NTNU

IT2901 - INFORMATICS PROJECT II
REPORT

COLOROPHONE

GROUP 12:
Fauchald, Erlend
Iversen, Iver
Skylstad, Thea Karin
Sørli, Stian
Thorsen, Jonas Einar
Verma, Shivam

May 30, 2018

Abstract

This report describes the development of a virtual reality mobile application and the web interface to manage it.

The application was made for Colorophone to enable a study, researching whether it is possible for people to interpret visual information through sound, and whether doing this over time will change neurological paths in the brain. Colorophone's end goal is to create an affordable and easy to use tool for blind people.

The virtual reality mobile application lets users explore virtual rooms by moving their head. The application will play different sounds depending on the color and distance of the object the user is looking at. The aim is for the user to understand what they are looking at by interpreting the sounds generated by the application. Therefore, the exploring is done without visual input. A questionnaire is also displayed and answered in the application to collect data for the study.

The rooms are uploaded through a web interface where answers to the questions are displayed along with a page for administration of users and a settings page for changing the way visual input is converted to sound.

All initial and later requirements from the customer were implemented and tested, and the customer expressed his satisfaction with the final version of the product. The acceptance test result can be seen in appendix E.2.

Preface

This paper was written by students at the Norwegian University of Science and Technology (NTNU) as part of the course IT2901 - Informatics Project II. We would like to thank the Colorophone project for taking us on as a part of the team, especially Dominik Osiński for being both motivating and understanding.

[illegible]
Erlend Fauchald

[illegible]
Iver Iversen

[illegible]
Stian Sørli

Jonas Thorsen
Jonas Thorsen

[illegible]
Thea Skylstad

[illegible]
Shivam Verma

Trondheim, 30.05.2018

Contents

List of Tables

List of Figures

Introduction

Imagine being blind. Right now you are probably envisioning stumbling around in the dark, not really knowing what is around you. Maybe you are thinking about all the beautiful things in the world that a blind person will never see. But have you ever considered the simple things like the fact that a blind person might not even know the color of the t-shirt they are wearing? Then consider this; what if it is possible to let blind people see color?

Colorophone is a project that want to help blind people by developing an affordable sensory substitution device [3]. The device is built around the concept of color sonification, that is, conversion of color into non-speech sound. Colorophone is developing the conversion algorithm and researching the effects of sensory substitution on the human brain. The project end goal is to induce synaesthesia [6] in the users of the device. They want the users to actually experience color and not just interpret the color from the sounds they are hearing.

This report will describe the work of a developer team hired by Colorophone to create a virtual reality (VR) mobile application that will be used in further research.

1.1 Colorophone

The Colorophone project was funded and is lead by Dominik Osiński, assistant professor at the Norwegian University of Science and Technology.

Over the last few years more people, both students and university staff, have joined in on the Colorophone project. Colorophone's focus is to give the visually impaired the ability to see colors through sound. They are doing research into the brains plasticity, specifically on whether it is possible to train the brain into processing audio stimuli in the visual cortex. The expected result is that sound could not only be interpreted as color, but actually perceived as color.

Colorophone has already developed a wearable prototype. The Colorophone device, as seen in figure 1.1, is created as a set of glasses with a camera that connects to a phone. The visual input from the camera is processed by a mobile application that converts the input to the appropriate sound depending on the color of the input (Sonification). This application is called CLRF Navigator and was made by a group of students in 2017 as part of their bachelor project. The sounds are played through a bone conducting headset to let the user retain their sense of hearing while using the device. Colorophone chose to use mobile phones to make the system affordable as phones are widespread and not necessarily a new investment for the user. The application can also use a mobile phone's embedded camera, making it more convenient for the user.

The algorithm that is used to convert color into sound is still being tested and is constantly optimized. No real quantitative research into the effect of using the Colorophone device over time has been done. That is where this project begins.

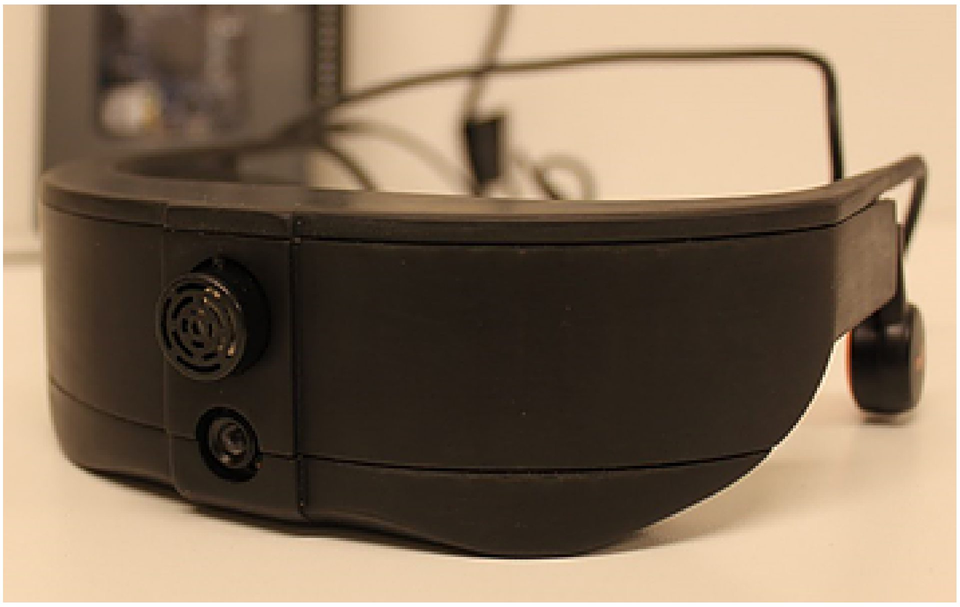


Figure 1.1: The Colorophone device. Photo: Dominik Osiński

1.2 Problem Description

Colorophone have started a research project in collaboration with the Jagiellonian University in Krakow. They want to conduct a study where they will examine how the brain changes after repeated use of the Colorophone device over three months. Around thirty volunteers will use the device daily and MRI scans will be taken of their brains both before and after the three month period.

The study will be conducted by a research group with background in psychology at the Jagiellonian University. The research group has created a customizable laboratory for the study. The laboratory is simply a room consisting of walls and objects of different colors that can be moved around to change the layout of the room. Here they will conduct blindfolded experiments with the Colorophone device, trying to train the volunteer's brains to link the auditory input to the different colors.

Colorophone hired the development team to create a virtual experiment environment that simulates the laboratory at the Jagiellonian University. This way the volunteers can do the daily exercises at their own convenience and from home. The virtual experiment environment will reduce the impact of the study on the lives of the volunteers as it does not require scheduled meeting and possible travel time. It will also free up resources for the research group as they do not have to meet with all 30 of the volunteers every day to take them through the experiments.

The virtual experiment environment was to be created as a mobile application using the existing algorithm for converting colors to sound (Sonification algorithm). The virtual rooms used were also to be designed by the research team and not by the development team. The volunteers will be equipped with VR headsets that they can put the phones into to simulate VR glasses. The virtual rooms used were also to be designed by the research team and not by the development team. When the study starts the volunteers will be equipped with VR headsets that they can put the phones into to simulate VR glasses.

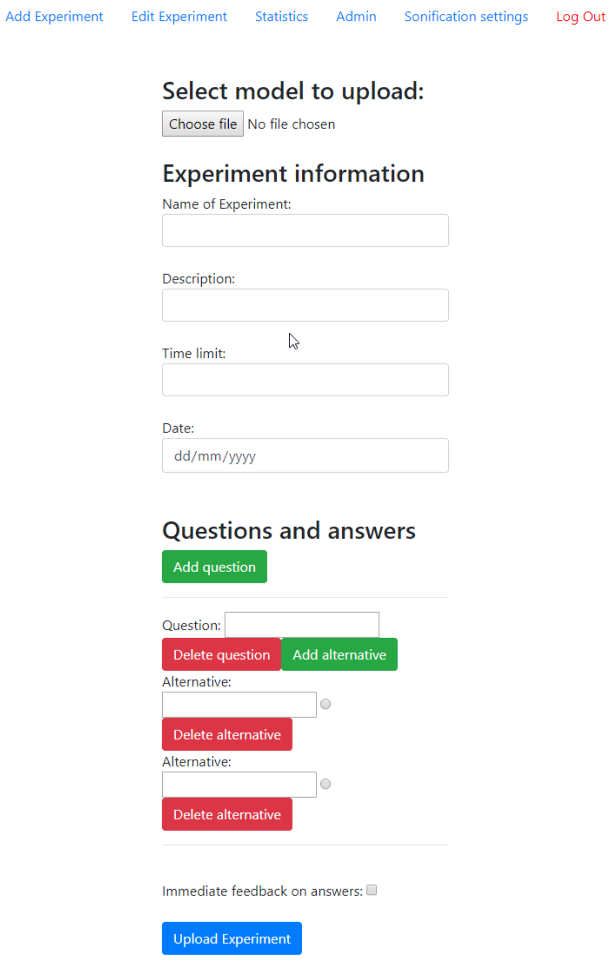
1.3 The project

The development team consisted of six student and this project was undertaken as a completion of their studies in Informatics. The objective was for the students to gain practical experience with software development as part of a self organized team, while working to carry out a software development project for a customer.

All team members contributed with different skills, but all have extensive experience with teamwork and software development as well as some experience with simple web development, database management, human-machine interaction principles and different programming languages.

The customer, Colorophone, wanted a virtual test environment to use in their research. Since the finished system will be used together with tests in an actual laboratory, the experiences had to be somewhat consistent. Therefore, the team had to adhere to quite specific requirements for the experiment phase of the system. Additionally, the customer wanted a way to manage the experiments and to see the results. As the Colorophone project lacked expertise in software development, it was up to the development team to figuring out the best way of accomplishing this while at the same time trying to accommodate Colorophone's other wishes.

The finished system consists of the mobile application, and an associated web application. The web application configures and manages the mobile application and it's content. The mobile application was created with the Unity game engine and is the environment in which the experiments will be conducted. The research group can create new experiments and configure them in the web application. It is possible to add instructions, multiple choice questions, time restrictions and select which days the given experiment should be available. This can be seen in figure 1.2. Each experiment has an unique three dimensional (3D) model of a room simulating the laboratory. This model is created by the research group and uploaded to a web server through the web application. The 3D models are made in accordance with guidelines made by the developer team.



Select model to upload:

No file chosen

Experiment information

Name of Experiment:

Description:

Time limit:

Date: dd/mm/yyyy

Questions and answers

Question:

Alternative:

Alternative:

Immediate feedback on answers: ☐

Figure 1.2: Screenshot of the web application's add experiment page.

The mobile application opens to a two dimensional (2D) menu, as shown in figure 1.3. Here the user can choose which experiment to do. The web server provides the mobile application information about which experiments are available and the mobile application then displays these in the menu. When an experiment is selected by the user, the mobile application downloads the 3D model for that experiment. The user is shown instructions on how to mount the phone into the VR headset. The mobile applications then enters 3D mode and a screen presenting the instructions for the experiment is shown. The user will be asked to confirm when they are ready to start the experiment, and when they do, the mobile application loads the virtual space. The user will the be permitted to explore for the time set in the web application. After the time has run out, the user will be asked to answer questions about the room they were exploring. The questions are presented as they are shown in figure 1.4. The mobile application then sends the results from the questions back to the web server and the research group can then access them from the web application.

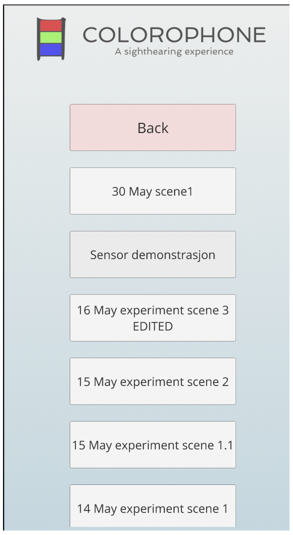

COLOROPHONE
A sighthearing experience
Back
30 May scene1
Sensor demonstrasjon
16 May experiment scene 3 EDITED
15 May experiment scene 2
15 May experiment scene 1.1
14 May experiment scene 1

Figure 1.3: Screenshot of mobile application's select experiment menu.

The researchers can configure the sonification algorithm and manage the different users through the web application. A user given admin privilege will be able to view all created experiments. In admin mode the experiments will be conducted without the ”blindfold” so the researchers can ensure that the models are correct. This will also make it easier for Colorophone to test different sonification algorithms.
The customer wanted the team to implement an algorithm for converting distance to sound (Distance sonification) in the mobile application. Distance sonification was not part of Colorophone's original vision and thus the Colorophone device did not have this feature. The customer had already designed the hardware components for using distance sonification with the Colorophone device, but it was not integrated with the existing CLRF Navigator app. Colorophone therefore wanted the developer team to implement distance sonification in the CLRF Navigator app as an additional project. The team agreed and called the

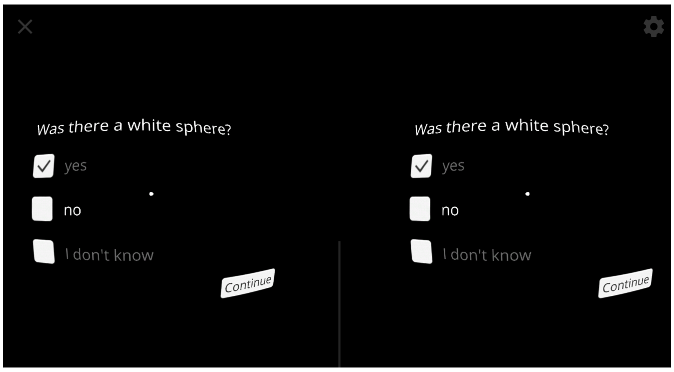


Figure 1.4: Screenshot of mobile application showing questions after an experiment.

project CLRF BLE. The hardware transfers the distance information to the phone through Bluetooth low energy. CLRF Navigator was expanded to receive the information and to translate it through the distance sonification algorithm to sound. This was done in collaboration with the electrical engineers that were working on the hardware components. How this was done is described in appendix A.

1.4 About this report

This report gives a detailed description of the development and solution for the VR project given by Colorophone. Chapter 2 explains what was done in the prestudy and shows the results of the prestudy phase. Chapter 3 describes how the project was organized and managed. The final requirements are described in chapter 4, while chapter 5 demonstrates how these are solved. Quality assurance by testing is assessed in chapter 6. Chapter 7 gives some final toughs after the project. Information about CLRF BLE can be read in appendix A.

1.5 Definitions and Abbreviations

3D-model Room with walls and objects created in 3D being used in the experiments.

AWS Amazon Web Services.

BLE Bluetooth Low Energy

CLRF Navigator The project worked on by a bachelor group in the spring of 2017 resulting in an app.

Experiment Is defined as the sequence of introduction to the task, exploring of a room in VR and answering questions when the exploration is done. Every experiment includes a name, description, date of deployment, time limit, 3D-model and question(s) and alternatives.

Gaze click When the user looks at an interactive object, a timed cursor starts and as soon as the animation is finished, a click is simulated.

Happy path A default scenario in which no exceptional or error conditions arise.

Mobile application An application running on a mobile device.

Research study The study that will be run by the Colorophone project.

Reticle In VR, the reticle is a mark that shows where the camera is pointing.

RGB Red Green Blue. An additive color model based on the 3 colors red green and blue

RYGBW Red Yellow Green Blue White. A custom color model based on the eye's properties. Used for sonification purposes.

SDK Software development kit.

Sonification The process/ algorithm of converting information about color and distance to sound. The color and distance sonifications are two separate algorithms.

Stack (software) Group of programs that work in tandem to produce a result or achieve a common goal.

Study volunteer The volunteers participating in the research study.

UI User interface.

UML Unified Modeling Language.

VCR Version Control System.

VR Virtual reality.

Web application Server with operating system, web site, database and file management.

Prestudy

An extensive prestudy was conducted by the development team to learn technologies necessary to fulfill the requirements given by the customer. The development team wanted to understand what was possible to make, how it could be implemented, be able to estimate the time scope, and choose the optimal technologies to use so that more refined requirements could be defined in together with the customer.

2.1 Exploration of existing technologies

A study of alternative technologies was conducted in order to get a broad overview of technical possibilities and potential solutions to the problem description. The chosen technologies should not restrict the project scope. By making good choices, the technologies could free up the teams resources by providing ”off the shelf” solutions. This would make it possible to expand the project scope. To make it easier for the researchers to work efficiently it was decided that the system should contain a web application for experiment data, that would also work as an interface for controlling the mobile application. The mobile application would have to implement VR and be editable and extendable. The technologies explored were therefore split into two parts. One for the VR application running on the phones and one for the server running on the web.

2.1.1 Mobile app frameworks

For the mobile app framework, it was necessary to use a well documented engine which could be used for prototyping and developing in VR as fast as possible. A solid framework for creating 3D models was also necessary. The project required building a big number of the models and it should therefor be as simple as possible. In addition, web integration was key, because the mobile application would have to download all the information about the experiments and the 3D rooms after it was deployed. The following is a list of technologies the team explored.

ReactVR

ReactVR is a framework for building VR apps based on React syntax which some of the team members had previous experience with. Despite this, it would be time consuming to use React VR as the team observed that it does not have a GUI for development and it does not have a well developed interface for communicating with a server. ReactVR has limited functionality for loading external assets and models. It seemed like ReactVR would require a lot of resources to use as most of the functionality needed would have to be implemented by the team.

Unity

Unity is a more developed and widely used game engine for building VR-apps. It has an GUI-interface which would reduce the complexity of implementation greatly. It also has a general interface for communicating with external servers, and it has built-in functionality for loading external models and assets at runtime. The research group also tested unity and stated that they found it fairly easy to create their own models in a 3D environment via free plug-ins. It uses C#, which is a well documented code language.

2.1.2 Web application

The web application needed to be hosted on a server. A database with a web interface was needed for setting up new experiments with questions, answers, date of deployment and other data connected to each 3D environment. It should also be possible to upload the model through the web-page and save it on the server.

ReactJS with Apache

ReactJS with Apache is a relatively new, but well documented framework. It has a good development environment with built in functionality for easily deploying the application on a Apache server. It also has an interface for integrating a database. However, its functionality for handling file transfer to the server was not well developed.

LAMP-stack

LAMP-stack is a framework that also run on an Apache server. It uses PHP which is a server-side scripting language widely used for external communication, satisfying the requirements for transferring data to the mobile application and the server. It comes with a built in database which supports communication to the web site through PHP.

2.1.3 Decision

The team chose Unity as the mobile application framework. This is primarily because of its support for developing ’plug-and-play’ VR applications on Android phones. In addition, it has a large ecology of free API’s, which is used client-side to both create modular 3D environments and to package these for simple web transfers. Unity is also one of the game engines recommended by Google for developing Cardboard games [4], and this means that it is well supported and has a properly documented codebases. A Cardboard game is an applications run on a phone in VR headsets, similar to the one used by Colorophone.

For the web application framework, the file handling turned out to be the deciding factor. This project would have to support a lot of files being transferred to and from the end users, making this a very important factor. Due to ReactJS and other frameworks inability to fulfill this requirement, the LAMP stack was chosen as the web application framework. The framework includes deployment to the server, database support and a running website containing logic for communicating with the database and saving files on the server. The inclusion of well documented source code over time is a consideration taken as well.

2.2 The sonification algorithm

One of the first things the customer introduced was the concept of sonification. Sonification is in essence to use non-speech audio to convey information. In Colorophone, the aim is to convey color information as an audio stream. This is a complicated problem with regards to both mapping and user experience. Many types of visual-to-audio sonification methods have already been developed, for example Eyemusic [1] and The vOICe [7]. The vOICe app scans the field of view from left to right and plays back notes corresponding to the y-axis position of objects in the field of view [7]. This means that it does not play back the picture in real time, it takes a picture, scans it and plays it back to the user with a slight delay. Eyemusic does the same, only with a musical scale(Pentatonic) and different notes corresponding to different colors [1]. From the customer's perspective, these solutions have limited quality of user experience mainly because of this delay between visual input and audio. This is where Colorophone hopes to deliver with its audio stream algorithm(s).

The Colorophone sonification method is based on how the human eye receives and translates colors. The eye has three cone receptors for red, green and blue light, where yellow is a combination of red and green, whereas white is a combination of all colors. The algorithm translates the visual inputs based on how much the eye receptors would be stimulated and gives out an output representing sound. The Colorophone system plays five simultaneous sounds representing the colors red, yellow, green, blue and white. The amplitude of each sound signal is adjusted corresponding to the color intensities of each respective color value. The user identify colors by listening to the intensity of the sounds.

The system starts off by calculating the average red-, green- and blue- values of all the pixels in a rectangular area in the center of the camera's recorded view. A conversion algorithm is applied to convert the RGB values into RYGBW values. These RYGBW values are then used as parameters by the sonification algorithm to adjust the amplitude of each respective sound signal. The sound signals are then mixed and played for the user to hear. This whole procedure is done for every frame captured by the camera, or every time the VR application renders a new frame.

2.3 Alternative solutions for the customer

The team had some issues regarding the limitations of the solution. The goal of the feasibility study was to get a better understanding of what these limitations were, as well as getting a better overview of strengths and weaknesses of different approaches. This was done parallel to choosing the different technologies used in the project and the to processes mutually affected each other.

The development team's approach to conducting the study consisted of formulating four different solutions to the customer, as well as making two ”proof of concept” functioning prototypes. This allowed both the team, and the client, to compare alternatives against each other with regards to their own interests.

By looking at already existing VR-apps and their implementation the team made a picture of what was possible and tried to match that to the problem description and time schedule.

The primary thing the team need to consider was the wish of the client to have new VR models to explore every day. The client wanted a huge number of VR experiments available in the app, and that preferably, new experiments could be fetched from the internet. Because VR is achieved through a game engine (Unity), this is a quite complicated requirement. The discussion that followed led to the most important architectural decision during the prestudy phase. It was necessary to clarify how much involvement the client wanted in making these VR-rooms and the complexity of them (3D/2D). This would also require having a reliable system for designing, uploading and displaying the correct rooms. The following solutions were considered:

Solution 1: The client making all the experiments before installation on target phones

This would be a solution where no dynamic updates are needed, meaning that all 3D-rooms and the rest of the application would be packaged together and installed once. The 3D-rooms could then be made with full freedom with no protocols or restrictions. If new experiments would be added, the study volunteers have to come back to the lab for a new installation. It also means that the client would have to make most of the 3D rooms before starting the study. This would have been the easiest solution to implement, and the one that initially was considered. However, new information stated that the sufficient amount of rooms would not be finished in time for the study period and therefore needed to be loaded after the phones were delivered to the volunteers.

Solution 2: The client making the 3D-rooms in Unity

This would mean a lot of work for the client, but also more flexibility in the design. With this solution, the client would create the 3D-rooms themselves independently in Unity. The room creator then would have to package the room correctly and upload it to a server where it can be downloaded from the application on the target phones. The client would use a well built template and would have to follow a strict, well-defined procedure for creating rooms to minimize the risk of bugs. In addition, these rooms then would have to be linked to experiment data in the web application. In this solution, the mobile app would have a minimal initial size, only consisting of a framework including the sonification algorithm and logic for downloading and unpacking packaged rooms from the web. A concrete scenario could look like this:

1. The client makes a 3D room in Unity from a template provided by the development team.

2. The room is packaged to a Unity 'Asset bundle' with an asset bundle API.

3. The client creates a new experiment by uploading the 'asset bundle' and attaching description, date, duration and questions and answers to it.

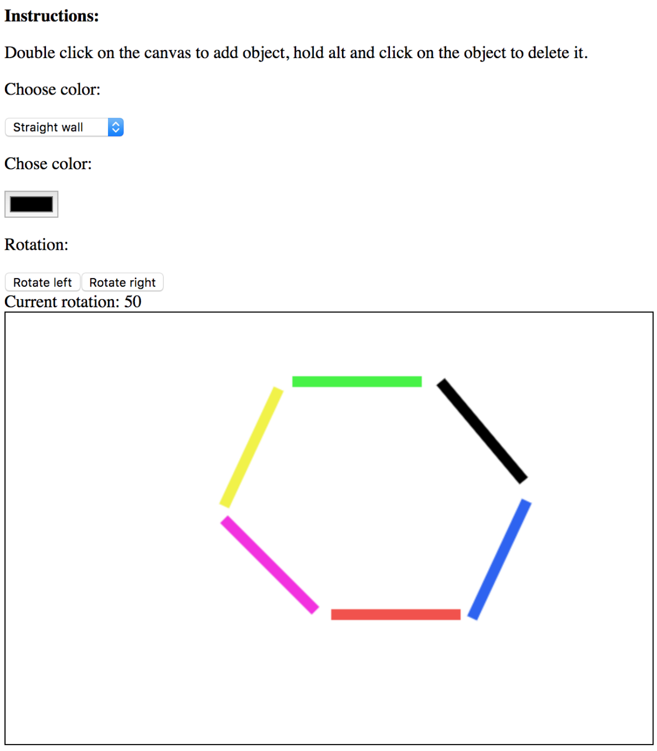
4. The experiment pops up in the menu of the target phone's application and can be run with sonification on top.

5. When the experiment ends, data is sent back to the web server for the researchers to analyze.

A proof of concept demo was made for this solution, and was shown to the customer. This solution would allow full creative freedom in the room creation process, but it would increase the risk of human errors crashing the system.

Solution 3: The client configures parameters for each new experiment

Through a 2D web-application the client could add a new experiment each day by configuring a set of parameters like the number, position and color of objects. These settings would then have to be saved in text format and parsed by the Unity application on the target phones during run time. The unity application would then use the parameters to position different elements creating a 3D-room. This would require less memory, bandwidth and computational resources compared to loading complete rooms. It would also save implementation time for the development team, room designing time for the client and decrease room for error, but would limit flexibility in creating the rooms. Figure 2.1 shows a proof of concept demo that was made of this solution.



Instructions:

Double click on the canvas to add object, hold alt and click on the object to delete it.

Choose color:

Straight wall

Chose color:

Rotation:

Current rotation: 50

Figure 2.1: Screenshot of proof of concept for Solution 2.

Solution 4: Automatically generating VR-Images

Another idea was to implement an algorithm that automatically configures different 3D-environments. This could be implemented as randomly generating environments that follows criteria from the client or downloading images available on the web (e.g Google street view) and displaying these. The time to implement would be high, but would save design time for the client. The flexibility would depend on the complexity of the criteria.

Result

Solution 2 (The client making the 3D-rooms in Unity during development phase) was chosen. In this decision, the customer's desire of having full freedom in creating experiment rooms after the study period had started was important. It gave the researchers more time and full flexibility in creating the rooms, which was needed for conducting a sufficient study. Also, solutions 3 and 4 would only allow for certain pre-defined objects and colors to be present in the experiments, and this was an unwanted constraint by the customer.
The solution would take more time to implement than the others considered, and it was clarified that this could affect the completion of other requirements.

The feasibility study provided the team with a good exercise in understanding the frameworks chosen in the exploration, making it clear what was possible and not in the technology utilized.

Process

This chapter describes the process chosen for building the system. The development methodology is first described, followed by the roles and responsibilities of the development team. After this comes a description of risk- and time management issues and how they where solved. A list of tools used in both project management and development is presented at last.

3.1 Method overview

When formulating a development method, it is important to consider the actual task at hand. An outline of the systems requirement was formulated at the beginning of the project. It was also unclear what priority the different requirements had, and if they where within the time scope. The development team was small, and sometimes had to prioritize their time on other subjects.

The project architecture and requirements were iteratively built throughout the whole development process as neither the development team nor the customer had experience with VR technology. Therefore, the team chose practices that would maximize learning both for developers and customer. Lastly, the team wanted to minimize time spent on functionality that was not vital in order to stay within the limited time scope.

All these factors made it clear that Lean Software Development[10] with selected Scrum practices was the formal framework that suited the needs of all stakeholders. This section will outline how Lean Software Development with minor Scrum principles and practices were followed, or in some cases, not followed. The methodology contains values and principles, as well as a formalized development method including different mandatory activities and practices.

3.1.1 Values and principles

When the group set out to define the project's values, Lean Software Development was used for inspiration. The seven first chapters of "Lean Software Development: An Agile Toolkit" were adapted to the project, and taken into consideration when developing. The ideas of minimizing waste and maximizing learning were especially important for this project. The Lean principles are:

1. Eliminate waste: Due to the limited time scope, prioritizing activities held key importance. As a consequence, trade-offs had to be made in testing-, planning- and management practices, which will be outlined later in this chapter. Waste is something that appeared during development. Unity is a closed source game engine, which in practice means that estimating which possibilities and limitations exist is complicated. In addition, the project could turn into a black box as it grows in size, something that is devastating for debugging. In other words, wasting time is a pitfall that shows up at many occasions in Unity development. Recognizing and eliminating waste became an important part of the project, and was mainly done by always thoroughly checking the Unity manual and forums when implementing a new feature, making sure that the proposed solutions would actually work within the engine.

2. Amplify learning: Throughout development, the team was in frequent contact with the customer. Learning about the the limitations of the chosen technologies and solutions in cooperation with the customer made communication efficient. By having the customer informed, both requirements and the customer's overall expectations became more realistic and practical. Also, instead of planning every aspect of development ahead of time, the team always tried out different ideas both to learn and to reach a solution. This is also one of the reasons for conducting the prestudy in chapter 2.

3. Decide as late as possible: When presented with different solutions to a problem (see section 2.3), deciding on one of them is something that should be done last minute. This is because a question of feasibility is always uncertain. When starting the project for example, the team knew nothing about the powerful Asset Bundle API built into Unity. This API was crucial for the whole system design, and was decided upon at a stage in development where the team had already implemented alternative solutions. Having the possibility of a change of mind proved very useful in the process.

4. Deliver as fast as possible: Presenting both working, and proof of concept demos as fast as possible was an important part of the customer contact in this project. Being transparent about progression and setbacks is important in both formulating new requirements and managing expectations. Always having a working prototype ready also made the product more modifiable and solid. This was done by having working demos of the newest implementations for the customer every two weeks.

5. Empower the team: As the development team was small and self organized, empowering and gaining ownership of the product came naturally. The customer was also concerned with the team gaining ownership, giving the team freedom in the implementation. The process of creating and the evaluating the solution was also a contributing factor to this.

6. Build integrity in: Both conceptual integrity and build integrity is important. Conceptual integrity is interpreted as solving the problem and learning about its domain simultaneously. Build integrity was achieved with working demos and continuous code refactoring.

7. See the whole: Every team member should at all times be up to date on the product, as well as having a good understanding of the agreed methodology. To accomplish this, daily stand-ups where every team member explained what they were working on were held. The team also met together to work minimum two times per week so that everyone could get a complete understanding of how the system would be. Inconsistencies were discussed in retrospective meetings held every two weeks. A preliminary team guidelines document was also signed at the very beginning of the project to help the team get started in this effort. This guidelines document can be found at Appendix D.

3.1.2 Practices

At the core of the methodology were two week long iterations (Sprints), with some flexibility regarding holidays and feature completions, and retrospectives at the end of each iteration. Formalizing the actual activities more than this was unnecessary, especially in the long run, because of the differences in workflow between certain phases of the project. However, some practices and conventions were agreed upon and are outlined here.

Code Conventions

Coding conventions were defined for each programming language in order to increase the readability of the code and make the codebase easier to maintain. The team agreed upon these conventions:

C-Sharp: For consistency, class-, method- and variable names were to be in camel casing. Help new developers understand the program and in order to enable future development, names of classes, methods and variables should be descriptive. All comments should be written in English.

PHP: There are two big coding styles that are recommended to use in PHP [9]; the dual procedural and the object-oriented and procedural approach. The Object-oriented and procedural code style is used throughout all PHP code in the project. This is because the need for reuse of code in functions used between different components. Variable names are written in camel case.

Javascript: Javascript is written in an object-oriented way where all code is wrapped in a function. This follows the natural use of the language where different functions often are called by interactive elements. Variable names are written in camel case.

Database Parameter names in the database are written in underscore case.

Feature Branching

Git was used as a version control system for the development of the web application. Feature development took place in dedicated feature branches. This made it easier for developers to work on a specific feature without disturbing the rest of the code base, and the master branch would never contain broken code which allowed for continuous deployment.

Each feature got its own branch. Once a feature passed the definition of done, it got merged into a dedicated branch for development. At the end of each sprint the development branch got merged into the main branch.

Definition of requirements

Prioritized requirements provided by the customer were formalized as user stories. Each story represents what the customer wanted to achieve. After defining requirements from the customers by constructing user stories, they where further decomposed into functional requirements (FR). Each user story was independent and represented a potential product increment while the FRs represented development tasks. The user stories gave an overview of the whole project, while the FRs gave an overview of what needed to be done. User stories were picked at the beginning of each sprint and placed on a kanban board which the development team would pick from based on their priority and work on until completion.

Estimation

After decomposing user stories into functional requirements, the functional requirements were analyzed one by one and given a relative estimation score from 1-100. The group quickly concluded that these estimations added little value to the planning phase, and dropped them. The estimations were considered waste, because at this point, the team had a narrow understanding of the potential implementation and the estimations would simply be too inaccurate to consider.

Acceptance criteria

The functional requirements were presented to the customer to clarify the team's understanding of the requirements. After they were agreed upon by both parts, they served as acceptance criteria for the system.

Definition of Done

A definition of done assures that a feature can be accepted by the different stakeholders. Before a user story was considered done, it was required to have passed the acceptance test, code review and be able to integrate into the system without breaking it.
If a sprint ended with a user story not meeting the definition of done, that user story was moved to the next sprint's backlog.

Sprint planning

Sprint planning took place on the first day of each new sprint. During these meetings the team agreed on user stories and the coherent functional requirements to work on. User stories and FRs would either be transferred from previous sprints or selected from the backlog, based on their priority.

Through the use of iterative development and planning the group was able to modify the focus of each sprint as a response to the customer's shift of focus. This was very important in the early sprints of the project as there was uncertainty among the developers of what was possible and what the customer wanted. The sprint could therefore focus on gaining more knowledge and information about the system's technologies on the part the customer was interested in.

Retrospectives

During the first two sprints the team experienced the need for a platform to express their concerns. Concerns were expressed during work meetings in an unstructured manner. This made it difficult to remember which concerns was discussed and which solutions were agreed upon leading to time spent discussing the same issues several times. The solution was having retrospective meetings at the end of each sprint. During these meetings the team reflected on work done during the previous sprint and made decisions on how this can be improved in the next sprint. Issues, risks and mitigation strategies were documented so the team had a structured overview of what was done during each sprint, changes and issues experienced and how to adapt to the different changes. Any issues or risks brought up during a retrospective would be mitigated during the next sprint. The summary notes of the retrospective meetings from sprint 3 and onwards can be found in appendix C.

Review of code

The team had extensive focus on reviewing each other's code. This ensured a higher quality of the code, but more importantly increased the teams familiarity with the whole code base. Pair programming was used to both make code writing more effective and share knowledge and familiarity with the code written. Code reviews where done before the team considered a user story done. This is explained further in section 6.1.

3.2 Roles and responsibilities

The team is divided into roles where every team member have their own area of responsibility. This is to ensure responsibility is evenly divided among the team members, and having one person responsible of ensuring work is done on their delegated area. This doesn't necessarily mean that they have to do all the work in their section, but they have to have an overview of what have been done and what needs to be done by delegating work to other team member if it is needed in their area of responsibility.

Shivam - Scrum master, Group leader, Server and Web developer
Have control over the scrum process and manage conflicts and disagreement between the team members. Also initializes and enable communication amongst the team. Need to attend group leader meetings and report back to the group. Responsible for the server and making the website.

Erlend and Thea - Report manager, Unity developer
Responsible for writing the report and ensuring that the team also work on the report and document properly so that it could be used in the report later. This imply having an overview of what has been written as well as what needs to be done in the future. Responsible for implementing the VR environment and it's user interface.

Stian - Bluetooth implementation of distance sensor
Responsible for planning and implementing Bluetooth communication in the App created by the Colorophone bachelor-group of 2017.

Jonas - Test- , Database- and website developer
Responsible for understanding what, when and how to test and implement the tests or enable the team to do so. Was also responsible for creating the database and building the website.

Iver - Sonification developer
Implementing sonification in the VR environment according to specifications given by the customer.

3.3 Risk management

A risk analyses was conducted early in the project period to identify and address risks for both the team and the project. The risks where prioritized by analyzing consequences and probability of each risk. Identified risks along with mitigation strategies are described in the following tables.

Team

Risk Nr	Description	Mitigation	Response	Responsible	Priority

T8	Some team members not working as much as others	Have a timetable showing how much each team member works. Have a clear definition of how much to work each week and penalty for not complying.	Catch up on time not spent working in previous weeks.	Whole team	1
T2	Issues with version control	Define clear rules on team member should use Github.	Reiterate the rules and explain why they need to be followed. Fix inconsistency as result of wrong use.	Stian	2
T3	Different level of skill/ understanding of the implementation	Scheduled work-meetings and pair programming	Other team member helps with getting the individual(s) on the same level of understanding	Whole team	3
T1	Team member cannot attend to scheduled meeting due to unpredicted circumstances	Not plan meeting if uncertain. List of available time slots for the whole group	Notify the team. Share important information from home. Schedule substitute if needed.	Whole team	4
T6	Some team members working more than others	Timetable of hours spent by the team members.	Social pressure from the team. Make up for time spent not working.	Whole team	5

T7	Members not sure of what to do and don't ask.	Trello, Information meeting, Retrospect meeting and scrum standup forcing sharing of information.	Scrum master get notified and handles the situation.	Shivam	6
T4	Team-member falling out of the project.	Work-meetings. Communication channels as Slack and overview through Trello.	Notify supervisor.	Whole team	7
T5	Conflict between team members	Work-contract	Majority vote	Whole team	8

Table 3.1: Team

Project

Risk Nr	Description	Mitigation	Response	Responsible	Priority
P8	Too broad scope.	Having a feature freeze 2 weeks before project end, making this clear to the customer. Have prioritized list of requirements the team can fulfill (Plan A and plan B).	Schedule meeting with customer to narrow the scope based on what customer want the most.	Whole team	1
P2	Misconception of requirements.	Close communication with customer. Agile development.	Clarify with customer and change development plan to adapt to change.	Whole team	2

P3	Unrealistic requirements.	Agile Development.	Rewrite requirement to be realistic. Adapt project plan accordingly.	Whole team	3
P1	Change of requirements.	Agile development.	Adapt project plan to change.	Whole team	4
P7	Leakage of classified information.	Closed github provided by the client. Confidentiality agreement.	Investigate the leakage, prevent it from happening again and report to client.	Whole team	5
P4	Malfunction of tools from client.	Use carefully.	Contact client for new tools. Work with another team member in the meantime.	Whole team	6
P10	Security breach.	Secure potential breaches.	Hash passwords for website. Prevent SQL-injections.	Shivam, Jonas	7
P6	Missing deadline.	Document and reminders of important deadlines.	Contact the responsible person and get new deadline.	Shivam	8
P5	Customer not available.	Plan meetings for the whole project duration early.	Try to communicate electronically.	Whole team	9

P9	Data loss.	Backup data.	Regularly fetch logical backup of database. Have the research group save a copy of every 3D-model locally on their computer.	Shivam, Jonas	10

Table 3.2: Project risk management

3.4 Sprint time management

A high level time plan of project milestones was created early in the project. This follows below.

Event	Start date	End date	Summary
Sprint 1	31.01	18.02	Get an overview of the project context and initial requirements. Plan how to find solutions.
Report draft 1	15.02	15.02	The first report draft needs to be turned in.
Sprint 2	19.02	04.03	Further understand stakeholders, project and refine requirements. Explore solutions. Plan management of the team and the project.
Demo 1	23.02	02.03	Demonstration of what's been developed so far
Sprint 3	05.03	18.03	Finish exploration of solutions. Present solution to customer and agree on initial solution.
Report draft 2	18.03	18.03	The second report draft needs to be turned in
Sprint 4	19.03	01.04	Start work design and implementation of solution. Update requirements and priorities if needed.
Sprint 5	02.04	15.04	Continue implementation of solution. Update requirements and priorities if needed.
Demo 2	13.04	20.04	Demonstration of what has been made worked with so far.
Sprint 6	16.04	29.04	Continue implementation. Update requirements and priorities if needed.
Sprint 7	30.04	13.05	Freezing requirement and finishing implementation of existing requirements.
Final presentation	15.05	15.05	A tentative date for when the project should be presented on stand.
Sprint 8	14.05	30.05	Last adjustments, implementation/bug fixes and testing.
Final submission of report	30.05	30.05	The project should have ended and the report turned in

Table 3.3: Time plan of the project

3.5 Backlog

The backlog consists of all the productive tasks carried out by the team yet to be finished. This includes all the requirements given by the client, work to be done on the report and feasibility exploration tasks to conduct the feasibility study in section 2.

Feasibility exploration tasks are small tasks connected to exploration of possibilities described in chapter 2. They where made as a response to the client's desire for exploring alternative solutions. These items were not given any priority or time estimation as the factors were too uncertain to predict in a responsible manner at that point in time. The task were defined done when a demo where made or knowledge about exactly how the problem defined by the task could be solved could be implemented.

An additional task of implementing a distance sonification algorithm was mentioned early and considered a low priority task, it later became clear for the customer that this task was more important than first expected. This task was considered done when the customer could execute the implementation with their Colorophone Headset. This task is described in Appendix A.

Task	Description
S1	User stories connected to the study volunteers. Described in section 5.
R1-R6	User stories connected to the researchers. Described in section 5.
FE1	Explore if it is possible to spawn objects into 3D space given specification in a text file. Make a demo.
FE2	Explore if it is possible to make a user interface to create 3D models and convert these specification to a text file. Make a demo.
FE3	Explore if it is possible to send data between a server and a unity app built on android. Explore the difficulty of communication between a web interface and the same server
FE4	Explore how difficult it is to make Unity AssetBundles from a 3D-model. Make a demo.
FE5	Explore if it is possible to send AssetBundles from a server to a built Unity app on Android. Make a demo.
FE6	Explore if it is possible to make a question interface with alternatives for answers. Make a demo.
FE7	Explore the possibility of implementing the sonification algorithm from the CLRF Navigator app in Unity. Make a Demo
FE8	Explore how you can change/ tweak sonification settings.
FE9	Explore If it is possible to have pictures on the wall in the 3D-models and this would work with the sonification.
FE10	Explore If it is possible to have a 2D menu, switching into 3D and VR.
CB	The CLRF BLE sub-project to implement Bluetooth distance sensor in the CLRF Navigator app. This is described further in appendix A.
RE	Work do be done on the report.

Table 3.4: Backlog

Backlog time management

To get an overview and manage all the different tasks within the time scope, a timetable was made and changed continuously after each sprint. Backlog items was worked with during their respective sprints. In the case of a item not being finished, the item was moved to the next sprint.

Event	Date	Backlog items	Done	Moved to next sprint
Sprint 1 - Exploration	31.01-18.02	FE7, FE1, FE8, FE9, FE10	FE10	FE7, FE1, FE8
Sprint 2 - Exploration	19.02-04.03	FE7, FE1, FE8, FE9, FE2, FE3, FE4		FE7, FE1, FE8, FE9, FE2, FE3, FE4
Sprint 3 - Exploration	05.03-18.03	FE7, FE1, FE8, FE9, FE2, FE3, FE4, FE5, FE6	FE7, FE1, FE8, FE9, FE2, FE3, FE4, FE5, FE6, RE	
Sprint 4	19.03-01.04	S1, R1, R2, R3, R4, CB	R1	S1, R2, R3, R4, CB
Sprint 5	02.04-15.04	S1, R2, R3, R4, CB	R2	S1, R3, R4, CB
Sprint 6 - New features freeze	16.04-29.04	S1, R3, R4, R5, R6, CB, RE	CB	S1, R3, R4, R5, R6
Sprint 7 - Code freeze	30.04-13.05	S1, R3, R4, R5, R6, RE	S1, R3, R4, R5, R6	RE
Sprint 8 - Finishing report	14.05-31.05	RE	RE	

Table 3.5: Sprint plan

3.6 Tools

This is a presentation of the different tools used to complete this project. Some of them were requested by the customer and the rest was chosen based on research, experience and necessity.

3.6.1 Development

Unity Game development platform used for developing VR through scene creation and scripting in C#.

Unity Asset Bundle Browser Tool Unity add-on for creating asset bundles.

Google VR SDK for Unity SDK for VR development in Unity for mobile applications.

Github Hosting of project.

Git Version control system, used for source code management for the web application.

Unity Collaborate Version control system, used for source code management of the mobile application.

Android Studio For developing the CLRF Bluetooth project.

3.6.2 Project management

Slack Communication between team members with a separate channel for communication with the client. Also used for sharing of files between the client and group.

Skype Communication through video chat with stakeholders at the Jagiellonian University in Krakow.

ShareLatex Online LaTeX editor which allows real-time collaboration. Used for writing of formal documents.

Google drive Sharing and storing documents online with real-time collaboration.

Trello Overview of the project consisting of backlog, state of completion in addition to sprint specific information.

Draw.io Creating UML-diagrams.

Requirements

This chapter describes the applications stakeholders, user stories, functional- and nonfunctional requirements, and use cases. The stakeholders section is a list of people having a stake in the system. The use cases, in collaboration with functional requirements (FR,) should give an overview of how the system behaves. Use cases are made to describe how the user might operate the system. The quality attributes works as guidelines for where the quality focus of the system should be. They are accompanied by quantitative quality requirements (QA). Requirements for the CLRF BLE sub project is described in appendix A.

4.1 Stakeholders

Customer: Colorophone is lead by Dominik Osiński. Throughout the project, Dominik has had contact with the researchers in Krakow and passed on communication both ways between the team and some of the stakeholders.

Researchers: Researchers at the Jagellonian University of Krakow are interested in gathering data from the experiments. After an period of training using the Colorophone VR application at home, the data gathered will be analyzed to make scientific discoveries about how the algorithm affects the brain. To make meaningful discoveries it is important that their needs are adressed.
They are going to use the system to create, conduct and analyze experiments during a 3 month period in the future.

Study volunteers: Volunteers will use the application from home to take part in new experiments every day. Volunteers are chosen to represent the general population. It is also in the interest of the researchers that the application satisfy the needs of the end user in order to recruit- and keep volunteers participating in the research project.

Project team: The application will be developed by students from the Norwegian University of Science and Technology. Absolute time frame and predictable work load is in their interest. They also want to be challenged intellectually and gain experience relevant to future employment. There is also a desire to contribute to design decisions of the system to increase affiliation to the project.

Course staff: This project is part of the course IT2901 - Informatics Project II at NTNU. The course staff includes the course coordinator, lecturers and supervisors. The course coordinator evaluates the final product, while the lecturers and supervisors assisted the teams during the project.

4.2 User stories

User stories were formulated as a way to identify the needs of different stakeholders. The list of user stories defines all the features that the team considered to be within the scope of the project. Acceptance criteria was formulated as functional requirements, therefore the User story table refers directly to the functional requirements table that follows.

Description	**Functional acceptance criteria**
S1: As study volunteer I want to navigate in the app so that I can select and load experiments.	F9, F10, F11, F12, 13, F32
R1: As a researcher i want her sounds based on the color i am looking at so that i can identify objects in the virtual environment.	F1, F2, F3, F4
R2: As a researcher I want the recent sonification algorithm to be used so that I can make meaningful scientific discoveries.	F5, F6, F7, F8
R3: As a researcher I want to add/edit experiment data including questions and alternatives as well as 3D-models to each experiment.	F14, F15, F16, F17, F18, F19, F20
R4: As a researcher I want to change parameters for the sonification algorithm and sensory input.	F21, F22, F23
R5: As a researcher I want to view and analyze the data from the experiments so that I can use it for the study.	F24, F25, F26, F27
R6: As a researcher I want to explore all created experiments in the mobile app via visual input so that I can verify that everything works as expected.	F28, F29, F30, F31

Table 4.1: User stories

4.3 Functional requirements

F1:	The app needs to read colors correctly from the virtual environment.
F2:	The color data should be collected from an area.

F3:	The app should use the correct sonification algorithm.
F4:	The app should play the correct kind of sound.
F5:	The app needs to collect the distance to objects in the virtual enviroment.
F6:	The app needs to use the correct algorithm to play the desired sound.
F7:	The app needs to play sound corresponding to the distance.
F8:	The app needs to be runnable on target devices.
F9:	The mobile app should display a tutorial on the screen after an experiment is selected.
F10:	The mobile app should have a menu for launching experiments.
F11:	The system needs a model for displaying the questions.
F12:	The system needs to support gaze input.
F13:	The mobile app needs to display the questions from the database in a readable manner.
F14:	The system should provide a web interface for adding questions, alternatives and 3D-models for the experiments.
F15:	The system should provide a web interface for adding experiment name, description, date of deployment and time limit for the experiments.
F16:	The system should store experiment data in a database and 3D-model on a file server.
F17:	The mobile app needs to get the questions and alternatives from the system and load them correctly into the questions model in the corresponding experiment.
F18:	The mobile app needs to get the 3D-model from the file server and load it correctly.
F19:	The system needs to provide a template for creating 3D-models so that the researchers can make compatible models.
F20:	The mobile app needs a system for choosing, downloading and running the correct experiment.
F21:	The system needs a web interface to change sonification parameters.
F22:	The system needs to store sonification parameters from the web app in a database and fetch these to the mobile app.
F23:	The system needs logic in the mobile application for obtaining sonification parameters and change the algorithm used accordingly.
F24:	The web application should display the answers from the experiments sorted on both users and experiments.
F25:	The customer should get safe access to the database to structurally collect and analyze data via specific queries.
F26:	The mobile app should send answer data to the web application and it should be saved to a database.
F27:	Data collected should be connected to the user going through the experiment.
F28:	The mobile application needs to differentiate between experiment volunteers and researchers.
F29:	The web application need to have an interface to select which user is a researcher.
F30:	The mobile app needs to show all experiments to researchers, while only today's experiments are available to volunteers.

F31:	The mobile app needs to show experiments with visual input to researchers, but only auditory input to volunteers.
F32:	The mobile app needs to let researchers control who can participate.

Table 4.2: Functional requirements

4.4 Quality attributes and requirements

The system is going to be used by a small group of people in an unknown time period. It is also going to be given to the client for maintenance and further implementation. This affects the quality requirements of the system. This section describes the main motivation for the quality attributes in the system, followed by acceptance criteria for the quality requirement.

4.4.1 Security

The web application is only known by a very limited amount of people and the probability of anyone wanting to attack the system is deemed low. Little effort is put in to mitigate risk in bigger architectural decisions, though basic security mechanisms are needed. The use of the mobile application is limited to volunteers that are asserted to be trusted, and damage that can be done through it will have a low impact. The web application is however available on a server available to everyone. After going through top 10 security risk list from OWASP list the most likely attacks to occur[8] and considering the time scope with the client, the group have only prioritized securing the most vulnerable parts of the web application.

ID	QR 0
Attribute	Security
Description	SQL injection should not be possible .
Reason	Users could in the worst case delete the result of the study, willingly or by using names/ descriptions containing pseudo code.

Table 4.3: QR0

ID	QR 1
Attribute	Security
Description	The web application should not be penetrable by brute forcing the password.
Reason	Mitigates the risk of an attacker trying to get access to the system by running a brute forcing script.

Table 4.4: QR1

4.4.2 Modifiability

As a result of agile development with requirements changing often, the systems architecture naturally is required to have a certain degree of modifiability. This is also important if the system is going to be built on or changed in the future e.g by another bachelors group. The client desired high modifiability in the implementation of the sonification algorithm as this is likely to be changed in the future.

ID	QR 2
Attribute	Modifiability
Description	The whole sonifaction algorithm should be replaceable within 2 hours.
Reason	The client want to replace the algorithm after the system is delivered since it is developed and changed constantly. If big changes are needed, changes needs to be done in the source code and the app needs to be redeployed.

Table 4.5: QR2

ID	QR 3
Attribute	Modifiability
Description	The view of a page on the web site should be replaced within 2 hours.
Reason	The client may want to change the looks of the web interface in the future.

Table 4.6: QR3

ID	QR 4
Attribute	Modifiability
Description	It should be possible to migrate the web application to another server within 5 hours.
Reason	The client may need to migrate the web application to a cheaper hosting service or a local server in the future.

Table 4.7: QR4

4.4.3 Performance

This is more important for the mobile application than the web application. A VR-application requires a lot of resources, especially in combination with server communication and running a sophisticated sonification algorithm.

ID	QR 5
Attribute	Performance
Description	The time between the color input and the coherent simulation response should be less than 50ms.
Reason	It is important that when the user looks around, the system follows instantly. In the Colorophone VR application the user is ”blind”, which makes it even more crucial that there is no latency, as they will not be able to see the delay.

Table 4.8: QR5

ID	QR 6
Attribute	Performance
Description	The ”exploration phase” should have a frame rate of more than 10 fps in admin mode
Reason	For the researchers to properly check for errors in the 3D-model a minimum frame rate is required.

Table 4.9: QR6

4.4.4 Maintainability

As the system is likely to have little or no maintainance after delivery, but might be changed completely in the future, more focus has been put on modifyability rather than maintenance. However mechanisms for the researchers to see if the experiment they have uploaded works as planned needs to be present so that they can maintain scheduled creation and deployment of experiments daily.

ID	QR 7
Attribute	Maintainability
Description	It needs to be possible to replace the core sonification algorithm within 5 minutes.
Reason	The client want to replace the algorithm after the system is delivered since it is developed and changed constantly. This means changing algorithm-parameters in runtime unlike changing the whole algorithm as descibed in 4.5

Table 4.10: QR7

ID	QR 8
Attribute	Maintainability
Description	Researchers need to verify experiments within 10 minutes.
Reason	Researchers will have to add more than 90 experiments before and during the period of the study. They need to verify if the experiment work as accepted to make the appropriate changes if needed.

Table 4.11: QR8

4.4.5 Availability

It is very important that the application is available most of the day in the project period. This is most important in the end of the project period when the system will have many experiments and users. If the system is not available then, it could compromise the whole study.

ID	QR 9
Attribute	Availability
Description	The system should have up time of at least 99 percent during day time in the study period.
Reason	For conducting the experiments it is vital that the system is available at the convenience of the study volunteers during day time.

Table 4.12: QR9

ID	QR 10
Attribute	Availability
Description	The system should be able to manage up to 300 experiments with up to 35 users
Reason	It is important that the system could manage the strain of many experiments and users in the end of the study period. The researchers are going to create 1-3 experiments for each day over a 3 month period. The number of users will be at most 35 including the study volunteers and the researchers.

Table 4.13: QR10

4.4.6 Usability

The system is going to be used by both study volunteers and researchers. Usability is more important for the study volunteers because they have might have less technical knowledge than the researchers. Having high usability will save time on instructing the volunteers and decrease errors.

ID	QR 11
Attribute	Usability
Description	A user should be able to conduct an experiment on the first try without instructions.
Reason	The experiment should be easy to conduct for an average study volunteer so that time spent on instructing and error handling is minimized.

Table 4.14: QR11

4.5 Use cases

This section describes use cases for the main features of the system. Use case diagrams for whole system is found in chapter 5 System Design in figure 5.10 and figure 5.11 .

ID	UC1
Title	Add and edit experiment
Goal	Add new experiment to the system and edit it.
Actors	Researcher
Precondition	Successful login to website
Main Flow	1. User creates the 3D-Model according to instruction given in the user manual (Appendix B). 2. User navigates to the ”Add experiment” page. 3. User uploads the 3D-model created and fills in the name, description, date, time limit, questions and alternatives fields. 4. User clicks add experiment. The model gets uploaded to the file server and the experiment info gets saved into the database. 4. User navigates to edit experiment in the menu. 5. All the experiments are listed with ”edit” and ”delete” button. 6. Experiments are edited by clicking edit, the same form as ”add experiment” are prefilled with parameters loaded from the database, and changes are saved when ”Save changes” are clicked.

	7. By clicking ”Delete” the experiment are removed from the page and deleted from the database with all coherent question, alternatives and answers.

Table 4.15: UC1

ID	UC2
Title	New user
Goal	Add new user/ phone to the system.
Actors	Researcher and Customer
Precondition	Successful login to website
Main Flow	1. User launches the application on chosen phone. 2. User locates and notes the auto generated and unique Phone-id in the main menu of the screen. 3. User logs in the the web application and navigates to admin in the menu. 3. User scrolls down to application and accepts the new Phone-id. 4. User scrolls up and find the Phone-Id in the list of registered users. 5. Alias is changed from the Phone-id to e. g the name of the owner of the phone. Admin is toggled on if the phone is going to be used by a researcher.

Table 4.16: UC2

ID	UC3
Title	Launch experiment
Goal	Launch an experiment in the mobile application.
Actors	Researchers and Study volunteer
Precondition	UC1, UC2
Main Flow	1. User launches application. 2. Application verifies phone id with the database, and checks what privileges it should have. 3. Application presents user with different experiments, based on their privileges type. 3. User chooses an experiment from the menu. 4. The app downloads the 3D-model and experiment data from the server. 5. User mounts phone into VR headset according to tutorial on screen. 6. Application detects VR headset and starts VR. 7. User equips the headset and enters VR.

Table 4.17: UC3

ID	UC4
Title	Conducting experiment
Goal	Study volunteer conducts an experiment.
Actors	Study volunteer
Precondition	UC3
Main Flow	1. User finds the ”I’m ready” button in the virtual space. 2. User reads the task description clicks ”I’m ready” when finished. 3. The 3D-model loads, visual input gets disabled and the Sonification algorithm gets enabled. 4. User explore the 3D-model in VR using only auditory input for the time period loaded from the database. 5. Questions and alternatives gets loaded in a virtual form. 6. User answers the questions by focusing a pointer on the chosen alternative and the answers are sent to the server. 7. A confirmation is shown on the screen and user is navigated back to the main menu.

Table 4.18: UC4

ID	UC5
Title	Verify experiment
Goal	See if the experiment created works according to expectations.
Actors	Researchers
Precondition	Successful login to website, UC3
Main Flow	1. User go to the web application and verify that admin is toggled on for the Phone-id used. 2. User restarts the mobile app and clicks select experiments 3. The Phone-id is sent and verified as admin by the server and all the experiments are shown. 4. User chose wanted experiment and explores it in VR with visual input as well as auditory input.

Table 4.19: UC5

ID	UC 6
Title	Analyze results from experiment
Goal	View answer data provided by users conducting experiments.
Actors	Researcher
Precondition	Successful login to website
Main Flow	1. Researcher logs in to the web application and navigates to the ”Statistics” page.
User selects ”Sort by experiment”:	2. A list of all experiments is loaded from the database shown with name, description and id to identify them. Desired experiment are clicked on by researcher. 3. Researcher first get an overview of questions and alternatives in the experiment by looking at the first table. The user can later see which answers was given sorted by users in the new table.
User selects ”Sort by users”:	2. A list of all users is loaded from the database and shown with alias, phone-id, number of experiments conduced and date of last experiment to identify them. Desired user are clicked on by the researcher. 3. A list of experiments conducted by the user is loaded from the database with percentage of correct answers and shown in a table. Desired experiment is selected. 4. Questions and answers given is loaded from the database and shown in a table.

Table 4.20: UC6

ID	UC7
Title	Edit the sonification algorithm.
Goal	Edit the core algorithm or parameters for the auditory input given when conducting experiments in the mobile application.
Actors	Researcher
Precondition	Successful login to website, UC5
Main Flow	1. Researcher logs in to the web application and navigates to the ”Sonification settings” page. 2. Researcher edits parameters or core algorithm and clicks save changes. 3. Researcher opens the mobile application according to UC5 and verifies the changes.

Table 4.21: UC7

System Design

The system architecture is designed to meet the requirements explained in chapter 4. Section 5.1 describes architectural drivers in the web- and mobile application. Section 5.2 explains the architecture from different viewpoints. Section 5.3, 5.4, 5.5 and 5.6 describes architectural styles, patterns and tactics used to fulfill the quality requirements.

5.1 Architectural drivers

This section briefly explains major architectural drivers and restrictions that impacted the system design.

5.1.1 Functional Requirements

Many of the functional requirements have profound impact on the architecture. Especially important was functional requirement F18; *The mobile app needs to get the 3D-model from the file server and load it correctly* [4.2]. This requirement means that experiments created by the researchers should be able to be loaded into the mobile application during runtime. It is the main reason for creating the web application consisting of the website, file upload, download system and database. The 3D-model used in the experiment needed to be downloaded from an external source together with experiment information and parameters for the sonification algorithm after the mobile app was deployed.

5.1.2 Defining Technologies

Some of the technologies chosen in the Prestudy phase (Chapter 2) impacted the design of the system. The technologies had restrictions that made it impossible to use some solutions and innate structure and conventions that invited the use of others.

LAMP Stack

LAMP was chosen due to it being a well documented and lightweight stack. It being lightweight means that few limitations were set for file structure, code conventions, object-oriented/procedural approach and framework-specific syntax as opposed to ReactJS where these things are restricted. Not having a framework means fewer limitations but also that the group had to find and use tactics and pattern to respond to quality requirements, instead of having these solutions incorporated in the framework.

Android Development Platform

The mobile application was required to work on an Android device. In order to fulfill this requirement the system makes use of Google's own assets. The application is built on top of the Android architecture stack, and is limited by the restrictions imposed by the Android SDK.

Unity

To develop the mobile application, Unity is used in combination with the android SDK. Unity comes with limited functionality for making 2D parts of the application like a menu for selecting experiments and options menu for changing settings. However, it is rich in functionality in developing 3D-applications, forcing the group to implement most of the user interactions as the answering of questions in a virtual environment.

5.2 Views

All of the stakeholders have different interests in the system, and various levels technical experience. The following section describes the system using the 4+1 architectural view model [2, *Advanced software architecture - Documenting in UML*]. The viewpoints are representations of the architecture that are meaningful and understandable for one or more stakeholders. For each view, table 5.1 contains a brief description of its purpose and a list of the target stakeholders. The views are often known by slightly different names, so the table lists the two most common for each.

View	**Purpose**	**Target stakeholders**
Structure, Logical	Describes the functionality of the system	Customer, project team, course staff
Behaviour, Process	Describe the dynamic aspects of the system	Researchers, customer, project team, course staff
Deployment, Physical	Describe the topology of software components and the physical connections between them.	Customer, project team, course staff
Development, Implementation	Describe how the code is organized during development	Project team, course staff
Use case, Scenario	Describe interactions between actors and the system itself	Researchers, customer, project team, course staff

Table 5.1: Architectural Views

5.2.1 Structure view

This section describes the system from the structure viewpoint.

Web application

The structure of the web application is relatively simple. The website contains five main components, which can be seen in figure 5.1. The *experiment administration* component handles adding, editing and deleting experiments. Statistics about user-generated content are displayed and handled by the *statistics* component. The *admin* component controls which smart phones should be able to participate in the experiments, and which of the participating phones should have access to admin mode. The *sonification settings* component lets the user edit certain parameters in the sonification algorithm. The *authentication* component handles authentication, such as login and logout. All the website components needs access to the database, while the experiment administration component also needs access to the file server to be able to upload the 3D-models.

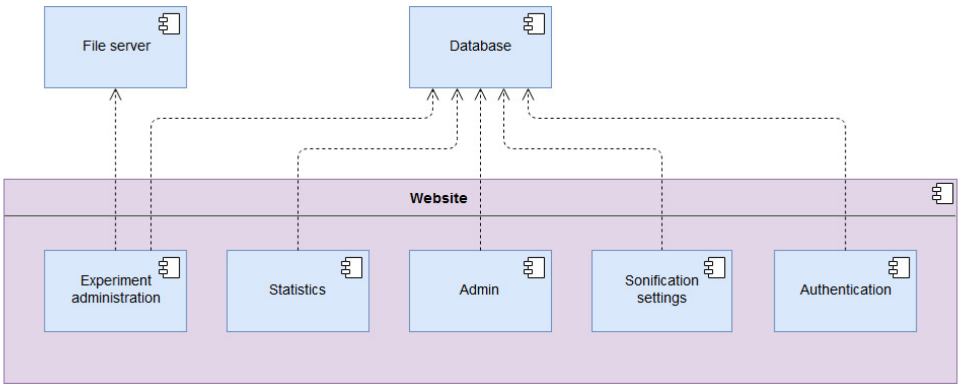


Figure 5.1: Component diagram of the web application

As explained in the upcoming section 5.4.1 Model-View-Controller, the web application uses the MVC pattern. The file server and database makes up the model, while each of the website components in figure 5.1 are split into controllers and views (see figure 5.12).

Mobile application

The most important components of the mobile application are shown in figure 5.2. They are split into player components and experiment components. The main experiment component is the experiment. An experiment consists of a 3D-model and associated questions. The 3D-model is downloaded from a file server, and the questions are fetched from the database. To get a working experiment, there also has to be a player component. The main player component is the camera. The camera acts as the eyes of the user. Normally the camera is stationary, but with the help of GoogleVR, a technology that provides VR support for most smartphones, the user can control the camera by moving their head around.

As the point of the experiment is to explore the virtual room by hearing, not vision, the blindfold and sonification needs to be added. The blindfold's purpose is the same as it is in the real world; to disable the wearer's vision. The sonification is what converts the visual input into sounds. See section 2.2 for information about how the sonification algorithms work.

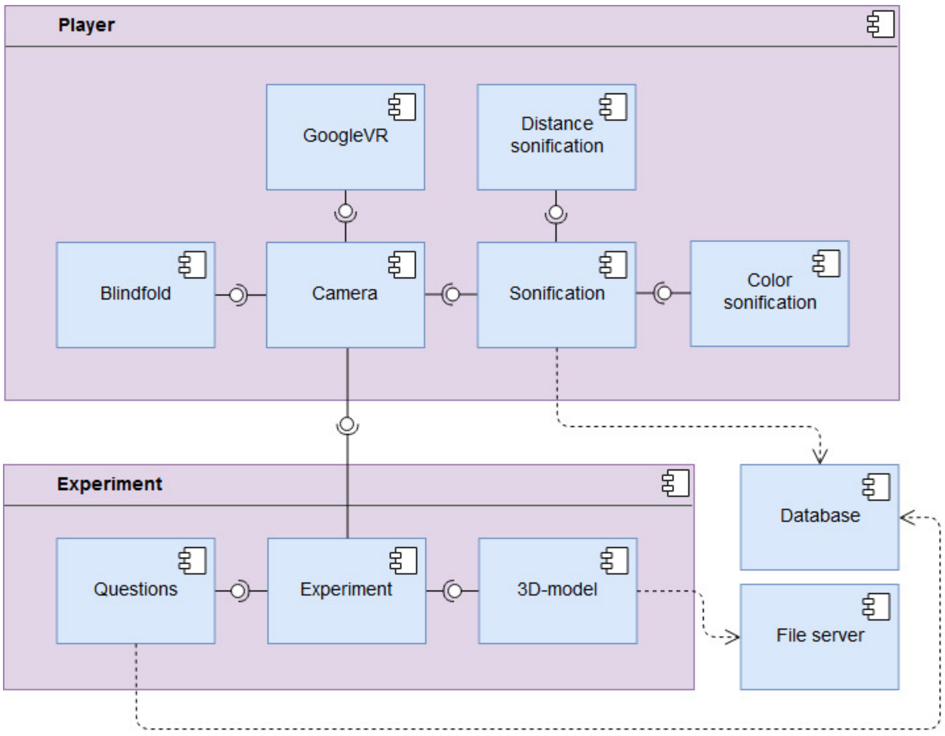


Figure 5.2: Component diagram of the mobile application

The blindfold can be toggled on or off by an admin user, and the experiments available automatically changes based on the current date. The sonification algorithms changes based on values from the database, which means that most parameters can be changed from the web application.

Database

The structure of the database is illustrated in the ER model in figure 5.3. ER stands for entity-relationship, and as the name suggests, it describes the relationship between different entities. In this case, the entities are in the form of tables in the database. Together, the *experiment*, *question* and *alternative* tables contain all the information needed for each experiment, except for the actual 3D model. Each question contains an experiment ID, which means that a question can only be connected to a single experiment, while an experiment can have several questions. The same relationship is between alternatives and questions.

Only one of the alternatives connected to a question is the correct answer, which is denoted by the *correct* field in alternative. The *sonification* table contains parameters used in the sonification algorithm. Considering the name, the *application* table might be confusing at first. In this context, an application refers to a request, not a piece of software. This table is automatically filled with new entries every time a new phone uses the mobile app. The users of these phones will be called *appliers* from here on. The *phone_id* field is a unique identifier generated by the phone itself. The applications has to be manually accepted or rejected by the researchers on the website. Appliers can not do anything in the mobile app, as long as their application has not been accepted yet. The purpose of this is to give the researchers more control of who can participate in the experiments. If an application is accepted, the phone is moved over to the *user* table. A recently accepted user has no admin privileges and alias set to the same as the *phone_id* by default. The only purpose of the alias is for the researchers to more easily be able to identify the phones. Both admin privileges and alias can be changed on the website. The *answerdata* table contains all answers given by the users in the mobile app, as well as the time it took them to answer.

ER-modelling is used to create a minimalistic and effective database. This is done to decrease overhead leading to an increase in performance in the mobile application. Attributes are relationally connected to each other making sure that if an experiment gets deleted, the connected questions and alternatives also gets deleted. This decreases the risk of errors occurring connected to deletion and editing of experiment during the study, which increases maintainability.

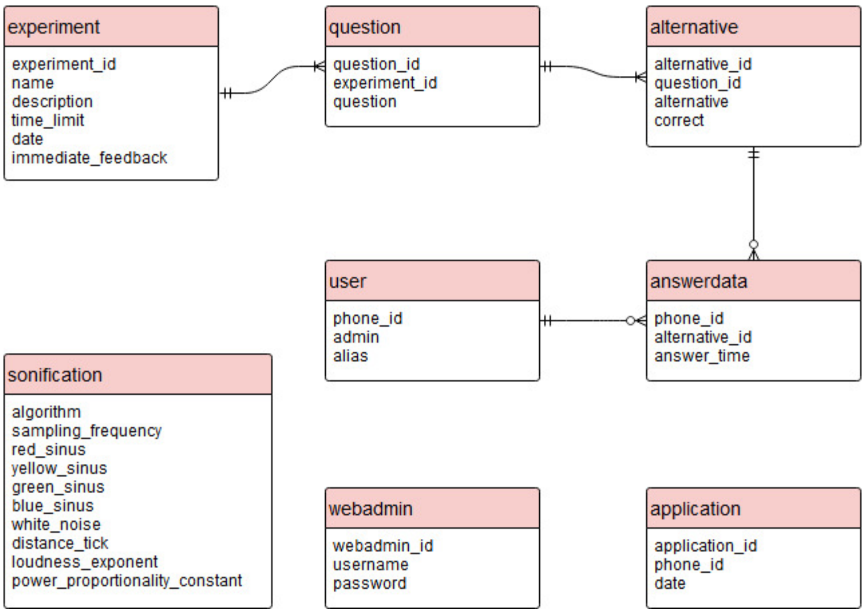


Figure 5.3: ER model of the database

5.2.2 Behaviour view

This section describes the behaviour of the system.

Web application

The functionality of the web application is separated into different pages. These pages and how to navigate between them is shown on the sitemap in figure 5.4.

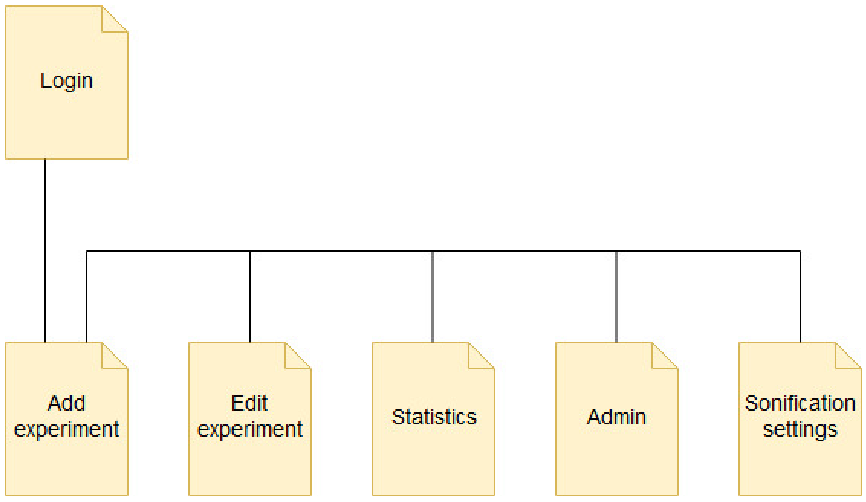


Figure 5.4: Sitemap of the webpage

The process of logging in and adding an experiment is illustrated in figure 5.5. All the different use cases for the web application (see use case diagram [5.10]) have fairly similar processes, so this figure should give an overview of how the system works. The figure shows a happy path, which means it is a default scenario where no exceptional or error conditions arise.

Before getting access to the main functionality of the web application, the user has to log in. The login credentials provided by the user are verified by a login controller. The controller finds the username in the database and checks if the passwords matches. If both of these steps succeeds, the user is redirected to the add experiment page. At this point the user has access to all the functionality. In the add experiment page, the user can create new experiments by uploading a 3D-model and filling the form with experiment information. When the user clicks the submit button, the add experiment controller uploads the file to the file server, inserts all information to the database, and presents the user with a status message. The status message informs the user if the operation was successful or not.

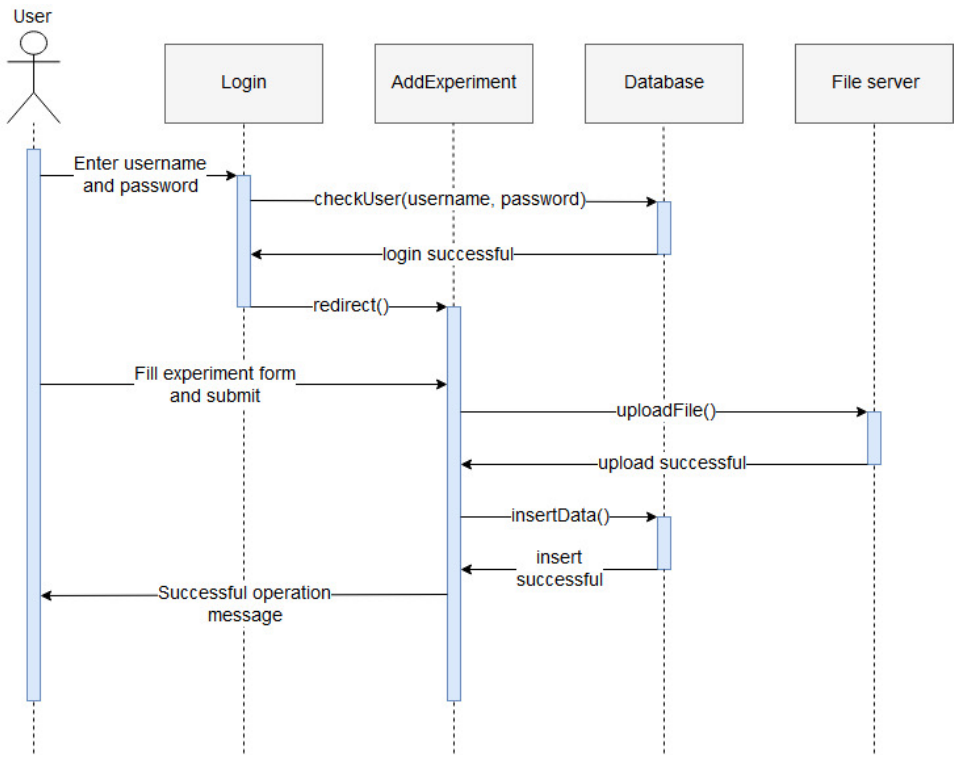


Figure 5.5: Sequence diagram of logging in and adding an experiment

Mobile application

The behaviour of the mobile application is illustrated in figure 5.7. The application starts off with VR disabled. The users are presented with a menu where they can choose what experiment they want to start. The list of experiments the users can see vary depending on the date and the phone's user type. *Admins* can see all the experiments set for any date. *Normal user accounts*, which the study volunteers should have, can only see experiments that are available for the current date. *Appliers*, which are phones yet to be accepted into the experiments, cannot see or start any experiments. When an experiment is chosen, instructions on how to mount the phone into the VR headset is shown. When the phone is placed inside of the VR headset, the user can no longer tap the screen. Therefore, to continue past the instructions the user has to hold the phone in a horizontal position for ten seconds. This is both to give the user an alternative way of interacting with the phone, and to prevent them from accidentally skipping the instructions. When the headset is mounted and the user has skipped past the headset instructions, the app goes into VR mode and displays instructions for the chosen experiment. In VR mode, gaze click is the only way to interact with buttons in the app (see figure 5.6). A gaze click is a timed click function attached to the reticle, which means that the user can 'click' on an interactive object by focusing the reticle on it for a short amount of time. The outer circle in the image shows an

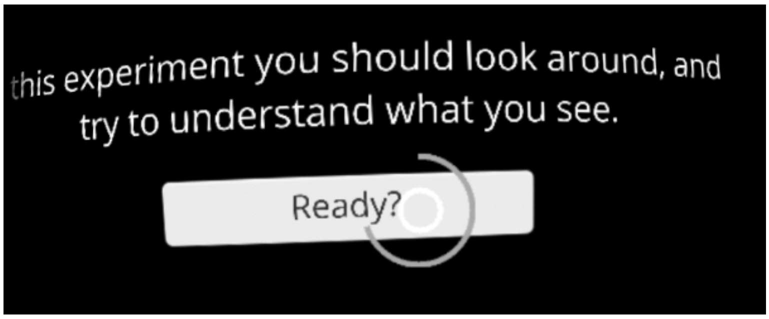

this experiment you should look around, and
try to understand what you see.
Ready?

Figure 5.6: Reticle with gaze click

animated timer of the gaze click, while the inner circle is the reticle. The reticle is always placed in the middle of the screen, and expands whenever the user looks at an interactive object.

To get past the experiment instructions and into the exploration phase of the experiment, the user has to gaze click a ready button. The exploration task is where the user should look around the virtual room and try to orient themselves through sound. After a set amount of time the exploration task ends, and the user is presented with multiple-choice questions about the virtual room. When the user has answered all the questions, the application displays a message thanking the user for their contribution, and asking them to take off the headset. After a few seconds the application disables VR and returns to the main menu.

If an error happens during an experiment, an error message appears asking the user to restart the current experiment. The user can quit the app at any time by using the Android home button.

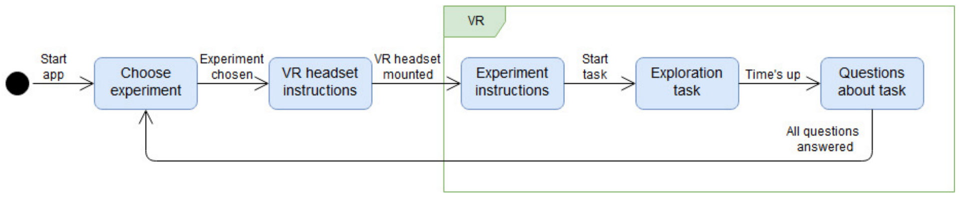


Figure 5.7: Activity diagram of the mobile application

5.2.3 Deployment view

The deployment view describes the physical connections between software components. The topology of the Colorophone VR project is illustrated in the deployment diagram [figure 5.8]. The web application consists of a website hosted on a webserver, a database and a file server. The web application is deployed on a cloud server called Amazon Web Services, or AWS for short. The mobile application is developed for Android, and should only be available to the customer, researchers and whoever they share it with. The app downloads experiment information and 3D-models through http GET requests to the webserver. The webserver handles the requests and presents the correct data to the mobile application.

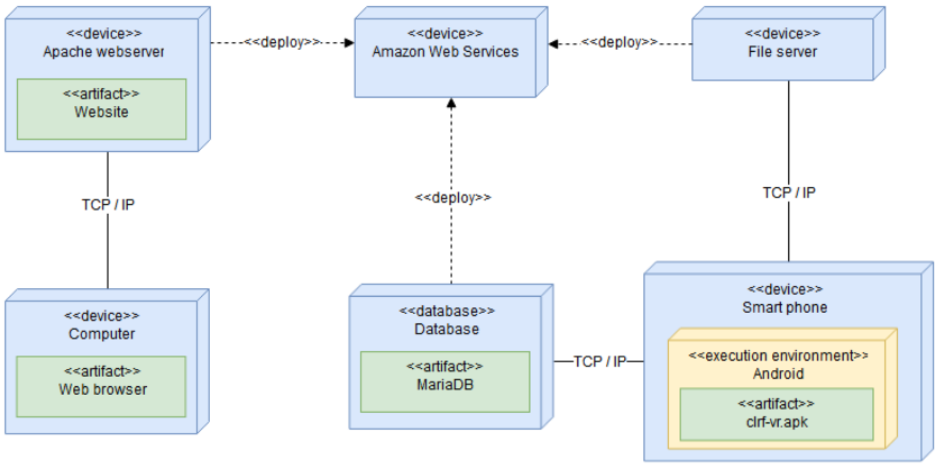


Figure 5.8: Deployment diagram of the whole system

5.2.4 Development view

Figure 5.9 displays a very simple package diagram. It is split into storage, mobile and web. See 5.2.1 Structure view for more information about what each of these packages contain.

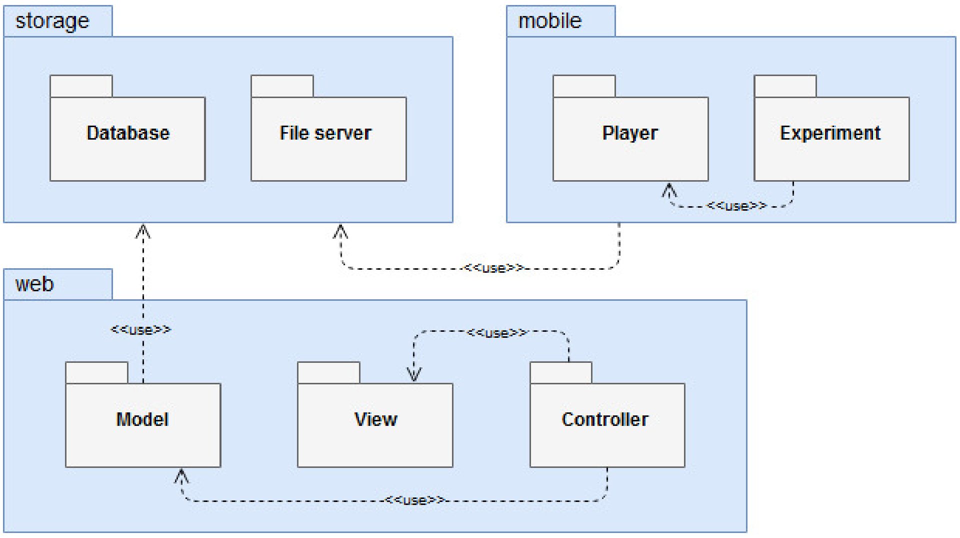


Figure 5.9: Package diagram of the whole system

5.2.5 Use case view

This section describes the system from the use case viewpoint.

Web application

Figure 5.10 illustrates all menus and functionality the website contains. The leftmost ellipses acts as categories, or menus on the web page, while the rightmost ellipses are the actual functionality. To prevent the diagram from being too extensive and cluttered, some functionality are grouped together. *User management* is where the researchers can toggle user type and set aliases for all the phones participating in the experiments. *Application management* is where the researchers can accept new phones into the experiments, or reject them from participating. All the functionality in the *experiment management* are labeled 'management', which means the elements can be added, edited or deleted. The *sound settings* and *algorithm settings* is where the parameters for algorithms can be changed. In *view statistics*, the researchers can view all the collected data and sort by various categories.

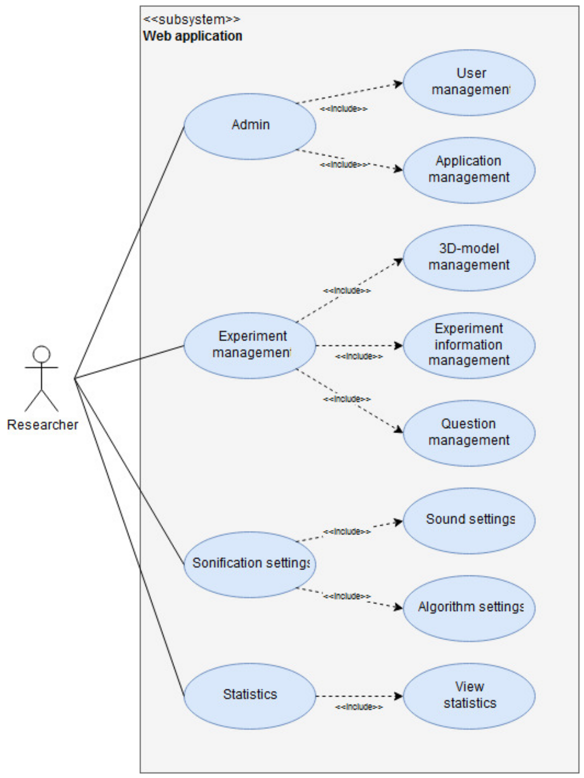


Figure 5.10: UML use case diagram for website functionality

Mobile application

Figure 5.11 illustrates the functionality of the mobile application. As mentioned earlier in this section, there are three different types of users; researchers with access to admin accounts, study volunteers with normal user accounts, and appliers which is anyone who has launched the application on their phone but has not been accepted as a user yet. The appliers do not have access to any functionality in the mobile application, so they are excluded from the diagram. There are a few differences between admin accounts and normal user accounts. Although both user types can browse and start experiments, the amount of available experiments differ. The volunteers can only see the experiments available for the current date, while the researchers can see all created experiments. In addition to this, the researchers can also explore the 3D-models without the blindfold.

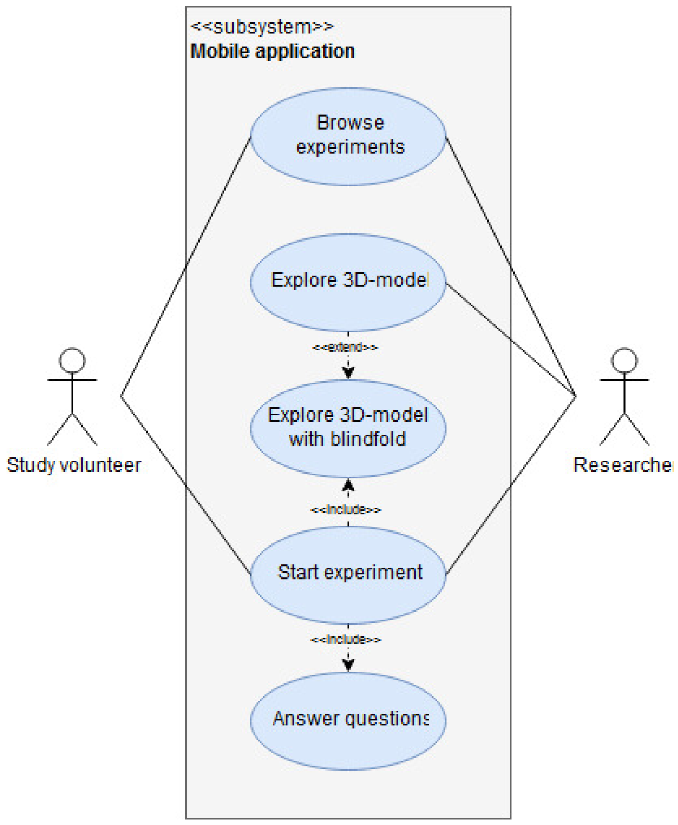


Figure 5.11: UML use case diagram for mobile application functionality

5.3 Architectural styles

Architectural styles explain application design at the highest level of abstraction.

5.3.1 Client-server

This style is used in both communication between the web site and server as well as between the mobile application and the server. In the first case the server is fetching the web page as well as hosting the database and communicating with the database. In the latter case it is serving database information and model files while saving statistical information (See the deployment diagram in section 5.2.3 Deployment view). A server is in this context mainly needed to ensure availability in contrast to a peer to peer solution that only would ensure availability when enough users are connected. The server is hosted on a cloud platform that guarantees almost constant availability.

5.4 Architectural Patterns

Architectural patterns are a ways to implement architectural styles.

5.4.1 Model-View-Controller

MVC is used on the website. Because of the use of LAMP stack without external frameworks, the web application needed a modular architecture to increase modifiability. Model-view-controller is used to gather logic for the three parts respectively in the models, views and controller folders of the web site. Logic for the connection to the database is saved in Models, logic for adding, editing, deleting and fetching information from the database in addition to functional scripts is saved in Controllers and the HTML for the web views is saved in Views. This makes it easier to develop different parts of the system separately by different people, and to understand where to find the desired logic if changes are to be made. It also makes it easier to change for example the view in the future without impacting the controller logic to much. Figure 5.12 shows the use of MVC in the Web application.

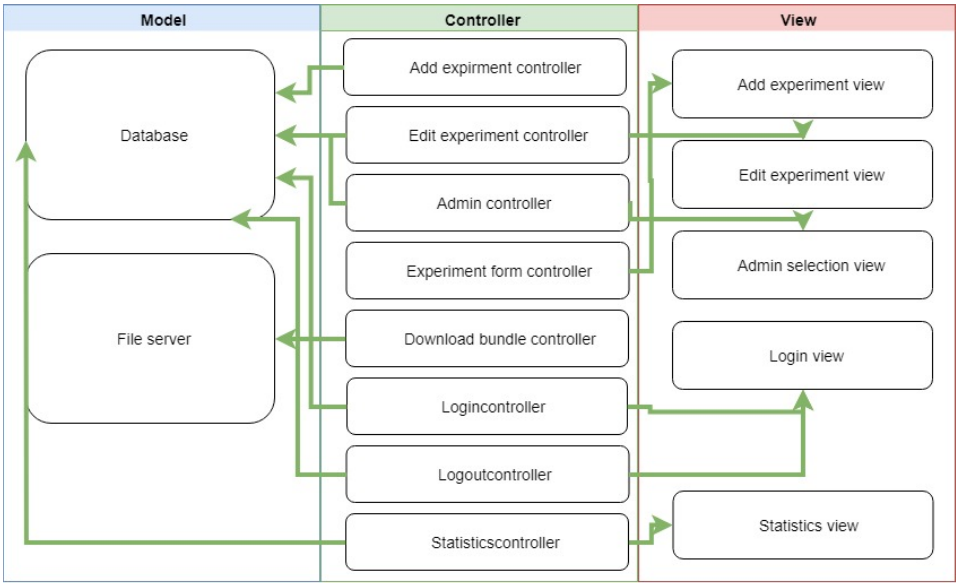


Figure 5.12: MVC view of the web application

5.5 Design patterns

Design patterns are meant to solve localised problems.

5.5.1 Exception handling

Exception handling is used in the web application to ensure that the system continues to run in case of common errors. The errors include the upload of wrong file format for the model, wrong experiment input format given by the user and wrong configuration when creating the models. In many cases an error message will be given to the researcher so solutions could be made by them. Another important way to handle exceptions is by have an ”Admin mode”. Here the researcher can verify or see what was wrong with the experiment by testing them in the application without the ”blindfold”. This enables the researchers to maintain the application on their own in addition to increasing availability.

5.5.2 Pipe-and-filter

Pipe-and-filter is here used as a design pattern rather than an architectural style as it is used only to solve a problem in a localized area. It is used in the Sonification algorithm in the mobile application. Color information within the visual rectangle of the user is given as a byte stream. It is later processed in filters including calculation of average, the sonification algorithm, intensity transformation, amplitude adjustments. The first filter consists of calculating the average RGBYW-values (red, green, blue, yellow, white) of the colors given by the byte stream. It is then converted in to new RGBYW-values using the customers algorithm. The third filter transforms the color intensities according to Stevens power law. In the last filter, the volume of the sound gets adjusted according to the intensities given in the previous step. Using this pattern makes enables the client to change only the filter containing logic for the sonification algorithm without affecting the others filters present.

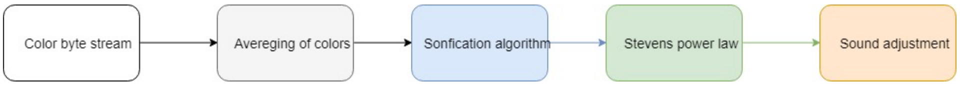


Figure 5.13: Pipe-and-filter view of the sonification algorithm

5.6 Tactics

Tactics are used to fill in or heighten focus on selected concerns in the architecture.

5.6.1 High cohesion, Low coupling

This tactic addresses the concern of having an interconnected system, making it difficult to change or replace parts of the system. Considerations have therefore been taken to structure logic for components likely to be changed so that it is easy to change those components. In the web application this is done by moving all logic for connecting to the database in a separate file. In case of the database changing, only this document would need to change. This tactic is especially effective in combination with the MVC pattern, because the pattern naturally parts the system into areas that are likely to change while the tactic makes sure the areas are loosely coupled.

5.6.2 Hashing

A simple hashing algorithm is implemented in the web application to have a minimal level of security. This is a high value trade off between time to implement and effect on security matching the risk assessment of the quality attribute.

Testing

This chapter describes how testing has been conducted for the duration of the project. Testing is a vital part of software development as it ensures that the system is working correctly and that it does what it is supposed to do as well as guaranteeing code of high quality which does not contain security risks or bloat.

It is important to have a testing strategy when working on a larger software project, and for this project it has been applied to most levels, with varying degree of application, based on how important it was found by us and the customer.

6.1 Manual Inspection and Code Review

Code reviews were the most consistently used method to keep code quality high, on the lowest level. The team performed periodic manual code reviews to ensure code was written according to specification and following best practices. Code reviews from other team members also helped the team understand and increase ownership of the code written, and facilitated knowledge sharing for the entire team. Such practises also support the team in understanding their own progress and future estimation.

6.2 Unit Test

Unit tests can be used both as a time-saving measurement and as a quality measure for a larger software development project. Due to the unfriendliness of unit testing in Unity and the structure of the project, unit tests were considered unnecessary and too time consuming for too little results. Unit tests could have saved time had the project been utilizing a more advanced form of continuous integration, but that was not suitable for Unity with its many temporary files and general volatility before being built.

Tests could have been written for the C# scripts used in the application, but the team decided to focus more on manual inspection to cover this level of testing as it made the process more agile and fitting our development practices.

6.2.1 Web application

The use of unit tests in the web application could save time debugging when building an application in PHP. It could also make the application have less security issues, increase performance, ensure quality and increase modifiability.

However, learning a new test-framework uses a lot of time resources. The increase in performance is deemed redundant in this project as it will not scale very much and the need for modifiability is met by the design. In addition to that the value it gives to security is prioritized low in our system. Based on this the team decided to prioritize time usage on processes with higher priority. Quality assurance and bug fixing for the web application were addressed by user tests, acceptance tests and code review.

6.3 Integration & system test

Independent modules of the system were tested by themselves before integrated with each other. Modules such as the VR view and 3D room loading module are given inputs and expected results, and in the case of a deviation the specified module was corrected.

After all modules have gone through integration testing they were integrated with each other and the entire system ran through system tests to guarantee the finalized system was satisfactory according to specifications.

6.3.1 Functional Test

Functional testing is a black-box testing method where functional pieces of the system are tested with certain different inputs to ensure the integrity of algorithms and functions as well as maintaining consistency.

VR Application

The consistency of the output of the many algorithms used in the VR application were incredibly important. Functional testing was therefor used here to test the different sonification algorithms, including the distance sonification.

Web Application

For the sake of making everything reliable and safe for the researchers to use it was necessary to make sure the web application was thoroughly manually tested. Different possible cases of inputs were tested to ensure this. The test was done most thoroughly in the ”Add Experiments” page, where there are many possible inputs, all which need to follow certain criteria.

6.4 Performance Test

The application was restricted by the phones used for the final research project, so having the same models available to us was crucial in understanding how well the application would be performing for the end users. At the same time, it was made possible to loosen up several restrictions due to the fact that all end users would be using the same device. It was not necessary for us to make sure the application would run smoothly on higher or lower resolutions, or with stronger or weaker hardware.

Performance testing is defined as how responsive and stable a system is under certain workloads. Certain key moments in the system's lifetime were tested and analyzed. These tests early discovered possible performance issues and this made the team take measures to meet minimums

6.4.1 Load testing

Load testing is done by putting a large workload on an application and evaluating its performance.[11] This gives valuable metrics on how well the application will be able to sustain under heavy pressure. For this application an effective way of load testing was done by downloading many 3D-models in the app and loading several of them repeatedly. During the load slight delays and hiccups were experienced, but these occurred during a phase where responsitivity was not an important factor.

6.4.2 Soak testing

Soak testing is done by putting a constant workload on an application over longer period.[5] The workload should simulates a typical workload for the system over time. The aim is to evaluate performance, and to test the deterioration of performance and stability of the system. The VR application was run through soak tests which consisted of continuous use of the VR application with smaller or no breaks in between sessions of VR exploration.

6.5 Interruption Test

Interruption testing is important for any mobile application, but in this project's case it was reasonable to rule out some of the common interruptions due to the application only being used in an isolated test environment.

Examples of test cases that could safely be ignored include: Interruptions by phone calls or SMS, interruptions by other applications, interruptions coming from memory being full.

In addition to these the customer explicitly asked to ignore some other potential interruptions as the end users would be able to mitigate these before they would become issues.

Examples of these interruptions include: Interruptions due to low or empty battery, interruptions from connecting devices and cables.

In reality, the only potential interruptions that could be expected to happen would either be outside interruptions by a third party or network disconnects. Network disconnects would not be a huge problem, as a stable network connection is only necessary during two phases of the experiment, when downloading new experiments and when sending results of answered questions, and it can be considered more than reasonable that the end users will be able to have an active connection for that duration. If, by any chance, an issue would happen due to network errors, the end user will be alerted.

6.6 Installation test

It was incredibly important to ensure everything was ”dummy-proof”. The development team would have no control or direct communication with the end users after the project's conclusion. Therefore it was important not only that the installation phase would finish without issues but also that our customers in Poland would have sufficient knowledge to be able to re-install the application in case it's needed.

6.7 User Test

The system is user tested in three phases. The first phase is testing by the immediate client, the second is by the researchers using the system and the third is by random users. This is done to pick up errors and to discover inconsistencies with expectations of the system across different stakeholders. Compared to other methods of testing the system, this is a relatively little time consuming tactic and is used because it is a high value trade off between time spent testing and errors discovered/ acceptance addressed. Spending time to let different user groups test also makes us more likely to uncover a broader range of errors.

6.8 Acceptance Test

The scope of the project is defined as a set of user stories with acceptance criteria. After testing the system, the customer have given his review of the system and whether it complies with the mutually agreed upon acceptance criteria.

6.9 Results

This section contains a simplified version of results derived from testing, along with key issues detected, how they were detected and how some were subsequently fixed.

A longer and more comprehensive list of all issues found through testing can be found in Appendix E where the raw results of several user tests have been accumulated.

- Occasional scene drift while not moving the phone.
- If “Loudness Exponent” is not equal 1 sounds get very unpleasant. Something is wrong with scaling there.
- The text showing which alternatives are correct and incorrect should be as intuitive as possible and without colors.

User testing revealed to us that when conducting an experiment the sonification area would not be in the center of visual field. This was due to the calculation of center using the screen size in vertical position, which did not change when having the phone in horizontal position.

Interestingly, an error in the sonification algorithm had caused blue and green colors to correlate to the same audio frequency. This was something that was not noticed by the developers, and only the researchers who were familiar with the sonification and its behavior were able to figure this out through user testing.

By performing soak testing of the VR app a performance issue was uncovered where performance would degrade heavily after a while. It became apparent the application needed optimization when running for a long duration.

Evaluation and Conclusion

This chapter contains the final evaluation of the project, and broadly outlines the overall learning outcome regarding teamwork and process from a project of this scale.

7.1 Group cooperation

When the project started, the group members were not acquainted. This was a challenge, but might also have proven to be positive for the overall group dynamic. It was a challenge because it could make it harder to stay open and honest, and positive because it could create a more professional relationship between the team members. The team addressed the challenge by talking about the different team members expectations for the project. This way the whole team could agree on an end goal together. Having people take initiative in meetings and during work hours is critical, but might be hindered by lack of familiarity with other team members. When examining notes from the retrospective meetings(see Appendix C), a general lack of initiative and communication is something that stood out. These concerns were addressed by increasing the frequency of meetings which led to less conflicts, a better work environment and increased motivation for working on the project.

7.2 Customer relationship

The group had weekly meetings with the customer during the project. In addition, day to day communication on Slack was important for clarifying misconceptions in requirements, communicating progress and communicating decisions from the researchers in Krakow, or for sharing documents and asking questions. Throughout development the customer was available, and always made sure that the whole group understood the different concepts behind the Colorophone project. The customer also made sure the team had access to the required hardware. All in all, the contact between the group and the customer was a cooperative effort rather than a service oriented relationship.

Deciding on final requirements was done relatively late in the project because neither team nor client had experience with VR technology. This process was very much a creative one, and within the group, it also acted as a way for the group members to gain ownership of the product. This uncertainty however, also proved a challenge in regards to planning and project management. At several stages in development, much of the group's resources was engaged in exploring alternative solutions for different requirements, and this could initially have been seen as a waste of time. However, it ensured that the problem's domain was fully explored, and in an ideal world, this would mean that the solution is the most optimal. This extensive exploration, with the motivation of creative freedom and the feeling of ownership it entailed is probably one of the reasons why the product came out as it did.

7.3 Development methodology

The need for using a specific development methodology was initially difficult to comprehend. However, after a short period where these practices were neglected it became apparent that they were necessary. The main reason for the neglection was that the team had little experience working on a project satisfying the needs of many different stakeholders. The stakeholders included other team members, a client, a supervisor, course staff and end users. They contributed with different needs related to quality assurance, cooperation, documentation, communication and time management.

The ideals of Lean Software Development combined with chosen Scrum practices (see Chapter 3) worked very well when applied to this type of project, especially given its somewhat erratic nature and the team's limited experience in software development. Following Scrum exactly not only requires experience, but also a lot of time for the activities. An example is the estimation given to user stories, which in our case provided little value (see 3.1.2). Having a less restrictive methodology with more focus on ideals allowed us to maximize the time spent on research and implementation.

Some of the Scrum practices, especially the sprints and retrospectives, helped the team stay grounded with routines. The sprints provided a solid framework for organizing work with scheduling and partitioning work, while the retrospectives forced the team to self evaluate every two weeks, which is vital in any development effort.

7.4 Conclusion

The group successfully made a VR application with a corresponding database according to the customers requirements. The database allows for uploading 3D environments into experiments with questions, answers and other data. These experiments can be downloaded and run on any modern Android device. In the final acceptance test (see Appendix E) all requirements were met and accepted and the customer expressed satisfaction with the final product.

The project gave the team valuable experience with a diverse set of tasks relevant to development in a work setting. This includes communication with a client, formulating and following a development method, the need for documenting requirements and architecture and how to cooperate with other people.

Bibliography

[1] Sami Abboud et al. “EyeMusic: Introducing a “visual” colorful experience for the blind using auditory sensory substitution”. In: *Restorative Neurology and Neuroscience* 32.2 (2014), pp. 247–257. URL: https://content.iospress.com/download/restorative-neurology-and-neuroscience/rnn130338?id=restorative-neurology-and-neuroscience%5C%2Frnn130338.

[2] Michel Chaudron et al. *Advanced software architecture - Documenting in UML.* 2016. URL: https://www.youtube.com/watch?v=7RtC1W16Ymo&list=PLA5EqHES4pl8Gz7pUNhIlTPO70y7UV9nc&index=24.

[3] *Colorophone, A sighthearing experience*. URL: https://www.colorophone.com/.

[4] Google. *Develop for Cardboard*. URL: https://vr.google.com/cardboard/developers//.

[5] USA Justin Ellingwood. *An Introduction to Continuous Integration, Delivery, and Deployment*. 2017. URL: https://www.digitalocean.com/community/tutorials/an-introduction-to-continuous-integration-delivery-and-deployment.

[6] Sydney Macquarie University. *Synaesthesia Research, What is Synaesthesia?* URL: http://www.cogsci.mq.edu.au/research/projects/synaesthesia/.

[7] Peter B. L. Meijer. “An Experimental System for Auditory Image Representations”. In: *IEEE Transactions on Biomedical Engineering* 39.2 (1992), pp. 112–121. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=121642.

[8] USA OWASP.org. *OWASP Top Ten Project*. 2017. URL: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project.

[9] USA php.net. *Dual procedural and object-oriented interface*. 2015/2016. URL: http://php.net/manual/en/mysqli.quickstart.dual-interface.php.

[10] Mary Poppendieck and Tom Poppendieck. *Lean Software Development: An Agile Toolkit*. Reading, Massachusetts: Addison-Wesley, 2003.

[11] Bob Wescott. *The Every Computer Performance Book, Chapter 6: Load Testing*. 2013. ISBN: 1482657759.

Appendix

Appendix A - Additional CLRF BLE integration

The CLRF BLE distance sonification is a subproject involving the CLRF Navigator app developed last year and a Bluetooth Low Energy distance measuring sensor developed by a master student tutored by Dominik.

The main purpose of the subproject is to implement a way to sense and sonify distance from a wireless Bluetooth Low Energy sensor in the already existing CLRF Navigator app. With this functionality in place, the app combined with a prototype of the Colorophone headset can function as a finalized prototype for the Colorophone project.

A.1 Prestudy

As a part of the project introduction the customer proposed the possible additional task of implementing an interface and algorithm for receiving and translating Bluetooth signals for the original CLRF Navigator app, the previous project worked on by Colorophone. These Bluetooth signals would come from a pair of distance measuring sensors, and our task was to map these values to corresponding ”ticks” of audio.

The purpose of these ticks is to give the user an auditory response to how far they are from an object they are facing, similar to the sensors used in cars to avoid collisions. A user will use the frequency of the ticks to indicate the proximity of what’s in front of them.

The current implementation uses a simple IR sensor, whereas the finished prototype will be using both an infrared and a ultrasound sensor. The use of two sensors will give a better concept of depth for the user, and it was specified by the customer that this will not change anything in the implementation, aside from changing a few variables related to Bluetooth discovery.

The customer explained to us how these ticks would be generated and at what rate they should be generated. Aside from the implementation and technical information, the prestudy had already been conducted by the customer who even had a prototype sketch of the entire system available to show.

A.2 Technical implementation

The most technical and challenging part of the task was making the Bluetooth device pair with the phone and maintaining the connection in order to keep streaming data. A lot of time was spent figuring out how Bluetooth Low Energy applications work and how to implement Bluetooth Low Energy connectivity and functionality in an Android application.

Unfortunately, due to a few misunderstandings and/or inexperience, some time was spent researching and working on Bluetooth implementations which ended up irrelevant for the project.

Due to the implementation being a part of the old CLRF Navigator application, a fair amount of prestudy was required to understand the existing code, to avoid causing bugs and accidentally removing or overwriting functionality that was implemented earlier. Wherever it was possible, old modular code was re-used.

A.3 Requirements

The customer decided not to specify any formal requirements for this project, other than a brief description of the task. In order to have a definition of done for the subproject and to keep track of what has been developed requirements were made by the development group which coincided with the customer's vision.

ID:	R1
Title:	Connect to a chosen BLE device
Role:	User
Description:	When a user activates the Bluetooth functionality they will get a list of all discovered device, and can click on the desired one to connect.

ID:	R2
Title:	Sonification
Role:	User
Description:	The application is able to reliably handle the incoming data from the Bluetooth device and translate it into sound without any noticeable performance loss

ID:	R3
Title:	Stable connection
Role:	User
Description:	The application must not disconnect from the Bluetooth device unless ordered to, and in the case of a disconnect the device will automatically reconnect.

ID:	R4
Title:	Sonification adjustment
Role:	Admin
Description:	An admin user wants to be able to change parameters in the sonification algorithm used in case it's imperfect.

ID:	R5
Title:	Sensor toggle
Role:	User
Description:	The application needs a toggle to swap between two incoming Bluetooth sensors, one infrared and one ultrasound

A.4 Architecture

The implementation uses the same architecture as the CLRF Navigator application and the added functionality of connecting to a Bluetooth device and receiving input data is built on top of this architecture, following the given code specification from the documentation and made with the same modularity in mind. The methods used to implement this functionality are primarily found in the Camera Activity class and its presenter class and settings used to tweak the algorithm are using the Android Shared Preferences API for storing and managing.

In addition a List View class named DeviceListView was made to accommodate a view on top of the Camera Activity to show all currently discovered Bluetooth devices and lets a user select which device to connect to.

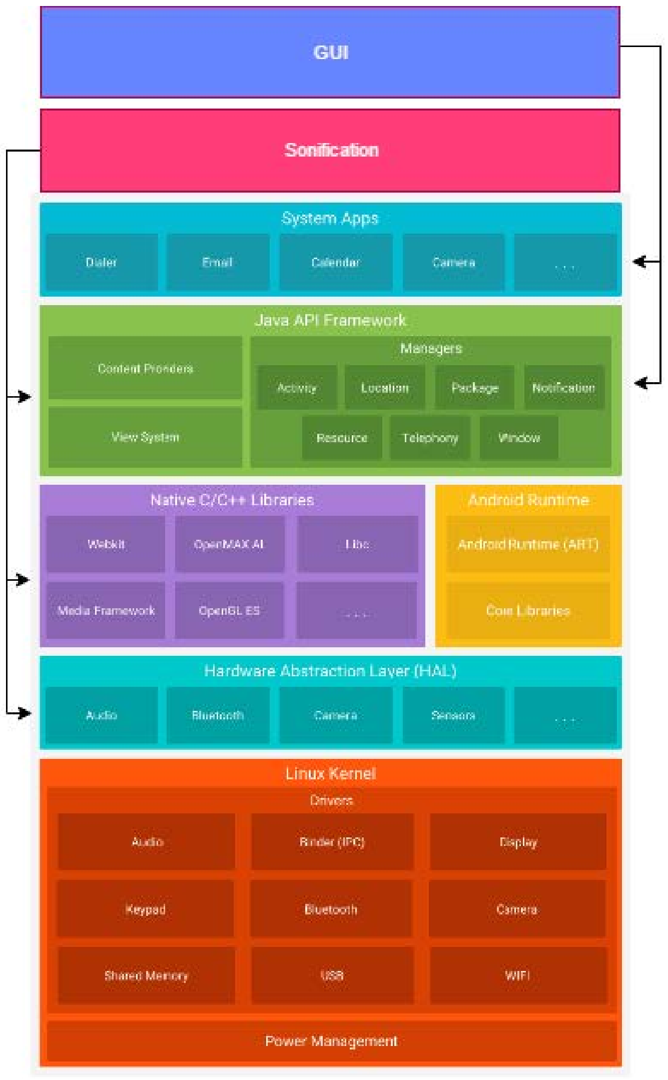


Figure A.1: CLRF Navigator architecture

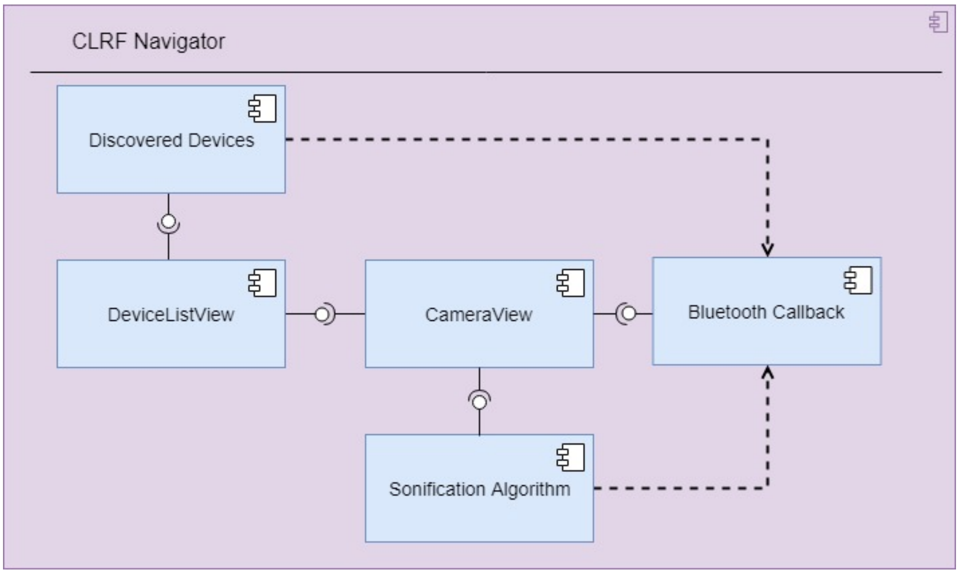


Figure A.2: Component diagram

This component diagram highlights the features added to CLRF Navigator. ”Bluetooth callback” is abstracted for ease of reading and understanding.

In the OnCharacteristicChanged callback invoked when the application connects to a BLE device a new thread is created for handling the incoming data stream. The thread asynchronously from the rest of the application uses the incoming value from the distance sensor in a summing algorithm which adds together until it reaches a threshold, at which it will generate a ticking noise. The value added to the sum is lower in the case of a longer distance, making it take longer time to reach the threshold. The way it is currently set up is equal to a linear function which generates ticks at approximately 1/second per 1 meter as required by the customer. This is the same rate as seen in the VR application.

A.5 Testing

Due to the lack of formal requirements for the subproject it was not put through sufficient unit testing, and large parts of the system has already gone through unit testing when it was first developed. The customer considered it satisfactory enough by seeing the implementation work as intended, and through user testing it was made sure the application worked as the customer wanted it to without noticeable errors. In order to heighten the quality of the product, several quality of life improvements were implemented into the application. Some of these were to help satisfy the requirements.

Scenario Nr	Description	Walkthrough
1	Enable Bluetooth discovery and connect to a selected device	Press Bluetooth discovery button and select the desired device from the list view
2	Automatically reconnect to device after unplanned disconnect	By forcing a disconnect by restarting the Bluetooth device an automatic callback ensures the application tries to reconnect once a disconnect happens. This happens without user interference
3	Change settings in the application to change the rate audio is generated	Using the same settings page already existing in the app, the user can easily change values used in the sonification algorithm

Table A.1: BLE testing

Appendix B - User manual

B.1 Initial configuration

1. Open Unity and press ‘New Project’.

2. In this window, ‘Project name’, ‘Location’ and ‘Organization’ you can set to whatever you want, also, ‘3D’ should be checked. If you want to use ProBuilder as well, it can be imported with the ‘Add Asset Package’ button.

   IF you already downloaded ‘Asset Bundle Browser’ in a previous project, simply press ‘Add Asset Package’, select ‘Asset Bundle Browser’, press ‘Done’, then ‘Create Project’ and skip to ‘B.2 Build the room.’.

   Click ‘Create Project’.

3. Go to ‘Asset Store’ in Unity and search for ‘Asset Bundle Browser’ by ‘Unity Technologies’. It should look something like this figure B.1.

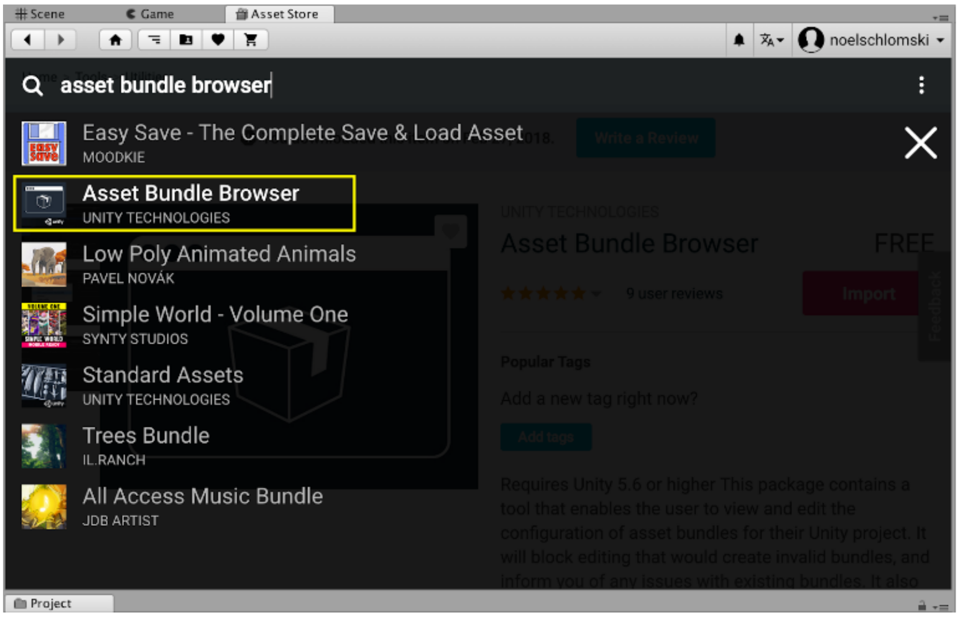
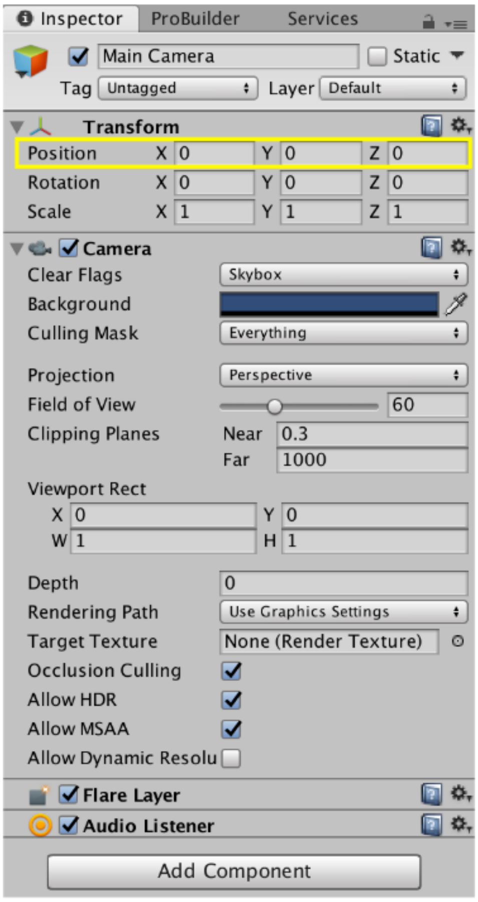


Figure B.1: Asset store

Click on download, then import. A screen will pop up asking you to select which assets to import. Select all and click import. If prompted: "API Update required", press "I Made a Backup. Go Ahead".

B.2 Build the room

1. Now we can start building the room. Set 'Main Camera' to position X:0, Y:0, Z:0, like in figure B.2.



Inspector | ProBuilder | Services

✓ Main Camera ☐ Static

Tag: Untagged | Layer: Default

Transform

Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

✓ Camera

Clear Flags	Skybox
Background	🟦
Culling Mask	Everything
Projection	Perspective
Field of View	60
Clipping Planes	Near 0.3
	Far 1000
Viewport Rect	X 0, Y 0
	W 1, H 1
Depth	0
Rendering Path	Use Graphics Settings
Target Texture	None (Render Texture)
Occlusion Culling	✓
Allow HDR	✓
Allow MSAA	✓
Allow Dynamic Resolu	☐

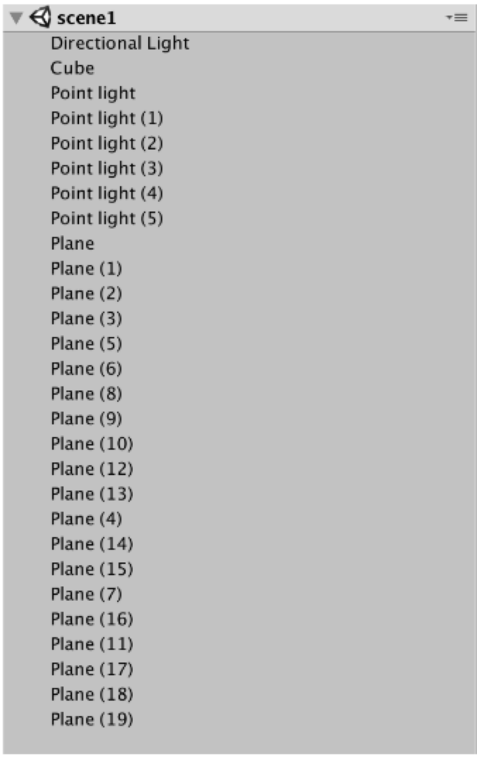
✓ Flare Layer

✓ Audio Listener

Figure B.2: Main camera

Then, build the room around it with whichever API you choose, e.g 'ProBuilder'. This is to make sure we get correct camera positioning when loading from mobile. The camera in our mobile app will load from the X:0, Y:0, Z:0 position.

2. When finished with room building, make sure the camera doesn't have any nested objects and stayed in position 0,0,0, then **DELETE** it. The scene should now only contain lightning and objects, like in figure B.3.



scene1
Directional Light
Cube
Point light
Point light (1)
Point light (2)
Point light (3)
Point light (4)
Point light (5)
Plane
Plane (1)
Plane (2)
Plane (3)
Plane (5)
Plane (6)
Plane (8)
Plane (9)
Plane (10)
Plane (12)
Plane (13)
Plane (4)
Plane (14)
Plane (15)
Plane (7)
Plane (16)
Plane (11)
Plane (17)
Plane (18)
Plane (19)

Figure B.3: Unity scene

3. Save the scene.

B.3 Packaging to ‘Asset Bundle’

1. When the room is finished, we can package it so that the mobile application can read it from the web. Select ‘Window’ ->‘AssetBundle Browser’ in the Unity toolbar. See figure B.4.

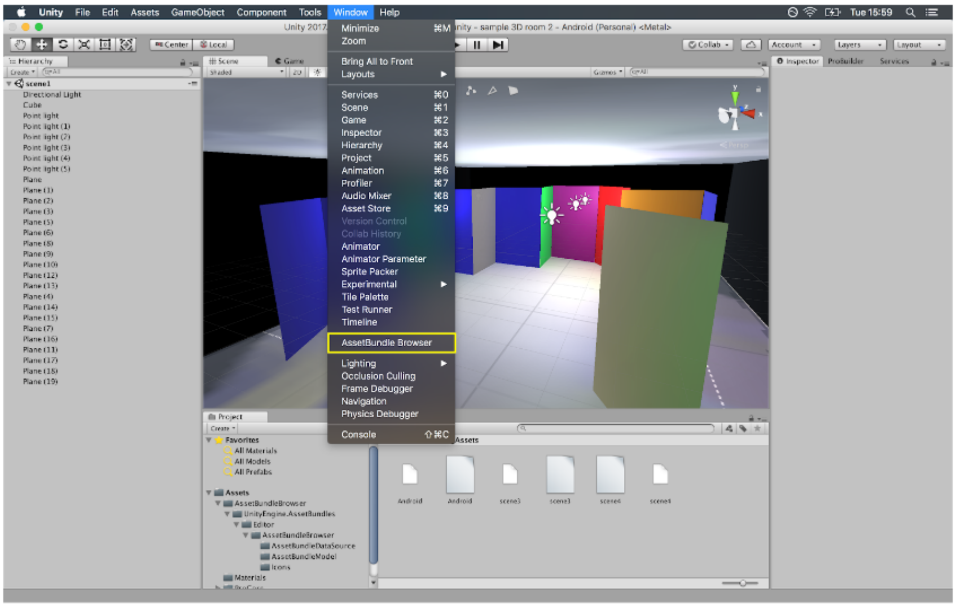


Figure B.4: Location of AssetBundle browser

The window shown in figure B.5 should open.

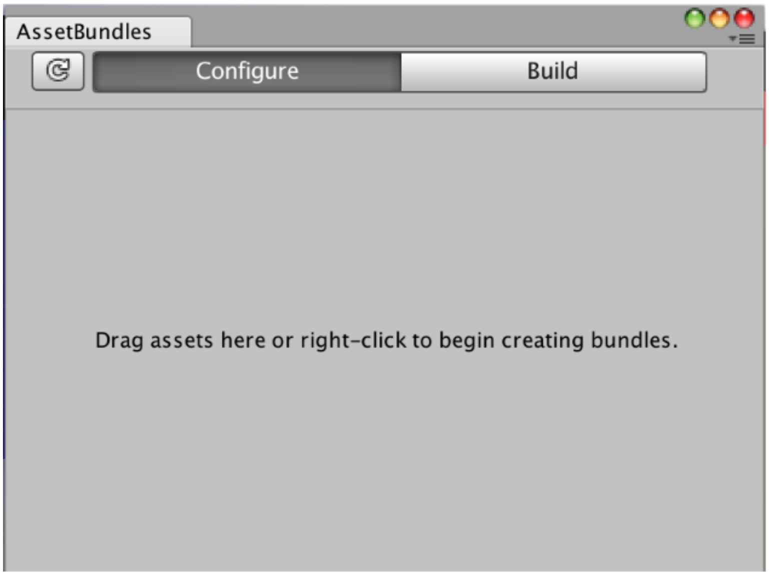
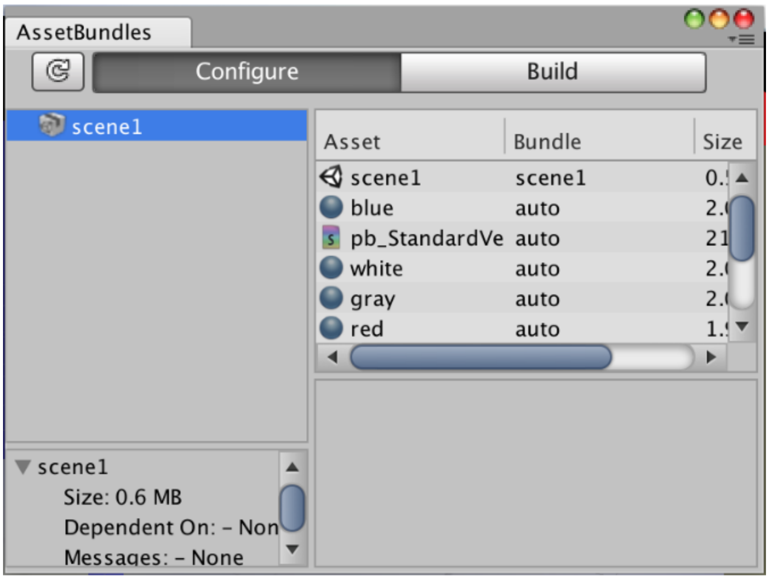


Figure B.5: AssetBundle browser

2. Drag your whole scene into this window, it should look something like in figure B.6.

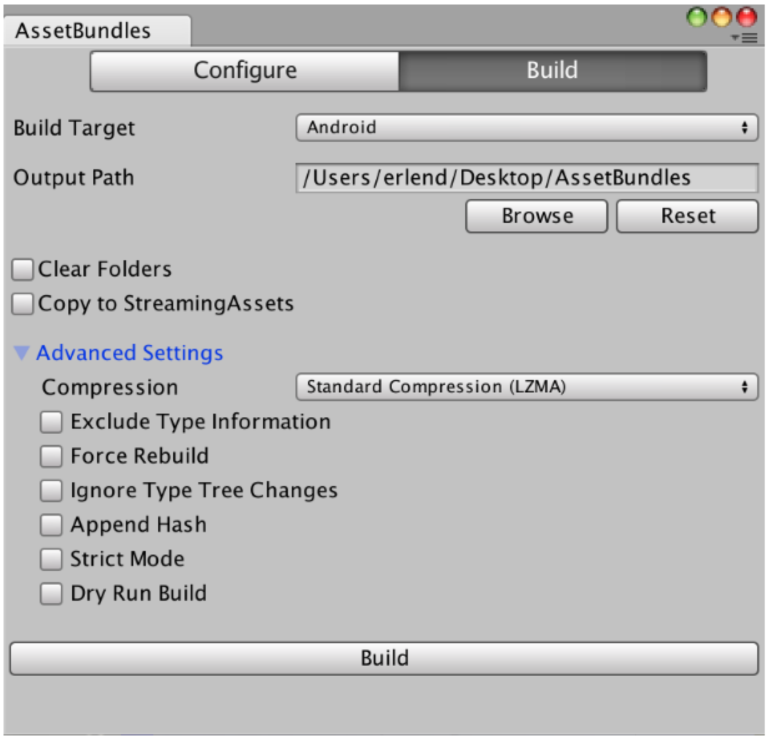


Asset	Bundle	Size
scene1	scene1	0.
blue	auto	2.
pb_StandardVe	auto	21
white	auto	2.
gray	auto	2.
red	auto	1.

scene1
Size: 0.6 MB
Dependent On: – Non
Messages: – None

Figure B.6: AssetBundle browser with a scene selected

3. Now open the ‘Build’ tab and set ‘Build target’ to ‘Android’ and ‘Output path’ to the folder where you want your asset bundle saved. It should look something like figure B.7.


AssetBundles

Build Target: Android

Output Path: /Users/erlend/Desktop/AssetBundles

Clear Folders: unchecked

Copy to StreamingAssets: unchecked

Advanced Settings

Compression: Standard Compression (LZMA)

Exclude Type Information: unchecked

Force Rebuild: unchecked

Ignore Type Tree Changes: unchecked

Append Hash: unchecked

Strict Mode: unchecked

Dry Run Build: unchecked

Figure B.7: AssetBundle browser build

4. Click ‘Build’... After some time packaging, the asset bundle should show up in the folder you chose in step 3c. It is the file with no file extension that is our asset bundle. (In this example, i got the files “scene1” and “scene1.manifest”. “scene1” is my asset bundle and “scene1.manifest” could simply be discarded as we do not use it.)

B.4 Creating experiment

After asset bundle is made.

1. Go to `http://54.85.64.72` and log in with:
   Username: clrf
   Password: Ask Dominik
   You will automatically be redirected to the “Add experiment” page.

2. Click on “Choose file” under “Select model to upload” and select the assetbundle created. Make sure the file is not zipped.

3. Fill in the fields:

Name - This will show up in the list of experiments and is used to identify and differentiate between different experiments. It could be smart to think of a good naming scheme so that experiments easily can be identified when the number of experiments gets high.

Description - This text will show up in the app as the "instruction" part before the experiment. It will show up after the "put on your headset" tutorial when the user have clicked ready.

Time limit - How long the user is going to "explore" the model before the question part starts. Values are given in seconds.

Date - The date of deployment. At this date the experiment will become available for the users. The user has one week to do the experiment before it becomes unavailable again. Admins can choose the experiment at any date.

Questions and alternatives - To the right of each alternative is a radio button to mark which alternative is the right one. Make sure one radio button is marked on every question, as the webpage does not require it. The questions will show up after the "exploration" phase when going through the experiment in the app. Each question is shown with its alternatives, and the time spent on answering the question is logged. The questions are shown in the order they are created on the webpage. The system is made to have up to 10 questions with up to 5 alternatives on each question.

NB: If you get an error message, please note it down and delete the experiment if it was created. Also if you are using ' or " the questions and alternatives containing these will not show up when editing the experiment but will show up in the experiment.

4. Click on "Upload experiment". It might take some time depending on the size of the model and the internet connection.

5. Congratulations! you have now uploaded an experiment.

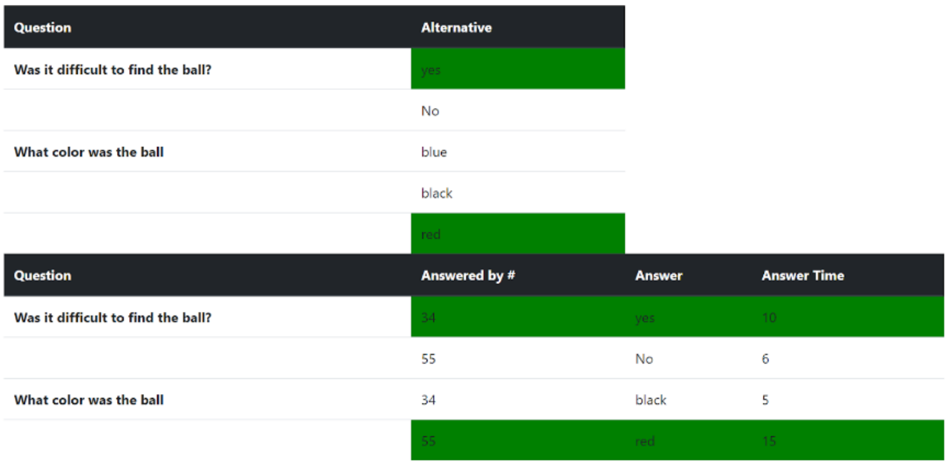
B.5 Edit experiments page

On this page you get an overview of previously created experiments. This can be used to verify if the experiment you created was successfully added or to edit it. It is not possible to edit experiments that has been conducted by a user. This is done to ensure that the database only contains results from experiments that is equal for all of the users. It is also not possible to change the model used in the experiment (this saves us for a lot of potential errors). If you want to change an experiment that already have been conducted by an user or change the model, please delete the old experiment and create a new one. User data from this experiment will also be deleted.

B.6 Statistics page

Here you will get an overview of user data gathered when conducting the experiment. Click on “Show all users” to get a list of Phone-ids. Click on the wanted Phone-id to get an overview of experiments conducted by that Phone-id.

To get the results of each experiment click on “Show all experiments” and click on the experiment id. First all the questions and alternatives connected will be listed with green background indicating right answer. Then the answers given by the users to each question will show with green indicating right answer. Figure B.8 is an example of how it should look with two questions answered by two different users:



Question	Alternative
Was it difficult to find the ball?	yes
	No
What color was the ball	blue
	black
	red

Question	Answered by #	Answer	Answer Time
Was it difficult to find the ball?	34	yes	10
	55	No	6
What color was the ball	34	black	5
	55	red	15

Figure B.8: Statistics

B.7 Admin page

Here you can toggle admin privileges for different phone-ids. A new phone-id is automatically generated when installing the app. The phone-id is shown in the main menu of the app and is connected to every single installation of the app. This means that if the app needs to be reinstalled, a new phone-id will be given.

When a new phone-id is registered it will show up in application. You can here either accept the application or decline it. Accepting it will add it to the list of users, while declining it will delete the request. This is used as a security mechanism to prohibit false users from registering and also gives you the opportunity to control which user is connected to which phone.

The experiment results is also connected to this phone-id, and it could therefore be smart to keep track of which phone-id is connected to which participants. This can be done by changing the aliases connected to the phone-ids. In case of a new installation, the old phone-id will contain results for experiments conducted before the reinstall, while the new one will contain results for after the reinstall.

B.8 Sonification settings

This is for Dominik to change various parameters in the algorithm.

Appendix C - Retrospective notes

Sprint number	What went well?	What did not go so well?	What can we do better next sprint?
3	- Worked more than last week, worked with more meeting time. - Got a little better started	- Went back on schedule - Did not stick to what we have (in relation to plan) - Uneven labor allocation - People coming late - Not write ”group meeting” instead of describing what you’re working on in work hour list. - Much delay from client	- Aim for everyone to work 20 hours a week. - Write what you have worked on in the work hour list. - Try to get a consistent solution with client and go for it. - Work from home if you are gone - Work on page of group meetings - Write good work hours notes after each job-sesh - Communicate better at Slack if you work from home - If you arrive late, withdraw from time spent (more than 15 minutes). - Tell group if you think about working at other times.

4	- The hours have grown better, people work up against 20 hours - We have actually written what we are working on in the work time table - We have a solution and we go for it, the decision of the customer - Everyone has the same idea of how the product should be - Coming quite a long way with the ”game-ification” product and what remains (technical).	- Work from home if you are gone. We may not have been so good at? - Lost a lot of work - Did not write anything on the report (!) - Lack of organization of the group (process) - Difficult to match what has been written in the report with what has actually happened and it is difficult to explain and argue for what we have done so far because we have not made proper decisions and decided how to work.	- Go through what we are going to do in a report - Define what our process entails, and justify the choice of working method - Keep working, have improved, but people can even stretch a bit longer to reach 20 hours. Also remember to keep these hours. - Get customer to decide on one solution, not keep changing - See new items in retrospect - Add some points in reflection / reasoning - Review organization of the group: - Difficult to distinguish between group meetings and work sessions, important that the group gets peace to work with the product without negotiations and discussions. decision Model - Better work habits - Time limit for meetings not to expire - Meeting agenda in own document. roles (distribution / limitation of responsibility) - Work more structured with the report - More shared work with the report

5	- Got much work done much despite half capacity (Easter holiday) - Good time management - Worked better individually - No conflicts in decisions	- The group needs to update others on what people work with - Time limit on meetings. Meeting taking to long - Still a little uncertain what the roles mean.	- Meeting at set meeting time - Communicate better what you have worked with so everyone gets an overview of the project - Review the roles again - Suggestions for new roles: - Shivam - Scrummaster and group leader, webmaster. - Erlend and Thea - Rapporteur - Jonas - Test Responsible, Database - Iver - Sonication Officer - Stian - Side project, backend.
6	- Good workflow. - Roles defined better. - People meeting on time	- Did not communicate good enough with customer, caused misunderstanding	- Communicate better. Spend more time on meetings and on slack with customer.
7	- Got a lot done on the report - Fixed annoying bugs - Seeing the end of the report	- Need to spend more time to complete the project. Everyone needs to meet up to work as only working from home is not enough.	- 10:00-15:00 is core time for working together, work 7 hours every day week 18. 20kr per half-hour being late

Table C.1: Summary of notes taken during retrospective meetings after sprints

Appendix D - Gruppekontrakt

Møtetidspunkter:

- Mandag 12.15 – 18.00
- Onsdag 14.15 – 19.00

Kommunikasjon:

- Tør å være uenig, ikke vær for høflig
- Angrip problemet, ikke personen
- Møt innspill med nysgjerrighet
- Husk at vi har ulik kompetanse

Prosess:

- Hold Trello oppdatert
- Respekter sprintmål
- Informer om planlagt fravær senest dagen før
- Større beslutninger tas i fellesskap, tvister avgjøres ved avstemning
- Alle kan endre all kode, men snakk sammen ved store endringer
- Last opp kode jevnlig til github
- Retrospektivmøte første mandag etter endt sprint
- Daglig standup-meeting hver mandag hvor utfordringer osv diskuteres.
- Dokumentere hva man har gjort/ hvor lenge man har jobba i timelista

Møtekultur

- De som deltar i et møte må være helt tilstede og helst bidra
- Gi et kort sammendrag av alle møter
- Alle skal jobbe   25 timer i uka
- Hvis man ikke rekker møtet burde man si ifra i god tid i forveien. God tid vil si så fort man vet at man ikke har mulighet.

Alle har like stort ansvar for å bidra med planlegging og implementasjon av prosjektet.

Appendix E - Test results

E.1 User test researchers and client:

General comment:

Excellent job guys! :-) There are some minor issues listed below. You can also find individual feedback from every tester at the end of this document. Please refer to the issue by the number associated below. Issues 1-9 are of high priority.

Issues:

1. Blue frequency is the same as green - check variable association. It seems that you are using green frequency for both green and blue sound.
2. Yellow sinus "Enabled" button is not turning off the yellow sound.
3. Sonification area is not in the center of visual field.
4. Experiments for a given day are not visible when admin mode is off (is it because they have been performed in admin mode?).
5. Sometimes scenes do not load. Experiment is loaded with instructions and questions after, but nothing is visible or heard.
6. It is not possible to turn off distance sonification from web interface.
7. Buzzing sound before every tick.
8. If "Loudness Exponent" is not equal 1 sounds get very unpleasant. Something is wrong with scaling there.
9. Screen is not black in non-admin mode (see Kasia's feedback for details).
10. Quit button does not quit app, just minimizes it. Can it lead to problems?
11. Sometimes buttons need to be pressed twice (in Select experiment section only).
12. Sometimes back button is not working on the "Statistics" page.
13. Occasional scene drift while not moving the phone.
14. Too short time of displaying the last message.
15. Spelling error in "Sampling frequency" is "Samling frequency" now.

Questions:

1. Did Iver update the distance sonification algorithm as we agreed? It seems that the slope is not starting to rise right after tick.
2. Can you explain what do following parameter mean in distance sonification?
   Gain
   Tooth Frequency
   Sine Frequency
   Counter
   Tooth Amplitude

Priority:

1. Can we have the sonification area marked as a rectangle in admin mode?
2. Could we have a possibility to delete all test data from the database for a given phone or get the full access to db, so we could do it by ourself?
3. Set up more than one day for experiments to be performed (one week buffer).

Optional:

1. It would be nice to have stereoscopic vision in admin mode.

Individual feedback:

Paweł:

- Enactive torch* might need some tweaking (although I'm not sure if that's not the cause of not having the feedback on the point of focus). I tend to move my head pretty fast and it not always could follow my movements to give me sufficient distance information I personally would prefer the signal to be more frequent.
- Sometimes some procedures do not open for no reason.
- I didn't face any lagging, but Magda reported some incongruencies in between sounds and perceived image.
- It would be nice to have stereoscopic vision in admin mode.
- In admin mode it would be useful to have some indication where current point of focus lies (like in the case of the actual colorophone).
- Great job overall, I'm really impressed. I really like how you guys handled questionnaire mode - it's very clear and intuitive how to use it, nice and user-friendly, also very toned down and simple in design.

*ticking connected with distance

Patrycja:

[APP] tested in admin mode:
- admin mode is working: it is visible at the bottom of the screen that the mode is on, you can see all experiments, even from the previous day or when already done once
- sometimes buttons need to be pressed twice (in Select experiment section only)
- all experiments, when launched first time after entering the app, were working, but when launched multiple times are crushing - I could press the button "Ready", and then the screen is black and you don't hear anything; after specified time questions are visible correctly
- when doing an experiment, only once (on the phone I was testing) it happened that the scene moved by itself, but after few seconds it was good again. Although I was also testing for shorter time the app on other our phones (all Sony Xperia XZ) and it happened that for some that was a very often appearing problem: when moving around with phone, scene was often coming back to one place, and it was not moving the same way I was moving the phone
- the area that was analyzed was not in the middle, but around upper right corner
- it would be good, if possible, to have a square of the area that is analyzed when running an experiment
- it would be good if we can change descriptions that are visible while taking an experiment (like "Put on the headset" or "Ready") to Polish language
- sometimes experiments don't show in admin mode, the app needs to be turned off and on again

- I have tested 8 other phones to check if the scene is moving/jumping by itself (10 experiments per phone): 2 of them in two first experiments failed it, the rest 8 was ok; 2 failed it for most of the experiments (one was working mostly fine one day, the next failed all

tests); 4 phones failed all the 10 experiments; it happens very often that the scene is coming back to "0" point (video2 on slack), doesn't matter which direction I turn it, while for some tests the scene is working more or less, but it's slightly moving (video3)
- no visible experiments for non admin mode

[WEB]
- if possible, there could be added one more information in section "Statistics" about how many times the experiment was taken (?)
- in section "Statistics", the back button sometimes is not working unless you refresh the website
- would it be possible to delete chosed phone IDs and results too? That will be helpful when we test experiments before giving phones to testers, so after we don't have fake data when analyzing all

Magda:

- Second and fourth experiment didn't worked - initial command was seen but then only a black screen was seen without any sound and at the end the question was displayed
- The last inscription at the end of procedure should be displayed a little bit longer at the first time - I couldn't read it
- The 'ready' command and the selection of questions look great, but we should mention our participants that sometimes they must move around to find the instruction and 'ready' command - they sometimes don't display at first glance when you wear the google and you must move for example 180 degrees
- In the first experiment the walls moved when I had my head steady in one point, but then in the next try there was no problem with it
- In admin mode there should be the square of the area that is analyzed by the Colorophone

Kasia:

- During non-admin mode I could see only one experiment (11.1). The screen wasn't black as supposed to, I've seen this:

- The page seems really great. However I was wondering, is it possible to retrieve the access to the experiment for a particular user? Some kind of manual control over their experiments?
- One thing that was bothering me (in admin-mode): sometimes when I concentrated on one thing (that means I didn't move so the sound should be constant) I could notice that, despite the lack of change in 'input', the sound differed (just before the click or while approaching the click), like it would be higher or louder.
- Besides I agree that it would be useful to know the point of focus and to manipulate the time of the message
- It happened twice that after starting the screen just got black and the phone locked (or at least that’s how I have interpreted it) and it come back to the menu after the some time (I guess the time of the experiment)’
- Overall good job! :)

E.2 Acceptance test client:

Acceptance test

Summary of acceptance test:
The team has developed a first VR based color sonification system in the world, which was a challenging task. The system has passed acceptance tests on all points with no comments. The system consisting of a mobile app and a web interface is working according to the requirements. Both components are synchronized between each other, in the way that the data can be exchanged in a real time. The web interface allows to upload, edit and delete experiments, including their descriptions, questions and setting duration time. User can also switch between volunteer and researcher mode, as well as see all the statistics, by choosing whether to see results of a specific user or experiment. The statistics are updated in real time, after finishing an experiment. Web interface allows to modify sonification parameters as well. This part is secured and requires logging in, in order to use it. The app has two modes: for volunteer and for researcher. After entering the app, user can see whether he is logged in as a researcher or volunteer as well as see ID of the phone. A pool with instructions is available, with description on how to use the app. In the section with experiments, user can either see all experiments, if he is logged in as a researcher, or see only those available for a specific day, including those he has not done the previous days. After entering an experiment as a researcher, an instruction on how to place the phone in a headset is shown and after positioning the phone horizontally, timer counts to 10 s and stereographic view of the experiment is visible. User can look around, image is stable, works together with head movements. Sonification is working well, colors can be distinguished based on the set algorithm. Experiment timer works well, the task is performed for a given amount of time, set in the web interface. After the experiment, there are questions and answers visible, if set so. Choosing an answer is intuitive, the app can indicate wrong and right answers if that option was chosen while uploading an experiment on the web interface. When all the questions are answered, or there are no questions, user can see a message that the experiment has ended. In the volunteer mode, after entering the experiment, instruction and counter is shown as in the researcher mode. After that, the screen goes black and user can only listen to sounds while moving head. When the set amount of time has passed, as it was in the researcher mode, questions and/or further instructions are visible on the screen. Overall, the system is working very well. It is easy and intuitive for both researchers and volunteers to use it with the provided instructions.

ID	Description	Accepted (yes/no)	Comment (optional)
F1	The app needs to read colors correctly from the virtual environment.	yes	
F2	The color should be collected from an area.	yes	
F3	The app should use the correct sonification algorithm.	yes	
F4	The app should play the correct kind of sound.	yes	
F5	The app needs to collect the distance to objects in the 3d room.	yes	
F6 (redundant)			
F7	The app needs to play the correct sound corresponding to the distance.	yes	
F8	The app needs to be runnable on target devices.	yes	
F9	The mobile app shall display a tutorial on the screen after an experiment is selected.	yes	
F10	The mobile app shall have a menu for launching experiments.	yes	
F11	The system needs an model for displaying the questions.	yes	
F12	The system needs to accept gaze input.	yes	
F13	The mobile app needs to load the questions from the database in a readable manner.	yes	
F14	The system shall provide a web interface for adding questions, alternatives and 3D-models for the experiments.	yes	

F15	The system shall provide a web interface for adding experiment name, description, date of deployment and time limit for the experiments	yes	
F16	The system need to store experiment data in a database and 3D-model on a file server.	yes	
F17	The mobile app need to get the questions and alternatives from the web server and load them correctly into the questions model in the corresponding experiment.	yes	
F18	The mobile app needs to get the 3D-model from the file server and load it correctly	yes	
F19	The system needs to provide a template for creating 3D-models so that the researchers can make compatible models	yes	
F20	The mobile app needs a system for choosing, downloading and running the correct experiment.	yes	
F21	The system needs a web interface to change sonification parameters.	yes	
F22	The system needs to store sonification parameters from the web app in a database and fetch these to the mobile app.	yes	
F23	The system needs logic in the mobile application for obtaining sonification parameters and change the algorithm used accordingly.	yes	
F24	The web application shall display the answers from the experiments sorted on both users and experiments.	yes	
F25	The customer should get	yes	

	safe access to the database to structurally collect and analyze data via specific queries.		
F26	The mobile app shall send answer data to the web application and it should be saved to a database.	yes	
F27	Data collected to be connected to the user going through the experiment.	yes	
F28	The mobile application needs to differentiate between experiment volunteers and researchers.	yes	
F29	The web application need to have an interface to select which user is a researcher.	yes	
F30	The mobile app needs to show all experiments to researchers, while only today’s experiments are available to volunteers.	yes	
F31	The mobile app needs to show experiments with visual input to researchers, but only auditory input to volunteers.	yes	

ID	Description	Accepted (yes/no)	Comment (optional)
R1	When a user activates the Bluetooth functionality they will get a list of all discovered device, and can click on the desired one to connect.	yes	
R2	The application is able to reliably handle the incoming data from the Bluetooth device and translate it into sound without any noticeable performance loss	yes	
R3	The application must not	yes	

	disconnect from the Bluetooth device unless ordered to, and in the case of a disconnect the device will automatically reconnect.		
R4	An admin user wants to be able to change parameters in the sonification algorithm used in case it's imperfect.	yes	
R5	The application needs a toggle to swap between two incoming Bluetooth sensors, one infrared and one ultrasound.	yes	

Patrycja Bizon and Dominik Osinski

Kraków and Trondheim, 30.05.2018