A Virtual Reality and Wearable Device Technology System to Monitor Health Data and Deliver Phobia-motivated Experiences

Submitted May 17, 2020, in partial fulfillment of
the conditions for the award of the degree **BSc Computer Science.**

Denis Stepanov
20027064

Supervised by Dr Dave Towey

School of Computer Science University of Nottingham Ningbo China

Acknowledgements

This project could not have been completed without the help from my supervisor - Dr Dave Towey, to whom I express my highest gratitude for helping me on every stage of the way and always supporting me during the hard times in this project.

I also want to express my appreciation to the convener of the Human-Computer Interaction and Final Year Project modules at University of Nottingham Ningbo China, Dr Matthew Pike, for allowing me to gather student feedback on my applications.

Additional recognition goes to the healthcare professionals who helped me during the testing stage of the Virtual Reality application, Dr Eugeniia Zhukova and Dr Olga Philipchenko.

Abstract

Virtual Reality (VR) has the potential to help to evaluate and assess post-traumatic stress disorders (PTSDs), delivered through Virtual Reality Environments (VREs). It can help to analyze the stress levels more safely and provide a possible solution or treatment for the user with a specific fear or phobia. One of the greatest advantages of VR is that individuals know that the reality they are facing is virtual, but their body and mind act as if it was real. Therefore, it is easier to help healthcare professionals to understand the impact of PTSD on users without real-life exposure to stressful situations. However, background research shows that there are not many VR applications in the healthcare industry that are targeting the evaluation of stress disorders at the moment. This project provides a system where a healthcare professional could potentially see the results of body sensors attached to a person (wearable technology) and analyze the impact of the VR exposure to stressful situations. The sensors on the wearable device should show the typical reaction to the fear, such as a changing heart rate. The goal of this project was to provide a safe environment for the user who would be immersed in the VRE of their fear and to access as much data as possible using wearable technology.

Publications

The work surrounding this project has resulted in two academic papers. One presented at IEEE conference on Teaching, Assessment and Learning for Engineering (IEEE TALE) [1] and one accepted to appear in the Proceedings of the 44$^{\text{th}}$ IEEE Annual Computer Software and Applications Conference, COMPSAC 2020 [2].

Contents

List of Figures

List of Tables

List of Abbreviations

CC BY	Creative Commons Attribution License
CC	Creative Commons
ECG	Electrocardiogram
EEG	Electroencephalogram
FDA	Food and Drug Administration
GSR	Galvanic Skin Response
GVR	Google Virtual Reality
HR	Heart Rate
IDE	Integrated Development Environment
iOS	iPhone Operating System
macOS	Macintosh Operating System
MR	Metamorphic Relation
MT	Metamorphic Testing
OER	Open Educational Resources
PTSD	Post-Traumatic Stress Disorder
RAM	Random-Access Memory
SDK	Software Development Kit
SQA	Software Quality Assurance
UI	User Interface
VRET	Virtual Reality Exposure Therapy
VRE	Virtual Reality Environment
VR	Virtual Reality

Chapter 1

Introduction

Since the release of the first VR headsets in the 1960s, the VR technology has been evolving and progressing all over the world [9]. Now, there are more affordable headsets on the market, such as Google Daydream [10], Oculus Go [11] and others, which interest many researchers, especially in the healthcare industry [12]. Considering that the first VR interactions were introduced more than 50 years ago, there is still a lot of potential to grow in terms of helping patients and doctors in the field. VR systems and Virtual Reality Environments (VREs) are becoming more common in the healthcare applications for training, educating professionals, and treating patients [13]. This chapter describes the motivation and background of this project, and outlines the aims and objectives of the work.

1.1 Background

The technological changes in the healthcare industry significantly affected the education of healthcare professionals. A study shows that, on average, a person needs up to thirty years to find a job in the healthcare industry [14]. However, advances in technology offer new innovative tools when it comes to education in the healthcare industry. VR is now used in various fields of medical treatment, including professional education and training, diagnostic assistance, rehabilitation and diverse mental treatments for patients [15].

Anatomy is one of the most studied fields in the healthcare industry [16]. For the past five years, almost 90% of the applications for VR were made to simulate surgery, treating patients' mental disorders and precisely diagnosing various mental diseases [17]. However, there are still not many applications that are present on the market that help to analyze the problem and the diseases. The study of brain activity has been one of the biggest interests in the VR healthcare industry for the past ten years [16]. Almost all of the brain activity research in VR is concentrated on human feelings that can be acquired virtually, such as fear or stress. A recent study shows that more than 90% of people have phobias, and healthcare professionals look for ways to analyze and treat them [16].

1.2 Motivation

Searching through the VR healthcare applications that are on the market now, it can be concluded that there are not many products available that help assessing personal fears. Limbix VR library shows that it is very important for the patient to completely immerse in the situation of fear and try to present it as realistically as possible [4]. There are several applications, such as Limbix [4] and C2phobia [3], which provide the necessary experiences, but none of them collect any health data to evaluate the impact of the virtual phobia environment. The problem behind displaying phobias in VR is the lack of a proper system which would help healthcare professionals to analyze the world of phobias and to assist their patients according to the proper diagnoses.

There are products in the healthcare industry that are made specifically for data collection. Qardio [5] is the arm monitor that provides details for the users about their health, such as heart rate and blood pressure. It is an example of wearable technology connected with a phone application to display data. However, Qardio only functions as a standalone measurement tool and does not include any VR environment to be tested with. Therefore, it is worth investigating wearable technology combined with VREs as a way to evaluate the impact of phobia-motivated experiences and develop a standalone platform.

1.3 Aims and Objectives

The main objective of this project is to create a system capable of providing VREs for the user with phobia(s) and deliver the patient's data to the healthcare professional(s) using wearable technology and mobile devices. Any VR headset capable of holding a mobile phone, an Apple Smart Watch and a mobile device with gyroscope (a sensor for detecting the orientation of the device) would be sufficient to run this project. This research has several key objectives:

1. Investigate previous and current methods that are used to treat patients with phobia(s).
2. Design a comprehensive system of cooperating hardware and software tools to display VREs addressing phobia(s).
3. Develop a mobile device application communicating with wearable technology which sends and displays the patient's information to the healthcare professionals. Include application testing to enhance the quality of the software.
4. Test the system on different hardware components, such as headsets, mobile devices, and smart watches.

1.4 Dissertation Outline

This chapter has summarized the VR history in the healthcare industry, outlined general background of the related work and stated the aims and objectives of this project. Chapter 2 describes previous work in the VR healthcare industry and gives specific details about what products are currently available on the market. Chapter 3 outlines software requirements specifications for this project, highlights the risks, and presents the design of the system. Chapter 4 states the implementation steps to create the mobile applications and performed testing processes. Chapter 5 describes the project management and feedback that was gathered after the system implementation. Finally, Chapter 6 summarizes the project, and states the reflections and considerations for future work.

Chapter 2

Background & Related Work

This chapter outlines major studies on VR exposure as a way to treat phobias, and wearable technology as a valid method to record stress levels. This chapter also includes the latest products on the market related to the VR healthcare industry.

2.1 VR Phobia Research

The idea of analyzing and treating PTSDs using VR exposure started recently with the basic implementation of virtual environments displaying various 3D images [13]. This section describes the potential advantages of VR exposure on patients and the risks that could be caused by it. It also argues why wearable technology is becoming one of the effective tools to collect personal health data from the human body.

2.1.1 VR Exposure as a Safe Method to Assess Stress

Considering the available therapies to treat PTSDs today, there are two main approaches: Virtual Reality Exposure Therapy (VRET) and in-vivo therapy [18]. VRET is a well-known practice nowadays when it comes to assessing the stress level. The main idea behind it is to immerse a user into the VRE where they would face a stressful situation. VRET is different from in-vivo therapy, where the patients or users would be accompanied by a psychologist or a related-field healthcare professional. In-vivo therapy is time-consuming, expensive and requires a lot of personal background knowledge about the individual when it comes to phobia treatment. VR exposure is challenging that method and tries to provide a robust approach for treating PTSDs at minimal cost and risk for the user.

VR therapy has been applied in many case studies, such as in fear of flying [19], nervousness in social situations, or social phobia [20], and acrophobia, a fear of heights [21]. In all cases, participants were put under stressful situations that are specific to them. After several sessions, it was confirmed that VR therapy lowered participants' stress levels and reduced anxiety [19, 20, 21]. Participants stated the effectiveness of VR exposure therapy as a method to reduce stress levels when facing their phobias. For example, 93% of the people who were afraid of flying and went through VRET, flew on airplanes after several sessions without any fear [19]. Cases presented here indicate that VR exposure is an effective method to treat phobias and possibly replace expensive and time-consuming in-vivo therapy.

2.1.2 Social Phobia Study

This section describes a social anxiety study [20], in which a VR application and 3D images were used to analyze and treat the social phobia, a common disorder where a person feels stressed by being around other people.

The users had to interact with several VR scenarios, such as walking down the street, entering a party, or having a conversation in a group of people. By the end of all sessions, the participants noted that their social anxiety had disappeared or dropped to a minimum (decreased 75% on average), improving their way of life and changing the way they think. On average, only 21 minutes were needed to complete one session; the improvement was shown after seven sessions. Participants also mentioned that the realism of the images was better than expected while no real-life footage was used to produce the VREs.

2.1.3 Smart Watch as a Stress-Measuring Tool

When it comes to the ways of measuring stress levels, there are several approaches, including body sensors, wearable technology, and invasive healthcare tools. Assessing stress levels is a complicated task because people react differently to stressful situations. There are many aspects to be taken into consideration, such as the measures that display the stress level, invasive or noninvasive technologies and the validity of the software and hardware that records the data. A recent study about human emotions indicates that a person's body temperature, blood pressure, and heart rates are changing, rising in all cases, when a person is facing a stressful situation [22].

Today, there are not many tools that would measure the stress levels with all metrics available. However, wearable technology is considered to be one of them. Disregarding the fact of not being able to record blood pressure, skin temperature and heart rate altogether, the smartwatch industry focuses on one of those measures - the Heart Rate (HR). A study on the validation of the Apple Smart Watch shows that 90% of all the heartbeats that were recorded during the testing stages were valid and correct, showing the same results as industrial heart rate monitors [23]. The results stipulate that wearable technology can compete with industrial heart rate monitors on the open market as a tool to measure heart conditions. Furthermore, choosing Smart Watch technology for this project is one of the innovative methods because it has never (to the best of the author's knowledge) been used before with VR to collect health data while the user is immersed in VREs.

2.2 Industry Research

As stated in Chapter 1, there are multiple applications on the market trying to combine VR technology with the healthcare industry, such as surgery simulations, and professional training. However, most of those applications lack the functionality to fully implement the product, to evaluate the impact of VREs. This section describes the products in detail and their potential impact on the market and this project.

2.2.1 C2Care

C2Care [3] is one of the largest products on the market nowadays that helps treating phobias using VREs. Moreover, it provides VREs for functional disorders, eating disorders and addictions, such as tobacco, alcohol or gambling. According to the information on C2Phobia website [3], a part of C2Care, the current selection of VR content is very large, and most of it is implemented using 3D animations (Figure 2.1).

Figure 2.1: C2Care Main Page [3]

To gain access to full content, the user has to pay for the subscription or buy a VR kit from C2Phobia directly. After the registration for the demo course, the user would be brought to their account page where it is possible to launch a VR environment for any specific phobia from the VREs list (Figure 2.2). C2Phobia works with wired headsets, such as HTC Vive [24] or Oculus Rift [25] (Figure 2.3) and mobile headsets, such as Oculus Go [11] and Samsung Gear VR [26].

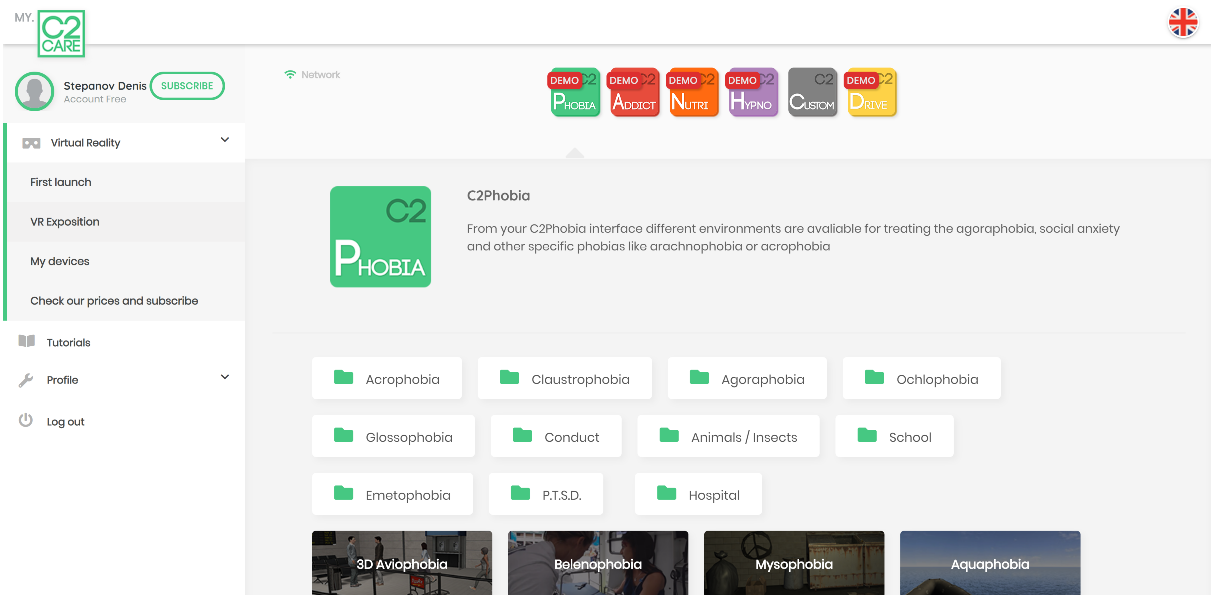


Figure 2.2: C2Phobia Environments Selection [3]

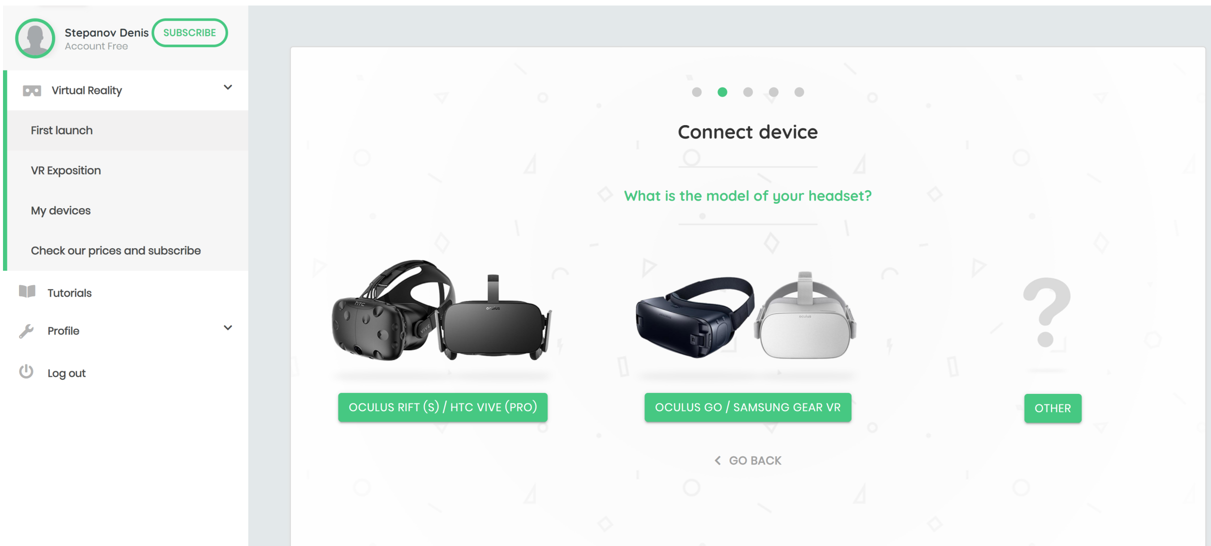


Figure 2.3: C2Care VR Headsets Selection [3]

C2Phobia is one of the largest Food and Drug Administration (FDA) approved systems on the market, developing VR applications in the healthcare industry to treat disorders, addictions, and phobias. However, C2Phobia does not collect any personal data or notify the healthcare professionals about the sessions that the user has taken. C2Phobia works only as a VRE media player for displaying phobias, but it does not evaluate the impact of the VRE. Nevertheless, C2Phobia is still one of the first and largest products on the market that shows the combined technology working between the healthcare and VR industries.

2.2.2 Limbix

Limbix is a product approved by the FDA for researching mental health issues using VR experiences [4]. Limbix built its foundation from the research on mental trauma, phobias, and anxieties, providing a good example of VR technology in the healthcare industry. The Limbix package includes the VR Kit (Figure 2.4), consisting of a VR wireless headset, a tablet, and a docking station to charge and transport the equipment. However, Limbix is not cheap: it costs 2000 US dollars per kit with all features included.

Figure 2.4: Limbix VR Kit [4]

A person cannot register for the Limbix program as the website does not have a sign up option. It is only possible to view the information on the website about the research, product pipelines, exposure therapy environments, and other functionality. Looking at the description of the software from the Limbix website, it was noted that they use real-world images, which is different from C2Phobia. Additionally, Limbix content library has many VR videos available with extra features, such as user interaction during the video and annotation. Moreover, most of the VREs come in multiple languages, which is important to have for users from different countries.

The downside of Limbix is that all of their VR content is hosted on YouTube [27] which is not accessible in various countries due to the restriction policies. Furthermore, similar to C2Phobia, Limbix does not collect any personal data. The Limbix kit is used only to play the VREs as video files.

2.2.3 Qardio

Qardio [5] is one of the best modern ways to monitor users' health activity. Qardio specializes in measuring heart rate, electrocardiogram (ECG), respiratory rate and skin temperature. Qardio has different devices, such as QardioArm, QardioBase and QardioCore, each one for a different use. All of them are focused on checking the user's heart activity and reporting it to the application on the phone. Qardio works for both iPhone Operating System (iOS) and Android, including wearable technology (Figure 2.5). It comes with an application for mobile phones with various features, including storing, sharing and evaluating the data of the user. Qardio also has a special bundle for healthcare professionals with extra features which are approved by many administrations all over the world.

Figure 2.5: QardioArm Kit [5]

Qardio, Limbix, and C2Care are the three main products that are present in the healthcare market nowadays. They are efficient in what they do, but they are lacking some functionality, such as the collection of personal data, live images, wearable technology or VR headsets for mobile phones.

2.3 Resources

As concluded in Section 2.2, most of the VR applications today do not use real-life footage for the VR content. Replacing 3D animations with live images could enhance the effect of reality when the user is immersed in VR. However, due to the time constraints and the lack of equipment, creating VR content for use in this project was out of scope. Therefore, one solution was to use the free resources online.

YouTube [27] provides a large video library of all types, including 360 panoramic VR videos. By default, YouTube videos are covered by Creative Commons Attribution License (CC BY) [28] which allows distribution and display of the work only if the user gives the credits to the author (attribution). Since the version 2.0, all Creative Commons licences require attribution to the creator and include the BY element.

VR applications could include such videos from YouTube with a given accreditation to the authors in one of the pages of the application. Nature and Horror themes VR videos were taken into consideration to be used in the application. YouTube channels, such as National Geographic and Discovery have two of the most largest VR content libraries, thus their videos were considered for the usage in the project.

Chapter 2 has summarized the background information from the related work and available products in the VR healthcare industry today. The next chapter introduces the software requirements specifications for the project and the design of the system.

Chapter 3

Design & Specifications

This chapter describes the software and hardware specifications, developer tools, and frameworks that helped to build the project. Additionally, this chapter presents the design of the system with the description of the components.

3.1 Design

At the early stages of the design process, it was decided to develop two applications for iOS-supported mobile devices. One of the applications would display the scenarios of stressful situations through videos in VR. The second application would collect data from a smartwatch device and use online services to deliver health data to healthcare professionals. The choice was based on the availability of the hardware at the moment and access to services online on different platforms. Considering the cost options, latest news and availability of help resources online, the choice to develop a mobile application for an iOS platform was confirmed. Additionally, it was noticed later that to achieve better results in the the health data collection component (establish a data collection process using wearable device sensors), there needed to be a separate application for watchOS devices, to communicate with an iOS device.

The system consists of two main components: VR application and HR Monitor application, working for the users who will be interacting with both of those components (Figure 3.1). The VR and HR Monitor components are functioning independently from each other but appear at the same interface for the user. The VR component is a standalone application for iOS devices which works on the iOS version 12.0 and higher. HR Monitor component consists of two applications: the Apple Watch application (watchOS 5.0 and higher) and the iOS application (iOS 12.0 and higher).

The VR application was designed to store and display VR videos for the user. The HR Monitor application is responsible for fetching data from the HR sensor on the watch and store it for the user on iOS device.

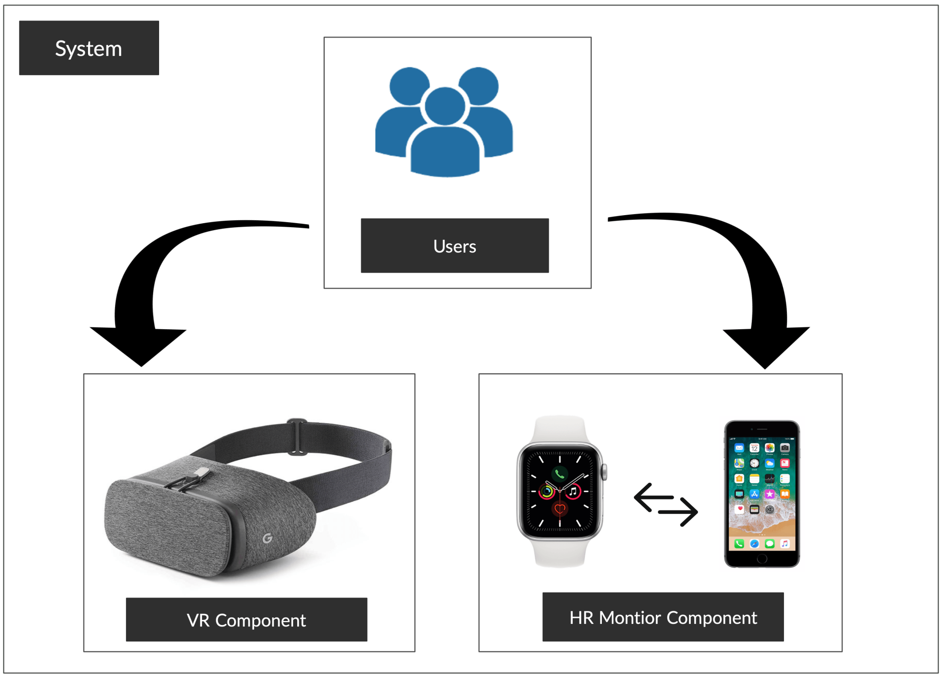


Figure 3.1: System Design

3.1.1 Tools & Frameworks

Unity [29] was chosen as the main tool to create the VR application. Unity is a game engine which works as an Integrated Development Environment (IDE) and produces the prototype of the applications in a short amount of time using C Sharp (C#) programming language and Google VR Software Development Kit (SDK) library. Unity's Profiler [30] was used to get the performance information about the VR application during the implementation stage. A Mac Book Air (macOS Mojave) was used as the machine to run Unity and other software tools.

Xcode [31] was chosen as the main IDE for implementing the iOS and watchOS applications. The main idea behind the design was to create a communication channel between the two devices and establish data collection process. Considering the availability of the hardware, Apple Watch 5 with HR sensor and iPhone X (64GB) were used as the main tools for the implementation. The initial design consisted of only one application for the iOS device. However, later it was decided to create a separate application for watchOS because of sensor privacy issues on Apple's side [32]. The user would have to download both applications to measure, record and share the results with the healthcare professionals.

HealthKit (HK) [33] was used as the main framework to receive the heart rate data from the watch. HK helps developers to create apps which retrieve user's health data (with their permissions), such as HR, workout details, ECG readings, etc. However, HealthKit is not an open source tool. There are also two open source frameworks: ResearchKit [34] and CareKit [35]. ResearchKit is mostly used for creating surveys, taking consent from users and performing various types of research with people (not patients) [36]. CareKit is used mostly for personal use of applications, such as connecting with the patients outside of the regular visits, delivering care plans and tracking daily symptoms [36]. All frameworks from Apple are capable of communication with the sensor on the Apple Watch. It was decided to select the HK framework based on more thorough documentation and examples online.

3.2 Project Requirements

This section states the main requirements for the project. There are two types of requirements: functional and non-functional. Functional requirements define what the system should do, while non-functional requirements specify how the system should behave [37]. The software requirements below are applicable for the implementation of the VR and HR Monitor applications. The software requirements engineering process was completed based on general knowledge in mobile applications development.

3.2.1 Functional Requirements

1. Anybody can start the VR application and use it immediately. Users must give permissions to collect HR data before they can start using the HR Monitor application.

2. Both the VR and HR Monitor applications must be supported on iOS platform (version 12.0 and higher). Watch component must be supported on watchOS 5.0 and higher.

3. The VR Application must use button interaction with gyroscope (Google VR Pointer [38]). The HR Monitor application must include buttons on the screen for user interaction.

4. The VR application must use only landscape mode (VR googles do not work in portrait orientation). The HR Monitor application must support both portrait and landscape mode.

5. The VR application must provide tips page describing the application aims, terms of use and copyright information. The HR Monitor application must include on-screen instructions for the user to start data collection process.

6. Both application must be restricted from storing any data on the local device memory without a user's agreement.

7. The VR component contains video media files. The files must be stored locally and do not exceed the size of 100MB.

8. The HR component contains input fields for the user. Text fields must be checked and validated by the application.

9. The HR component must maintain the connectivity between an iOS device and watchOS device to collect data. The application must restrict any user action if the watchOS device is not paired to the iOS device.

10. The measuring process of the HR component must be available in the background (application turned off or switched to another window). If the user switches to other applications or turns off the screen (locks the iOS device), the application must continue data collection process.

11. The HR component on watchOS must have a time limit to eliminate complete drain of the battery. After time limit is reached, watchOS application must send a signal to iOS device that the limit has been reached.

12. The HR Monitor application must include sharing services, such as email.

13. Both applications contain assets, such as images, videos and animations. Every asset that is not produced by the author must be accredited in the specific About page of the application or in the code.

14. Both applications must be built using responsive design; all assets are positioned the same according to the screen resolutions on different devices.

15. English language must be supported on both applications. It includes user input fields, name of the buttons, descriptions and anything that could contain text.

16. All code for both applications must be documented with clear explanations of the functions, methods and classes that were used to implement the functionality.

3.2.2 Non-Functional Requirements

Usability
Buttons and text fields should be clear and readable. It means that the color of the buttons and the size of the text should be visible on a clear background. Any user should understand the meaning of each button, text field, or figure that are used in the application without any additional explanation. The buttons' names should be self-explanatory and lead to specific scenes or pages with the corresponding functionality. The usability could be measured by the average run time and pause time on each scene to see if the user has difficulties with understanding the context of the application.

Performance
Both applications should load within 5 seconds of clicking the launcher icon. The selection of any button or feature in the application should take at most 2 seconds to load. There should be no frozen scenes or stuck moments during video playback in VR component. The application should run on mobile devices with gyroscope that allow VR view (split screen). If there is a bug or an error with the application, it should be removed with a patch or a fix in the next application release.

Online Access
The device that runs the HR component should have an Internet connection to ensure that the user can use sharing features, such as sending the data collection results through email. However, it is not required to have an access to the Internet at all times as the main part of the application does not require an Internet connection.

Operating Environment
The applications should work on watchOS and iOS-supported devices, exclusively iPhones, version 12.0 and higher, Apple Watch, version 5.0 and higher.

Security
VR application should not store any personal data. It would work only as a video player for the user. However, the HR Monitor application should have permissions to read health data, such as heart rate sensor information. The application should collect the data and send it to the mobile application of the healthcare professional with user's personal details. The details should include only general information, which could be name, age and possible phobia name. None of the applications should have access to personal data unless the user gives permission to the application before the start.

Safety
The application should show safety tips for VR usage during the first run to provide a secure environment for the users. The application should prevent the start of any video unless the user gives consent and reads the safety tips.

3.2.3 Priority & Risk

During the software requirements engineering process, it is important to know which tasks need to be implemented first, or prioritized over others. It is a common practice to evaluate all software specifications and place them in order of importance [37]. However, almost all requirements have their risks, a probability that the specified requirement will not be completed on time. Each requirement has a chance of failure based on either time, resources or any other factor that the developer usually does not anticipate. Therefore, a developer should produce an estimate chart where they know what tasks should be completed first. The value of loss probability could be measured by common practices and previous work in the related field. For example, implementing the menu for mobile application on Android platform could take longer than iOS depending on the programming skills of the developer. According to the time estimates from previous work on building menus for Android application, the developer should check the timeline for the current project to build the menu for iOS. The developer should then calculate the probability of a loss for building the menu, the chance that the task will not be completed on time.

Table 3.1 specifies which most important tasks and chances of a specific task failure during the development stage. Priority is ordered from Low to High. Loss Probability is ordered from Low to High, where Low means that functionality is unlikely to fail and High is most likely to fail.

ID	Requirement	Priority	Risk of Functionality Loss
FR01	Application Start	High	Low
FR02	Device Support	High	Low
FR03	User Interaction	High	Low
FR04	Application Orientation	Medium	Medium
FR05	Application Description	Medium	High
FR06	Data Storage	High	Low
FR07	Application Media	Medium	Medium
FR08	User Input	Medium	Medium
FR09	Device Connectivity	High	Low
FR10	Background Mode	Medium	Medium
FR11	Time Constraints	Low	Medium
FR12	Services	Medium	Medium
FR13	Copyright	High	Low
FR14	Responsive Design	Medium	High
FR15	Language	High	Low
FR16	Documentation	High	Low

Table 3.1: Priorities and Risks of Functional Requirements

3.2.4 Modified Requirements

As the project progressed, some of the functional requirements were modified. Requirement 1 was added after the first prototype implementation because the author was unfamiliar with the documentation of the frameworks and data retrieval. Requirement 4 was modified to only have portrait mode for the HR component. The decision to remove a landscape mode was made after the first prototype implementation to save time. Requirement 7 was added during the testing phase of the VR application. The author has noticed that the files that are larger than 100MB are freezing the screen and do not render at all. Requirement 8 was modified and set to higher priority to ensure the quality of the user's input. Requirement 9 was added after the first testing results on iOS application. The measurement process would not start unless both applications are installed and launched. Requirement 10 was added and set to higher importance because it was noticed during the testing stage that the measurement process stops if the user exits the application. Requirement 11 was added as a result of testing the HR Monitor application. Using HR sensors drains the battery three times faster. Requirement 13 was modified to include animations.

There was no change in the non-functional requirements because they were set as a standard for the author to follow through all the project development process from the start.

This chapter presented the software and hardware specifications, as well as the developer tools that helped to build the project. The next chapter shows the implementation steps to create the VR and HR monitor applications.

Chapter 4

Implementation

This chapter specifies the implementation steps of the VR and HR monitor application development based on the software requirements specifications (Sections 3.2.1 – 3.2.2) and design plans (Section 3.1). This chapter also covers the hardware and software preparations, choice of environment and tools, application features and testing process.

4.1 Software & Hardware Preparations

Based on the selection of hardware and software from the design (Section 3.1), the choice was to implement both applications on iOS and watchOS devices to retrieve sensor data. Figure 4.1 shows all the hardware components that were needed for a successful implementation of the project. An iPhone X (64GB) and Apple Watch 5 were chosen as the main testing devices, MacBook Air as the main machine for writing scripts and application compiling, and Google Daydream 2 as the VR headset to help interact with the VR application. It is worth noting that the VR headset and headphones are not required to build an application and only are needed during the system testing phase. Moreover, any phone with the gyroscope can be used for the VR application. It is also possible to run the application on the emulator, but for VR application development, testing on emulators is not recommended because emulators do not have a gyroscope.

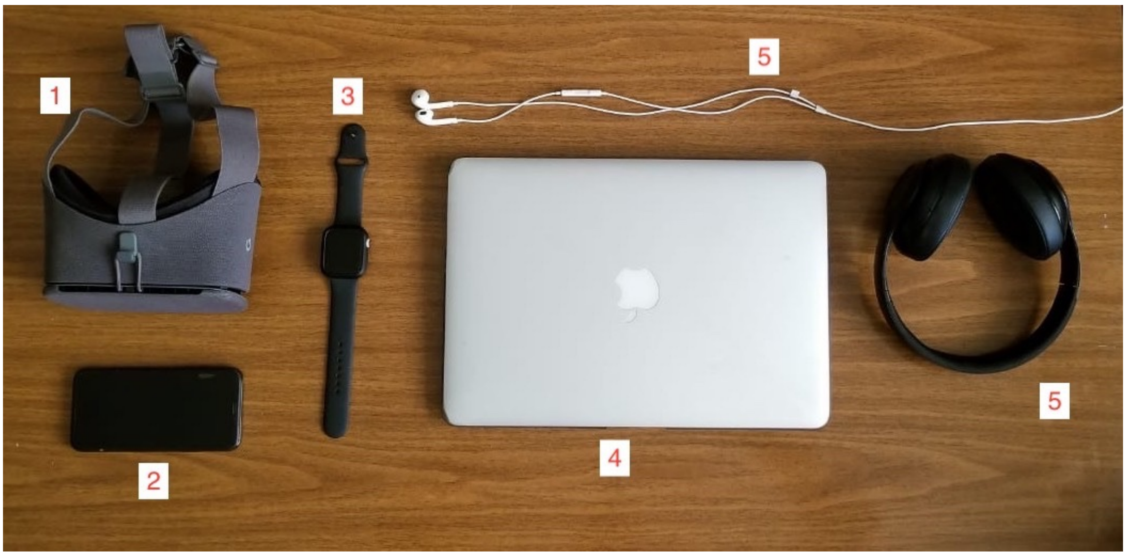


Figure 4.1: Hardware Components

For the software part, it was decided to use Unity tools and Google VR library because they have the code samples and tutorials on how to build VR applications. Xcode [31] is available on any macOS device as a free application from the App Store. It is also recommended to update all software and hardware to the latest version available before the start of the project. For additional information about tools specifications, refer to Appendix B.

4.2 VR Application

As discussed in Section 3.1, the VR application is responsible for storing and displaying VR videos. All activity inside the application is managed by the head movement and the gyroscope on the mobile device. All interactions around the application were implemented with a focus function, such as Gaze [38]. Gaze is a method of interacting with objects in VR using cursor focusing: a user looks at the object with the cursor that is located in the middle of the headset and can perform various actions with it (the code implementation of Gaze function is available in Appendix A). In the example of our application, the user has four options at the application start: tips, VR video gallery, about us page, and quit (Figures 4.2 & 4.3).

Figure 4.2: User Interface PC View

Figure 4.3: User Interface VR View

The buttons on the screen are linked to C# scripts to switch the windows of the application. For example, the VR video gallery button will lead to the page which will consist of VR-ready videos (Figure 4.4). A button with an image will lead the user to the video. (Figure 4.5).

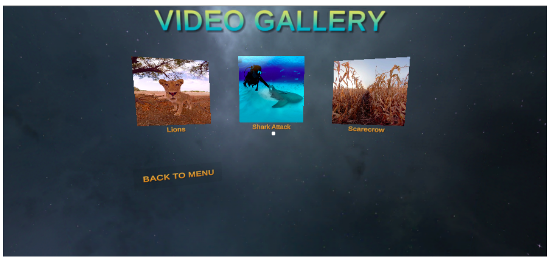


Figure 4.4: VR Video Gallery

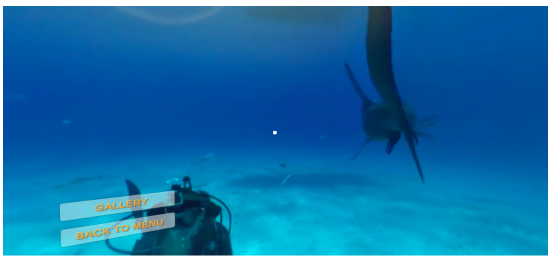


Figure 4.5: Shark Attack Video

Currently the application includes three videos. All videos are stored locally on the device inside the application storage. As the testing phase would show later (Section 4.2.2), the size of the videos was affecting the run of the application. It was noticed that the videos larger than 100MB were not rendering and crash the screen upon loading. The solution was to store the videos online and use the application as a method to access those videos. GitHub [39] and Dropbox [40] were used as hosts, but did not work well. Accessing the videos online restricted the user's ability to use the application offline. Moreover, accessing videos online drastically reduced the quality because of the media players that are hosting them. The quality only goes up if the Internet connection is higher than 2MB/s which might not always be accessible for the user.

Another solution was to restrict the videos to only 100MB or less. There were no noticed interruptions during the implementation run and all videos were stored locally without any problem. The current size of the application is around 1GB, which is large, but acceptable as most mobile devices nowadays have a large storage capacity [41]. Videos were edited and cut to one or two minutes in length to reduce the VR session time to a minimum. The tips page in the VR application main menu also shows the recommended VR usage time.

4.2.1 Feedback

The VR application was tested only by the author as it is not permitted to let others use it at this stage due to the type of content in the application. The author has received only general feedback about the application from the neurologists and one physician, who approved the VR application as a possible tool to immerse a user into the VRE of their fear to evaluate the impact. Testing of the VR application was done in the presence of a neurologist, Dr Olga Philipchenko, and one physician, Dr Eugeniia Zhukova (Figure 4.6).

Figure 4.6: VR Application Testing

4.2.2 VR Application Testing

When the main portion of the VR application was finished, the author began to test it on various headsets and mobile devices. Due to the lack of data structures, loops and conditionals in VR applications, VR testing is inefficient using standard methods [42]. Therefore, it was decided to apply the Fault-Based approach [43].

There were two known issues with the VR application, all based on its performance. First, the application responds faster or slower depending on the video file sizes. It was noticed that the program freezes or crashes when large video files were included. Second, the VR application responds differently based on the device that it was run on. Due to the time constraints and the limitations from the developer (no access to the headsets), VR application testing was postponed to future work (Section 6.2.2).

Testing on Different Headsets

The application was tested on an iPhone X with iOS version 13.2. Five headsets were used: Google Cardboard [44], Google Daydream 2 [10], Merge VR [45], Park VR (non-brand copy of Google Cardboard), and XiaoZhai Z6 [46]. The headsets were chosen based on different prices, online reviews in their price ranges, and overall grade by the companies in the VR industry.

Since the headsets that were used for testing only work as holders for mobile devices and do no interfere with the performance of the application, it was decided to split the ranking into several categories, such as quality, comfort, and included controllers. The choice of categories for ranking was based purely on the developer's opinion. Quality ranking is based on the headset image quality. Comfort ranking is ordered from the most comfortable headset to use while seated, standing and moving around. Finally, the last category shows whether the headset has native controllers (hardware components to help the user interact with the VREs). Table 4.1 summarizes developer's satisfaction with using each headset model during testing phase of the VR application. Likert rating scale was used to rank each headset (terrible, below average, average, above average, and exceptional) [47].

Headset Model	Image Quality	Comfort	Controllers
Google Cardboard	Terrible	Terrible	
Google Daydream	Above Average	Above Average	✓
Merge VR	Exceptional	Average	
Park VR	Below Average	Below Average	
XiaoZhai Z6	Above Average	Exceptional	

Table 4.1: Headset Testing

Testing with Different Size Media Files

During the first round of the VR application testing, it was noticed that the mobile device does not handle large media files properly. The screen would get stuck at one point or the application would crash. Considering that it is a known bug in the program, it was decided to implement a Fault-Based approach [43] as the next step of the testing process. Five iOS devices were used: iPhone X, iPhone 5s, iPhone 7, iPhone 7+, iPhone XR and iPhone 11. All devices had an iOS version 13.2, except iPhone 5s had the version 12.2 at the moment of testing. The tables below show the iOS devices running the application with different size media files.

Phone Model	Video Interruption	Application Crash
iPhone 5s	✓	
iPhone 7		
iPhone 7+		
iPhone X		
iPhone XR		
iPhone 11		

Table 4.2: Video Size <100MB

Phone Model	Video Interruption	Application Crash
iPhone 5s	✓	✓
iPhone 7	✓	
iPhone 7+	✓	
iPhone X		
iPhone XR		
iPhone 11		

Table 4.3: Video Size <500MB

Phone Model	Video Interruption	Application Crash
iPhone 5s	✓	✓
iPhone 7	✓	
iPhone 7+	✓	
iPhone X		
iPhone XR		
iPhone 11		

Table 4.4: Video Size 1GB - 2GB

Phone Model	Video Interruption	Application Crash
iPhone 5s	✓	✓
iPhone 7	✓	
iPhone 7+	✓	
iPhone X		
iPhone XR		
iPhone 11		

Table 4.5: Video Size >2GB

By the completion of the first test runs, it was concluded that the large size video files are not handled well by mobile devices that are older than iPhone 7 (Release date: September 2016).

Google Daydream 2 [10] is the best option for application development today, as it is the best overall headset available which has documentation online. XiaoZhai Z6 [46] is an efficient runner-up which will be used for future application development as a second testing option.

4.3 HR Monitor Application

HR Monitor is a tool to measure the heart rate of the user by acquiring data from the HR sensor on the Apple watch. The HR monitor application has several features, such as measuring and displaying heart rate, the ability to record user's name, age and phobia name, and sharing the results through email or other services. This section covers the information about Apple Watch sensor, permissions, implementation steps and the UI design of the application.

4.3.1 HR Sensor

Apple Watch has a HR measuring sensor. It is located on the back of the watch and consists of multiple green LEDs which are responsible for measuring HR of the user. By default, the sensor is not directly accessible, and the only way that the developer could retrieve the information was by using frameworks that Apple provides (Section 3.1.1). Additionally, there are other complications with the sensors. On Apple's website, it says the sensor is only measuring the HR of the user when it is changing [48]. It means that the embedded application Heart Rate on the watch is not directly accessible either. To solve this, Apple recommends to use Workout Mode, which enhances sensor measurement and gives access to the data from the watch to the developer. Sensor data is updated faster based on time, not on HR change. Therefore, it was decided that the HR monitor application should include two components: a watchOS application which enables Workout Out mode and tracks the user's HR, and an iOS component which is used as a signal sender and receiver to acquire and display data from the watch.

4.3.2 Security

On the first launch of the application, the users are prompted with a permission screen (Figure 4.7) where they would need to agree on collecting HR data for the application. Considering that the heart rate data is linked to the HealthtKit framework, there was a way to implement the security of the application by checking the permissions that were given to the application. `HKAutorizationstatus` is a method to determine the permissions status of the application on a particular type of permissions of health data (heart rate, height, weight) [8]. Table 4.6 summarizes the three states of `HKAuthorizationStatus`:

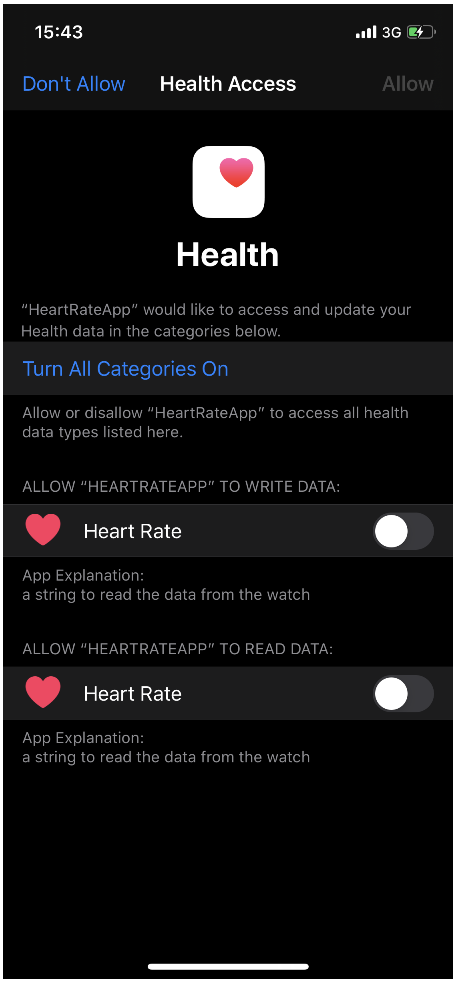


Figure 4.7: Permission Request

Authorization Status	Description
notDetermined	The user has not yet chosen to authorize access to the specified data type
sharingDenied	The user has explicitly denied your app permission to save data of the specified type
sharingAuthorized	The user has explicitly authorized your app to save data of the specified type

Table 4.6: Authorization Status Types [8]

Further investigation and testing of the HR monitor application uncovered a presumed security bug in the permissions check execution on watchOS devices. The bug was found during the `HKAuthorizationstatus` unit testing. Rather than indicating the actual state, this method consistently returns a condition of `shraingDenied` (even when the user agrees to the permissions). Regardless, the functionality appears to work as intended, with the app accessing the data if the user has given permission, even though the `HKAuthorizarionstatus` method returns a denied status. Due to the time constraints, and the currently working functionality, full examination of this suspected bug has been postponed to future work.

4.3.3 HR Monitor UI Design

Based on the knowledge of the UI design in mobile applications and lack of experience working with the Swift programming language, the first application prototype was built poorly without following any set of UI design rules or principles [49]. Additionally, it was noticed that as soon as the Apple Watch screen changes to a sleep mode, the measurement process would stop on the iOS device. The solution was to rebuild the component and include a separate watchOS application to implement continuous measurements for different application states (background mode support). This section summarizes the first and latest prototypes, their differences and similarities.

Initial Prototype

First prototype consisted of many objects, such as buttons, various frames and text view to display messages (Figures 4.8 & 4.9). Based on the author's inexperience, it was not clear at the moment what functionality on the screen should be abstracted, removed from user's view. The prototype was built without following any specific rules on UI design, but mostly focusing on the functionality of the application.

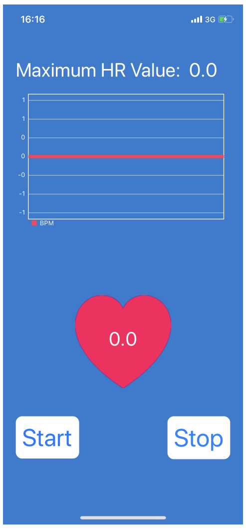


Figure 4.8: Initial Prototype Launch Screen

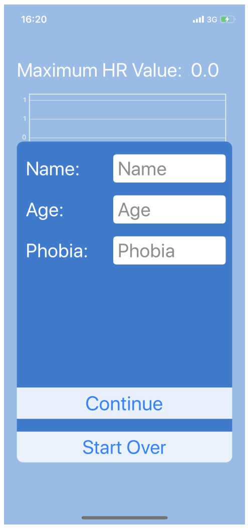


Figure 4.9: Initial Prototype User Input

The idea behind the main functionality (data collection process) of the application was to establish a communication between the watchOS and iOS devices. The WatchConnectivity framework [50] was used to implement the communication and to test it between the devices.

The UI of the application was implemented using standard tools from Xcode [31]. Initial prototype included three main classes: launch screen, user input, and the results page. Launch screen was responsible for the permissions check, measurements process, and presenting the current heart rate in form of a graph on the screen. User input class was implemented to validate the user's information. Results page was responsible for displaying the information that was recorded throughout the measurements process and user input stage. Results class also has a feature of sharing the information through the channels online, such as email.

Latest Prototype

After a month of implementing the HR monitor application, it was concluded that the design could be improved by following the standards from Apple guidelines on human-computer interaction [51]. The latest application prototype had the same functionality as the first one, but the UI was modified. The background color was changed to darker, Start and Stop buttons were reduced to only one with changing state on push, and text labels were aligned better across the screen (Figures 4.10, 4.11, 4.12 & 4.13).

Figure 4.10: Latest Prototype Screen #1

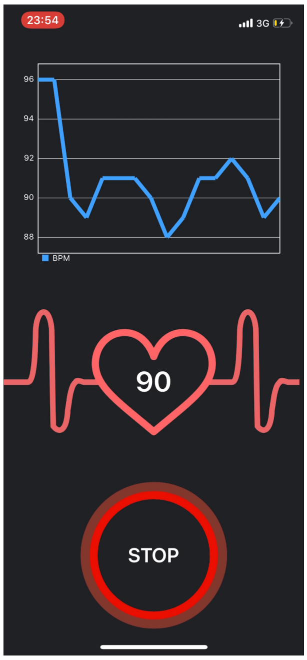


Figure 4.11: Latest Prototype Screen #2

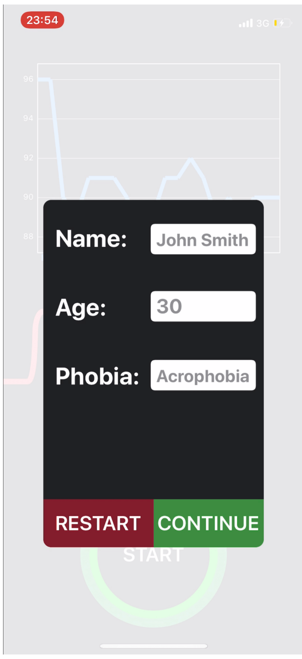


Figure 4.12: Latest Prototype User Input

Figure 4.13: Latest Prototype Results

The application was restricted to only portrait orientation as there were difficulties during the landscape orientation implementation and design responsiveness. Responsive design is an approach to design that makes objects on the screen render well on a variety of devices and window or screen sizes [52]. Results page was one of the examples that had to include the responsive design technique because it has a feature of sharing the results information through online services, such as email. Attaching the image of the chart on different devices requires dynamic rendering, the image location had to be captured on the screen for a snapshot and later attachment in the email. Considering the fact that different screen sizes on iOS devices affect the image allocation, and the lack of documentation from iOSCharts [53] library, responsive technique had to be implemented. Figure 4.14 shows the example of an email with the correct attachment of the chart image. Additional responsive design details and the corresponding code samples are described in Appendix D.

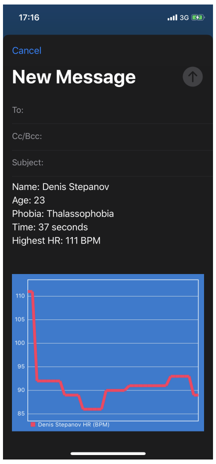


Figure 4.14: Sharing Results Through Email

4.3.4 HR Monitor Application Testing

The XCTest [54] framework, provided by Apple, allows developers to create and run unit, performance, and UI tests for any Xcode project. The XCTest was used to perform all tests for the HR monitor application.

All unit tests were categorized by classes, specifically appointed to the functionality that was tested in the application. There were two major classes that were tested with XCTest: user input and UI responsiveness.

User Input

The HR monitor application includes only three possible user input fields. They are all located in the user input window where the user has to enter the information, such as the name, age and phobia. All input is restricted by regular expressions and shows a specified error message if the user violates the rules of the expression. The rules for the user input were not taken from any outside source. They were created purely from the developer's opinion on the common standards for input error-check implementation.

There were approximately twenty test cases for each input testing on boundaries, special characters and white space location in the string of the test input. The application shows a specified error message if any of the requirements were violated and tells the user what needs to be fixed. However, XCTest test cases found an issue with regular expression syntax. The boundary check was set incorrectly and did not show an error message when the user enters a string exceeding the allowed character limit. The error was fixed and noted in the test class.

UI Testing

UI testing was performed using same XCTest framework. Considering that responsive design was implemented (Section D.1), and most of the objects on the screen are constrained programatically, UI testing cases were simplified. The goal of UI testing is to check whether all objects on the screen are accessible and visible.

One of the features by Apple in UI testing is that it allows screen recording of the user action with conversion to Swift language statements. For example, a tester can start the recording, click on the available object on the screen and the code will be generated inside a test file, specific to the object that was used. The recording tool is very helpful because the tester can create various test cases to check whether the UI is working as intended based on the code that is generated by the screen recording tool. The UI was tested in three different classes: main class which displays heart beat animation and a measuring process button, user input class which controls the user data validation, and the results class which displays the information to the user with the control buttons. It is important to note that testing HR monitor application on the emulator is not efficient. The sensor on the watch is a hardware component which is not accessible on emulators. In cases like this, it is recommended to test the UI on an actual device. The downside of this is that a tester is limited to the physical devices that are available at the moment.

It was challenging to test all three classes on a physical device. Delays in communication between the watch and the mobile device break almost every test case and do not record correctly the user-recorded steps to get to another class of the application. Nevertheless, test cases were created to simulate a walk through the application including the start of the measuring process, imitation of user input and checking the button actions.

The UI testing class is included in the application and can be accessed by running the test class itself (HeartRateMonitorUITests.swift). Challenges with the hardware communication showed that there is a need for a better way to test the UI of the application in the future.

This chapter specified the implementation steps of the VR and HR monitor applications development based on the software requirements specifications and design plans, including the application testing. A user manual describing the installation steps and how to interact with the VR and HR monitor applications is included in Appendix E. The next chapter outlines the project management and reflections on the project development.

Chapter 5

Evaluation

This chapter discusses the the time management, critical assessment of the progress and the feedback on the applications.Considering the current situation with COVID-19 pandemic around the work, this section was updated according to the latest news.

5.1 Project Management

Project development started with collecting the necessary hardware and software tools (Section 4.1). Then, the implementation of the VR application to display phobia(s) and Smart Watch application to collect data. The development of the applications required the most of the time because it required a lot of time to write code and get familiar with the documentation. Most of the time during the project development was spent on learning the new programming languages and writing the code for the applications. Testing and evaluation also took a significant part of project development process.

The Gannt chart below (Figure 5.1) describes the original project plan. Using a Waterfall methodology [55], I planned to conclude testing on each major stage of the software implementation. Some of the tasks were combined due to the importance levels, risk management and wait time for other various tasks that do not depend on the developer, such as ethics forms approval.

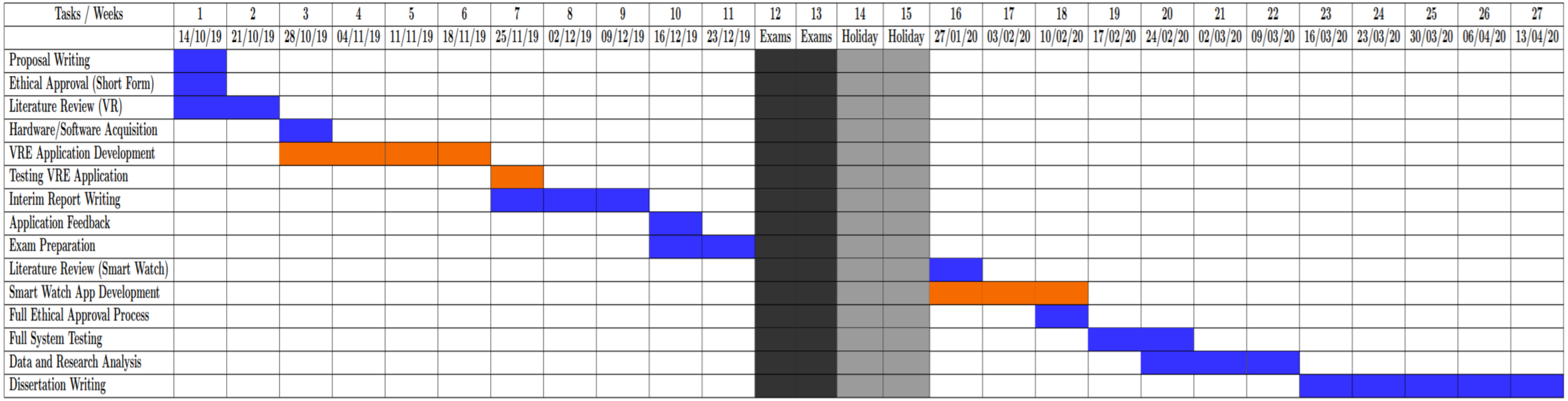


Figure 5.1: Original Timeline

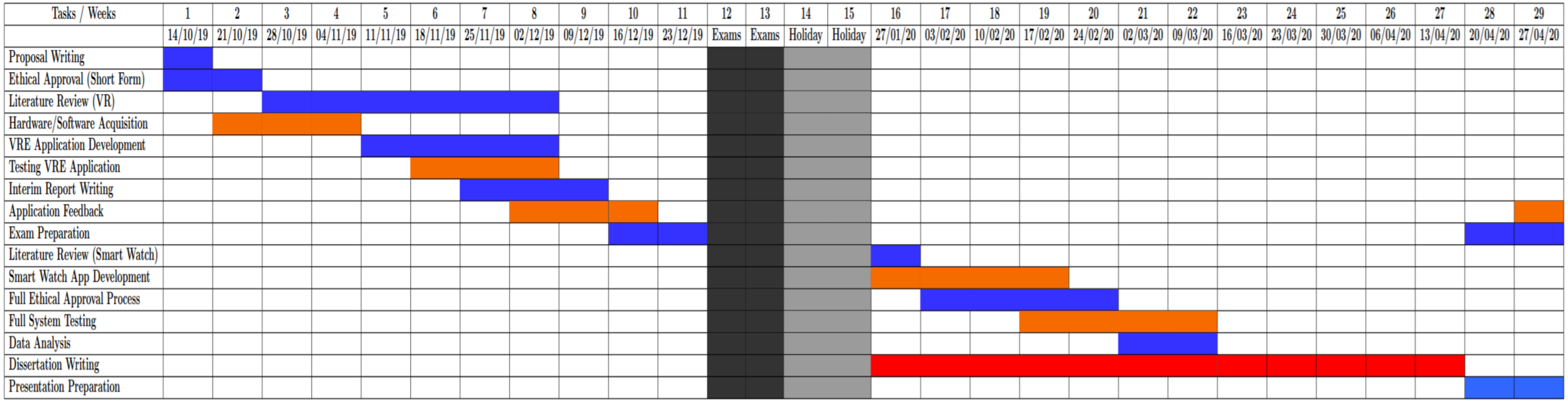


Figure 5.2: Updated Timeline

However, several changes were made after the first ten weeks of project development. The Waterfall methodology did not seem suited at the time and was replaced by Agile [56]. Most of the tasks ran in parallel with each other and did not depend on the completion of the processes before them. The implementation of the VR application was done alongside the testing on each phase of development. The dissertation writing part was not assigned properly before because most of the research progress was done during the first weeks and continued through the whole timeline of the project. Therefore, the restructured format and timeline were applied to the new chart with adjusted predictions for the second half of the year in Figure 5.2.

In the beginning of the second semester, COVID-19 pandemic [57] [58] started to spread around the world. Therefore, the updated timeline was not valid anymore. The sudden change in the curriculum has seriously impacted on the students' work efficiency, communication, and psychological factors [59]. Considering our university and the lockdown in China, the date of the project completion appeared to be unpredictable at first. All deadlines were moved more than once throughout the second semester, thus creating a hectic atmosphere for the students. Nevertheless, the duration for the tasks in the updated timeline stayed the same, with all tasks managed on time, adjusted to the changing curriculum schedule.

5.2 Project Appraisal

When the application development and testing phases were finished, it was decided to collect feedback from the user and the developer based on the HR monitor application. As discussed before, the VR application was not permitted to be tested by others due to the content type, thus the feedback was limited to a verbal review from the healthcare professionals who were present during the testing phase.

It was planned to collect the feedback from two channels: conduct an online survey with the participants from UNNC and gather the information from TestFlight [60] community.

5.2.1 Survey

Qualtrics [61] is an online platform to create questionnaires. Considering that the survey would be available to only UNNC students, it had to be stored on a private server with a restricted access. After approval from the Ethics Committee on the forms, the survey about the design of the HR monitor application was published.

The survey included eight questions about the initial and latest prototypes of the HR monitor application. It was important for the author to get the details from the users about the design of the application and improve it, if possible. Therefore, all questions were targeted specifically at the presentation of the application.

The survey is available until the end of the August 2020. It does not include any private information about the participants. Every person who takes part in the survey must accept the consent form and read the participant information sheet. The full survey with consent form and participant information sheet are included in Appendix F.

Currently, the survey includes 108 replies, three of which were not completely finished. There is not a lot of variation in the answers, especially for short answer questions where the participants were asked for additional ideas or advises. Considering, that the survey was focused on getting feedback about the changes of the HR monitor application design, most of the answers were positive about the new design over the initial prototype. The answers of multiple choice questions were ranked in five categories following Likert [47] rating scale: 1 - terrible, 2 - poor, 3 - average, 4 - good, 5 - excellent. 75% of the participants rated the initial prototype terrible or poor. However, more than 95% of the participants rated the latest prototype 'good' or 'excellent'. The open form questions were focused on gathering additional information and asking for new ideas from the participants. Unfortunately, all the answers were either empty or stated that nothing new is required. The detailed answers report is available in Appendix F.1.

5.2.2 TestFlight

TestFlight [60] allows Apple developers to publish their beta apps for testing and get feedback from anybody who has valid Apple ID before submitting the application to AppStore. TestFlight allows developers to include multiple testers at the same time. The invitation for downloading the application and testing it must be sent to the tester's email. Considering that the HR monitor application includes both iOS and watchOS component, there was a lack of testers who I knew at the moment to run the application.

One solution was to use online communities and forums, such as Reddit [62]. Reddit is a network of communities based on people's interests. r/Testflight [63] is a community for iOS developers and testers all around the world. People can submit their apps to be tested or present themselves as testers and state their information (device and version).

The community is private, people need to apply for membership to join it. TestFlight has 6800 iOS developers and is one of the fastest and easiest methods to get feedback on the application by other developers. After sending a request for membership approval, any Reddit's registered user can write a draft post (Figure 5.3) for receiving feedback on their application.

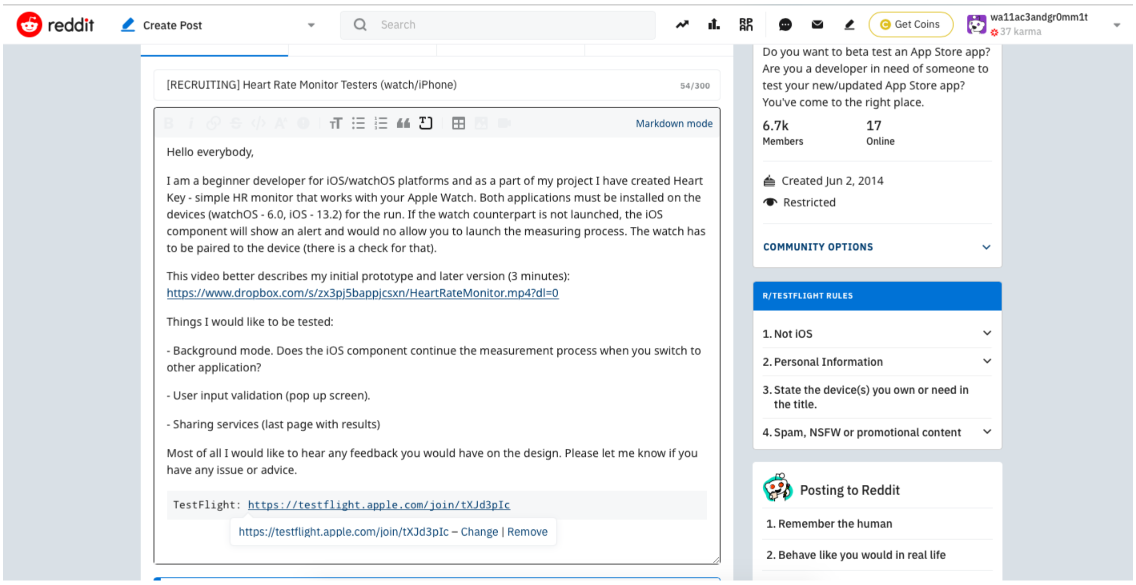


Figure 5.3: Feedback Post Draft

However, even after applying three times, my request was rejected without any explanation and the feedback was not gathered on time.

This chapter has described the time management and introduced the initial and updated timelines. It had also outlined the evaluation of the work that has been done with the user's feedback on the HR monitor application. The next chapter summarizes the project, outlines considerations for the future work and states the reflections of the project.

Chapter 6

Summary

This chapter outlines the reflections on the project, self assessment, and future considerations for this project in terms on new features and functionality.

6.1 Reflections

In the beginning of the project the goal was clear and straightforward. Due to the lack of background knowledge on the subject, I have encountered multiple issues with the project time management and software utilization. However, thorough research and literature review helped during the implementation stages and allowed me to overcome the problems in the beginning of the project.

In the second half of the year, the scope of this project increased. Learning a new programming language and ensuring the functionality of the applications under a specific environment, as well as gathering the feedback from the user, was a challenging goal. Additionally, the level of project difficulty was increased because of those functionalities. Therefore, required measures were taken, such as revised time management and change of software engineering methodology. Moreover, the situation with COVID-19 pandemic has affected the work of this project. VR application testing on different headsets was not accessible anymore, thus testing was postponed to future work.

Nevertheless, the speed of the project development was good and the deadlines were strictly met by all deliverables submitted on time. The work that was done during the first weeks covered most parts of the related field research and later followed through to the application development stage based on the information that was collected. However, considering the timeline that was set at the beginning of the project, there were difficulties during the software and hardware installation. The time that was taken to solve the problems seriously affected the work and the timeline of the project development.

In the end, the VR and HR monitor applications were implemented on time and work as intended. All applications were tested, received feedback and satisfied the requirements that were proposed in this project. It is now important to continue maintaining the system and consider possible future functionality that could improve the project.

6.1.1 Self Assessment

This section explains my thoughts about the project and my personal evaluation on the progress throughout the year.

I think that most of the work was done successfully and the goal of the project was reached. The system of two applications that are working independently from each for the user to display VR videos and measure users' HR is completed. The problem that I faced is the overflow of the material and misleading in the things to focus on. In the beginning of the project, it seemed to me that everything was in order, what tasks need to be done and when. However, with the time and progress passing by, I started to notice that the topic that I am covering is very large and needs to be constructed better, focus on a specific, smaller idea. Currently, I think that the system is working well and includes all the features that were included in the system specifications. However, it is not perfect. Due to the time constraints, I did not implement all testing that I wanted to do in the beginning of the project and I have not received enough feedback to completely understand what others think of the application. Nevertheless, this project has been a great experience for me to improve my critical thinking skills, coding, presentation and writing.

6.2 Future Work

This section describes what could be added in the future work to improve the system and what are the next steps for this project.

6.2.1 Sensors

A heart rate sensor might not be enough to measure the stress level of the user. However, it is enough to see the impact on the users, if they are experiencing discomfort [23]. This section lists potential sensors that could be used in the future to help in evaluation for healthcare professionals.

The sensors that provide the most accurate results due to minimal body part movement and number of receiving parts are: EEG, GSR (EDA - electrodermal activity) and Eye-Blinking Rate [64] [65].

EEG

One of the ways to measure the stress level is to look at the brain activity. An electroencephalogram (EEG) is a test used to evaluate the electrical activity in the brain. It is mostly used to measure abnormal activity of the brain by the healthcare professionals. It is a large machine with a lot of wires connected to the head sending signals to the computer and displaying the brain activity as waves in the applications (Figure 6.1). However, EEG tools have been evolving and becoming more compact, such as Muse [66] and EMOTIV [67]. Muse is a tool for meditation that reads the emotions and converts them to sounds that might simulate those emotions (Calm mood - light wind and good weather sounds, Stressed - storm and lighting sounds). EMOTIV is a portable EEG device with a smaller amount of sensors and comes in a variety of sizes and shapes.

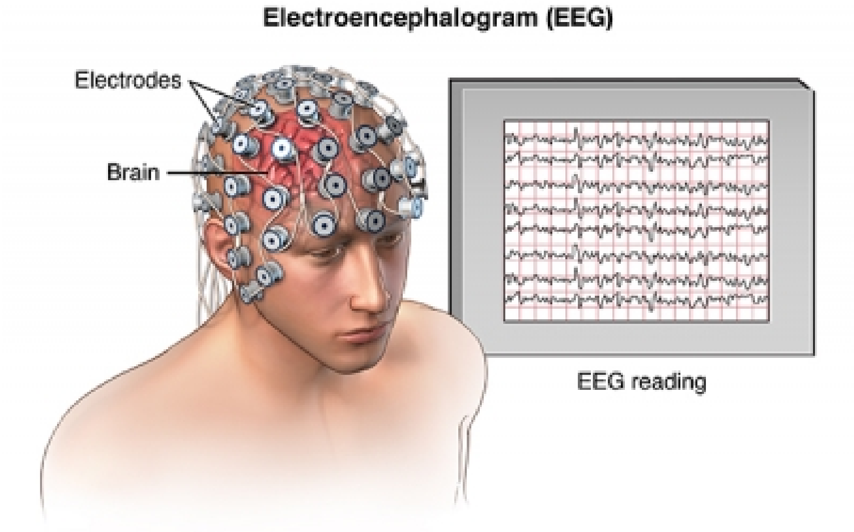


Figure 6.1: EEG Tool [6]

EEG is a good tool for measuring stress level and brain activity. However, the size of the tools and the high price (over 1000 US dollars) makes them hard to acquire right now. Therefore, it could be a good addition to the project in the future.

GSR

Galvanic Skin Response (GSR) is a measure used to understand how intense is the specific emotion of a person. GSR refers to the changes in sweat gland activities. Our emotions change based on the environment that we are in — sometimes calm, angry, or stressed. Every single emotion has an intensity level, and when a person experiences higher or lower level of a specific emotion intensity, the activity in sweat glands are changing too [68].

The average price of a GSR sensor (Figure 6.2) varies between 20 USD and 100 USD depending on the quality and the amount of receiving sensors. However, GSR sensors are not completely robust — there needs to be a connection to some kind of micro-controller, such as Raspberry Pi or Arduino Uno. GSR would also need a Bluetooth module to successfully pass the data to a mobile application.

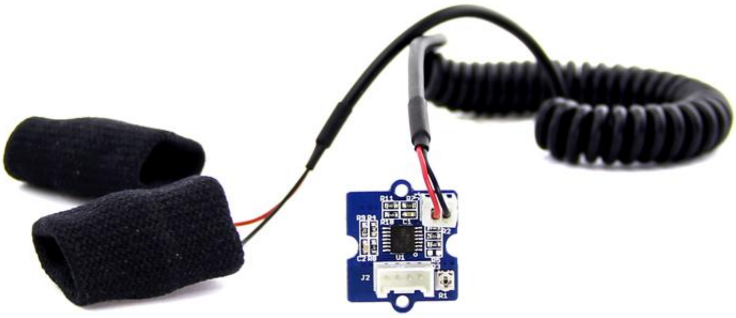


Figure 6.2: GSR [7]

Adding GSR sensors would the best option for the next implementation step of this project. GSR sensors are cheap, compact, and have a lot of documentation online.

6.2.2 Additional Testing

The testing process of the VR application (Section 4.2.2) is based on known faults and bugs in the applications. It does not cover new errors detection and thus it needs improvement. Due to the time constraints, the testing phase of the applications was not complete and requires more time in the future [2].

VR software may be considered to suffer from the Oracle Problem [69], regardless of the current situation with the VR application testing and the methods that were practical in this project. During testing, an oracle is a mechanism used by testers to determine the correctness of software behaviour or output [70]. The lack of practical oracle in VR applications create an Oracle Problem which is difficult to solve using traditional testing methods. Metamorphic Testing (MT) [71, 72] is a Software Quality Assurance (SQA) approach that has been proven to alleviate this problem [73]. MT uses Metamorphic Relations (MRs) among multiple executions of the program to help the tester identify problems. A violation in MR would show the tester a presence of a bug in the program. Inspired by MT, two MRs were produced to include in the VR application testing [2]:

- HMR1: After a first run of the application with a video of file size X (e.g., 10 MB) ($input_x$) as input, a second run of the application with a larger file Y (e.g., 50 MB) ($input_y$), should take longer to launch the video. Essentially, assuming no other changes to the application or its inputs, other than the video file size, then:
  $filesize(input_y) \geq filesize(input_x) \vDash$
  $timetolaunch(input_y) \geq timetolaunch(input_x)$
- HMR2: Launching the app on a device with more Random-Access Memory, RAM

should take less time. After a first run of the application on any device (e.g. iPhone X (3GB RAM)) ($device_x$), a second run of the application using the same content, but on a device with less RAM (e.g., iPhone 5S (1GB RAM)) ($device_y$), should take longer to launch the video. Essentially, assuming no other changes to the application or its inputs, other than the hardware used to launch it, then:

$$RAMsize(device_y) \leq RAMsize(device_x) \vDash$$
$$timetolaunch(device_y) \geq timetolaunch(device_x)$$

As discussed in Section 4.2.2, testing according to HMR1 revealed a problem when compiling large video files. The application crashed when using the files larger than 100MB. Additional exploration showed that the crash appears sooner if the video file size is larger. However, using MAC emulator to test the VR application seems to solve the issue; the application does not crash, regardless of the video file size.

To the best of the author's knowledge, there are no current applications of MT for VR application, thus it is worth investigating and include MT as a part of testing method for this project in the future.

6.2.3 Technology Combination

One of the main goals of this project in the future would be to combine the two applications into one. The VR application could handle both interaction with VR video library and the wearable device sensor's data collection. Combining the two applications would simplify the system and require only one application download. However, the downside would be removal of the results sharing option. The user will not be able to type their name or age while immersed in VR because the lack of the physical keyboard. There are several approaches to solve this issue, but due to the scope of the project, this problem will be considered in the future work.

6.2.4 Open Educational Resource Goal

Open Educational Resources (OERs) are teaching materials that are either in the public domain or released under a license that allows them to be freely used, changed and shared [74]. Considering the current state of the project, it is anticipated to eventually be released to the OER community [2]. The steps to make it work would be publishing a license using Creative Commons (CC) [75] tools and removing all copyrighted materials in the project, such as source code or media files. Currently, there is no outside source code material, but the VR media files were taken from YouTube [27]. As discussed in Section 2.3, those videos are covered by the CC BY license, thus they do no possess any challenge and can be used in the project with a proper accreditation to the authors. Therefore, the system that was created in this project could be published on any online platform, along with author's created CC license, and presented as an OER in the future [2].

This chapter has summarized the reflections, self assessment, and the future work for this project. All in all, I am very pleased with the outcome of this project and the system that was implemented. Everything that was planned at the start of the project development has been achieved. The system with its components is working well and has a potential to become an OER platform to help healthcare professionals in phobia evaluation. Regardless of the pitfalls that I had during the project development, I think that the most important part of it was the learning process. This project has given me a lot of valuable knowledge, and I hope to continue working on it in the future.

References

[1] James Walker, Denis Stepanov, Dave Towey, Ahmed Elamin, Matthew Pike, and Ran Wei. Creating a 4D photoreal VR environment to teach civil engineering. In *IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE 2019)*, pages 1006–1013, 2019.

[2] Denis Stepanov, Dave Towey, Tsong Yueh Chen, and Zhi Quan Zhou. A Virtual Reality OER Platform to Deliver Phobia-motivated Experiences, N.D. submitted for publication.

[3] C2Care. Virtual Reality at the Service of Health. `https://www.c2.care/en/`. Accessed 11-April-2020.

[4] Limbix. About Limbix. `https://www.limbix.com/about`. Accessed 24-October-2019.

[5] Qardio. About Us. `https://www.getqardio.com/about-us`. Accessed 24-October-2019.

[6] Bright Brain Centre London. EEG and Brainwaves. `https://www.brightbraincentre.co.uk/electroencephalogram-eeg-brainwaves/`. Accessed 04-April-2020.

[7] Seeed Technology. GSR Sensor. `http://wiki.seeedstudio.com/Grove-GSR_Sensor/`. Accessed 06-April-2020.

[8] Apple Inc. HKAuthorizationStatus. `https://developer.apple.com/documentation/healthkit/hkauthorizationstatus`. Accessed 05-March-2020.

[9] Cipresso, Pietro and Giglioli, Irene Alice Chicchi and Raya, Mariano Alcañiz and Riva, Giuseppe. The Past, Present, and Future of Virtual and Augmented Reality Research: A Network and Cluster Analysis of the Literature. *Frontiers in Psychology*, 9:2086, 2018.

[10] Google. Daydream. `https://vr.google.com/daydream/`. Accessed 18-December-2019.

[11] Oculus. Oculus Go: Standalone VR Headset. `https://www.oculus.com/go/`. Accessed 24-March-2020.

[12] Min-Chai Hsieh and Jia-Jin Lee. Preliminary Study of VR and AR Applications in Medical and Healthcare Education. *Journal of Nursing and Health Studies*, 3(1):1, 2018.

[13] Lan Li, Fei Yu, Dongquan Shi, Jianping Shi, Zongjun Tian, Jiquan Yang, Xingsong Wang, and Phoenix Zhang. Application of Virtual Reality Technology in Clinical Medicine. *American Journal of Translational Research*, 9:3867–3880, 2017.

[14] Mantovani, Fabrizia and Castelnuovo, Gianluca and Gaggioli, Andrea and Riva, Giuseppe. Virtual Reality Training for Health-Care Professionals. *CyberPsychology & Behavior*, 6(4):389–395, 2003.

[15] Ichalkaranje, Nikhil and Ichalkaranje, Ajita. *Intelligent Paradigms for Assistive and Preventive Healthcare*. Springer, 2011.

[16] Srivastava, Kalpana and Chaudhury, Suprakash and Das, Ram. Virtual Reality Applications in Mental Health: Challenges and Perspectives. *Industrial Psychiatry Journal*, 23(2):83, 2014.

[17] Cecília Lányi. Virtual Reality in Healthcare. *Intelligent Paradigms for Assistive and Preventive Healthcare*, pages 87–116, 2006.

[18] Alexander Miloff, Philip Lindner, Peter Dafgård, Stefan Deak, Maria Garke, William Hamilton, Julia Heinsoo, Glenn Kristoffersson, Jonas Rafi, Kerstin Sindermark, Jessica Sjölund, Maria Zenger, Lena (Lotta) Reuterskiöld, Gerhard Andersson, and Per Carlbring. Automated virtual reality exposure therapy for spider phobia vs. in-vivo one-session treatment: A randomized non-inferiority trial. *Behaviour Research and Therapy*, 118:130–140, 2019.

[19] Barbara Olasov Rothbaum, Larry Hodges, Samantha Smith, Jeong Hwan Lee, and Larry Price. A Controlled Study of Virtual Reality Exposure Therapy for the Fear of Flying. *Journal of Consulting and Clinical Psychology*, 68(6):1020–1026, 2000.

[20] Cristiane Gebara, Tito De Barros-Neto, Leticia Gertsenchtein, and Francisco Lotufo-Neto. Virtual reality exposure using three-dimensional images for the treatment of social phobia. *Revista Brasileira de Psiquiatria*, 38(1):24–29, 2015.

[21] Paul Emmelkamp and Lex Hulsbosch. Virtual Reality Treatment Versus Exposure In Vivo: a Comparative Evaluation in Acrophobia. *Behaviour Research and Therapy*, 40(5):509–516, 2002.

[22] Maria Valderas, Juan Bolea, Pablo Laguna, Montserrat Vallverdu, and Raquel Bailón. Human Emotion Recognition Using Heart Rate Variability Analysis with Spectral Bands Based on Respiration. 2014.

[23] David Hernando, Surya Roca, Jorge Sancho, Alvaro Alesanco, and Raquel Bailón. Validation of the Apple Watch for Heart Rate Variability Measurements during Relax and Mental Stress in Healthy Subjects. *Sensors*, 18:2619, 2018.

[24] HTC. Htc vive vr headset. `https://www.vive.com/us/product/`. Accessed 16-May-2020.

[25] Oculus. Oculus rift s. `https://www.oculus.com/rift-s/`. Accessed 16-May-2020.

[26] Samsung. Samsung Gear VR. `https://www.samsung.com/global/galaxy/gear-vr/`. Accessed 24-March-2020.

[27] YouTube. About YouTube. `https://www.youtube.com/intl/en-GB/about/`. Accessed 18-December-2019.

[28] Creative Commons. Creative Commons License Deed. https://creativecommons.org/licenses/by/4.0/. Accessed 18-December-2019.

[29] Unity. Unity Real-Time Development Platform. https://unity.com/. Accessed 15-December-2019.

[30] Unity Technologies. Profiler Overview. https://docs.unity3d.com/Manual/Profiler.html. Accessed 18-December-2019.

[31] Apple Inc. Xcode. https://developer.apple.com/xcode/. Accessed 18-December-2019.

[32] Apple Inc. Authorizing Access to Health Data. https://developer.apple.com/documentation/healthkit/authorizing_access_to_health_data. Accessed 05-March-2020.

[33] Apple Inc. Health Kit. https://developer.apple.com/healthkit/. Accessed 05-March-2020.

[34] Apple Inc. ResearchKit. https://developer.apple.com/researchkit/. Accessed 05-March-2020.

[35] Apple Inc. CareKit. https://developer.apple.com/carekit/. Accessed 05-March-2020.

[36] Apple Inc. Overview - ResearchKit & CareKit. https://www.researchandcare.org/. Accessed 05-March-2020.

[37] Ian Sommerville. *Software Engineering*. International Computer Science Series. Pearson, 9 edition, 2011.

[38] Google Developers. GVRReticlePointer Prefab. https://developers.google.com/vr/reference/unity/prefab/GvrReticlePointer. Accessed 18-December-2019.

[39] GitHub Inc. About GitHub. https://github.com/about. Accessed 06-April-2020.

[40] Dropbox. About Dropbox. https://www.dropbox.com/about. Accessed 06-April-2020.

[41] Pooja Tanaji Patil. A Study On Evolution Of Storage Infrastructure. *International Journal*, 6(7), 2016.

[42] Andrade, Stevão and Nunes, Fátima and Delamaro, Márcio. A Fault-Based Testing Approach to VR Applications. pages 5–6, 2019.

[43] Bernhard Aichernig and Corrales Delgado. From Faults Via Test Purposes to Test Cases: On the Fault-Based Testing of Concurrent Systems. *Fundamental Approaches to Software Engineering Lecture Notes in Computer Science*, pages 324–338, 2006.

[44] Google. Google Cardboard - Google VR. https://arvr.google.com/cardboard/. Accessed 05-April-2020.

[45] Merge. Merge VR Headset. https://mergeedu.com/headset. Accessed 05-April-2020.

[46] XiaoZhai. XiaoZhai3 Z6 VR Headset. https://www.xiaozhai360.com/z6/. Accessed 19-December-2019.

[47] Fern K Willits, Gene L Theodori, and AE Luloff. Another Look at Likert Scales. *Journal of Rural Social Sciences*, 31(3):6, 2016.

[48] Apple Inc. Your heart rate. What it means, and where on Apple Watch you'll find it. https://support.apple.com/en-us/HT204666. Accessed 06-April-2020.

[49] Xin Meng, Chong Feng, and YuChuan Ji. A Study on the Application of UI Design in Shaping Product Image. In *4th International Conference on Education, Management, Arts, Economics and Social Science (ICEMAESS 2017)*. Atlantis Press, 2017.

[50] Apple Inc. WatchConnectivity Framework. https://developer.apple.com/documentation/watchconnectivity. Accessed 06-April-2020.

[51] Apple Inc. Human Interface Guidelines. https://developer.apple.com/design/human-interface-guidelines/. Accessed 18-May-2020.

[52] Cheri Mullins. Responsive, mobile app, mobile first: untangling the ux design web in practical experience. In *Proceedings of the 33rd Annual International Conference on the Design of Communication*, pages 1–6, 2015.

[53] Daniel Cohen Gindi. Charts. https://github.com/danielgindi/Charts. Accessed 06-April-2020.

[54] Apple Inc. XCTest. https://developer.apple.com/documentation/xctest. Accessed 05-April-2020.

[55] Sundramoorthy Balaji and Sundararajan Murugaiyan. Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management*, 2(1):26–30, 2012.

[56] David Parsons, Ramesh Lal, Hokyoung Ryu, and Manfred Lange. Software Development Methodologies, Agile Development and Usability Engineering. In *18th Australasian Conference on Information Systems*, 2007.

[57] Hussin A Rothan and Siddappa N Byrareddy. The epidemiology and pathogenesis of coronavirus disease (covid-19) outbreak. *Journal of autoimmunity*, page 102433, 2020.

[58] Zunyou Wu and Jennifer M McGoogan. Characteristics of and important lessons from the coronavirus disease 2019 (covid-19) outbreak in china: summary of a report of 72 314 cases from the chinese center for disease control and prevention. *Jama*, 323(13):1239–1242, 2020.

[59] Wenjun Cao, Ziwei Fang, Guoqiang Hou, Mei Han, Xinrong Xu, Jiaxin Dong, and Jianzhong Zheng. The psychological impact of the covid-19 epidemic on college students in china. *Psychiatry research*, page 112934, 2020.

[60] Apple Inc. TestFlight. https://developer.apple.com/testflight/. Accessed 06-April-2020.

[61] Qualtrics. The Leading Reseach, Survey & Experience Software. `https://www.qualtrics.com/uk/`. Accessed 04-March-2020.

[62] Reddit. Reddit: The Front Page of the Internet. `https://www.reddit.com/`. Accessed 29-March-2020.

[63] Reddit. Recruitment for App Store Beta Testers. `https://www.reddit.com/r/TestFlight/`. Accessed 29-March-2020.

[64] Jong-Suk Choi, Jae Bang, Hwan Heo, and Kang Park. Evaluation of Fear Using Nonintrusive Measurement of Multimodal Sensors. *Sensors*, 15(7):17507–17533, 2015.

[65] Yekta Said Can, Niaz Chalabianloo, Deniz Ekiz, and Cem Ersoy. Continuous Stress Detection Using Wearable Sensors in Real Life: Algorithmic Programming Contest Case Study. *Sensors*, 19(8):1849, 2019.

[66] Muse. Muse - Meditation Made Easy with the Muse Headband. `https://choosemuse.com/`. Accessed 06-March-2020.

[67] EMOTIV. Brain Data Measuring Hardware and Software. `https://www.emotiv.com/`. Accessed 06-March-2020.

[68] Wolfram Boucsein. *Electrodermal Activity*. Springer, 2014.

[69] Earl T Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. The oracle problem in software testing: A survey. *IEEE transactions on software engineering*, 41(5):507–525, 2014.

[70] Zhi Quan Zhou, Dave Towey, Pak-Lok Poon, and T.H. Tse. Introduction to the special issue on test oracles. *Journal of Systems and Software*, 136:187, 2018.

[71] Sergio Segura, Gordon Fraser, Ana B Sanchez, and Antonio Ruiz-Cortés. A survey on metamorphic testing. *IEEE Transactions on software engineering*, 42(9):805–824, 2016.

[72] Tsong Chen, Fei-Ching Kuo, Huai Liu, Pak-Lok Poon, Dave Towey, T.H. Tse, and Zhi Quan Zhou. Metamorphic Testing: A Review of Challenges and Opportunities. *ACM Computing Surveys*, 51:4:1–4:27, 2018.

[73] Huai Liu, Fei-Ching Kuo, Dave Towey, and Tsong Yueh Chen. How effectively does metamorphic testing alleviate the oracle problem? *IEEE Transactions on Software Engineering*, 40(1):4–22, 2013.

[74] The William and Flora Hewlett Foundation. Open educational resources. `http://www.hewlett.org/programs/education/open-educational-resources`. Accessed 18-May-2020.

[75] Creative Commons. Creative Commons. `https://creativecommons.org`. Accessed 18-April-2020.

[76] Apple Inc. Code Signing. `https://developer.apple.com/library/archive/documentation/Security/Conceptual/CodeSigningGuide/Introduction/Introduction.html`. Accessed 07-May-2020.

Appendix A

Gaze Function

The examples in Section 4.2 do not show the interaction with buttons. On the computer, it is always a click of the mouse or keyboard button, but in VR it is different. All interactions in VR are done by Gaze [38] software method or by using the controllers (Google Daydream has one, but Cardboard does not).

Gaze is a method of manipulating or interacting with any object in VR (button, figure, text, etc) just by looking at it. Gaze consists of mainly three things: Pointer, Timer and Image. Pointer is usually a pre-build package made by Google VR (GVR) and shows a white point (dot) always in the middle of the screen. Timer is the function that will count down the time until there is a call for any other function or action. Image is the figure that will be displayed when the timer is loading.

There were three options to implement the Gaze in the VR application:

1. Pointer - do not implement Timer or Image functions, just look at something and perform the action.
2. Pointer and Timer - do not add any animations to the functions, look at the object for some time and then do the assigned function.
3. Pointer, Timer and Image - perform complete, game style animation. Look at the something and see loading animation that is related to the timer.

The last was chosen to implement in the VR application - Pointer, Timer and Image. No relevant tutorial or documentation about Gaze in Unity was found online. The screenshot below shows the in-process action of pushing the button - half of the time has passed, green circles are indicating it. When the circle will be finished (full on-screen), the scene will change (Figure A.1). The script for Gaze is built in C# language and available in Section C.

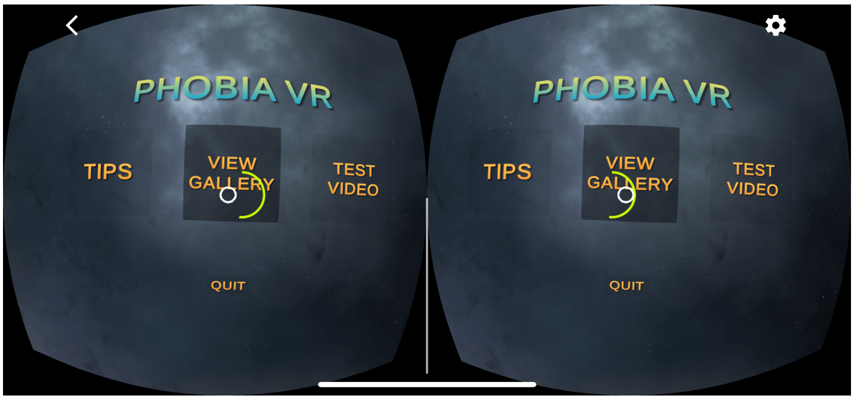


Figure A.1: GVR Menu Interaction

Appendix B

Software & Hardware Specifications

Based on information from Section 4.2.2, it can be concluded which hardware and software is required to run the VR application. Assumed devices and tools for a smooth run of the app are listed in the Table B.1.

Hardware / Software	Version	Cost	Comments
iPhone X	iOS 13.2	$999 USD	
MacBook Air	macOS 10.15	$999 USD	
Google Daydream View 2	N/A	$99 USD	Not available anymore
Unity	2019.f.02.1	Free	
Xcode	11	Free	Download from AppStore
Apple Developer License	N/A	$99 USD	Required to publish iOS apps

Table B.1: Hardware & Software Specifications

The versions of the hardware are minimal requirements and it is unknown whether the application would work with the versions lower than the ones in Table B.1. Furthermore, there are no other platform specifications available except iOS and macOS since the application was only tested on those platforms. It is planned to include support for other platforms after the end of final version release for iOS.

Appendix C

Code Samples

The code sample below is the script for Gaze function to initiate the timer and animation when the users focuses on a button or text field. It is written in C# programming language and works only when the user looks directly at the object with the cursor in the middle of the headset.

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

public class GVRButton : MonoBehaviour
{
    public Image GVRImage;
    public UnityEvent GVRClick;
    public bool GVRStatus;
    public float TotalTime = 2;
    public float GVRTimer;
    // Update is called once per frame
    void Update()
    {
        if (GVRStatus)
        {
            GVRTimer += Time.deltaTime;
            //GVRImage.fillAmount += GVRTimer / TotalTime ;
            GVRImage.fillAmount += 0.01f;
        }
        if (GVRTimer > TotalTime)
        {
            GVRClick.Invoke();
        }
    }

    public void GVROn()
    {
        GVRStatus = true;
    }

    public void GVROff()
    {
        GVRStatus = false;
        GVRTimer = 0;
        GVRImage.fillAmount = 0;
    }
```

```
}
```

Listing C.1: C# Gaze Interaction

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

public class SceneChanger : MonoBehaviour
{
    public void LoadScene(string sceneName)
    {
        SceneManager.LoadScene(sceneName);
    }
}
```

Listing C.2: Scene Changer

Appendix D

Responsive Design

This section describes the responsive design techniques that were used in the HR monitor application and presents the corresponding code samples.

D.1 Device Classes

Responsive Design

There are different types (classes) of iOS devices which are distinguished by the screen dimensions. For example, iPhone X goes by wC hR (width: Compact, height: Regular). If a developer would not use the responsive design and leave one layout for all classes, the layout would look something like in the Figure D.1, launch screen for iPhone 4s. Applying responsive design techniques is important to ensure that the application is working as intended on all iOS devices and allows the user to interact with all objects on the screen as it was produced in the initial layout (target device).

The HR monitor application accepts all possible device classes and changes the layout accordingly to the width and height ratio of the device. Most of the objects on the screen are constrained programatically by the dimensions of the current screen, thus preventing the problem of broken layouts, such as on Figure D.1.

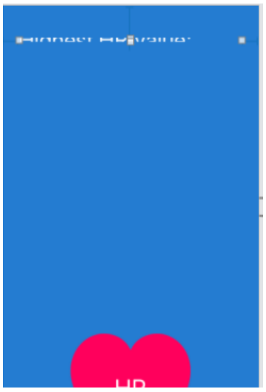


Figure D.1: iPhone 4s Launch Screen

D.2 Chart Image Email Attachment

As discussed in the Section 4.3.3, responsive design techniques were used to correctly attach the chart image in the email when the user shares the results. Unfortunately, iOSCharts [53] library includes a function to save the chart image which does not work D.2.

```
UImageWriteToSavedPhotosAlbum(chart.getChartImage
    (transparent:false)!, nil, nil, nil)
```

Listing D.1: Save Image Function

This function saves the chart on the device's gallery, but when the user would look at it, the chart would be empty and the background color would be changed to white (Figure D.2).

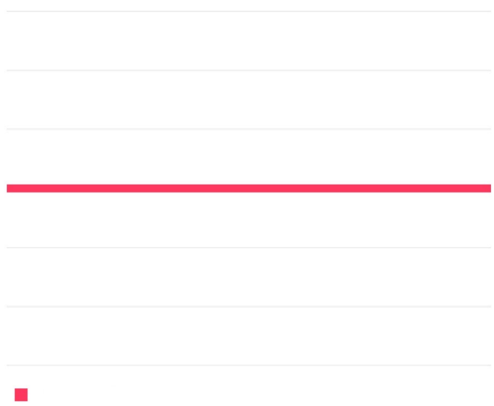


Figure D.2: Invalid Chart

The solution was to save a snapshot of the screen. However, the user does not need the screenshot of the full screen, but the chart alone. Below is the code sample that gets the image for the snapshot:

```
UIGraphicsBeginImageContextWithOptions(CGSize(width: x1,height: y1),
   false, 0);

self.view.drawHierarchy(in: CGRect(x: x2,y: y2,width: view.bounds.size.
   width,height: view.bounds.size.height), afterScreenUpdates: true);
```

Listing D.2: Save Chart Function

x1 and y1 in this code sample are the size of the image (width and height). x2 and y2 are the positions of x and y coordinates of the image in the window (view). The goal was to make a snapshot of the chart dynamic, to make sure that devices with different screen size could save the correct snapshot of the screen including only the chart. Charts [53] documentation is not updated on that topic, thus the author had to come up with a creative solution.

```
UIGraphicsBeginImageContextWithOptions(CGSize(width: chtChart.bounds.
   width,height: chtChart.bounds.height), false, 0);
```

```

let xCoord = chtChart.frame.origin.x * (-1)
let yCoord = chtChart.frame.origin.y * (-1)

self.view.drawHierarchy(in: CGRect(x: xCoord,y: yCoord,width: view.
    bounds.size.width,height: view.bounds.size.height),
    afterScreenUpdates: true)
```

Listing D.3: Saving Chart Function

The function that is shown in the code sample above makes sure the size of the snapshot is change to the size of the screen. Additionally, x2 and y2 are now representing the coordinates xCoord and yCoord which are retrieved from the left upper corner of the chart. It works on devices with different screen sizes (Section D.1) and saves the image of the chart correctly. Figure 4.14 in Section 4.3.3 shows the result of sharing the information through the email with the correct size chart image attached.

Appendix E

User Manual

Following section explains how to launch the applications and test classes. There are several methods to run the applications and view the code, all are listed below.

E.1 Setting Up

Due to the the upload limitations on Moodle and Microsoft Forms (MS Forms), several changes to the submission files were made. Please follow this section step-by-step to set up all the necessary files to run the applications.

First download the four files available from the MS Forms and put them in one folder. Remove the .ppt extensions by editing the name of the files (Figures E.1 - E.2). Make sure you are changing the file type and not just the name of the file. You can check if the type was changed by clicking on the file and then `Get Info`. Under the `Name & Extensions` make sure that there is not trailing .ppt extension.

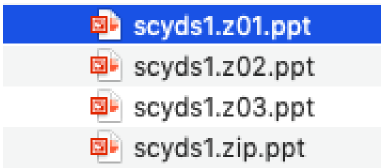


Figure E.1: Project Files

Figure E.2: Project Files Edited

Next, make sure you are in the same directory with the files. Run the following command to combine the files into one zip file:

```
$ zip -F scyds1.zip --out ../<new_name>.zip
```

The name of the file has to be `scyds1.zip`, but you need to output the command to another directory to eliminate the duplication of the file names. The produced file will be around 2GB in size and contain all the project materials. Run the md5 command to get the hash of the new zip file:

```
$ md5 scyds1.zip
```

Save the hash in a text file or just leave it in the terminal to be displayed.

Next, Go to Moodle and download the `scyds1.txt` from the related materials. It contains the md5 hash of the original zip archive that had all the project files. The hash in the text file should be the same with the hash string you have got from the zip file. Check if it is correct and unzip the file. `FYPFiles` folder contains all the necessary materials to run the application which will be described in the next section.

E.2 HR Monitor Application Installation

HR monitor application can be launched on a simulator and a physical device. However, to interact and start data collection process, a physical watch and an iOS device are needed. Embedded simulator in Xcode does not have sensors, thus this section is split into two methods to launch the application.

E.2.1 Unsigned Method

In your downloaded folder find *HRMonitorApplication*, then *HeartRateMonitorIPA* folder. Go to *Apps* and find *HeartRateMonitor.ipa* file (Figure E.3).

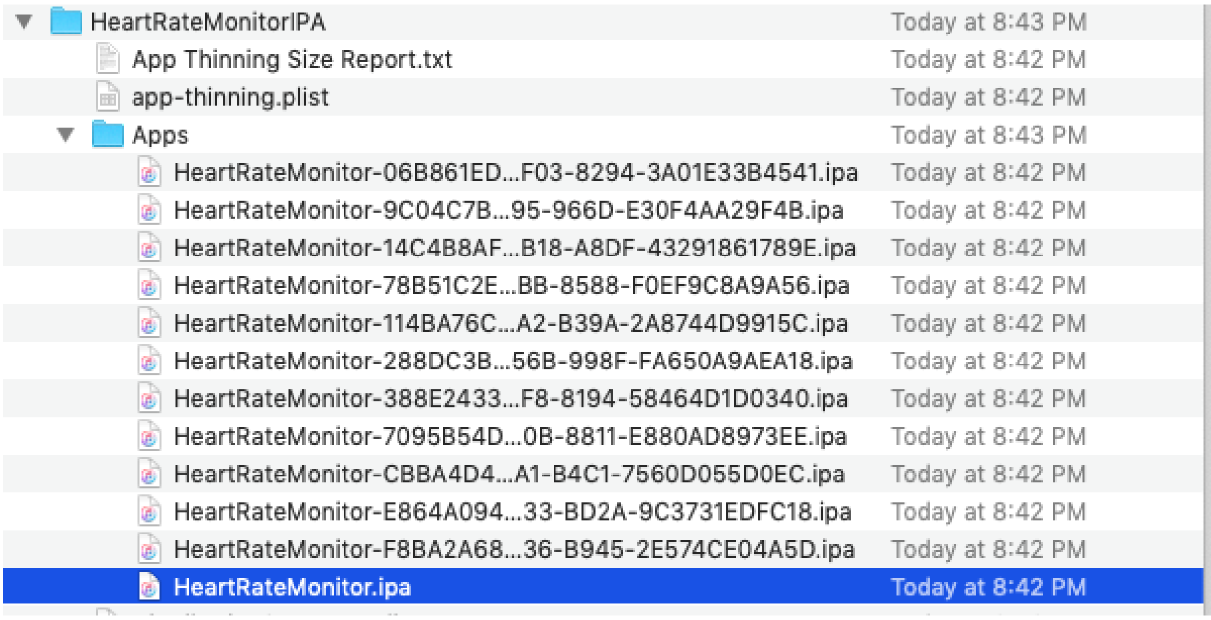


Figure E.3: Finding Image File

Next, launch Xcode IDE and click on *Open another project...* button (Figure E.4).

Figure E.4: Launching Xcode

Here you can follow through *HeartRateMonitor* folder and find *HeartRateMonitor.xcodeproj* file (Figure E.5). Selecting this file will launch the project and all files included in it.

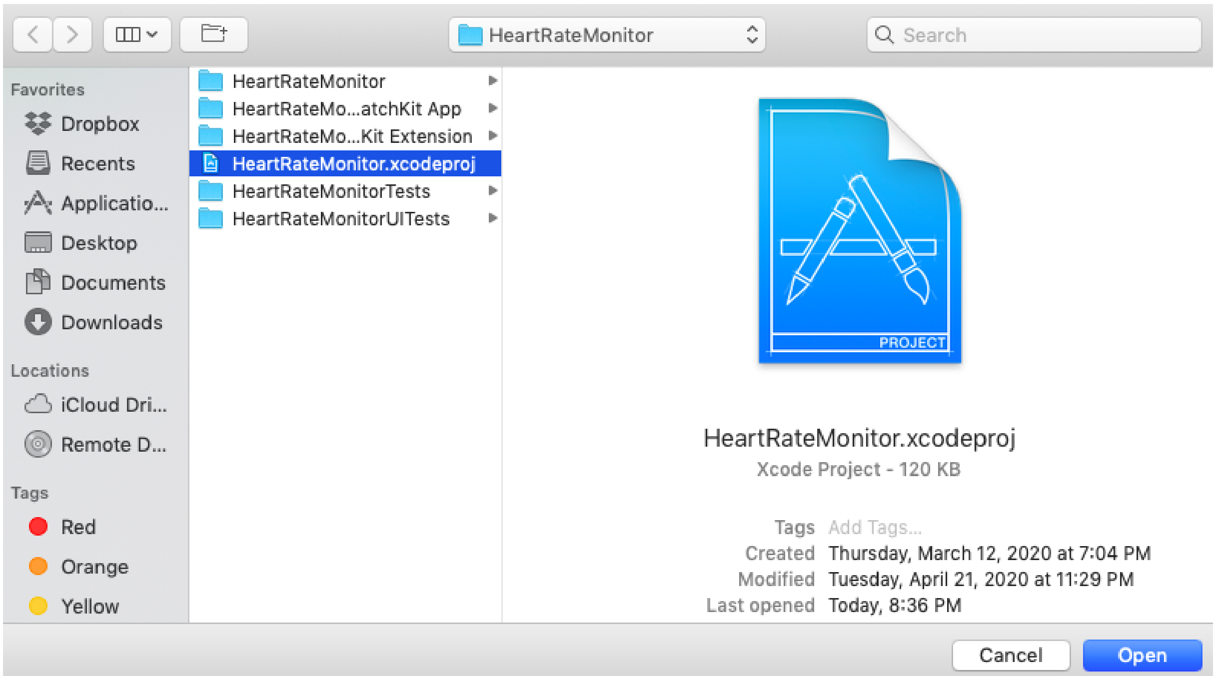


Figure E.5: Selecting Project File

Next, click on *Window - Devices and Simulators*. Choose your physical connected device (a simulator will not work), drag and drop the .ipa file that you have found in the beginning to the *Installed Apps* window. This action will automatically install the application on your device (Figure E.6). It is important to note that both iOS and watchOS devices must have .ipa files installed. Otherwise, the data collection process will not start and prompt an alert message.

Figure E.6: Drag & Drop Image File

E.2.2 Signed Method

If you wish to run the test classes or modify the code and then run the application, you would need to perform the Code Signing [76]. It is a security feature by Apple to certify that the application was created by you. Since we are planning to alter the application and run it on the device that does not belong to the signer, we need to change the developer's team.

Select your project name at the top left corner. Then choose *Signing & Capabilities*. For the *Team* name you have to select your Apple ID (Figure E.7). The application can only be launched from your device if unique ID is selected. Additionally, you can run the application on simulators and launch any test classes that are included in the project.

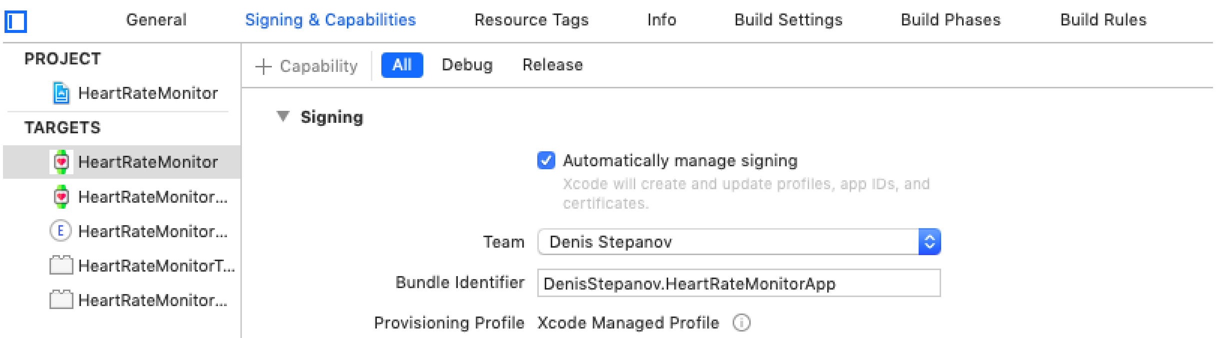


Figure E.7: Developer Singing

After you have finished signing, you can launch specific test classes or the whole application by selecting a device on the top left corner and clicking play button (Figure E.8).

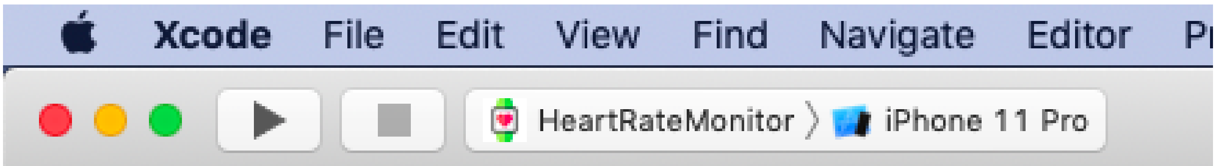


Figure E.8: Unsigned Method Application Launch

E.2.3 VR Application Installation

Due to the large size of Unity project (6.76GB), VR application was first converted to the file type for iOS devices. The submissions allows only 250MB, thus this section describes how to install VR application through Xcode only. The application could run on simulator with the unsigned method, but you will not be able to interact with it because simulators do not have a gyroscope, and all interaction is based on gyroscope functionality (Appendix A).

Most of the files in the Xcode project are automatically generated by Unity. You will not have an access to the scripts, such as button interactions and media playback implementation because they are encrypted before the compilation in Unity project (Code samples are included in Appendix C). However, you can access the application and launch in on your device with similar methods that were done for HR monitor application: signed and unsigned.

The signed method requires the same actions as for HR monitor application installation. Find the .ipa file (*VRApplication - VRPhobiaApplicationIPA - Apps - Unity-iPhone.ipa*) and drag it to the *Installed Apps* of your physical device in Xcode. The application would be automatically installed on your device for you to run it.

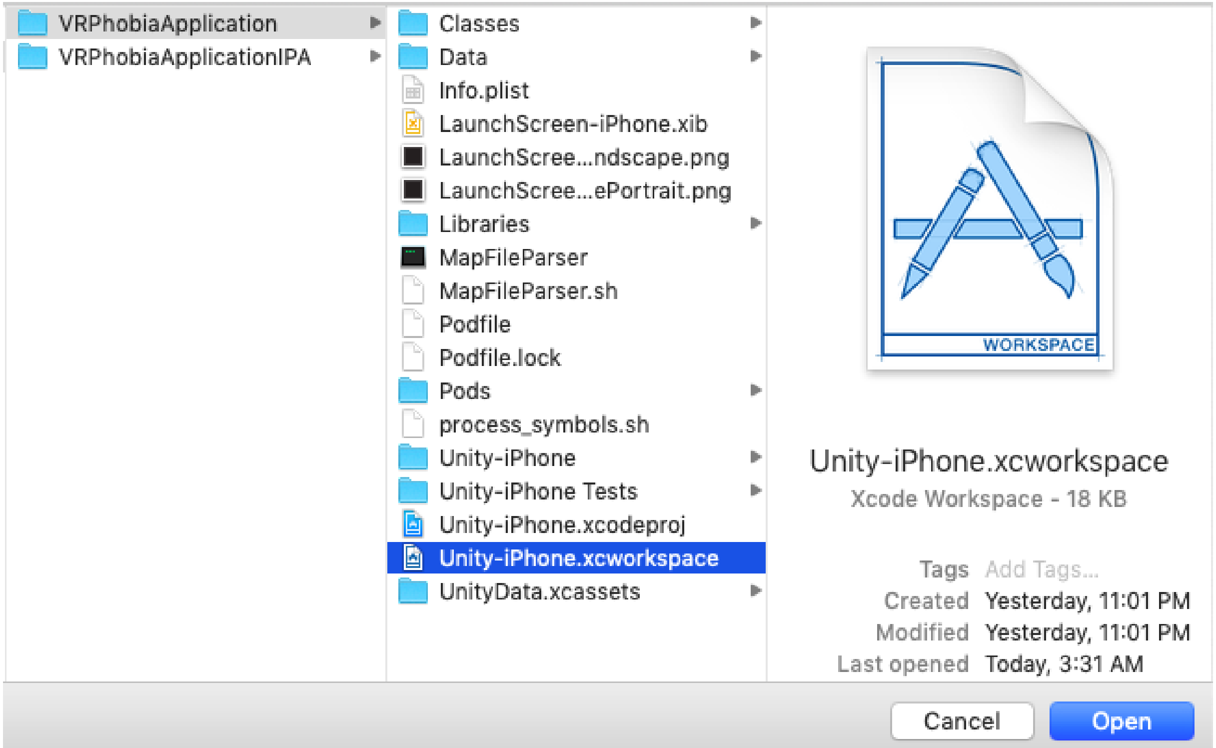


Figure E.9: VR Project File Selection

The unsigned method requires almost the same actions with developer's signing as in HR monitor application installation. You would need to select your identifier for *Team* in *Signing & Capabilities*. After the signing is done, you can run the application on any simulator or a physical device. The only difference between VR and HR component installation is that you need to open another project file to start it. Instead of selecting .xcodeproj file, such as in Figure E.5 for HR component, you need to open .xcworkspace file (Figure E.9).

E.3 Interaction

This section specifies the interaction steps after the application was launched. There are various possibilities on how to run the application, but the general idea of application use is described below.

First, start off by launching both iOS and watchOS Heart Rate Monitor applications (Figures E.10 & E.11). Your watch must be paired with your iOS device.

Figure E.10: HR Monitor iOS Launch Screen

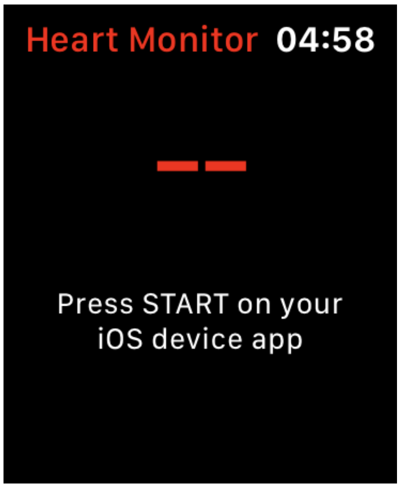


Figure E.11: HR Monitor watchOS Launch Screen

Your watchOS application will request permissions on the first run. Agree to the permission on reading heart rate data (Figure E.12).

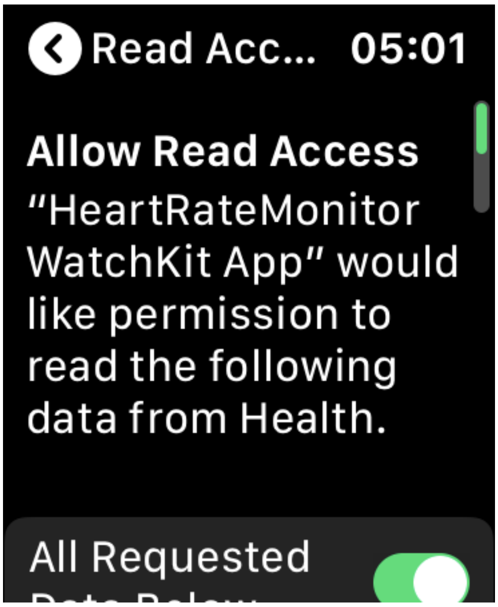


Figure E.12: watchOS Permission Window

On the iOS device, push *START* button to launch the process. The button will turn red and the chart will show up at the top of the screen. The watchOS device will change text labels and display the updated heart rate of the user (Figures E.13 & E.14). It is worth noting that the chart will not appear instantly. The iOS device waits for the first response from the watch to set the initial value on the chart and then displays it on the screen.

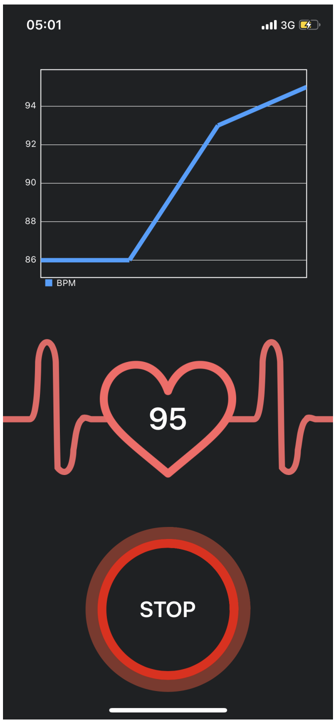


Figure E.13: HR Monitor iOS Process Start

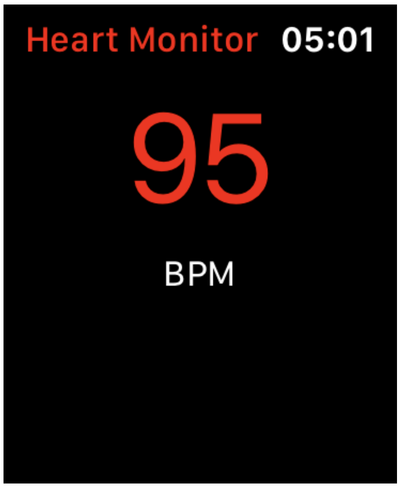


Figure E.14: HR Monitor watchOS Process Start

Next, put the phone in the headset and launch the VR application (Figure E.15).

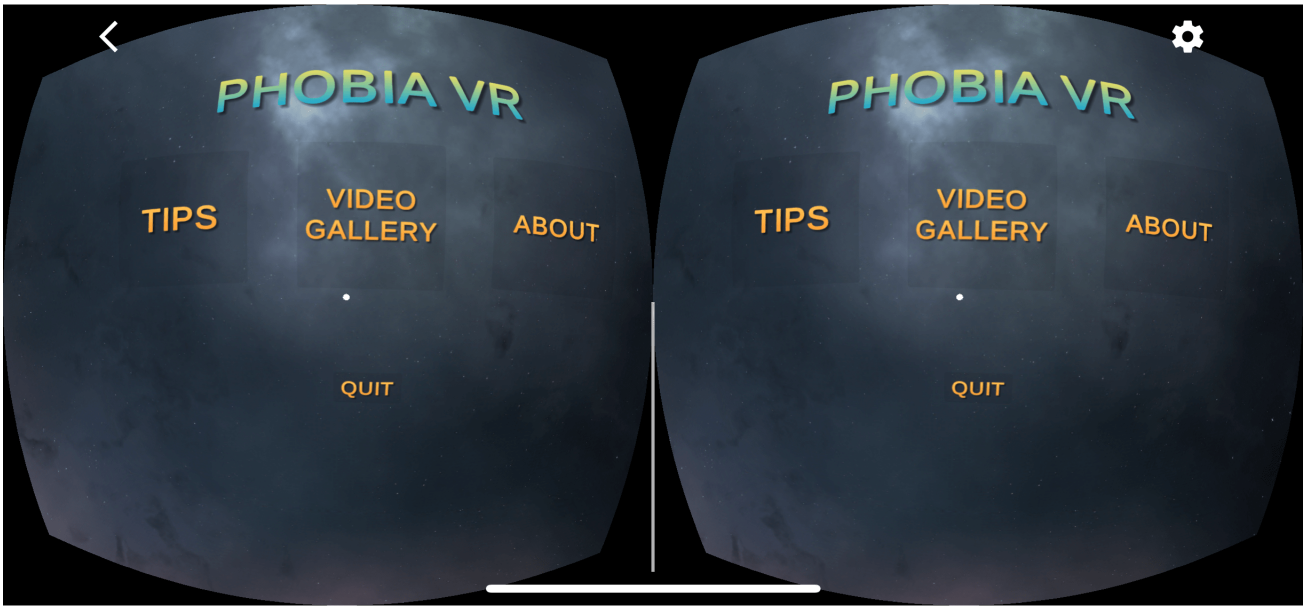


Figure E.15: VR Application Launch Screen

You can choose the buttons by focusing the white cursor on them (Appendix A explained how this feature works). You can lock your view on *Video Gallery* for two seconds and the application will change the window (Figure E.16).

Figure E.16: VR Application Video Gallery

You will have three videos to choose from. Select any video you like and lock your cursor on the video thumbnail for two seconds. The video will be launched (Figure E.17).

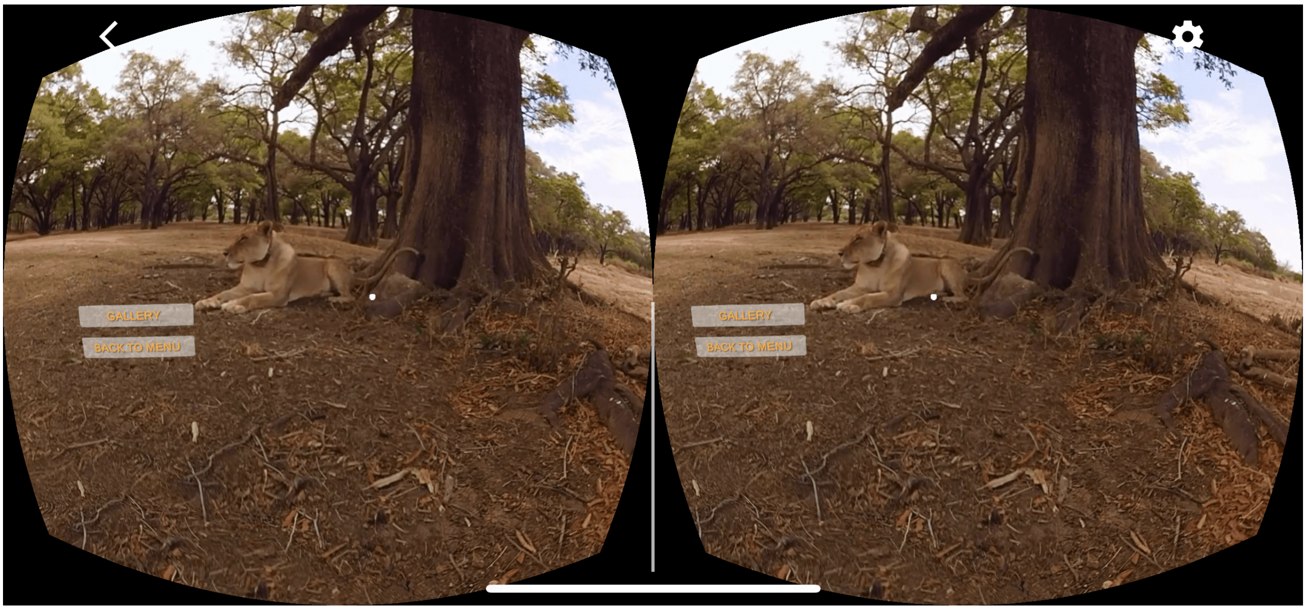


Figure E.17: VR Application Lions Video

At any point during the playback you can exit to the gallery or the main menu by focusing the cursor on one of the buttons (Figure E.17). When you are finished with the VR session, you can find the quit button in the main menu.

Now you can return to HR monitor application. The whole time you were immersed in your VR session, the HR sensor on the watch was tracking your heart rate and sending it to the iOS application. You should be able to see the updated chart on the top of the screen (Figure E.18).

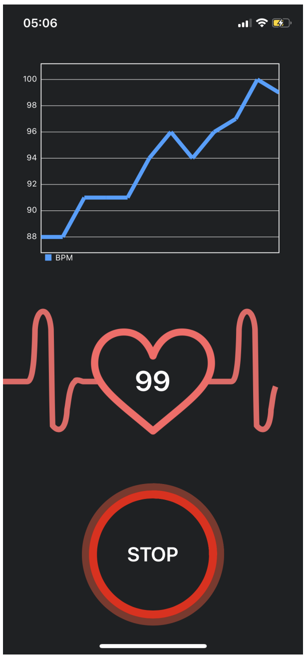


Figure E.18: HR Component Updated Screen

Push the stop button. It will stop the measuring process and show you the pop up window. Insert your name, age and phobia name. If you do not know the name of your phobia, type "others" (Figures E.19 & E.20).

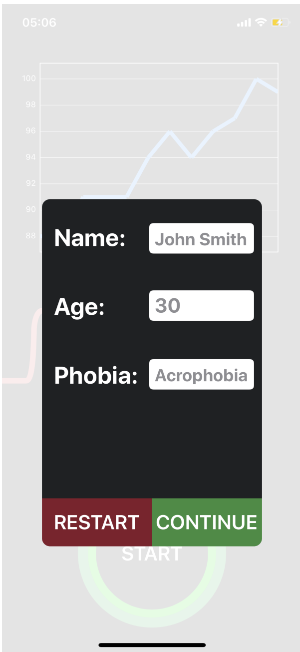


Figure E.19: HR Monitor Pop Up Screen

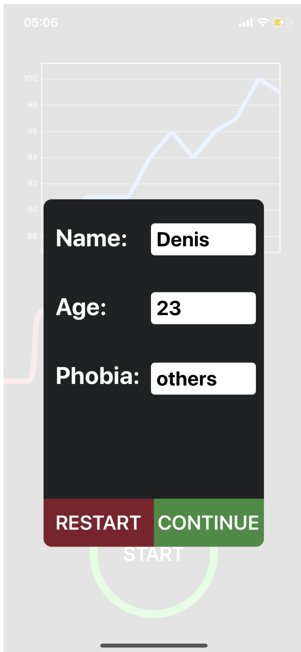


Figure E.20: HR Monitor User Input

Proceed by pushing *CONTINUE* button. You will now see the results screen (Figure E.21). Check if the information is correct.

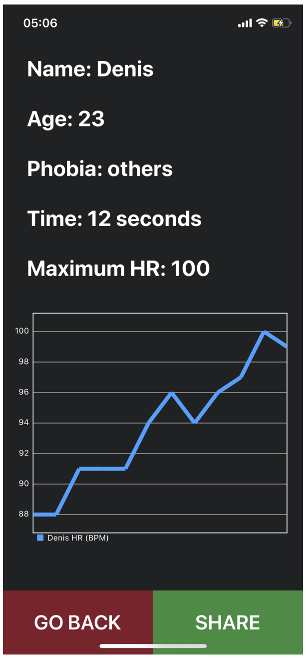


Figure E.21: HR Component Results Screen

Click on *SHARE* button and select how you want to share your results, such as email (Figure E.22). Send the information to the channel that is available and check if it has arrived successfully.

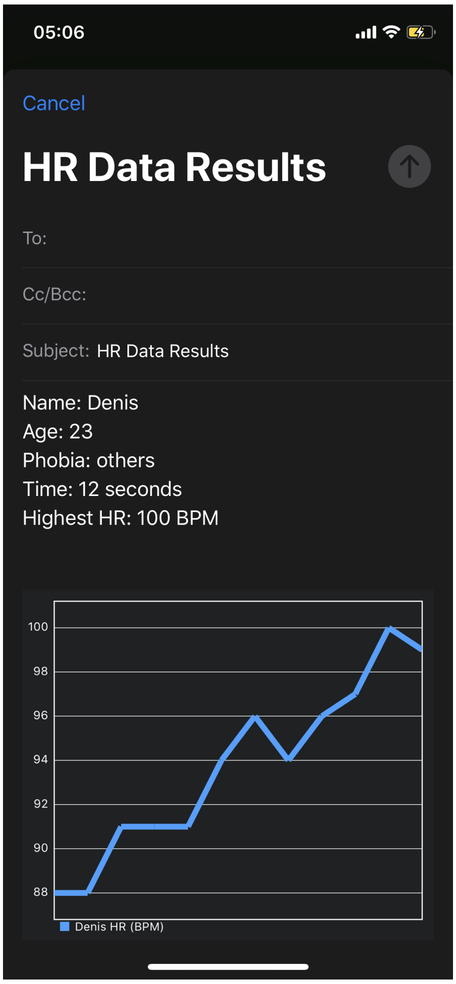


Figure E.22: HR Component Email Share Screen

Appendix F

Survey

This section presents the questionnaire that was used to gather feedback about the HR monitor application design. It consists of six multiple choice questions and two questions with open form answers. Section F.1 shows the answers report for multiple choice questions. Open form answers were not included in the report as most of them state that nothing needs to be changed in the application or the answer box was left empty. Table F.1 lists the differences between the initial and final prototypes of HR monitor application.

	Initial Design	Latest Design
Animations		✓
User Input Validation		✓
Background Mode		✓
Standalone Watch App		✓
Error Messages		✓
Responsive Design		✓
Color Choice		

Table F.1: Initial vs. Latest HR Application Design

Participant Information Sheet

Evaluating the Impact of Phobia-Motived experiences Delivered through Virtual Environment

Dear Participant,

Thank you for agreeing to participate in this questionnaire survey in connection with my Undergraduate coursework at the University of Nottingham Ningbo. The project is a study of phobias and how people react to them in Virtual Reality. Your participation in the survey is voluntary. You are able to withdraw from the survey at any time and to request that the information you have provided is not used in the project. Any information provided will be confidential. Your identity will not be disclosed in any use of the information you have supplied during the survey. The research project has been reviewed according to the ethical review processes in place in the University of Nottingham Ningbo. These processes are governed by the University's Code of Research Conduct and Research Ethics. Should you have any question now or in the future, please contact me or my supervisor. Should you have concerns related to my conduct of the survey or research ethics, please contact my supervisor or the University's Ethics Committee.

Yours truly,

Denis Stepanov

Contact details:

Student Researcher: Denis Stepanov scyds1@nottingham.edu.cn

Supervisor: Dr Dave Towey dave.towey@nottingham.edu.cn

University Research Ethics Committee Coordinator: Ms Joanna Huang
Joanna.Huang@nottingham.edu.cn

Click 'I Understand' to continue the survey.

论文题目：Evaluating the Impact of Phobia-Motived experiences Delivered through Virtual Environment

尊敬的参与者：　　谢谢您参与这次问卷调查。这次问卷调查是我在宁波英国诺丁汉大学本科论文相联系的。我的论文是有关恐惧症及人们在虚拟现实的表现。　　您是自愿参与此次问卷调查的。您可以在任何时候选择放弃这次的问卷调查，并要求您提供的信息不被使用在此次调查中。您提供的所有信息都是保密的。在使用您提供的信息时不会涉及您的身份以及个人信息。　　宁波诺丁汉大学已根据研究道德检查程序对这项研究项目进行检查。这一程序是在学校关于研究行为和研究道德的行为标准的指导下进行的。如果您现在或将来有任何疑问，请联系本人或我的导师。如果您对我在问卷中的研究行为或研究道德有任何质疑，请联系我的导师或者英国诺丁汉大学的道德委员会。

Denis Stepanov

联系方式：

研究员：　Denis Stepanov　scyds1@nottingham.edu.cn

导师：　Supervisor:　Dr Dave Towey　dave.towey@nottingham.edu.cn

诺丁汉大学研究道德委员会秘书：　Ms Joanna Huang Joanna.Huang@nottingham.edu.cn

点击'I Undesrtand'来开始这次问卷调查

○ I Understand

PARTICIPANT CONSENT FORM

Project title Evaluating the Impact of Phobia-Motived experiences Delivered through Virtual Environment

Researcher's name Denis Stepanov
Supervisor's name Dr Dave Towey

I have read the Participant Information Sheet and the nature and purpose of the research project has been explained to me.

I understand and agree to take part.

I understand the purpose of the research project and my involvement in it.

I understand that I may withdraw from the research project at any stage and that this will not affect my status now or in the future.

I understand that while information gained during the study may be published, I will not be identified, and my personal results will remain confidential.

I understand that the data collection will be recorded.

I understand that data will be stored in accordance with data protection laws.

I understand that I may contact the researcher or supervisor if I require more information about the research, and that I may contact the Research Ethics Sub-Committee of the University of Nottingham, Ningbo if I wish to make a complaint related to my involvement in the research.

I have watched the demo video presenting the differences between initial and latest prototype of HR monitor application.

Contact details

Student Researcher: Denis Stepanov scyds1@nottingham.edu.cn
Supervisor: Dr Dave Towey dave.towey@nottingham.edu.cn
University Research Ethics Committee Coordinator: Ms Joanna Huang
Joanna.Huang@nottingham.edu.cn

Click 'I Agree' to continue the survey.

○ I Agree

Based on the demo of INITIAL prototype, rate the user interface from Terrible to Excellent.

- Terrible
- Poor
- Average
- Good
- Excellent

Based on the demo of INITIAL prototype, rate how well the application displays the information for the users about their heart rate.

- Terrible
- Poor
- Average
- Good
- Excellent

Based on the demo of INITIAL prototype, rate the color choice for the application (buttons, background, text).

- Terrible
- Poor
- Average
- Good
- Excellent

Based on the demo of LATEST application prototype, rate the user interface from Terrible to Excellent.

- ◯ Terrible
- ◯ Poor
- ◯ Average
- ◯ Good
- ◯ Excellent

Based on the demo of LATEST prototype, rate how well the application displays the information for the users about their heart rate.

- ◯ Terrible
- ◯ Poor
- ◯ Average
- ◯ Good
- ◯ Excellent

Based on the demo of LATEST prototype, rate the color choice for the application (buttons, background, text).

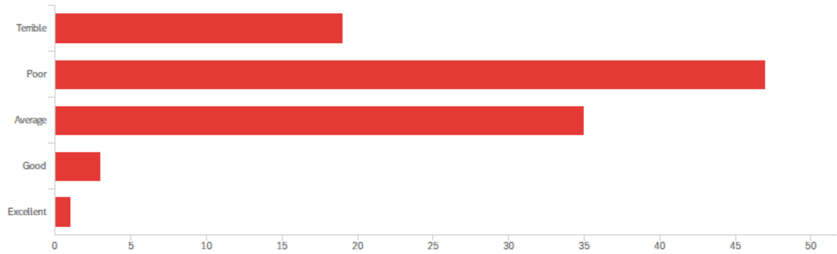
- ◯ Terrible
- ◯ Poor
- ◯ Average
- ◯ Good
- ◯ Excellent

Based on the application functionality that you have seen in the demo video, what do you think is missing?

Based on the application design that you have seen in the demo video, what do you think is missing?

F.1 Data Report

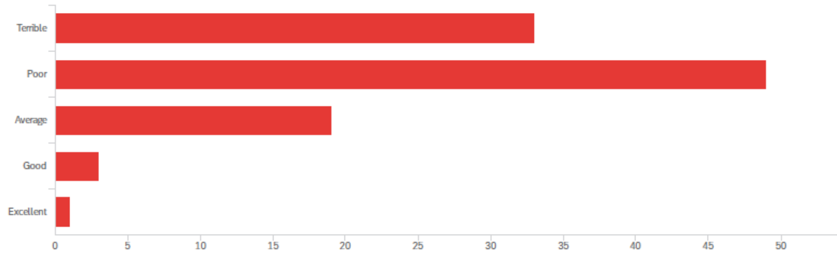
Question #1: Based on the demo of INITIAL prototype, rate the user interface from Terrible to Excellent (Figure F.1).



#	Field	Choice Count	
1	Terrible	18.10%	19
2	Poor	44.76%	47
3	Average	33.33%	35
4	Good	2.86%	3
5	Excellent	0.95%	1
			105

Figure F.1: Question #1

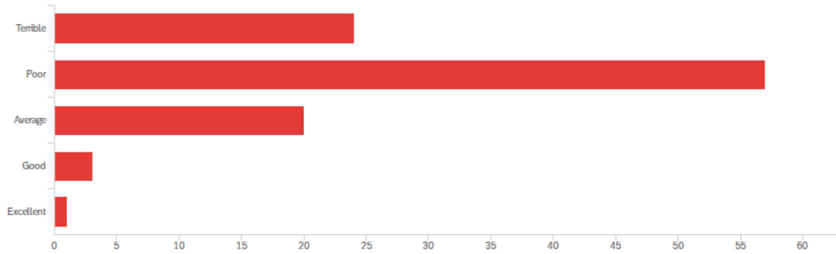
Question #2: Based on the demo of INITIAL prototype, rate how well the application displays the information for the users about their heart rate (Figure F.2).



#	Field	Choice Count	
1	Terrible	31.43%	33
2	Poor	46.67%	49
3	Average	18.10%	19
4	Good	2.86%	3
5	Excellent	0.95%	1
			105

Figure F.2: Question #2

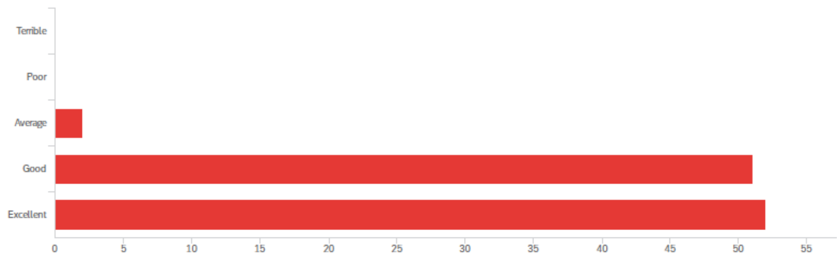
Question #3: Based on the demo of INITIAL prototype, rate the color choice for the application (buttons, background, text) (Figure F.3).



#	Field		Choice Count
1	Terrible	22.86%	24
2	Poor	54.29%	57
3	Average	19.05%	20
4	Good	2.86%	3
5	Excellent	0.95%	1
			105

Figure F.3: Question #3

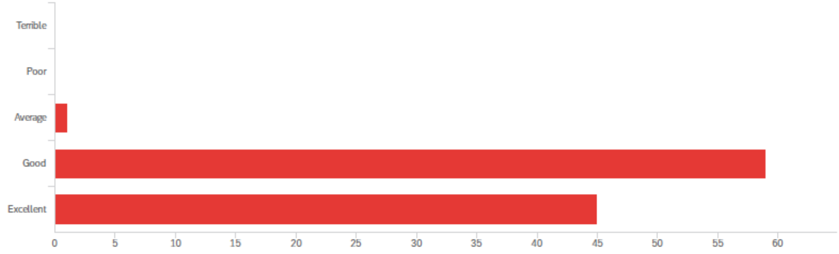
Question #4: Based on the demo of LATEST application prototype, rate the user interface from Terrible to Excellent (Figure F.4).



#	Field		Choice Count
1	Terrible	0.00%	0
2	Poor	0.00%	0
3	Average	1.90%	2
4	Good	48.57%	51
5	Excellent	49.52%	52
			105

Figure F.4: Question #4

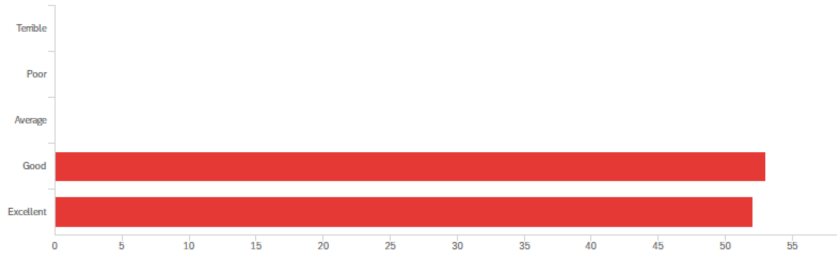
Question #5: Based on the demo of LATEST prototype, rate how well the application displays the information for the users about their heart rate (Figure F.5).



#	Field	Choice Count	
1	Terrible	0.00%	0
2	Poor	0.00%	0
3	Average	0.95%	1
4	Good	56.19%	59
5	Excellent	42.86%	45
			105

Figure F.5: Question #5

Question #6: Based on the demo of LATEST prototype, rate the color choice for the application (buttons, background, text) (Figure F.6).



#	Field	Choice Count	
1	Terrible	0.00%	0
2	Poor	0.00%	0
3	Average	0.00%	0
4	Good	50.48%	53
5	Excellent	49.52%	52
			105

Figure F.6: Question #6