



**university of  
 groningen**

**university college  
 groningen**

# **Time to Learn**

**A new way of learning with the use of a smartwatch**

## **Bachelor Thesis**

**Rick Nienhuis & Niels Haan**

**Faculty of Mathematics and Natural Sciences  
University of Groningen**

**31. Augustus 2016**

**Supervisor:**

**Dr. Mircea Lungu**

# Abstract

In this project we are trying to answer the following question: *How ~~do users~~→can we use a smartwatch application to accelerate the memorization process of somebody who is learning the vocabulary of a second language?* This question is of interest ~~useful for people who want to use a smartwatch for educational purposes. The application is built~~ for busy people who ~~can read and understand the language, but~~ want to improve their vocabulary in the most efficient way ML ~~►because your app could work also for beginners... only the integration with zeeguu makes it appropriate for advanced learners◄~~.

In order to give these people the best possible experience, we first built such a smartwatch application, and then performed a study ~~a research was conducted~~ to investigate how ~~the~~→actual users ~~currently use~~→benefit from the app→such an application.

The main goal of the application – named Time to Learn – is to let users easily invest more time in learning, ~~this is done~~ by ~~filling~~→taking advantage of many small time units during the day. This is called micro learning [2]. All these little time units together are a reasonable amount of time spent on studying.

The results of the research show that the users invest a lot more time into studying words while using the smartwatch application. There are however still some features which could be added to give the user an even better experience.

~~In the past there have been very similar approaches, the main difference was the device choice which was a smartphone instead of a smartwatch. The usage of the smartwatch is an improvement, because it is more efficient and easier than a smartphone [10].~~ ML ~~►Not here but in the related work, if it's not there already!◄~~

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Related Work</b>	<b>4</b>
2.1	Zeeguu . . . . .	4
2.1.1	Chrome plugin . . . . .	5
2.1.2	Android applications . . . . .	5
2.1.3	iOS application . . . . .	5
2.2	Comparable research projects . . . . .	5
2.3	The best way to study words . . . . .	6
<b>3</b>	<b>The Design</b>	<b>9</b>
3.1	Login screen . . . . .	9
3.2	Main screen . . . . .	11
3.2.1	Time . . . . .	12
3.2.2	Words . . . . .	13
3.2.3	Background . . . . .	14
3.2.4	Buttons . . . . .	16
3.2.5	Information components . . . . .	19
3.2.6	Option lists . . . . .	20
3.2.7	Profile . . . . .	23
3.2.8	Effects . . . . .	25
<b>4</b>	<b>The Implementation</b>	<b>26</b>
4.1	Flash card algorithm . . . . .	26
4.2	Getting new words from the server . . . . .	28
4.3	Usage Tracking . . . . .	32
4.3.1	Events . . . . .	32
4.3.2	Touch tracking . . . . .	33
4.4	Changing the watchface and sending and saving the events and taps . . . . .	33
4.5	Drawing the user interface . . . . .	34
4.6	Testing . . . . .	35

<b>5</b>	<b>Usage Study</b>	<b>36</b>
5.1	Usage results . . . . .	37
5.1.1	Overall usage. . . . .	37
5.1.2	Pie chart diagram presenting the learning sessions in seconds . .	39
5.1.3	Bar chart visualizing ratio between ‘right’ and ‘wrong’ . . . . .	40
5.1.4	Table with median reaction time and daily usage . . . . .	41
5.2	Questionnaires results . . . . .	43
5.2.1	General information about the user . . . . .	43
5.2.2	Smartwatch app usage . . . . .	44
5.3	Threats to Validity . . . . .	44
<b>6</b>	<b>Conclusion and Future Work</b>	<b>45</b>
6.1	Conclusion . . . . .	45
6.2	Future work . . . . .	46
<b>A</b>	<b>Test Cases</b>	<b>49</b>
<b>B</b>	<b>Questionnaire</b>	<b>56</b>
B.1	General information . . . . .	56
B.2	Questions after using the smartwatch app . . . . .	57
<b>C</b>	<b>Questionnaire answers</b>	<b>59</b>
C.1	General information . . . . .	59
C.2	Questions after using the smartwatch app . . . . .	61
<b>D</b>	<b>Script to Obtain Different Session Lengths</b>	<b>65</b>

# 1

## Introduction

The way people learn has changed over the years. The development of information and communication technology has led to new ways of learning. People can now find fast and really specific information on the web and learn new things using ICT tools, like laptops, iPads and other electronic devices. The traditional book is being less and less used. We can see this in high schools where students learn their languages on the laptop with a rehearsing program. These rehearsing programs should accelerate the memorization process, so the student can work more efficiently and therefore has more time for other stuff and can work more efficiently.

**ML** ► *This next paragraph was too redundant. Several things have to be deleted and it's going to be better.* ◀ But even with rehearsing programs learning a second language can be a hard and time consuming thing to do. Since learning a second language is so time consuming, people often ~~don't have enough time for it~~ → give it up completely. ~~They are~~ → This is true especially with people who are busy with other important things like work ~~and don't have the time to really learn every day~~. Although for these type of people it seems like they have no time to learn, in fact the time they lose for waiting for the bus, taking the elevator and walking to the car could be used ~~more efficiently~~ → for learning. Using these little time units is called micro learning [2]. In this thesis we are going to try to find out how we can fill these little time spots in the best way using a smartwatch.

The application for the smartwatch will be a watch face, this is the screen that the user

will see when the screen is on. We also had the option to choose for a separate app using push notifications, however push notifications in general are bad. They have a negative effect on task performance [4] and they could be perceived as annoying. ML ► *I like that you found a recent paper :)*◀ The main function of the watch face is to show the time, however it also offers the user a word for which the translation can be revealed by tapping on the screen. The user will be stimulated to think of the right translation before seeing it, after revealing it the user can give the watch feedback by pressing the green ('I had it right in my head') or red button ('I had it wrong in my head'). [to speed up the memorization process](#) The algorithm used for displaying words is based on the way in which people learn with flashcards.→: This is wrong words will be repeated earlier then words which the user had right in his mind.

The app for the smartwatch is part of Zeeguu ecosystem, which is a research project designed to speed & fun up vocabulary learning in a new language based on three fundamental principles: only read material the learner likes, have words everywhere with you and practice with personalized exercises. This basically means the users read articles they like, tap on the words they don't know, get "in text" translations so they can continue with their reading, and have the chance to practice them later. The words will be saved in their accounts and can be accessed at any time.

In this thesis, we introduce an app for the smartwatch called Time to Learn , so users have an complementary tool for learning their words in the Zeeguu account and thus increase the memorization process of learning a second new language. In this thesis we will research what is the best way to build such an app, therefore a user study was designed to answer the following research question:

How do users use a smartwatch application to accelerate the memorization process of somebody who is learning the vocabulary of a second language?

To answer this question, we ask and answer the following subquestions:

- How long ~~does a~~→[do](#) sessions on a smartwatch take (i.e., general usage)?
- How long ~~does a~~→[do](#) learning sessions take on a smartwatch (i.e., only using Time to Learn app)?
- How long do users use the app daily ~~on average~~? ML ► *average is again not the right way to go. Never use average when data is not normally distributed!*◀

### Structure of this Thesis

The structure of the remainder of this document is the following:

In Chapter 2 (Related Work) the general research field is described and how others tried to solve this or similar problems.

In Chapter 3 (The Design) the user interface is described, explaining the choices which have been made.

In Chapter 4 (The Implementation) the code structure, algorithms and implementation choices are explained.

In chapter 5 (Usage Study) explaining the tests and a summary of the results with various diagrams.

In chapter 6 (Evaluation) a discussion about why this research was successful.

In chapter 7 (Conclusion and Future Work) conclusions about the usage results and describing possible future work.

# 2

## Related Work

The app is built on the Zeeguu platform. Therefore this section will start by discussing Zeeguu and other applications in the Zeeguu ecosystem.

The smartwatch is a relatively new gadget and therefore the number of research studies on the use of smartwatches in learning are inexistent to the extent of our knowledge. There are however comparable approaches that use smartphones for the same purpose, that is making learning available wherever the user is. ~~which~~ We will discuss ~~in~~ several such approaches in this chapter.

Learning new words during short periods of time – or micro-learning – is the main purpose of the app and thus it is important to ~~research~~ understand the ~~best way to learn words~~ state of the art in the domain of learning. ~~Especially when the user will only use, learning, since the smartwatch app is a complementary tool. We~~ This is why we close this chapter by discussing some techniques that were found during projects that investigated how to make learning new words more efficient.

### 2.1 Zeeguu

The smartwatch app can currently only be used when the user has a profile on Zeeguu. The Zeeguu platform is an ecosystem that is built around an open API[6] which aims to



improve learning words of a foreign language. We decided to do this in order to avoid having to implement an extra system which would allow users to upload the words they want to study.

Multiple projects are part of the ecosystem: a Chrome plugin, several Android applications, an iOS application and after this project a smartwatch app can be added to the list.

### 2.1.1 Chrome plugin

The Chrome plugin can be installed to quickly add new words to the users profile.[6] With the plugin the user can click words the user wants to learn. By clicking the translation of that word will appear above it and by clicking the translation the word is saved to the profile of the user. When a translation does not appear to be correct, the user can change the translation.

### 2.1.2 Android applications

Android applications were added to the ecosystem to give the users more possibilities to use Zeeguu since Android is the most popular operating system on smartphones.[3] These applications will lead to an increase of time that the user spends on learning words. New exercises designed to improve the learning process were implemented for the applications.

### 2.1.3 iOS application

For iOS devices an application was made in which users could read news articles from different websites.[9] The users would read articles in the language they wanted to learn or to improve. If a word is unknown, the user could touch the word and the app would show the translation. This word could then be saved on their Zeeguu account.

## 2.2 Comparable research projects

D. Dearman and K.N. Truong presented a the “Vocabulary Wallpaper” ~~which is~~ a language tool designed to provide quick access to vocabulary to support short session of learning.[2] This app for the smartphone shows some similarity compared to the app

for the smartwatch. The app consists of a vocabulary wallpaper that can be accessed from the lock screen. This idea is similar to the idea for the smartwatch app. The conclusion of this approach showed that in four short sessions of approximately 4 hours, the participants were able to increase the number of second language vocabulary that they could recognize and recall when using the vocabulary wallpaper.

The use of a smartwatch can help users saving time by making processes efficient and easier. D. Pradhan and N. Sujatmiko described why the use of a smartwatch is more efficient and easier compared to a smartphone.[10] ~~An example mentioned~~ → [A strong argument](#) is the number of steps that are required to use a smartphone. They described the following steps:

1. Unlocking the phone. While in pocket, a phone is typically screen-locked to avoid unwanted use. Unlocking requires several taps or slides.
2. Finding app. A person typically has number of apps that is more than one phone screen can hold and therefore apps icons are placed in multiple home-screens. Hence finding the apps required another one or two taps or slides.
3. Running the app itself. This may vary based on how easy the app's user interface is, but it definitely needs more than just two taps.

These steps make it inconvenient to use a smartphone in situations like: driving, cycling or walking and it could even be unsafe. A smartwatch would only require an arm twist to activate the screen and some swipes (depending on the position of the app) and the watch is ready to go.

~~Another disadvantage could be the fact that the learner needs to carry the smartphone with him. This will create a chance of forgetting the place where the smartphone is placed. This could lead to losing time when the owner tries to find his smartphone, which is not possible when a device is worn on the wrist.~~ **ML** ► *this argument is too weak. i sometimes forget where i put my smartwatch too... so let's just remove it.*◀

## 2.3 The best way to study words

There are a lot of research projects about finding the most efficient way for learning new words. Due to the large number of studies, it was not possible to implement all the conclusions of each research. Those who did not make the final version of the app could be implemented in the future.

S. Thornbury described what is needed to learn a new word.[11] According to S. Thornbury knowing a word involves knowing its form and its meaning. Knowing its form

means that you know whether it is a noun, a verb or a preposition etc. and how to spell it. When focussing on its meaning you not only need to know what the word means, but also what register it is used in, what category of words it belongs to: for instance: fruit, animals, plants, transport or even to groups of abstract words. And you need to know how it is used in a sentence or what chunks or collocations it can be used in. In the app the user can see the translation of the word and the sentence in which the word appeared. However the user is not able to see whether it is a noun, a verb etc.

P. Nation came with guidelines how one should learn vocabulary in another language.[7]

1. Learn new words by using flashcards. The word is written on one side of the card and the translation on the other side. When both the word and the translation are not seen simultaneously, the connection between them will be strengthened.
2. The words should be learned in groups of fifteen to twenty words at the same time. When the words appeared to be difficult, the groups should be made smaller.
3. The learning sessions should be alternated with pauses of about one hour. In addition the sessions should be repeated after days to consolidate the words in the long-term memory. When a word appears to be learned, the time between testing this word should be gradually extended.
4. The memory process can be accelerated by saying the translations aloud.
5. A translation that is hard to remember can be more easily retrieved when it has many associations. Seeing the word in a sentence can help this process.
6. During a learning session words with similar spelling or meaning should be avoided.
7. The order in which the words are learned should change from time to time to prevent knowing the order instead of the words.
8. The context of the word helps with learning the translation.

**The app** → [Time to Learn](#) uses a flashcard method which means that words that are not recognized are more rapidly repeated than words that are recognized. The words are learned in small groups of five. Since the words are placed back in the list of words when a word is learned or not, the order of words changes every time and therefore serial learning is not possible. The user is also able to see the context of the word.

J.P. Anderson and A.M. Jordan measured recall immediately after learning, after one week, after three weeks and after eight weeks.[1] The percentages of material retained were 66%, 48%, 39% and 37% respectively. These numbers indicates that new words should be repeated very soon after the words appeared for the first time. This is to prevent the word is forgotten before the word appears again.

The app registers the number of times the user knows a word. This number is used to decide at what position the word should be placed when the user indicates the word is still known. The higher the number the further the word is placed in the list and therefore the longer it will take to see that word again.

I.S.P. Nation investigated the effect of a delay between the presentation of a word and its meaning.[8] According to the researchers the learners have an opportunity to make an effort to guess the meaning. This extra effort will result in faster and longer retained learning. The guessing will only succeed if the learner could recognize the translation in the foreign word. When the learner sees a word for the first time, a simultaneous presentation of a word and its meaning is the best. After the first encounter the best way to show the word and its meaning is with a delayed presentation.

In order to understand the full meaning of a word, the learner should be able to read the context of the word.

The app shows the word first without the translation which gives the user the opportunity to try to remember the translation or to do an educated guess. Then by tapping the word, the translation will appear.

# 3

## The Design

Making an app for a smartwatch involves quite some thinking about the design. Compared to a smartphone the available display is circular and much smaller and thus designing and redesigning the layout with the time, the words, the buttons and some information components took some time. The app is a watchface and this comes with some restrictions when it comes to the different inputs the watch can receive from the user. An app that is set as a watchface can only detect touch, swipe up and swipe down. The app begins with a login screen and after entering a valid code the main screen is shown. All the decisions related to the designs of those two screens are described below.

### 3.1 Login screen

Before the user can use the app a code is needed that is used for identifying the user with the server. To get this code the app starts with a login screen. The login screen first consisted of ten buttons with the ten numbers, a 'clear' button and a 'okay' button. On the top of the screen there are four small rectangles for displaying the pressed numbers (see figure 3.1).



Figure 3.1: The first version of the login screen with the small numbers.

After some testing and discussions it was decided that the feedback of the pressed numbers on the top of the screen were too small and thus some rethinking was needed to come up with a new design. An idea was to make four rotating disks that was inspired from securing a travel suitcases (see figure 3.2).



Figure 3.2: The design of this lock was used as inspiration.

The advantage is that no buttons for the numbers were needed so the numbers could be placed in the middle. In the middle they can be displayed in a larger font since more space is available due to the circular shape. However, the detection of swipe events were insufficient for rotating the disks and thus the idea for rotating disks was changed to improve the functionality. The positions of the numbers stayed the same, but instead of swiping an increase and decrease button was added above and below each number (see figure 3.3).



Figure 3.3: The second version of the login screen with the larger numbers.

To increase the safety of a user's account it was decided to use a code of eight digits and therefore a page indicator was placed below the decrease buttons to notify the user that the user should insert 4 digits two times. On the bottom of the screen a 'next' button is placed to go to the next page for inserting the second 4 digits of the code or to confirm that the eight digits were inserted.

## 3.2 Main screen

When a valid code is inserted, the main screen becomes visible. During the project, the main screen had had different layouts. ~~All the~~→Each category of decisions ~~are sorted in~~→is discussed in a separate ~~categories~~→section: time, words, background, buttons, information components, option lists, profile and effects.

### 3.2.1 Time

An important part of the app is displaying the time, since the app will be the first thing the user will see when the watch screen turns on. This importance was not clear at the beginning of the project, therefore the first design contained the time in a small black font. The size of the font was based on the hill that was displayed on the background. With the chosen font the time fitted in the hill in the middle to increase the readability (see figure 3.4).



Figure 3.4: The first version of the main screen ~~with the wrong ratio~~ → [put too little emphasis on time](#).

After the first discussion it was decided that the time should be displayed on 50% of the screen since the app would be use for checking the time and for learning words (see figure 3.7). The time however was not readable enough and this was solved by changing the font color to white and by changing the background (see figure 3.5). This new design worked well and therefore this became the final design for the time.





Figure 3.5: The final design of the time.

### 3.2.2 Words

The other important part of the app is showing the wordpairs, the word and the translation, the user wants to learn. Displaying wordpairs on a small circular screen was quite challenging which resulted in different designs for the watchface (see figure 3.6).



Figure 3.6: The different layouts designed for optimal usage of the circular screen.

The sixth design was chosen as the most convincing design. In the first version the wordpairs were placed in the middle of the screen (see figure 3.4) with a white font. Due to the circular shape there is more space available in the middle of the screen and

therefore the wordpairs can be in a larger font. The color white was chosen to have a maximum contrast with the background since the background mostly consists of dark colors. The words have a larger font than the translations to have a clear distinction between the two types and later on the smaller font was selected to make sure that there is enough space to display the translations. Since the translations are placed lower than the words, less space is available. When only 50% of the screen was available for the wordpairs after the first discussion, the wordpairs were placed just below the middle where the space lost is minimal (see figure 3.7).



Figure 3.7: The second version of the time display uses 50% of the screen for showing the time.

### 3.2.3 Background

The app consists of two visual parts, one for the time and one for the wordpairs. Both parts have different backgrounds. The background for the wordpairs was selected in the beginning of the project and did not change. The background consists of different dark grayish colors. With the white colored text the user should not have any problems reading the wordpairs.

At first the idea for the background of the time was that it should support the current time. Therefore four different backgrounds were designed for the morning, afternoon, evening and night (see figure 3.8).



Figure 3.8: The first designs for the background of the time.

The space for the time then changed to 50% of the screen and therefore the backgrounds were unusable because of the old dimensions of the images. Instead of changing the images to the new dimensions this opportunity was used to think again about the background of the time. The different images for the different moments of the day had to be saved on the watch. This storage issue was solved by using one image. The new background consisted of the sun, the moon and a transition between blue and black. This background rotated around its center which makes it possible to support the current time with the positions of the sun and the moon (see figure 3.9).



Figure 3.9: The rotating background image of the time.

Due to the rotation a sunset and sunrise could be seen on the watch. To emphasize the sunset and sunrise a landscape was added to make, for example, the sun appear from the horizon with a sunset. The used colors for the landscape were different from the colors in the rotating background and hence a new background with the sun and moon was designed to match the landscape (see figure 3.10). The landscape worked well with the

rotating background and therefore two other landscapes were designed from which the user can choose (see figure 3.11).



Figure 3.10: Version 5 of the main screen with a new background and a landscape.



Figure 3.11: The designs of the other two landscapes the user could choose from.

### 3.2.4 Buttons

As mentioned before swipe detection was insufficient so it was decided that the app should only use touch events. To navigate through the app with only touch events it seemed logical to use buttons. With the time on top of the screen and the wordpairs in the middle there was unused space in the bottom for the buttons. In the first design the user should be able to

- reveal the translation of the shown word, or
- to go to the next wordpair.

This resulted in two buttons, a red one with glasses on it and a green one with an arrow to the right (see figure 3.4). Glasses were chosen for revealing because they suggest looking something up and that we found intuitive for revealing the translation.

Since the app is displayed on a small screen, the buttons should not be too small and thus almost all the available space in the bottom was used for the buttons to ensure that there was enough space for the user to press a button. Different tests showed that the buttons could be made smaller. The design of the buttons was not really pleasing and needed some rethinking. The first modification was erasing the reveal button. Instead of the button the user could touch the word to reveal the translation. The 'next' button was widened so it covered the bottom of the screen (see figure 3.5). During development more options were implemented and therefore a new button was made next to the 'next' button. The 'settings' button gave access to these options (see figure 3.12). The curves on the buttons as can be seen in the first design were replaced with right angles to create a modern feeling.



Figure 3.12: Version 4 with the new 'settings' button.

The app was tested and more attention was paid to improve the order in which the wordpairs are shown. This process should become smarter by the use of a knowledge estimator that would estimate what the next word should be to maximize the learning process. This estimation could only be made when the server could receive feedback of every single word. It is therefore necessary that the user should give feedback after every word. The design of the app was adapted to this new idea by changing the 'next' button

to a 'reveal' button (see figure 3.10). When the user pressed this button the translation appeared and the 'reveal' button changed in three different buttons: a 'wrong' button, a 'menu' button and a 'right' button (see figure 3.13).



Figure 3.13: The mandatory feedback page displayed after revealing a word.

This new design forced the user to give feedback after every word. This feedback could be used to optimize the order of the words. The first design of the buttons were a cross and a checkmark to let the user know what to press when a word was not know or was known. Later on a book and a graduation cap were used for the same purpose but the feedback to the user would be less harsh with the new design (see figure 3.14).



Figure 3.14: The mandatory feedback page with the redesigned buttons

### 3.2.5 Information components

Besides the time the app also shows the date. The date was added later when the design of the time was completed. In the first design the date had his own small icon on top of the screen (see figure 3.5). This icon was partly covered by a Tizen icon for the swipe down menu. Therefore the icon for the date was moved downwards near the time (see figure 3.10). During a discussion it was mentioned that it would be nice to be able to see the temperature and the weather type on the watchface. After some research a free API<sup>1</sup> was found that could provide the app with this information. In the beginning of the implementation the temperature was placed to the left of the date and the weather type was placed to the right of the date. This was barely readable and it did not fit in the realized design hence a new icon was designed that contained the date, temperature, weather type and an open area in the middle for the Tizen icon for the swipe down menu (see figure 3.15).

---

<sup>1</sup><http://openweathermap.org/>



Figure 3.15: Version 6, the final design

### 3.2.6 Option lists

The app has access to two different option lists:

- the settings which can be reached by double tapping on the time, and
- the menu that can be reached by tapping in the middle of the translation.

The app started with one option list. When the app was able to show the words with the translation, the user should have an option to tell the app that a word had a wrong translation or that a word was learned and thus in both cases it should not reappear again. Due to the small screen it was not convenient to add more buttons for each of these options to the watchface and therefore we decided to make a menu with these extra options.

Besides these options the user should be able to reverse the order in which the wordpairs were asked (when the words were displayed from German to English, the words would be displayed from English to German after the button was pressed), to insert the number of words that the user wanted to learn and to log out of the app. All these options were reachable for the user by a 'menu' button in the bottom. The 'next' button was replaced by two buttons, the 'menu' and the 'next' button (see figure 3.16).





Figure 3.16: The design of the menu in version 4.

In the bottom the user could close the menu or open the settings. By pressing the ‘settings’ button the user would open the settings page with the buttons for reverse, number of words and log out (see figure 3.17).



Figure 3.17: The design of the settings in version 4.

After some discussions the overall opinion was that the ‘menu’ button should not be this

large on the watchface since the menu would give complementary options which were not part of the main function of the app. Therefore the button should not cover half of the bottom and a solution was to split the menu and the settings to two different pages that the user could reach independently. The user could reach the menu by tapping in the middle of the translation (see figure 3.13) and the settings could be reach by double tapping on the time since this space was hardly used. The double tap was necessary because tapping the time was already assigned to changing the background. The menu and the settings both had a black transparent background for a long time. This was inspired from the transparent and blurred swipe down menu from a well-known OS. Unfortunately the blur effect was not a success, but the transparency remained.

During testing the text in the buttons were hard to read and the buttons of the main page were visible through the buttons on the bottom. Therefore it was decided that the menu and the settings page should have a background. For the background the same image was used as the background of the wordpairs to preserve the unity of the design. Many translations depend on the context of a word. Hence the user should be able to see the context of the word and this option was added to the menu page (see figure 3.18).



Figure 3.18: The new design of the menu in version 6.

Later the idea was abandoned of letting the user choose how many words the user wants to learn. The 'profile' button was the latest addition to the settings page (see figure 3.19).



Figure 3.19: The new design of the settings in version 6.

### 3.2.7 Profile

The app has a profile page where the user can see the four medals that can be earned by using the app (see figure 3.20). The four categories are: words learned, total time, longest session and longest streak.



Figure 3.20: The profile page in which the user can find the latest earned achievements.

The idea was that when users can earn medals, they would be more willing to use the app. When the user presses the 'I learned it' button, the number of 'words learned' will increase and after 10 new learned words, a popup will appear with a motivating message to continue.

The total time the user used the app is also registered. When this time is the same as one of the pre-determined minutes a popup appears (see figure version 6 popup.png). With longest session the time is measured that the user uses the app continuously and with the longest streak the number of days is registered in which the app is used daily. One day of not using the app resets the longest streak.

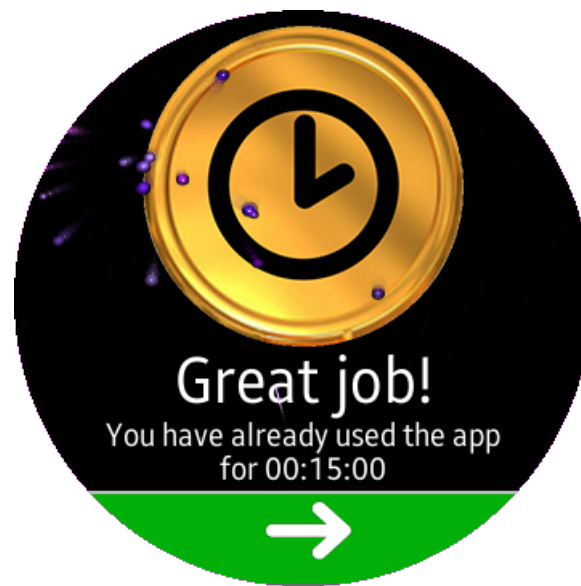


Figure 3.21: The design of the popup screen when the user improved an achievement.

### 3.2.8 Effects

To improve the experience of the app some effects were added: to give feedback after a button was pressed, to give information (why an option was not available or why a medal was earned) or to beautify a popup.

Because of the small screen the user could think the wrong button was pressed and therefore the user should get feedback about which button was pressed. The most important buttons in the app are the buttons used for indicating whether a word was known or not. When one of these buttons is pressed a green or red image appears depending on if right or wrong is pressed. This image stays on the display for a few milliseconds before it fades out.

In the app several popups are added to give the user feedback when an option is not available. The design of the popup is based on the first design of the option lists. The popup has a black transparent background with white text on it. The popup could appear when: there is no connection with the internet, when a wrong code is inserted, when the user has too few words, when there are too few words left on the watch or when a user earned a new medal. The popup that appears when a new medal is earned is beautified by a particle animation representing fireworks.

# 4

## The Implementation

In this part the implementation choices are described. In this project writing the code was one of the biggest challenges. The code has been changed a lot during the process. One of the reasons was to increase readability, but also because of design changes and new features which had to be implemented. The code is written in JavaScript in combination with HTML 5, the main web technologies at the moment. The code is build with a framework called require.js, this makes it possible to have multiple files (modules) in javascript, in order to increase readability and structure. In the following sections, we will zoom in some interesting and important decisions and choices related to the implementation.

### 4.1 Flash card algorithm

For presenting the words we used an algorithm which will make sure users will learn faster. The algorithm is based on the flashcard method (see fig. 4.1). The algorithm works in such a way that the user is able to give feedback in the form of ‘wrong’ and ‘right’. The words which are marked as ‘wrong’ will be faster repeated than words which were marked ‘right’. As explained in the design the user must indicate whether he had the answer wrong or right in his head. If the user had it wrong the word will be repeated after five more words, when the user had it right the timing when the word will

be repeated depends on the number of times the user had it correct. The word will be  $timesCorrect * 5$  positions moved in case the user had it right in his head.

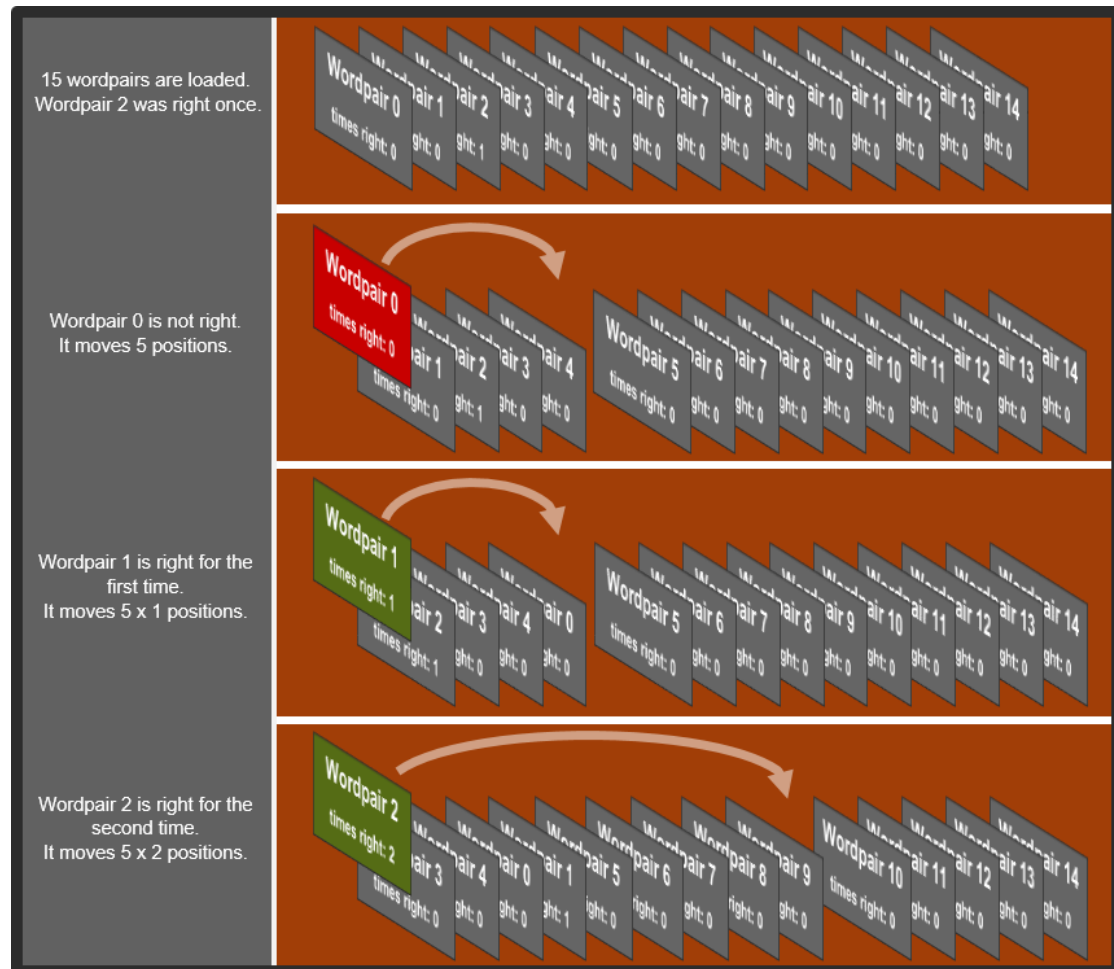


Figure 4.1: An example Flashcard simulation

Every time the user taps wrong or right the function *updateWordPair* is called (see listing 1). If the user is right the number of times correct will be increased. After that the current word will be moved in the array. This happens by first adding the word some positions further and then deleting the current word (i.e., deleting element zero in the array). If the user was wrong the current progression will be reset (i.e., *timesCorrect* becomes zero again), and the word will be moved five positions to the right (see figure 4.1). This means the word will be repeated reasonably fast.

Because the first element of the array is always shown on the screen, the next word will be drawn on the screen, since the previous first element is deleted. In *userData.js*

everything related to the user is implemented, so this is the place where words can be found. A wordPair is characterized with the following attributes: word, translation, id, context and number of times correct. Getting the word pairs from the server is explained in the next section.

```

1      updateWordPair: function(wordIsRight) {
2          if (wordIsRight) {
3              wordPair[0].timesCorrect++;
4              wordPair.splice(wordPair[0].timesCorrect *
NUMBER_OF_FLASHCARDS, 0, wordPair[0]);
5          } else {
6              wordPair[0].timesCorrect = 0;
7              wordPair.splice(NUMBER_OF_FLASHCARDS, 0, wordPair[0]);
8          }
9          wordPair.splice(0, 1);
10     },

```

Listing 1: flashcard implemenation in userData.js

## 4.2 Getting new words from the server

The module **session.js** declares function *getWords* which fetches words from the server. The module **session.js** is loaded in the module **main.js**, there the function *session.create* is called. This function takes the login code, which is provided by the **login.js** module (i.e., entered by the user or loaded from the userData). This module is also loaded in **main.js**. The login code is required to be submitted with every request to the server. The code is unique for every user. The login code can be found in the account when logged in the browser. The code contains eight digits. When a session is created (see listing 2), the first thing to look at is whether there any words on the watch. There are two situations:

- **If there words on the watch:** the global variable *status* in **session.js** will become ‘success’ (see listing 2 line 8). The user will now get in the screen where the words will be presented.
- **In case there are no words on the watch yet:** the watch has to communicate with the server and get the words with the endpoint ‘bookmarks\_to\_study’. This endpoint will return the words which are currently the most important to study for that particular user. The communication happens in the *getWords* function (see listing 2 line 5). In the implementation we made the choice to get fifty words in case the user has no internet connection for longer periods of time. The function *getWords* (see listing 3) can set the status to different states in different situations:



- if the words are successfully fetched from the server the status will become ‘success’ as described earlier on (see listing 2 line 26).
- if the code is invalid the status will become ‘wrong session number’ (see listing 2 line 30).
- if there is no internet connection the status will become ‘no connection’ (see listing 2 line 36).
- if there are too few words in the account of the user the status will become ‘too few words’ (see listing 2 line 11).

One of these states will be returned to the **main.js** module and this module will give this state through to the **login.js** module, in this module the user can then be informed by a popup that will inform the user about a specific situation. ML  
 ► *this is strange. you call `getWords(, false)` and this means `async=false`. But `async` should be `TRUE`! you want to have `async` calls! could it be that the name of the var should be actually “synchronous”* ◀

```

1 create: function(ctx, code) {
2   ctxWords = ctx;
3
4   if (!userData.areThereWords()) {
5     this.getWords(code, false);
6     this.updateWords();
7   } else {
8     status = "SUCCESS";
9   }
10 },

```

Listing 2: *session.create* in *session.js*

```

1 getWords: function(session, asynchronous) {
2   if (userData.getAllWords().length < NUMBER_OF_WORDS) {
3     try {
4       var xhr = new XMLHttpRequest();
5       xhr.open('GET', SESSION_ENDPOINT + BOOKMARK_SESSION +
6         NUMBER_OF_WORDS + "?session=" + session, asynchronous);
7       xhr.onload = function () {
8         try {
9           var obj = JSON.parse(this.responseText);
10          var wordNumber = 0;
11          if (length(obj) < userData.numberOfFlashcards()) {
12            status = "TOO_FEW_WORDS";
13          } else {
14            for (var i = 0; i < length(obj); i++) {
15              ctxWords.font = WORD_FONT;
16              // test both cases, because user may reverse the
17              words

```

```

16         if (isWordFittingTheScreen(obj[i].from,
MAX_WORD_LENGTH) && isWordFittingTheScreen(obj[i].to,
MAX_WORD_LENGTH)) {
17             ctxWords.font = TRANSLATION_FONT;
18             if (isWordFittingTheScreen(obj[i].from,
MAX_TRANSLATION_LENGTH) && isWordFittingTheScreen(obj[i].to,
MAX_TRANSLATION_LENGTH)) {
19                 setWordPair(wordNumber, obj[i].from, obj[i].to,
obj[i].id, obj[i].context);
20                 wordNumber++;
21             }
22         }
23     }
24     // extract the new words
25     receivedWords = getNewWordPairs(receivedWords,
userData.getAllWords());
26     status = "SUCCESS";
27 }
28 } catch(err) {
29     // the session number is unknown to the server
30     status = "WRONG_SESSION_NUMBER";
31 }
32 };
33 xhr.send();
34 } catch(err) {
35     // there is no internet connection
36     status = "NO_CONNECTION";
37 }
38 }
39 },

```

Listing 3: *session.getWords* in *session.js* showing communication with the server

**ML** ► *code between lines 16 and 19 should be extracted in a different function... this function is too long*◄

A successful login (i.e., the status was ‘success’) is not the only way to let the watch fetch new words. This is because the word list can become smaller if the user has marked a couple of words as ‘learned it’ or as ‘wrong translation’. After evaluating several scenarios, it was decided that the best moment for fetching new words was on a `screenOn` event (i.e., user makes arm twist to look at time). The new words will then be added to the current wordlist on a `screenOff` event (i.e., user makes again a arm twist to indicate he is not looking anymore), adding words at this point is to prevent any conflicts. The user is not actually using the wordlist when the screen turns off.

When writing the code the initial preference was to get words and add them at the same time on a screen off event, since the user won’t be using the application at that moment

and it would really feel as everything happens in the background. Unfortunately this didn't work the watch gets in a sort of sleep mode and won't be able to communicate with the server anymore. The GET request wasn't executed so it was impossible to do this on a screen off event. **ML** ► *but you can still update the list on a screen off! can we assume that network connections are not executed on screen off, but simple memory operations, like updating an array are? because in that case, you should write it here to be clear!*◀

### Synchronous vs. Asynchronous calls

When the watch tries to get new words from the server, this happens asynchronously to prevent the watch from getting slow. In the first implementation when it was implemented synchronously, **therefore** the user could have some delay before being able to interact with the watch. It was first implemented in this way to prevent conflicts, user cannot do something with a list while words are being added. This is now solved by getting the new wordPairs on a screen on (see listing 4) and add them later on a screen off (see listing 5).

Implementing this asynchronously resulted in considerable speedup, because when the user is able to accumulate a large number of words in his account (e.g., some user had already 10000 words) the endpoint will be very slow. One of the main reasons for slowness is that the list of words is not already sorted based importance on the server side.

```

1  function getNewWordPairs(newWords, currentWords) {
2    if (currentWords.length === 0) {
3      return newWords;
4    } else {
5      for (var j=0; j<currentWords.length; j++) {
6        for (var i=0; i<newWords.length; i++) {
7          if (currentWords[j].id === newWords[i].id) {
8            newWords.splice(i, 1);
9            break;
10         }
11       }
12     }
13   }
14   return newWords;
15 }
```

Listing 4: getting new wordPairs in session.js

```

1  addWords: function(numberOfWords, newWords) {
2    wordPair = wordPair.concat(newWords);
3    wordPair = wordPair.slice(0, numberOfWords);
4  },
```

Listing 5: adding new words to wordPair in userData.js

## 4.3 Usage Tracking

### 4.3.1 Events

Events are implemented to give the knowledge estimator information which can be used to analyze the knowledge of the user. The knowledge estimator is implemented on the server side. This means we had to decide at what point we should send the events. Because the knowledge estimator decided what the next word is going to be for the user, the events should be send when the user taps wrong or right. This is the moment the user will be presented with a new word. The knowledge estimator can receive the following events: `reveal`, `right`, `wrong`, `wrong translation`, `learned it`, `showContext`, `screenOn` and `screenOff`. In the table 4.1 the events and their definitions are described.

Table 4.1: Events

Type	Definition
<code>screenOn</code>	The screen lightens up.
<code>reveal</code>	The user wants to see the translation and presses the ‘reveal’ button or area.
<code>right</code>	The user knows the translation and presses the ‘right’ button or area.
<code>wrong</code>	The user does not know the translation and presses the ‘wrong’ or area.
<code>wrongTranslation</code>	The user thinks the translation is incorrect and presses the ‘wrong translation’ button in them menu.
<code>learnedIt</code>	The user thinks the word is learned and presses the ‘I learned it’ button in the menu.
<code>showContext</code>	The user wants to see the context of the word and presses the ‘show context’ button in the menu.
<code>reverse</code>	The user wants to learn the other way around and presses the ‘reverse’ button in settings.
<code>screenOff</code>	The screen turns off (automatically, after a timeout or arm twist).

### 4.3.2 Touch tracking

The touch tracker is designed for usability purposes. Its purpose is to track where the user taps, so the coordinates are saved. Not only the tap position is saved, also the type of the tap (e.g., user taps on reveal). For some functions this can be really interesting like the right button; the user can tap on the button itself or on the green space above the button. Both actions result in the same thing. In the future we could change the design based on these results to give the user a better experience. To conclude the touch tracker is not implemented as a user-facing feature, it is purely for telemetry purposes.

## 4.4 Changing the watchface and sending and saving the events and taps

Sending and saving the taps and events was a challenge in this project. The user is able to change the watch face at every moment, and this ~~will~~→could ML ► *or do you really lose data? if not then don't say will!*◀ result in losing data. User interaction taps and events should be saved to the storage of the watch immediately. We do this by saving them to the local storage. This is implemented in the same way as saving to the local storage in a web browser.

The taps in the storage are sent to the Zeeguu server with a PUT request on a screen-on event asynchronously, so it won't influence the speed of the application and it happens in the background. The endpoint is used for tracking all sorts of user activity and is called 'upload\_user\_activity\_data'. When the taps are successfully sent to the server, the server will return 'OK', if the watch then receives this message the taps in the storage can be freed, they are now at the server. The events in the storage are sent on 'right' and 'wrong' as previously explained to inform the knowledge estimator. Just as with the taps the events are deleted from the storage if they successfully have been ~~arrived~~→received on the server side.

Losing data with changing the watch face not only happens with taps and events, it also happens when the user reverses the way the words are asked (e.g., English - Dutch to Dutch - English) and with changing the background.

The login code is being saved the first time and used for automatic login after a watch face change.

## 4.5 Drawing the user interface

The user interface is rendered every second on the screen, this happens in the module `gui.js` in the `render` function which is public (shown in listing 8). The `main.js` module has an function which calls this function in the `gui.js` module every second (shown in listing 9). The screen is thus updated every second. The `render` function consists of a `update` function and a `draw` function.

In the `update` function on `profile.refresh` the achievements are refreshed which were described in the design chapter. The time is updated and the background is rotated based on the current time, and sunset and sunrise of the current location (see listing 6 line 4).

In the `draw` function the time, battery and current weather are drawn on the screen. If there is no internet connection the weather won't be drawn and nothing is visible where the weather normally is shown.

**ML** ► *Guys! i still want this rotation thing to be moved to it's own function. it is not at the same abstraction level as the previous two .refresh() and .refresh()! this looks bad!◀*

```

1  function update() {
2      profile.refresh();
3      time.refresh();
4      background.rotate(weather.getSunrise(), weather.getSunset(), time.
        getHours()*60 + time.getMinutes()*1);
5  }
```

Listing 6: update function in gui.js

```

1  function draw() {
2      time.draw();
3      time.drawDate();
4      battery.draw();
5      weather.draw();
6  }
```

Listing 7: draw function in gui.js

```

1  render: function() {
2      update();
3      draw();
4  },
```

Listing 8: render function in gui.js

```
1  function updateScreenEverySecond() {  
2      gui.render();  
3      setTimeout(updateScreenEverySecond, 1000);  
4  }
```

Listing 9: updateScreenEverySecond function in main.js

## 4.6 Testing

During the project the implementation of the app changed constantly. In order to maintain the functionality of the app after these adjustments, test cases were made (see appendix A) to test whether the different features were still functional. The test cases are divided into different scenarios: login, background, settings, reveal, feedback, menu, profile, mainpage and popup.

For each of these scenarios different test cases were made with the same format:

1. Test Case: describes what the test case is about.
2. Pre-conditions: describes the steps that must be fulfilled in order to start the test.
3. Test Step: describes the steps that should be taken for the test.
4. Test Data: gives a list of the necessary data for the test.
5. Post-conditions: describes anything that applies after the test case completes.
6. Expected Result: describes what the app should look like after the test.
7. Actual Result: describes what the app looks like after the test.
8. Pass/Fail: compares the expected result with the actual result and decide whether the app passes or fails for the test.

# 5

## Usage Study

After the implementation the app was ready to enter the test phase. The results of the test period should give answers to the following two questions: is the app used during short intervals (this would indicate that the users used the app for micro learning) and is the app designed properly (e.g., are all the options found during the test period).

The test period consisted of four parts:

1. All participants started with making an account on Zeeguu and used Zeeguu Reader for reading. During this usage, the platform saved words they did not understand.
2. The participant received the smartwatch with the app as a watchface and basic instructions about how to use the app.
3. The participant used the app for four days in a row.
4. After four days the participants handed in the smartwatch and they filled in a questionnaire about the test period and how they experienced the app.

During the test period the smartwatch kept track of the usage by using telemetry. All the buttons and touch areas for the existing options had different events attached to them so when a button or area was pressed, an event would be sent to the server. The events and their definitions are shown in table 4.1.



## 5.1 Usage results

After all the participants finished their test period all the events per user were collected from the server and analyzed. For answering the research questions and for giving a clear overview about how the app was used during the test period, the following diagrams were made per user:

- A pie chart diagram presenting the overall usage of the smartwatch in seconds
- A pie chart diagram about the duration of the learning sessions on the app in seconds
- A bar chart about the number of times a user pressed ‘right’ and ‘wrong’
- A table with the median reaction time between a ‘reveal’ and a ‘right’ or ‘wrong’ and the time the app was used

### 5.1.1 Overall usage.

**ML** ► *the diagrams are not the main point of the section. they are only the way you present the data. the main point is the Overall Usage! Same thing for all the remaining sections. Title should be the topic of discussion, not the type of diagram!! Also, “in seconds” was removed.* ◀ These diagrams (see fig. 5.1 and 5.2) were created by collecting all the time intervals between a `screenOn` and a `screenOff` event. This data was then sorted into five different intervals:

**ML** ► *Refactored this text to move the discussion near the time interval. Please re-read and make sure that the text make sense and fix where needed!* ◀

- $\text{time} \leq 2\text{s}$ . The first interval was chosen, assuming that checking the time or other arm movements would not take more than two seconds.
- $2\text{s} < \text{time} \leq 5\text{s}$ . When the duration is longer than two seconds the user probably does more than only checking the time. The user could check the time, but because a word is shown too, the user might reveal that word by pressing ‘reveal’ and then the user might also give feedback by pressing ‘right’ or ‘wrong’. These actions for one word would take not more than five seconds. Summarized, these could be learning sessions that were not intended to be a learning session.
- $5 < \text{time} \leq 15$ . ~~Then~~ → These are the intended learning sessions but for a really short time
- $15 < \text{time} \leq 60$ . These are the indended sessions that took a bit longer.

- time > 60. As mentioned before the learning app will probably only be used for short sessions and thus sessions longer than one minute will probably hardly occur.

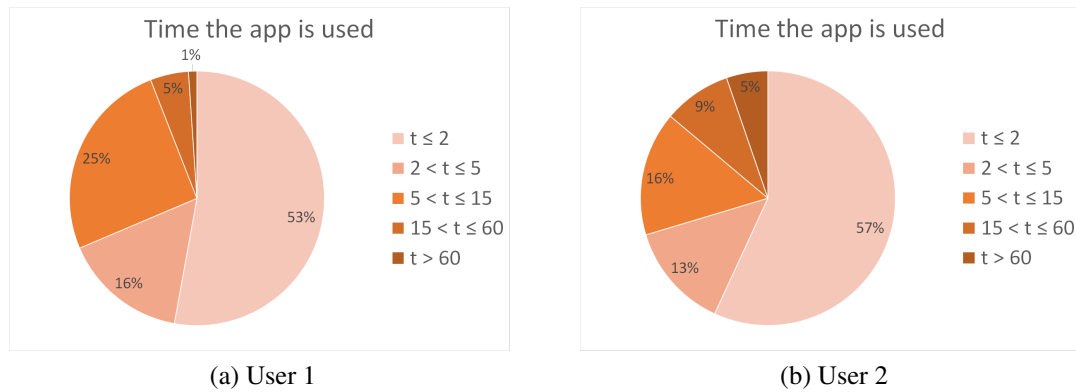


Figure 5.1: General usage users 1 and 2

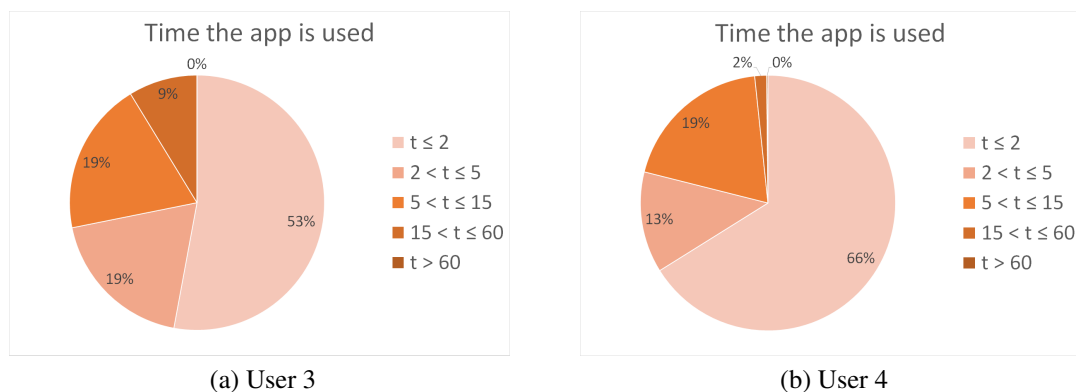


Figure 5.2: General usage users 3 and 4

The graph about the overall usage of the smartwatch provides insight into the general behavior of the smartwatch users. The graph shows that most of the time (53% till 66%) the smartwatch is used less than or equal to two seconds. This can be related to checking the time which **could mean that the ratio between checking the time and learning words is roughly 50 to 50.** ML ►nice conclusion. can be emphasized◀ This is in a well agreement with the design where 50% of the screen is used for the time and the other 50% for learning words.

The graphs also shows that a learning session on the smartwatch rarely lasts for more than 60 seconds. The smartwatch is not suitable or convenient enough to be constantly used for a long period.

### 5.1.2 Pie chart diagram presenting the learning sessions in seconds

The durations of the learning sessions were found by using an algorithm (see listing 10). This algorithm first sorted out all the `screenOn` events that were immediately followed by a `screenOff` event. A user that used the app for learning and not for checking the time would use at least one option the app provides for learning new words before the screen would turn off. Therefore the combinations of a `screenOn` and `screenOff` events were seen as not learning sessions and these pairs were erased from the list of events<sup>1</sup>. In the remaining events the time was measured between a `screenOn` and a `screenOff` event and afterwards the data was sorted into four intervals and visualized in pie chart diagrams (see fig. 5.3 and 5.4):

- $\text{time} \leq 5$
- $5 < \text{time} \leq 15$
- $15 < \text{time} \leq 60$
- $\text{time} > 60$

The app will presumably be used for micro learning. Therefore the number of sessions will probably decrease exponentially when the duration increases. This is the reason why the length of the intervals increases faster than a linear growth.

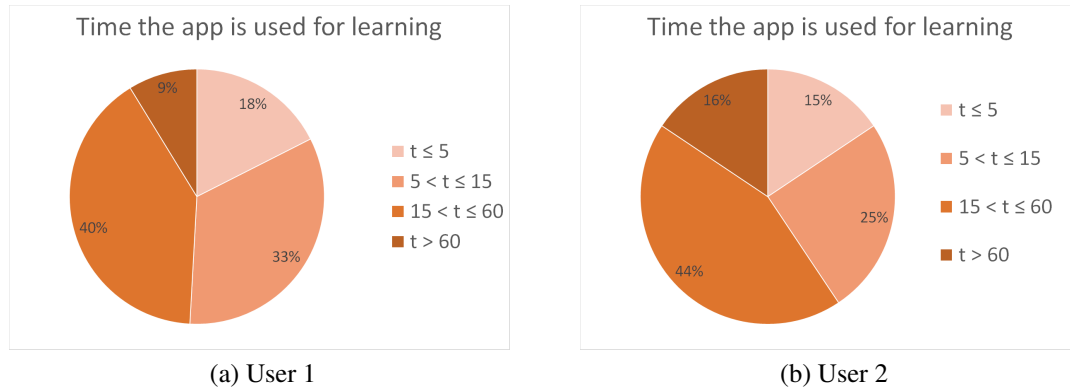


Figure 5.3: Learning sessions time users 1 and 2

<sup>1</sup>There is possibly some noise here as a user might have left the word and the translation on the screen and see them displayed nevertheless

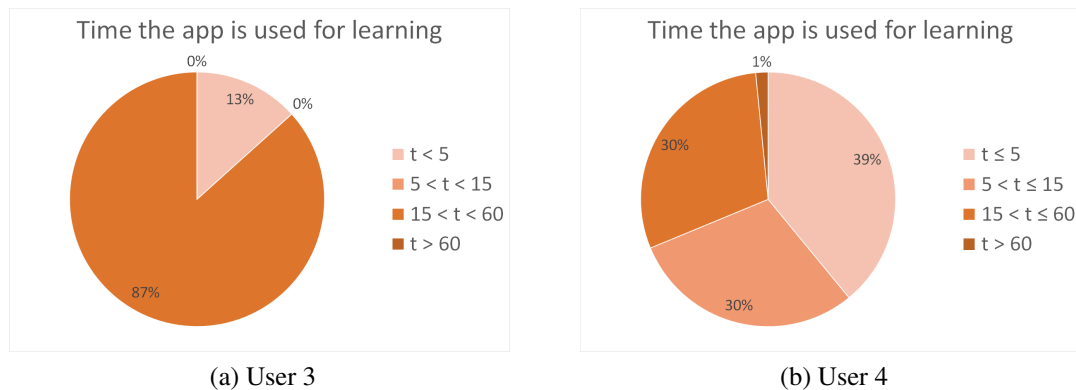


Figure 5.4: Learning sessions time users 3 and 4

The graph about the learning sessions shows a different distribution when it comes to the duration of the sessions. **When the smartwatch is used to learn words, the session last for → between 5 till → and 60 seconds most of the time where → and the interval of 15 till → to 60 seconds is often the largest.** ML ► *again, nice conclusion – can be bold-ed* ◀ Sessions that are used for learning and that last for more than 60 seconds are still a small percentage of all the sessions. In the graph from user 3 the distribution is a little bit skewed, this is probably due to a lack of results. This user didn't use the watch as often as the other users.

### 5.1.3 Bar chart visualizing ratio between 'right' and 'wrong'

For each day the app was used the number of right and wrong events were counted and plotted in a bar chart (see fig. 5.5 and 5.6).

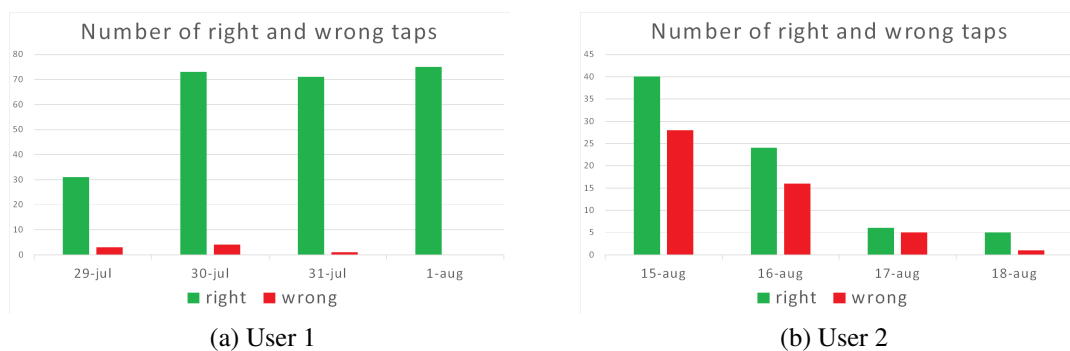


Figure 5.5: Number of right and wrong taps for users 1 and 2

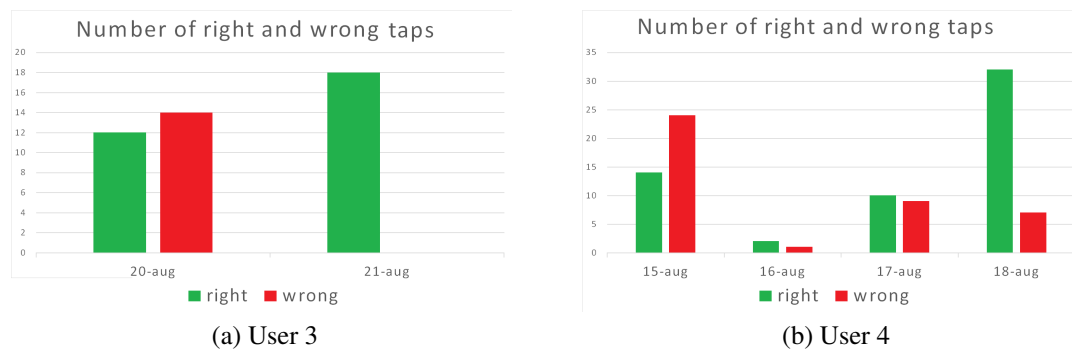


Figure 5.6: Number of right and wrong taps for users 3 and 4

The graphs show that ‘right’ is more often pressed than ‘wrong’. This might come as a surprise, since learning new words should start with more ‘wrong’ than ‘right’. In the graph of user 1 (see fig. 5.5) this is shown extreme, this user had a German background and was learning German. This might be a possible explanation for the high number of ‘right’ taps. In the graph of user 2 (see fig. 5.5), a decrease in usage is seen. At 17 August the [battery of the](#) watch was depleted, so therefore usage was less. At 18 August usage was less since it was not a full day of usage, the user handed in the watch at noon. In the graph of user 3 (see fig. 5.1.4) it can be seen that the watch is unfortunately only used for 2 days, there is however a clear learning curve which can be observed in the graphs of users 1 and 4.

### 5.1.4 Table with median reaction time and daily usage

The reaction time is the time it took the user to press ‘right’ or ‘wrong’ after ‘reveal’ was pressed. The average was calculated for each single day and for the whole period of four days. Besides the average time the total time is mentioned too. This indicates for how long the user used the app during the four days of testing (see tables 5.1, 5.2, 5.3 and 5.4).

Table 5.1: User 1 average reaction time and total usage time

<b>Date</b>	<b>Average</b>	<b>Time</b>
29-07-2016	00:00:02	00:08:39
30-07-2016	00:00:13	00:13:10
31-07-2016	00:00:01	00:45:43
01-08-2016	00:00:36	00:37:07
<b>Average</b>	00:00:07	00:26:10
<b>Total</b>		01:44:39

Table 5.2: User 2 average reaction time and total usage time

<b>Date</b>	<b>Average</b>	<b>Time</b>
15-08-2016	00:00:08	00:19:45
16-08-2016	00:00:02	02:15:03
17-08-2016	00:00:02	00:36:00
18-08-2016	00:00:02	00:01:00
<b>Average</b>	00:00:05	00:47:57
<b>Total</b>		02:35:48

Table 5.3: User 3 average reaction time and total usage time

<b>Date</b>	<b>Average</b>	<b>Time</b>
21-08-2016	00:04:20	00:18:21
22-08-2016	00:00:04	00:02:23
<b>Average</b>	00:02:12	00:10:22
<b>Total</b>		00:20:44

Table 5.4: User 4 average reaction time and total usage time

<b>Date</b>	<b>Average</b>	<b>Time</b>
16-08-2016	00:03:54	00:37:23
17-08-2016	00:00:03	04:05:46
18-08-2016	00:12:34	00:28:02
19-08-2016	00:02:30	00:13:41
<b>Average</b>	00:04:45	01:21:13
<b>Total</b>		05:24:52

The average reaction time between a `reveal` and a `right` or `wrong` differs per user. The first two participants had nearly the same reaction time which is quite fast compared to the other participants. A fast reaction could mean that they are fast learners and they do not have to spend a lot of time to learn the translation. This thought is confirmed by the bar chart where the two participants with a high reaction time had pressed ‘right’ more often than ‘wrong’.

The other two participants had a much longer reaction time in comparison with the first two reaction times. It could be that these users needed more time to learn a new word and therefore they required more time to learn the translation of a certain word. In the bar chart of the user 4 (see fig. ) it can be seen that in the first day ‘wrong’ is pressed more than ‘right’ what could indicate that the user was confronted with some hard-to-learn words.

## 5.2 Questionnaires results

In order to get even more feedback from the user it was decided to let them answer a questionnaire after their test period. The questionnaire can be found in appendix B and the answers of the users in appendix C. The purpose was to get specific information about the app which cannot be obtained by only using the events. Although the test group was really small, it only consisted of four persons, we can still learn from it. The questionnaire was divided into two sections, questions related to:

- general information about the user (i.e., what kind of users do we have)
- the usage of the smartwatch app.

In the sections below the results are summarized and evaluated.

### 5.2.1 General information about the user

All four of the users were relatively young, the youngest person was 22 and the oldest 34. It can thus be concluded that the test group was quite young. This could mean that they prefer certain ways of learning which fits the generation.

It was interesting to see that people don’t invest much time in learning new words (see question 3 from the general information in the questionnaire). This could simply mean that they don’t have much time, or that they just hate learning new words for long periods of time. Either case using the smartwatch app helps them invest more time in learning new words using micro learning.

For the people who already were learning, the most frequent method was by reading texts in the other language. This would suggest that using the Zeeguu ecosystem [5] (as explained in related work) in combination with the smartwatch is way that people who are enthusiastic about learning a new language would like.

### 5.2.2 Smartwatch app usage

In general the reactions to the design were positive, the context function might need a larger font, but this will also result in more words being removed from the list, since the context won't fit on the screen. There were also some complains about not being able to find all the features the app offers. This was due to not having instructed some of the testers properly, but also because some features might not be very logical to find. The settings for example could be find by double tapping the time. This may not be the most convenient place for the user. A solution would be to have a sort of short instruction manual in the app itself, or to make some small button on the screen for settings.

There were two interesting features mentioned that people would like to have in app. The first one was a pronunciation function, this however is hard to implement, because the Samsung Gear S2 does not have a speaker built in. Another feature which was mentioned was "I don't want to learn this word", one way of doing this would be to combine "wrong translation" and "I don't want to learn this word" to "trash", in either way the word has to be removed. The disadvantage would be less specific user feedback.

The users used the app exactly how we hoped they would, they used it mostly when they had to wait for something. This is exactly what micro learning is all about, the small wait moments will now be filled.

## 5.3 Threats to Validity

[The main threats to the validity of the conclusions in this chapter are:](#)

- Threats to External Validity. The conclusions can not be generalized based on only four users. However, they represent useful incipient data.
- Threats to Internal Validity. It could be that there is some noise in the data. The smartwatch froze several times, and this might determine some very long sessions to be recorded.
- **ML** ► *anything else that you think could affect the validity of the conclusions, must be added here.*◄



# 6

## Conclusion and Future Work

### 6.1 Conclusion

There are still some improvements that should be implemented in the app eventually. During the test period the users were dealing with a few crashes that were probably caused by functions that require an internet connection.

After analyzing the questionnaire it was clear that the users did not find all features the app has to offer them. This was caused by the lack of a proper instruction and the navigation that appeared to be not very intuitive for some features.

In the questionnaire the users stated the time they spent on learning new words daily. When we compared these results with the time the users spent on the smartwatch for learning, we can conclude that they spent more time on learning with the watch than without the smartwatch. This could be caused by the ability to have short learning session that is doable for the users. The users could experience some trouble with concentrating on the words for a long time, since the learning sessions without a smartwatch are much longer than with the smartwatch. Because the use of micro learning in combination with the app the users were able to increase the time they spent on learning.

The design of the app was simple and neat according to the participants. Hence this design will probably be used in the future.

After analyzing the results of the user study the questions stated in the introduction can

be answered.

**How do users use a smartwatch application to accelerate the memorization process of somebody who is learning the vocabulary of a second language?**

The users that use a smartwatch application to learn new words use micro learning in order to learn new words. These short learning session can take place whenever the user has some time left. A few examples mentioned in the questionnaire are waiting for the traffic lights, taking the elevator or during watching television.

- **How long does a session on a smartwatch take (i.e., general usage)?**

Most of the sessions last for less than three seconds and sessions of more than 60 seconds hardly appear as can be seen in the results from the user study. The smartwatch is therefore not used for long sessions by our users.

- **How long does a learning session take on a smartwatch (i.e., only using Time to Learn app)?**

The duration of a learning session is between 5 and 60 seconds most of the time. From the four different intervals the group between 15 and 60 seconds was the largest. These short learning sessions are in alignment with micro learning.

- **How long do users use the app daily on average?**

The time that the users spent daily on the app differs from one another. This difference between the daily usage of each user is too large to give a general answer to this question. The usage differs from one minute till four hours.

## 6.2 Future work

As far as we know this is the first time a smartwatch app was made for learning vocabulary. In this project we tried to give the user the best possible experience to make learning as efficient and fun as possible. Naturally there are still some interesting features that can be implemented to improve the functionality of the app. Some examples are:

- **Improvements** → **Improving the server side**

Several events are sent to the server when the app is used. These events were used for the test results. However these events could also be used to personalize the learning process. Every user has a different learning curve that plots the time it takes before a word will be forgotten. It is therefore important that a learning app shows that word again, before the user forgets the translation. A recent research project developed at the RUG tried to calculate this curve in order to maximize the efficiency of a learning app[12]. The algorithm for the learning curve could be implemented on the server side where it could use the events to estimate the learning curve for all users. The algorithm will send a word to the

watch that the user should learn according to his learning curve. The downside of this approach for the used smartwatch model will be that the watch should constantly be connected to the internet to keep the learning curve up-to-date. The connection is also important for receiving new words from the server. There are smartwatch models that support 3G and therefore keeping a connection with the internet should not be a problem for these models.

- **Knowing the form of the word**

In the ‘Related Work’ chapter some techniques are mentioned which could improve the time it takes to learn new words. One of these techniques was categorizing the words into different forms, like: verbs, nouns or prepositions etc. so it is clear for the learner in which situation the word should be used. We are still unsure whether there is a database with the form for different words. If it exists, the app could show this form for every word although it will be hard to find space on the small screen.

**ML** ► *renamed these two next bullets because they sounded too similar to each other!* ◀

- **Exemplifying the Pronunciation of a word**

After the test period almost all the users would like to have a pronunciation function. The smartwatch model we used did not have speakers, so this function would not work for this model. However, there are models which do have speakers and there are databases available with the pronunciation of a lot of words in different languages which means that this idea could be implemented in a later version of the app.

- **Verifying a users’ pronunciation**

Another way to improve learning is to say the translation aloud. It can be read in the ‘Related Work’ chapter that pronouncing a word aloud helps the saving process in the brains. The smartwatch model we used has a build-in microphone like many more models. This makes it possible to check if the user says something. This function could even be improved by verifying if the word is pronounced the right way.

- **Avoiding interference with words of similar spelling or meaning**

Words with a similar spelling or meaning should not be shown one after another in order to prevent confusion. This would require an algorithm and a database that decides whether two words have a similar spelling and whether these words have a similar meaning.

- **Improve→ing the functionality of the app**

The method for showing new words on the app can be improved by showing the

word and the translation simultaneously if it is the first time the user sees that word. After the first time the word and the translation can be shown in the same manner the app is showing the words in the latest version.

Since knowing the context of a certain word is important for the learning process, the ‘reveal’ button could be split in two buttons with on the left the option to show the context and right the option to reveal the translation. The user will then be able to make an educated guess when reading the context of the word. In the latest version this is not possible since the option ‘show context’ is only available when the translation is already revealed.

The settings should be opened when the user taps the time and double tap on the time should change the background. Changing the background is less important than the settings and thus entering the settings should need less afford than changing the background.



## Test Cases

In this appendix the test cases are included which were described in chapter 5.







### Test cases

Test Scenario	Test Case	Pre-conditions	Test Step	Test Data	Post-conditions	Expected Result	Actual Result	Pass/Fail
Check login functionality without code in storage.	Check response on entering valid code.	<ol style="list-style-type: none"> <li>1. The app 'timeToLearn' must be installed.</li> <li>2. The user tries to log in for the first time. No code is saved in storage.</li> <li>3. 'timeToLearn' is the current watch face.</li> <li>4. The watch is connected to the internet.</li> </ol>	<ol style="list-style-type: none"> <li>1. The user enters the first four digits using the 'plus' and 'minus' buttons.</li> <li>2. The user presses next.</li> <li>3. The users enters the last four digits using the 'plus' and 'minus' buttons.</li> <li>4. The user presses next.</li> </ol>	Code: 61015763	<ol style="list-style-type: none"> <li>1. The code is saved to the localStorage.</li> <li>2. The words of the account are loaded with the code. And saved to the localStorage.</li> <li>3. The mainPage becomes visible.</li> </ol>	Login must be successful, the user is prompted with the main screen.	See expected result.	✓
	Check response on entering invalid code.	<ol style="list-style-type: none"> <li>1. The app 'timeToLearn' must be installed.</li> <li>2. The user tries to log in for the first time. No code is saved in storage.</li> <li>3. 'timeToLearn' is the current watch face.</li> <li>4. The watch is connected to the internet</li> </ol>	<ol style="list-style-type: none"> <li>1. The user enters the first four digits using the 'plus' and 'minus' buttons.</li> <li>2. The user presses next.</li> <li>3. The users enters the last four digits using the 'plus' and 'minus' buttons.</li> <li>4. The user presses next.</li> <li>5. The user presses the screen when the popup appears.</li> </ol>	Code: 00000000	<ol style="list-style-type: none"> <li>1. The entered code will be erased.</li> <li>2. The login screen will be shown again.</li> </ol>	A popup appears telling the user the code is wrong. When the popup is pressed, the user is back to login.	See expected result.	✓
	Check response if there is no connection with the internet.	<ol style="list-style-type: none"> <li>1. The app 'timeToLearn' must be installed.</li> <li>2. The user tries to log in for the first time. No code is saved in storage.</li> <li>3. 'timeToLearn' is the current watch face.</li> <li>4. The watch is not connected to the internet.</li> </ol>	<ol style="list-style-type: none"> <li>1. The user enters the first four digits using the 'plus' and 'minus' buttons.</li> <li>2. The user presses next.</li> <li>3. The users enters the last four digits using the 'plus' and 'minus' buttons.</li> <li>4. The user presses next.</li> <li>5. The user presses the screen when the popup appears.</li> </ol>	Code: 61015763 (Any code can be entered here)	<ol style="list-style-type: none"> <li>1. The entered code will be erased.</li> <li>2. The login screen will be shown again.</li> </ol>	A popup appears telling the user there is no connection to the internet. When the popup is pressed, the user is back to login.	See expected result.	✓
	Check response if the user has to few words in his account	<ol style="list-style-type: none"> <li>1. The app 'timeToLearn' must be installed.</li> <li>2. The user tries to log in for the first time. No code is saved in storage.</li> <li>3. 'timeToLearn' is the current watch face.</li> <li>4. The user has less than five words in his account</li> </ol>	<ol style="list-style-type: none"> <li>1. The user enters the first four digits using the 'plus' and 'minus' buttons.</li> <li>2. The user presses next.</li> <li>3. The users enters the last four digits using the 'plus' and 'minus' buttons.</li> <li>4. The user presses next.</li> <li>5. The user presses the screen when the popup appears.</li> </ol>	Account with less than 5 words saved.	<ol style="list-style-type: none"> <li>1. The entered code will be erased.</li> <li>2. The login screen will be shown again.</li> </ol>	A popup appears telling the user there are too few words in his account. When the popup is pressed, the user is back to login.	See expected result.	✓
Check login functionality with code in storage.	Check response after restart with code in storage.	<ol style="list-style-type: none"> <li>1. The app 'timeToLearn' must be installed.</li> <li>2. 'timeToLearn' is the current watch face.</li> <li>3. The user has to be logged in at least once, so a code is saved.</li> <li>4. The watch is currently off.</li> </ol>	The user holds the power button.	None	<ol style="list-style-type: none"> <li>1. The mainPage is loaded.</li> <li>2. The userData is loaded.</li> </ol>	The watch goes on, the watch face is loaded and the user will be in main screen where the user sees the last seen word.	See expected result.	✓








Test Scenario	Test Case	Pre-conditions	Test Step	Test Data	Post-conditions	Expected Result	Actual Result	Pass/Fail
	Check response after changing watch face with code in storage	<ol style="list-style-type: none"> <li>1. The app 'timeToLearn' must be installed.</li> <li>2. 'timeToLearn' is the current watch face.</li> <li>3. The user has to be logged in at least once, so a code is saved.</li> </ol>	<ol style="list-style-type: none"> <li>1. The user changes the watchface by pressing and holding the screen.</li> <li>2. The user selects a different watch face.</li> <li>3. The users presses and holds the screen again.</li> <li>4. The user selects the 'timeToLearn' app as his watch face</li> </ol>	None	<ol style="list-style-type: none"> <li>1. The mainPage is loaded.</li> <li>2. The userData is loaded.</li> </ol>	The app used for the watch face will be changed to some other app and then changed back to the 'timeToLearn' app. The user will be in the main screen where user sees the last seen word.	See expected result.	✓
check background functionality	Check response on tapping on the time.	The main page is shown.	Tap on the time.	None	The index of the array where all landscapes are saved is increased by one.	The landscape of the background of the time changes to another landscape	See expected result.	✓
	Check response on double tapping on the time.	The main page is shown.	Double tap on the time.	None	The settings layer is set to visible.	Over the main page the settings page is shown.	See expected result.	✓
	Check response when changing the watchface	The main page is shown.	<ol style="list-style-type: none"> <li>1. Tap on the time to change background.</li> <li>2. Hold screen to change the watch face.</li> <li>3. Change back to timeToLearn watch face.</li> </ol>	None	Background number is increased and saved to localStorage.	Background is the same as before changing the watch face.	See expected result.	✓
check settings functionality	Check response on tapping on "reverse".	1. User is on the main page.	<ol style="list-style-type: none"> <li>1. Double tap on the time.</li> <li>2. Tap the button with the text "reverse".</li> <li>3. Tap the word.</li> <li>4. Tap on the left or on the right of the word to get another word.</li> </ol>	None	The variable "reverse" in userData is changed.	The translation of the previous shown word is shown instead of the word. After tapping the translation, the previous shown word is shown below the translation. All words after are shown in reversed order.	See expected result.	✓
	Check if reverse state is the same after changing watchface.	User is on the main page	<ol style="list-style-type: none"> <li>1. Double tap on the time.</li> <li>2. Tap the button with the text "reverse".</li> <li>3. Change the watch face</li> </ol>	None	The variable "reverse" in userData is changed and saved to local storage.	After changing the watch face and switching back to TimeToLearn app, the order of the words is the same as before.	See expected result.	✓
	Check response on tapping "profile"	The main page is shown.	<ol style="list-style-type: none"> <li>1. Double tap on the time.</li> <li>2. Tap the button with the text "profile".</li> </ol>	None	The visibility of profilePage is set to visible.	The profile page is shown with on the top a banner with "profile", in the middle a grey space with the medal for "words learned" is shown together with the number of words you have learned. On the bottom there is a back button to leave the profile page. On the side there are two buttons to navigate through the medals.	See expected result.	✓
	Check response on tapping "log out"	The main page is shown.	<ol style="list-style-type: none"> <li>1. Double tap on the time.</li> <li>2. Tap the button with the text "log out".</li> </ol>	None	The localStorage is erased and the login screen visibility is set to visible.	After "log out" is clicked, the settings page disappears and the login screen is visible again.	See expected result.	✓
	Check response on tapping on the settings page.	The main page is shown.	<ol style="list-style-type: none"> <li>1. Double tap on the time.</li> <li>2. Tap the screen anywhere besides the other buttons.</li> </ol>	None	The visibility of the settings page is set to hidden and thus the main page is visible again.	By tapping anywhere besides the other buttons, the settings page fades out and thus the main page is visible again.	See expected result.	✓

Test Scenario	Test Case	Pre-conditions	Test Step	Test Data	Post-conditions	Expected Result	Actual Result	Pass/Fail
	Check response on tapping on the back button in settings.	The main page is shown.	1. Double tap on the time. 2. Tap the button with the back arrow on it.	None	The visibility of the settings page is set to hidden and thus the main page is visible again.	By pressing the button, the settings page fades out and thus the main page is visible again.	See expected result.	<input checked="" type="checkbox"/>
	Check response after an screen off.	The main page is shown.	1. Double tap on the time. 2. Wait for the watch screen to switch off. 3. After a screen off press the power button to turn on the screen.	None	The visibility of the settings page is set to hidden when the screen of the watch turns off and thus the main page is visible again when the watch is switched on.	When the screen turns on. The main page is visible.	See expected result.	<input checked="" type="checkbox"/>
check reveal functionality	Check response on tapping on the word.	The main page is shown with a word or some words visible just below the middle, an open space and a orange button with glasses on the bottom of the screen.	Tap on the shown word(s).	None	The revealed page is set to visible. This page consists of the word(s), the translation and three other buttons: wrong, menu, right	Below the previous shown word(s) the translation appears. On the left and on the right a red and green area appears to indicate that this space detects taps too. On the bottom three buttons appear. From left to right: wrong (pressed when the user did not know the translation), menu (for extra options related to the words) and right (pressed when the user did know the translation).	See expected result.	<input checked="" type="checkbox"/>
	Check response on tapping the button with the glasses icon.	The main page is shown with a word or some words visible just below the middle, an open space and a orange button with glasses on the bottom of the screen.	Tap on the orange button with the glasses on it.	None	The revealed page is set to visible. This page consists of the word(s), the translation and three other buttons: wrong, menu, right	Below the previous shown word(s) the translation appears. On the left and on the right a red and green area appears to indicate that this space detects taps too. On the bottom three buttons appears. From left to right: wrong (pressed when the user did not know the translation), menu (for extra options related to the words) and right (pressed when the user did know the translation).	See expected result.	<input checked="" type="checkbox"/>
	Check response on tapping the wrong space.	The main page is shown with a word or some words and its translation under it. Left and right there is a red and a green area and below there are three buttons: red, orange and green.	Tap on the left side of the screen, on the red area.	None	By tapping the wrong space an event is sent to the server if there is a connection. If not the event is saved so it can be sent when there is a connection. The word is placed 5 steps further in the wordPair so it appears again after 5 words.	After clicking the wrong space a red image appears with an open book to notify the user about the clicked space. This image fades out and a new word appears on the main page.	See expected result.	<input checked="" type="checkbox"/>
	Check response on tapping the wrong button.	The main page is shown with a word or some words and its translation under it. Left and right there is a red and a green area and below there are three buttons: red, orange and green.	Tap on the left red button on the bottom of the screen.	None	By tapping the wrong button an event is sent to the server if there is a connection. If not the event is saved so it can be sent when there is a connection. The word is placed 5 steps further in the wordPair so it appears again after 5 words.	After clicking the wrong button a red image appears with an open book to notify the user about the clicked space. This image fades out and a new word appears on the main page.	See expected result.	<input checked="" type="checkbox"/>



Test Scenario	Test Case	Pre-conditions	Test Step	Test Data	Post-conditions	Expected Result	Actual Result	Pass/Fail
check feedback functionality	Check response on tapping the right space.	The main page is shown with a word or some words and its translation under it. Left and right there is a red and a green area and below there are three buttons: red, orange and green.	Tap on the right side of the screen, on the green area.	None	By tapping the right space an event is sent to the server if there is a connection. If not the event is saved so it can be sent when there is a connection. The word is placed 5 steps further after the first encounter in the wordPair so it appears again after 5 words. If the word is right again it will be placed 10 positions further in the wordPair.	After clicking the right space a green image appears with a graduation cap to notify the user about the clicked space. This image fades out and a new word appears on the main page.	See expected result.	
	Check response on tapping the right button.	The main page is shown with a word or some words and its translation under it. Left and right there is a red and a green area and below there are three buttons: red, orange and green.	Tap on the right green button on the bottom of the screen.	None	By tapping the right button an event is sent to the server if there is a connection. If not the event is saved so it can be sent when there is a connection. The word is placed 5 steps further after the first encounter in the wordPair so it appears again after 5 words. If the word is right again it will be placed 10 positions further in the wordPair.	After clicking the right button a green image appears with a graduation cap to notify the user about the clicked space. This image fades out and a new word appears on the main page.	See expected result.	
check menu functionality	Check response on tapping the menu space.	The main page is shown with a word or some words and its translation under it. Left and right there is a red and a green area and below there are three buttons: red, orange and green.	Tap in the middle of the space where the words are displayed.	None	The visibility of the menu page is set to visible.	After tapping the space the menu appears with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	See expected result.	
	Check response on pressing the menu button.	The main page is shown with a word or some words and its translation under it. Left and right there is a red and a green area and below there are three buttons: red, orange and green.	Tap on the middle orange button on the bottom of the screen.	None	The visibility of the menu page is set to visible.	After tapping the space the menu appears with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	See expected result.	
	Check response on pressing the "wrong translation" button.	The menu page is shown with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	Press the button with the text "wrong translation".	None	An event is sent to the server (when there is a connection), notifying the server that the current displayed word has to be erased. The word pair is always deleted from the local storage. The menu page visibility is set to hidden so the main page can be seen again.	After pressing the button the menu fades out and the main page is visible again. On the word space a red image appears with a trash can to notify the user about his or her action. When the image fades out a new word is on the main page.	See expected result.	
	Check response on pressing the "I learned it" button	The menu page is shown with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	Press the button with the text "I learned it".	None	An event is sent to the server (when there is a connection), notifying the server that the current displayed word should not be returned to the watch. The word pair is always deleted from the local storage. The menu page visibility is set to hidden so the main page can be seen again.	After pressing the button the menu fades out and the main page is visible again. On the word space a green image appears with a graduation cap to notify the user about his or her action. When the image fades out a new word is on the main page.	See expected result.	

Test Scenario	Test Case	Pre-conditions	Test Step	Test Data	Post-conditions	Expected Result	Actual Result	Pass/Fail
	Check response on pressing the "show context" button	The menu page is shown with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	1. Press the button with the text "show context". 2. Tap on the context in the word space.	None	The visibility of the menu page is set to hidden and a canvas becomes visible in which the sentence is printed in which the word was found. This canvas lies on top of the word space. After tapping the canvas the word is shown again.	The menu page fades out and the context appears on the same place were the words are normally displayed. After tapping the context, the context disappears and the word becomes visible again.	See expected result.	✓
	Check response on tapping on the menu page.	The menu page is shown with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	Tap the screen anywhere besides the other buttons.	None	The visibility of the menu page is set to hidden and thus the main page is visible again.	The menu page fades out and thus the main page is visible again.	See expected result.	✓
	Check response on tapping on the back button in menu.	The menu page is shown with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	Tap the button with the back arrow on it.	None	The visibility of the menu page is set to hidden and thus the main page is visible again.	The menu page fades out and thus the main page is visible again.	See expected result.	✓
	Check response after an screen off.	The menu page is shown with three buttons "wrong translation", "I learned it" and "show context". On the bottom of the screen the back button is displayed.	1. Wait for the watch screen to switch off or do a arm twist. 2. After a screen off press the power button or do a arm twist to turn on the screen.	None	The visibility of the menu page is set to hidden when the screen of the watch turns off and thus the main page is visible again when the watch is switched on.	The main page is visible.	See expected result.	✓
check profile functionality	Check response after pressing the left button.	The profile page is shown.	Tap the button on the left of the screen.	None	The medalPos is lowered by one. If the medalPos is zero, it will become four. And the medal is drawn on the screen.	The page icon number is updated, the previous medal is shown.	See expected result.	✓
	Check response after pressing the right button	The profile page is shown.	Tap the button the right of the screen	None	The medalPos is increased by one. If the medalPos is four, it will become zero. And the medal is drawn on the screen.	The page icon number is updated, the next medal is shown.	See expected result.	✓
	Check response after pressing the back button	The profile page is shown.	Tap the button on the bottom of the screen with the back arrow on it.	None	The visibility of the profile page is set to hidden.	The profile page fades out and the main page appears again.	See expected result.	✓
	Check response after an screen off.	The profile page is shown with two buttons on the side of the screen, a medal with information in the middle and a back button on the bottom.	1. Wait for the watch screen to switch off. 2. After a screen off press the power button to turn on the screen.	None	The visibility of the profile page is set to hidden when the watch turns off and thus the main page is visible again when the watch is switched on.	The main page is visible.	See expected result.	✓
check mainpage functionality	Check if the temperature is correct.	The main page is shown.	1. Check the current temperature at your current location. 2. Check the displayed temperature on the watch.	Temperature at the current location.	With an api the location is determined and with the location the temperature is loaded from another api.	The correct temperature is displayed on the top of the screen.	See expected result.	✓
	Check if the weather type is correct.	The main page is shown.	1. Check the current weather type at your current location. 2. Check the displayed weather type on the watch.	Weather type at the current location.	With an api the location is determined and with the location the weather type is loaded from another api.	The correct weather type is displayed on the top of the screen.	See expected result.	✓

Test Scenario	Test Case	Pre-conditions	Test Step	Test Data	Post-conditions	Expected Result	Actual Result	Pass/Fail
	Check if the date is correct.	The main page is shown.	1. Check the current date at your current location. 2. Check the displayed date on the watch.	Date at the current location.	With a function provided by tizen the date is loaded from the OS.	The correct date is displayed on the top of the screen.	See expected result.	
	Check if the time is correct.	The main page is shown.	1. Check the current time at your current location. 2. Check the displayed time on the watch.	Time at the current location.	With a function provided by tizen the time is loaded from the OS.	The time is displayed on the upper part of the screen.	See expected result.	
check popup functionality	Check if a popup appears when there are too few words left.	The main page is shown.	1. Tap the shown word to reveal the translation. 2. Tap the button in the middle on the bottom of the screen. 3. Press the button "wrong translation" or "I learned it". 4. Repeat this action until a popup appears.	None	In the app there should be at least the same amount of words as there are flashcards. When this amount is reached, a popup on top of the word space becomes visible.	The popup appears on top of the word space which disappears after 3 seconds.	See expected result.	
	Check if a popup appears when the learning streak is increased	The app is used for one day.	1. Use the app the next day. 2. Wait until the popup appears. 3. Close the popup by pressing the button on the bottom of the screen.	None	The date when the app is used, is saved to the local storage. When the app is used on another day, the date is compared to the saved date and if the difference is one day, the streak is increased. The popup canvas is not visible anymore.	A popup is shown with a medal (with glasses), the current learning streak and fireworks. The popup fades out when the button is pressed and the main page is visible.	See expected result.	
	Check if a popup appears when a new achievement is accomplished for learning session	The app is used for some minutes.	1. Use the app until the popup appears without the screen going off. 2. Close the popup by pressing the button on the bottom of the screen.	None	Longest session time is updated and saved to localStorage. The popup canvas is not visible anymore.	In the profile page the longest session will be updated.	See expected result.	
	Check if a popup appears when a new achievement is accomplished for total time	The app is used for some minutes.	1. Use the app at least 15 minutes 2. Close the popup by pressing the button on the bottom of the screen	None	The popup canvas is not visible anymore.	Popup is shown on the screen and can be closed with the button	See expected result.	
	Check if a popup appears when a new achievement is accomplished for words learned	No words are currently learned	1. Tap the menu button on the revealedPage 2. Tap 'I learned it' button. 3. Close the popup by pressing the button on the bottom of the screen	None	wordsLearned is updated and saved to the localStorage. The popup canvas is not visible anymore.	Popup is shown on the screen with the message 'you learned your first word!'. In profile words learned is now 1.	See expected result.	

# B

## Questionnaire

The questionnaire asked the following questions to the participants:

### **B.1 General information**

1. What is your age?
2. How many times a day are you checking the time?
3. If you are learning new words, how much time do you spend daily on learning?
4. What is your native language?
5. In which language have you learned the words on the app?
6. What is your current language level in this language that you are learning?
  - A1 - Beginner. Can recognize and use simple phrases.
  - A2 - Elementary. Using simple words, can describe his or her surroundings and communicate immediate needs.
  - B1 - Intermediate. Can understand the main points of clear standard speech. Can narrate an event, an experience or a dream.

- B2 - Upper Intermediate. Can speak in a clear, detailed way on a number of subjects; express an opinion on current affairs, giving the advantages and disadvantages of the various options.
  - C1 - Advanced. Can use the language effectively and fluently in a social, professional or academic context.
  - C2 - Master. Can express him or herself precisely in a spontaneous, fluent way, conveying finer shades of meaning precisely.
7. For how many years have you been studying this language?
  8. How much time do/did you spend daily on learning before the app?
  9. How much words do/did you learn daily?
  10. In what way are you learning the language? (multiple choices)
    - I'm not learning it yet, but I will.
    - Reading texts in the other language.
    - Using a textbook.
    - Talking/chatting with other people speaking the foreign language.
    - Apps for the smartphone.
    - Other:...

## **B.2 Questions after using the smartwatch app**

1. What do you think about the design? (e.g., Were the buttons large enough, was the text readable etc.)
2. Was it clear how the app should be used? If not, why?
3. Some people did not use all of the features: show context, I learned it, wrong translation, reverse and profile. If you also didn't, could you let us know why?
4. What didn't you like about the app?
5. Which features did you miss that would have been useful?
6. Which features did you like?
7. What did you think about the medals with the achievements in the app?

8. When did you use the app the most? (e.g., waiting for the bus)
9. When you didn't know a word, did you reveal the translation instantly or did you take time to think about the answer?
10. Did you sometimes press right when you actually didn't really know the word? If so, what was the reason?
11. What score would you give the TimeToLearn app from 1 to 10?
12. Would you recommend the TimeToLearn app to your friends?
  - Yes
  - No

# C

## Questionnaire answers

In this appendix the answers of the users are included, in the user study we used numbers to separate the users. The same numbers are used here to separate the user.

### C.1 General information

1. What is your age?
  - (i) 24
  - (ii) 22
  - (iii) 26
  - (iv) 34
2. How many times a day are you checking the time?
  - (i) It depends if I am waiting or not. Moreover, I do it subconsciously. I would say: About 10-15 times
  - (ii) 24 approximately
  - (iii) 7

- (iv) 10
- 3. If you are learning new words, how much time do you spend daily on learning?
  - (i) Once a day, maybe 10 minutes
  - (ii) 1 hour
  - (iii) 10 min
  - (iv) not much...
- 4. What is your native language?
  - (i) Dutch
  - (ii) Dutch
  - (iii) Romanian
  - (iv) Romanian
- 5. In which language have you learned the words on the app?
  - (i) German
  - (ii) German
  - (iii) Danish
  - (iv) German
- 6. What is your current language level in this language that you are learning?
  - (i) C2
  - (ii) B1
  - (iii) A2
  - (iv) B1
- 7. For how many years have you been studying this language?
  - (i) The first time having it in school was 13 years ago
  - (ii) 6 (only at school)
  - (iii) 1
  - (iv) 3
- 8. How much time do/did you spend daily on learning before the app?



- (i) None
  - (ii) none
  - (iii) Almost at all
  - (iv) not learning daily...
9. How much words do/did you learn daily?
- (i) I picked about 5-6 words a day
  - (ii) none
  - (iii) Around 14
  - (iv) i didn't put time aside daily...
10. In what way are you learning the language? (multiple choices)
- (i)
    - Reading texts in the other language
    - Talking/chatting with other people speaking the foreign language.
  - (ii) I am not learning it yet, but I will.
  - (iii)
    - Reading texts in the other language
    - Using a textbook
    - Talking/chatting with other people speaking the foreign language.
  - (iv) Reading texts in the other language.

## **C.2 Questions after using the smartwatch app**

1. What do you think about the design? (e.g., Were the buttons large enough, was the text readable etc.)
- (i) The design was great!
  - (ii) The design was alright, it was excellently readable.
  - (iii) The design was overall ok, it is simple and easy to use. You need to get closer to read the word in the sentence but that's an extra function.
  - (iv) context was not large enough i was missing an "i don't want to learn this word" option.

2. Was it clear how the app should be used? If not, why?
  - (i) It didn't come naturally to me that there were more options in the app, like "I have learned the word". So I got the same words over and over, that I already knew.
  - (ii) It was clear from the instructions which I obtained from Niels, but to do it myself I think it would be confusing
  - (iii) Yes, it was quite clear. I was not sure when the words will be removed from the list and that's why I used the function 'I know the word, don't show it again'
  - (iv) clear for me
3. Some people did not use all of the features: show context, I learned it, wrong translation, reverse and profile. If you also didn't, could you let us know why?
  - (i) Because I didn't know the app had these features. A small tutorial might have done the job? Or a message "You haven't used all features of this app. Did you know that..."
  - (ii) I only used these features once. I have no specific reason why I didn't use them, I forgot they existed sometimes and other times I just wanted a quick check if the translation I had in mind was correct.
  - (iii) I didn't use wrong translation because I was not sure about the translation, and didn't have time to check the word in another dictionary. It's still a very new language for me
  - (iv) reverse is not easy to find not always clear if a translation is wrong...
4. What didn't you like about the app?
  - (i) The translations weren't always correct in the context, as I sometimes already knew what it meant. It does have a option to choose a different translation, but a person learning a new language wouldn't know the exact word. That's just a problem with translation sites though and not the app itself. During testing the app screen froze like 2-3 times and sometimes reacted a bit slow.
  - (ii) None.
  - (iii) The fact that I was not sure if it is the right translation and also, that I couldn't hear the pronunciation
  - (iv) nothing
5. Which features did you miss that would have been useful?

- (i) As explained earlier: A feature to encourage you to fully use the app or a small tutorial, as I was new to the app.
  - (ii) Maybe some feature to learn the pronunciation as well.
  - (iii) pronunciation
  - (iv) i would like to be able to see the context before the answer, maybe the context would help me remember the answer
6. Which features did you like?
- (i) Achievements and the general looks
  - (ii) The context feature is really helpful in my opinion although I didn't use it much.
  - (iii) The 'show the word in the context' button
  - (iv) like being able to learn vocabulary in elevators
7. What did you think about the medals with the achievements in the app?
- (i) Great!
  - (ii) I did not pay much attention to this.
  - (iii) Were fun at the beginning but would not be so interesting on long term
  - (iv) Yes (I liked it)
8. When did you use the app the most? (e.g., waiting for the bus)
- (i) Waiting, before going to sleep and out of boredom
  - (ii) When I was doing nothing at all or waiting, mostly in the evenings while watching tv and so on.
  - (iii) Waiting for the green light while biking
  - (iv) when waiting for people, or things (e.g. elevator)
9. When you didn't know a word, did you reveal the translation instantly or did you take time to think about the answer?
- (i) Most of the time instantly. I was impatient and curious to know what it meant.
  - (ii) sometimes I accidentally hit reveal translation but mostly I tried to think about the correct translation.
  - (iii) Waited few seconds but not very long

- (iv) i would reveal the answer if i didn't know the word
10. Did you sometimes press right when you actually didn't really know the word? If so, what was the reason?
- (i) No, perhaps a miss click
  - (ii) Yes sometimes, only by accident.
  - (iii) Seldom, and if so, because I believe the word it's not so important
  - (iv) maybe... so i can feel good about myself
11. What score would you give the TimeToLearn app from 1 to 10?
- (i) No, perhaps a miss click
  - (ii) 9
  - (iii) 7 (maybe develop it for smartphones, I don't like having a watch, would like to have also pronunciation and a very accurate translation)
  - (iv) 8/10
12. Would you recommend the TimeToLearn app to your friends?
- (i) Yes
  - (ii) Yes
  - (iii) Yes
  - (iv) Yes

# D

## Script to Obtain Different Session Lengths

In this appendix the script is included, which was used to obtain the different session lengths for our research described in the usage study chapter. The code is written in JavaScript again.

```
1 var USER_CODE = 73140814;
2 var ENDPOINT = "https://zeeguu.unibe.ch/get_smartwatch_events";
3 var events;
4 var numberOfOccurrences;
5
6 function length(obj) {
7     return Object.keys(obj).length;
8 }
9
10 function getEvents() {
11     var xhr = new XMLHttpRequest();
12     xhr.open('GET', ENDPOINT + "?session=" + USER_CODE, false);
13     xhr.onload = function () {
14         events = JSON.parse(this.responseText);
15     };
16     xhr.send();
17 }
18
19 function removeDuplicates() {
```

```
20  var count = 0;
21  for (var i=0; i+1 < length(events); i++) {
22      if (events[i].event === events[i+1].event && events[i].time ===
          events[i+1].time && events[i].bookmark_id === events[i+1].
          bookmark_id) {
23          events.splice(i, 1);
24          i--;
25          count++;
26      }
27  }
28  console.log("number of duplicates: " + count);
29  }
30
31  function removeScreenOnFollowedByScreenOff() {
32      var count = 0;
33      for (var i=0; i+1 < length(events); i++) {
34          if (events[i].event === "screenOn" && events[i+1].event === "
              screenOff") {
35              events.splice(i, 2);
36              i--;
37              count++;
38          }
39      }
40      console.log("number of screenOn followed immediately by screenOff
          removed: " + count);
41  }
42
43  function getDateObject(timeString) {
44      year = timeString.substr(0, 4);
45      month = timeString.substr(5, 2);
46      day = timeString.substr(8, 2);
47      hours = timeString.substr(11, 2);
48      minutes = timeString.substr(14, 2);
49      seconds = timeString.substr(17, 2);
50
51      return new Date(year, month, day, hours, minutes, seconds, 0);
52  }
53
54  function countDifferentSessionLengths() {
55      var count = 0;
56      var screenOn = -1, screenOff = -1;
57      var timeScreenOn, timeScreenOff, difference;
58      var t5 = 0, t15 = 0, t60 = 0, t_other = 0;
59
60
61      for (var i=0; i+1 < length(events); i++) {
62          if (events[i].event === "screenOn") {
63              screenOn = i;
64          } else if (events[i].event === "screenOff" && screenOn !== -1) {
```

```
65     screenOff = i;
66
67     timeScreenOn = getDateObject(events[screenOn].time);
68     timeScreenOff = getDateObject(events[screenOff].time);
69     difference = (timeScreenOff - timeScreenOn)/1000;
70
71     if (difference <= 5) {
72         t5++;
73     } else if (difference <= 15) {
74         t15++;
75     } else if (difference <= 60) {
76         t60++;
77     } else {
78         // > 60
79         t_other++;
80     }
81
82     //reset
83     screenOn = -1;
84     screenOff = -1;
85 }
86 }
87 console.log("t <= 5: " + t5);
88 console.log("5 < t <= 15: " + t15);
89 console.log("15 < t <= 60: " + t60);
90 console.log("t > 60: " + t_other);
91 }
92
93 getEvents();
94 console.log("total number of events: " + length(events));
95 removeDuplicates();
96 console.log("total number of events after removing duplicates: " +
    length(events));
97 removeScreenOnFollowedByScreenOff();
98 countDifferentSessionLengths();
```

Listing 10: script to get different session lengths

# Bibliography

- [1] JP Anderson and AM Jordan. Learning and retention of latin words and phrases. *Journal of Educational Psychology*, 19(7):485, 1928.
- [2] David Dearman and Khai Truong. Evaluating the implicit acquisition of second language vocabulary using a live wallpaper. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1391–1400. ACM, 2012.
- [3] Pascal Giehl, Oscar Nierstrasz, and Mircea Lungu. Zeeguu translate application. *University of Berne*, 2015.
- [4] Seul-Kee Kim, So-Yeong Kim, and Hang-Bong Kang. An analysis of the effects of smartphone push notifications on task performance with regard to smartphone overuse using erp. *Computational Intelligence and Neuroscience*, 2016, 2016.
- [5] Mircea Lungu, Karan Sethi, Simon Marti, and Linus Schwab. The Zeeguu API - Modeling Learner Progress to Accelerate Vocabulary Acquisition, July 2016.
- [6] Simon Marti. A platform for second language acquisition through free reading and repetition. *University of Berne*, 2013.
- [7] Ian SP Nation. *Learning vocabulary in another language*. Ernst Klett Sprachen, 2001.
- [8] IS Paul Nation. Beginning to learn foreign vocabulary: A review of the research. *RELC journal*, 13(1):14–36, 1982.
- [9] Jorrit Oosterhof, Mircea Lungu, and George Digkas. Making reading in a second language more enjoyable. *University of Groningen*, 2015.
- [10] Dipesh Pradhan and Nugroho Sujatmiko. Can smartwatch help users save time by making processes efficient and easier. *Master’s thesis. University of Oslo*, 18, 2014.
- [11] Scott Thornbury. *How to teach vocabulary*. Pearson Education India, 2006.
- [12] H Van Rijn and M Nijboer. Optimaal feiten leren met ict. *Weten Wat Werkt En Waarom 4W*, 2012.