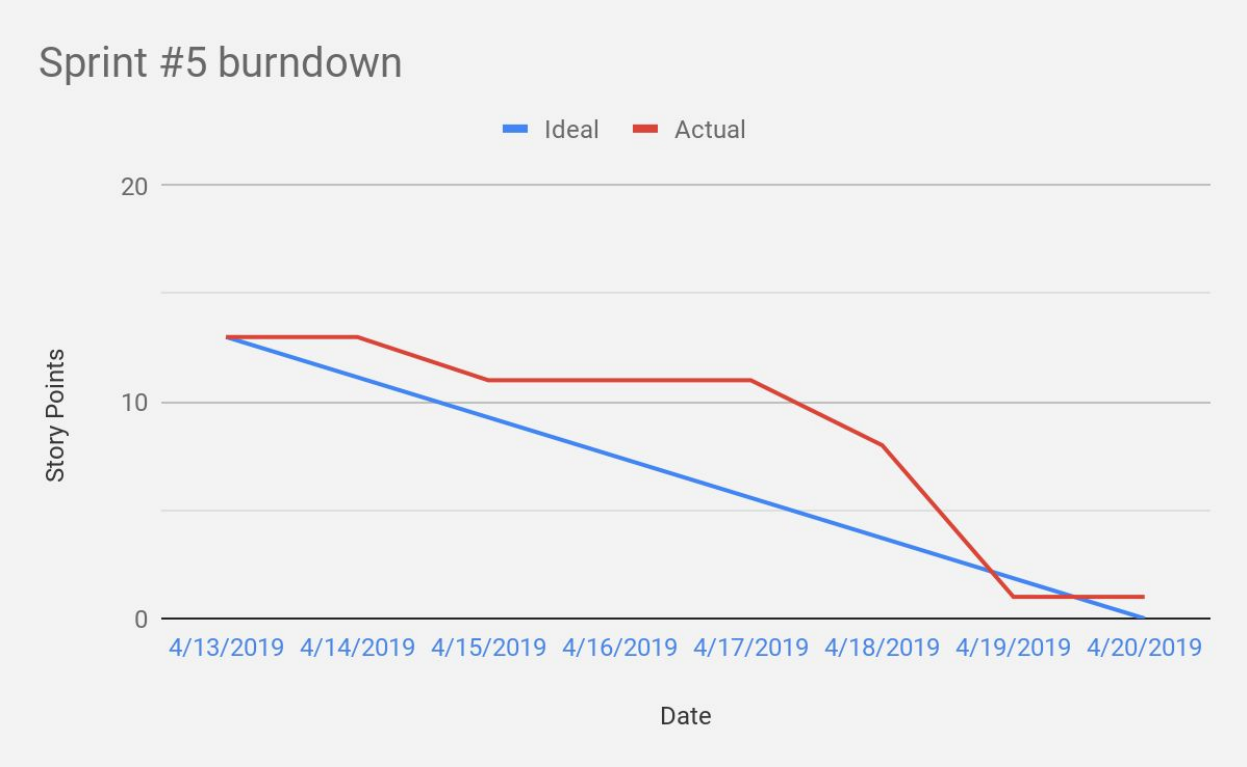
8.6 Sprint 5

April 14th - April 19th

This sprint was originally set to be from April 14th to April 20th but almost all story points were finished by the end of April 19th. See further explanation below, in chapter 7.7.1.

The theme of this sprint was getting things going again after a lengthy exam pause. It was decided to have a slow start, while everyone was recovering from a lengthy and difficult exam period. Some team members finished exams very late and were unable to join up until April 18th. Consequently, it was decided to put only 13 story points worth of stories in the backlog of this sprint.
It may be said that the secondary theme of this sprint was user stories as, for the first time, there were many user stories being worked on. All of them were finished.

8.6.1 Sprint burndown



As stated before, this sprint was originally intended to be until April 20th. This is reflected in the burndown chart above. By April 19th, there was only one story point left which was, at that time, not likely to be finished anytime soon as its corresponding task was incorrectly weighed. Thus, it

was decided to move that single task, with one story point, to sprint #6. It was the task of receiving sound data in the mobile app and play the sound. Sprint #6 started at April 20th.

As the chart shows, the sprint started nicely but progress slowed a bit down after April 15th. It quickly gained pace on April 18th and all tasks were finished except the one that was mentioned before. Most of them were user story tasks.

This is the first sprint that (almost) all story points are finished.

8.6.2 Sprint backlog

Partially finished stories are marked with a fraction in the Finished column. It is to be read this way: story points finished / total story points of story.

If not a single task was finished, the finished column simply contains No. Fully completed stories are marked with Yes.

In the Pts. column, 5(3), for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

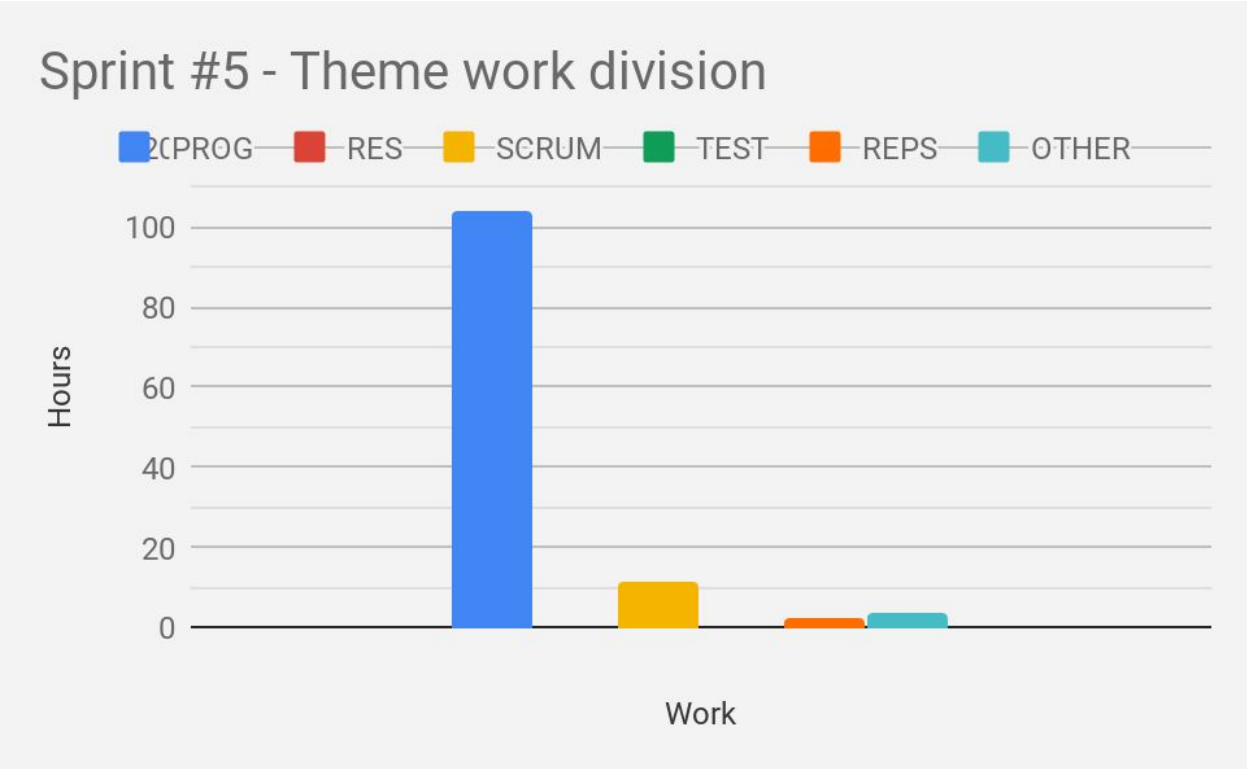
#	Priority	Type	Description	Pts.	Started	Finished
32	**A**	Developer	As a developer I want to receive sound from AWS Polly and route it to the app	5(1)	Sprint #2	No
33	**B**	Developer	As a developer I want to set up Landsbankinn's mock API	5(2)	Sprint #3	Yes
11	**A**	User	As a user I want to know the share rates	3(1)	Sprint #3	Yes
9	**A**	User	As a user I want to get an overview of unpaid bills	3	Here	Yes
12	**A**	User	As a user I want to get information about a specific unpaid bill	3	Here	Yes
15	**A**	User	As a user I want to know the status of my credit card (funds available + balance)	3	Here	Yes

This sprint inherited unfinished stories #32, #33, and #11 from previous sprints. Two of them were finally finished but story #32 had one task left which was not finished. (Fun fact: it was finished in the very beginning of sprint #6).

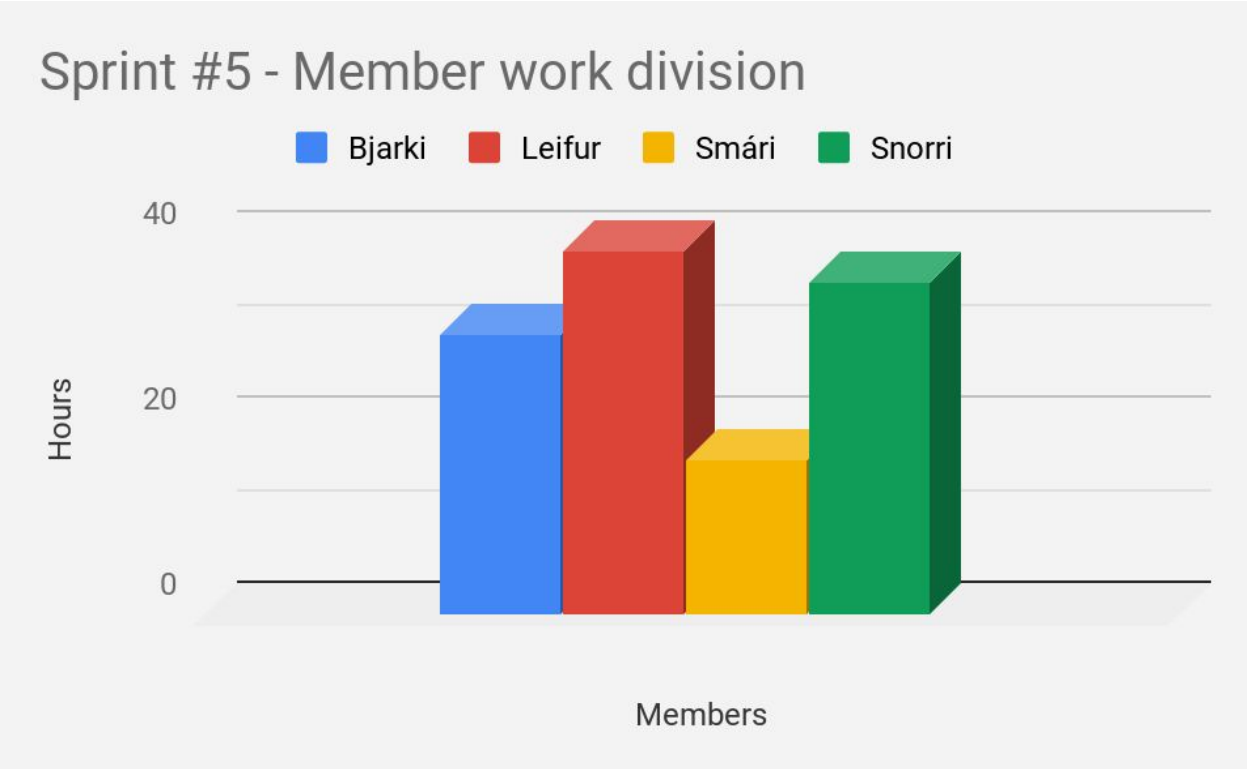
This sprint introduced three new user stories: #9, #12, #15. Their corresponding tasks were all finished along with what was left of user story #11.

It should be noted, though, that although these user stories were finished, the system was not in a state such that it was able to let a user actually perform the actions they describe. The only thing that was needed for that was the single unfinished task of story #32 which was an audible playback of sound in the mobile app.

8.6.3 Sprint work division



As in the previous sprint, most of the time was used for programming. A lot of that time was used to move Talbankinn's API and the mock data API, respectively, to the Azure cloud platform as well as finishing a handful of user stories. The planned user testing was put on hold until in sprint #6, thus no testing was performed in this sprint.



As before, the member work division was mostly evenly distributed. Bjarki and Smári, although, did not start as early to work on the project as Leifur and Snorri. That was mainly because of the exam period. Smári, for instance, did not get off until April 18th. Nevertheless, the sprint started off well and an acceptable amount of work was finished.

8.6.4 Sprint retrospective

Over all, this was a successful sprint and a good start after the lengthy exam period.

Some major milestones were made, for example getting Talbankinn API and Mock Data API hosted on the Azure cloud platform. These were some worthy obstacles that were tackled quite well. Over all, the project as a whole looked good by the end of the sprint.

There were no major setbacks during the sprint. However, no sprint is perfect. One member got quite sick and did not manage to show up half of the time. He worked from home, though, so it went well after all. There was also some lack of general organization, for instance there were no regular breaks and members seemed to show up to work at semi-random times. That, although, is quite normal as things were getting into the right track after the exam period.

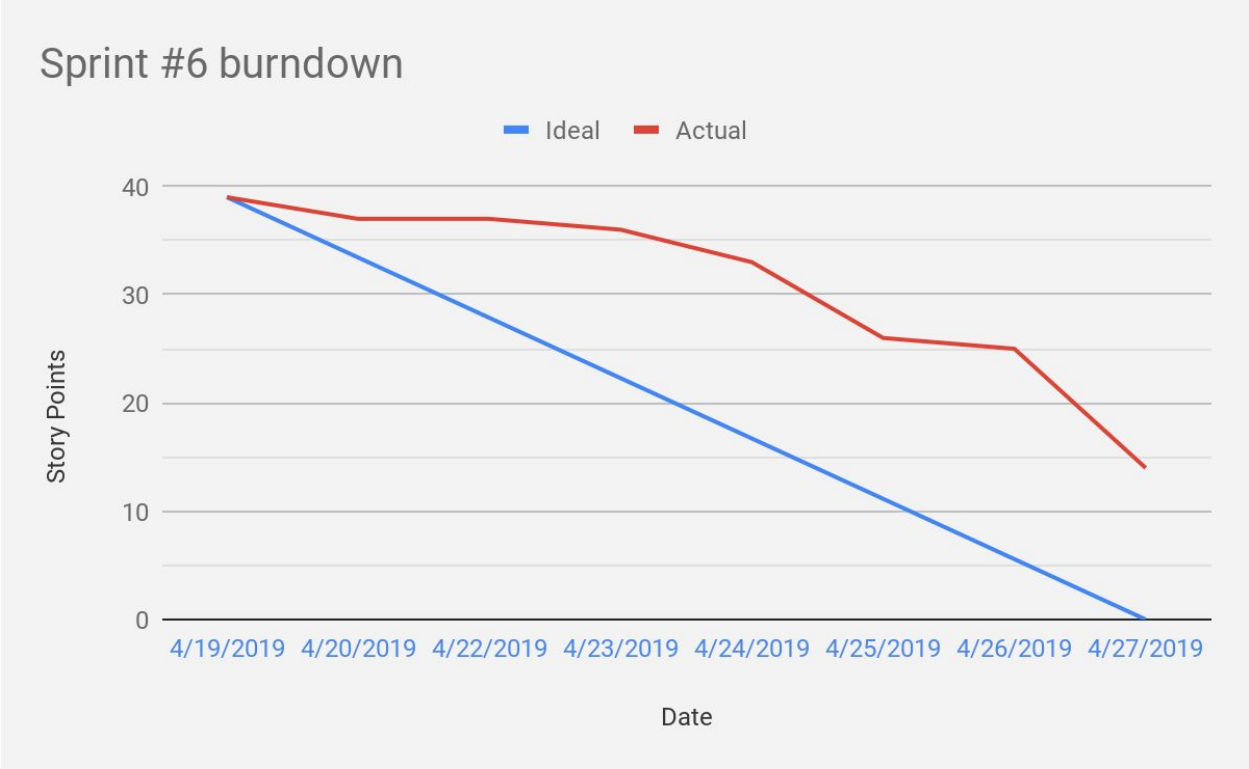
8.7 Sprint 6

April 20th - April 27th

This was the first full-speed sprint after the exam period, with a story point count of 39. The theme of this sprint was, with no doubt, the initial implementation of multi-step user requests. That is, requests that comprise of an actual dialog between the user and the system. The user asks Talbankinn to do something, Talbankinn asks for more information, the user answers with the information, Talbankinn processes the information and may ask for more information. This may be in several steps. The implementation of such requests is much more complex and intricate than single single-step requests because they requires the API to be able to distinct between users and each user must keep a state.

The biggest breakthrough of this sprint was that the implementation of receiving of sound data from API to app and play it in the app was completed. This meant that the whole “pipeline”, app record sound->api receive sound->api process data->api send sound answer->app receive sound anwer->app play sound data, had been implemented completely.

8.7.1 Sprint burndown



This is the first time since sprint #1 working at the sprint capacity of 41.1. The difference this time is that the sprint length was one week rather that two weeks. As the graph shows, the sprint started off very slowly but gained pace when it was approximately halfway through. There were 14 story points left. There were some normal setbacks due to this being the first time for everyone implementing stateful voice interaction. The remaining tasks were moved to sprint #7.

8.7.2 Sprint backlog

Partially finished stories are marked with a fraction in the Finished column. It is to be read this way: story points finished / total story points of story.

If not a single task was finished, the finished column simply contains No. Fully completed stories are marked with Yes.

In the Pts. column, 5(3), for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

#	Priority	Type	Description	Pts.	Started	Finished
32	**A**	Developer	As a developer I want to receive sound from AWS Polly and route it to app	5(1)	Sprint #2	Yes
8	**A**	User	As a user I want to transfer money from my account to a known recipient	5	Here	(3/5)
18	**A**	User	As a user I want to transfer money between two of my accounts	8	Here	(3/8)
35	**A**	Research	As a developer I want to research how users phrase specific requests	8(5)	Sprint #4	No
36	**B**	Developer	As a developer I want the Talbankinn API to distinct between multiple users and keep some state	12	Here	Yes
19	**B**	User	As a user I want to pay my bills	8	Here	(6/8)
24*	**B**	Developer	As a developer I want to have logs of failed AI text interpretations	8	Here	Yes

This sprint inherited unfinished stories #32 and #35 from previous sprints. The big breakthrough was when the last task of story #32 was finally finished. That was the task of having the mobile app playing received sound data from the API. That, as is mentioned above, marked the completion of the system pipeline. There was not time to continue as planned with story #35.

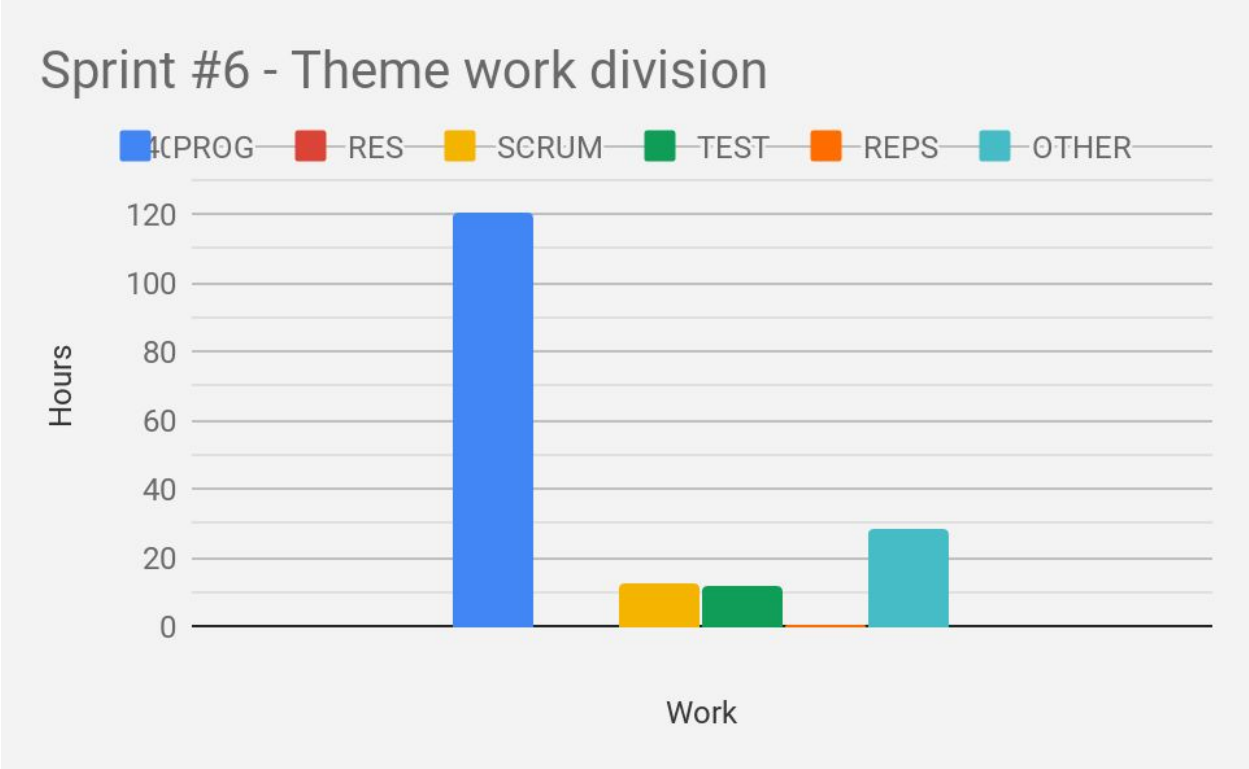
Many of the stories were left partially finished, their corresponding unfinished tasks were moved to sprint #7.

The biggest story was the research and development of multiple user distinction. A user map was implemented for the API. It uses a user ID, sent from the mobile app, as a key to map which returns a corresponding user's info, such as state and history. That story started in this sprint and was finished as well.

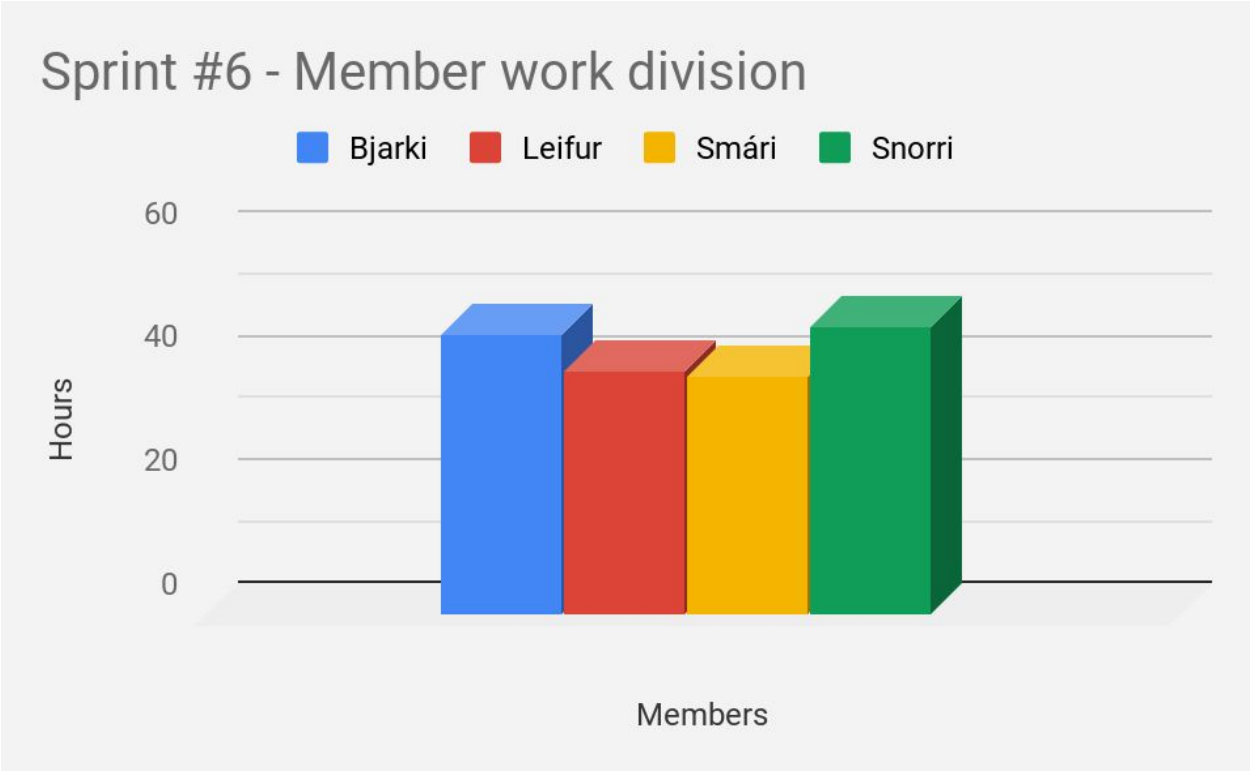
As was mentioned before, the user stories were not finished because of normal initial difficulties of implementing stateful vocal interaction.

*Story #24 happened to be completed without it being a planned subject for this sprint. Note: this is not taken into account in the burndown chart.

8.7.3 Sprint work division



This chart shows clearly the ongoing theme of programming. Even though programming is most dominant in these charts, the research factor is inevitably present behind the scenes. For example, the multistep dialog designs were being researched as they were being implemented. Most of the OTHER column contains script-making and dialog designs.



As every team member had wrapped up all other remaining academic duties, the work division was spread quite evenly. It should be noted, though, that the mean workload in hours is considerably higher than in previous sprints, considering that this sprint was only one week. This is normal, as every team member spent the majority of their free time working on the project during the sprint.

8.7.4 Sprint retrospective

This sprint went quite well, even though only 25 story points out of 39 were finished. The engine was still being warmed up for the final rounds of the project. The biggest breakthroughs were the multi-user, stateful implementations as well as finishing the pipeline by getting the mobile app to play sound.

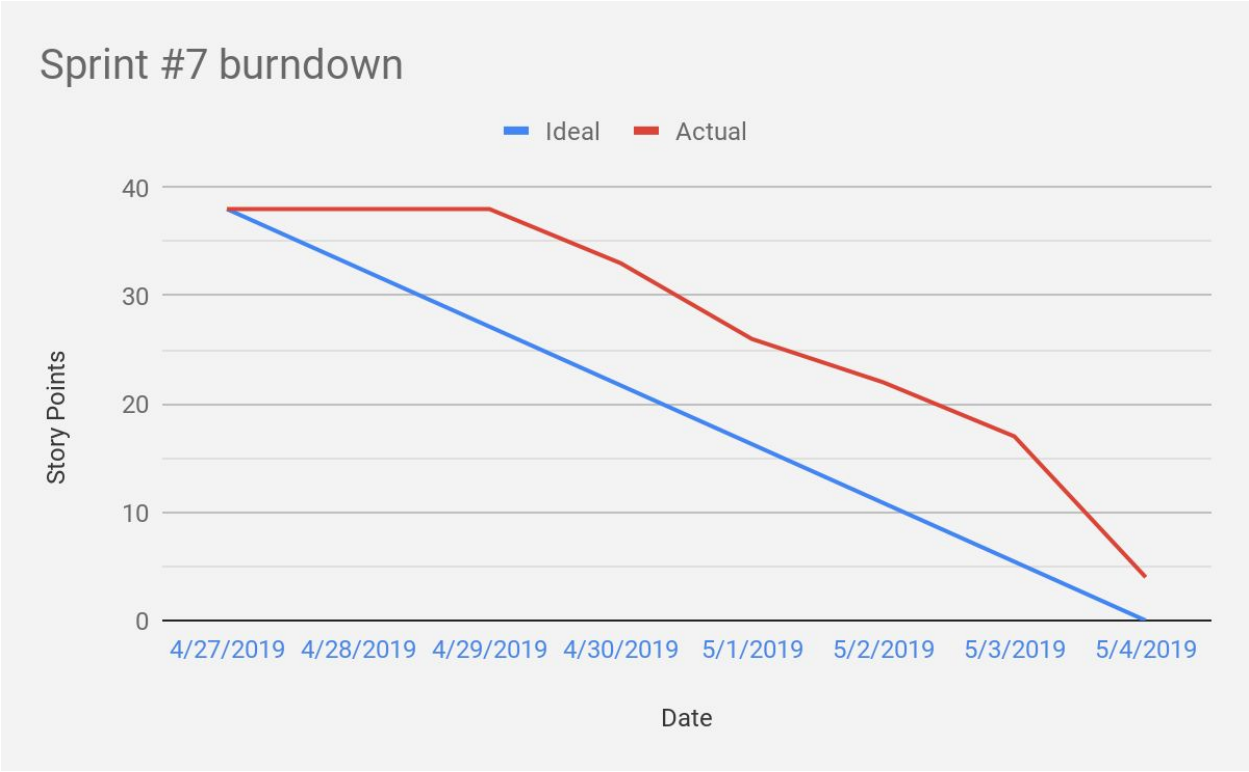
The morale was mostly good. There was, although, a little too much noise in the surrounding work area which was quite irritating and disturbed the otherwise peaceful workflow. Hopefully, Landsbankinn will be able to provide more private and less noisy working environment in sprint #7.

8.8 Sprint 7

April 28th - May 4th

This was the second full-speed sprint with a story point count of 38. The theme was continued development of the various features of *Talbankinn*. Both continued refinement of already existing features along with addition of new ones, such as currency conversion. A quite amount of time was also spent on user testing where valuable phrasing data was gathered from users with various backgrounds. The data was used to enable *Talbankinn* to understand more user utterances. A big breakthrough was made when Bjarki managed to fix a bug in React which had been around for many weeks. It was a breaking bug which prevented sound responses from playing. This was the last sprint before feature freeze.

8.8.1 Sprint burndown



The sprint capacity of this sprint was 41.1 story point but it contained 38 story points. As usual the sprint started off slowly but the burndown gathered some pace on April 30th. The pace was kept quite steady for most of the time.

Even though it was behind schedule for the entire sprint, 34 out of 38 story points were finished. This was the best completed/uncompleted points ratio at this point in the project. As before, the few remaining task were moved to sprint #8.

8.8.2 Sprint backlog

Partially finished stories are marked with a fraction in the *Finished* column. It is to be read this way: *story points finished / total story points of story*.

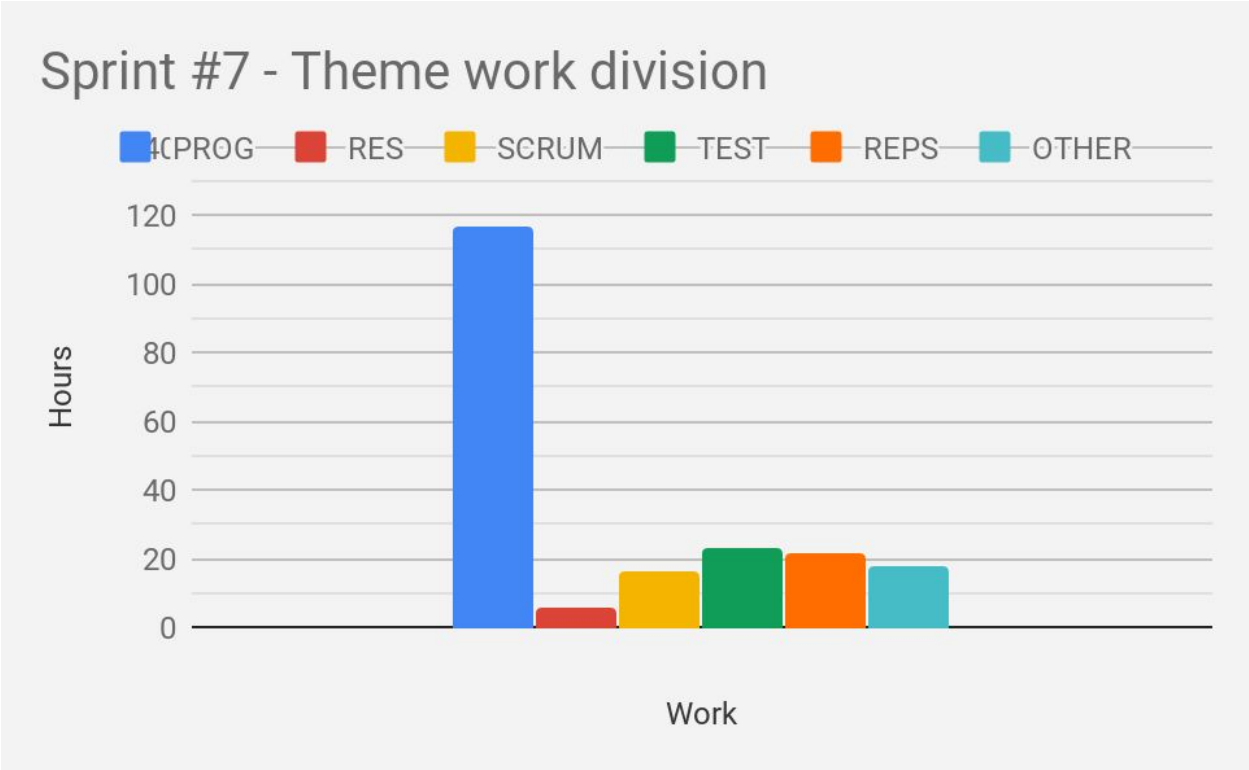
If not a single task was finished, the finished column simply contains *No*. Fully completed stories are marked with *Yes*.

In the *Pts.* column, *5(3)*, for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

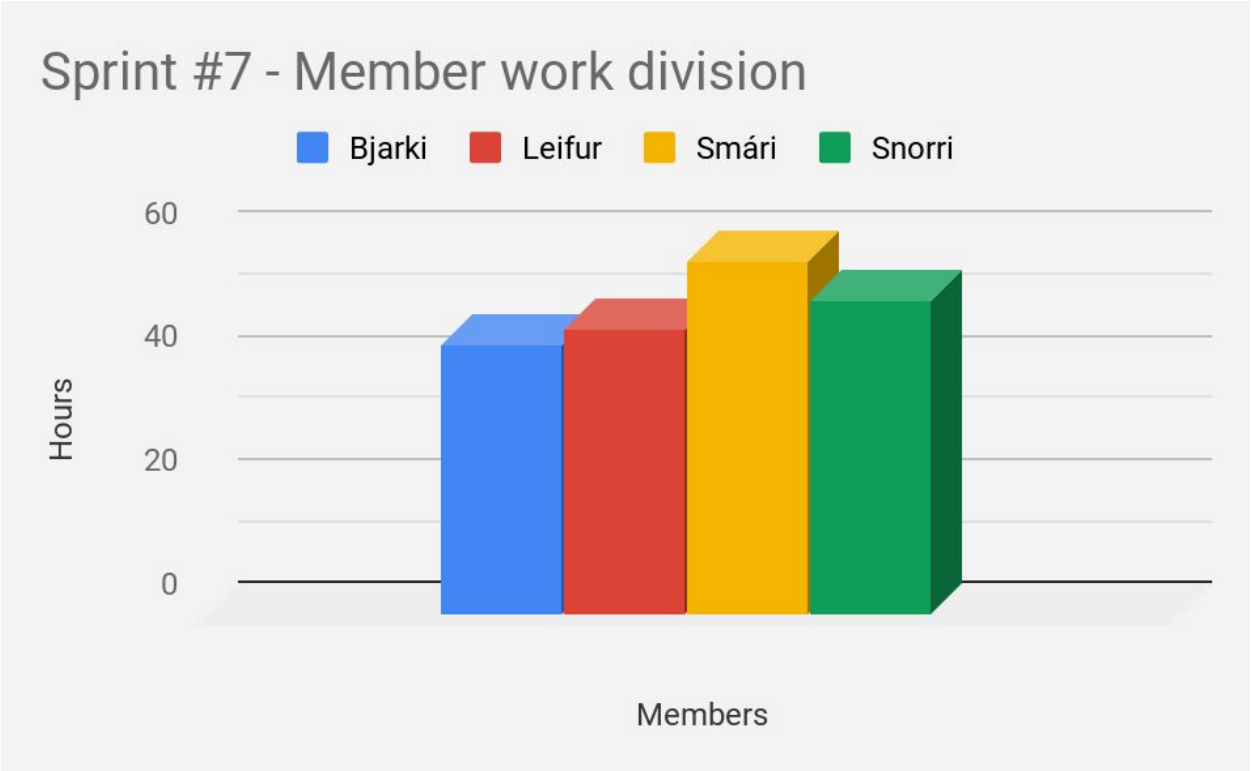
#	Priority	Type	Description	Pts.	Started	Finished
8	**A**	User	As a user I want to transfer money from my account to a known recipient	5(2)	Sprint #6	Yes
18	**A**	User	As a user I want to transfer money between two of my accounts	8(5)	Sprint #6	Yes
35	**A**	Research	As a developer I want to research how users phrase specific requests	12(9)	Sprint #4	(7/9)
19	**B**	User	As a user I want to pay my bills	8(2)	Sprint #6	Yes
42	**B**	User	As a user I want to convert one currency to another	5	Here	Yes
37	**A**	Research	As a developer I want to research and report on how certain dialogs are structured by the API	5	Here	(3/5)
38	**A**	Research	As a developer I want to write a report on the process it took to record and send sound	3	Here	Yes
39	**A**	Research	As a developer I want to add details about deployment features to the design chapter	2	Here	Yes
40	**A**	Research	As a developer I want to make a short report about authentication possibilities	5	Here	Yes
41	**A**	Research	As a developer I want to report on the difficulties that were encountered with Icelandic declension	3	Here	Yes

As can be seen above, the sprint contained 38 story points and four story points were left unburned. This sprint went very well, a lot of ground was covered and many tasks were finished. Never before have as many stories been started and finished in the same sprint. Story #35 happened to be larger than originally thought and its worth was increased from 8 points to 12. The actual testing was finished but the writing of the report remained unfinished by the end of the sprint. Same goes with story #37, the dialog trees had been drawn but the written report remained unfinished. Stories #37 - 40 were created right before this sprint and added to it. These are research reports that were felt needed in the final report before the upcoming due date of the project. All of them were finished, except for story #37. Story #41 was not originally in this sprint but was done anyway. This is not displayed on the sprint burndown chart.

8.8.3 Sprint work division



A little bit of every theme was touched upon in this sprint, although programming was, obviously, the dominating theme. As before, the sprint theme is clearly reflected in this chart as most members spent most of their time developing the various features of the project. As was mentioned before, there was a considerable amount of time spent on testing and report writing as can be seen in the chart.



The record of amount of hours worked in one sprint was broken in this sprint. A total of 201 hours were spent working on the project, the old record was 174 hours. The amount of time was mostly evenly spread between team members. Smári decided to add up for the work hours he missed in sprints #4 and #5.

8.8.4 Sprint retrospective

This sprint was undoubtedly the most productive sprint yet at the point it was finished. Every member worked very actively and a large amount of stories and tasks were finished. As the sprint was around halfway through, a pleasant news were received from the product owners at Landsbankinn as they had found a new workplace. The base of operations was moved from the department of computer science at Reykjavik University to Fjártækniklasinn, Laugavegur 77. The working conditions there were much better in every aspect, with less disturbances and grocery stores and restaurants in walking distance.

No sprint, however, is perfect. The first few days of the sprint, before moving to the new workplace, were quite difficult. The noise was unbearable and made it very difficult to converse and test the speaking aspect of *Talbankinn*.

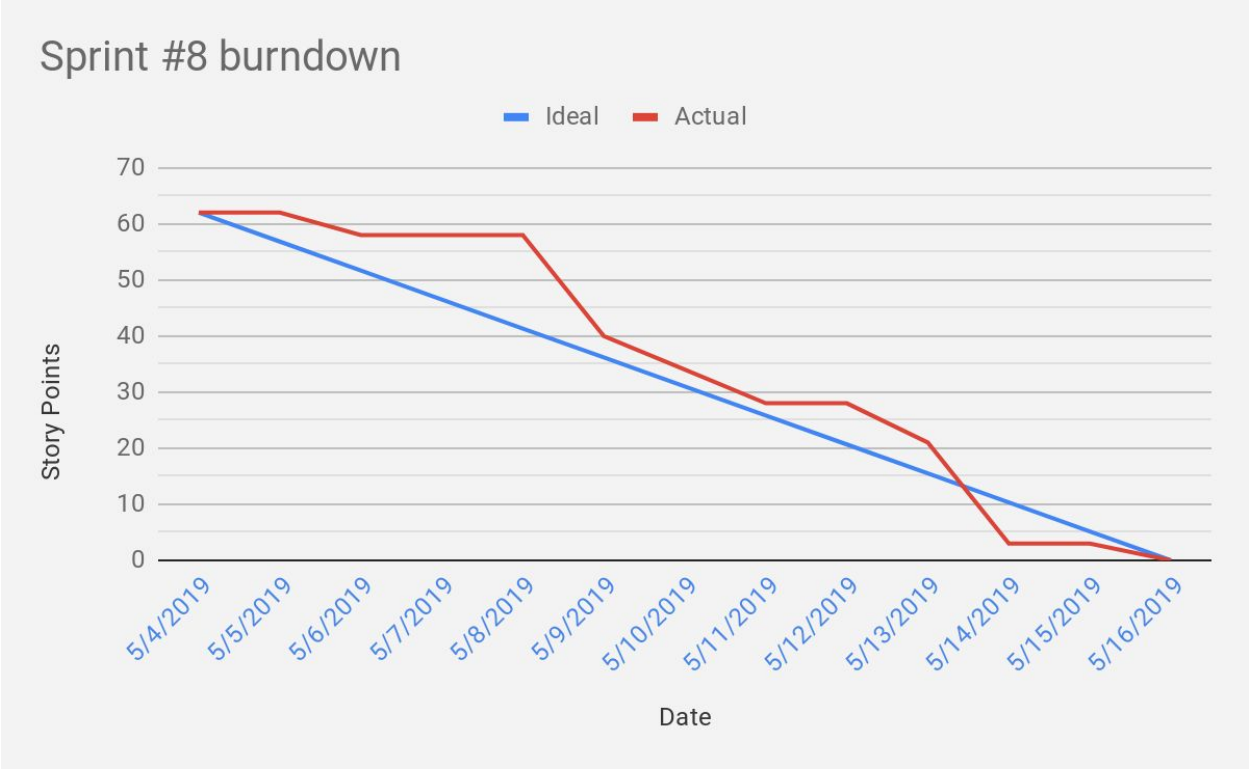
8.9 Sprint 8

May 5th - May 16th

This was the final sprint of this project. It was initially supposed to be two sprints, sprint #8 which was supposed to be one week long and sprint #9 which was supposed to be five days. Instead, it was decided to merge the two sprints into one megasprint consisting of 62 story points. The theme was refinement and optimization of already existing features as at this point in the project, feature freeze was in effect as of May 5th. The majority of the story points, or 54 of them, consisted of a new megastory which consisted of refining, improving and optimizing already existing features. These features were all products of user stories. Among other things that were done was giving the mobile app a major overhaul and writing manuals.

This last sprint went very well and it was the first and the last sprint where all story points were finished. It was also, by far, the sprint which contained most story points.

8.9.1 Sprint burndown



Due to the increased length of this sprint, its capacity was far more than in other sprints. The sprint capacity was 61.7 story points and it was utilized to its fullest as the sprint contained 62 story points. Like so many times before, the sprint started off quite slowly but as time went by

the team managed to gain pace quite fast. Eventually, it was managed to keep up the pace and by the end of the sprint, the burnt story points surpassed the ideal expected amount.

8.9.2 Sprint backlog

Partially finished stories are marked with a fraction in the *Finished* column. It is to be read this way: *story points finished / total story points of story*.

If not a single task was finished, the finished column simply contains *No*. Fully completed stories are marked with *Yes*.

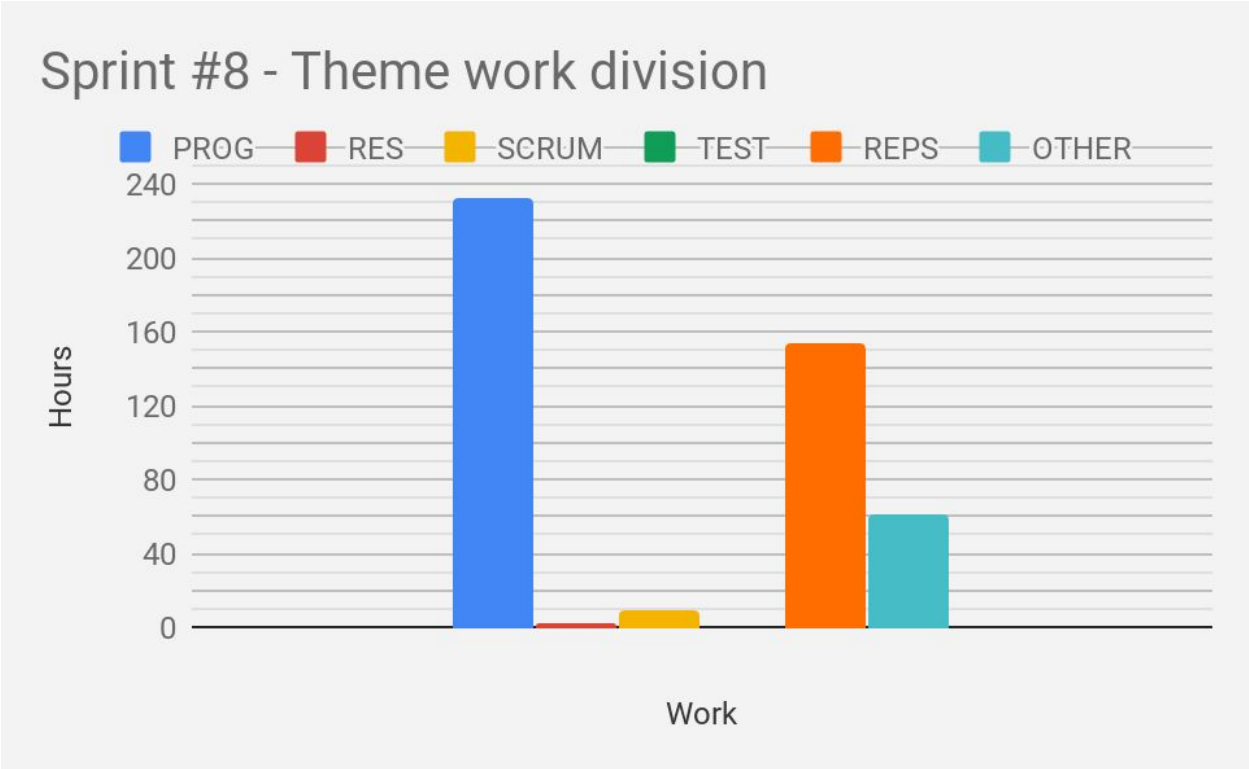
In the *Pts.* column, *5(3)*, for instance, denotes that the story is initially worth 5 points but 2 of them were already finished in another sprint which leaves only 3 points left for this sprint.

#	Priority	Type	Description	Pts.	Started	Finished
35	**A**	Research	As a developer I want to research how users phrase specific requests	12(2)	Sprint #4	Yes
37	**A**	Research	As a developer I want to research and report on how certain dialogs are structured by the API	5(2)	Sprint #7	Yes
43	**A**	Developer	As a developer I want to write a user instruction manual	1	Here	Yes
44	**A**	Developer	As a developer I want to write an operations manual	3	Here	Yes
45	**A**	Developer	As a developer I want to optimize actions and make sure they perform well	54	Here	Yes

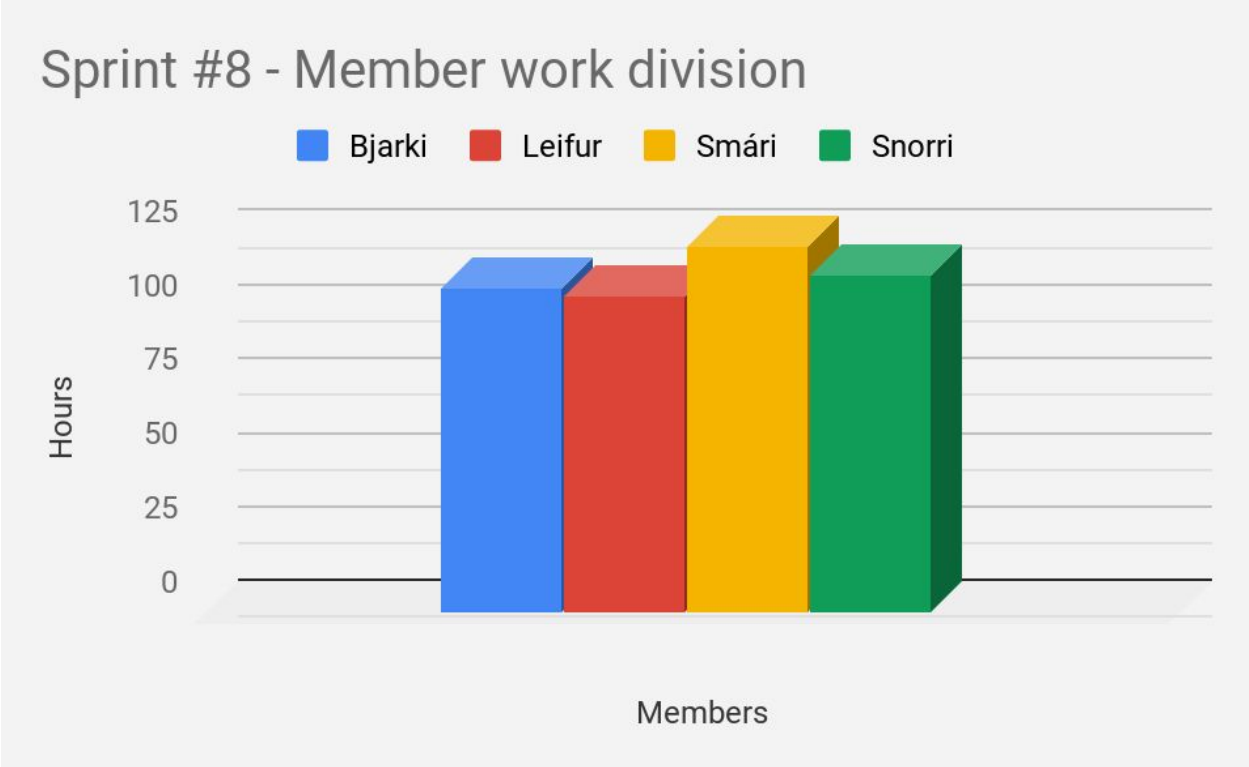
The most notable aspect of this backlog is the lack of user stories. That was due to the feature freeze being in effect. No new features were added and the stories consisted mostly of refinement and wrapping up the project. Stories #43 - #45 were created specifically for this sprint. The most notable story is story #45 with the whopping weight of 54 story points. In this story, every already added feature (user action) was improved and made more robust and complete. Some of them were only partially implemented, their implementations were finished. Other more completed features were made even better than they were before.

The sprint went very well and every story was finished successfully.

8.9.3 Sprint work division



As most of the refinement and optimization work done in this sprint (54 story points) consisted of programming, the programming theme is yet again dominating. A part of this refinement process was teaching Luis more utterances and synonyms for various phrases and utterances. That is what the OTHER theme consisted mostly of. As this was the last sprint, it also contained a lot of report writing while wrapping the project up. See REPS theme.



Records are made to be broken. The old records of number of hours worked was in sprint #7 with 201 work hours. In this sprint, more than 350 hours were worked. It must, although, be taken into account that it was slightly longer than other sprints. The amount of work done by each member was more even than ever. Yet again, Smári decided to make up for his absence in sprints #4 and #5.

8.9.4 Sprint retrospective

This sprint looked quite intimidating in the beginning as it contained 62 story points, opposed to the more common amount of somewhere near 30 story points. It had been expected before the sprint started that its workload and pressure would be quite more than usual. It certainly was more difficult than previous sprints and in many cases the workdays were over 10 hours long.

Despite the immense workload, the atmosphere and productivity was good. Everyone was happy that this great journey was about to come to an end. Everyone made their best effort to make the product of this project, Talbankinn, as flawless as possible. It turned out to work out very nicely.

9 Summary and results

This report has covered all the work that went into designing and creating Talbankinn, an Internet bank service that you can talk to,an idea pitched by Landsbankinn. It covered tasks which were described with stories, risk analysis, various designs, testing and research.

The finished tasks resulted in a bank service with a speech interface where it is possible to transfer money between own accounts and known recipients, get an overview of owned accounts and of bills, pay bills, get exchange rates and convert one currency to another.

There was a lot of uncertainty and risks involved in implementing Talbankinn. The most noteworthy risk that came true involved not being able to use smart speakers, such as Google Home and Amazon Alexa. Howerer, a way to create Talbankinn was discovered which involved a design that used a mobile app to communicate to an API that communicates with multiple services and a mock data API which communicates with a mock database.

The most noteworthy tests performed were the user tests which were conducted in order to gather information on how users would phrase their requests to Talbankinn. A lot of good input was gathered from those tests and used when designing the dialog flow.

Extensive researches were conducted to determine which text-to-speech, language comprehension and speech-to-text services would be used for Talbankinn. The speech-to-text service that was chosen is Google Cloud’s speech-to-text because it has a lot of good documentation and converts spoken numbers to digits. Microsoft LUIS was chosen for language comprehension, because it has the best accuracy at comprehending the entities of the text it receives. AWS Polly was chosen for text-to-speech, because it had the one of the most natural speaking voices and the easiest to set up.

Research on the best services for text-to-speech, language comprehension and speech-to-text services were not the only ones conducted. Other researches include smart speaker plugin research which determined that it was not possible to use smart speakers with Talbankinn. The differences between Expo CLI, and React Native CLI which was not intended to be a research at first but became crucial later on. Authentication methods research was a requirement made by Landsbankinn for future development and were thoroughly researched.

10 Future vision

In order for Talbankinn to become fully functional for customers, security is the most important feature to improve in the current version. As stated in chapter 7.6: Authentication methods, various methods are available to implement the security. The most favorable methods for this type of project are those that do not require touch. Although the most important requirements within the scope of the project have been implemented, more has to be done in order for visually or touch impaired users to use the service in the manner intended. The mobile app is opened like most common apps but one of the requirements left hanging was to implement a wake up call. The user would wake the app by using a keyphrase e.g. “Okei Talbankinn”, and the app would be invoked.

The system also needs to be quicker when it comes to the round trip time of requests, the API must also be more agile in its understanding of requests, and the app must be built in a way that touch is rarely required. The reason why touchless communications and wake up calls are important is because of the likely user groups. People that are touch and/or visually impaired will need the feature, that is why it is included in this future vision.

11 Appendix 1: User Test: Wording - Kit

11.1 Step 1: A friendly Welcome

Takk fyrir að taka þátt í prófuninni. Við erum að vinna í lokaverkefni fyrir Bachelor gráðu í tölvunarfræði við Háskólann í Reykjavík. Verkefnið heitir Talbankinn og snýst annarsvegar um hvernig staða íslenskrar máltækni er í dag og hins vegar að búa til heimabankaþjónustu sem er hægt að tala við og talar til baka.

Markmiðið með prófuninni í dag er að greina þá mismunandi vegu sem fólk mun orða fyrirspurnir sínar til Talbankans. Til dæmis hvernig fólk biður um að fá reikningsyfirlit. Við erum með **15** fyrirspurnir sem við viljum að þú framkvæmir, þær eru flestar mjög stuttar en þú færð eins langan tíma og þú villt til þess að framkvæma hverja og eina. Við munum taka hljóðupptöku á meðan þú framkvæmir fyrirspurnirnar, ef það er í lagi þín vegna. Þessar upptökur verða síðan greindar og notaðar til þess að búa til kerfi sem er auðvelt og eðlilegt að tala við.

11.2 Step 2: The Context Questions

Áður en við byrjum vil ég spyrja þig nokkrar spurningar um sjálfa(n) þig.

1. Hversu gamall/gömul ert þú?
2. Hversu oft í mánuði notar þú heimabankann þinn?
3. Notar þú heimabanka app bankans þíns?
4. Hversu góð myndir þú segja að tölvuþekkingin þín sé á skalanum 1 til 10?
5. Hefur þú einhverja reynslu af tal viðmótum og tækjum á borð við Google Home og Amazon Alexa og ef svo er hvað notar þú þessi tæki í að gera?

11.3 Step 3: Introduce the prototype

Varan sem við erum að þróa fyrir verkefnið er ekki tilbúin til notendaprófana svo að þú þarft þess í stað að ímynda þér að þú værir að tala við Talbankann. Í framhaldi þessara prófana verður Talbankinn vonandi tilbúinn til notendaprófana innan viku. Sumar af fyrirspurnunum krefjast uppgjöf á reikningsnöfnum eða númerum en þá mátt þú skálda þær upplýsingar.

Ég vil minna þig á að tilgangurinn með prófuninni er að safna upplýsingum um mismunandi orðun til þess að búa til kerfi sem er auðvelt og náttúrulegt að tala við. Það er því engin rétt eða röng leið til þess að orða fyrirspurnirnar. Þú orðar hana eins og þér finnst vera eðlilegast.

11.4 Step 4: The tasks

Til þess að gera verkefnin skiljanlegri og til þess að hafa sem minnst áhrif á hvernig þú orðar fyrirspurnirnar þá ætla ég að láta þig fá lista með verkefnunum þar sem hverju verkefni er gefin atburðarrás sem leiðir til þess að þú þurfir að framkvæma fyrirspurnina. Áður en þú framkvæmir hverja fyrirspurn viljum við að þú lesir einu sinni yfir alla atburðarásina. Þegar þú ert búin/búinn að því viljum við að þú lesir upphátt númerið á verkefninu og segir okkur fyrirspurnina sem tengist verkefninu hátt og skýrt á þann veg sem þér finnst eðlilegast að segja hana.

Moderator version

Task number	Story number	Description	Scenario
1	**14**	As a user I want to know the balance of my accounts	Það eru bara nokkrir mánuðir í það að þú farir í sumarfrí til Spánar og villt vita hvort þú eigir ekki örugglega efni á því. Athugaðu hversu mikinn pening þú átt á reikningunum þínum.
2	**10**	As a user I want to know exchange rates	Þú ert lent/lentur á flugvellinum á Spáni og það er mjög heitt úti og þú ert orðin/orðinn mjög þyrst/þyrstur. Þú ferð í næstu sjoppu og sérð að þar er hægt að kaupa vatnsflösku á eina Evru. Athugaðu hversu mikið vatnsflaskan kostar í íslenskum krónum.
3	**15**	As a user I want to know the status of my credit card (funds available + balance)	Þú ert búinn að vera önnum kafin/kafinn að versla jólagjafir fyrir alla fjölskylduna þína og ert núna alveg að deyja úr hungri. Þú sérð fínann veitingastað en veist ekki hvort þú sért með næga innistæðu á kreditkortinu til að fá þér þar að borða. Athugaðu stöðuna á kortinu þínu.
4	**25**	As a user I want to check my account statements to monitor my finance	Þú ert búin/búinn að vera að safna fyrir nýjum bíl núna í nokkur ár. Þú fannst flotta Mercedes bifreið auglýsta í blaðinu en ert ekki alveg viss um að þú eigir efni á henni. Athugaðu hvort þú eigir nóg af pening.
5	**9**	As a user I want to get overview of unpaid bills	Það eru mánaðarmót og þú villt athuga hvort þú eigir einhverja óborgaða reikninga áður en þú ferð að skipuleggja hvað þú gerir við peningana þína í mánuðinum. Athugaðu hvort þú eigir einhverja óborgaða reikninga.
6	**12**	As a user I want to get information about a specific unpaid bill	Þér hefur borist reikningur frá Krabbameinsfélagi Íslands. Þú ert forvitin um hvenær eindagi reikningsins er svo þú vitir hvenær þú þurfir að vera búin/búinn að borga hann. Biddu um nánari upplýsingar um reikninginn.
7	**22**	As a user I want get overview of my loans	Það er búið að vera hundleiðinlegt veður á Íslandi í allan vetur. Þú sérð freistandi tilboð í sólarlandaferðir til Kanaríeyjar en veist ekki hvort þú munir ná að safna fyrir því út af mánaðarlegum

			greiðslum þínum í bæði húsnæðis lánið þitt og námslánið þitt. Finndu út hvað þú skuldar mikið.
8	**27**	As a user I want to transfer money from my account to an unknown Icelandic account	Litli bróðir makans þíns keypti fyrir þig nammi þegar hann fór að kaupa helgar nammi fyrir sjálfann sig. Hann lét þig fá nótuna fyrir kaupunum og er búinn að senda þér reiknings upplýsingar og kennitölu. Borgaðu honum til baka.
9	**21**	As a user I want to create a new known recipient	Aðra hvora helgi hittir þú vini þína og spilar spil. Vinur þinn Ásgeir kaupir alltaf matinn fyrir hittinginn og af því að þú ert sífellt að leggja inn á hann fyrir matnum vilt þú gera hann að þekktum viðtakanda. Bættu Ásgeiri við þekkta viðtakanda.
10	**8**	As a user I want to transfer money from my account to a known recipient	Vinur þinn Jói var að borga bíómiðann þinn af því að þú gleymdir kortinu þínu heima, miðinn kostaði eitt þúsund og fimmhundruð krónur. Þú ert búin/búinn að skrá Jóa sem þekktan viðtakanda í Talbankanum. Borgaðu Jóa til baka.
11	**26**	As a user I want to get details on a specific transaction in my account statements	Þér berst skyndilega þúsund krónur á sparireikninginn þinn frá aðila sem þú kannast ekki við. Biddu um upplýsingar um þessa millifærslu.
12	**18**	As a user I want to transfer money between two of my accounts	Það eru komin mánaðarmót og þú varst að fá útborgað á kortareikninginn/launareikninginn þinn. Þú ert að vinna í því að spara fyrir ferð til útlanda og hefur því búið til sparireikning sem heitir “sparireikningur 1”. Millifærðu tíu þúsund krónur af kortareikningnum/launareikningnum þínum á sparireikning 1.
13	**11**	As a user I want to know the share rates	Þú keyptir hlutabréf í sprotafyrirtækinu “Endalaus Hamingja” í seinasta mánuði og nú vilt þú vita hversu mikið hlutinn þinn er virði. Athugaðu hversu mikið hlutabréfin þín eru virði.
14	**19**	As a user I want to pay my bills	Það eru komin mánaðarmót og þú varst að fá útborgað. Nú er tilvalinn tími til þess að borga ógreidda reikninga. Borgaðu ógreidda reikninga.
15	**23**	As a user I want to get details about a specific loan	Þú byrjaðir í nýrri vinnu seinasta sumar og ert núna að fá útborgað mun meiri pening heldur en áður. Þú átt því loksins efni á því að kaupa þér íbúð en vantar hagstætt húsnæðislán. Þú heyrðir að óverðtryggt húsnæðislán sé sniðugt. Biddu um upplýsingar um lánið.

Participant version

Task number	Scenario
1	Það eru bara nokkrir mánuðir í það að þú farir í sumarfrí til Spánar og þú vilt vita hvort þú eigir ekki örugglega efni á því. Athugaðu hversu mikinn pening þú átt á reikningunum þínum.
2	Þú ert lent/lentur á flugvellinum á Spáni, það er mjög heitt úti og þú ert orðin/orðinn mjög þyrst/þyrstur. Þú ferð í næstu sjoppu og sérð að þar er hægt að kaupa vatnsflösku á eina Evru. Athugaðu hversu mikið vatnsflaskan kostar í íslenskum krónum.
3	Þú ert búinn að vera önnum kafinn að versla jólagjafir fyrir alla fjölskylduna þína og ert núna alveg að deyja úr hungri. Þú sérð fínan veitingastað en veist ekki hvort þú sért með nægа innistöðu á kreditkortinu til að fá þér þar að borða. Athugaðu stöðuna á kortinu þínu.
4	Þú ert búin/búinn að vera að safna fyrir nýjum bíl núna í nokkur ár. Þú fannst flotta Mercedes bifreið auglýsta í blaðinu en ert ekki alveg viss um að þú eigir efni á henni. Athugaðu hvort þú eigir nóg af pening.
5	Það eru mánaðarmót og þú vilt athuga hvort þú eigir einhverja óborgaða reikninga áður en þú ferð að skipuleggja hvað þú gerir við peningana þína í mánuðinum. Athugaðu hvort þú eigir einhverja óborgaða reikninga.
6	Þér hefur borist reikningur frá Krabbameinsfélagi Íslands. Þú ert forvitin/forvitinn um hvenær eindagi reikningsins er svo þú vitir hvenær þú þurfir að vera búin/búinn að borga hann. Biddu um nánari upplýsingar um reikninginn.
7	Það er búið að vera hundleiðinlegt veður á Íslandi í allan vetur. Þú sérð freistandi tilboð í sólarlandaferðir til Kanaríeyjar en veist ekki hvort þú munir ná að safna fyrir því út af mánaðarlegum greiðslum þínum í bæði húsnæðis lánið þitt og námslánið þitt. Finndu hvað þú skuldar mikið.
8	Litli bróðir makans þíns keypti fyrir þig nammi þegar hann fór að kaupa helgar nammi fyrir sjálfann sig. Hann lét þig fá nótuna fyrir kaupunum og er búinn að senda þér reiknings upplýsingar og kennitölu. Borgaðu honum til baka.
9	Aðra hvora helgi hittir þú vini þína og spilar spil. Vinur þinn Ásgeir kaupir alltaf matinn fyrir hittinginn og af því að þú er sífellt að leggja inná hann fyrir matnum vilt þú gera hann að þekktum viðtakanda. Bættu Ásgeiri við þekkta viðtakanda.
10	Vinur þinn Jói var að borga bíómiðann þinn af því að þú gleymdir kortinu þínu heima, miðinn kostaði eitt þúsund og fimmhundruð krónur. Þú ert búin/búinn að skrá Jóa sem þekktan viðtakanda í Talbankanum. Borgaðu Jóa til baka.
11	Þér berst skyndilega þúsund krónur á sparireikninginn þinn frá aðila sem þú kannast ekki við. Biddu um upplýsingar um þessa millifærslu.
12	Það eru komin mánaðarmót og þú varst að fá útborgað á kortareikninginn/launareikninginn þinn. Þú ert að vinna í því að spara fyrir ferð til útlanda og hefur því búið til sparireikning sem heitir

	“sparireikningur 1”. Millifærðu tíu þúsund krónur af kortareikningnum/launareikningnum þínum á sparireikning 1.
13	Þú keyptir hlutabréf í sprotafyrirtækinu “Endalaus Hamingja” í seinasta mánuði og nú vilt þú vita hversu mikið hlutinn þinn er virði. Athugaðu hversu mikið hlutabréfin þín eru virði.
14	Það eru komin mánaðarmót og þú varst að fá útborgað. Nú er tilvalinn tími til þess að borga ógreidda reikninga. Borgaðu ógreidda reikninga.
15	Þú byrjaðir í nýrri vinnu seinasta sumar og ert núna að fá útborgað mun meiri pening heldur en áður. Þú átt því loksins efni á því að kaupa þér íbúð en vantar hagstætt húsnæðislán. Þú heyrðir að óverðtryggt húsnæðislán sé sniðugt. Biddu um upplýsingar um lánið.

11.5 Step 5: The debriefing

1. Myndir þú treysta því að stunda bankaviðskipti með talþjónustu eins og Talbankanum og ef ekki, af hverju?
2. Gætir þú ímyndað þér að nota þjónustu á borð við Talbankann og hvar myndir þú helst vilja nota hana og með hvaða tæki?
3. Hvað myndir þú helst vilja nýta þjónustuna í að gera?

Takk fyrir að taka þátt.

12 References

[1] “Classroom Reynsla Af Notendamiðuðum Aðferðum (R7-17: UCD Methods In Scrum).” [Online]. Available: https://echo360.org.uk/lesson/d5c461e3-976a-402a-a08d-df28a90e2375/classroom#sortDirection=desc. [Accessed: 13-May-2019].

[2] M. Cohn, “Planning Poker: An Agile Estimating and Planning Technique,” *Mountain Goat Software*. [Online]. Available: https://www.mountaingoatsoftware.com/agile/planning-poker. [Accessed: 13-May-2019].

[3] matvelloso, “Principles of bot design - Bot Service.” [Online]. Available: https://docs.microsoft.com/en-us/azure/bot-service/bot-service-design-principles. [Accessed: 06-May-2019].

[4] “What affects the way we speak? - ppt download.” [Online]. Available: https://slideplayer.com/slide/4248797/. [Accessed: 11-Apr-2019].

[5] “Google Home Usability Study,” *Tressa Coultard*. [Online]. Available: https://www.tressacoultard.com/google-home-usability-study. [Accessed: 11-Apr-2019].

[6] “Watson Assistant | IBM Cloud.” [Online]. Available: https://www.ibm.com/cloud/watson-assistant/. [Accessed: 14-May-2019].

[7] “Language Understanding (LUIS) - Tutorials, Quickstarts, API Reference | Microsoft Docs.” [Online]. Available: https://docs.microsoft.com/en-us/azure/cognitive-services/LUIS/. [Accessed: 14-May-2019].

[8] “Would you rather your virtual assistant be female, male, or gender neutral? (Poll of the Week) - Android Authority.” [Online]. Available: https://www.androidauthority.com/virtual-assistant-voice-poll-886294/. [Accessed: 07-Feb-2019].

[9] “Actions on Google | Actions on Google | Google Developers.” [Online]. Available: https://developers.google.com/actions/. [Accessed: 24-Feb-2019].

[10] “Overview | Actions on Google | Google Developers.” [Online]. Available: https://developers.google.com/actions/extending-the-assistant. [Accessed: 24-Feb-2019].

[11] “Setup and Developing | Actions on Google | Google Developers.” [Online]. Available: https://developers.google.com/actions/console/setup-and-developing. [Accessed: 24-Feb-2019].

[12] “Explicit Invocation | Actions on Google | Google Developers.” [Online]. Available: https://developers.google.com/actions/discovery/explicit. [Accessed: 24-Feb-2019].

[13] “Cloud Speech-to-Text - Speech Recognition | Cloud Speech-to-Text API | Google Cloud.” [Online]. Available: https://cloud.google.com/speech-to-text/. [Accessed: 24-Feb-2019].

[14] “Dialogflow - Wikipedia.” [Online]. Available: https://en.wikipedia.org/wiki/Dialogflow. [Accessed: 24-Feb-2019].

[15] “Actions on Google Conversation HTTP/JSON Webhook API | Actions on Google | Google Developers.” [Online]. Available: https://developers.google.com/actions/build/json/. [Accessed: 24-Feb-2019].

[16] “Cloud Text-to-Speech - Speech Synthesis | Cloud Text-to-Speech API | Google Cloud.” [Online]. Available: https://cloud.google.com/text-to-speech/. [Accessed: 24-Feb-2019].

[17] “SSML | Actions on Google | Google Developers.” [Online]. Available:

https://developers.google.com/actions/reference/ssml. [Accessed: 24-Feb-2019].
[18] “Account linking | Actions on Google | Google Developers.” [Online]. Available: https://developers.google.com/actions/identity/. [Accessed: 25-Feb-2019].
[19] “Alexa Skills Kit - Build for Voice with Amazon.” [Online]. Available: https://developer.amazon.com/alexa-skills-kit. [Accessed: 25-Feb-2019].
[20] “Understanding Amazon’s Alexa and Building Alexa Skill.” [Online]. Available: https://medium.com/@ashish_fagna/understanding-amazons-alexa-and-alexa-skill-6749785683b0. [Accessed: 04-Mar-2019].
[21] “Amazon Polly Voices in Alexa Skills Now Generally Available : Alexa Blogs.” [Online]. Available: https://developer.amazon.com/blogs/alexa/post/baee53c1-5b03-4580-b57a-ee9510413354/amazon-polly-voices-in-alexa-skills-now-generally-available. [Accessed: 04-Mar-2019].
[22] “Audio - Expo Documentation.” [Online]. Available: https://docs.expo.io/versions/latest/sdk/audio/. [Accessed: 15-May-2019].
[23] “Introduction to audio encoding | Cloud Speech-to-Text,” *Google Cloud*. [Online]. Available: https://cloud.google.com/speech-to-text/docs/encoding. [Accessed: 15-May-2019].
[24] D. Ward, “Expo vs React Native CLI: A Guide to Bootstrapping New React Native Apps,” *Level Up Your Coding*, 25-Sep-2018. [Online]. Available: https://levelup.gitconnected.com/expo-vs-react-native-cli-a-guide-to-bootstrapping-new-react-native-apps-6f0fcafee58f. [Accessed: 15-May-2019].
[25] A. Biometrics, “Voice Authentication Technology - Aware Biometrics Software,” *Aware*. .
[26] “Biometrics - Voice Verification.” [Online]. Available: https://www.globalsecurity.org/security/systems/biometrics-voice.htm. [Accessed: 01-May-2019].
[27] “Link your voice to your speaker or Smart Display with Voice Match - Android - Google Assistant Help.” [Online]. Available: https://support.google.com/assistant/answer/9071681?hl=en. [Accessed: 01-May-2019].
[28] “Amazon Transcribe – Automatic Speech Recognition - AWS,” *Amazon Web Services, Inc.* [Online]. Available: https://aws.amazon.com/transcribe/. [Accessed: 01-May-2019].
[29] “Speaker Recognition API | Microsoft Azure.” [Online]. Available: https://azure.microsoft.com/en-us/services/cognitive-services/speaker-recognition/. [Accessed: 01-May-2019].
[30] Shams, “List of All Fingerprint Scanner Enabled Smartphones,” *WebCusp.Com - Make living with Blogging and no Coding web design*, 24-Apr-2018. .
[31] “Landsbankaappið - Landsbankinn,” *Landsbankinn.is*. [Online]. Available: http://www.landsbankinn.is/einstaklingar/rafraenar-lausnir/landsbankaappid/?gclid=Cj0KCQjwh6XmBRDRARIsAKNInDHpJoufJTLdFnsNZYjShMkYQLBzNlhXrrW6-p1XGnBGXVFpsNn19QQaAhWsEALw_wcB. [Accessed: 01-May-2019].
[32] “react-native-biometrics,” *npm*. [Online]. Available: https://www.npmjs.com/package/react-native-biometrics. [Accessed: 02-May-2019].
[33] A. Biometrics, “Facial Recognition Technology - Aware Biometrics Software,” *Aware*. .
[34] “Vision AI | Derive Image Insights via ML,” *Google Cloud*. [Online]. Available: https://cloud.google.com/vision/. [Accessed: 01-May-2019].
[35] “Image Processing with the Computer Vision API | Microsoft Azure.” [Online]. Available: https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision/. [Accessed: 01-May-2019].

[36] “OAuth 2.0 — OAuth.” [Online]. Available: https://oauth.net/2/. [Accessed: 02-May-2019].
[37] “oauth tutorial - Google Search.” [Online]. Available: https://www.google.com/search?q=oauth+tutorial&oq=oauth+tut&aqs=chrome.0.0j69i57j69i60j0l3.10397j0j7&sourceid=chrome&ie=UTF-8. [Accessed: 02-May-2019].
[38] “Using OAuth 2.0 to Access Google APIs | Google Identity Platform,” *Google Developers*. [Online]. Available: https://developers.google.com/identity/protocols/OAuth2. [Accessed: 02-May-2019].
[39] “Rafræn skilríki á farsíma,” *Skilriki.is*. [Online]. Available: https://www.skilriki.is/notendur/skilriki-i-farsima/. [Accessed: 02-May-2019].
[40] “Netbanki einstaklinga.” [Online]. Available: https://netbanki.landsbankinn.is/Shell/Login.aspx#elecid. [Accessed: 02-May-2019].
[41] “Futurae: Strong Authentication (2FA) and App Security.” [Online]. Available: https://futurae.com/product/iotauth/. [Accessed: 02-May-2019].