Re-Design of a Serious Game for eHealth

By Jordi Weldink
S1731912

Bachelor Thesis- Creative Technology, University of Twente
6 juni 2018

Client: Roessingh Research and Development (RRD)
Supervisors: Dr. J. Zwiers, M. Broekhuis & S. ter Stal.

Abstract

A high fidelity re-design of the serious game ‘Stranded’ was created based on usability evaluations with older adults(55-65 years). The game is designed to be a motivational tool for this target group to perform their physical exercises on a regular basis. This project is to about researching how the usability of this game can be improved by creating a re-design. The UCD- and Creative Technology design process were combined and applied to accomplish this goal. The project consist of two phases which form the iteration cycle for this project: 1) Usability evaluation and analysis and 2) Create a re-design based on results and literature. Usability evaluations were conducted with 10 older adults to find out what issues respondents encounter while playing the game. It was found that the user interface caused most issues, especially with wayfinding and misinterpretation of information. Recommendations were made to solve these issues by conducting an expert review, then they were translated into design solutions. This translation was done by setting prerequisites/limits and validation by literature. The re-design was realized in Unity2D from scratch by implementing the design solutions. It was an iterative process which led to a re-design capable of representing the original game.

Acknowledgements

I wish to thank my supervisors for their contribution and guidance throughout this project; Dr. J. Zwiers for his valuable ideas, useful critiques and feedback; S. ter Stal for her valuable ideas and feedback in the process of realizing a re-design of this game.

Special thanks should be given to M.Broekhuis for her useful help, advice and assistance offered throughout this project, especially her role during the evaluation and analysis of usability for this research was a big help.

I would also like to extend my thanks to the RRD and everyone else involved; The RRD organization for providing me with a place to work and enabling my research by providing me access to proper equipment; Everyone who helped me with collecting data by either recruiting people or participating voluntarily in the usability evaluations and playtesting themselves. Also the experts who participated voluntarily in the expert review to discuss negative usability issues.

Table of Contents

1. Introduction

The Roessingh Research and Development centre (RRD)[1] has created a tool that supports a physical activity program. RRD developed a serious game for eHealth called Stranded to motivate older adults (55-65 years old) to increase their physical activities at home in a fun and enjoyable way. A concept where the game is integrated with the serious purpose of improving health by using internet resources. These resources enable the user to take active control of their well-being at home. It has much potential in the field of preventing declination caused by ageing, however it can only be a viable motivation tool when the user needs are fulfilled (Wiemeyer, J. & Kliem, A., 2012), which is where user experience (UX) comes into play. UX is the overall experience of the target group while playing the game. The game needs a good user experience design (UXD) to fulfil the user needs, therefore it is required to know and understand the potential users of this game. By having a good UX in the game, it has more chance of achieving its ultimate goal.

The choice has been made to improve the usability of the Stranded, because the existing game is currently being evaluated on usability with older adults and other target groups. It is part of an bigger project called IMI-Sprint. Improving the usability of the game contributes to a better UX. The term usability for this project is defined as follows*:* the extent to which Stranded can be used by older adults to increase their physical activity with effectiveness, efficiency and satisfaction by regularly playing the game at home (Jokela et al., 2003). Usability evaluations is one of the disciplines used to design good user experiences and to see what users currently think of the game. During some evaluations for example it became quickly apparent that lots of usability problems are caused by the user interface (UI). A user interface or abbreviated as UI are the means that allow the user and game to interact with each other like e.g. buttons, icons, text navigation. The choice has therefore been made to definitely do a re-design of the UI. The UXD can be improved by enhancing usability, accessibility and pleasure. Although the choice is already made to improve usability, an often mentioned constraint is the lack of proper usability. Having proper usability should resolves almost any problem caused by the system and the mentioned user interface (Nap, H.H., de kort, Y.A.W., & Ijsselsteijn, W.A.,2009, Johnson, R. & Kent, S.,2007).

The focus of this project is to improve the usability of the game by means of an improved design based on results and literature. The user-centered design process or abbreviated as UCD (Jokela et al.,2003) is the method applied to accomplish this. It is however applied in an unusual way since an existing game is the starting point for this project. The UCD process starts at evaluating and analysing the usability of the game with the target group and then to improve its usability by creating a re-design based on results and literature, which are consecutively steps 4 and 3 in the UCD process that form an iteration cycle as can be seen in *Figure 1*. To achieve the ultimate goal of the game, proper usability is necessary. The game needs to be convenient to use and easy to learn. Therefore it is necessary to evaluate the current usability of the game by conducting tests with the target group. The RRD has made a usability protocol *(A1 – Usability protocol)* to evaluate the user-centred design of this game. The predefined protocol uses both qualitative and quantitative methods to collect the desired data from the respondents, which are explained further in *chapter 3*. Once all testing is done, all results collected from evaluations are processed and then analysed to come up with recommendations to re-design the game. These recommendations are then compared with what is known in the literature. In the end they are used as design solutions, which are communicated by a visual mock-up that represents the re-design of this game. This leads to the following research questions for this project:

[1] http://www.rrd.nl/en

- How to enhance/improve the usability of the game Stranded for older adults?
  - How to analyse and organise usability findings?
  - How to effectively communicate design solutions by using a mock-up?

To answer these research questions, this project is separated in the following two phases: 1) Evaluating and analysing the usability of Stranded with the target group and 2) Create an improved design based on results and literature. Before phase 1 starts, a state-of-the-art research is executed to gain more knowledge and insights on how current thinking is at the fields of user interfaces and translating usability findings into design solutions. This project uses the UCD model (Jokela et al.,2003) and the Creative Technology design process (Mader, A., & Eggink, W., 2014). Both models are used in a unconventional way because usability evaluations of an existing game is the starting point for this project.

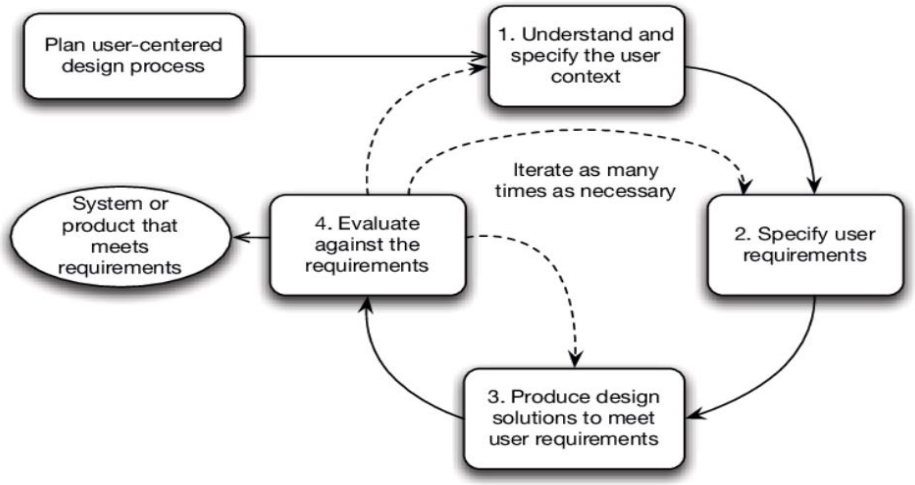


Figure 1. The user-centered design process based on ISO 9241-210:2010 [15, p.11].

Phase 1 and 2 together form an iteration round in the UCD model by using the route via step 3 and 4 like mentioned before. The same phases as described in the Creative Technology design process are used, but in an different order. Evaluation in phase 1 and ideation, specification and realisation consecutively in phase 2.

Phase 1 describes how usability evaluation were conducted, how data from evaluations was analysed and what the found results are. The results were used as input to conduct an expert review, which yield output in the form of recommendations for re-design.

Phase 2 describes how the re-design is made by executing ideation, specification and realisation phases. This triplet of phases turned recommendations into design solutions, which eventually were implemented into the final re-design.

In closing, there is a conclusion/discussion chapter and afterwards a future work chapter, which discusses what the potential next steps of this project could be.

2. State-of-the-art on user interface and translating usability findings into design solutions

State-of-the-art review is done to get insights on the current thinking in the field, recent developments and methodologies relevant for this project. It can be helpful to gain new perspectives on solving the research question this graduation project wants to answer. The decision is made to re-design the user interface as mentioned in the *Introduction*, that is why this research looks into user interface design for older adults. Furthermore this project is about 2 major phases mentioned in the *introduction*, therefore the state-of-the-art research investigates those 2 phases as well. It is already clear how usability evaluations are conducted, so state-of-the-art looks at the stages after that. The stages after that are about translating usability findings into design solutions for the re-design.

2.1 User interface

Current user interface is a combination of a graphics and textual information, better known as Graphical User Interface(GUI) and Textual User Interface(TUI) respectively. A user interface (UI) acts and reacts on what users want to do and guarantee its elements are accessible, understandable, and allow necessary actions to be performed. The user interacts with a GUI through graphical icons and visual indicators, but also makes use of detailed textual navigation or TUI, as shown in *Figure 2*. The following knowledge is needed in order to design a suitable UI for older adults: 1) what older adults need to do in the game, 2) how current UI elements are used/ interpreted and 3) what older adults need in an user interface. 1) and 2) are answered after *phase 1: Usability evaluation and analysis of Stranded* and 3) is further researched in this paragraph.

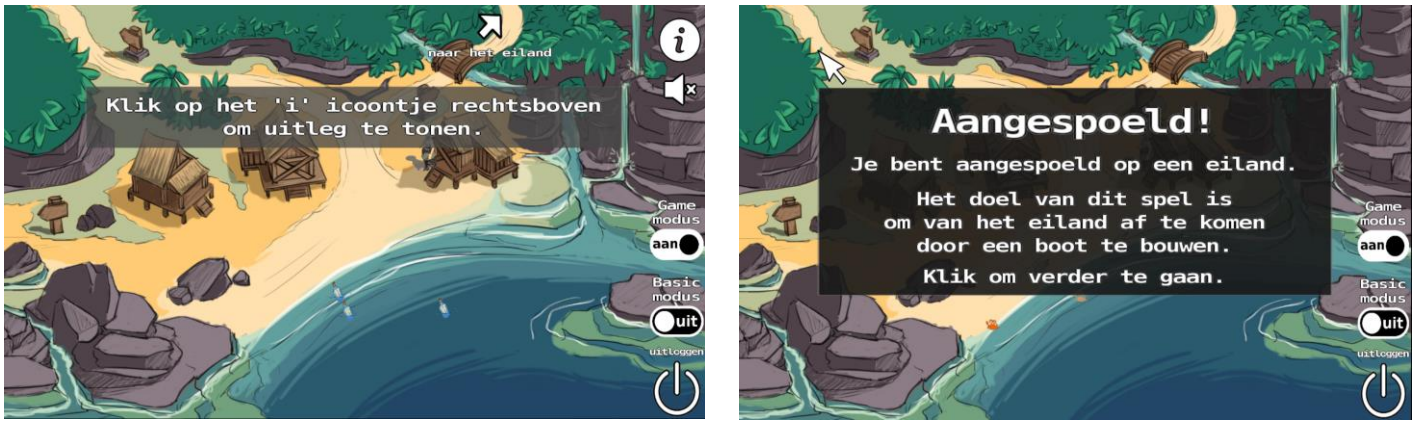


Figure 2: User interface of the game Stranded.

Designing user interfaces (UI) for older adults is about taking their age-related changes into account. Visual, physical or cognitive impairments are the age-related changes that need to be taken into account while designing games for older adults (Johnson, R. & Kent, S.,2007, Nap, H.H., de kort, Y.A.W. & ijsselsteijn, W.A. 2009, W, I. J., Nap, H. H., De Kort, Y., & Poels, K.,2007). W, I. J., Nap, H. H., De Kort, Y., & Poels, K.(2007) and Gerling, K. M., Schulte, F. P., Smeddinck, J., & Masuch, M. (2012) state that visuals in the game need distinguishable elements by having proper contrast and carrying readable information. Gerling, K. M., Schulte, F. P., Smeddinck, J., & Masuch, M. (2012) argues that user preferences need acknowledgement by giving the control to the user when it comes to changing fonts, colors and window size. Physically it should be easy for the user to interact with the game by using few simple operations only, for instance undo actions by a single mouse click. The cognitive

processes and memory load of the user should be kept low as possible. Johnson, R. & Kent, S.(2007) state that older adults have the tendency to read instructions first, but one should not display big chunks of information at once. The UI should provide sufficient information in order to be able to interact with the game. However, Gerling, K. M., Schulte, F. P., Smeddinck, J., & Masuch, M. (2012) do agree, but argues that core mechanics in the game need to be simple and easy to learn to reduce cognitive load. Further they noted that older adults do not have prior gaming experiences in general, therefore should avoid complex functionality and embrace simple elements and rules like casual games (e.g. Dr Kawashima's Brain Training for Nintendo DS). These game have simple rules, no steep learning curve and suitable gaming speed. The target group does not need specific demands for the UI, mentioned generic solutions are suitable enough. There is an agreement on the fact that user interfaces play a vital role in games and that user interface design principles should be used create a good simple basic UI that provides clear and sufficient information and easy/simple core elements that takes cognitive load of the user into account (e.g. UI consistency and fault tolerance).

2.2 Translating usability issues into design solutions

This project translates usability findings into design solutions, which are then applied in a re-design. The steps in this process are to turn usability findings into recommendations, comparing recommendations with literature and transform recommendations into working design solutions for the re-design. This paragraphs researches what methods in literature are used to translate usability issues into design solutions. The goal is to learn from this research and apply this knowledge into this project approach. Recent literature on translating usability issues into design solutions were researched and global steps were extracted. The extracted steps were analysis of usability findings, extracting useful and usable recommendations and communication of findings by mock-up. Which is somewhat similar to the steps used in this project, but not entirely. The steps are individually discussed in the following paragraphs.

2.3 Analysis

The analysis is done to process all the data from usability evaluations in order to extract valid and meaningful findings. The usability tests produced raw materials that cannot be immediately used to yield outcomes for the new version of the game. They need to be processed first in a useful data format. Data can either be qualitative or quantitative. Qualitative data is all recorded audio from usability tests and demographics. Quantitative data is the data gained from all questionnaires. All data is processed into an accessible spreadsheet where data is arranged by categories or other metrics. This is true for both qualitative and quantitative data, but there is more to qualitative data than just converting it into a spreadsheet. Usability issues have to be extracted and categorized by severity after conversion. Also it is good to analyse the errors that cause the issue, learning more about its nature and underlaying reason (Davis and Douglas.,2015). They state that usability findings often are categorized by severity and/or frequency in a three-point scale:

- Critical: Task cannot be completed without solving this issue.
- Serious/warning: Task is likely not be completed, frustrations and loops often occur.
- Minor/annoyance: Task can be completed, however users are going to be annoyed.

Severity ratings are always presented in combinations with frequency of occurrence. For instance some critical issue X occurring 1 time, does not have priority over a minor issue Y occurring 20 times. Priority of issues is determined by both severity and frequency of occurrence.

2.4 Useful and usable recommendations

Recommending a solution for a usability issue is a good habit, however it should be useful and usable it is to ensure its quality. Molich, Jeffries and Dumas (2007) describe useful and usable recommendations as “recommendations for solving usability problems that lead to changes that efficiently improve the usability of a product”. They also describe usefulness rating scale and usability rating scale both from 1-5. The highest rated or perfect recommendation is effectively describing the perfect solution to a usability problem in precise and reasonable detail such that the product team immediately knows what to do to implement that solution. This would be the ideal scenario of communicating recommendations. However, the chance of a solution not having any drawbacks or introducing new usability problems is very difficult. Writing recommendations should be written as requirements for a programmer, that he or she understands what and why to implement something in the re-design.

2.5 Communicate findings

This project is executed for a client, it is good to determine how to visually communicate useful and usable recommendations to them. It is harder to understand recommendations from their description, than by showing a visual representation containing recommendations. Especially when describing interactions and their reasoning. Davis and Douglas (2015) agrees by stating that besides writing a report, whenever possible one should present data in another way as well, preferably visually. This can be accomplished by using videos, mock-ups, comments from usability testing, charts, graphs and many more. This visual way of representing results does a better job at illustrating the overall usability of a project. As the saying goes, a picture is worth more than a 1000 words. The choice has been made to create a functional and interactive prototype, or a high fidelity prototype. These prototypes have the ability to allow user interactions and can also be tested with real users in the end. To find out if the working design solutions in the re-design are working as intended.

2.6 Conclusion

State-of-the-art review on user interface and translating usability findings into design solutions was done to gain more insights on the current thinking in the field, recent developments and methodologies relevant for this project. User interface design principles should be used create a good simple basic UI that provides clear and sufficient information and easy/simple core elements that takes cognitive load of the user into account. The first step in translating usability findings into design solutions is conducting an analysis to process all the data from usability evaluations to extract usability issues categorized and prioritized by severity and/or frequency of occurrence. Results from the analysis are used to write recommendations. These should be written in a clear, concise way, as if they were requirements for a programmer, that he or she understands what and why to implement something in the re-design. The final re-design is a high-fidelity prototype that has the ability to allow user interactions and can also be tested with real users. In the end the working design solutions in the re-design are test to check if they work as intended. The research questions of this project are novel, because the existing game Stranded is exposed to a new target group of older adults (55-65 years old), also because of its unusual application of UCD process in combination with Creative technology design process.

Phase 1: Usability evaluation and analysis of Stranded

Phase 1 contains the usability evaluations and analysis of the game Stranded. Respondents were recruited for usability evaluations. These were conducted by using the procedure described in the usability protocol (A1 – Usability protocol). The evaluations resulted in a pile of raw data. By performing an analysis this data became useful and negative usability issues could be extracted from it. These issues were then used as fuel to conduct an expert review, which resulted into recommendations and ideas for the re-design.

Chapter 3 describes the game, the methods from usability evaluations, the analysis and the extracted results. In the end closure is given by a conclusion. First of all method describes the methods used for evaluation, analysis and expert review. Secondly, the results from analysis and expert review are discussed and finally a conclusion is given.

3. Evaluation and analysis of usability Stranded

Before getting into aspects of evaluations and analysis, it is important to understand what they are and why they are executed in the first place. Usability evaluations focuses on to what degree the target group is able to learn and utilise the game in order to perform their physical exercises on a regular basis (usability.gov). This evaluation uses combination of are a standard usability test, a think-aloud protocol, surveys and an interview. After the evaluations, analysis is done to transform evaluation data into useful data from which the usability performance and negative usability issues can be extracted.

3.1 The game

The name of the serious game is 'Stranded', as the name suggest the user is lost somewhere. In this case lost on an uninhabited island far from civilization. The goal is to escape the island by boat, the user has to explore the island to find materials in order to build it. By performing exercises one can progress by unlocking different new elements like for instance building materials or other rewards.

3.2 Method

This paragraph describes the methods used for usability evaluations/ analysis and expert review. The usability evaluations are conducted by following the procedure of the existing usability protocol (*A1 – usability protocol*) step by step. It also contains the used setup for evaluations and the recruitment process of respondents.

3.2.1 Usability protocol

The predefined procedure written in the usability protocol (A1-usability protocol) is followed to conduct usability evaluations with. Stranded is evaluated on usability for different target groups. This protocol is used to ensure that all research on the game Stranded is done in the same way. The steps in usability evaluations are discussed in chronological order. A signed consent form is legally needed first, else the evaluation cannot start at all. This form is an informative letter about the research on using a computer game to improve physical health, what the outline of the evaluation looks like and that audio is recorded during the evaluation. Also it is mentioned that respondents stay anonymous throughout the research and have the right to quit. Whenever the respondent feels like quitting, it can do without having to submit any reason for that. After having a signed consent, the evaluation can start. The chronological order of steps conducted in the evaluation are: collecting demographics, motivation questionnaire, think-aloud protocol, system usability scale questionnaire (SUS), post-assessment interview. Each step is explained in its own paragraph.

Collecting demographics

Each respondent is asked about their demographic data. These demographics are used in a different study. The following demographics variables are collected: Gender, date of birth, education, technology usage and technology preference. Technology usage is about the systems people use at home (e.g. PC, laptop, tablet, smartphone, game computer), and technology preference about what systems respondent use when playing games. Also it is asked which games they like to play on those systems.

Motivation questionnaire

The motivation questionnaire is used to find out to what degree the target group is motivated to live healthy. The questionnaire is created by adjusting the existing motivation Scale II (SMS II) to make it suitable for this project. This is not treated into more detail.

Think-aloud protocol

Respondents execute a total of 5 tasks in the game and are deemed to loudly speak out their thoughts. In this way insights are obtained about considerations and decisions made in the game. There is a practise task to familiarize respondents with this way of working. The first task is unique, since it is split into a login part and a exploration part. The exploration part can only be completed when the time limit is reached. Here below is an overview of tasks is provided:

- Practise task: Search the timetable of a train ride between Enschede and Hengelo at the given time.
- Task 1a: Login with the given credentials.
- Task 1b: Explore the island for 5 minutes.
- Task 2: A new exercise is ready, perform this exercise.
- Task 3: Check if you have received a new message.
- Task 4: Check how many ingredients are needed to prepare the following meal: Boiled potatoes with crab.
- Task 5: Play a level of the game Riverbank (Rivieroever).

Each task has a time limit of 5 minutes and afterwards its own after scenario questionnaire as can be seen in A1-usability protocol page containing “Taak 1: Vragenlijst”. These questions are…. And are used to measure the satisfaction per task. The task is stopped when respondents show evidence of frustration or irritation and can also stop the task itself if wished for.

System Usability Scale

The System Usability Scale (SUS) is a low-cost tool to measure the global usability of the game. Brooke, J. (1996) describes its tool as a “quick and dirty” way to assess a systems usability. The tool he made is a ten-item questionnaire which can be answered in by Likert scale(1-5) ranging from ‘strongly agree’ to ‘strongly disagree’. The ten-item questionnaire is adjusted to make it suitable for this game and can be seen in appendix A1-Usability Protocol with the title “SUS vragenlijst”. Scores from the SUS are not used in this project.

Post-assessment interview

The respondent is asked a final set of three questions about their perception of the game, since they have played it. Perceived benefits, perceived task-technology fit (TTF) and perceived intention to use can be extracted from the answers upon these questions. The final set of questions consist of:

1. What do you think are the advantages of using Stranded?

2. Do you think that your physical wellbeing can be improved by using Stranded?
3. Do you think that you would use Stranded to improve your physical wellbeing?

3.2.2 Recruitment Respondents

An amount of 10 respondents were successfully recruited thanks to people from RRD and inner circles. Certain conditions needed to be met for participation. Respondents need to be within an age range of 55-65 years old (originating from 1953-1963), speak Dutch fluently, be able to properly use a mouse and do not have any serious health problems. These problems can prevent the respondent from successfully finishing the evaluation or cause harm during the evaluations, no unnecessary risk is taken.

3.2.3 Setup

Usability evaluations are always executed with the same setup, only the location may differ per evaluation. The RRD provides a laptop with reference to protection of personal data. This laptop uses audio recording software called CamStudio[2], which enables screen and sound capture. A microphone points towards the respondent in order to capture good quality sound. Moreover in order to play the game a mouse is required to enable interaction and a internet connection to start the web based game in the Firefox web browser. Finally, the location does not really matter. However there should be not too much disturbance and the respondent should be properly hearable.

3.2.4 Analysis

The analysis is done to process all the data from usability evaluations in order to extract valid and meaningful findings. The analysis is already discussed for the most part in *chapter 2: State-of-the-art research.* As mentioned there usability tests produced raw materials that cannot be immediately used to yield outcomes for the new version of the game. They need to be processed first in a useful data format. All qualitative and quantitative data is processed, but only the analysis of think-aloud protocol data is discussed. The other data is not relevant in the process of creating a re-design based on results and literature. The think-aloud protocol data was analysed by using the following methods: transcription, register respondent actions and extraction of negative usability issues. Each method is described here below in its own paragraph. All methods were placed in a predefined spreadsheet template as can be seen in *Table 1*, also all work was double checked by a second coder that made sure it is done properly and bias is prevented.

Table 1: Predefined spreadsheet template

Respondent xx	**Locations**		**Actions**	**Transcript**	**Action-errors**	**Transcript-errors**
Task 1	ID	Name	(…)	(…)	(…)	(…)

Transcription

The process of transcribing is to transform the data of audio-recordings into a written version. Every word said or done is written down in a text file by using a movie script format. This format separates the text from researcher and respondent in a clear way.

[2] http://camstudio.org/

Register respondent actions

All actions from the respondent during the execution of a task(except practise task) as described in the *think-aloud protocol* are registered in a spreadsheet and written down as precise as possible.

Extraction of negative usability issues

Negative usability issues are descriptions of issues prioritized by severity and frequency of occurrence and extracted by looking at the action-errors and transcript-errors. As mentioned in the *state-of-the-art research* it is good to analyse the errors that cause the issue, learning more about its nature and underlaying reason. That is exactly what has been done by looking at action- and transcript errors. Action-errors are mistakes respondents make in their actions while executing the tasks. Mistakes in this case are actions that do not have any contribution in successfully finishing the task. Transcript-errors are remarkable or wrong quotes about the game. these errors are difficult to determine, because it is the coder's interpretation of the transcript. Furthermore issues are categorized by the following severity in three-point scale:

- Critical: Task cannot be completed without solving this issue.
- Serious: Task is likely not be completed, frustrations and loops often occur.
- Minor: Task can be completed, however users are going to be annoyed.

3.2.5 Expert review

Expert review is the method used in this project to translate negative usability issues into recommendations for the game with the help of experts. This review is conducted to relive the user experience of Stranded based on what respondents encountered during their evaluations. Sauli, L. (2006) states that expert evaluations provide novel and useful data for game development, so doing another evaluations is not redundant. Normally speaking this method a usability-inspection method with which (often independent) UX experts inspect the game Stranded, however different in this case. The review existed out of a PowerPoint presentation with the following components: 1) Discussing usability issues per location, 2) Providing information and 3) Discuss pen-and-paper prototype. These components were discussed with the entire group at once. Every expert got an overview of issues (A2- negative usability findings).

1) Discussing Usability issues per location

Every participating expert got a list containing all negative usability issues including location, occasion and accompanying severity rating. These ratings are explained in the analysis. For now it is important to know that the list only contains 'serious' and 'critical' issues per location. The severity 'serious' means that the respondent is delayed by such disturbance/irritation, but eventually finished the task. The severity 'Critical' means that the issue detained the respondent from completing the task at all.

2) Providing information

Providing information is important part of this review, because from evaluation results it became clear that information buttons are popular components of the game. Lot of respondents seek for helping information time after time, which is an issue. These issues are covered in the *Discussing usability issues per location,* However the location Cabins (Hutten) deserved special attention. This locations forms the basis of the most frequent issues registered during usability evaluations.

3) Pen-and-paper prototype

Paper can be a fast and easy way to communicate new demands visually by sketching them. All locations, except the Boat(Boot) and Juttershut, got a paper template that contains the basic

structure of that location with a blank user interface. One template is already used to check out some basic ideas, which are discussed in chapter 4.2.1. This one is also shown in this review for some feedback. The templates are also meant to be used by experts to draw upon if it is easier to communicate their ideas or suggestions.

3.3 Results

Usability evaluation data is transformed into a list of negative usability issues by performing an analysis and expert review. The analysis processed evaluation data into a useful data format, which was then used to extract negative usability issues. From serious and critical issues it became clear that the re-design should focus on wayfinding and providing information. Negative usability issues were used to conduct an expert review with. This section is not including all results from data evaluations, because they are either confidential or not relevant in the process of creating a re-design (like mentioned in *3.2.4 Analysis*).

3.3.1 negative usability issues

The list (A2- Negative usability findings) is the result of analysing data collected by think-aloud protocol in usability evaluations. The list prioritizes issues by severity and frequency of occurrence. The decision has been made to exclude the minor issues, because it is expected that there is not enough time to fix them and often they are a direct consequence from a serious or critical issue. A closer look into the list of issues reveals that most frequent serious and critical issues are dealing with navigational problems and misunderstanding of information in the game. Furthermore the issues share the tendency to happen to a group of locations in the game. If those issues are solved, usability will most likely improve. Therefore it suffices to focus on wayfinding and providing information and only use locations in which serious and/or critical issues occur for the re-design and to exclude the rest, which is further discussed in *phase 2*.

3.3.2 Expert review

The conducted expert review delivered results in the form of recommendations for the game with the help of participating experts. To recall , the review existed out of the following components: 1) Discussing usability issues per location, 2) Providing information and 3) Discuss pen-and-paper prototype. This triplet of components contributed to clear recommendations for the game, which results are discussed by one at the time.

1) Discussing usability issues per location

The decision was made to include specific locations with serious and critical issues only, which either deal with navigation or providing information. Results from expert review are put into tables containing the issues, occasions and the recommendations from left to right. They can be seen on the next pages.

1.1 Cabins (Hutten)

Negative usability issues	Occasions/errors	Recommendations/remarks
• Uncertainty about the game mode button functionality.	• Puts game mode on 'off' • Clicks and drags the game modus button.	• An actual game mode button instead of a slider would solve the respondent's behaviour to drag this button. • A settings button close by the information button which provides an overview of options. All these options should also be clearly commented and contends with the existing uncertainty.
• Uncertainty about the wayfinding within the game, respondent not sure where to go.	• Respondent goes the wrong way. • Seeks help at the information button.	• The cabins need a permanent floating text that immediately shows what they are. • Giving the cabins extra attention by showing an outline on mouse hover. • Route signs should be used as trigger to gain more attention. Also putting permanent floating text above them.
• Ingredients for a meal should be named, not a necessity to actually possess them.	• Respondent is distracted by walking crabs and tries to catch it. • Respondent wants to harvest potatoes from the home garden.	• *No recommendations or remarks.*
• Difficulties navigating to the home garden.	• Clicks wrong on the route guiding to the home garden.	• Changes areas of interaction, maybe only click on the route signs. However would potentially increase difficult of wayfinding. • Areas of interaction from message cabin and Island are to close to one another.
• Purpose of the cabins is not clear.	• Chooses wrong cabin.	• The cabins need a permanent floating text that immediately shows what they are. • Giving the cabins extra attention by showing an outline on mouse hover.

1.2 Home garden (Moestuin)

Negative Usability issues	Occasions/ error	Recommendations/remarks
• Could not find entrance to Kitchen.	• Leaves the home garden. • Seeks for help at the information button.	• Create 2 distinct buttons for 'leave the home garden' and 'back to cabins' to put extra emphasis on them, which might increase usage. • Allow interaction with route sign. • Change the button content to make their purpose more clear.
• Functionality and visual display of inventory elements is not clear.	• Not understanding the inventory elements. • Multiple clicking on inventory elements.	• Show name of inventory element upon mouse hover. • Click on inventory element to show additional information about it.
• Uncertainty about how the home garden could be left.	• Clicks on the back button.	• Create 2 distinct buttons for 'leave the home garden' and 'back to cabins' to put extra emphasis on them, which might solve this problem as well. • Global homebutton.

1.3 Kitchen (Keuken)

Negative usability issues	occasions / error	Recommendations/remarks
• Meal description and its ingredients are wrongly interpreted.	• Clicks on red colored text of the meal. • Respondent mentions the number 50. • Respondent mentions numbers but no ingredients.	• Cooking book idea: In the center of the page put a unfolded cooking book containing multiple pages displaying different meals.
• Clicks on not-clickable elements.	• Respondent expects campfire to have some function	• Cooking book idea of first issue can be used to tackle this issue as well by covering it.
• Functionality and visual display of inventory elements is not clear.	• Not understanding the inventory elements. • Keeps clicking on the inventory elements.	• Show name of inventory element upon mouse hover. • Click on inventory element to show additional information about it..

1.4 Island overview (Eiland-overzicht)

Negative usability issues	Occasions / errors	Recommendations/remarks
• Functionality of island overview not understood.	• Seeks help at the information button.	• Permanently add the names of minigames in the island overview. • Making a menu that contains all minigames, which is interactive.
• Cannot find riverbanks from island overview.	• Goes back to the cabins. • Goes to different minigame on the island.	• Permanently add the names of minigames in the island overview. • Making a menu that contains all minigames, which is interactive.
• Navigeren middels een niet-spel button	• Clicks on the back button.	• Global home button to the cabins.

1.5 Riverbank (Rivieroever)

Negative usability issues	Occasion / error	Recommendations/remarks
• The minigame riverbank is not clear.	• Seeks help at the information button. • Goes back to 'menu' • Clicks on the text introduction'	• Putting information button on the right side, because it is more consistent. • Add explanation of using arrow keys. • Add a timer which shows a popup on how to play the game and control the boat.
• Uncertainty on how to play the game riverbank.	• Clicks on boat and fish. • Clicks and drags the boat across screen.	• Clicks to continue instead of using spacebar as indicated. • Add explanation of using arrow keys. • Add a timer which shows a popup on how to play the game and control the boat.
• Functionality of the restart buttons is not understood.	• Multiple clicking on restart button.	• Multiple clicking on the restart buttons, is most likely a result of the other issues described here.

1.6 Boat (boot)

Negative usability issues	occasion / error	Recommendations/remark
• Avatar cannot move to the cabin floating on water.	• Multiple clicking on cabin floating on water • Respondent expects to build a boat there.	• Information pop-up in the middle of the screen to show the respondent which parts of the boat are already collected/ need to unlock. • Add information pop-up to trophies and achievements.
• difficulty to find way back to the cabins.	• Clicks on 'go back' in the wrong way. • Avatar get stuck between the cabins and the boat.	• Global home button can solve this issue. • Change the area of interaction between these locations.
• Goes in the wrong way.	• Clicks on comber cabin.	• *Check out 1.7 about comber cabin.*

1.7 the combers cabin (Juttershut)

The combers cabin (Juttershut) did neither have a serious nor critical negative issue, however still was discussed. The discussion was about removing the combers cabin, this location has too much minor issues and is generally not understood by respondents. Found objects do not really have any added value in the game.

1.8 extra ideas

While brainstorming on solving negative issues, also brings along other ideas worth mentioning. That is why they are considered to be extra. To ensure future users from accidently logging out, a confirmation pop-up needs to appear with the question whether or not the users wants to confirm its decision of logging out. The web portals were not discussed during the expert review, but there were thoughts on improving the connection between the game and the web portals. For instance when the web portals receive new messages or exercises the user should be notified in game about this. This notification should also be clickable to guide the user to that specific location. The last idea was related to discovering the riverbank minigame for the first time. When users find the riverbank, they should immediately play the introduction level instead of being directed to the level-overview in which they can select any level they want. If the introduction level is completed the user should be directed back to the level-overview.

2) Providing information

The location cabins (Hutten) is often not understood, which results in respondents not knowing where to go or what to do. The information presented by clicking on the information button is not helpful, because issues are not solved by that information. The information given gives a general overview of how the navigation works and what the purpose of the game is. During the review it became clear that the experts unanimously agreed that it should be completely changed.

There was agreement on the fact that the location cabins (Hutten) should provide extended information and details about the locations and wayfinding. The idea is to present the information in some sort of overview in which each theme can be clicked for more detailed information. The

themes for instance being the information cabin, messages cabin, exercise cabin, game elements and wayfinding.

3) Pen-and-paper prototype

The usage of pen-and-paper prototypes resulted into recommendations for improving wayfinding. The template made to communicate basic ideas already contained some of the recommendations from usability issues, so it was not discussed further. Experts used the template to try out a new wayfinding system and that resulted into *figure 3*. with a block representing one location and the amount of arrows the different ways the user can choose to go. In this way users can navigate by using outstanding arrow, which implies that all problems concerning the guiding signs can be solved.

There was also a remark about zooming in on the island overview. In that way all existing locations would fit on the map, creating a better structure and a better overview of the island. A good recommendation, however in the future more minigames are going to be added into Stranded on expert said. So this recommendations is not going to be used for the re-design.

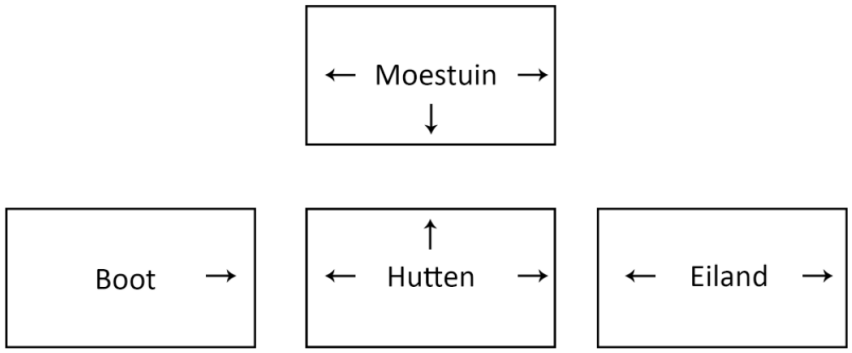


Figure 3: Using arrows to guide the user from one location to another.

3.4 Conclusion

Phase 1 of this project was about describing the evaluation and analysis of Stranded with target group. Usability evaluations conducted by predefined protocol, they delivered raw data in different data formats. This data was then processed into useful data by conducting an analysis by transcribing registering respondent actions and extracting negative usability issues. By looking at these issues it was found that serious and critical issues occurred on specific locations only and mainly are problems with wayfinding and misinterpretation of information. Finally, the extracted issues were used to conduct an expert review, which turned negative issues into recommendations and ideas for the re-design of the game Stranded. The goal of this phase was to conduct usability evaluations with the target group and to have recommendations based on usability findings, this was successfully done and now it is time to enter phase 2 were the recommendations of phase 1 are used to create an improved design based on literature and results.

Phase 2: Creating a re-design based on results and literature

Phase 2 exist of describing how the improved design was created on basis of results and literature. It describes the ideation, specification and realisation phases consecutively, which help develop the improved design step by step. The ideation was performed to get an more elaborate idea on what the re-design should be, then the specification lists all demands and requirements needed for the re-design and finally the realisation describes the process of making the re-design from scratch by Unity2D. All performed phases within phase 2 eventually led to describing how the re-design became capable of representing the new version of the game.

Chapter 4 describes the ideation phase. Decisions were made to focus on wayfinding, providing information and serious/critical issues, so the re-design is not a full remake. This phase contains decisions about software and prerequisites, and trying out an early design concept by using pen-and-paper prototypes.

Chapter 5 describes the specification phase, in which recommendations were compared by literature to check their validity and form a list of all demands and requirements necessary to make a re-design.

Chapter 6 describes the realisation phase, it contains the process of realizing the actual re-design from scratch by turning recommendations into working design solution in Unity2D. The re-design is also play tested to see how the design solutions work in practise.

4. Ideation phase

The purpose of having an ideation phase is to get a more elaborated idea on how the re-design should be made, and what the re-design should look like. This is an important first step, since decisions were made about focusing on wayfinding, providing information and serious/critical issues which impact the process of creating an improved design. The aim of this project is not to remake the entire game, but to communicate design solutions by a working re-design. Therefore it is necessary to encapsulate only the necessities for making the re-design by setting prerequisites. Also additional design ideas were part of the result by using pen-and-paper prototypes.

4.1 Prerequisites

The improved design is not going to be a full remake of Stranded, the goal is to communicate the design solutions and not to make an entirely new operational game. It would have been interesting to make the complete new game, however the period of time is too small in order to be able to do that. Also because the original web game is created by using C as programming language, which is very low level and therefore difficult to learn and costly in time. To overcome these constrains and make a re-design within the period of time, alternatives needs to be found and limits needs to be set.

4.1.1 Limits in making re-design Stranded

Communicating design solutions does not require an entire remake, this paragraph discusses which parts are and are not in available in the re-design and why. The limits that are going to be applied to the re-design are decisions based on usability findings and own skill set with creating prototypes. To improve the usability, current usability issues need to be solved, in particular the serious and critical ones. As mentioned in *3.3.1 negative usability issues* and visible in *appendix A2- negative usability*

findings most frequent serious and critical usability issues are dealing with navigation problems and misunderstanding of information. These issues share the tendency to happen at specific locations in the game. If those issues are solved, usability will improve most likely. Therefore it suffices to only use those locations in the re-design and exclude several parts.

The following paragraphs provides an overview of all decisions made on which parts stays and what parts leave. Starting with global elements in the game, most functionality of the UI is not going to be present in the re-design. Game Modus, Basic Modus and logging out buttons are only going to be visually present. Information buttons will be made functional as it was a popular button during the usability evaluations. Transitions and other animations that do not have any added value will be left out as well. Continuing to the game elements. The following parts are going to be excluded: all web portals, login screen, introduction movie and all minigames except the riverbank (Rivieroever). That means that the re-design will contain the following locations: cabins (Hutten), home garden(Moestuin), kitchen(keuken), Boat(Boot), Juttershut, Island(Eiland) and Riverbank(rivieroever).

Some locations that are present in the re-design are going to be limited in their functionality, because either the functionality is not necessary or the implementation of it is expected to be time consuming. The cabins (Hutten) are not going to have a walking avatar anymore, this implies the absence of catching crabs and collecting stranded bottles as well. Also since the web portals are disabled, the cabins cannot be clicked anymore. The home garden(Moestuin) is next in line, all its functionality with regard to manipulating the state of plants are left out, including the actual planting of crops in the garden. The kitchen (Keuken) location loses the possibilities of making meals, which implies the absence of a scoring system as well. Like mentioned before all minigames but the riverbank are left out, resulting in the island overview only providing access to the riverbank minigame. Finally the riverbank itself is, which is key in order to complete task 5 (Play a level of Riverbank) of the usability evaluation (*Appendix A1- Usability protocol*), will only also be limited. Meaning that the level overview only contains the level 1 setup (introduction level) in which the boat can be controlled, however the level cannot be completed successfully.

4.1.2 Software choice re-design

This paragraph describes the quest to find suitable software to create the re-design with, since the original web based game is written in C language as mentioned before. The most ideal scenario would be find already familiar software that can be used with no constraints. Before looking into different kinds of software, it is good to list all the requirements software should have built in. This can be done by looking at the characteristics of Stranded.

The game is 2D dimensional and top-down with a slight angle; the game type allows an avatar to walk into 4 different directions. Sprites and objects tend to overlap one another. This style allows for better graphics. The software for the re-design should be able to make a web based prototype, camera angles and in some way work conformable with different layer options and overlap. Furthermore in order to add new visual elements, software should allow importing assets in different kind of formats like PNG or PSD (Photoshop). Also the additional of visual elements indicates that it is necessary to have a graphic renderer for different sprites and objects.

From own experience with web based technologies the possibilities with using JavaScript, HTML, and CSS are endless. These endless possibilities however come at a price. In this case more versatility means more difficulty and time. Still an viable option, since it checks every box on requirements. But

JavaScript would probably cost some time to master and get used to, even if there is a JavaScript library out there which makes it easier.

Choices about the re-design were discussed with supervisors and other people, which yield some valuable second opinions on this matter. These opinions provided good feedback on what to do and what not to do in terms of using software. They listed all kinds of software for a re-design. Wireframe software, interactive PowerPoint presentation, Adobe DX, JavaScript libraries, JavaScript engines and Unity engine. From the meetings it became clear that the re-design should be interactive and allow for user interaction, that is why only JavaScript libraries, JavaScript engines and Unity remain from this list. After a little research it became clear that Unity is the final choice, or even better Unity2D.

Unity is a game engine used for development of 2D/3D games, simulations and other purposes. It uses the so called drag-and-drop functionality within a user friendly UI. Has easier options for layering sprites and objects than JavaScript. The programming language is C#, which is a more friendly language than C and personally had good earlier experiences with. Also the game can be exported as a WebGL build. This means that the game can be put on a compatible webpage without having to worry about the graphics, plug-ins and other adjustments. Unity checks all the requirements boxes and is therefore also a viable options. But because of good earlier experiences the personal preference goes to Unity2D as choice for re-designing Stranded.

4.2 Method

4.2.1 Pen-and-paper prototypes

During and after conducting usability evaluations as observer you get a feeling for the game and develop an understanding of what works and what does not work. Finding out if those ideas can work can be done by evaluating prototypes. Which is an early design concept of the game that can be learned from. Prototypes can be either low-fidelity or high-fidelity. Low-fidelity is more sketchy and incomplete, whereas high-fidelity is fully functioning and allows user interactivity.

Low-fidelity pen-and paper prototypes is the perfect choice at this moment for ideation. Pen-and-paper prototypes can be created in a short amount of time, is cheap, the quality does not have to be perfect and no programming code has to be written in order for it succeed. Walker, M., Takayama, L., & Landay, J. A. (2002) and Sefelin, R., Tscheligi, M., & Giller, V. (2003) compared low and high fidelity prototypes in both computer and paper media and came to the conclusion that both prototypes were equally successful in uncovering issues. Also Snyder, C.(2001) found the same conclusion in the more than 100 usability issues she conducted with pen-and-paper prototypes. She also states that they are especially useful to get feedback about navigation/workflow, content, page layout and terminology. Which perfectly syncs with the choice made in *3.3.1 Negative usability issues* to re-design for wayfinding and providing information. One interesting side note Sefelin, R., Tscheligi, M., & Giller, V. (2003) puts on using pen-and paper prototypes is that users find low-fidelity prototypes unconformable, which is a drawback to consider. One the other hand Snyder, C.(2001) and Walker, M., Takayama, L., & Landay, J. A. (2002) do not state anything about users being uncomfortable , so the side note is debatable.

The pen-and-paper prototypes were constructed out of photoshopped screenshots, which are considered the templates of each location. They lack the presence of UI elements. The idea behind it is to edit the templates by hand to quickly iterate over ideas. Providing information is seen as second hand, because the game should be made self-explanatory in first place (as much as

possible). Also there are numbers drawn to create a simple navigation system. A scheme has been made which guides the user without help to each location.

4.3 Results

The pen-and-paper prototype templates were sketched to showcase the first basic ideas, and to discuss them with the supervisors for additional feedback. The sketches can be seen in *appendix A3-Pen-and-paper prototype Stranded* with the navigation scheme as well. This sketch contains ideas about making the buttons overall more consistent by giving them a solid position in the UI. Furthermore text that is interactive should have its own borders and some padding in order to emphasize it has functionality. The sketch also attempts to use new content for some buttons in order to make them more clear.

During meetings to showcase the pen-and-paper prototype with supervisors, additional ideas came to live. The most important ideas consist of adding a global home button, getting rid of the location combers cabins (Juttershut), using a timer when playing riverbank to show information about controls and put more emphasis on the direction signs throughout the game by adding glow or repositioning. These ideas are going to mentioned in the *specification Phase* and will be implemented in the *Realisation Phase* of this project. The first steps are made to improve the current design and answering on what requirements this re-design should have. These requirements will be listed in the *Specification Phase* which will provide one big overview of requirements for the re-design.

4.4 Conclusion

The ideation phase led to a more elaborate idea on how to re-design the game and what it should look like. The requisites limited the re-design to only include locations with serious and/or critical issues without web portals and provide basic functionality. Also the software choice being Unity2D, a user friendly game engine that has all the listed requirements in order to make the re-design. A better idea on what the re-design should look like was gained by showing pen-and-paper prototype during meetings to the supervisors of this project. All necessities regarding the creation of the re-design are encapsulated in an better understanding on how to finish the create the re-design.

5. Specification phase

The specification phase checks recommendations for validity by literature research and then list them in a useful and usable way. A programmer should be able to understand what to do if he/she has to create the re-design. The state-of-art research concluded that the re-design of the UI should take age-related changes into account by making use of UI design principles. This method section describes how UI design principles and MoSCoW prioritizing is used to validate recommendations. Then the result section is a MoSCoW prioritization list of validated recommendations ready to be used in the *Realisation phase* and finally a conclusion about this phase is given.

5.1 Method

5.1.1 UI design principles

The generic solutions described in the state-of-the-art research and principles for interaction design described by Nielsen were used as guidelines to validate recommendations for the re-design. Lots of publications and webpages describe principles and guidelines for UI design, but they can almost always be backtracked to 10 usability heuristics for interface design by Nielsen, J.(2005). The recommendations are compared to these heuristics to shape them into useful and usable recommendations that can be used as design solutions for the re-design.

5.1.2 MoSCoW prioritizing

The MoSCoW method is a prioritizing method with the focus of delivering the most important aspects first and the rest later if possible. It is used to create an understandable overview of all recommendations by sorting them in priority and locations. Often used when projects have a certain fixed period of time, to make sure it is a success by first implementing the crucial requirements of the re-design. Clegg, D. & Barker, R. (1994) named this method with the acronym MoSCoW meaning Must haves(M), Should haves (S), Could haves(C) and Won't haves(W). The Must haves are the critical requirements necessary to make the project a success. The Should haves and Could haves are removed when time does not allow implementing them. The Won't haves are requirements either for future work or disposed requirements. This method leads to a priority list containing useful and usable recommendations based on results and literature.

5.2 Results

5.2.1 MoSCoW prioritizing list

The recommendations are retrieved from *chapter 3 Expert review results.*

Must haves(M)

- User Interface: The game mode button should be made into an actual button instead of a slider.
- User Interface: Using arrows to guide the user from one location to another as described in *4.3 Expert review: pen-and-paperprototypes.*
- Cabins(Hutten): Above each cabin the name should be displayed in a text holder, indicating what they are.
- Cabins (Hutten) Provide extended information and details about the locations and wayfinding. The idea is to present the information in some sort of overview in which each theme can be clicked for more detailed information.
- Home garden(Moestuin) : Create 2 distinct buttons holders for 'leave the home garden' and 'back to cabins'. Also add vertical padding between them.
- Home garden(Moestuin): Change button text to make their purpose more clear.

- Kitchen(Keuken): Implement cooking book idea- In the middle of the page put an overview of different meals. Displaying their ingredients by numbers and icons, how much already made, score per meal and the total score.
- Inventory in home garden and kitchen: On mouse hover on icons, show their names in a pop-up.
- Riverbank (Rivieroever): Add explanation of using arrows keys to control boat in the information section within the introduction level.
- Riverbank (Rivieroever): Add a timer for x time which counts down to 0 if user is not controlling the boat and then show a popup for x time on how to play the game and control the boat.

Should haves(S)

- User Interface: A settings button right under the information button which provides an overview of in-game options. All these options should also be clearly commented and show distinction between being enabled and/or being disabled.
- User Interface: If a location is either 2 clicks or more away from the cabins. A Global home button should be added in the upper left corner.
- User Interface: The position of the information button should be on a consistent position throughout the game.
- User Interface: Add confirmation screen on logging out, which can be answered by 'yes' or 'no'.
- Cabins(Hutten): Giving the cabins extra attention by showing an outline on mouse hover.
- Cabins(Hutten) and home garden(Moestuin): Allow interaction with route signs.
- Boat(Boot): Information pop-up in the middle of the screen which shows everything that has been collected and/or needs to be unlocked.
- Boat(Boot): Getting rid of the combers cabin(Juttershut).

Could haves(C)

- Cabins(hutten) and Moestuin(home garden): Route signs should be used as trigger to gain more attention. Also putting permanent floating text above them.
- Cabins(Hutten): Changes areas of interaction, which are to close each other.
- Island overview(Eiland-overzicht): Permanently add the names of minigames in the island overview.
- Island overview(Eiland-overzicht): Making an interactive menu that contains all minigames.
- Riverbank(Rivieroever): Make every pop-up 'click to continue' instead of 'press spacebar to continue'.
- Inventory in home garden and kitchen: Click on inventory element to show additional information about it.
- Boat(Boot): Add information pop-up to showcase trophies and/or other achievements.

Won't haves(W)

- *4.1.1 Limitations in making re-design Stranded* described what is excluded from the re-design.
- Zooming in on island overview as described in *Expert review extra ideas*
- Provide in-game notifications if something new occurs in the web portals.
- Adaptable UI that can be controlled by the user.

5.3 Conclusion

The specification phase used the recommendations from the expert review and checks them on validity by literature research, after that they were listed in a useful and usable way to make them more suitable to use as design solutions for the re-design. The recommendations from the expert review were verified by the 10 usability heuristics for user interface design by Nielsen, J.(2005). The MoSCoW prioritizing method is applied to create an understandable overview of all validated recommendations sorted by priority and location in the game. The specification phase transformed recommendations from expert review into a list of validated recommendations ready to be used as design solutions in the *Realisation phase.*

6. Realisation phase

The realisation phase contains the process of realizing the actual re-design from scratch by turning the list of validated recommendations from the specification phase into working design solution, which are implemented in the re-design. First the list of Must haves was implemented, then the Should haves and Could haves. After the re-design was made, a small evaluation by playtesting was conducted to see how the design solutions work in practise.

Chapter 6 discusses the method, results and conclusion about realizing the new version of the game. The method section first discusses about how recommendations turned into working design solutions. Then discusses each stage of making prototypes in Unity2D. Finally the results are presented by showing before/after screenshots of the game and discusses the feedback gained from playtesting with 2 testers.

6.1 Method

6.1.1 Translating recommendations in working design solutions (not yet complete

This paragraph discusses the reasoning behind added, changed and/or visual elements throughout the game. What was the recommendations? and how did it became a working a design solution? Translating recommendations in design solutions is not always an intuitive thing to do. Some solutions proposed had to be implemented properly to avoid introducing new usability issues, while the goal is to get rid of them in the re-design. The truth is that there is no way recommendations can be implemented and at the same time ensured it is not introducing new issues, several methods were used to prevent this as good as possible:

- Meetings after each stage of building the prototype were held to get valuable feedback on the implemented features.
- Apply universal designs or icons whenever possible.

6.1.2 Playtesting

Prototype 2.0 is finished and is able to represent a re-design of the original game to show the implementation of the made design solutions. The representation is with respect to wayfinding, providing information, basic functionality and setup. It has the ability to allow for user interaction in almost the same way as the original game is being presented and played by respondents. There is however one remaining question mark. What will users think about the implemented design solutions made in the re-design? These solutions are made because of various issues and remarks, in theory they should solve all the problems encountered. However how does that work in practise? In order to find out if the solutions do work and what users think about it, it should be tested. To properly test it, you should do similar usability evaluations again with new people from the target group and compare the scores and feedback you get. This is not the intention of this project, so the re-design will be tested in a different way.

The type of testing that is going to be used is called playtesting, it is going to put the implemented design solutions into practise. Playtesting is about getting people to play with our re-design to see if the decisions made for the re-design were in fact good decisions. From experience this can be a fast and viable method to get proper feedback in a short amount of time, without the need of it being formal. To get the most result out of playtesting, it is necessary to have specific goals in mind. Therefore Schell, J.(2015) wrote several rules to accomplish that, Lens #103: The lens of Playtesting will used to determine these goals. In Lens #103 Schell, J.(2015) states that “playtesting is your chance to see your game in action. To ensure playtests are as good as they are, ask yourself

these questions: Why are we doing a playtest? Who should be there? Where should we hold it? What will we look for? How will we get the information we need?" (pp.446). Let's answer these questions one by one.

Why are we doing a playtest?

Without a specific goal in mind about why there should be a playtest in the first place, it is necessary to know what questions the playtest needs to answer ideally. These are the important questions that need an answer:

- Do players understand the wayfinding via UI elements in the game?
- Are the design solutions as good in theory as they are in practise?
- What changes, if any, should be made the make the re-design work better?

Who should be there?

The people that will be used to conduct the playtest are going to friends and family. They are available on short notice, are able to provide answers and are also tissue testers. Which are testers who have never seen the game before and can therefore see the game in a fresh way, which allows them to see new things the designer will not notice anymore.

Where should we hold it?

Friends and family will be invited at the play testers home. Which allows for close observations on how the game is interacted with and the seeing their genuine attitude about the game while they are playing. Furthermore the play testers home is somewhat of a natural habitat to family and friends.

What will we look for?

What to look for in the playtest exist out of things you actively look for and things that you do not know you are looking for. The playtest needs to be designed in such a way that is provides answers to questions proposed. Not all locations in the game are necessary to be reviewed entirely, however the tester is free to explore the island . The locations that are important to test are: Cabins (Hutten), home garden (Moestuin), kitchen (Keuken) and riverbank (Rivieroever). Therefore the decision is made to let play testers perform task 1, 4 and 5 from the original usability test (A1-Usability protocol reference). These tasks allow for exploration, yet forces the tester to explore to visit the important locations and will most likely succeed the best in answering the questions. For each task the tester will be given 3 minutes to complete them.

How will we get the information we need?

The structure of the playtest needs to be proper in order to get our wanted information. The decisions are made to tell the tester upfront the re-design is about improving the wayfinding and the UI within the game, not the functionality. To sit next to the tester to observe them in their decision making and facial expressions. According to Schell,J.(2015) these contain the most valuable

information you need. During the test the tester needs to focus on the competing task and therefore the observer will not ask any questions during the tasks that might disturb them. Other important data to look at is the amount of clicks and time testers use in an attempt to finish the task. Questions will be asked afterwards in a small interview. This interview will ask the tester what he or she thinks about the game, how they think wayfinding is, what information they additionally would have liked to have and according what I notice during test some additional questions.

Rough structure playtest

Now that all five questions are answered the playtest and all its details can be constructed. The playtest consist out of 3 parts: Setup, tasks, interview. The setup will be the same setup as the original usability evaluations. The game will be made available online in a webpage and the tester will test the game by means of a laptop and mouse. Once the tester knows what is expected of him/her it will get 5 minutes for each tasks. And executing the tasks mentioned earlier consecutively, with some pause in between different tasks. After all tasks are done, the interview part will begin. During the interview some standard questions are asked and additional spontaneous questions will be asked depending on how the playtest went.

6.2 Results

The re-design is finished and capable of communicating the design solutions made throughout this project. Making the re-design was an iterative process in which each stage had its own prototype version where features were added, changed or removed. Each stage of this process is discussed in *Prototyping in Unity2D*. The reasoning behind new, changed and/or removed features are discussed in *Design solutions and reasoning.* To see the differences between the old and new design, check out *Appendix A5-Screenshots original design versus re-design*. At the end of the appendix some screenshots of new features were added as well.

6.2.1 Prototyping in Unity2D

This section discusses the whole process of making the re-design in Unity2D. From using assets to programming with C# to ensure basic functionality. The source code of prototypes is included in *Appendix A6- Source code Unity2D,* but is not discussed into technical details. At each stage supervisors were consulted to provide feedback, which was used to further develop the re-design. Prototypes are indicated by a number X dot Y. in which X stands for including recommendations MoSCoW priority list, and Y applying feedback from supervisors.

Prototype 1.0

Prototype 1.0 is the re-design which includes the Must haves and is capable of running user tests. In this prototype the global UI was made and the cabins (Hutten) finished as first location. All locations were created as empty scene and equipped with navigation options. The location Hutten was finished first to serve as a template for the other locations. The UI and settings were copied and paste in all locations to make the re-design consistent. The icons in the inventory in home garden and kitchen were created and allow mouse hover to show their names via pop-ups. The riverbank got basic functionality of moving the boat, permanent explanation pop-up containing image of arrow keys and text. It was also decided that providing information is going to be second hand, because the re-design should be initially self-explanatory. This prototype 1.0 is not ready capable of running user tests yet, since some must haves took more time than expected.

Prototype 1.1

This prototype is prototype 1.0 with implemented feedback from supervisors. The feedback was about the game setup and WebGL build(Building the game to be played in web browsers). The re-design should have the same game setup as Stranded. Meaning that it should be playable full screen inside a different web browsers. After some research about making Unity games full screen, additional changes have been made. Internal files of the build had to be adjusted, as well as some settings within Unity. The code had to be replaced in the index.html and style.css file. This is now possible with the explanation and code from *Appendix A7- Replace index and style code*. Once that is done, the prototype is ready to be played full screen inside different web browsers.

Prototype 2.0

Prototype 2.0 is the re-design which includes Should haves and Could haves. The remaining must haves were implemented first. A global settings icon and a settings menu were created. The settings contains all game options except wayfinding and information. A design for the kitchen menu was made and implemented. The combers cabin(Juttershut) was deleted from the boat scene and this scene also got a progression menu. This menu showcases all the achievements the user has made so far. For future work also add things that need to be unlocked. This prototype was ready to conduct user testing with, since it contained at least all must haves.

Prototype 2.1

Prototype 2.0 with implemented feedback from supervisors. The feedback was about adding finishing touches and adjusting some details. The major adjustments are discussed in here.

Ideas were to add text clouds to the cabins on mouse hover, in this way you can provide easy and fast information towards the user. Make the wayfinding arrows into a button, to have a more consistent UI of just buttons. Furthermore to change the layout of the home garden to allow more space for the tool to be put on the left side of the inventory. Restart the riverbank level should show the pop-up about controlling the boat again. And finally to divide the progression menu in the boat scene into 2 parts: 1) Quest description and 2) the rewards. These changes were implemented and the result was a much more vivid and clear re-design.

Prototype 2.2

After implementing the feedback that made prototype 2.1, final feedback was gathered from the supervisors. Again some finishing touches and small adjustments to improve the re-design. The following adjustments were made:

- Every scene should have small pop-ups explaining not all functionality is available in this re-design.
- The menu containing the meals in the kitchen should be activated by clicking on the campfire in the background.
- Remove the game button option from the settings menu.

After implementing the adjustments in for this prototype, it became the re-design that was used in playtesting described in *6.1.3 Playtesting*.

Prototype 3.0

This prototype does not exist, but would have been the final version of the game. Due to the period of time feedback from playtesting was not implemented, however discussed in *6.2.2 feedback playtest* and in *chapter 8 future work.*

6.2.2 Design solutions and reasoning

This section discusses the reasoning behind design solutions which resulted into the introduction of brand new visual elements in the game. Elements that were changed or removed are not discussed, only elements which are currently present in the re-design.

Global home button

The home button is a round-shaped button with a transparent icon of a house in there. Lots of webpages use the same design for a home button since the icon is universally understood.

Outline colour in cabins, boat and kitchen

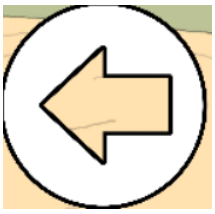
The re-design introduces the use of thick outlines around an object as hover indicator. The bright yellow used in the outlines for the cabins were chosen because they allow extra emphasis on that object despite its surroundings. The other colors of outlines throughout the game were chosen because the surrounding also have that same color attribute.

Settings menu in UI

This a newly introduced menu containing several options in the game. It uses the same background as the information pop-ups. The color of the active buttons match the blueish shades in the game, which show good contrast on a black background The inactive buttons have a greyish shade to blend in more with the static black background and provide the sense of being inactive. The menu can be existed by pressing a universal bright red “X” button to exit.

Arrow navigation buttons in UI

To make the user interface consistent throughout the game, the choice has been made to give the arrow a round-shaped button as well. The concept of this idea is described in *3.3.3 expert review: extra ideas.*

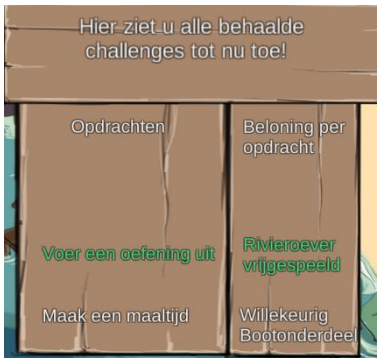


Tools menu in the home garden(moestuin)

Tools in the home garden are repositioned and divided into a block of 2x2 squares. The repositioning and shrinking the height of the tools allows the user interface to be consistent. Division in 2x2 squares with edges makes each tool int the tools menu more distinguishable.

Progression menu in the Boat location(Boot)

the boat locations was neither had a clear purpose nor was the background used. A progression menu is made for the user to see their progression by quests and rewards. This menu can active by clicking the cabin floating on water, in this way the background is not covered by the progression menu the entire time. This menu uses the same sprites as the tool menu in home garden, it is easy to distinguish for the rest and the theme of building a wooden boat is somewhat preserved. Quests and reward can have white text(quest not completed) and green text(quest completed).



Meal menu in kitchen(keuken)

Meal menu is made into a table which provided clear overview of the meals and its attributes.

The campfire need to be clicked to active the menu (same principle as progression menu at boat location). The meal menu is distinguishable from other elements, but uses existing colors from the kitchen to blend into the environment.

6.2.3 Feedback playtest

The playtest was conducted 2 times to gather some final results to improve the re-design. The 2 testers that participated will be called T1 and T2 respectively from now on. The tester exist of family and friends and again are not from the target group the re-design is actually designed for. Not going to describe the tester, but sketch an global image of who participated. T1 is a 20 year old male student, T2 is a 42 year old female with a working career. Both testers were able to finish all the tasks within the time, and were able to guide themselves through the game without much trouble.

Remarks Playtest of Tester 1 (T1)

Tester 1 understand the naturally understood the wayfinding via UI elements in the game. He used all button in the correct way. And did not take any detour with respect to navigating through different locations in order to finish the task. The remarks made while playing and from the interview afterwards are summed down here:

- Lack of sound and music while playing the game disturbs him.
- He noticed that the buttons in the settings menu were not working
- The location Boat (boot) is not having any sign of an actual boat or a place that indicates that a boat can be build there.
- He did not understand the purpose of the game immediately, although the name and locations gave some hints.
- Small information boxes not always noticed.

Remark Playtest of Tester 2 (T2)

Tester 2 also had a fast way of understanding the UI elements in the game, however showed more difficulty with the names of specific locations. She made some small detours while finishing the tasks. The remarks made while playing and from the interview afterwards are summed down here:

- Found it also odd the location Boat (boot) is not having any sign of an actual boat or place that indicated a boat can be build there. Told she expected something like a dock.
- On the task on playing at the riverbank (Rivieroever) she went to location boat (boot) and clicking on the green text indicating riverbank is unlocked.
- Meal description was not complete in the kitchen.
- Tried to apply potatoes to the home garden.
- Had no difficulty noticing the small information boxes.

What does this mean for the re-design

Overall the re-design proved theory works in practise, UI elements were clear and used correctly. Navigation was used in a natural way and succeeds in achieving the goal it was designed for. The need of having additional information was not there, however that is not a valid conclusion since it was a small playtest, and testers from outside the target group. Furthermore remarks from both play testers were helpful. It became clear that the boat location needs to show something that resembles an actual boat presence and fixing some minor attention points.

6.3 Conclusion

This realisation phase is conducted to realize the final re-design in Unity2D from scratch by turning list of validated recommendations established in the specification phase into working design solutions. Prototyping is an iterative process, every stage delivers new versions. Each new version brings features which are new, changed and/or removed. Every staged was checked by supervisors. Translating recommendations in working design solutions is not always intuitive to do, especially if new visual elements are used. Feedback from meetings and the application of universal designs and icons were used to prevent introduction of new usability issues as best as possible. After the re-design was successfully made a final evaluation called playtesting was conducted. This method put the implemented design solutions from the re-design into practise. Playtesting is about getting people to play with the re-design to see if the decisions made for the re-design were in fact good decisions. The final evaluation showed that the design solutions did not introduce new usability issues and is therefore ready to represent a re-design of the original game.

7. Conclusion

A high fidelity re-design of Stranded is realized to show the implemented design solutions formed from evaluation data throughout this project. Usability evaluations and analysis were conducted on the game Stranded with the target group of older adults(55-65 years). This resulted into recommendations which were validated by literature and then used as design solutions to realize an improved design of Stranded in Unity2D. The re-design is a high fidelity prototype which means it allows user interaction and can be used to test the implemented design solutions. It is, however limited to basic functionality and only includes specific locations, UI, wayfinding, providing information.

The main research question was about how to enhance/improve the usability of the game Stranded for the target groups? Usability evaluations and analysis were necessary in finding out what problems users encountered in the game, solving these issues is expected to increase the overall usability. The choice has been made to improve the usability by re-designing the UI and focus on tackling the serious and critical issues. These issues occurred only on specific locations and were mainly dealing with problems in wayfinding and misinterpretation of information. Re-designing UIs for older adults is about taking their age-related changes into account. However, this can be accomplished by generic design solutions applied for older people. Therefore it suffices to validate recommendations by comparing them with 10 heuristics described by Nielsen, J.(2005). Translating validated recommendations in design solutions is found to be not intuitive to do. There is no specific way to do that. Feedback from meetings and the application of universal designs and icons were used to make this translation happen and prevent introduction of new usability issues as best as possible at the same time.

A high fidelity prototype offers the opportunity for a last evaluation by playtesting and also future evaluations/testing. Feedback from playtesting was positive and showed that the solutions presented in the re-design are working correctly and do not introduce new usability issues. However, testers were not people from the target group. To find out if the re-design is actually improving the current usability, one should redo the exact same usability evaluations with different respondents.

8. discussion

The usability of the game Stranded has been improved by tackling serious/critical issues found by usability evaluations and analysis, the solutions to these issues have been communicated by means of a high fidelity re-design. Improving the usability was important because the game is designed to be a motivational tool for older adults to perform their physical exercises on a regular basis. It can only reach this purpose if certain constraints are met(*2.1 user interface)*. Improving the usability is the chosen way to overcome those constraints.

Usability evaluations and analysis was done with 10 older adults to find out what negative usability issues respondents encounter while playing the game. Also to provide recommendations to solve these issues. After some evaluations the decision was made to re-design the UI, since it caused most of the problems. This was confirmed after all evaluations were done, by looking at all issues it was found that serious and critical issues occurred on specific locations only and mainly deal with wayfinding and misinterpretation of information. It became clear that the re-design should only include locations where serious or critical issues occurred with the focus on re-designing the UI, especially on wayfinding and providing information These issues were discussed in an expert review, resulting into recommendation on how to solve them.

The recommendations from the expert review needed to be validated by literature research. It was found that re-designing for older adults is about taking age-related changes into considerations. Which could be done by applying generic solutions described in literature(e.g. bigger fonts, contrast). Also it mentioned that this group has the tendency to read instructions first. Translating recommendations in design solutions had the risk of introducing new usability problems. Feedback from meetings and the application of universal designs and icons were used to prevent this as best as possible. The re-design was made and evaluated by playtesting with 2 users to see how design solutions work in practise. These evaluations showed that the design solutions did not introduce new usability issues and is therefore ready to represent a re-design of the original game.

Findings from usability evaluation and analysis do agree on the literature out there. It was found that the re-design should focus on the UI, wayfinding and providing information in particular. Lot of sources agreed on the fact that the UIs play an important role and should contain easy core elements and provide clear/sufficient information in the game. This is difficult to provide, since a lot of emphasis was put on the keeping user's cognitive load as minimal as possible. Basically keep the amount of information low. However, it was also mentioned that older adults have the tendency to read instructions first. This means that the information should be minimal, but provide sufficient information about the game, which requires balance between minimal and sufficient. That users did have problems with wayfinding was not really found in literature, but I think that is because it is influenced by both the UI design and providing information.

Turning recommendations into design solutions was a difficult practise, since not a lot of literature provided guidelines or certain methods to accomplish that successfully. It was found that recommendations could be validated by using generic solutions for age-related changes, using heuristics from Nielsen, J.(2005) was sufficient to use. From there one it became a more difficult process, since there are no methods or guidelines out there. The only thing that could be done was asking feedback, using universal designs and use intuition. More research on this process should be done in my opinion, since it is the last important step to realize the re-design. The risk of introducing new usability issues is pretty decent, since the designer does understand the design but that does not always hold for older adults as well. Testing is the only way to find out if design solutions do work, but that takes a lot of time. From playtesting it was found that no new usability problems were

found, but it can be argued that is not a valid result. The testers do not originate from the target group. Also only 2 tests were conducted, so it is difficult to determine the weight of these results.

It can be concluded that more knowledge is needed to create a successful process of transforming evaluation data into working design solutions. The solutions for now is to test a lot with the target group, which is time costly but effective. The *future work* builds upon this discussion by providing ideas and suggestions on how to potentially continue for this project.

9. Future work

A high fidelity re-design of Stranded is realized and the final result of this project, but some testing, adaptations and functionality were left out due to time constraints. Future work concerns of more iteration cycles to improving the usability of the game Stranded. Throughout this project, new ideas came in my mind related to future work. These ideas for future work were gained from my acquired experience on user testing and valuable suggestions from literature research. A Bulletin list with ideas/suggestions is provided here below:

- Perform more UCD iteration cycles to find out whether or not the usability of Stranded is improved for the target group of older adults.
- A study on Information Architecture(IA); It is about how to present content to the user in order to make things understandable. It is necessary to study on how different elements in the game fit together in order to present for example clear instructions in the game. Some serious and critical issues were due to misinterpretation of information, but solved in the re-design by creating content which I believed was good enough, not basic on any fundaments or guidelines. In the *discussion* it was described that there should be a study on the balance between providing minimal, but sufficient information, which IA is suitable for in my opinion.
- Effects of sound to improve the user experience; Users from playtesting told that the absence of sound somewhat disturbed their user experience. I think it is interesting to see if sound can be used as a potential factor to increase of user experience.
- Adaptable user preferences; The game does not allow any adaptation of game elements, but according to Gerling, K. M., Schulte, F. P., Smeddinck, J., & Masuch, M. (2012) it needs to be included. The user should have the ability to control settings in the game (e.g. changing fonts, colors and window size). I am curious to see if this would really work in the game, and what the impact of implementing adaptability of user preferences is for the users.
- Learning from game preferences of older adults; the respondents that were tested were asked about what games they usually play, but this data was never used in this project. The data can contain valuable information which can be applied in this game by for instance learning about different user interface elements of games older adults play.
- Creating a better connection between different game elements; for example: the user can gain score points by making meals, but this has no added value in the game. The idea was to use the wishing well in the background of the kitchen location as an interactive object to spend the collected score points on. Every 100 points is equal to 1 wishing well credit, which can reward the user by unlocking new elements or even boat components.
- Turning recommendations into design solutions; literature does not provide a lot of help on accomplishing this task. It is more an intuitive process, with the hope of not introducing new usability issues. Therefore I suggest more research needs to be done on this process.

Each of these suggestions probably need their own iteration cycle to find out what their effects on usability or other factors will be. Iteration cycles take a lot of time to be completed, but low-fidelity prototypes can potentially reduce that time drastically. Walker, M., Takayama, L., & Landay, J. A. (2002) and Sefelin, R., Tscheligi, M., & Giller, V. (2003) indicate that pen-and-paperprototypes are able to uncover almost the same problems as a computer prototype would do, which could be an option for future development of Stranded. Also the game can be further improved by using investing more disciplines that affect the overall user experience (e.g. Information Architecture).

Appendix

A1 – Usability protocol

Respondent xx

1	**Welkom**		
2	Uitleg van onderzoek		2
3	Toestemmingsformulier		2
5	Demografische gegevens		2
6	Motivatievragenlijst		8
7	Hardop-denkmethode	Uitleg en audio toestemming	1
		Oefenronde	2
8		Taak 1: Log in en verken het eiland voor vijf minute.	5
9		Taak 1: after-scenario questionnaire	2
10		Taak 2: Er staat een nieuwe oefening voor u klaar, voer deze uit.	5
11		Taak 2: after-scenario questionnaire	2
12		Taak 3: Bekijk of er een bericht is van uw fysiotherapeut	5
13		Taak 3: after-scenario questionnaire	2
14		Taak 4: Bekijk hoeveel van welke ingrediënten u nodig heeft voor de volgende maaltijd: gepofte aardappel met krab.	5
15		Taak 4: after-scenario questionnaire	2
16		Task 5: Speel een level van het spel 'Rivieroever'	5
17		Task 5: after-scenario questionnaire	2
18	SUS vragenlijst		6
19	Post-assessment	3 vragen	4
20	Afronden		2

1.Informed consent

2. Demografische gegevens

Wat is uw geslacht?	
Wat is uw geboortedatum ?	
Wat is de hoogste opleiding die u heeft afgerond?	
Gebruikt u een of meerdere van volgende systemen thuis? - PC en/of laptop - Tablet - Smartphone - Game-computer (Kinect, wii, playstation) - Geen van deze - Anders, namelijk	
Als u een spel speelt, welk systeem is dan uw favoriet? - Fysiek spel (bordspel, kaartspel, puzzels) - PC of laptop - Tablet - Smartphone - Game-computer - Anders, namelijk	

3. Gezond leven vragenlijst

4. Uitleg hardop-denkmethode

4.1 Oefentreinreis van Enschede naar Hengelo

5. Audio toestemming

6. Taken

Taak 1

Beschrijving: Log in en verken het eiland voor vijf minuten.

Tijd:

Voltooid:

Einde: After-scenario questionnaire

Taak 2

Beschrijving: Er staat een nieuwe oefening voor u klaar, voer deze oefening uit.

Tijd:

Voltooid:

Einde: After-scenario questionnaire

Taak 3

Beschrijving: Bekijk of er u een bericht heeft ontvangen

Tijd:

Voltooid:

Einde: After-scenario questionnaire

Taak 4

Beschrijving: Bekijk hoeveel van welke ingrediënten u nodig heeft voor de volgende maaltijd: gepofte aardappel met krab.

Tijd:

Voltooid:

Einde: After-scenario questionnaire

Taak 5

Beschrijving: Speel een level van het spel ‘Rivieroever’

Tijd:

Voltooid:

Einde: After-scenario questionnaire

7. Gebruiksvriendelijkheid vragenlijst

8. Post-onderzoek interview

1) Wat denkt u dat voordelen zijn van het gebruiken van ‘Aangespoeld ?

2) Denkt u dat ‘Aangespoeld’ u zou helpen om uw fysieke conditie te verbeteren ?

3) Denkt u dat u ‘Aangespoeld’ zou gebruiken om uw fysieke conditie te verbeteren ?

9. Afsluiting

Taak 1:

Log in en verken het eiland voor vijf minuten.

Taak 1: Vragenlijst

	Sterk mee oneens						Sterk mee eens
1. Ik vond het uitvoeren van deze taak gemakkelijk	1	2	3	4	5	6	7
2. Het uitvoeren van deze taak kostte me niet veel tijd	1	2	3	4	5	6	7
3. De website gaf me voldoende hulp om deze taak uit te voeren	1	2	3	4	5	6	7

Taak 2:

Er staat een oefening voor u klaar, voer deze oefening uit.

Taak 2: Vragenlijst

	Sterk mee oneens						Sterk mee eens
1. Ik vond het uitvoeren van deze taak gemakkelijk	1	2	3	4	5	6	7
2. Het uitvoeren van deze taak kostte me niet veel tijd	1	2	3	4	5	6	7
3. De website gaf me voldoende hulp om deze taak uit te voeren	1	2	3	4	5	6	7

Taak 3:

Bekijk of u een bericht heeft ontvangen.

Taak 3: Vragenlijst

	Sterk mee oneens						Sterk mee eens
1. Ik vond het uitvoeren van deze taak gemakkelijk	1	2	3	4	5	6	7
2. Het uitvoeren van deze taak kostte me niet veel tijd	1	2	3	4	5	6	7
3. De website gaf me voldoende hulp om deze taak uit te voeren	1	2	3	4	5	6	7

Taak 4:

Bekijk hoeveel van welke ingrediënten u nodig heeft voor de volgende maaltijd: gepofte aardappel met krab.

Taak 4: Vragenlijst

	Sterk mee oneens						Sterk mee eens
1. Ik vond het uitvoeren van deze taak gemakkelijk	☐	☐	☐	☐	☐	☐	☐
	1	2	3	4	5	6	7
2. Het uitvoeren van deze taak kostte me niet veel tijd	☐	☐	☐	☐	☐	☐	☐
	1	2	3	4	5	6	7
3. De website gaf me voldoende hulp om deze taak uit te voeren	☐	☐	☐	☐	☐	☐	☐
	1	2	3	4	5	6	7

Taak 5:

Speel een level van het spel ‘Rivieroever’.

Taak 5: Vragenlijst

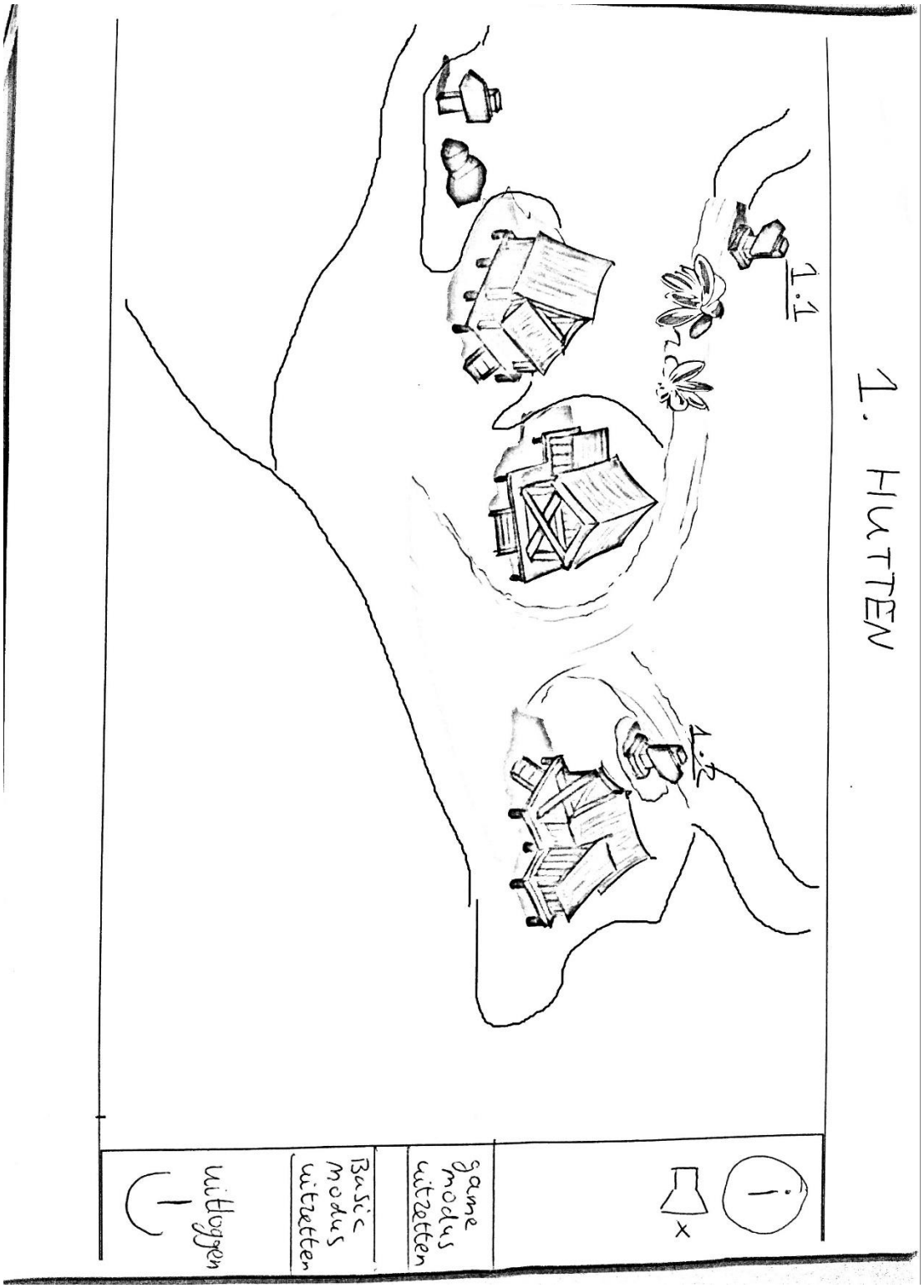
	Sterk mee oneens						Sterk mee eens
1. Ik vond het uitvoeren van deze taak gemakkelijk	1	2	3	4	5	6	7
2. Het uitvoeren van deze taak kostte me niet veel tijd	1	2	3	4	5	6	7
3. De website gaf me voldoende hulp om deze taak uit te voeren	1	2	3	4	5	6	7

A2 – Negative usability findings

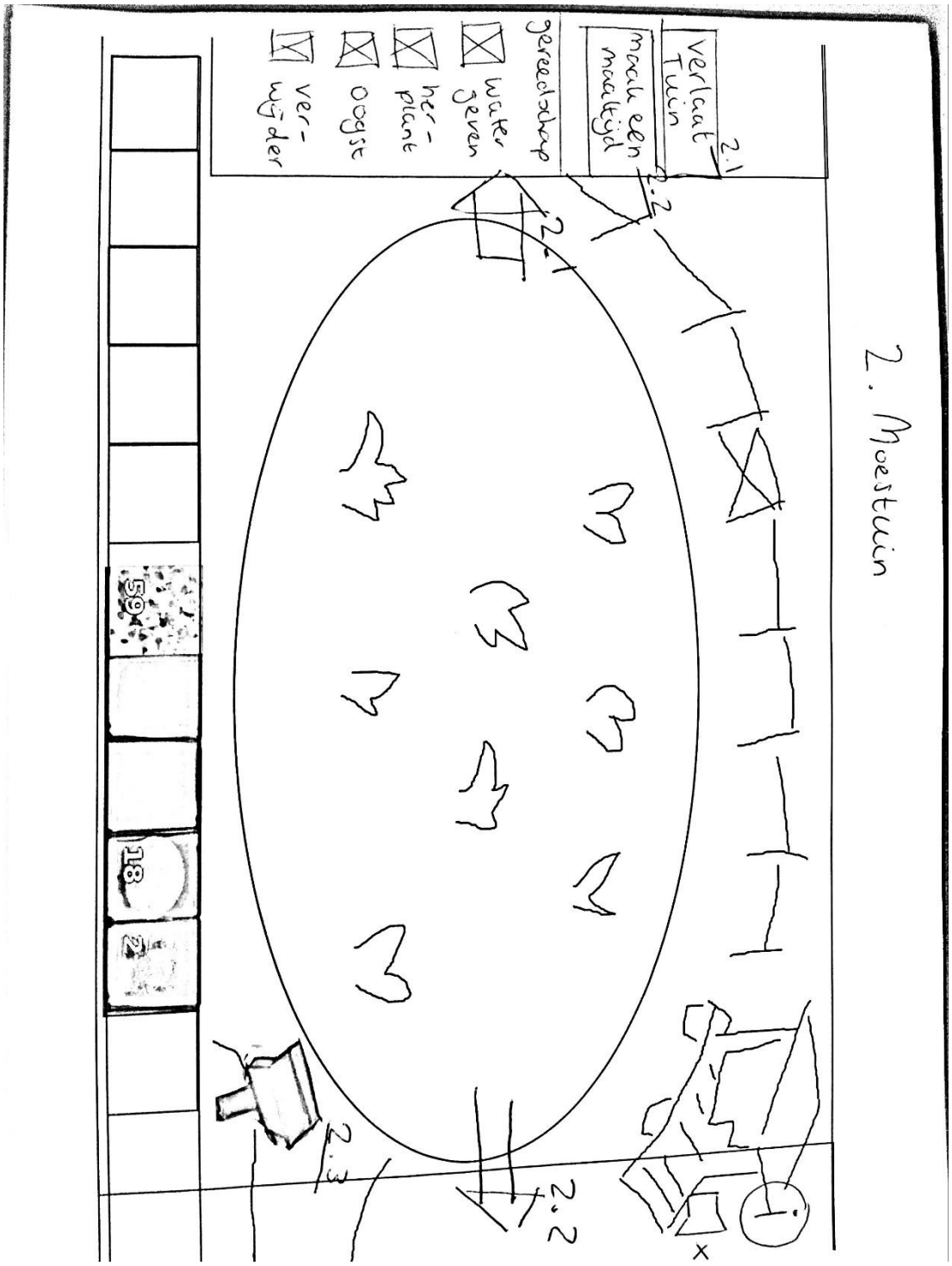
Level of severity	Description of usability issue	Location of issue (if applicable)	Respondents										
			R02	R04	R05	R06	R07	R08	R09	R10	R11	R12	Occurence
Critical	Uitleg binnen level van rivieroever geeft geen informatie over de besturing.	Rivieroever	■		■	■				■		■	5
	Kan ingang naar de Keuken niet vinden.	Moestuin	■		■	■			■	■			5
	Onduidelijkheid over de besturing van het spelletje rivieroever.	Rivieroever	■		■					■		■	4
	Onduidelijkheid over wat te moeten doen bij het spel rivieroever.	Rivieroever	■		■					■		■	4
	Beschrijving van maaltijd en haar ingredienten wordt verkeerd geintepreteerd.	Keuken		■				■					2
	Onduidelijkheid over de functie van de gamemodus knop	Hutten										■	1
Serious													
	Onduidelijkheid over de navigatie binnen het spel, respondent weet niet waar die heen moet.	Hutten/eiland	■	■	■	■	■	■	■	■	■	■	10
	Onduidelijkheid over wat te moeten doen of waar respondent heen moet.		■	■	■		■	■	■	■	■	■	9
	Navigeert middels een niet-spel button.	Alle hutten/ Moestuin	■		■			■	■	■		■	6
	Niet duidelijk waar de hutten voor dienen.		■	■			■			■		■	5
	Moeite met cijfers en/of speciale leestekens bij het inloggen.	Aanmeldscherm	■			■		■				■	4
	Klikt op onklikbare elementen in de achtergrond	Keuken		■				■		■	■		4
	Niet duidelijk dat ontvangen berichten al op de pagina wordt weergegeven	Berichtenportaal			■				■			■	3
	Beschrijving van maaltijd en haar ingredienten wordt verkeerd geintepreteerd.	Keuken						■		■	■		3
	Onduidelijk over de functie van de 'herstart' knop	Rivieroever			■					■		■	3
	Benodigdheden maaltijd moeten opgenoemen worden, niet een vereiste de benodigdheden te hebben.	Hutten	■					■				■	3
	Onduidelijkheid over de functie gamemodus knop.	Hutten	■					■		■			2
	Moeite om naar de Moestuin te navigeren.	Hutten							■		■		2
	Functie visuele weergave van elementen in de inventory niet duidelijk.	Keuken					■	■					2
	Onduidelijkheid over hoe de oefening opgestart kan worden.	Oefenportaal							■	■			2
	Nieuw mailbericht verdwijnt uit de ontvangen berichten door archiveren.	Berichtenportaal										■	1
	Onduidelijk hoe de tuin verlaten kan worden.	Moestuin				■							1
	Onduidelijk waar het eiland voor dient.	Eilandoverzicht		■									1
	Kan de rivieroever niet vinden	Eilandoverzicht				■							1
	Onduidelijk hoe de tuin verlaten kan worden.	Moestuin				■							1
	Kan ingang naar de keuken niet vinden.	Moestuin									■		1
	Onduidelijkheid over de functie van de blauwe informatiebalk	Berichtenportaal / Oefenportaal							■				1
	Verwacht de oefening correct uitgevoerd te hebben.	Oefenportaal									■		1

A3 – Pen-and-paper prototype Stranded

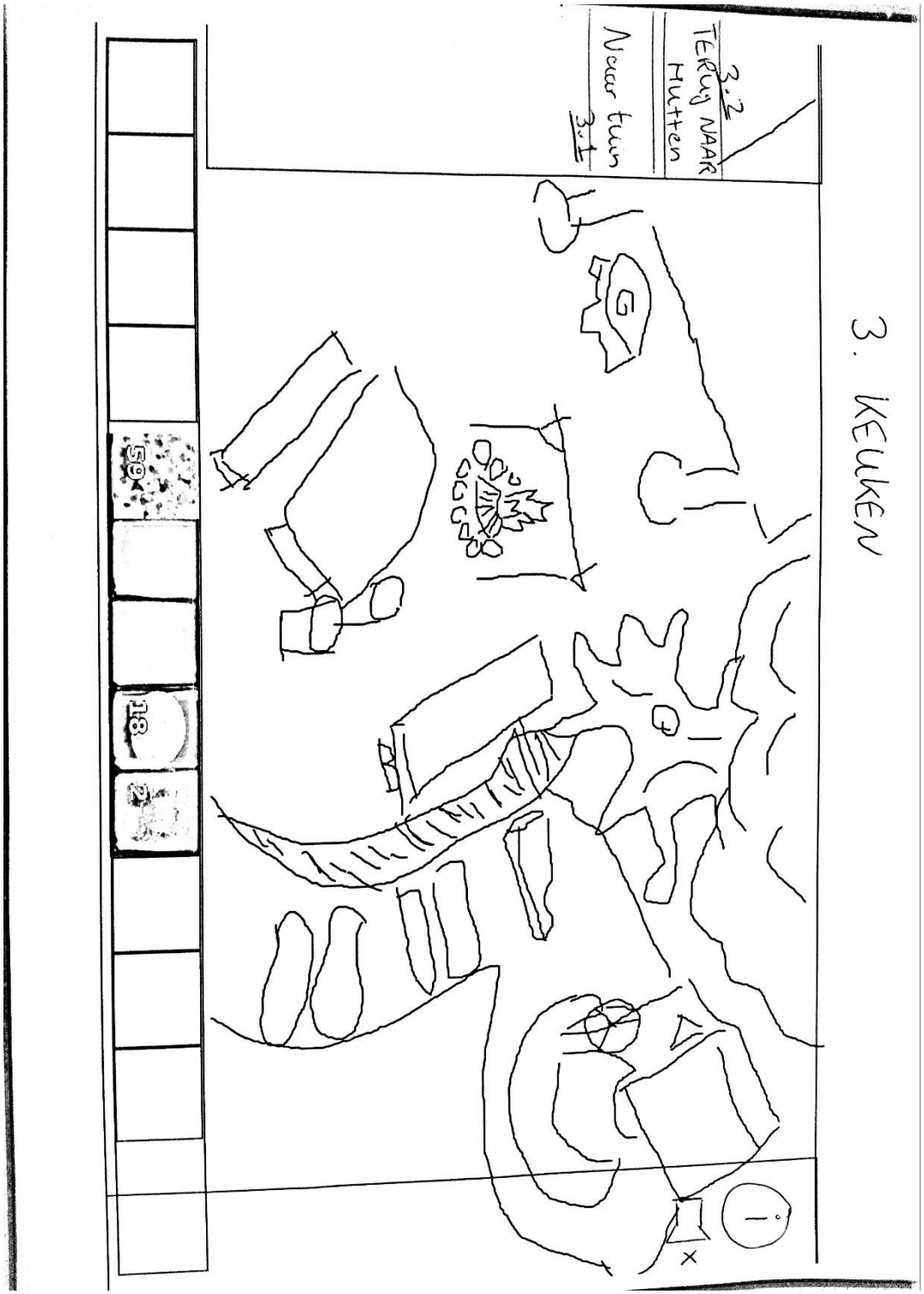
A3- figure 1: Stranded location the Cabins (de Hutten)



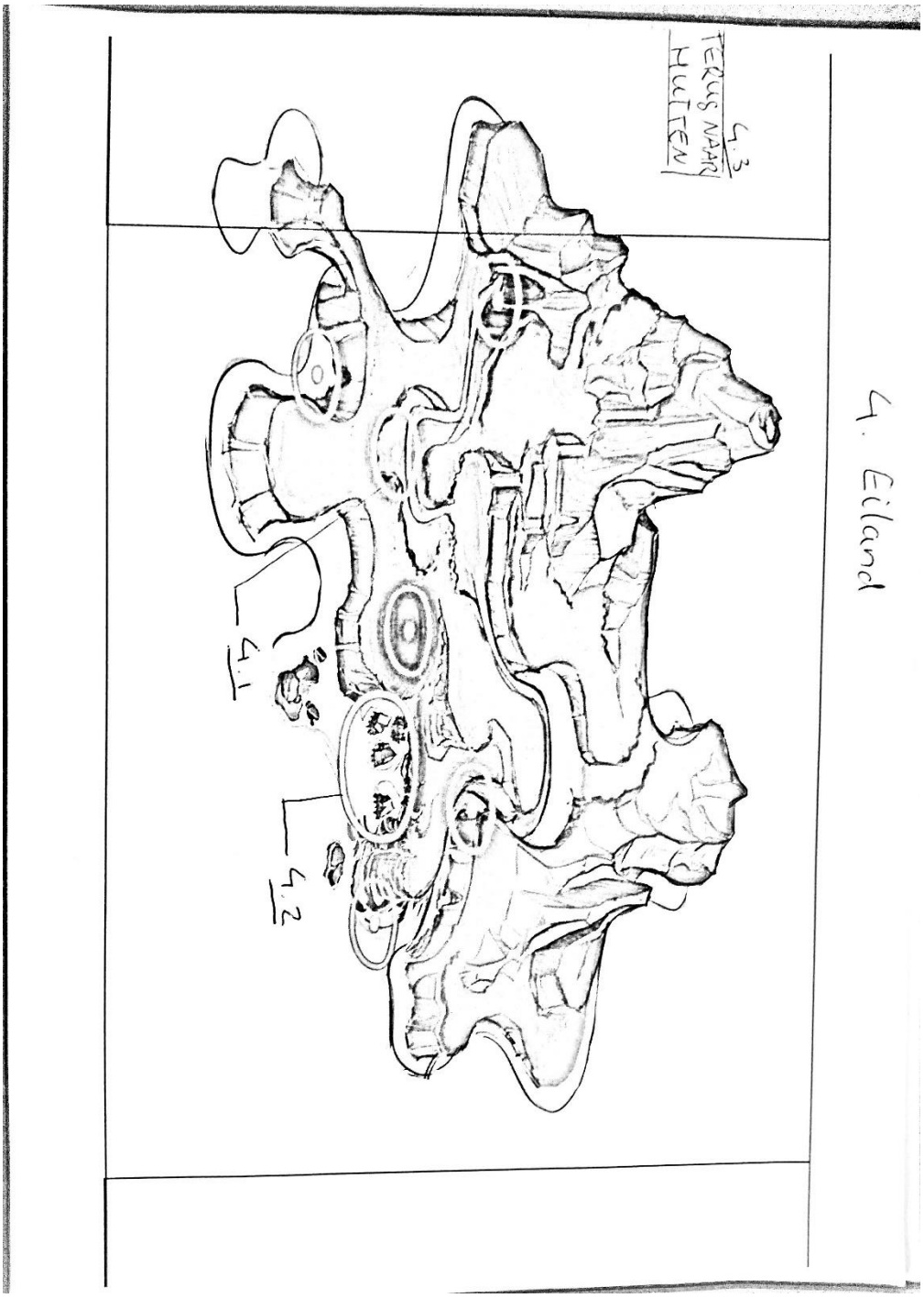
A3- figure 2: Stranded location the home garden (de Moestuin)



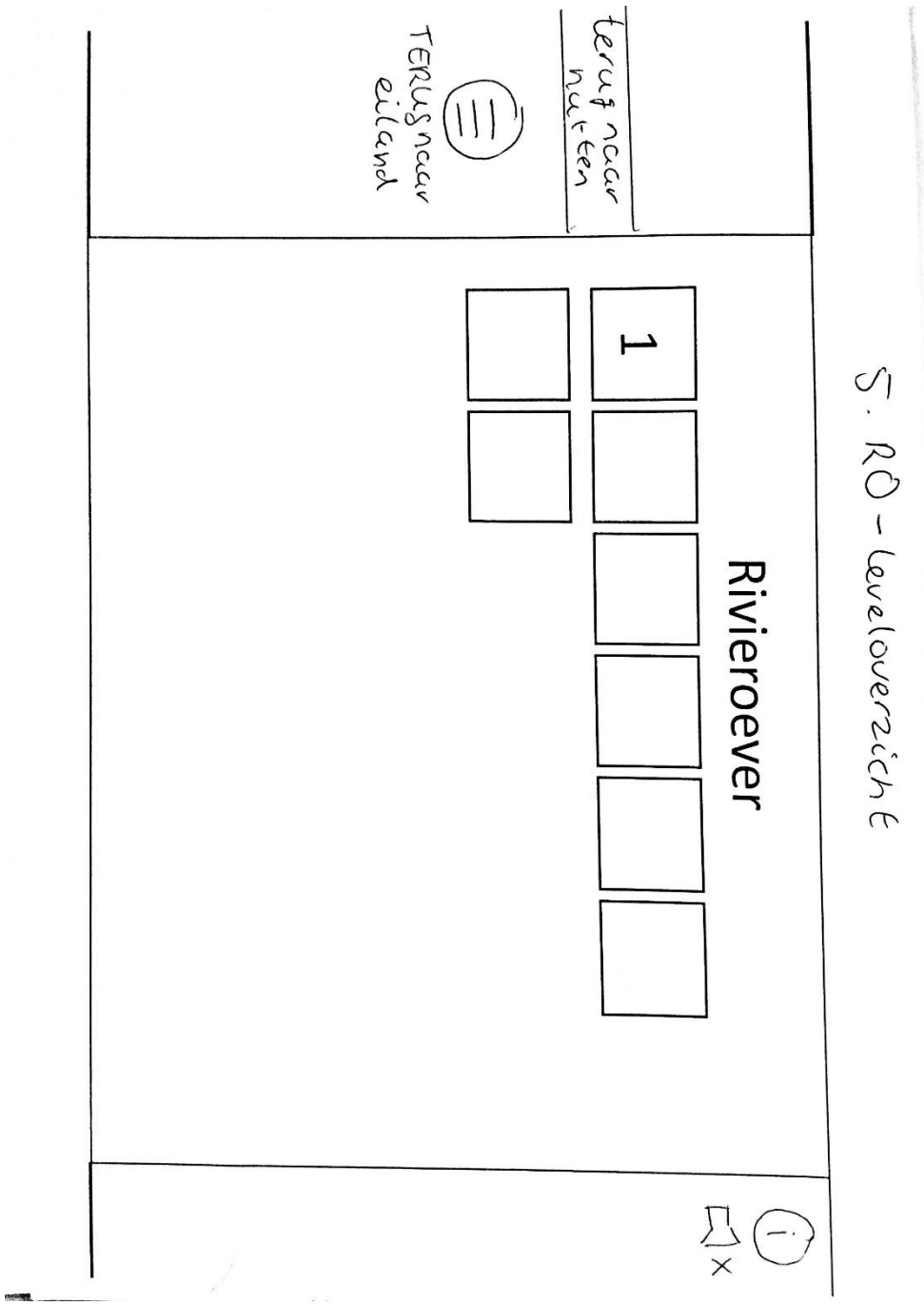
A3- figure 3: Stranded location the Kitchen (de Keuken)



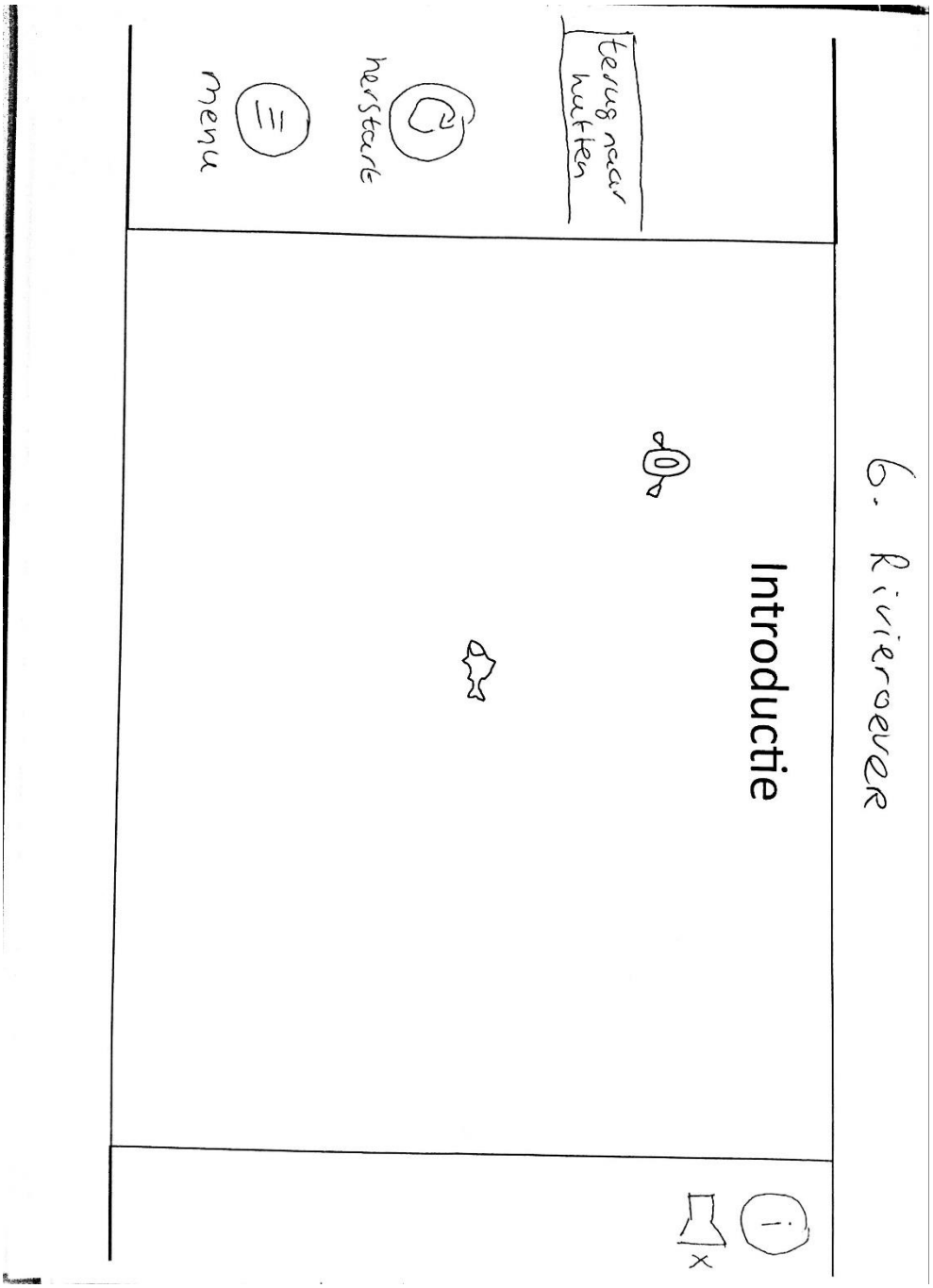
A3- figure 4: Stranded location the Island (het Eiland)



A3- figure 5: Stranded location the Riverbank- level overview (Rivieroever- leveloverzicht)

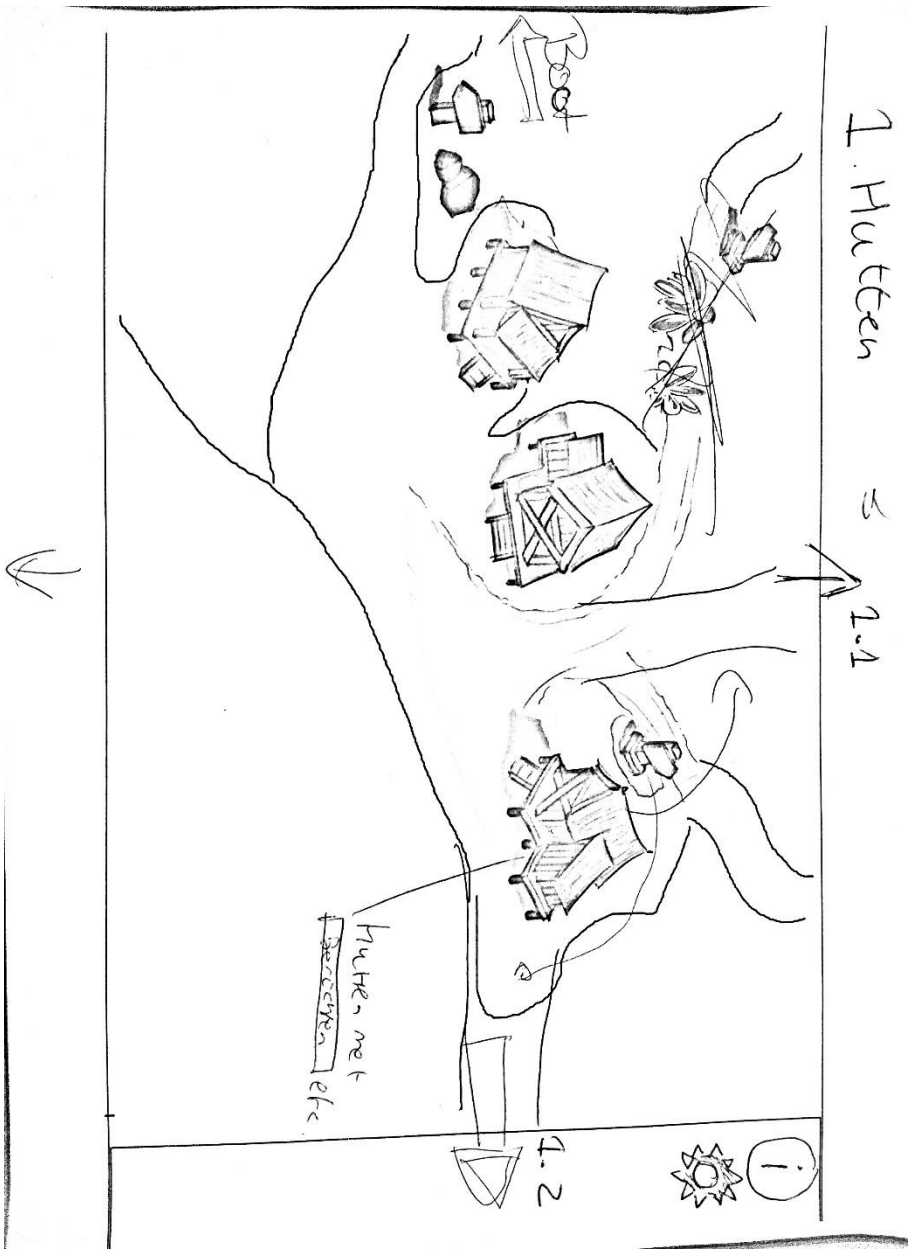


A3- figure 6: Stranded location the Riverbank (de Rivieroever)

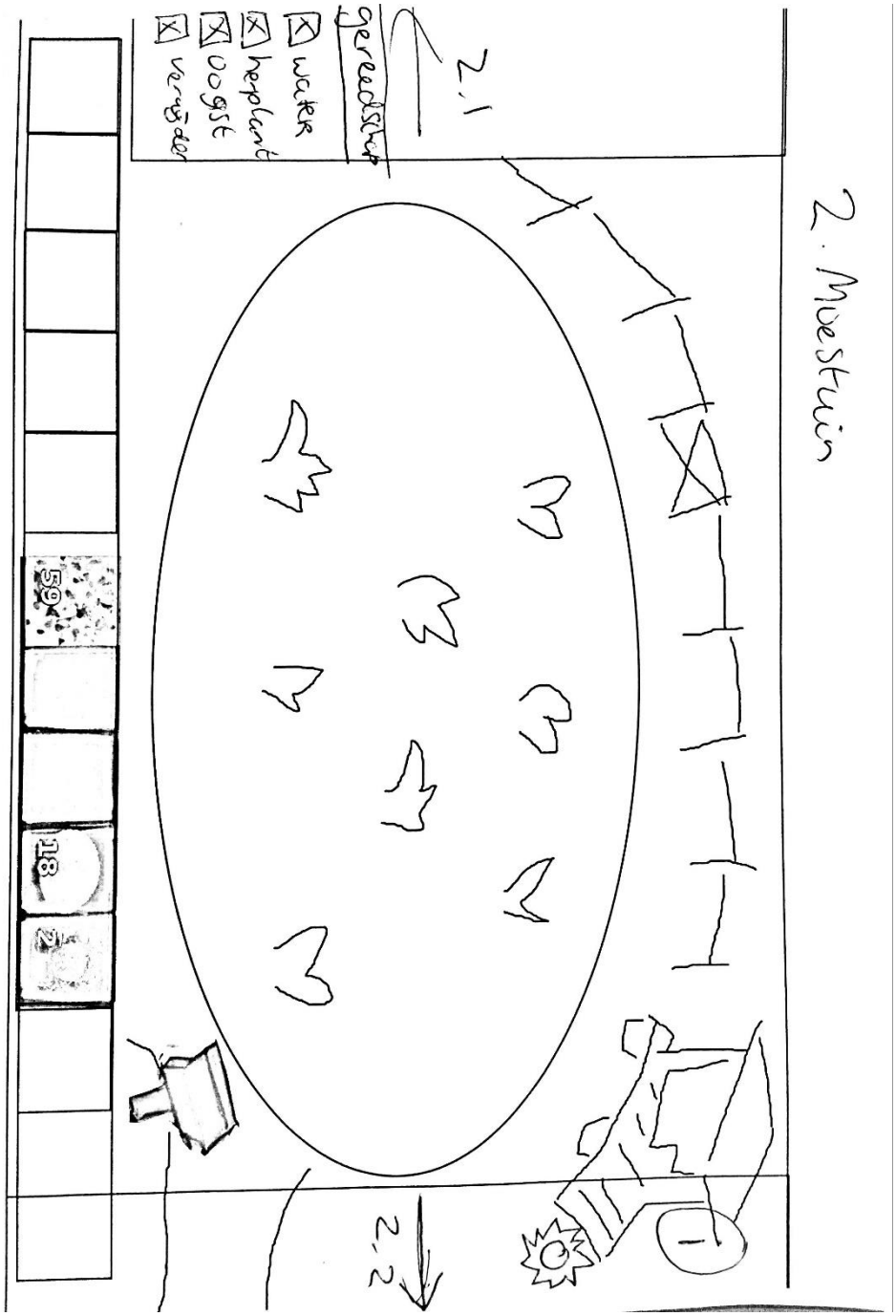


A4 – Pen-and-paper prototype Stranded after Expert Review

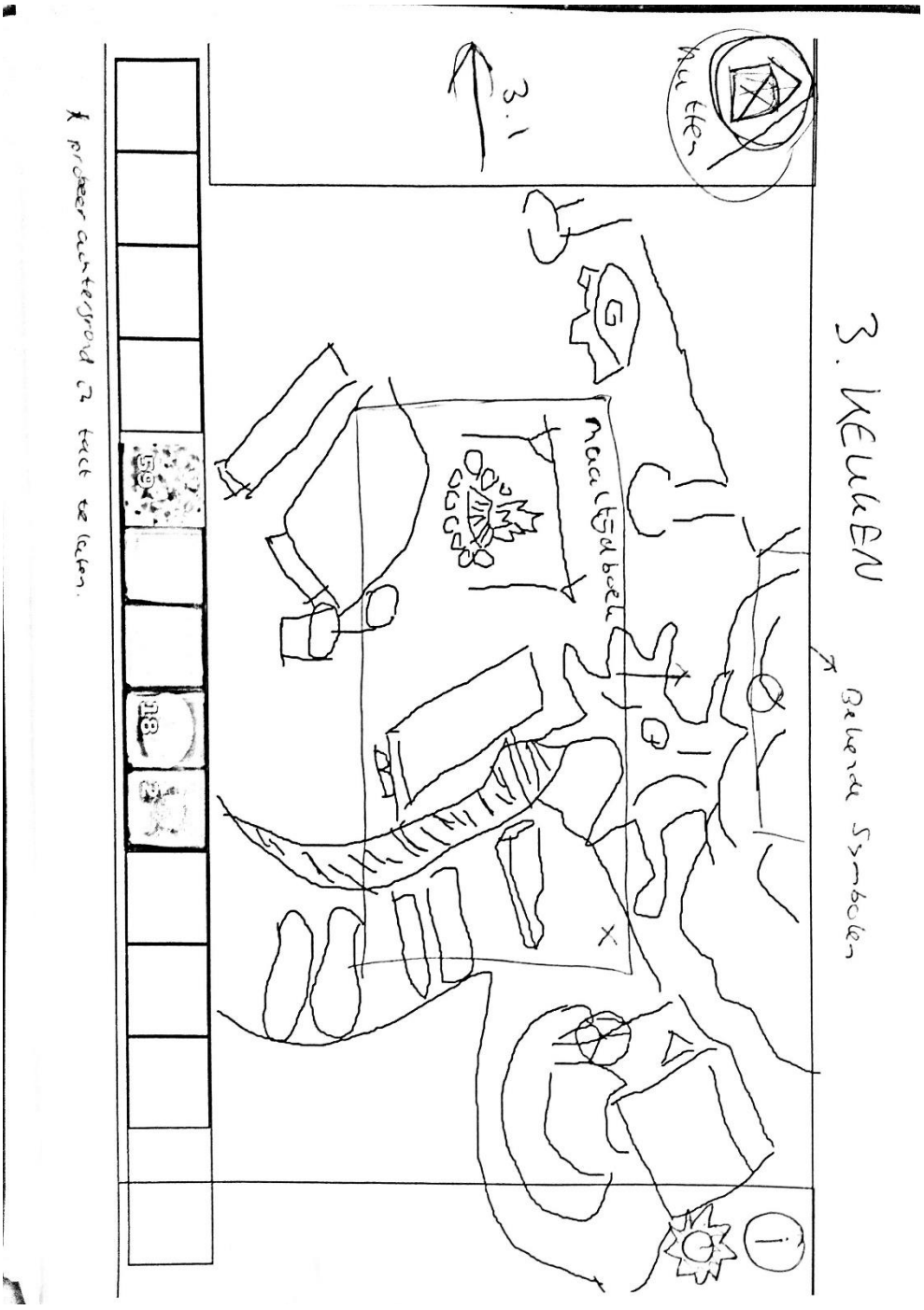
A4- figure 1: Stranded location the Cabins (de Hutten)



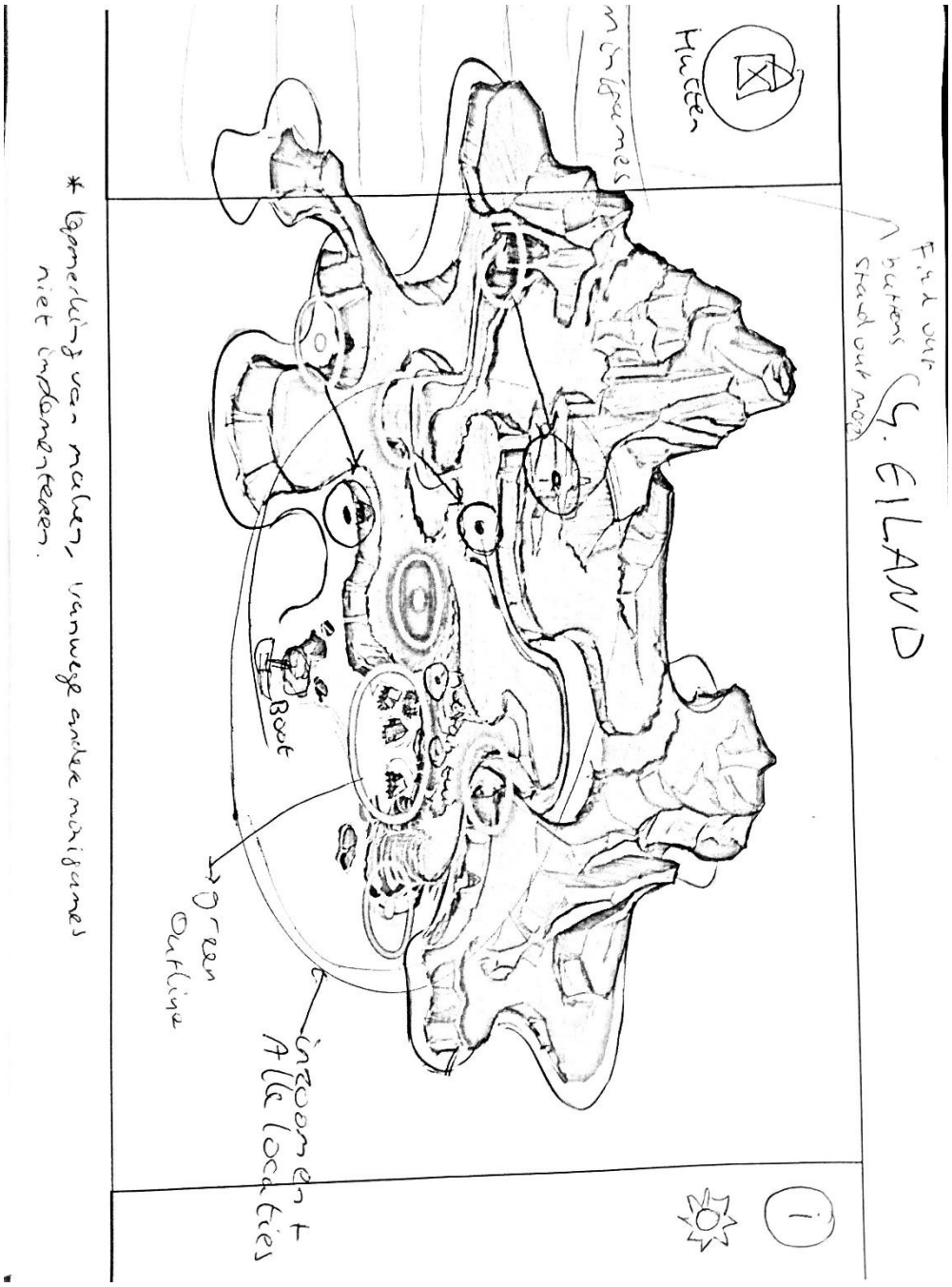
A4- figure 2: Stranded location the home garden (de Moestuin)



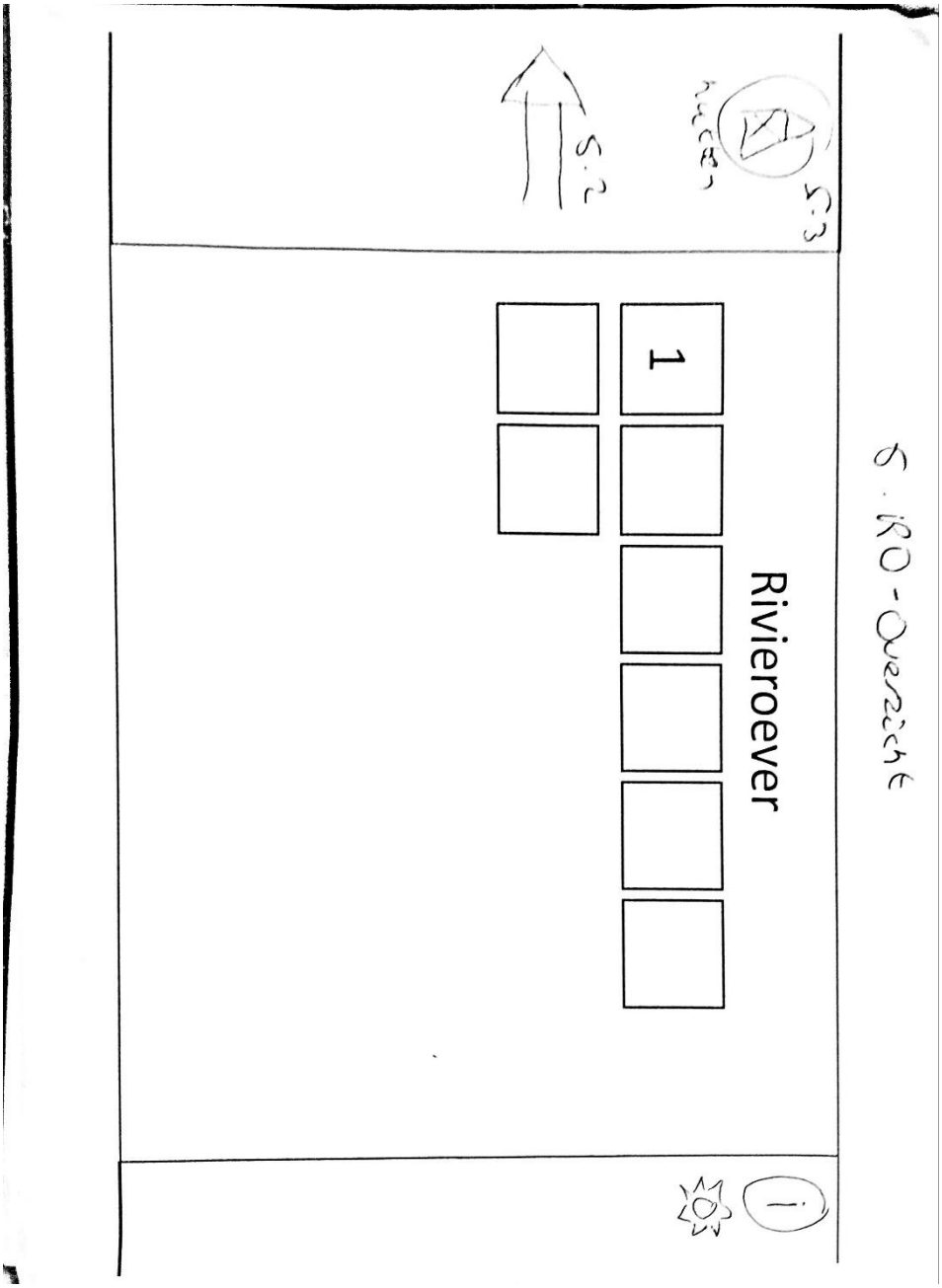
A4- figure 3: Stranded location the kitchen (de Keuken)



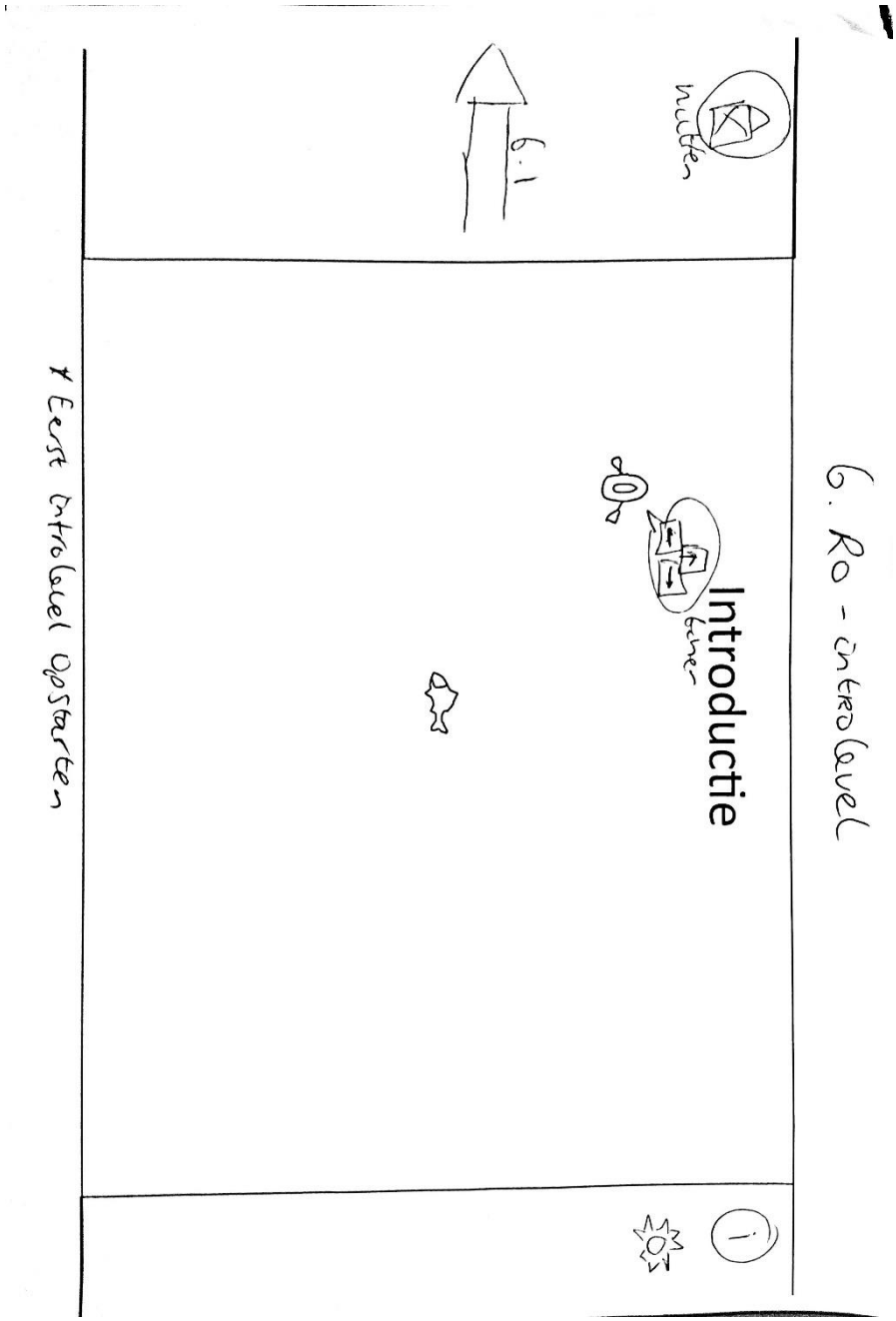
A4- figure 4: Stranded location the Island (het Eiland)



A4- figure 5: Stranded location the Riverbank level overview (het Rivieroever-leveloverzicht)



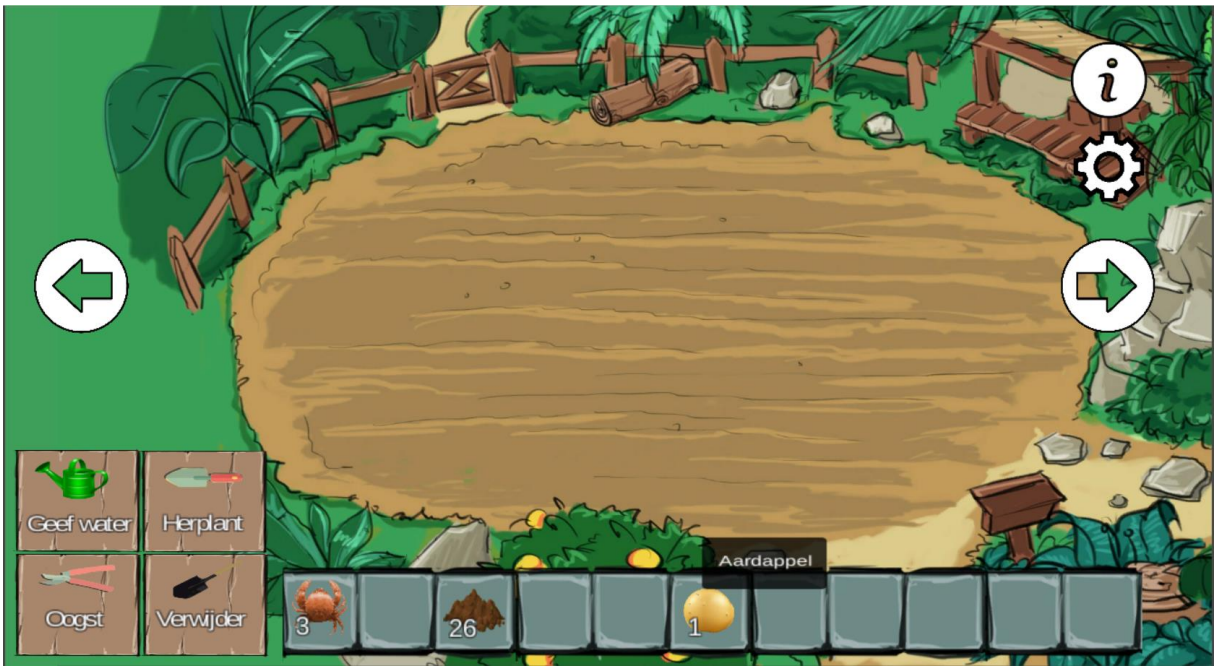
A4- figure 6: Stranded location the Rivierbank (Rivieroever)



A5- Screenshots original design versus re-design

Location the cabins (hutten)

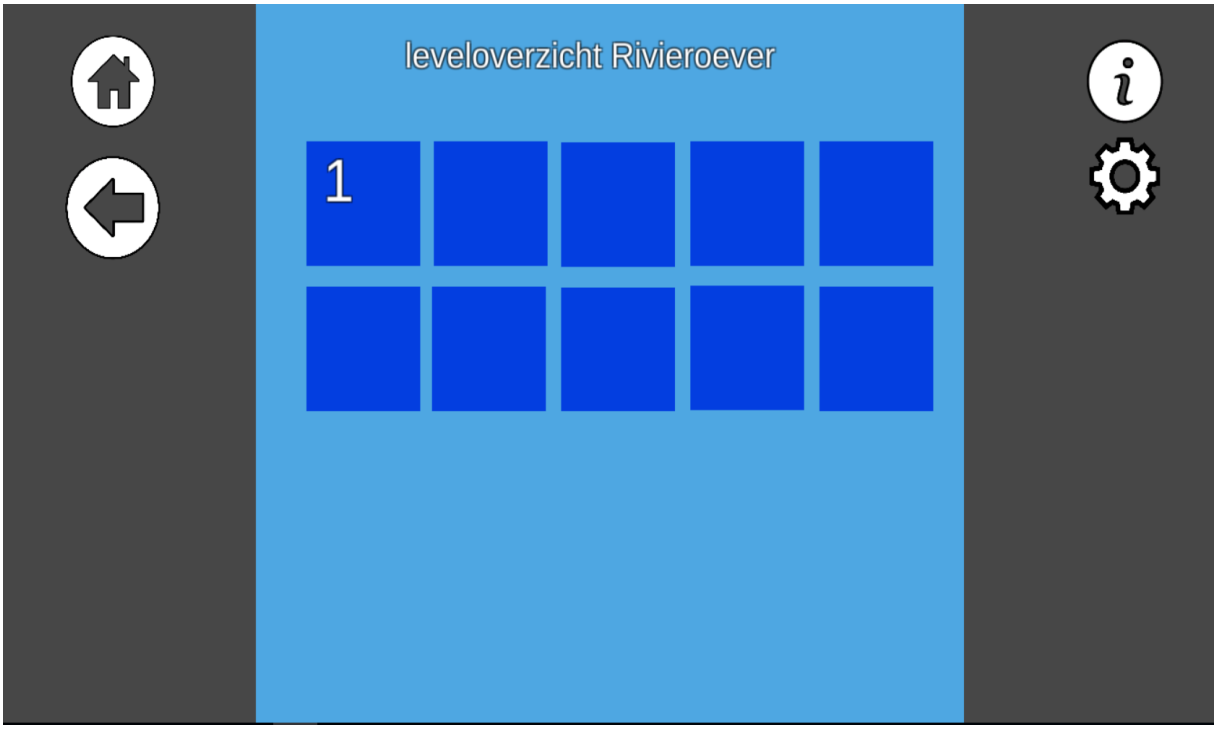
Location de home garden (Moestuin)



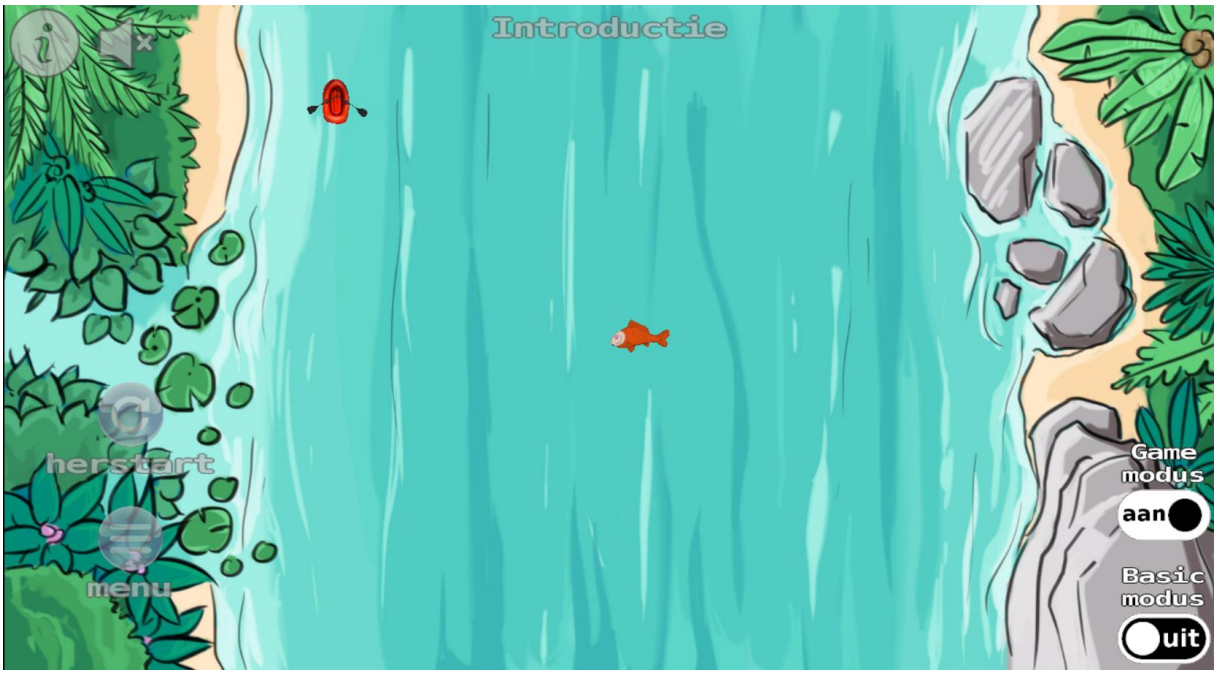
Location de kitchen (Keuken)

Location the island overview(Eiland-overzicht)

Location the Riverbank leveloverview (Rivieroever- leveloverzicht)



Location Riverbank (Rivieroever)



Location de boat (Boot)

Additional images of new features

Settings screen

Progression menu

De kitchen menu for meals

A6 – Source code Unity2D

boatController.cs

```
Using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class boatController : MonoBehaviour
{
    //check every frame for input, and update it

    public float speed;
    private Vector2 initialPosition; //safe initial position
    private Vector2 currPosition; // updated position
    private int rotate;
    private bool up, down, left, right;

    private void Start()
    {
        //safe initial position for the boat for further implementation and restart
        initialPosition = new Vector2(transform.localPosition.x,
transform.localPosition.y);
        currPosition = initialPosition;
        up = false;
        down = false;
        left = false;
        right = false;
    }

    private void Update() //maybe check fixedupdate also
    {
        if (Input.GetKey(KeyCode.LeftArrow))
        {
            if (left != true) {
                currPosition.x -= speed;
            }
            rotate = 90;
        }
        else if (Input.GetKey(KeyCode.UpArrow))
        {
            if (up != true)
            {
                currPosition.y += speed;
            }
            rotate = 0;
        }
        else if (Input.GetKey(KeyCode.RightArrow))
        {
            if (right != true)
            {
                currPosition.x += speed;
            }
            rotate = 270;
        }
        else if (Input.GetKey(KeyCode.DownArrow))
        {
            if (down != true)
            {
                currPosition.y -= speed;
            }
```

```
            rotate = 180;
        }

        transform.localPosition = currPosition;
        transform.rotation = Quaternion.Euler(0, 0, rotate); //directly alter the z-
axis rotation


    }

    private void OnCollisionEnter2D(Collision2D collision) //detect which wall is
being collided with
    {
        if (collision.collider.gameObject.name.Contains("Up"))
        {
            up = true;
        }
        if (collision.collider.gameObject.name.Contains("Down"))
        {
            down = true;
        }
        if (collision.collider.gameObject.name.Contains("Left"))
        {
            left = true;
        }
        if (collision.collider.gameObject.name.Contains("Right"))
        {
            right = true;
        }
    }

    private void OnCollisionExit2D(Collision2D collision)
    {
        if (collision.collider.gameObject.name.Contains("Up"))
        {
            up = false;
        }
        if (collision.collider.gameObject.name.Contains("Down"))
        {
            down = false;
        }
        if (collision.collider.gameObject.name.Contains("Left"))
        {
            left = false;
        }
        if (collision.collider.gameObject.name.Contains("Right"))
        {
            right = false;
        }
    }

    //function here to restart the game, should be called from elsewhere
    public void restart()
    {
        currPosition = initialPosition;
    }
}
```

BootShowProgression.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BootShowProgression : MonoBehaviour {

        private GameObject Left;
        private GameObject Right;
        private int counter; //odd is false, even is true

        private void Start()
        {
            Left = transform.GetChild(0).gameObject;
            Right = transform.GetChild(1).gameObject;
            Left.SetActive(false);
            Right.SetActive(false);
            counter = 1;
        }

    private void OnMouseDown()
    {
        counter++;
        if (counter % 2 == 0)
        {
            Left.SetActive(true);
            Right.SetActive(true);
        }
        else
        {
            Left.SetActive(false);
            Right.SetActive(false);
        }
    }

}
```

ClickToContinue.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ClickToContinue : MonoBehaviour {

    private void OnMouseDown()
    {
        informationPopup informationPopup =
GameObject.FindGameObjectWithTag("ClickToContinue").GetComponent<informationPopup>();
        informationPopup.ResetPopup();
    }
}
```

CloseSettings.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class CloseSettings : MonoBehaviour {

    private Button close;

    private void Start()
    {
        close = GetComponent<Button>();
        close.onClick.AddListener(CloseWindow); // on click call method on CloseWindow
    }

    private void CloseWindow() //when close button is clicked
    {
        ConfigurationPopup config =
GameObject.FindGameObjectWithTag("ConfigurationButton").GetComponent<ConfigurationPopu
p>();
        config.ResetPopup();
    }
}
```

ConfigurationPopup.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ConfigurationPopup : MonoBehaviour {

    //variables
    private Vector3 safeTransform; //safe initial scale values
    private Vector3 tempTransform; //new temporary increased state
    public float increaseScale;
    public GameObject ConfiguratieButtonBox;
    private object[] colliders;

    // Use this for initialization
    void Start()
    {
        safeTransform = new Vector3(transform.localScale.x, transform.localScale.y,
transform.localScale.z); //safe initial scale values
        tempTransform = new Vector3(transform.localScale.x + increaseScale,
transform.localScale.y + increaseScale, transform.localScale.z);
        ConfiguratieButtonBox.SetActive(false);
    }

    //while hover over colliders is true execute code
    private void OnMouseOver()
    {
        transform.localScale = tempTransform;
    }

    private void OnMouseDown()
    {
        ConfiguratieButtonBox.SetActive(true);
```

```
        colliders = FindObjectsOfType(typeof(Collider2D)); //get an array of colliders
in whole scene
        foreach (object c in colliders) //execute code for each element c in colliders
        {
            Collider2D col = (Collider2D)c; //typecast all colliders

            if (!col.CompareTag(ConfiguratieButtonBox.tag))
            {
                col.enabled = false;// disable all colliders in the scene
            }

        }
    }

    private void OnMouseExit()
    {
        transform.localScale = safeTransform; //reset transform
    }

    public void ResetPopup() //this function is called from outside to reset colliders
etc
    {

        ConfiguratieButtonBox.SetActive(false);//also collider not active then

        foreach (object c in colliders) //execute code for each element c in colliders
        {
            Collider2D col = (Collider2D)c; //typecast all colliders
            col.enabled = true;// enable all colliders in the scene
        }
    }


}
```

informationPopup.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class onClickVoortgang : MonoBehaviour {

    private bool hover;
    private int counter;

    private void Start()
    {
        hover = false;
        counter = 0;
    }

    private void Update()
    {
        if (Input.GetMouseButtonDown(0) && hover == false)
        {

            if (counter == 1) {
                //attempt to reset Voortgang menu
                onHoverBootHut onHoverBootHut =
GameObject.FindGameObjectWithTag("BgBoot").GetComponent<onHoverBootHut>();
```

```
                onHoverBootHut.resetVoortgang();
                counter = -1;
            }
            counter++;
        }

     }

    // Update is called once per frame
    private void OnMouseOver()
    {
        hover = true;
    }

    private void OnMouseExit()
    {
        hover = false;
    }

}
```

onClickMenu.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class onClickMenu : MonoBehaviour {

    private bool hover;
    private int counter;


    private void Start()
    {
        hover = false;
        counter = 0;

    }

    private void Update()
    {
        if (Input.GetMouseButtonDown(0) && hover == false)
        {
            if (counter == 1)
            {
                //attempt to reset Voortgang menu
                onHoverCampfire onHoverCampfire =
GameObject.FindGameObjectWithTag("BgKeuken").GetComponent<onHoverCampfire>();
                onHoverCampfire.resetMenu();
                counter = -1;
            }
            counter++;
        }

    }

    // Update is called once per frame
    private void OnMouseOver()
    {
```

```
        hover = true;
    }



    private void OnMouseExit()
    {
        hover = false;
    }

}
```

onClickVoortgang.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class onClickVoortgang : MonoBehaviour {

    private bool hover;
    private int counter;

    private void Start()
    {
        hover = false;
        counter = 0;
    }

    private void Update()
    {
        if (Input.GetMouseButtonDown(0) && hover == false)
        {

            if (counter == 1) {
                //attempt to reset Voortgang menu
                onHoverBootHut onHoverBootHut =
GameObject.FindGameObjectWithTag("BgBoot").GetComponent<onHoverBootHut>();
                onHoverBootHut.resetVoortgang();
                counter = -1;
            }
            counter++;
        }

     }

    // Update is called once per frame
    private void OnMouseOver()
    {
        hover = true;
    }

    private void OnMouseExit()
    {
        hover = false;
    }

}
```

onHoverArrowNavigation.cs

```
using UnityEngine;
using UnityEngine.UI;

/*
 * Jordi Weldink
 *
 * onHoverArrowNavigation:
 * This script handles behaviour concerning the hovering over navigation arrows at the
'Hutten'.
 * On hovering the Text will be set active, arrow scale increase. Furthermore a grey
transparent overlay
 * will even put more emphasis on the arrow
 */


public class onHoverArrowNavigation : MonoBehaviour
{
    //variables
    public Text ArrowText; //text reference from arrow
    private Vector3 safeScale; //safe initial scale values
    private Vector3 tempScale; //new temporary increased state
    private Vector3 safePosition; //safe initial depth
    private Vector3 tempPosition; //contains temporary new depth
    private float decreaseDepth = -5F;
    private float increase = 0.05F; //the increase in scale

    //initialize once
    void Start()
    {
        safeScale = new Vector3(transform.localScale.x, transform.localScale.y,
transform.localScale.z); //safe initial scale values
        tempScale = new Vector3(transform.localScale.x + increase,
transform.localScale.y + increase, transform.localScale.z); //safe increased version
        safePosition = new Vector3(transform.localPosition.x,
transform.localPosition.y, transform.localPosition.z);
        tempPosition = new Vector3(transform.localPosition.x,
transform.localPosition.y, transform.localPosition.z + decreaseDepth);
        ArrowText.gameObject.SetActive(false);

    }

    //when hover is true
    private void OnMouseOver()
    {
        ArrowText.gameObject.SetActive(true);
        transform.localScale = tempScale; //change to temporary state
        transform.localPosition = tempPosition;
    }

    //when hover becomes false
    private void OnMouseExit()
    {
        ArrowText.gameObject.SetActive(false);
        transform.localScale = safeScale; //reset transform
        transform.localPosition = safePosition;
    }

}
```

onHoverBootHut.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class onHoverBootHut : MonoBehaviour {


    public GameObject Voortgang; //the horizontal wooden strip
    public GameObject VoortgangCanvas;
    private GameObject Outline; //outline of cabin
    private GameObject Header; //top part of voortgang menu
    private GameObject Left; //left and right are part of the progress menu
    private GameObject Right;
    private bool active;
    private Collider2D col;
    private Collider2D colHut;

    private void Start()
    {
        Outline = transform.GetChild(0).gameObject;
        Outline.SetActive(false); //disable the outline
        Voortgang.gameObject.SetActive(false); //disable the header
        VoortgangCanvas.gameObject.SetActive(false);
        colHut = GetComponent<Collider2D>();
        col = Voortgang.GetComponent<Collider2D>();
        col.enabled = false;
        Left = Voortgang.transform.GetChild(0).gameObject;
        Right = Voortgang.transform.GetChild(1).gameObject;
        Header = Voortgang.transform.GetChild(2).gameObject;
    }

    private void OnMouseDown() // on click
    {
        if (active == false) {
            Voortgang.gameObject.SetActive(true); //show entire voortgang menu
            VoortgangCanvas.gameObject.SetActive(true);
            Header.transform.GetChild(0).gameObject.SetActive(false);// show voortgang
with text child 1
            Header.transform.GetChild(1).gameObject.SetActive(true);
            Left.SetActive(true);//disable rest of menu
            Right.SetActive(true);
            col.enabled = true;
            colHut.enabled = false;
            active = true;
        }

    }

    private void OnMouseOver()// on hover
    {

        Outline.SetActive(true); //show outline
        if (active == false) {// if odd clicks
            Voortgang.SetActive(true);
            Header.transform.GetChild(0).gameObject.SetActive(true);//set child 0 text
on true
            Header.transform.GetChild(1).gameObject.SetActive(false);
            Left.SetActive(false);//disable rest of menu
            Right.SetActive(false);
```

```
        }
    }

    private void OnMouseExit() // on leaving mouse
    {
        Outline.SetActive(false); //hide outline
        if (active == false)
        {
            Voortgang.SetActive(false);
            VoortgangCanvas.gameObject.SetActive(false);
        }
    }

    public void resetVoortgang()
    {
        if (active == true)
        {
            Voortgang.gameObject.SetActive(false); //hide entire menu
            VoortgangCanvas.gameObject.SetActive(false);
            Header.transform.GetChild(0).gameObject.SetActive(true);//set child 0 text
on true
            Header.transform.GetChild(1).gameObject.SetActive(false);
            Left.SetActive(false);//disable rest of menu
            Right.SetActive(false); ;
            active = false;
            col.enabled = false;
            colHut.enabled = true;
        }
    }

}
```

onHoverCampfire.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class onHoverCampfire : MonoBehaviour {

    public GameObject Menu; //the horizontal wooden strip
    private GameObject Outline; //outline of cabin
    private GameObject Header; //top part of voortgang menu
    private bool active;
    private Collider2D col;
    private Collider2D ColCampfire;

    private void Start()
    {
        Outline = transform.GetChild(0).gameObject;
        Outline.SetActive(false); //disable the outline
        Header = GetComponent<GameObject>();
        Header = transform.GetChild(1).gameObject;
        Header.gameObject.SetActive(false); //disable the information header
        Menu.gameObject.SetActive(false); //disable the header
        ColCampfire = GetComponent<Collider2D>();
        col = Menu.GetComponent<Collider2D>();
        col.enabled = false;
        ColCampfire.enabled = true;
    }

    private void OnMouseDown() // on click
    {
```

```
        if (active == false)
        {
            Menu.gameObject.SetActive(true); //show entire voortgang menu
            col.enabled = true;
            ColCampfire.enabled = false;
            active = true;
        }

    }

    private void OnMouseOver()// on hover
    {

        Outline.SetActive(true); //show outline
        Header.SetActive(true);

    }

    private void OnMouseExit() // on leaving mouse
    {
        Outline.SetActive(false); //hide outline
        Header.SetActive(false);
        if (active == false)
        {
            Menu.SetActive(false);
        }
    }

    public void resetMenu()
    {
        if (active == true)
        {
            Menu.gameObject.SetActive(false); //show entire voortgang menu
            active = false;
            col.enabled = false;
            ColCampfire.enabled = true;
        }
    }
}
```

onHoverHutten.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class onHoverHutten : MonoBehaviour {

    private GameObject Outline;
    private GameObject textCloud;
    public GameObject textBox;
    public Text header;
    public Text text;

    private void Start()
    {
        textCloud = transform.GetChild(0).gameObject;
        Outline = transform.GetChild(1).gameObject;
        text.gameObject.SetActive(false);
        header.gameObject.SetActive(false);
```

```
        textCloud.SetActive(false);
        textBox.SetActive(false);
        Outline.SetActive(false);
    }

    private void OnMouseOver()
    {
        //show child outline around the cabin
        text.gameObject.SetActive(true);
        header.gameObject.SetActive(true);
        Outline.SetActive(true);
       textCloud.SetActive(true);
        textBox.SetActive(true);
    }

    private void OnMouseExit()
    {
        //hide outline around the specified cabin
        text.gameObject.SetActive(false);
        header.gameObject.SetActive(false);
        Outline.SetActive(false);
        textCloud.SetActive(false);
        textBox.SetActive(false);
    }

}
```

onHoverInventory.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class onHoverInventory : MonoBehaviour {

    public GameObject popup;
    private Text text;

    private void Start()
    {
        popup.SetActive(false);
        text = popup.transform.GetChild(0).GetComponent<Text>();

    }

    //when hover is true
    private void OnMouseOver()
    {
        popup.gameObject.SetActive(true);
        text.text = tag;
    }

    //when hover becomes false
    private void OnMouseExit()
    {
        popup.gameObject.SetActive(false);
        text.text = "";
    }
```

```
}
```

onHoverMeal.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class onHoverMeal : MonoBehaviour {

    private GameObject layer;


       // Use this for initialization
       void Start () {
        layer = transform.GetChild(0).gameObject;
        layer.SetActive(false);
    }

    private void OnMouseOver()
    {
        layer.SetActive(true);

    }

    private void OnMouseExit()
    {
        layer.SetActive(false);
    }

    private void OnMouseDown()
    {
        PopupController popupcontroller =
GameObject.FindGameObjectWithTag("Popup").GetComponent<PopupController>();
        popupcontroller.restart();
    }
}
```

onHoverMoestuinTools.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class onHoverMoestuinTools : MonoBehaviour {

    //variables
    private Vector3 safeTransform; //safe initial scale values
    private Vector3 tempTransform; //new temporary increased state
    public float increaseScale;
    private Transform tool;



    // Use this for initialization
    void Start()
    {
        //tool = transform.GetComponentInChildren<SpriteRenderer>().transform;
        tool = transform.GetChild(0).gameObject.transform;
```

```
        safeTransform = new Vector3(tool.localScale.x, tool.localScale.y,
tool.localScale.z); //safe initial scale values
        tempTransform = new Vector3(tool.localScale.x + increaseScale,
tool.localScale.y + increaseScale, tool.localScale.z);
    }

    //while hover over colliders is true execute code
    private void OnMouseOver()
    {
        tool.localScale = tempTransform;
    }

    private void OnMouseExit()
    {
        tool.localScale = safeTransform; //reset transform
    }

    private void OnMouseDown()
    {
        PopupController popupcontroller =
GameObject.FindGameObjectWithTag("Popup").GetComponent<PopupController>();
        popupcontroller.restart();
    }

}
```

onHoverROlevel.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class onHoverROlevel : MonoBehaviour {

    //variables
    private Vector3 safeTransform; //safe initial scale values
    private Vector3 tempTransform; //new temporary increased state
    public float increaseScale;
    public int increaseFont;
    public Text text;
    private int safeFont;
    private int tempFont;

    // Use this for initialization
    void Start()
    {
        //tool = transform.GetComponentInChildren<SpriteRenderer>().transform;
        safeTransform = new Vector3(transform.localScale.x, transform.localScale.y,
transform.localScale.z); //safe initial scale values
        tempTransform = new Vector3(transform.localScale.x + increaseScale,
transform.localScale.y + increaseScale, transform.localScale.z);
        safeFont = text.fontSize;
        tempFont = text.fontSize + increaseFont;
    }

    //while hover over colliders is true execute code
    private void OnMouseOver()
    {

        transform.localScale = tempTransform;
```

```
        text.fontSize = tempFont;
    }

    private void OnMouseExit()
    {

        transform.localScale = safeTransform; //reset transform
        text.fontSize = safeFont;
    }
}
```

onMouseDown.cs

```
using UnityEngine;
using UnityEngine.SceneManagement;



/*
 onMouseDown:
 Teleports user to another scene when clicking on a collider by using tags
 All navigation elements like arrows and cabins contain it
 */

public class onMouseDown : MonoBehaviour {

    //if clicked on the collider
    private void OnMouseDown()
    {
        //add a delay in here?
        SceneManager.LoadScene(tag.ToString());
    }
}
```

onRestart.cs

```
using UnityEngine;
using UnityEngine.UI;

public class onRestart : MonoBehaviour {


    //if pressed restart, call public function in boatController which resets the
position to initial.
    private void OnMouseDown()
    {
        boatController boatController =
GameObject.FindGameObjectWithTag("boat").GetComponent<boatController>();
        boatController.restart();
        PopupControllerRivieroever popupcontroller =
GameObject.FindGameObjectWithTag("Popup").GetComponent<PopupControllerRivieroever>();
        popupcontroller.restart();
    }
}
```

PopupController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class PopupController : MonoBehaviour
{

    public float duration; //how long remain on screen
    public float fade; //time in order to fade
    public float countdown; //time of starting fade
    private float initialAlpha;
    private float stopwatch;
    private float totalDuration;
    private bool showPopup;

    //Reset all alphaValues and start popup sequence
    private void Start()
    {
        initialAlpha = transform.GetComponent<Image>().color.a;
        ResetValues(0);
        totalDuration = countdown + 2 * fade + duration;
        stopwatch = totalDuration;
    }

    private void Update() //Keeps executing each frame
    {

        if (stopwatch >= 0.0F) {
            stopwatch -= Time.deltaTime; //at each frame countdown time


            if (stopwatch <= totalDuration - countdown && stopwatch >= totalDuration -
countdown - fade)
            {
                ActivateFade(totalDuration - stopwatch - countdown, 1);
            }
            else if (stopwatch <= totalDuration - countdown - fade - duration &&
stopwatch > 0.0F)
            {
                ActivateFade(totalDuration - stopwatch - countdown - fade - duration,
0);
            }
        }
    }



    //Fade in or out depending on the alpha value
    private void ActivateFade(float t, int a)
    {
        float AlphaColor = initialAlpha;
        float temp = t;

        if (a == 1) //fade in
        {
            AlphaColor = temp / fade * initialAlpha;
        }
        else if (a == 0) //fade out
        {
```

```
            AlphaColor = initialAlpha - (temp / fade * initialAlpha);
        }

        ResetValues(AlphaColor);
    }

    private void ResetValues(float alpha)
    {
        //reset background
        Color bg = transform.GetComponent<Image>().color;
        bg.a = alpha;
        transform.GetComponent<Image>().color = bg;
        //reset Text
        Color text = transform.GetChild(0).GetComponent<Text>().color;
        text.a = alpha;
        transform.GetChild(0).GetComponent<Text>().color = text;
    }


    //Do not call other functions from here, then update() will not be overrided
    public void restart()
    {

        if (stopwatch != totalDuration || stopwatch <= 0.0F &&
transform.GetComponent<Image>().color.a == 0) //dont interfere with current popup
animation
        {
            stopwatch = totalDuration;
        }
    }
}
```

PopupControllerRivieroever.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class PopupControllerRivieroever : MonoBehaviour {

    public float duration; //how long remain on screen
    public float fade; //time in order to fade
    public float countdown; //time of starting fade
    private float initialAlpha;
    private float stopwatch;
    private float totalDuration;

    //Reset all alphaValues and start popup sequence
    private void Start()
    {
        initialAlpha = transform.GetComponent<Image>().color.a;
        resetValues(0);
        totalDuration = countdown + 2 * fade + duration;
        stopwatch = totalDuration;
    }

    private void Update() //Keeps executing each frame
    {
        stopwatch -= Time.deltaTime; //at each frame countdown time
```

```
        if (stopwatch <= totalDuration - countdown && stopwatch >= totalDuration -
countdown - fade)
        {
            ActivateFade(totalDuration - stopwatch - countdown, 1);
        }
         else if (stopwatch <= totalDuration - countdown - fade - duration &&
stopwatch > 0.0F)
        {
            ActivateFade(totalDuration - stopwatch - countdown - fade - duration, 0);
        } else if (stopwatch <=0.0F)
        {
            Debug.Log("still doing stuff");
        }
    }


    //Fade in or out depending on the alpha value
    private void ActivateFade(float t, int a)
    {
        float AlphaColor = initialAlpha;
        float temp = t;

        if (a == 1) //fade in
        {
            AlphaColor = temp / fade * initialAlpha;
        }
        else if (a == 0) //fade out
        {
            AlphaColor = initialAlpha - (temp / fade * initialAlpha);
        }

        resetValues(AlphaColor);
    }

    private void resetValues(float alpha)
    {
        //reset background
        Color bg = transform.GetComponent<Image>().color;
        bg.a = alpha;
        transform.GetComponent<Image>().color = bg;
        //reset Text
        Color text = transform.GetChild(0).GetComponent<Text>().color;
        text.a = alpha;
        transform.GetChild(0).GetComponent<Text>().color = text;
        //reset sprite
        Color sprite = transform.GetChild(1).GetComponent<SpriteRenderer>().color;
        sprite.a = alpha;
        transform.GetChild(1).GetComponent<SpriteRenderer>().color = sprite;
    }


    //Do not call other functions from here, then update() will not be overrided
    public void restart()
    {
        if (stopwatch != totalDuration || stopwatch <= 0.0F &&
transform.GetComponent<Image>().color.a == 0) //dont interfere with current popup
animation
        {
            stopwatch = totalDuration;
        }
    }
}
```

A7 – replace index and style code

Replacing the index.html code:
Go to the folder build > index.html, and open it in a text editor. Then delete the existing code and replace it by the code underneath:

New code Index.html

```
<!DOCTYPE html>
<html lang="en-us">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>Unity WebGL Player | StrandedRedesign1.0</title>
    <link rel="shortcut icon" href="TemplateData/favicon.ico">
    <link rel="stylesheet" href="TemplateData/style.css">
    <script src="TemplateData/UnityProgress.js"></script>
    <script src="Build/UnityLoader.js"></script>
    <script>
      var gameInstance = UnityLoader.instantiate("gameContainer", "Build/build.json", {onProgress:
UnityProgress});
    </script>
  </head>
  <body>
    <div class="webgl-content">
      <div id="gameContainer" style="position:absolute; left:0px; top:0px; min-height:100%; min-
width:100%; max-height:100%; max-width:100%; height:auto; width:auto;"></div>
    </div>
  </body>
</html>
```

Replacing the style.css code:
Go to the folder build > TemplateData > style.css, and open it in a text editor. Then delete the existing code and replace it by the code underneath:

New code style.css:

```
* {
        margin:0;
        padding:0;
}

canvas {
        position:absolute; left:0px; top:0px; min-height:100%; min-width:100%; max-
        height:100%; max-width:100%; height:auto; width:auto;
}
```

References

Ritterfeld, U., Cody, M.J., Vorderer, P.: Serious games: Mechanics and effects. Taylor & Francis(2009). Page 3

Nap, H.H., de kort, Y.A.W., & ijsselsteijn, W.A. 2009. Senior gamers: Preferences, motivations and needs. Gerontechnology, 8(4), 247-262.

Warburton, D. E. R., Nicol, C. W., & Bredin, S. S. D. (2006). Health benefits of physical activity: The evidence. Cmaj, 174(6), 801–809. https://doi.org/10.1503/cmaj.05135

Rockmann, R. & Gerwald, H.(2015). Elderly people in eHealth: who are they?. *The 1st* International Workshop on Adoption and Usage of Information and Communication Technologies in Healthcare by the Elderly: vol. 63 (pp. 505-510). https://doi.org/10.1016/j.procs.2015.08.376

Johnson, R. & Kent, S.(2007) Designing universal access: web-applications for the elderly and disabled. Cogn Tech Work 9: 209. https://doi.org/10.1007/s10111-007-0063-2

Jokela, Timo & Iivari, Netta & Matero, Juha & Karukka, Minna. (2003). The standard of user-centered design and the standard definition of usability. 53. 10.1145/944519.944525.

Wiemeyer, J., & Kliem, A. (2012). Serious games in prevention and rehabilitation-a new panacea for elderly people? *European Review of Aging and Physical Activity*, *9*(1), 41–50. https://doi.org/10.1007/s11556-011-0093-x

Gerling, K. M., Schulte, F. P., Smeddinck, J., & Masuch, M. (2012). Game Design for Older Adults: Effects of Age-Related Changes on Structural Elements of Digital Games, 235–242. https://doi.org/10.1007/978-3-642-33542-6_20

W, I. J., Nap, H. H., De Kort, Y., & Poels, K. (2007). Digital game design for elderly users. *Proceedings of the 2007 Conference on Future Play, Future Play '07*, 17–22. https://doi.org/10.1145/1328202.1328206

Nielsen, J. (2005). 10 Usability Heuristics for User Interface Design. *Conference Companion on Human Factors in Computing Systems CHI 94*, 152–158. https://doi.org/10.1145/191666.191729

Davis, K., & Douglas, D. (2015). Translating usability test data into client-ready findings and recommendations. Proceedings of the Human Factors and Ergonomics Society, 2015–Janua, 1764–1768. https://doi.org/10.1177/1541931215591381

Molich, R., Jeffries, R., & Dumas, J. S. (2007). Making Usability Recommendations Useful and Usable. Journal of Usability Studies, 2(4), 162–179. Retrieved from http://dl.acm.org/citation.cfm?id=2835552.2835554

Middleton, B., Bloomrosen, M., Dente, M. A., Hashmat, B., Koppel, R., Overhage, J. M., … Zhang, J. (2013). Enhancing patient safety and quality of care by improving the usability of electronic health record systems: Recommendations from AMIA. Journal of the American Medical Informatics Association, 20(E1). https://doi.org/10.1136/amiajnl-2012-001458

Schell, J.(2015). Good Games Are Created Through Playtesting. The Art of Game Design. A book of Lenses (second edition)

Mader, A., & Eggink, W. (2014). A Design Process for Creative Technology. Proceedings of the 16th International Conference on Engineering and Product Design Education; University of Twente, (September), 1–6.

Snyder, C. (2001). Paper Prototyping. *Paper Prototyping*, 197–219. https://doi.org/10.1016/B978-155860870-2/50036-X

Walker, M., Takayama, L., & Landay, J. A. (2002). High-Fidelity or Low-Fidelity, Paper or Computer? Choosing Attributes when Testing Web Prototypes. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *46*(5), 661–665. https://doi.org/10.1177/154193120204600513

Sefelin, R., Tscheligi, M., & Giller, V. (2003). Paper prototyping - what is it good for? *CHI '03 Extended Abstracts on Human Factors in Computing Systems - CHI '03*, 778. https://doi.org/10.1145/765891.765986

Sauli, L. (2006). Do Usability Expert Evaluation and Testing Provide Novel and Useful Data For Game Development? *Journal of Usability Studies*, *1*(2), 64–75. https://doi.org/10.1145/985921.986102

Aurora Harley (2018, February 25). Nielsen Norman Group: UX Expert Reviews. Retrieved from: https://www.nngroup.com/articles/ux-expert-reviews/

Clegg, D. & Barker, R. (1994). CASE method fast-track. A RAD approach. Addison-Wesley. ISBN 978-0-201-62432-8.

David Travis (2016, June 6). What user researchers ought to know about informed consent. Retrieved from: https://www.userfocus.co.uk/articles/what_user_researchers_ought_to_know_about_informed_consent.html

Brooke, J. (1996). "SUS: A "quick and dirty" usability scale". In *Usability evaluation in industry*, Edited by: Jordan, P. W., Thomas, B. A. Weerdmeester and McClelland, I. L. 189–194. London: Taylor & Francis.