UiT The Arctic University of Norway

Faculty of Science and Technology

Department of Computer Science

Implementing Motivational Features for an Augmented Reality Game Encouraging Physical Activity for Persons with Intellectual Disabilities

Thomas Luzi

Master's thesis in Computer Science INF-3981 June 2022

Acknowledgements

I would like to start by thanking my main supervisor, André Henriksen, for his continuous guidance throughout this project and advice during the writing process. I would also like to thank my co-supervisors, Henriette Michalsen, Audny Anke, Gunnar Hartvigsen, Erlend Johannesen, Santiago Martinez, and Susanna Pelagatti, for providing valuable feedback and interesting discussions during my supervision meetings.

Lastly, I would like to thank my co-students who have become dear friends during the last five years. Our journey shall continue to the capital of Norway. Hear, hear!

Abstract

Evidence shows lower physical activity (PA) levels in persons with an intellectual disability (ID) compared to the general population. Physical activity promotes multiple health benefits, such as prolonged life and reduced risk for obtaining various chronic diseases. There are numerous applications targeting increased physical activity for the general population, but these tend to be too complex to use, and have therefore not been adopted, by persons with ID. Exergames promote increased physical activity by utilizing the entertaining aspects of games as a motivator, and some research has been explicitly aimed toward providing such a platform for persons with ID. One application targeting increased PA levels in persons with ID is Sorterius.

While Sorterius promotes PA during utilization, it is somewhat limited in providing motivational elements to its users, and an increase in motivational elements could greatly increase usage of the application. Therefore, the main goal of this project was to enhance the overall enjoyment of playing Sorterius. To reach this goal, the project utilized knowledge gained from a prior literature review assessing which game aspects persons with ID perceived as motivating. The project also relied on continuous discussion with experts in the field.

The resulting design includes a customizable mascot where users can unlock new items for their mascot and an automated difficulty adjustment based on user performance. To evaluate the application, a usability test was conducted on two experts and six persons with ID for a duration of two weeks. Due to various reasons, we were only able to retrieve written consent from one of the persons with ID before submitting the thesis.

Results from testing indicated a similar degree of usability as the previous version while providing an increased number of motivational elements towards its users. The new version was also tested on the intended users compared to the previous version.

Table of Contents

List of Tables

List of Figures

Abbreviations

AAIDD - American Association on Intellectual and Developmental Disabilities

API – Application Programmable Interface

AR – Augmented Reality

ASD - Autism Spectrum Disorder

DD – Developmental disability

DDA – Dynamic Difficulty Adjustment

DS – Down syndrome

ID – Intellectual Disability

IDE – Integrated development environment

IQ – Intelligence Quotient

PA – Physical activity

PRISMA – Preferred Reporting Items for Systematic Reviews and Meta-Analyses

WHO – World Health Organization

1 Introduction

1.1 Background

There is strong evidence to suggest the multiple gained benefits from performing physical activity (PA) [1, 2]. Research suggests that performing PA regularly can improve physical and mental health and reduce the risk of developing chronic diseases such as cardiovascular diseases, diabetes, and cancer. Increased PA also reduces symptoms of anxiety and depression and can improve the overall well-being [2]. Physical activity also ensures healthy growth in adolescents. The World Health Organization (WHO) recommends that adults (between 18 and 65 years) spend a weekly minimum of 150-300 minutes of aerobic physical activity with moderate intensity or 75-150 minutes of vigorous intensity. Adults are also recommended to perform muscular training involving all major muscle groups at least twice a week. It is estimated that 25% of adults do not reach these recommended PA levels [2].

Persons with an intellectual disability (ID) are known to be insufficiently physically active [3]. A study conducted by Wouters et al. [4] measured PA levels in 68 children with ID (2-18 years) and found that only 47% of the participants met the recommended levels stated by WHO. Also, Dairo et al. [5] conducted a systematic review of 15 studies assessing PA levels in adults with ID. They found that only 9% of the participants in these studies met the WHO guidelines for PA levels. Both these findings indicate low PA levels in persons with ID, and it is therefore a need to promote increased physical activity for these persons.

Over the last decades, video games have become a popular leisure activity. Combining aspects of games with learning real-life skills and behavior change show promise in providing an entertaining platform for obtaining such skills and behaviors [6-8]. Multiple applications utilizing gamification elements have been developed to promote increased PA levels for the general population [9]. However, these applications have not been adopted by persons with ID as they tend to be too complex to use. There is also limited knowledge of the effects such applications would impose on persons with ID. Still, some research has been explicitly aimed at providing a platform for persons with ID to increase their PA levels. One such application is Sorterius, developed by Stellander [10].

Sorterius is an augmented reality game promoting increased PA levels in persons with ID. The game scenario consists of aiding a mascot in collecting and sorting located trash objects. The application is designed to accommodate varying levels of intellectual disability by providing three difficulty levels. The user interface is designed to be utilized by persons with ID, consisting of large buttons, easy-to-read text, and providing instructions automatically without user interference [11]. The application utilizes a star-point system and weekly rewards to motivate its users to play regularly. While the game does offer some motivational aspects, additional elements could significantly enhance the potential of the application, which will be the main focus area for this thesis.

This project aims to utilize findings from a previous literature review on motivational techniques for persons with ID to increase the overall enjoyment of playing Sorterius [12]. The ambition is to enable Sorterius to be a regularly utilized tool for increasing PA levels in persons with ID.

This thesis describes the process for extending Sorterius.

1.2 Scope and research problem

The overall ambition for this project is to provide an entertaining platform to help persons with ID to reach the recommended PA levels. To fulfill this ambition, the project aims to extend the augmented reality game Sorterius, which is explicitly developed to increase PA levels in persons with ID. The existing version of Sorterius provides a good foundation to reach the project's ambition but it is limited in motivational elements to encourage utilization of the game. To achieve its intent, the application must be frequently used, and it is crucial that the game is perceived as enjoyable and motivating. The main research problem for this project is stated as follows:

RP: How can we motivate persons with ID to increase their PA levels through the use of exergames?

1.2.1 Sub-problems

The main research problem is further divided into three sub-problems.

Sub-problem #1

When introducing a new application to a user, the application must prompt initial curiosity and interest. If a user finds no interest in the game after initial use, the user may not be motivated to revisit the application. Therefore, it is vital to properly introduce users to the application itself and some of the motivational aspects. The former refers to informing users how to operate the application in a comprehensible manner, while the latter refers to establishing interest in revisiting the application. The first sub-problem is stated as follows:

SP1: How can we ensure initial interest when trying Sorterius for the first time?

Sub-problem #2

Secondly, while a user might show initial interest in utilizing the application, a lack of variation in the game is likely to reduce usage after some time. As the application must be used on a regular basis to fulfill its intent, users must find the application entertaining over an extended duration. Therefore, the application should consist of both short-term and long-term goals to preserve users' interest in the game. The second sub-problem is stated as:

SP2: How can we maintain users' interest in Sorterius over an extended period?

Sub-problem #3

As this project aims to add new features, it is important that its users understand these features. If a user finds it difficult to understand an element of the game, it may not be perceived as motivating and might lead to frustration. In addition, the current solution consists of a simple user interface intended for persons with ID. Adding more features to a pre-existing solution could quickly lead to the application becoming overly complicated. The new solution must be coherent and easy to use by its targeted population. New features should therefore complement the overall behavior of the game without imposing complexity. The final sub-problem is stated as:

SP3: How can we add motivational features without affecting the overall perceived complexity of Sorterius?

1.2.2 Assumptions

As no one with ID have tested the existing version created by Stellander [5], we cannot properly evaluate the success of the various elements. However, a usability test was conducted on people working closely with persons with ID. This testing resulted in a total score of 61/100 and was rated between "OK" and "good" [10]. Therefore, for this project, we assumed that the current application was usable by persons with ID. This affected the project's outcome as the existing user interface and functionality were preserved, and new features were designed to be integrated with the existing solution.

As the primary goal of this project is to provide a mobile application to persons with ID, it is assumed that its users have basic skills in operating a mobile device.

1.3 Structure of the Thesis

This section outlines the structure for the remainder of this thesis.

Chapter 1 – Introduction presents a short introduction and states the research-problems for this project

Chapter 2 – Theoretical Framework outlines required background information for this project. This chapter covers topics such as intellectual disability, Serious games, and exergames, and provides an overview of the current implementation of Sorterius and its utilized technology. This chapter also describes a shortened version of the methodology and findings from a literature review on motivational elements for persons with ID, conducted during a previous project [12].

Chapter 3 – Methods presents the methods used for extending Sorterius and how the final solution was evaluated.

Chapter 4 – Requirements states the functional and non-functional requirements made for this project.

Chapter 5 – Design presents the new mascot design and animation features added to Sorterius. It also presents a proposed design for automatically adjusting difficulty levels in Sorterius.

Chapter 6 – Implementation describes the process for achieving a multi-color mascot and implementation-specific details for the added features.

Chapter 7 – Results presents the test results and general feedback. This chapter also presents interesting findings from an observation conducted during testing.

Chapter 8 – Discussion evaluates test results and discusses the proposed design features to the stated research problems. Also proposes future work for Sorterius

Chapter 9 – Conclusion outlines a summary of the project along with concluding remarks

2 Theoretical framework

2.1 Intellectual Disability Overview

Intellectual disability is defined by the American Association on Intellectual and Developmental Disabilities (AAIDD) as “significant limitations both in intellectual functioning and adaptive behavior as expressed in conceptual, social, and practical adaptive skills” [13, 14]. The assessment of whether a person should be considered intellectually disabled are usually done through a series of standardized tests. A score below 70 on a full-scale intelligence quotient (IQ) test indicates limitations in intellectual functioning. Other standardized tests or observations can determine conceptual, social, and practical skills (adaptive behavior) limitations. The disability must also present itself during the developmental period, which was considered until recently (2021) until the age of 18. However, recent studies suggest that the developmental period continues until the age of 22 [13]. It is estimated that persons with ID account for 1-3% of the general population [15].

Subclassification of Intellectual Disability

Intellectual disability can be further grouped into four subclasses based on the severity of the condition, which is assessed based on approximate IQ, everyday skills, and level of support needed. Table 1 describes the categorization for each subclass of ID [16].

Table 1: Subgrouping criteria for intellectual disability

Severity Category and Approximate Percent Distribution of Cases by severity	**Severity levels based only on IQ categories**	**Severity classified on the basis of daily skills**	**Severity classified on the basis of the intensity of support needed)**
Mild 85%	Approximate IQ range 50–69	Can live independently with minimum levels of support.	Intermittent support needed during transitions or periods of uncertainty.
Moderate 10%	Approximate IQ range 36–49	Independent living may be achieved with moderate levels of support.	Limited support needed in daily situations.
Severe 3.5%	Approximate IQ range 20–35	Requires daily assistance with self-care activities and safety supervision.	Extensive support needed for daily activities.

Profound 1.5%	IQ < 20	Requires 24-hour care.	Pervasive support needed for every aspect of daily routines.

IQ: Intelligent quotient.

Functioning of persons with mild ID

As shown in Table 1, the majority (85%) of persons with ID fall under the mild subcategory. They are not always properly diagnosed as the need for support is limited. Persons with mild ID have an approximate IQ ranging between 50 and 69 and have difficulties obtaining and understanding complex language and academic skills [15]. Most acquire the skills needed to interact in day-to-day conversations and, with proper support, can achieve basic literate and mathematical skills [15, 17]. Persons with mild ID can also obtain most of the skills (e.g., personal hygiene, house chores, etc.) required to live independently, allowing them to live with a minimum of external support [15, 17]. Many possess the ability to learn job-related skills such as performing simple tasks and working at scheduled hours and can be employed successfully in practical jobs [15, 17].

Functioning of persons with moderate ID

Approximately 10% of persons with ID are diagnosed with a moderate severity. Persons with Moderate ID have a measured IQ ranging between 35 and 49 [15]. Acquiring basic language skills happens slowly, and persons with moderate ID show significant limitations in obtaining academic skills. Still, some may develop basic academic skills similar to early elementary school levels [15]. Educational programs explicitly aimed are beneficial in educating these persons. With proper support and teaching, persons with moderate ID usually obtain some day-to-day and job skills and exert practical work given continuous overseeing and guidance [15].

Functioning of persons with severe ID

Persons diagnosed with a severe intellectual disability account for approximately 3.5% of the total population with ID. These persons usually have an estimated IQ of 20-25. They are often significantly limited in verbal language, consisting of simple words and often hand gestures for communication. Persons with severe ID require extensive support for day-to-day activities and self-care.

Functioning of persons with profound ID

Approximately 1.5% of persons with ID have a profound intellectual disability. Profound ID is the most severe subclassification, and persons with this diagnosis usually have an estimated IQ below 20-25 [15]. Persons with profound ID require continuous high-intensity care and assistance. They have profound limitations in communicating through verbal language and need alternative communication methods. Profound ID is also associated with other impairments such as motor disabilities affecting their motoric functioning, often limiting independent mobility.

2.2 Serious Game

Games are a highly popular leisure activity with a central focus on entertainment. Even though there is a significant variation between the various available games, they typically consist of presenting a series of challenges to the player, for which the player is rewarded on completion. Serious games aim to utilize the entertaining aspects of games (gamification) to provide a platform for learning real-life skills and imposing behavioral change [18]. The application domain for serious games is many, including education, healthcare, and business.

Exergame

Exergames are a subcategory of serious games which focuses mainly on physical activity interventions [19]. Exergames also aims to integrate gamification aspects to provide an entertaining and motivating platform for increasing physical activity levels. Conducted research shows potential for exergames to increase physical activity levels [20].

2.3 Sorterius

2.3.1 Overview

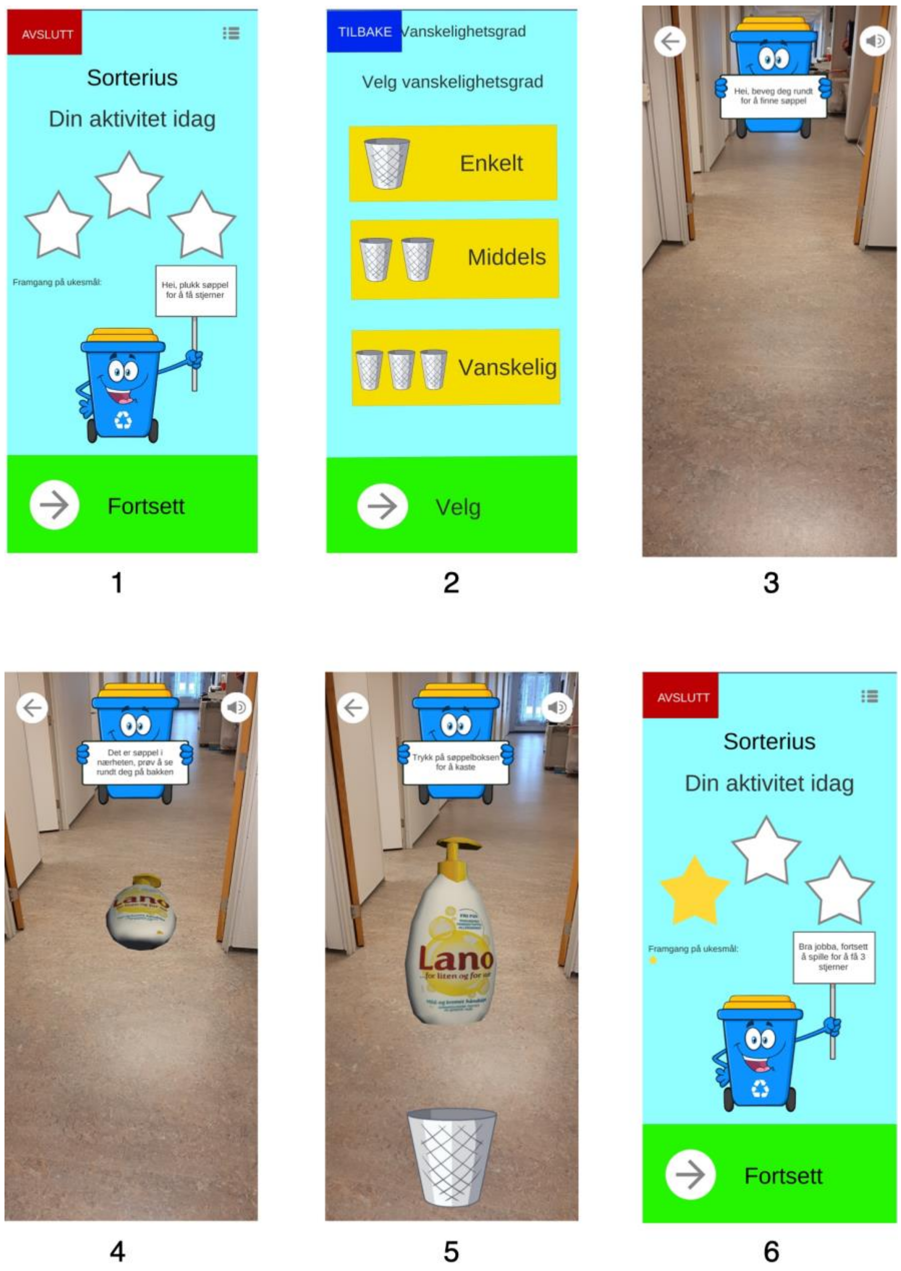
Sorterius is an exergame developed by Stellander [10], targeting increased PA levels for persons with mild to moderate ID. Built using the Unity game engine [21], Sorterius is an augmented reality game that promotes outdoor physical activity by having its users collect and sort garbage. The application utilizes a minimalistic user interface with few elements for each view and large buttons for navigation. Users are rewarded for their effort through a star point system and a custom prize for reaching desired PA levels. Sorterius has been further adapted to support data to be persisted remotely [15]. This solution utilizes mSpider [22] to host the relevant data.

2.3.2 Gameplay

The game consists of a simple plot where the user must assist the game mascot (Sorterius) in collecting garbage in an attempt to rid the world of misplaced waste. During gameplay, the mascot continuously gives the player instructional and positive feedback on their progression, which is provided both as text on-screen and (optionally) through text-to-speech. Players are also rewarded for their effort, receiving stars as they progress towards their daily and weekly goals. Figure 1 illustrates the gameplay as described below.

When the game starts, the player enters the main menu and is greeted by the mascot, encouraging the player to start a new session. The main menu displays both weekly and daily progression and provides the options for entering a new session and navigating to the parent/assistant menu (see section 2.3.5 for a detailed description of the parent/assistant menu). When players want to start a new session, they are first prompted to select the desired difficulty. After the desired difficulty level has been selected, the player may enter the new session.

During a game session, the player is instructed by the mascot to walk and try to locate nearby trash on the ground. The main mobile camera is utilized during gameplay, displaying a real-time feed on-screen as the player moves around. Each step performed by the player is registered through the phone's built-in accelerometer. For every 10-30 recorded steps, the mascot notifies the user of trash located somewhere nearby, and subsequently, the application places a trash object on the ground within reach of the player. The player must tap on the trash object and drop it in a trash bin to collect the newly discovered item. Depending on the selected difficulty level, the game may present several different trash bins requiring the user to drop the trash object in the correct bin (described in detail in section 2.3.4). After the trash object has been successfully collected, the player is instructed to walk and search for more trash objects. The session ends either if a player chooses to exit prematurely or when a certain amount (depends on the difficulty level, see 2.3.4) of trash objects has been collected. Finally, the player is brought back to the main menu and is met with some encouragement from the mascot.



Hei, plukk søppel for å få stjerner

Hei, beveg deg rundt for å finne søppel

Det er søppel i nærheten, prøv å se rundt deg på bakken

Trykk på søppelboksen for å kaste

Bra jobba, fortsett å spille for å få 3 stjerner

Figure 1: Sorterius gameplay

2.3.3 Mascot

The current solution uses a mascot to assist and provide positive feedback during gameplay continuously. The mascot is available in multiple versions, of which some contain a sign used to display information to the user. Figure 2 shows the mascot versions [23].

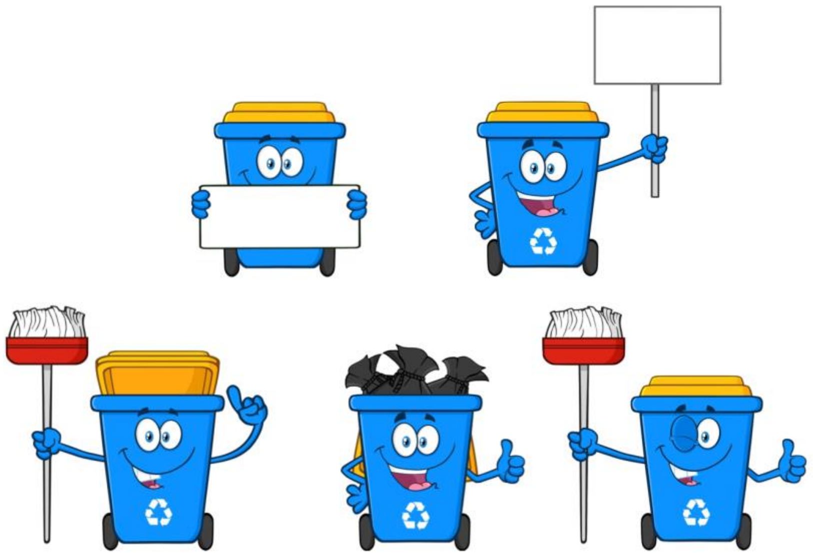


Figure 2: Mascot versions

2.3.4 Levels of Difficulty

To accommodate a diverse group of individuals with varying degrees of cognitive and physical abilities and skills, Sorterius provides players with three different difficulty levels, namely, easy, medium, and hard. The difference between the levels in respect lies with the required number of objects a player must collect to complete a session and the inclusion of multiple trash bins requiring the user to sort the discovered trash objects in their respective bins.

Easy

To complete a session of easy difficulty, a player must be able to collect a predefined amount of 10 trash objects. The game also provides one uncategorized trash bin on this level, so a player can drop all collected items without sorting the respective items.

Medium

On medium difficulty, a player is prompted to select the required amount of trash objects needed to complete the session and is given the option of choosing between 10, 20, or 30 objects. A player is also presented with two possible bins (food and plastic waste), of which they must sort the discovered trash objects accordingly.

Hard

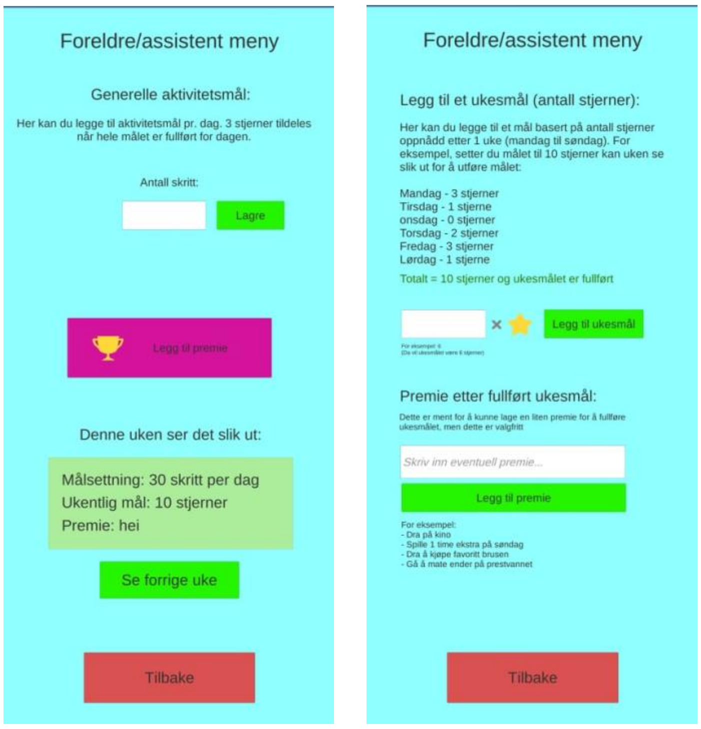
On hard difficulty, a player is prompted with the same option as with medium difficulty regarding the required number of trash objects needed to complete a session. But, while the medium difficulty presents a player with two possible trash bins, players on hard difficulty will have to sort trash between four possible trash bins. The four categories for sorting trash while on hard difficulty are food-, plastic-, glass-, and paper waste.

2.3.5 Assistant Menu

As the population of persons with ID is comprised of individuals with a diverse set of interests and varying physical capacities, Sorterius offers some features to be tailored to each player's needs. The customization of these features resides, in addition to the levels of difficulty, in the assistant menu (see Figure 3). It includes the possibility of setting daily and weekly activity goals and a custom prize a player is awarded for completing their weekly goal.

Daily goals are defined as the number of steps a player must exert to receive the maximum of three (not customizable) stars. The awarding of stars is calculated from a player's achieved step count relative to the daily goal (e.g., earning one star requires a third of the targeted step count, two stars require two-thirds, etc.).

Weekly goals are defined as the total quantity of stars a player must receive within a week (Monday to Sunday) to win their custom prize. The custom award is a textual string that can be set to present the reward a player is to receive outside of the game (e.g., go bowling).



Her kan du legge til aktivitetsmål pr. dag. 3 stjerner tildeles når hele målet er fullført for dagen.

Målsetting: 30 skritt per dag
Ukentlig mål: 10 stjerner
Premie: hei

Her kan du legge til et mål basert på antall stjerner oppnådd etter 1 uke (mandag til søndag). For eksempel, setter du målet til 10 stjerner kan uken se slik ut for å utføre målet:

Mandag - 3 stjerner
Tirsdag - 1 stjerne
onsdag - 0 stjerner
Torsdag - 2 stjerner
Fredag - 3 stjerner
Lørdag - 1 stjerne
Totalt = 10 stjerner og ukesmålet er fullført

For eksempel: 6
(Da vil ukesmålet være 6 stjerner)

Dette er ment for å kunne lage en liten premie for å fullføre ukesmålet, men dette er valgfritt

For eksempel:
- Dra på kino
- Spille 1 time ekstra på søndag
- Dra å kjøpe favoritt brusen
- Gå å mate ender på prestvannet

Figure 3: Sorterius assistant menu

2.3.6 mSpider

mSpider is an experimental system developed by Henriksen et al. [22] for collecting and storing activity data from various consumer-based fitness trackers (e.g., Google, Apple, etc.). mSpider encompasses three main components: a frontend, a backend, and a mobile application. The web frontend is responsible for managing surveys and handling authorization to grant access to users' fitness trackers. The backend server is responsible for handling communication from various cloud storages where activity data of the fitness trackers resides.

The backend server is two-fold, consisting of a document store [24] for storing user and activity data and an Application Programmable Interface (API) for handling mSpider's communication with cloud storage where the activity data is uploaded initially. The backend server was further extended by Luzi [12] to also handle incoming requests from Sorterius and store its data appropriately.

2.4 Related Work

Similar to Sorterius, other applications has been explicitly aimed at increasing PA levels in persons with ID.

2.4.1 Dyrejakten

Dyrejakten is an augmented reality game created by Haugland [25]. The game scenario consists of having users collect various animals during walks. The application provides users with some options during gameplay, such as animal types to be collected (pig, sheep, cow, and chicken) and texture to be displayed on the ground (grass or lava). Before starting a new game session, users are prompted to select the number of animals to be collected before completing the session. Motivational aspects of the game are provided primarily by rewarding users with a medal after completing a game session. Figure 4 Shows the gameplay for Dyrejakten [25].

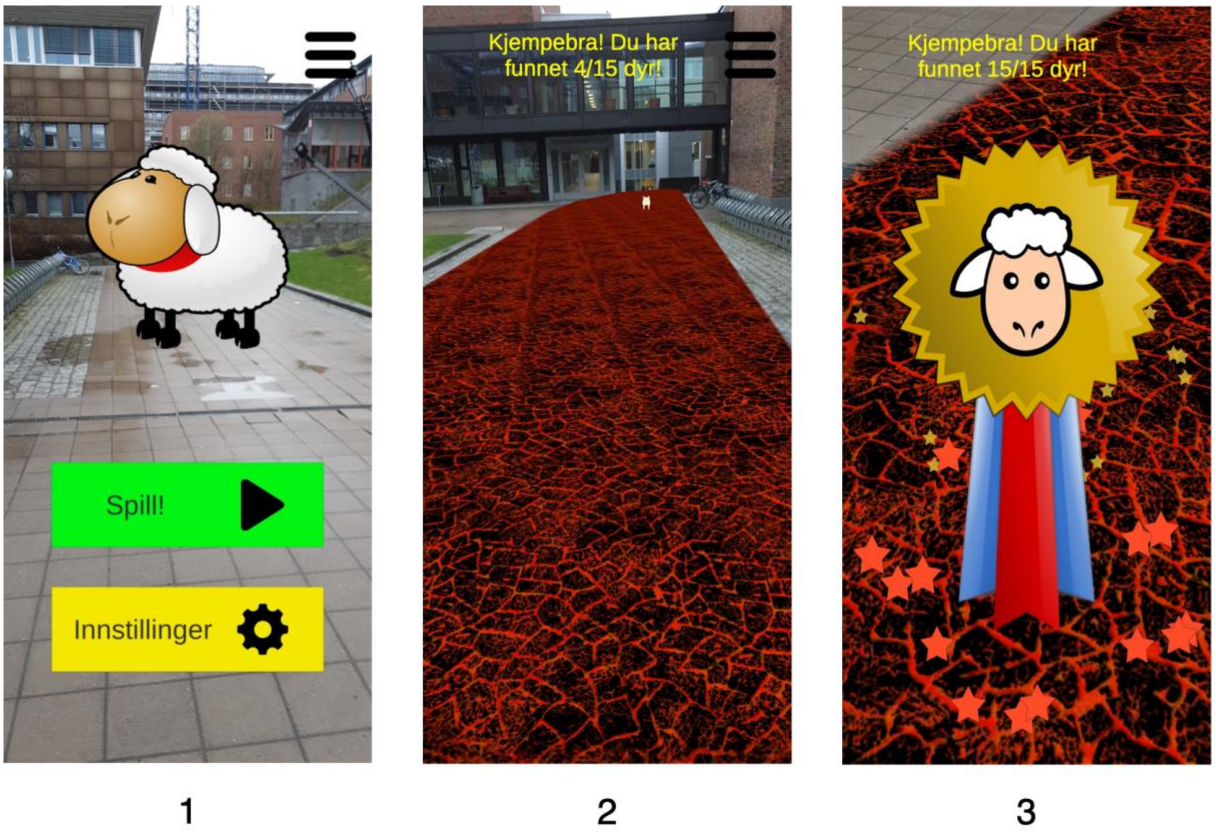

Spill!
Innstillinger
Kjempebra! Du har funnet 4/15 dyr!
Kjempebra! Du har funnet 15/15 dyr!
1
2
3

Figure 4: Dyrejakten gameplay

2.4.2 Activity Game Avatar (AGA)

AGA is an exergame initially created by Wiik [26] and later extended by Eilertsen [27]. The game consists of a 3D avatar demonstrating how to perform various dances and training exercises. The game supports multiple users in which each user can customize the 3D avatar to their preferred look. In addition, AGA provides a leaderboard where users on the same device can compete. Other motivational features include a star-point system where users are rewarded with animal companions for achieving a certain number of stars. Figure 5 shows the gameplay for AGA [26, 27].

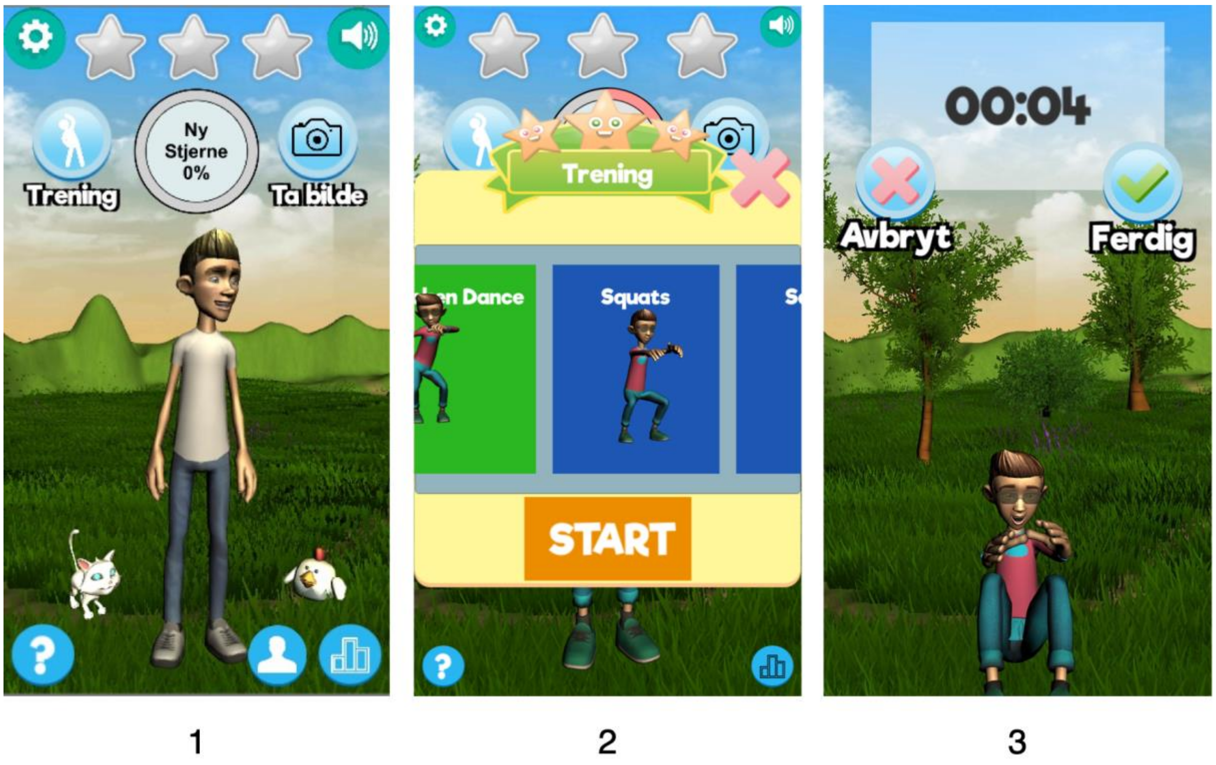


Figure 5: AGA gameplay

2.5 Technology - Unity

2.5.1 Unity

Unity [21] is a cross-platform engine supporting development for various game types. The game engine supports development for both traditional 2D and 3D games as well as augmented reality games and supports building games for a variety of different desktop (windows, macOS, Linux), mobile (iOS, Android, etc.), and console (PlayStation, Xbox, etc.) platforms. The game engine uses C# [28] as its programming language for custom-written scripts.

2.5.2 AR Foundation framework

Unity supports the development of augmented reality (AR) applications through its own AR Foundation framework [29]. The framework allows AR games to be built for iOS and Android platforms. This is done by acting as an interface between the developer and ARCore [30] and ARKit [31], the AR plugins for Android and iOS, respectively.

2.5.3 Animation

Unity provides the developer with animation features that allow game objects to perform pre-recorded animations during certain events.

2.6 Literature Review

This section outlines the literature review conducted during a previous project [12].

A state-of-the-art was conducted between September and November 2021 during a previous project to evaluate motivational elements found in existing applications targeting persons with ID. The overall goal in the prior project was to prepare Sorterius for future motivational features making the findings from the literature review highly relevant for this project.

2.6.1 Databases searched and criteria for exclusion

A total of five databases were used for sourcing literature. The databases selected focused on either computer science (ACM and IEEE Xplore) or health-related research (Embase and PubMed) or included both (Web of Science).

Literature not fulfilling all of the following criteria was excluded:

- Literature must be available in English
- Literature must incorporate a gaming aspect
- Literature must contain either a motivational element or physical activity interventions
- Literature must be accessible in full text

2.6.2 Approach for sourcing literature

A query comprised of a set of required keywords was created to source relevant literature, and the search was conducted on October 5th.

(Intellectual disability OR Cognitive Disability OR Development Disability OR Down Syndrome OR Autism)
AND
(Video Game OR Exergame OR Serious Game)
AND
(Social Skills OR Motivation OR Reward OR Exercise)
NOT
Cognitive Rehabilitation

Figure 6: Search query

Keywords used in the query (see Figure 6 [12]) were grouped into four subsets: 1) Literature must apply for intellectual disability, 2) Literature must incorporate gaming, 3) Literature must incorporate either cognitive or physical activity interventions or motivational elements, 4) Literature focusing on rehabilitating cognitive skills was considered irrelevant.

It should be noted that the first subset includes autism spectrum disorder (ASD) which does not necessarily impose an intellectual disability. However, many persons with ASD are also diagnosed with an intellectual disability [32]. This means that some literature targeting persons with ASD may also apply to persons with ID.

2.6.3 Literature selection

Sourced literature was reviewed systematically in accordance with PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) [33] guidelines (see Figure 7 [12]). The search query performed resulted in 352 records, of which 325 papers were unique (including one external record). The following review process was two-fold: 1) screen literature based on title, 2) assess relevance based on abstract and subsequently the full text. The first step consisted of evaluating whether the title indicated the literature to be in scope. The second step discarded literature for various reasons, including lack of gaming aspect and being unrelated towards persons with ID (see original paper for detailed description [12]). The systematic review resulted in nine papers included in the qualitative synthesis.

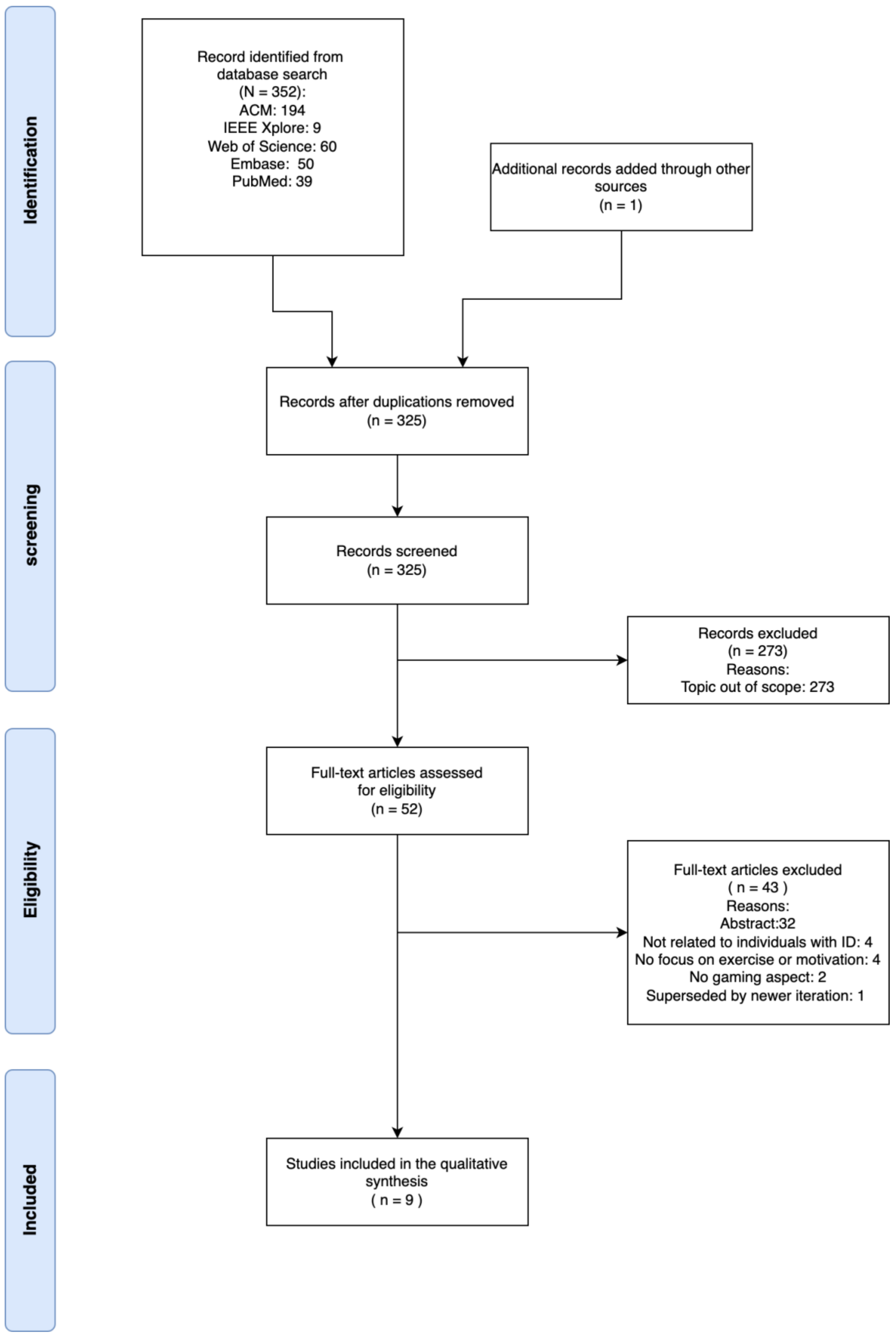

Identification
screening
Eligibility
Included
Record identified from database search (N = 352): ACM: 194 IEEE Xplore: 9 Web of Science: 60 Embase: 50 PubMed: 39
Additional records added through other sources (n = 1)
Records after duplications removed (n = 325)
Records screened (n = 325)
Records excluded (n = 273) Reasons: Topic out of scope: 273
Full-text articles assessed for eligibility (n = 52)
Full-text articles excluded (n = 43) Reasons: Abstract:32 Not related to individuals with ID: 4 No focus on exercise or motivation: 4 No gaming aspect: 2 Superseded by newer iteration: 1
Studies included in the qualitative synthesis (n = 9)

Figure 7: PRISMA diagram

2.6.4 Results

Results from the systematic review consisted of literature targeting four different aspects of serious games:

- Exergames: (n = 3)
- Cognitive skill-based games: (n = 3)
- Systematic reviews to assess the effects of exergames on persons with ID: (n = 2)
- A systematic review of motivational elements in existing computer-based interventions for persons with ASD: (n = 1)

2.6.5 Findings

Findings from included literature presented multiple elements perceived as motivating through observation and feedback from persons with ID. Integrating some of these features with Sorterius seemed infeasible as they would require drastic changes to the application's core functionality. This applies especially to suggestions [34, 35] regarding a storyline as a critical motivational element, which implies adding multiple game scenarios and levels.

Other suggestions seemed promising to be integrated with Sorterius, which include social aspect [35, 36], dynamic level of difficulty [34], and customization [34]. Some of the participants in one of the included papers [37] also expressed a desire for dynamic content.

3 Methods

3.1 Design

The proposed design features result from utilizing knowledge from the conducted literature review (see section 2.6) and continuous discussions with professionals in the field of psychology and ID. The design process consisted of two iterative phases: 1) brainstorming ideas to be added to Sorterius and 2) implementing features in Sorterius.

The Volere specification template [38] was used to state the requirements for this project. The Volere template provides a foundation to state functional and nonfunctional requirements consisting of a set of properties to define a requirement, the rationale of a requirement, and how a system should be measured to assess the implementation of a requirement.

3.2 Methodology for implementing motivational features

3.2.1 Initial phase

Initially, the project consisted of brainstorming features that could benefit the game by being perceived as motivating towards the user group. This was an iterative process that consisted of utilizing knowledge gained from a systematic review conducted during a previous project (see section 2.6) [12]. This systematic review indicated some areas that could potentially increase motivation in persons with ID, and the primary focus revolved around how such elements could be adapted to fit Sorterius. These ideas were then presented and discussed before being modified or discarded depending on the feedback. Due to time constraints, it was also important to assess whether an idea could be realized within the project's duration and which ideas were thought to be most promising. Another important aspect was that some ideas would require external expertise due to the inherent nature of obtaining necessary skills in an area (mainly revolving around creating custom 2D graphics and 3D objects for the game). The results from the initial phase found several features that could potentially improve the game. However, only some materialized.

3.2.2 Development phase

The next phase consisted of implementing some of the features from the initial phase. This process was conducted iteratively and incrementally and consisted of three steps: an implementation step and two testing steps (see Figure 8). The first step consisted of implementing a simplistic design of the desired feature. Other master students then tested the application within the same research group for feedback (alpha testing). The solution was then corrected based on the received feedback, and additional functionality was also added. This step was valuable for discovering undesired behaviors (edge-cases) of the solution and other minor improvements for better usability and could also be performed at short notice and in rapid intervals. After a more refined and complete solution was implemented, the application was tested by some of the supervisors (beta testing).

This step provided valuable feedback for uncovering more undesired behaviors and improving some elements of the design to make it easier to comprehend for the user.

Finally, the solution was re-designed in accordance with the received feedback.

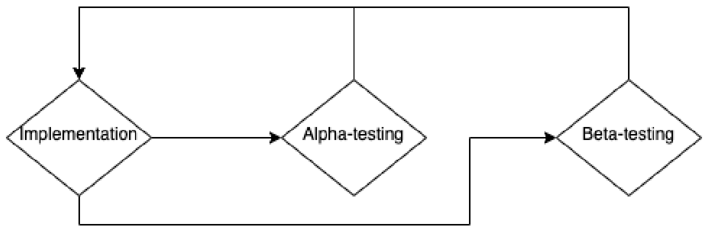


Figure 8: Iterative development

3.3 Evaluation Method

3.3.1 Discussion with Experts in the field of psychology and Intellectual Disability

Throughout the entire project, invaluable contributions have been made from experts in the field of psychology and intellectual disability, consisting of head physician professor Audny Anke and psychologist Henriette Michalsen. They oversee the project “Effects of Physical Activity with E-health Support in Individuals with Intellectual Disabilities”.

They were essential during the early stages of the project when assessing how Sorterius could be extended, and later when designing the new features. Their focus was heavily revolved around users' perspectives to the proposed features which were crucial for the resulting design.

3.3.2 System usability scale

Feedback from testing was done using a System Usability Scale.

The System Usability Scale (SUS) is a ten-items scale to map a global view of subjective aspects regarding usability [39]. SUS is a Likert scale in which a statement is provided, and the respondent can express their degree of agreement or disagreement with the statement through a point system. The point system ranges from 1 (strongly disagree) to five (strongly agree).

The ten items are arranged to alternate between positive and negative items to remove answer biases caused by the respondent not properly assessing each statement. Examples of positive vs. negative items could be:

1. I found the application easy to use.
2. I thought the application to be unnecessarily complex.

A perfect result for the SUS would therefore consist of half of the statements (positive items) to be assessed to "strongly agree" and the other statements (negative items) to "strongly disagree".

3.3.3 Evaluation Setup

Evaluation of the application is based on a usability test conducted on six persons with ID between May 2nd and May 13th, 2022. Due to privacy concerns, the location of the conducted experiments and any information identifying the persons testing the application will not be provided in this thesis.

The initial phase consisted of presenting the project's overall goal and giving a short introduction of the application to the assistants. The assistants were then given the required hardware (three iPhones) with the application installed. It was initially agreed to have three persons test the application for five days (Monday through Friday). However, while talking with the assistants, it was agreed to extend the duration to the following Friday instead (12 days in total). The assistants also proposed allowing more persons than the original three to test the application, which was greatly appreciated. Participants would utilize the application during walks as the application is best suited outside.

On the fifth day of the experiment, we were allowed to visit and observe three of the participants utilizing the application. This was a rewarding experience as it gave an indication of how the persons interacted with the app, which features seemed intuitive, and which were too complex. The results from the conducted testing are provided in form of barcharts.

3.4 Tools used

3.4.1 Software

Rider IDE

Creation and interaction with C# scripts in Unity were done solely using the integrated development environment (IDE) Rider (version 2021.3.2) [40]. Rider is developed by JetBrains specifically targeting .NET applications [41] and has built-in support for interacting with the Unity editor. The built-in support provides functionality that enables seamless transitioning between the Unity editor and IDE, making it the preferred choice for this project.

GIMP

One of the features to be materialized required users to be able to change the color of the game's mascot. GIMP (GNU Image Manipulation Program) is an editor for manipulating graphics and images and was used to manipulate the mascot's colors. This tool was also used to sketch the initial design for customizing the mascot. The reasoning for selecting GIMP was due to its easy learning curve and some experience from elementary school.

3.4.2 Hardware

Development platform

All development was performed using a Mac mini (M1, 2020) with OSX Monterey installed. As this machine utilizes Apple's silicon processor (RISC-architecture) and Sorterius was created with a version of Unity (v2019.3.11f) intended for x86 architecture, the machine had to rely on software (Rosetta 2) to translate all instructions. This turned out to be rather slow and unstable and resulted in multiple crashes. As a result, the choice was made to migrate Sorterius to Unity version 2021.2.15f1, which runs natively on silicon processors.

Mobile platform

The application was continuously tested during development. As the targeted platforms for this application are both Android and iOS, Sorterius was tested using an Android s21 and iPhone XR.

3.5 Critique of methods used

One critique of this project is that discussion with persons with ID or the assistants working daily with persons with ID was not conducted until the testing phase. Including them in the early phases of the project could result in an improved solution and other features to be added.

Another critique regarding the methods used in this project is the reliance on a somewhat old literature review conducted during an earlier project. The problem is not utilizing findings from this literature review (which proposed several features), but rather that a new search should have been conducted also to include literature released after the review was conducted. This could result in other findings which could affect the overall project. However, as research in this domain is small, significant findings during these couple of months are not likely.

4 Requirements

This chapter describes methods used for defining requirements and the resulting functional and non-functional requirements stated for this project.

4.1 Functional Requirements

Functional requirements state what properties a system should have to achieve its desired functionality. These requirements depict how the system should behave during certain events and state a system's output given certain inputs. In other words, functional requirements describe the cause and effect of a system. Functional requirements for this project are stated using the Volere requirement specification template [38]. The Volere template usually includes specifications for customer satisfaction and dissatisfaction for each requirement. However, due to the inherent nature of this project, such criteria would be infeasible to determine and are therefore replaced with a priority label ranging from one (highest) to five (lowest). The criteria for each requirement are stated as follows:

- **Requirement #:** Unique reference to the requirement
- **Description:** Describe the intended behavior of the system
- **Rationale:** Why should a system meet this requirement
- **Source:** State the origin of the requirement
- **Fit Criterion:** How the system should be measured to determine whether a requirement is successfully implemented
- **Priority:** The level of importance for the system to meet the requirement.
- **Dependencies:** Other requirements the system must fulfill to realize the intended requirement.

Table 2: Functional requirements

#	Description	Rational	Source	Fit Criterion	Priority	Dependencies
1	Visual hints should be given to the	Users might find it frustrating	Supervisors	The correct trashcan is animated to	1	

	user succeeding multiple failed attempts at sorting trash objects	and demotivating if they fail in sorting the discovered trash object		distinguish itself from other trashcans		
2	Visual hints should be given to the user when earning a new star.	Increase the chance for users to understand the existing star-point system.	Author	Newly awarded stars are animated, initially minuscule increasing to full size	3	
3	Add more trash objects to the game	Limited number of objects often results in same object being placed multiple times in a row during game session	Future work in Stellander's thesis [10]	More objects to be found during gameplay.	3	
4	Add more trash categories	Allows objects of different categories to be added to the game	Author	User finds trash object of a different category than food, plastic,	3	3

				glass, and cardboard		
5	Dynamically Increasing number of trash objects and categories that can be placed as user progress	Current solution has flat difficulty structures which might become monotonous and uninteresting as users become better at the game	Author, Supervisors	New objects previously undiscovered	3	3, 4
6	Enable users to customize their mascot	Currently no functionality for users to personalize their experience	Author	Users are provided with a menu for changing the color of the mascot and fit the mascot with various wearables	1	
7	Users can unlock new colors and wearables for their mascot	Users might be motivated to use the application to unlock new items	Author	Users discover new items for their mascot during game sessions	1	6

		for their mascot				
8	Users can choose to apply the new item for their mascot	User can modify their mascot during game sessions	Author	Users are provided with the option for applying their new item shortly after discovery	3	6, 7
9	Users should be able to apply previously discovered items at a later stage	Users might change their mind of the desired look for their mascot	Author, supervisors	Menu for customizing mascot is always updated with all unlocked items	2	6, 7
10	Discovered items are only available for one week	Users might find it motivating to play regularly to maintain the desired look for their mascot	Supervisors	Every new week, previously discovered items become unavailable for the user	4	6,7,9
11	System should be able to share	Enables the system to support	Author, supervisors	Device can retrieve data of other	5	

	data between devices	social aspects		users from backend		
12	Users should be able to add friends	Users might find it motivating to see friends playing	Author	System provides a menu for adding and viewing current friends	5	11
13	Users should be able to compete with friends	Users might find competition motivating	Author	System provides a leaderboard to its users	5	11,12
14	Achievements for number of specific objects collected	Users might find it motivating to receive achievements for collecting a certain number of specified objects	Author, Supervisors	Users earn a medal after collecting 10 objects of a specific item	5	

4.2 Non-functional Requirements

Contrary to functional requirements, which state the behavior of a system, non-functional requirements specify how the system should perform such behavior. Non-functional requirements typically describe the expected metrics for a system regarding usability, quality, security, and privacy. The non-functional requirements stated for this project are stated below.

4.2.1 Usability

The user interface for the current application is designed to accommodate the abilities of various levels of ID [10], and any further implementations on this application should preserve these core principles. Navigation should be easy to comprehend with clear indications of how the user can perform desired behavior from the application. These indications should be delivered visually, written, and verbally (text-to-speech). The latter is important due to limited literacy skills and increased risk of visual impairment for persons with ID.

4.2.2 Complexity

It is essential that the application's functionality is easy to understand and does not pose a steep learning curve for its users. As this project focuses on adding more features to the game, these features must be implemented in such a manner that minimal complexity is added to the application. Also, as the goal for this project revolves around increasing the overall enjoyment of the game, this project should assess whether existing functionality can be further improved to reduce complexity.

4.2.3 Security

Since the population of the targeted group is fairly small, it is vital that the application does not expose user identity. This requirement is especially important for any multiplayer functionality to be added where data would have to be stored at a remote location.

4.2.4 Performance

Due to the limited computational power provided by mobile platforms, any added functionality must account for these limitations to prevent low performance. This is an important aspect as low performance would result in frustration over the application and lead to reduced enjoyment of the game.

4.2.5 Maintainability

As development on the application might continue after the end of this project, new functionality should be implemented in a manner that can easily be maintained and extended by future developers.

5 Design

This section describes the intended design for motivational features to be integrated with Sorterius.

5.1 Customization of mascot

To allow users to personalize their experience, The choice was made to provide users with the ability to create their own mascot using predefined options. The original mascot was only available in blue color, and the proposed design was to extend this to provide the mascot in multiple colors. In addition, the proposed design included various garments to dress up their mascot.

5.1.1 Mascot Colors

The new design provides the mascot in eight different colors, which are shown in Figure 9. These colors were chosen to offer multiple options for the user and accommodate the several different preferences users might have.

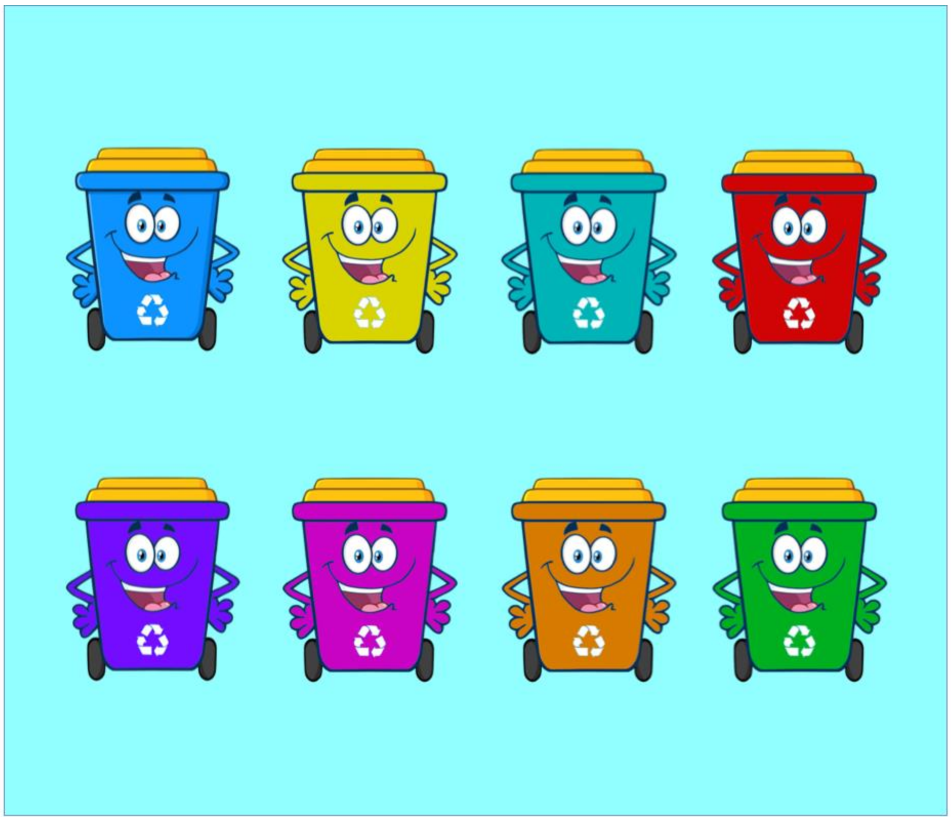


Figure 9: Mascot colors

5.1.2 Wearables

The new design provides users with eight wearables to outfit their mascot (see Figure 10). These were chosen as they were thought to make the mascot look “silly”/fun and more entertaining. As persons with ID have widely different interests and hobbies, the included wearables are of several different categories. The included wearables are as follows:

- Moustache and necktie [42]
- Straw hat [43]
- Pirate hat [44]
- Police hat [45]
- Baseball cap [46]
- Rainbow wig [47]
- Astronaut helmet [48]
- Magician hat [49]

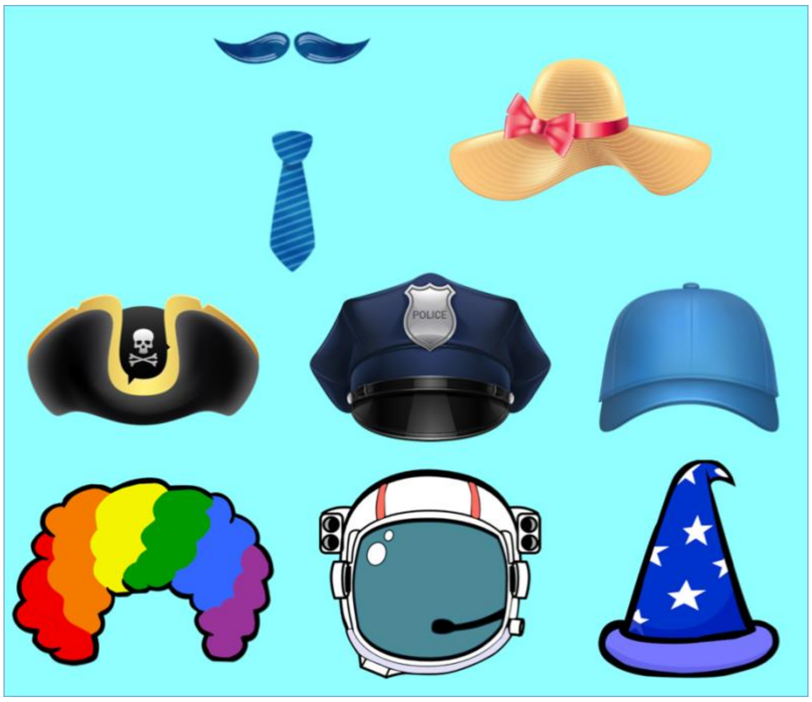


Figure 10: Mascot wearables

5.1.3 Customization Menu

The application provides the ability to change the mascot's appearance through a dedicated customization menu, fulfilling functional requirement #6. Subsequently entering this menu, users can choose their preferred color for the mascot and dress the mascot up in various wearables. To solve functional requirement #9 all items previously unlocked is accessible in this menu. The user interface for this menu is designed to be simplistic and easy to comprehend. When sketching the design for this menu (see Figure 11), a total of three options were provided to the user, namely, head garment, mascot color, and body garment. However, this was re-designed to two options during implementation (see Figure 12) by combining the body and head garment options into one (wearable) option. Each option has two corresponding arrows (next/previous) [50] for iterating through the various unlocked items.

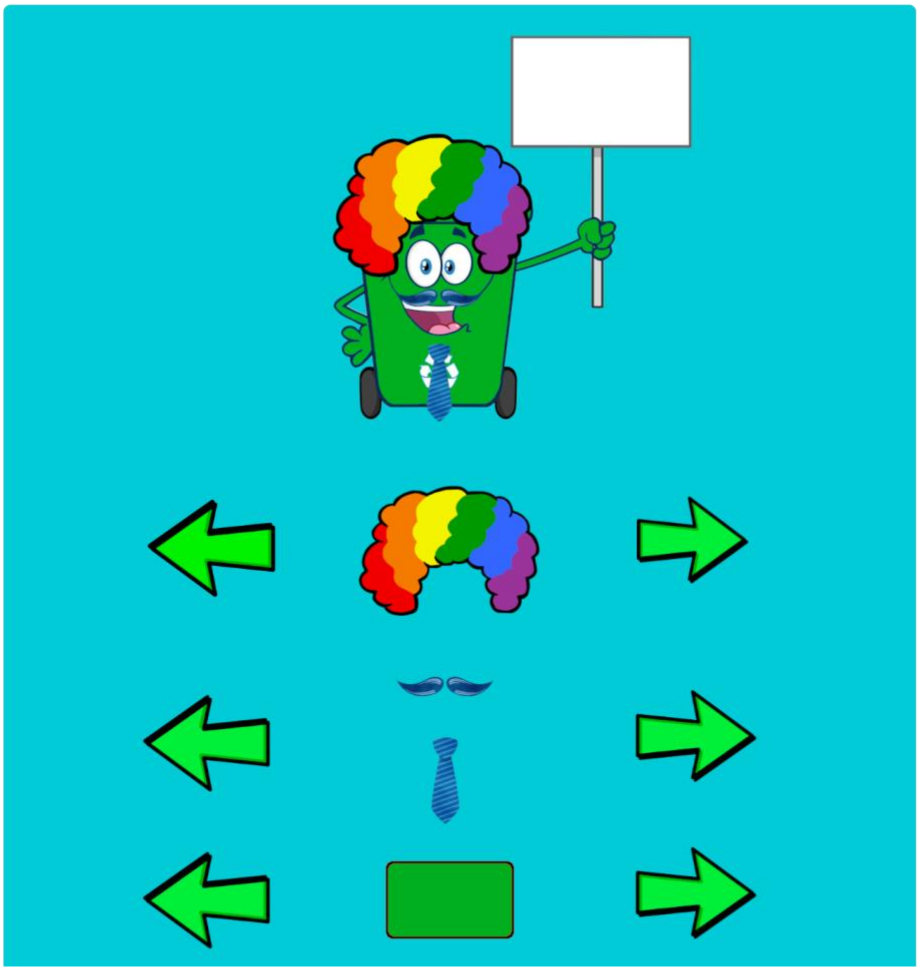


Figure 11: Initial design for customization menu

In addition to the provided options, a preview of the mascot is displayed to allow users to instantly view the outcome of the mascot as they iterate through the various unlocked items. As none of the existing mascot versions seemed suitable for this menu, a new version was created (using GIMP).

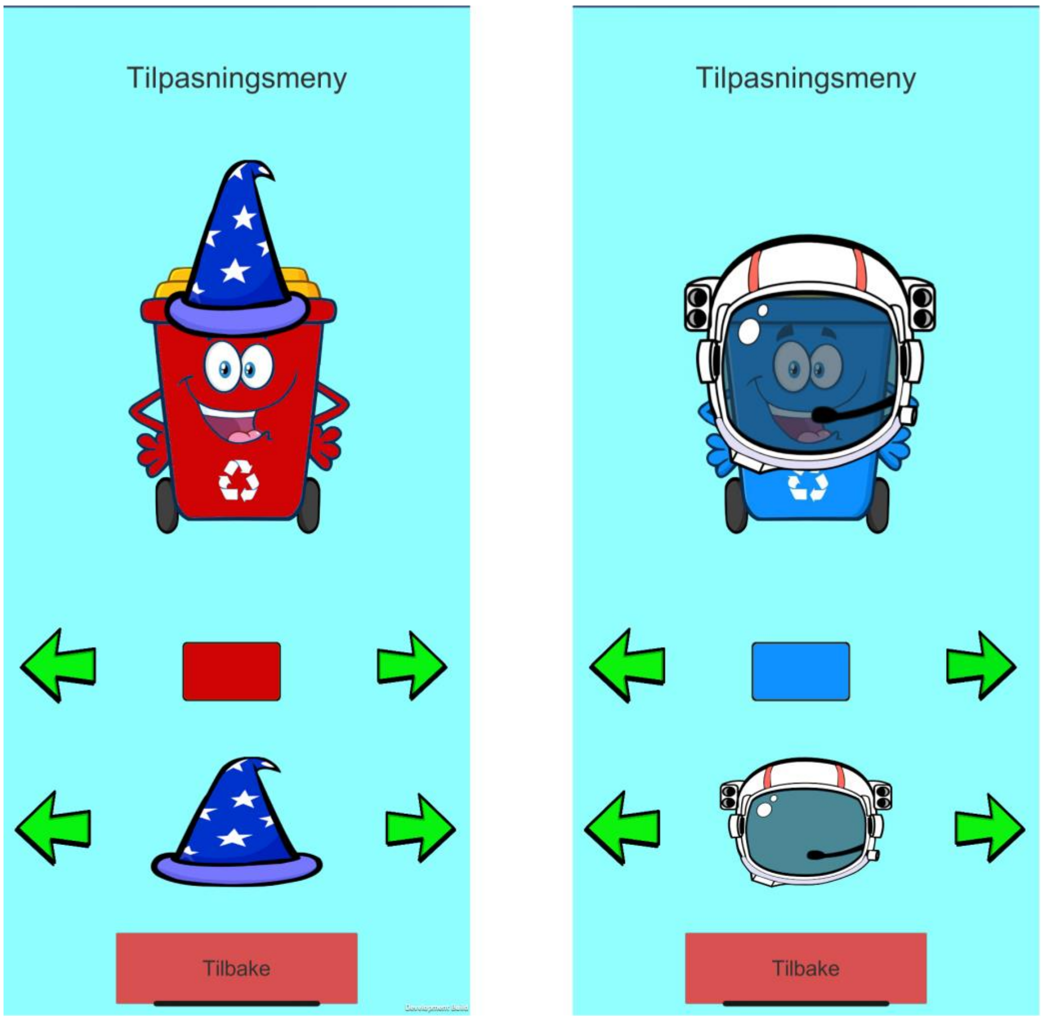


Figure 12: Customization menu

Accessing the Customization Menu

An important aspect of adding a customization menu to Sorterius is to make it easily accessible to the user. If a user does not understand how to navigate to the menu to alter the look of the mascot, the potential of the overall feature is drastically reduced. It was therefore decided to make this menu accessible from the main menu. This preserves the existing design of the application, where all major elements of the game are accessible from the main menu.

As most of the space in the main menu was used to display various game elements, little room was left available to place the button. The initial design proposed a button with text to navigate the new menu. However, this turned out to be difficult as the button would have to be of significant size with a large font to describe its functionality. Instead, an image of three wearables with a green background color was added (see Figure 13). The green background color was chosen as it is the same color as the button to enter a new session. This preserves the existing design where buttons in green are used to navigate forward.

The choice for selecting which wearables to be displayed on the button also had a purpose. Two of the three items included were unlocked by default and familiarized with the user during the initial setup of the mascot. Suppose a user remembers either of these items from the setup menu or notices that their selected wearable is identical to one of the displayed items on the button. In that case, the user might press that button which would navigate them to the customization menu. The last wearable included in the button was added to indicate that more items can be obtained. Lastly, the button was animated to have a pulsating color to further increase the button's likelihood of being noticed.

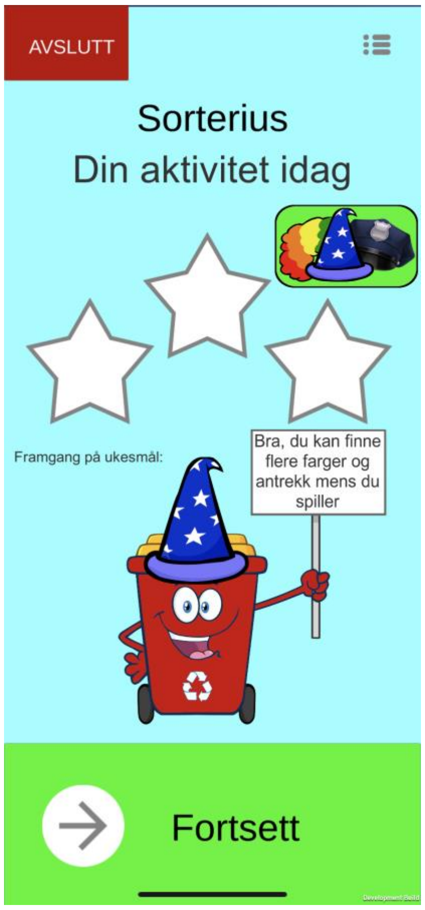


Figure 13: Button for navigating to the customization menu

5.1.4 Personalization from to go

To familiarize users with the customization features, the application provides users with some unlocked items by default. This design choice serves multiple purposes: 1) Users are introduced to the functionality of the system from the start, 2) Users can personalize their experience, and 3) Users might become motivated from the start to play the game to collect more items for the mascot. To make this feature noticeable to the user, a setup menu is displayed the first time a user starts the application (see Figure 14). This setup menu is almost identical to the customization menu used later in the game to reduce the risk of confusing the user. During this step, the mascot informs the user, both written and verbally (text-to-speech), of how to operate the customization menu. After a user has decided on their preferred mascot, they are informed of more items to be collected when playing the game.

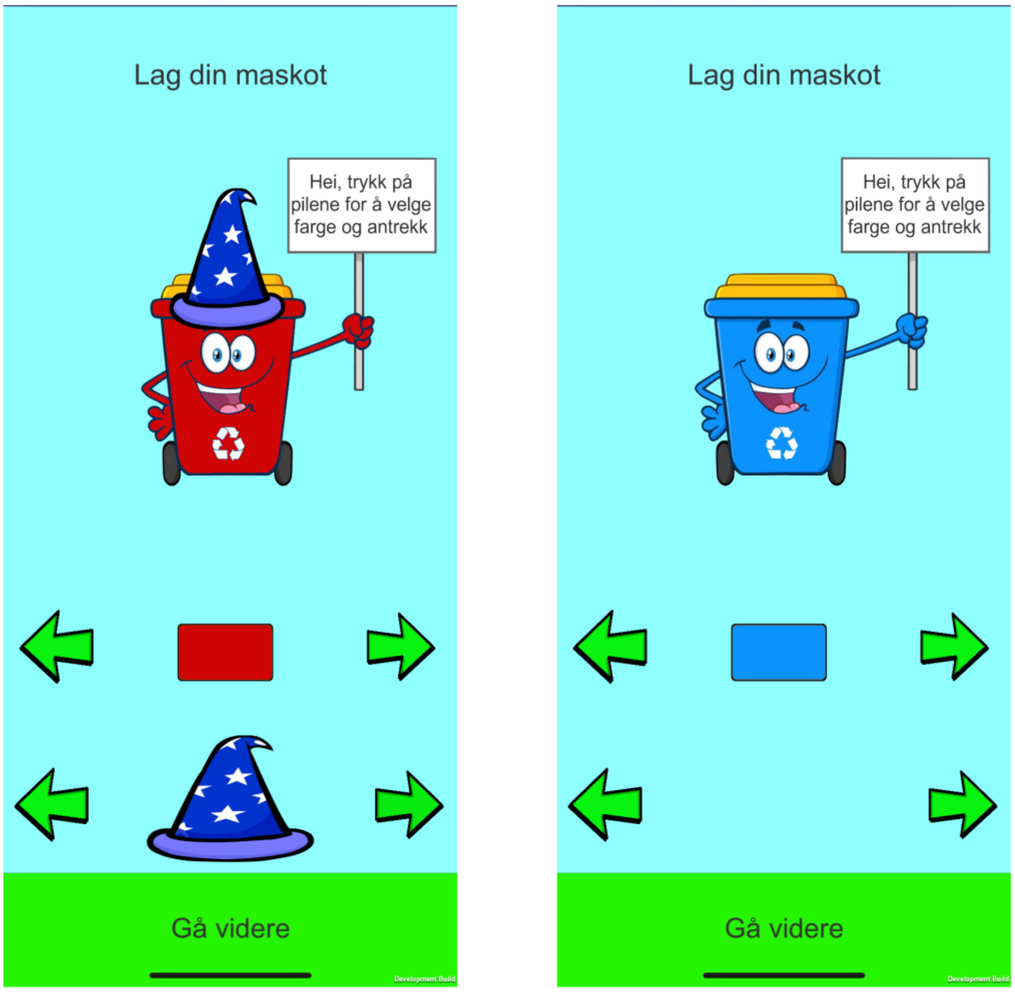

Lag din maskot
Hei, trykk på pilene for å velge farge og antrekk
Gå videre
Development Build
Lag din maskot
Hei, trykk på pilene for å velge farge og antrekk
Gå videre
Development Build

Figure 14: Mascot setup

5.1.5 Reward system

The application will unlock up to two items for the mascot per day to reward users for their effort. A new item is unlocked at specific thresholds, which are when a user reaches one-third of their daily goal and subsequently after completing their daily goal. Figure 15 depicts the procedure for unlocking new items. To make the unlocking of new items more interesting, items are enclosed by a 3D chest placed on the ground during game sessions, similar to trash objects. The chest is animated to jump, indicating that something inside is trying to break free. When the user taps on the chest, it is moved central to the screen (similarly to trash objects) and rotated to display the front of the chest. Subsequently, following a second tap, the padlock is unlocked, and the top lid bursts open. The new item (wearable or mascot color) is then revealed to the user, rising from the chest while rotating horizontally. Simultaneously, a music clip is played to further emphasize that the user has made an achievement. After the animation is finished, a menu pops up to allow the user to apply the new item instantly. This menu features two images of the mascot (The current mascot and a preview using the new item). These mascot images are also displayed similarly as when selecting the desired difficulty level before a game session to make it easier for the user to comprehend its intended functionality. The user can either choose one of the desired looks for the mascot and press the save button or press the abort button to leave the menu. Regardless of the button pressed, the user is brought back to the game session. Unlocking a new item is considered by the application to count as collecting a trash object, and if the required number of objects to be collected has been met, the session ends. If the user is rewarded with a new color, a mascot of the color in respect is revealed inside the chest. This design enables the system to meet functional requirements #7 and #8.

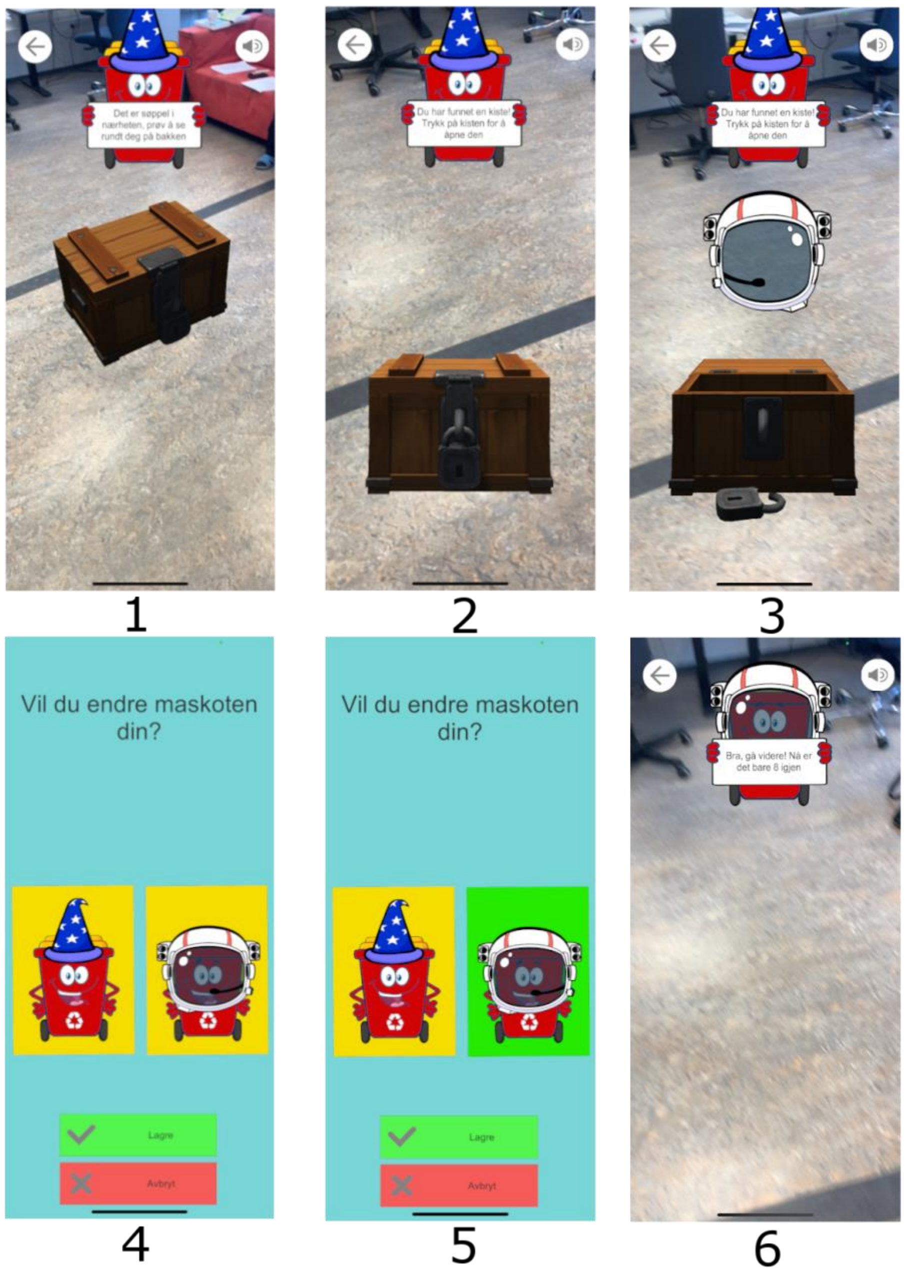

Det er søppel i nærheten, prøv å se rundt deg på bakken
1
Du har funnet en kiste! Trykk på kisten for å åpne den
2
Du har funnet en kiste! Trykk på kisten for å åpne den
3
Vil du endre maskoten din?
Lagre
Avbryt
4
Vil du endre maskoten din?
Lagre
Avbryt
5
Bra, gå videre! Nå er det bare 8 igjen
6

Figure 15: Item unlock for mascot

5.1.6 Unlock order

As there are two option categories to be unlocked for the mascot, the choice was made to switch every other time between which option category the user will receive from. In other words, a user is, for example first awarded with a wearable item and then a new color for their mascot. Naturally, this only applies while there are items in both categories to be unlocked. If more wearables items were to be added at a later stage (causing a skewed distribution), the system would only reward users with the remaining items of this category after all colors were unlocked. The order of items to be unlocked is also deterministic, meaning that users who have unlocked an equal number of items will possess the same items.

5.2 User Hint During Game Session

When discovering an item previously unseen or of an unknown category to the user, several failed attempts might cause frustration and a lack of motivation to continue playing the game. There have also been cases where the registered touch position is misaligned with the actual pressed position. The choice was therefore made to implement some visual hints to the user. If a user attempts to sort the trash object unsuccessfully two times for an object, the correct bin is animated with minor rapid rotations to indicate that the user should try to sort the object in that bin. The functionality allows the application to meet functional requirement #1. After the user has successfully sorted that item, the animation is switched off, providing no hints for future trash objects unless another two failed attempts occur.

5.3 Animation of star point system

To make the star point system easier to comprehend and make the game appear more fluid, the choice was made to add animations to the achieved daily stars in the main menu. Remembering the number of pre-earned stars might be difficult, meaning new stars might not be noticed. This is a missed opportunity for the application to successfully reward users for their achievements. Therefore, newly awarded stars are animated to visually display to the user which stars were earned from their last effort and which were awarded during previous game sessions that same day. This fulfills functional requirement #2.

5.4 Automated Difficulty Adjustment

This subsection describes the intended design for dynamically adjusting the difficulty based on users' performance. The design follows the concept Dynamic Difficulty Adjustment [51]. Due to time constraints, this design has not been materialized in the application.

The goal of the intended design is to introduce a dynamic difficulty that accommodates the varying abilities of users to preserve interest in the game for a prolonged duration. This feature is also intended to operate autonomously without requiring the user to manually interfere, which would impose added complexity on the operability of the game. Finally, this feature is designed to be integrated with the medium and hard game modes, not replace them.

5.4.1 Increase the Number of Trash Objects and Categories

The current solution consists of three difficulty levels of a flat structure. Regardless of the time spent utilizing the application, the application behaves identically regarding the trash objects to be spawned and their corresponding garbage bins for sorting. This might become monotonous and uninteresting over time, making the application less used.

The current solution is also somewhat limited in the number (12) of potential trash objects to be spawned. Choosing things at random often results in the same object being placed multiple times during a session. The proposed solution is to extend this functionality by increasing the number of trash objects and categories used in the game, which would fulfill functional requirements #3 and 4#.

5.4.2 Logging Sorted Items

The system logs all sorted objects and their success rate to adjust the difficulty based on user performance. This functionality should be implemented utilizing the existing persistent storage. The system should track how many times an object has been placed and the average number of attempts before successfully sorting the object. This would also allow a more even distribution of the various trash objects by placing the least used items.

5.4.3 Increasing Difficulty Based on User Performance

At first, the game will be unfamiliar to the user. Therefore, the application should consist of a limited number of easily identifiable objects to be placed during its early phase. Somewhat similar functionality exists in the current solution as the easy game mode utilizes items most familiar to the user, while the hard game mode also includes more complex items (e.g., wine bottle). As the user becomes more familiar with the application and learns how to sort the appearing objects correctly, the system can gradually introduce new items to the user. The application should only introduce one item at a time and at a slow pace to not overwhelm the user. The success rate for sorting the new objects can then be evaluated to determine whether they impose too much difficulty on the user. If the user still finds it difficult to sort a new item after appearing a certain number of times, the item should stop appearing. The application should then evaluate whether a different object should be placed or if the current number of objects is sufficient. Implementing this behavior would solve functional requirement #5.

If the user learns to sort the new item correctly and still successfully sorts the other known objects, the application can proceed to unlock an additional new item. Only items of similar categories compared to the default objects should be introduced during the early phase.

Introducing New Categories

The main difference between the medium and hard game modes is the number of options provided to the user when sorting trash objects. Medium provides the user with two options (food and plastic), while hard provides four options (food, plastic, paper, and glass). When a user sorts all items of the available trash categories with a high success rate, the application can gradually introduce items from a new category. The proposed design intends to maintain the number of options provided for each object but instead swap between the possible options. For the medium game mode, this means that a user is only provided with two options, of which one of the options is correct. For example, if a user finds an eaten apple, the possible sorting options may be food or plastic. However, if a user finds a tin can, the provided options may be metal and plastic. This allows additional categories to be added without increasing the complexity of adding more options to the user. This feature should be treated equally as introducing new items of familiar categories in which categories imposing too much difficulty should be removed to prevent frustration.

6 Implementation

6.1 Customizable Mascot

This subsection describes the process for implementing customization features for the game mascot.

6.1.1 Scaling and positioning in GIMP

As the original mascot versions had different alignments and were of various sizes which would require active repositioning and scaling depending on the image to be displayed, all mascot images were pre-fitted identically using GIMP. This also applied to the included wearables, which were positioned at the desired location relative to the mascot. This removed the need for repositioning the images when switching between, e.g., the mustache-necktie and police-hat (see Figure 1. Nb. Different positioning for the selected wearables) at run-time as they were prepositioned.

The customization menu was implemented to display two images of the selected wearable, one previewing the look of the mascot and a second for navigating between the accessible wearables (see Figure 16). The latter required all respective images to be aligned equally for the same reasons as with the mascot images. Therefore, a second copy for each wearable was created with trimmed borders (centralizing the image). In other words, each wearable had two copies of different positions.

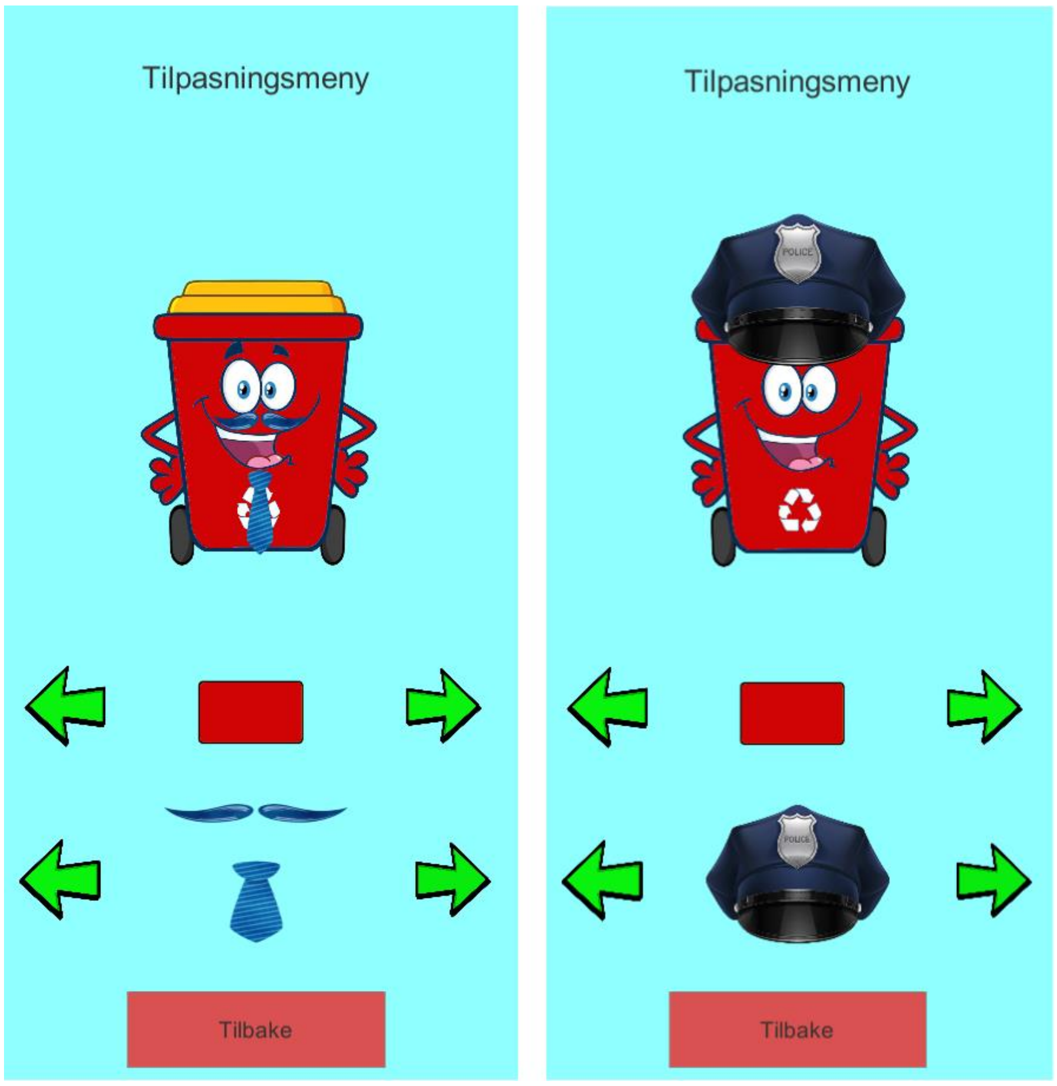


Figure 16: Wearable positioning

6.1.2 Mascot Coloring

Providing the mascot in multiple colors was done using GIMP's bucket fill which allows altering the color of an area to a different color. This tool was used to prevent the need to alter the color for each pixel manually. By specifying a threshold, all pixels within a specific range on the color spectrum relative to the targeted pixel were filled. This only applies to pixels enclosing the targeted position. A minimum threshold setting would result in only identical colors bordering the targeted position to be altered. In contrast, a maximum threshold would regard colors of the entire spectrum to be changed to the new color. It was, therefore, important to set the threshold properly to achieve a satisfying result. Finding this threshold was done through trial and error until the appearance was deemed adequate. Anti-aliasing was also enabled to prevent rough edges between different colored pixels.

6.1.3 Sign cut out

During the early stages of development, it was discovered that some of the wearables overlapped with the sign of one of the mascot versions (see left mascot in Figure 17). To combat this issue, the sign was manually cut out using GIMP and stored as a separate image. This allowed the sign to be layered in front of the mascot and wearable, displaying full visibility of the text (see Figure 17). This process was performed on all color versions.

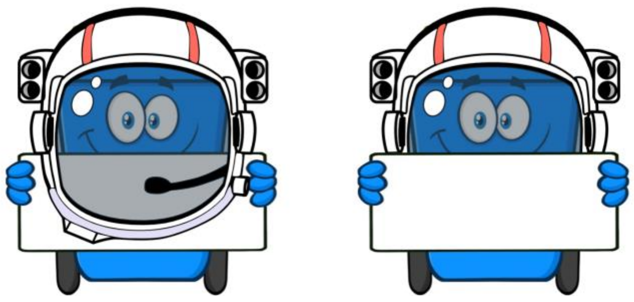


Figure 17: Fixing sign overlap

6.1.4 Mascot as a Game Object

The previous mascot was implemented as a single individual static image for each game view. Since changes applied to the mascot must be consistent throughout the game, the choice was made to implement the new mascot as a persistent game object. Since Unity by default destroys all objects associated with a scene during scene changes, and the mascot is intended to persist through the whole game (Sorterius consists of four scenes), the mascot object is implemented using Unity’s “DontDestroyOnLoad” function. This preserves the object when its affiliated scene is destroyed. The mascot is also implemented in accordance with the singleton pattern, which limits a class to only having one instantiated object at a given time. This eliminates the risk of instantiating a second mascot object if its affiliated scene is loaded a second time.

As the color of the mascot is consistent through gameplay (unless modified by the user), and the application continuously needs to swap between which versions to display, all versions of the chosen color are read to memory. The Mascot object was also implemented with a series of helper functions for selecting the appropriate image and position for each view.

6.1.5 3D Chest

The chest used for unlocking new items was retrieved from Unity’s asset store (CITE). The chest was already integrated with a series of animation clips for making the chest bounce and open the top lid, but these were all executed when instantiated. Therefore, the animation flow would have to be altered to fit its intended purpose. To modify the animation flow for the chest, Unity’s built-in animation controller was used. This allowed the bounce animation clip to be looped until the chest was tapped before transitioning to open the chest. This also enabled additional animation clips to be added, such as hovering and rotating the enclosed item after the top lid was opened.

6.1.6 Resource folder

All folders and sub-folders named “Resources” in a unity project are accessible at run-time through a provided scripting API [52]. Mascot images and wearables were stored in such a folder due to the non-deterministic behavior of which mascot color the user would choose. Figure 18 illustrates the resource folder structure where the images were stored. Since a given mascot version would look similar in every color and be utilized in the same scenarios, the choice was made to label corresponding images with similar names. Images were then grouped in subfolders based on their color. Each subfolder for the mascot would therefore contain five mascot images in addition to a mascot sign (see section 6.1.3) with names equal to the affiliated images residing in the other subfolders. These subfolders were then labeled with the color of the images stored within (e.g., blue mascot versions were stored in a subfolder named “Blue”). This allowed the mascot’s helper functions to be agnostic of the current color in use after being retrieved from storage.

As all wearables consist of two images, the same logic was implemented by assigning similar names for corresponding images and distinguishing versions with unique folder labels. These images were grouped based on whether they were fitted to be worn by the mascot or if they would be used for navigating the available wearable items.

Lastly, a subfolder (labeled colors) for storing the color icons used in the customization menu was added to the resource folder. These icons were labeled with the name of the color they represent, which allowed the system to dynamically load all color options during initialization (see section 6.2.1).

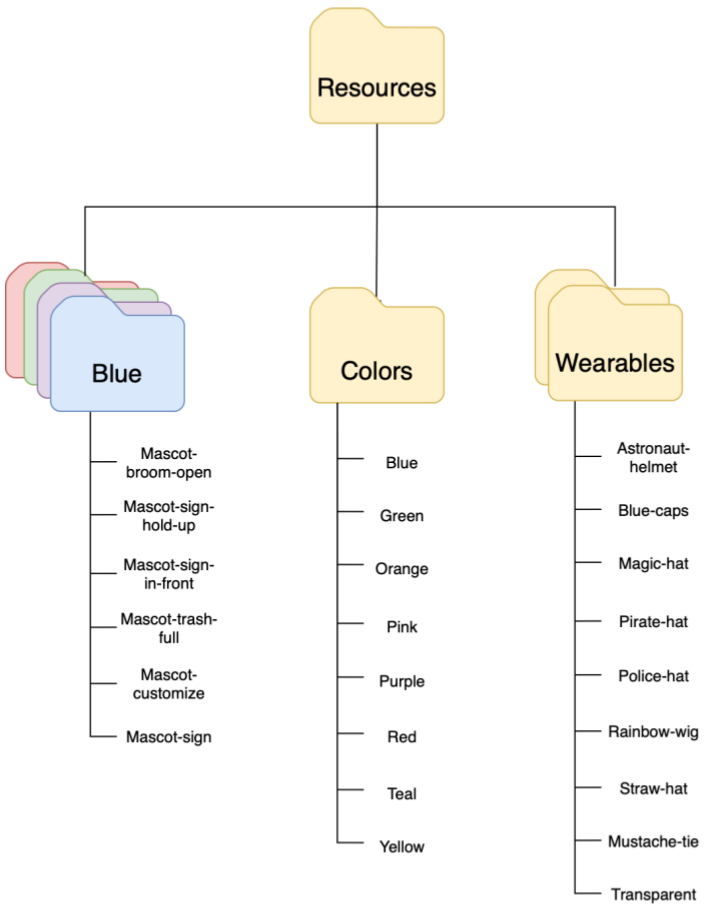


Figure 18: Folder structure

6.2 Gameplay

6.2.1 Initializing Mascot options

To preserve unlocked items and preferences regarding the mascot look, the application stores relevant information using the existing storage system for persisting data [12]. If no such data resides in storage (first time playing), the application retrieves all sprite images located in the wearable and color folder. Subsequently, it stores the name for each sprite in their designated lists (one for wearables and one for colors). The items in these lists consist of a name for the respective sprite and a Boolean value to determine whether that sprite should be regarded as locked or unlocked. Items unlocked by default are also set accordingly during this step, including three items in each category (number of wearables perceived as two options for the user since the application regards a transparent image as a wearable). After all mascot options have been initialized, the preferred color of the mascot is assigned to blue and the wearable to transparent. The appropriate images are then read to memory, and subsequently, the setup menu is displayed to the user (see 6.2.2).

6.2.2 Customization and Setup menu

As the customization and setup menu utilizes the same script and is identical (apart from the displayed mascot version and the button interface), a common description for both menus is provided.

When a user enters the customization menu, the application retrieves all unlocked wearables, mascot images of a specified version, and corresponding color icons. Two different copies (see section 6.1.1) for each wearable are retrieved and inserted in separate lists of identical order. Mascot images and corresponding color icons are also ordered equally in separate lists. After all sprites have been inserted into their corresponding lists, these lists are iterated until the current mascot preference is matched.

The menu consists of a series of image objects in addition to the mascot object. As the user navigates the various options using the provided arrows (next/previous), the appropriate lists are iterated through and display the next/previous items accordingly. The images for the mascot objects are manipulated directly which removes the need for the user to actively save its preference. If a user chooses a different color than the previously selected color, all mascot versions of that color are retrieved from storage when the user exits the customization menu.

6.2.3 Unlocking a New Item

As the user reaches certain thresholds in regard to number of steps in relation to their targeted daily goal (one-third and completion of the daily goal), a user is rewarded with a new item for their mascot. Figure 19 shows the process for selecting the next item to be placed.

For each trash object to be sorted, the application compares the number of steps achieved with the targeted goal. If one of the two thresholds for achieving a new item has been met and there exist more currently unavailable items, the application retrieves the next item to be unlocked.

To find the next item, the application checks for locked items in both categories (wearables and colors) and combines the total number of locked items. If the total number of locked items is even, the application iterates through the list of all colors until a color previously unavailable to the user is located. This item is then retrieved from storage and subsequently assigned to an image object enclosed by a 3D chest. It is important to note that this item is still considered locked by the application during this step. If the total number of locked items is odd, the application instead retrieves a wearable. If all items of a certain category have been unlocked, the application retrieves an item from the category still possessing locked items.

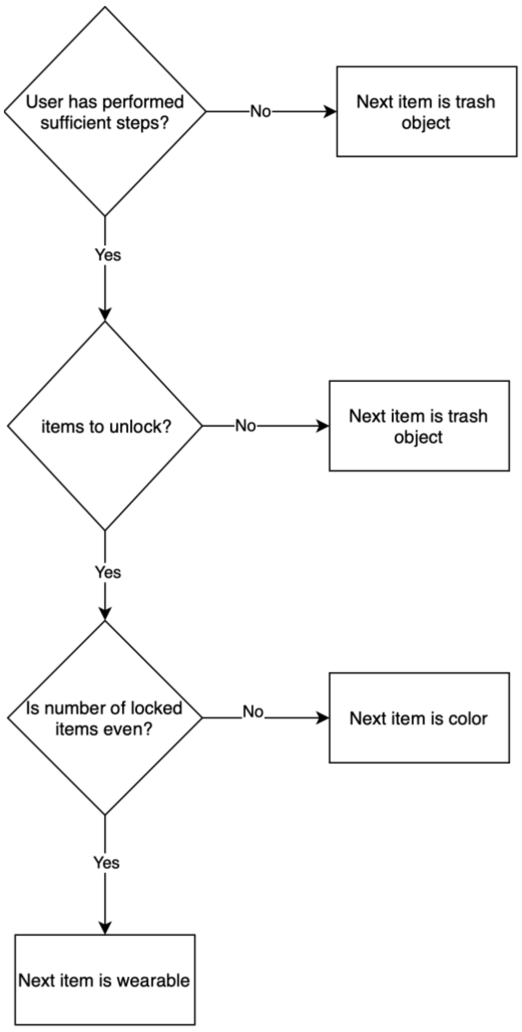

User has performed sufficient steps?
No
Next item is trash object
Yes
items to unlock?
No
Next item is trash object
Yes
Is number of locked items even?
No
Next item is color
Yes
Next item is wearable

Figure 19: Selecting next item to be placed

When a user has reached a certain number of steps, the 3D chest is instantiated (similarly to trash objects). This chest object contains an animator controller and a series of various animations for the chest. The first animation is activated and looped by default which makes the chest bounce vigorously. The image residing within the chest is initially deactivated to prevent parts of the new item to pierce the chest, being visible to the user. When the user touches the chest, the chest is positioned centrally on the screen with the front of the chest facing the user. When the user touches the object a second time, the image is activated, and the next animation is played (the new item is now regarded as unlocked by the application). The new animation first unlocks the padlock and bursts open the top lid while playing a melody, and then hovers the new item above the chest while rotating horizontally. This animation is only executed once and is implemented to activate the menu for applying the new item at the end of its cycle.

6.2.4 Selecting new item

The menu for selecting the new item displays two adjacent mascot images. These images are not retrieved from storage as they already exist in memory. The image on the left displays the look of the current mascot, while the image on the right displays the mascot with the new item. If the new item is a wearable, the right image displays the mascot in its current preferred color with a new item. If the item is a color, the mascot is displayed with the new color wearing its current preferred wearable. Initially, both images consist of a yellow background color (similar to choosing difficulty levels [10]). When the user taps either of the images, the background color is altered to a green color to indicate the choice a user has made. This feature is implemented to only display one of the buttons in a green background color at any given time as both choices cannot be applied. When a preferred look for the mascot has been selected, the user can save its preferences by pressing the save button. If the new item is selected, the application applies this item to the mascot object. If the current mascot has been selected or if no selection has been made, no update on the mascot object is performed. Finally, the user is brought back to the game session with/without the new item. The user is also provided with an abort button which also brings the user back to the current session.

6.3 Animations

This subsection describes the process for adding animations to various objects.

Animating objects for the application was done using Unity's built-in animator module. This module provides an interface (see Figure 20) consisting of a record button (red button in the top-left corner) and a timeline. After the desired object was selected, the record button was pressed which logged all initial properties of the object. The position of the timeline was then moved to the desired position and subsequently, the desired properties were altered. The newly created animation would then gradually alter the selected properties from their initial value to the altered position. These properties were also portrayed as graphs that could be manipulated to allow fluctuating behavior.[52]

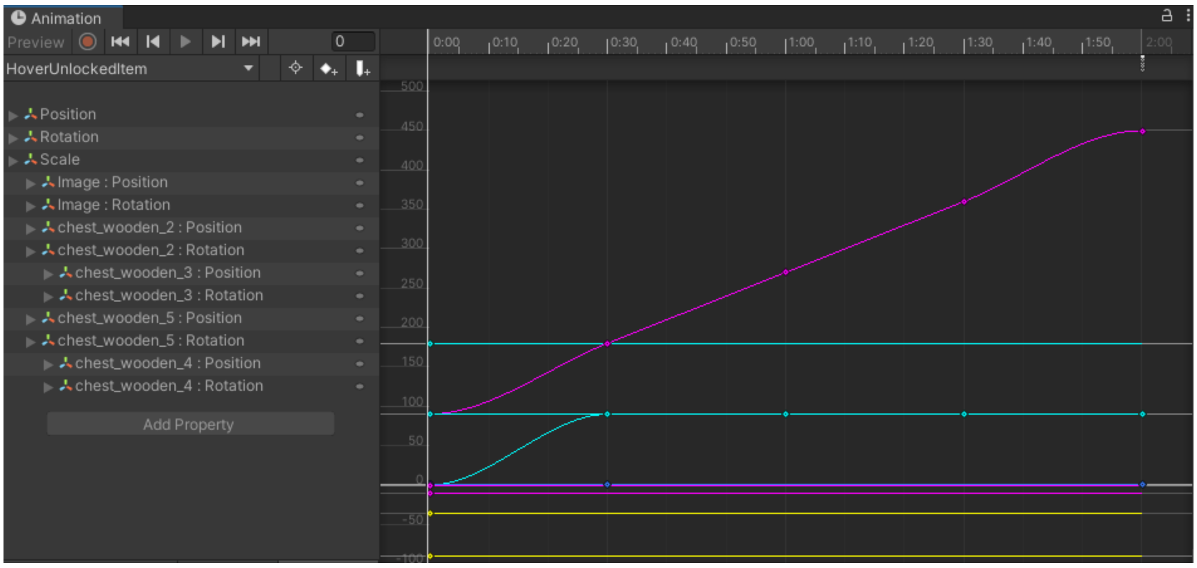

Animation
Preview
0
HoverUnlockedItem
Position
Rotation
Scale
Image : Position
Image : Rotation
chest_wooden_2 : Position
chest_wooden_2 : Rotation
chest_wooden_3 : Position
chest_wooden_3 : Rotation
chest_wooden_5 : Position
chest_wooden_5 : Rotation
chest_wooden_4 : Position
chest_wooden_4 : Rotation
Add Property
0:00
0:10
0:20
0:30
0:40
0:50
1:00
1:10
1:20
1:30
1:40
1:50
2:00
500
450
400
350
300
250
200
150
100
50
0
-50
-100

Figure 20: Animation interface

To allow various animations for an object to be executed under some conditions, the animator module consists of a controller. Figure 21 shows the animation flow for the chest object. The controller consists of a series of states for playing specific animation clips. The controller switches between the various states when certain conditions are met.

The controller allows animation clips to be associated and arranged in an arbitrary order. Figure 21 shows the animation flow for the chest object. When the chest is instantiated, the controller transitions to the Bounce state (orange box) looping the animation for making the chest bounce. The controller will maintain this state until the user taps on the chest, which then transitions to the "open" state. After the animation clip for opening the chest is completed the next state is activated hovering the new item above the chest before finally displaying the menu for selecting the new item to its users.

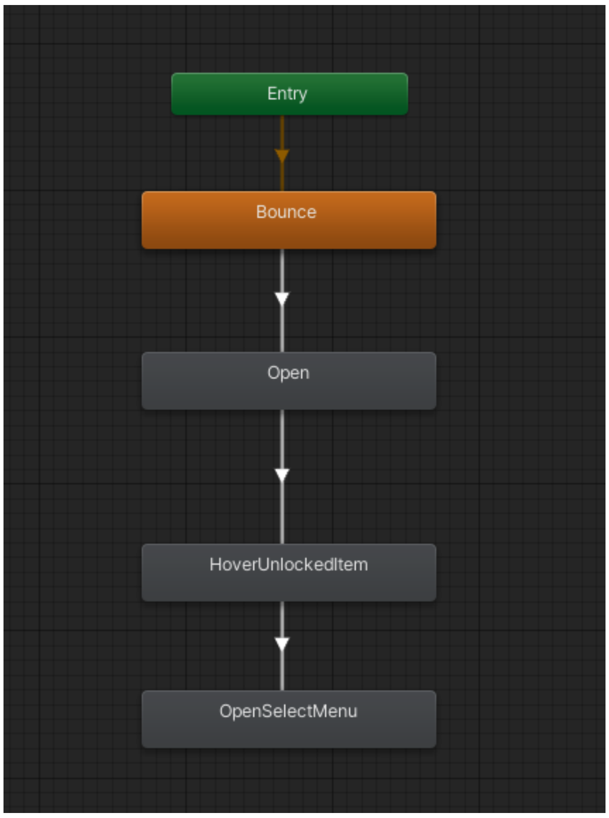


Figure 21: Animator controller for chest

6.4 Replacing the Text-to-speech voice

As the new features required more information to be provided to the user, more text-to-speech would have to be generated to complement written text. The details of the previously used text-to-speech generator were not provided. To prevent mismatching voices to be used in the game, all text-to-speech was regenerated with a new voice. Text-to-speech was generated using Google's Text-to-Speech API [53].

7 Testing and Results

This chapter presents the results from testing the application experts in the field and persons with ID. The chapter will first present results from testing the application on experts in the field, and then the results from persons with ID

7.1 Participants

The application was tested on two experts in the fields of psychology and ID, and six persons with ID. However, due to a lack of written consent by five of the persons with ID, only one person is included in the results. The persons with ID had daily access to the application throughout the test period and the resulting SUS was filled out by an assistant. The SUS is identical to the one used in the previous iteration to compare the prior and current solution.

7.2 SUS score and interpretation

Each item in the SUS can be rated from 1 (strongly disagree) to 5 (strongly agree). To calculate the total score, positive items are subtracted 1 point from the item's rating, while negative items are subtracted 5 points. Each item can therefore achieve a score ranging from 0-4. Note that looking at the score of each item isolated does not provide any meaningfulness, but the accumulated score for all items does provide a measurement of usability.

Figure 22 shows the accumulated score for each question where participants' individual rating is distinguished by color. As there are three participants, the accumulated result for individual items ranges between 0 and 12.

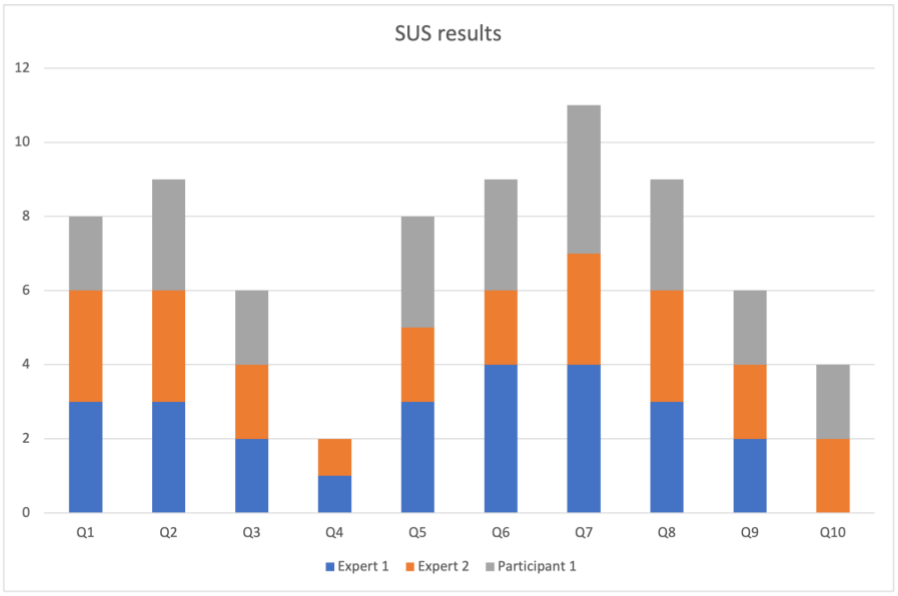


Figure 22: SUS results

After the score for all items has been combined, the score is multiplied by 2,5 resulting in a total score ranging from 0 to 100. Figure 23 shows the various methods for interpreting the total score, as summarized by Sauro et al. [54].

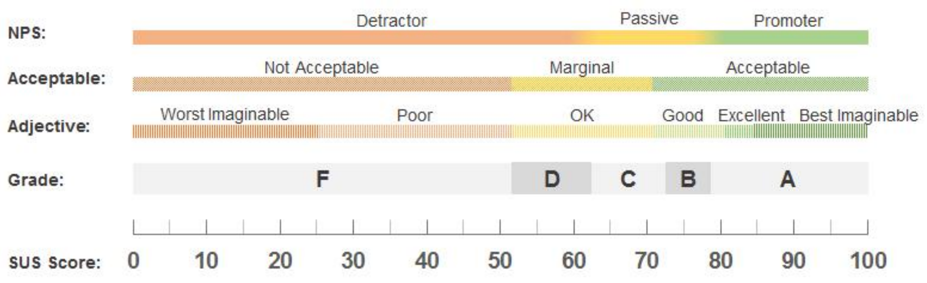


Figure 23: SUS score interpretation

7.3 Results

Figure 24 shows the total score for each participant ranging between 57,5 and 62,5. The average score for the application is $\frac{57,5+60+62,5}{3} = 60$.

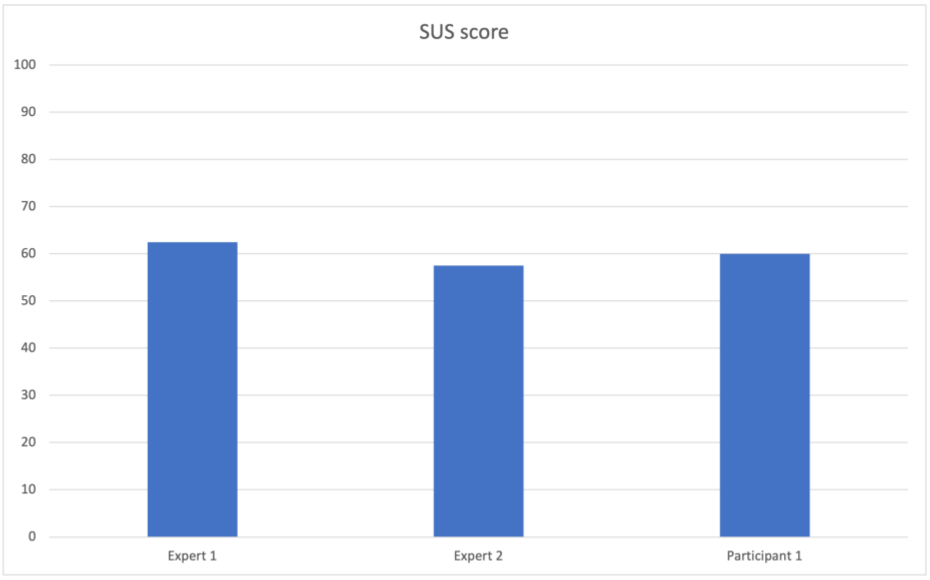


Figure 24: SUS score

Comparing the score with Figure 23, the application receives:

- An acceptability score of marginal
- Rated as “OK”
- Grade D

From interpreting the score, testing indicates that the application falls marginally between acceptable and unacceptable. This illustrates the application to be somewhat usable, although there is still room for improvement. This score is also identical to the one received in the previous project [10], which is discussed in section 8.2.

7.3.1 Observation

After five days of testing, we were allowed to visit and observe the application in use which took place outside during a 20–30-minute walk. One element of the game which proved to be essential was the text-to-speech functionality. After entering a game session, the mascot instructed the user to “move around to find trash objects”. This led the user to lower the phone and start walking without paying any attention to the screen. The user kept walking until they heard the mascot inform them that trash was nearby and then put their phone in front of them. As the user did not pay any attention to the screen while walking, they would not notice when they were supposed to stop and look for trash objects.

Another observation was that guidance was continuously provided by their assistant while navigating the menu. This indicated that some assistance may be required during the first couple of times utilizing the application. However, it should be noted that guidance was provided quickly without leaving much time for the user to try it for themselves first.

7.3.2 Feedback

Experts

After testing the application on experts, some feedback was provided. One comment regarded the menu for selecting newly awarded items (see section 5.1.5). The concern was that some users might find it difficult to comprehend the intended behavior of first selecting the desired mascot and then pressing the “apply” button to save the choice made. A proposed solution to simplify this menu is stated in the future work section 8.6.

Another comment was that the first time a new item was unlocked, the item was a wearable item which was exciting and motivating. However, when unlocking a second item, the new item was a new color for the mascot which was less exciting and rewarding compared to the wearable.

Assistants

During testing of the application, the assistants expressed that several of the participants enjoyed spending time with the application, but made some suggestions regarding the current solution.

Firstly, it was stated that the overall enjoyment of the game could be increased if users could collect bags of candy and soda bottles, as these items might be perceived as more enjoyable compared to cardboard and plastic bottles. At the same time, the suggested items could be associated with unhealthy habits.

Another comment was that users should be able to collect five items during one session, since ten items could be less motivating for some individuals. Completing a session might be perceived as motivating for some individuals, and therefore completing a shorter session might be preferrable over longer sessions.

One assistant also suggested making a similar solution as the existing application, but instead of collecting trash objects, users would receive a quiz question succeeding sufficient steps. The assistant expressed that in their experience, this group tends to enjoy quizzes, and this could therefore increase utilization of the application.

It was also suggested that items should be spawned during shorter intervals. The current solution places an object when the user has performed 20-30 steps. However, it was proposed that new objects should appear succeeding 10-15 steps.

Finally, it was proposed that trash objects and bins should be labeled to make it easier to identify the appearing objects.

8 Discussion

8.1 Strength and Limitations

To properly assess the effects of the proposed solution, testing would need to be conducted over an extended period. However, due to time constraints, testing was limited to two weeks and did only assess the usability of the application.

Even though the application was tested on six persons with ID, we were not able to get signed consent from more than one of the participants before submitting this thesis. As the participants could not give their own consent, this would have to be provided by a guardian, which turned out to be difficult. There are various reasons which led to the limited number of signed consents, most including miscommunication. As a consequence, this thesis cannot present results from the participants not providing written consent.

8.2 Evaluating Usability of Application

Utilizing an identical SUS, as in the previous iteration of Sorterius, allows a somewhat direct comparison between the two results. Some differences should be noted, however. Firstly, the previous project relied solely on participants working closely with persons with ID, not the actual intended users. Secondly, testing the new application on the person with ID was conducted for a duration of two weeks, while the previous solution was tested for one day. Lastly, the number of participants for this project is significantly lower than the prior project.

Still, this project aimed to increase the number of motivational elements for Sorterius while preserving its primary functionality and behavior. Based on the results from the previous project, it was assumed that the overall application was usable by persons with ID, which resulted in the application being extended instead of modified. Consequently, the user interface was preserved, and new features were integrated in accordance with the existing design. This means that this project never aimed at increasing usability of the application. However, the project (as stated in sub-problem #3) sought to implement additional features without imposing added complexity.

Testing usability of the new application is therefore essential to assess whether added features have increased the overall complexity of the game, which would reduce usability. As the new application achieved a similar score to the previous version, on

As the new application was rated equally usable as the previous version, one could argue that the added motivational elements have been successfully implemented without introducing complexity, solving sub-problem #3: How can we add motivational features without affecting the overall complexity of Sorterius?

8.3 Social Aspect

Initially, this project aimed to add social aspects to Sorterius by utilizing the persistent storage implemented during an earlier project. However, as mSpider (backend for storing data) required major modifications to operate in a reliable and secure matter, adding social aspects to Sorterius would be a comprehensive task. Also, during discussions with experts it was agreed that even though social aspects could be perceived as motivating for persons with ID, Sorterius lacked features to increase motivation of the actual gameplay. Social aspects should be considered as a complementary feature to the game, not as a main motivator. If users do not find the gameplay interesting, they will not utilize the application, hence, no use of social aspects. Therefore, due to time constraints and the inherent nature of social aspects, these features were not pursued further.

8.4 Mascot customization

During users' first time playing, it is important that the application sparks an interest to ensure further use. Users should therefore be introduced to some of the motivational elements of the game during their first time utilizing the application. In Sorterius, users are introduced to the customization features of the mascot during the initial setup. This aims to familiarize users with some of the game's aspects as well as allow users to personalize their game experience.

Personalization as a motivational aspect was one of the main findings from the literature review and enabling Sorterius to reward users with new items for their effort should further increase motivation for playing the game. Providing users with no information about the locked items aims to introduce some mystery and surprises to the game, encouraging users to reach their daily goal to reveal their new items.

Personalization was also implemented in the exergame AGA [26, 27] where users can customize a 3D avatar. However, there are some differences between the personalization features in Sorterius and AGA. The 3D avatar used in AGA is realistic and is intended to support users to create an avatar resembling themselves, while Sorterius utilizes a cartoony 2D mascot which can be dressed up in various garments to look "silly" and fun. Both applications provide users with personalization features initially when utilizing the game, but for Sorterius, this is merely an introduction to the customization features as users can unlock additional items during gameplay. The intent of the personalization features is therefore different for the two applications, where AGA aims to motivate by providing a realistic avatar resembling the user, and Sorterius aims to motivate users to reach their daily goals by rewarding them with new items for their mascot. It should be noted however that AGA do reward users for reaching their daily goals by unlocking various animal companions.

As users are introduced to personalization features initially and can be rewarded with new items during their first game session, the new mascot design should provide a solution to sub-problem 1: How can we ensure initial interest when trying Sorterius for the first time?

As there is a somewhat limited number of items for the mascot, daily use of the application will result in all items being unlocked after 1-2 weeks. Therefore, the current mascot implementation cannot be considered to preserve users' interest over an extended period, which is the second sub-problem for this project. However, adding more items (and options categories) to the game, or adding new functionality might enable the mascot feature to solve sub-problem 2.

8.4.1 Two Copies Per Wearable

As described in section 6.1.1, the choice was made to create two copies for each wearable which had different positioning using GIMP. One copy was fitted to be worn by the mascot, and the other would be used for navigating the customization menu. This removed the need to actively position each wearable at runtime. Positioning at runtime would require a significant amount of added code and would likely need to be solved by implementing a dedicated helper function for each wearable just to be worn properly by the mascot. However, requiring two copies of the same wearable is not considered optimal either, but this approach was deemed favorable of the two.

8.4.2 Holding Images in Memory

As described in section 6.1.4, all versions of the preferred mascot color along with the selected wearable are stored in memory. This choice was made as the mascot is continuously updated as users utilize the application. This drastically reduces the required number of reads from storage compared to only retrieving the exact image to be used for each new view.

Retrieving data from storage is considered slow compared to memory so the implemented solution should be more performant.

8.4.3 Alternative Solution

As described, this project implemented the mascot as a persistent game object. This required a series of helper functions to be implemented to display and position the correct mascot version throughout the game.

A different approach could be to utilize the previous implementation of the mascot consisting of a pre-positioned image object for each view. All images of the preferred mascot and wearable could be stored in memory located in the data manager object (or a designated object), and for each new view, the corresponding image object could retrieve the correct mascot and wearable image. This approach was considered early in the project but was abandoned for the following two reasons: (1) The implementation of the mascot would be scattered across multiple game objects and scripts (might require one script per view). The opted solution is implemented using one script for all mascot functions. (2) All preferred mascot images would have to be held in an external object (in memory) to be accessible for all views. The opted solution holds all images internally in the mascot game object.

8.4.4 Retrieve All Wearable Images from Storage During Initialization

During initialization, the system retrieves the appropriate images from storage. The application does not utilize the actual images but registers the name of these images to map all mascot items. This allows new items to be added at a later stage, which then can be dynamically added to the application. Retrieving images without using them is not optimal as they are quite large. However, as there was no provided functionality in Unity's Resource API (cite) to only retrieve filenames, the actual file was retrieved instead.

8.4.5 Deterministic unlocking order

As stated in section 5.1.6, the order of the items to be unlocked is implemented in a deterministic and predefined manner. An improved solution might be to modify this behavior to instead unlock items in random order. This would allow users to unlock different items which might increase motivation if two persons were to compare their findings. It could be the case that if a user was made aware of an item not in their possession, they would be motivated to play the game to unlock that exact item.

8.4.6 Replacing Wearables for Mascot

As the application is intended to be

Towards the end of this project, it was discovered that the wearable images used were not permitted for software development. These images were licensed for personal non-commercial use and it was therefore thought (as this is a non-commercial application) they were applicable for this project. However, since this was not the case, they have been replaced by other images with a commercial license. The images currently residing in the application are the following:

- Christmas hat [55]
- Juggler hat [56]
- Mexican hat [57]
- Sailor hat [58]
- Top hat [59]
- Wizard hat [60]
- Cook hat [61]
- Arrow [62]

8.5 Automated Difficulty Adjustment

One of the findings from the literature review was to gradually increase the difficulty level. The rationale is that users become familiar with the challenges and behavior of a game as they utilize the application. Therefore, the perceived difficulty might decrease over time, and the rewards from completing these challenges will not be perceived as rewarding. This, in turn, might lead to decreased use of the application over time. Also, one of the main obstacles is that users progress at different rates. Some challenges might be too difficult for some while others may find them easy to solve. If challenges become too difficult, users will not progress and might instead exit the game out of frustration.

The proposed design aims to provide users with increased difficulty as they become better at the game, but, at their own pace. Continually monitoring users' performance allows the application to dynamically alter the difficulty. This design also allows autonomous adjustment without any input from the user, which could increase the complexity of the game. As this was not implemented, the outcome of this design is unknown. There are also some aspects of the design which has not been addressed. For instance, what success rate for sorting trash should act as a threshold for introducing new items? How many failed attempts for sorting an item should result in the item to stop appearing? And should that item reappear at any later stage? These issues have not been solved, but the overall design might be a step closer to solving sub-problem 2: How can we maintain users' interest in Sorterius over an extended period?

The intended design follows the concept of Dynamic Difficulty Adjustment (DDA) where the difficulty is customized to the users [51]. Other games have also adapted such behavior. For instance, in Crash Bandicoot 2: Cortex Strikes Back [63], players continually dying on a certain level are either provided with a tool to help them stay alive, or the level itself is made easier to complete. Crash Bandicoot 2 consists of multiple levels that become more difficult as players progress. In order to finish the game, players must successfully complete all levels, and the DDA mechanics are intended to aid players stuck on a certain level. Still, as the game becomes harder for each level, the balancing effect of the DDA is somewhat limited, and dying too many times would result in the player losing all progression and having to start over from the first level. Comparing this game to the automated difficulty adjustment intended for Sorterius, the main difference is that the DDA effect in Sorterius would be more significant as it is not affected by any level progression. The intention for Sorterius is not to make the game difficult, but rather to maintain users' interest, which is the rationale for removing items that are perceived as difficult.

Another game that utilizes DDA is Resident Evil 4 [64] which uses a scale from 0-10 to grade users based on their performance. A low grading would result in passive enemies dealing reduced damage, and a high grading would result in aggressive enemies dealing increased damage. Instead of using levels, Resident Evil 4 consists of five chapters.

Sorterius share some similarities with Resident Evil 4 in terms of providing users with the option to select the desired difficulty and then utilizing DDA to complement the chosen difficulty. However, similarly to Crash Bandicoot 2, Resident Evil 4 also becomes more challenging as users make progression in the game, which is not necessarily the case for Sorterius.

When comparing Sorterius with the aforementioned games, it seems that the DDA used in the other games are intended to complement a static increase of difficulty implemented in the levels/chapters. For Sorterius there is no inherent increase of difficulty, as there are no progression levels. While all three applications aim to increase the difficulty based on user performance, Sorterius is less focused on providing a challenging experience for the users.

8.6 Future Work

8.6.1 Option to Collect Five Trash Objects During Session

The current solution provides users with the ability to choose 10, 15, or 20 objects to be collected during a game session. Future work should add the option to collect five objects as it could be perceived as motivating for users to complete multiple short sessions compared to one large session.

8.6.2 Customization of Mascot

The current solution provides users with two option categories (color and wearable) to personalize their mascot. There is also a total of 11 (excluding items unlocked by default) items that the user can unlock which is somewhat limited. Future work should focus on adding more items and possibly introduce a third option category. This would enable the motivational aspect of the mascot to be prolonged.

Reset unlocked mascot items each week

One solution to prevent all items to be unlocked could be to reset all items each new week. Users would then have to play on a weekly basis to preserve their look for the mascot, which might be perceived as motivating. When implementing such functionality, one should also assess whether items should be unlocked in identical orders every week, or if non-deterministic behavior is preferrable.

Menu for Selecting New Items

One of the comments made by experts when testing the application was a concern regarding the difficulty of utilizing the menu for applying newly awarded items. The proposed solution was to only display one image of the mascot with the new item. The user would then be asked whether they would like to apply the new item, in which the user is provided two buttons labeled “yes” and “no”.

Fail to Instantiate New Objects

There is a known malfunction in the current solution where the application might stop instantiating new objects even though the user has performed sufficient steps. The fault occurs sporadically, and several game sessions may be played with the application behaving as intended. The bug has not been fixed but it is thought that the culprit is the condition for assessing whether new trash objects should be instantiated.

9 Conclusion

This project has extended the augmented reality game Sorterius with new elements intended to motivate its users. The proposed elements are based on findings from a previous literature review and continuous discussions with experts in psychology and intellectual disabilities. The new Sorterius provides its users with a customizable mascot and a reward system where they unlock new items for their mascot while playing the game. To preserve users' interest over an extended duration, a proposed design for automatic difficulty adjustment based on user performance has been provided.

The new application was evaluated by conducting a usability test on experts and one person with ID. Testing was performed on six persons with ID, but due to various reasons, we were not able to retrieve a signed consent from more than one before submitting this thesis. Results from the conducted test indicate similar usability as the previous version, validating the intended outcome of not adding complexity to the game.

Contribution

Contribution of this project revolves around utilizing gamification aspects to provide an entertaining platform for persons with ID. While this application targets increased PA levels in persons with ID, features proposed in this thesis are general purposed and can be applied in a variety of games and other applications. The proposed feature follows common gamification aspect such as social aspect, personalization, and increased difficulty and illustrates one way to adopt such aspects to an application.

Another contribution from this project is the design choices of how users should interact with the new features, such as utilizing an image for navigating to the customization menu.

Testing of the application also revealed the text-to-speech functionality to be of major importance for persons with ID when utilizing the application.

Bibliography

1. Warburton DER, Nicol CW, Bredin SSD. Health benefits of physical activity: the evidence. Canadian Medical Association Journal. 2006;174(6):801-9.

2. WHO. Physical activity: World Health Organization; 2020 [Available from: https://www.who.int/news-room/fact-sheets/detail/physical-activity. Accessed: 2021-25. November]

3. Tomaszewski B, Savage MN, Hume K. Examining physical activity and quality of life in adults with autism spectrum disorder and intellectual disability. J Intellect Disabil. 2021:17446295211033467.

4. Wouters M, Evenhuis HM, Hilgenkamp TIM. Physical activity levels of children and adolescents with moderate-to-severe intellectual disability. J Appl Res Intellect Disabil. 2019;32(1):131-42.

5. Dairo YM, Collett J, Dawes H, Oskrochi GR. Physical activity levels in adults with intellectual disabilities: A systematic review. Preventive Medicine Reports. 2016;4:209-19.

6. Wouters PJM, Spek E, Oostendorp H. Current Practices in Serious Game Research: A Review from a Learning Outcomes Perspective. Games-Based Learning Advancements for Multi-Sensory Human Computer Interfaces: Techniques and Effective Practices. 2009.

7. Baranowski T, Buday R, Thompson DI, Baranowski J. Playing for Real: Video Games and Stories for Health-Related Behavior Change. American Journal of Preventive Medicine. 2008;34(1):74-82.e10.

8. Connolly TM, Boyle EA, MacArthur E, Hainey T, Boyle JM. A systematic literature review of empirical evidence on computer games and serious games. Computers & Education. 2012;59(2):661-86.

9. Lee M, Lee H, Kim Y, Kim J, Cho M, Jang J, et al. Mobile App-Based Health Promotion Programs: A Systematic Review of the Literature. Int J Environ Res Public Health. 2018;15(12).

10. Stellander M. Sorterius: Game-inspired App for Encouraging Outdoor Physical Activity for People with Intellectual Disabilities: UiT - Norges arktiske universitet; 2021.

11. Wold I. Creating an App Motivating People with Intellectual Disabilities to Do Physical Activity 2019.

12. Luzi T. Towards Persistent Storage for Sorterius [Unpublished Work]. 2021.

13. Schalock RL, Luckasson R, Tassé MJ. An Overview of Intellectual Disability: Definition, Diagnosis, Classification, and Systems of Supports (12th ed.). American Journal on Intellectual and Developmental Disabilities. 2021;126(6):439-42.

14. Schalock RL, Luckasson R, Tassé MJ. The contemporary view of intellectual and developmental disabilities: Implications for psychologists. Psicothema. 2019;31(3):223-8.

15. Patel DR, Cabral MD, Ho A, Merrick J. A clinical primer on intellectual disability. Translational Pediatrics. 2020:S23-S35.

16. Committee to Evaluate the Supplemental Security Income Disability Program for Children with Mental D, Board on the Health of Select P, Board on Children Y, Families, Institute of M, Division of B, et al. In: Boat TF, Wu JT, editors. Mental Disorders and Disabilities Among Low-Income Children. Washington (DC): National Academies Press (US) Copyright 2015 by the National Academy of Sciences. All rights reserved.; 2015.

17. World Health Organization. Division of Mental H. ICD-10 guide for mental retardation. Geneva: World Health Organization; 1996.

18. Laamarti F, Eid M, El Saddik A. An Overview of Serious Games. International Journal of Computer Games Technology. 2014;2014:358152.

19. Oh Y, Yang S. Defining exergames & exergaming2010.

20. Benzing V, Schmidt M. Exergaming for Children and Adolescents: Strengths, Weaknesses, Opportunities and Threats. Journal of clinical medicine. 2018;7(11):422.

21. Unity Technologies. Unity n.d. [Available from: https://unity.com/. Accessed: 2022-05-29]

22. Henriksen A, Johannessen E, Hartvigsen G, Grimsgaard S, Hopstock L. Consumer-Based Activity Trackers as a Tool for Physical Activity Monitoring in Epidemiological Studies During the COVID-19 Pandemic: Development and Usability Study. JMIR Public Health and Surveillance. 2021;7.

23. Mostphotos. [Available from: https://www.mostphotos.com/en-us/user/hittoon. Accessed: 2020-05-22]

24. MongoDB Inc. Build faster. Build smarter. n.d. [Available from: https://www.mongodb.com/. Accessed: 2022-05-29]

25. Haugland V. Game-inspired App for Encouraging Outdoor Physical Activity for People with Intellectual Disabilities 2019.

26. Wiik M. AGA: A Game-Inspired Mobile Application for Promoting Physical Activity in People With Intellectual Disabilities 2019.

27. Eilertsen T. Activity Game Avatar: A interactive exergame for people with intellectual disabilities 2021.

28. Microsoft Corporation. C# documentation n.d. [Available from: https://docs.microsoft.com/en-us/dotnet/csharp/. Accessed: 2022-05-29]

29. Unity. AR Foundation n.d. [Available from: https://unity.com/unity/features/arfoundation. Accessed: 2022-05-29]

30. Google LLC. ARCore n.d [Available from: https://developers.google.com/ar. Accessed: 2022-05-29]

31. Apple inc. ARKit n.d [Available from: https://developer.apple.com/augmented-reality/. Accessed: 2022-05-29]

32. Baio J, Wiggins L, Christensen DL, Maenner MJ, Daniels J, Warren Z, et al. Prevalence of Autism Spectrum Disorder Among Children Aged 8 Years - Autism and Developmental Disabilities Monitoring Network, 11 Sites, United States, 2014. MMWR Surveill Summ. 2018;67(6):1-23.

33. Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, et al. The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. BMJ. 2021;372:n71.

34. Derks S, Willemen AM, Wouda M, Meekel M, Sterkenburg PS. The co-creation design process of 'You & I': a serious game to support mentalizing and stress-regulating abilities in adults with mild to borderline intellectual disabilities. Behaviour & Information Technology. 2021.

35. Whyte EM, Smyth JM, Scherf KS. Designing Serious Game Interventions for Individuals with Autism. J Autism Dev Disord. 2015;45(12):3820-31.

36. Torrado JC, Wold I, Jaccheri L, Pelagatti S, Chessa S, Gomez J, et al. Developing software for motivating individuals with intellectual disabilities to do outdoor physical activity. Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering in Society; Seoul, South Korea: Association for Computing Machinery; 2020. p. 81–4.

37. Finkelstein S, Barnes T, Wartell Z, Suma EA, Ieee, editors. Evaluation of the Exertion and Motivation Factors of a Virtual Reality Exercise Game for Children with Autism. 1st Workshop on Virtual and Augmented Assistive Technology (VAAT); 2013 Mar 17; Orlando, FL2013.

38. Robertson J, Robertson S. Volere requirements specification template. 2012.

39. Brooke J. SUS: A quick and dirty usability scale. Usability Eval Ind. 1995;189.

40. JetBrains. Rider n.d. [Available from: https://www.jetbrains.com/rider/. Accessed: 2022-05-29]

41. Microsoft Corporation. .NET n.d. [Available from: https://dotnet.microsoft.com/en-us/. Accessed: 2022-05-29]

42. Mamina. [Available from: https://www.cleanpng.com/png-meter-non-commercial-activity-commerce-font-fashio-7549660/. Accessed: 2022-05-22]

43. Jefte. [Available from: https://www.cleanpng.com/png-woman-with-a-hat-sun-hat-straw-hat-sombrero-fashio-187935/. Accessed: 2022-05-22]

44. Calaji. [Available from: https://www.cleanpng.com/png-hat-piracy-stock-photography-clip-art-pirate-hat-c-648215/. Accessed: 2022-05-22]

45. Missivy. [Available from: https://www.cleanpng.com/png-police-hat-png-clipart-58789/. Accessed: 2022-05-22]

46. Swadipto. [Available from: https://www.cleanpng.com/png-baseball-cap-blue-png-clip-art-image-13988/. Accessed: 2022-05-22]

47. Auntmarie. [Available from: https://www.cleanpng.com/png-wig-clown-clip-art-wig-cliparts-coloring-201252/. Accessed: 2022-05-22]

48. Sceg. [Available from: https://www.cleanpng.com/png-astronaut-space-suit-helmet-outer-space-clip-art-s-161409/. Accessed: 2022-05-22]

49. Enrice. [Available from: https://www.cleanpng.com/png-hat-magician-cap-clip-art-multi-style-uniforms-4687691/. Accessed: 2022-05-22]

50. Ameer1. [Available from: https://www.cleanpng.com/png-arrow-clip-art-arrows-cliparts-128253/. Accessed: 2022-05-22]

51. Zohaib M. Dynamic Difficulty Adjustment (DDA) in Computer Games: A Review. Advances in Human-Computer Interaction. 2018;2018:5681652.

52. Unity. Resources [Available from: https://docs.unity3d.com/ScriptReference/Resources.html. Accessed:

53. Google LLC. Text-to-Speech n.d [Available from: https://cloud.google.com/text-to-speech. Accessed: 2022-05-29]

54. Sauro J. 5 Ways to Interpret a SUS Score 2018 [Available from: https://measuringu.com/interpret-sus-score/. Accessed: 2022-May 6th]

55. Yulia Y. NATURALISTIC 3D VERSION OF SANTA CLAUS HAT [Available from: https://www.mostphotos.com/en-us/29208294/naturalistic-3d-version-of-santa-claus-hat-on-a. Accessed: 2022-05-22]

56. YLIVDESIGN. JESTER HAT ICON, CARTOON STYLE [Available from: https://www.mostphotos.com/en-us/39995342/jester-hat-icon-cartoon-style. Accessed: 2022-05-22]

57. RING B. MEXICAN HAT CARTOON STICKER [Available from: https://www.mostphotos.com/en-us/51886392/mexican-hat-cartoon-sticker. Accessed: 2022-05-22]

58. YLIVDESIGN. SAILOR HAT ICON, CARTOON STYLE [Available from: https://www.mostphotos.com/en-us/40077133/sailor-hat-icon-cartoon-style. Accessed: 2022-05-22]

59. YLIVDESIGN. GENTLEMAN HAT ICON, CARTOON STYLE [Available from: https://www.mostphotos.com/en-us/40046062/gentleman-hat-icon-cartoon-style. Accessed: 2022-05-22]

60. YLIVDESIGN. WIZARD HAT ICON, CARTOON STYLE [Available from: https://www.mostphotos.com/en-us/39982241/wizard-hat-icon-cartoon-style. Accessed: 2022-05-22]

61. YLIVDESIGN. COOK HAT ICON [Available from: https://www.mostphotos.com/en-us/40072632/cook-hat-icon-cartoon-style. Accessed: 2022-05-22]

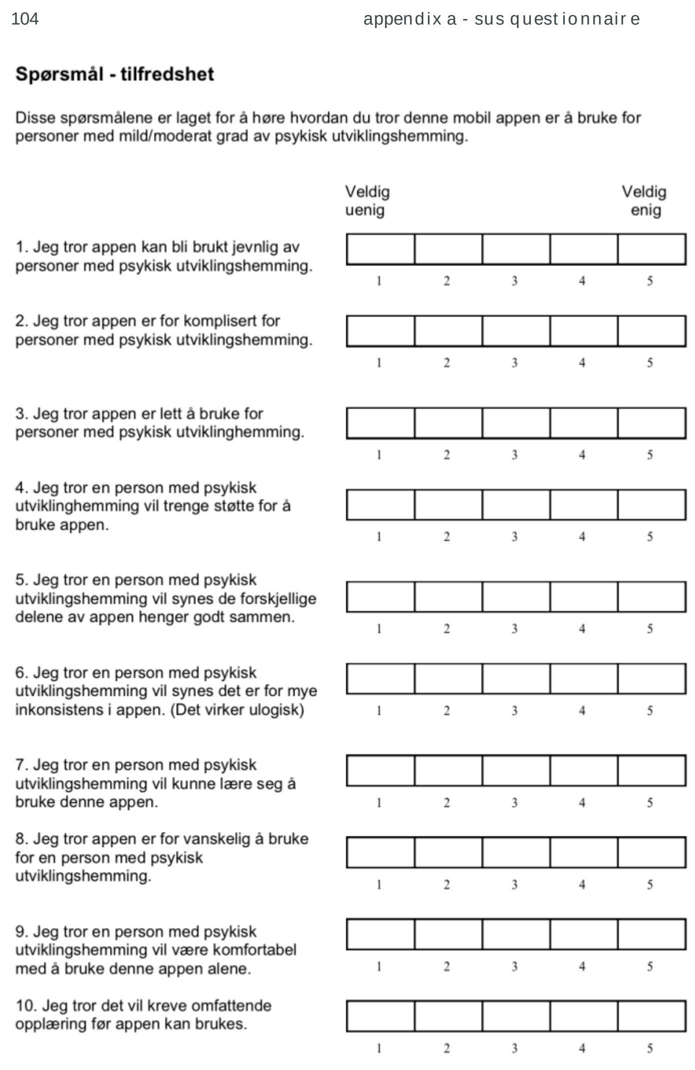
62. RING B. ISOLATED ARROW BUTTON ICON IN CARTOON STYLE [Available from: https://www.mostphotos.com/en-us/54021517/isolated-arrow-button-icon-in-cartoon-style. Accessed: 2022-05-22]

63. Dog N. Crash Bandicoot 2: Cortex Strikes Back. Sony Interactive Entertainment; 1997.

64. 4 CPS. Resident Evil 4. Capcom; 2005.

Appendix

SUS questionnaire



104 appendix a - sus questionnaire

Spørsmål - tilfredshet

Disse spørsmålene er laget for å høre hvordan du tror denne mobil appen er å bruke for personer med mild/moderat grad av psykisk utviklingshemming.

	Veldig uenig				Veldig enig
1. Jeg tror appen kan bli brukt jevnlig av personer med psykisk utviklingshemming.	1	2	3	4	5
2. Jeg tror appen er for komplisert for personer med psykisk utviklingshemming.	1	2	3	4	5
3. Jeg tror appen er lett å bruke for personer med psykisk utviklinghemming.	1	2	3	4	5
4. Jeg tror en person med psykisk utviklinghemming vil trenge støtte for å bruke appen.	1	2	3	4	5
5. Jeg tror en person med psykisk utviklingshemming vil synes de forskjellige delene av appen henger godt sammen.	1	2	3	4	5
6. Jeg tror en person med psykisk utviklingshemming vil synes det er for mye inkonsistens i appen. (Det virker ulogisk)	1	2	3	4	5
7. Jeg tror en person med psykisk utviklingshemming vil kunne lære seg å bruke denne appen.	1	2	3	4	5
8. Jeg tror appen er for vanskelig å bruke for en person med psykisk utviklingshemming.	1	2	3	4	5
9. Jeg tror en person med psykisk utviklingshemming vil være komfortabel med å bruke denne appen alene.	1	2	3	4	5
10. Jeg tror det vil kreve omfattende opplæring før appen kan brukes.	1	2	3	4	5

Figure 1: Appendix - SUS questionnaire