Physicalization of dynamic, physiological data

Written by: Dennis Peeters, S2372312
Program: Creative Technology
Supervisors: Champika Ranasinghe, Auriol Degbelo, Faizan Ahmed

Date: 07-07-2022
Version: 1.0

Abstract

Data is becoming a more prevalent part of our daily lives, and research into making it more accessible and intuitive to interact with for a broader audience has recently become more of a point of attention. However, the challenges and opportunities in the development of a physicalization of real time, dynamic data has been largely unexplored. One of the use cases that will be explored in this research is the use of physiological data in order to physicalize human affect. There are a broad spectrum of purposes for such a system, like reflective tasks, communication of physiological states for people who might not be able to or an intuitive tool to use reliable data regarding a user's affect with regards to interacting with a product in development for researchers.

Developing such a system come with several different aspects, such as sensors, physicalization types and classification of physiological states. These topics have been explored in the background research. Afterwards, the development of a physicalization of dynamic, real-time data was commenced, in which the first step was ideation. During this phase, a first design of such a system was developed. Afterwards, the different components of such a system were further specified. These were the physicalisation itself, the wearable to acquire physiological data and the pipeline, which is the backbone of the system. It describes how the data is acquired, transmitted, processed and stored throughout the full system, and forms a blueprint for future physicalizations. After specification, the system was built and tested with users to find out more in regards to the usefulness and effectiveness of the full system. This showed that a physicalisation of real-time dynamic data is possible and useful, however, several challenges regarding the accuracy of the classification of physiological state and the sensors were found. These were all described in the discussion, which provides a list of components that need future research. The result of the research proves the viability of a physicalization of dynamic data, and describes the challenges related to it, with a blueprint in the form of a pipeline as a basis for future developments of physicalizations of dynamic data.

Acknowledgement

This research would not have been possible without the help of several people. First of all, my supervisors Champika Ranasinghe, Auriol Degbelo and Faizan Ahmed, who have provided guidance and support whenever needed, both during meetings or via mail. Their feedback has been very helpful in order to guide the research in the right direction, and their expertise has proven to be invaluable during the more challenging parts of the research. Next to that, family and friends have been very helpful as well, for their support during the project and their guidance when challenges regarding building parts arose. Finally, all the people that have provided feedback, and especially the participants of the research conducted in the user study deserve a special thanks as well.

Table of Contents

List of Figures

Chapter 1 – Introduction

Physiological data such as heart rate, heart rate variation, respiration rate, skin conductance (obtained by sensing and processing physiological data) can reveal important insights about a person's emotional state such as valence and arousal at a given time. Knowing a person's emotional state at a given moment is useful for many purposes. For example, caretakers of mentally impaired people (such as for example autistics) can use them to better understand and reflect on emotional states of their cared-for, especially to understand stimuli that can change their arousal or valence level. On the other hand, knowing the affect level of a subject is also useful for researchers. For example, a researcher of a Virtual Reality (VR) application (e.g. an immersive analytics application or a game), might want to explore the user experience and emotional responses of a subject while using their VR application. Furthermore, one might want to communicate or share their emotional state with their loved ones or close friends for wellbeing related reasons. They can also be used for self-reflection - to reflect upon various stimuli and events that cause emotional arousal and valence. For almost all these potential applications, there is one fundamental requirement: representation of derived emotional states (arousal and valence) in an easy to interpret format and in an accessible manner.

Current representations of physiological data (acquired through smart wearables for example) and the emotional state are usually digital and use either numbers or plots of numbers visualized on digital screens. On the one hand, this makes interpretation a difficult task, especially to a general audience. On the other hand, as they are meant to be visualized on screen, usually, on a mobile device, people should make an explicit intentional action to look at that information. This makes, less frequent access and usage of derived emotional information. Data physicalizations have the potential to solve these problems, for example by representing abstract data in a more interpretable and graspable manner using various physical modalities and, by making physical installations that can be integrated to user's real environment eliminating the requirement for intentionally opening an app on a screen-based device.

Thus, this thesis focuses on developing a data physicalization that can be integrated to user's real environment to physically visualize a person's emotional state (arousal and valence). In order to do this, physiological data has to be acquired and physicalized based on regular time intervals. Although most wearables can acquire physiological data required to derive a person's emotional state (arousal and valence), they come with their own cloud-based eco-systems where user data can be accessed only through an API, where the upload does not always happen immediately and often require the user to issue the upload command manually [36]. This makes it challenging to use existing wearables for physicalizing data in a timely manner.

Next to challenges regarding the data collection, such a data physicalization will come with challenges like the computation needed, data collection and a proper representation of the data in a different way fitting into a person’s natural environment, so it can offer people an easy-to-use, intuitive, and interactive tool for reflective purposes.

Chapter 1.1 - Research question

This project aims to explore physical visualisation of dynamic data in real time using physiological data as a use-case, to see the challenges and opportunities that would come with such a system. As mentioned in chapter 1, using physicalizations can make sure users get a better understanding by interacting with the data in a different way, making the data accessible to a broader audience. This lead to the following research question:

What are the challenges and opportunities in developing an interactive and intuitive way of representing human affect using dynamic physiological data?

Chapter 2 – Background

In this chapter, a closer look will be taken at the different aspects of that are involved in the physicalization of dynamic data. In order to gain a broad understanding, this chapter consists of three parts. The first part describes data physicalizations in general, the second part presents the use of sensors for emotion recognition, and the third part describes the potential application areas of dynamic data physicalizations.

2.1 Data physicalization

As our surroundings are increasingly providing us with different types of visual and cognitive impulses, our bodies get more accustomed to them. That results in efforts to explore new ways to convey information to people in a memorable and intuitive way[14]. One of these ways is using physicalizations, a type of communication where physical modalities are used to convey information. An example is the physicalisation created using the modalities of temperature and vibration[37]. In this physicalization, these modalities were used to convey facts regarding SDG's(sustainable development goals), in order to effectively and engagingly advance the communication between citizens and the goals that need more attention. Other modalities that could be used are for example, sound, light, material and shape changing.

According to Zhao and Vander Moere[14], this type of information communication goes beyond the data itself, thanks to the tangible nature of the objects, encouraging reflection on its meaning. This is also supported by Lockton et al.[15], who argues that the use of data physicalization can offer new forms of understanding.

Data physicalization itself can be explored from many different angles[16], but also through many different means, like sound, smell or touch for example[17]. Using these different means open up the use of multilevel interaction with data[18]. This can be seen as something very beneficial, providing the use of data to a broader audience[18]. This is also further supported by Jansen et al. [17], who state that by using different modalities, people with for example a visual impairment could also interact and gain understanding of the data. Whereas it used to be that the use of data to base certain conclusions or understandings upon was only accessible to people that studied in the field of data science, using different means to convey and explore data, the use of data becomes more accessible to a broader audience[18].

Another factor that aids the rapid development of new types of physicalizations are the recent advancements in the field of digital fabrication, actuated tangible interfaces and new types of sensors[17]. These developments in technology have a booster-like effect on the domain of data physicalization, as with every new technology that is being developed, new ways of representing data can be created. Although some of the benefits have already been mentioned in the previous parts, there are a lot more. One of them is the leveraging of our perceptual exploration skills. As humans have a highly complex sensorimotor system, information extraction from the real worlds around us can be efficiently achieved[19]. Another one is that there is anecdotal evidence that the use of the physical instead of digital means can actually mean that people can share data more efficiently[17].

However, next to the benefits, there are also challenges regarding the use of data physicalizations. As previously mentioned, the advancements of technology have enabled new forms of data physicalization, yet it is still at the start of what is really possible in this field, as there are still numerous limitations with those emerging technologies[17]. Another limitation that can become prevalent is the encoded variables[17]. When using different modalities such as different materials or heat for example, the encoded variable might over time change due to the physicalization slightly deforming because of the heat or the wear of the material that is being used. This can severely hamper the accuracy of the installation.

Another challenge lies with the use of dynamic data. Although the use of dynamic data in a physicalization would be able to visually reflect changes in datasets, it should also be able to offer changes in data operations and data settings and should show transformations over time[17]. Yet very little research is done in how this would be implemented in a physicalization. Challenges in making sure the changes in the data are properly conveyed are therefor still unsolved[17].

Yet there are already examples of the use of dynamic data in combination with a physicalization. One of the more well-known ones is the Liana[20], a slow exercise feedback physicalization. The system is known as a shape-changing art piece, used to convey running data by using physical means through a slow feedback mechanism. Their physicalization takes the GPS data which represents a ran route by a person. After multiple days of running, similar routes might be run, or new routes might be explored. Liana uses small wooden pegs to move out in order to display the number of times a certain route was ran. The system can be easily reset by pushing the pegs back down. When a new route was chosen, a new peg will rise, while when a similar route was run, the peg used the previous time for that route will be raised further. It does however take some time between the running and the art-piece actually showing the change to the user.

2.2 Sensors for emotion recognition

it is important to mention that in studies which involve emotions, two different ways of describing emotions from data are used. One of these is the discrete emotional model. This model uses specific emotions or experiences that have been described. An example could be a binary model, where only happiness or unhappiness are described. Yet a 5 or more specific emotion distinguishment could also be used.

The second model is the dimensional model. In this model, it is not about the specific emotions, but about plotting the described emotions on a graph[21][22][23]. Valence and arousal are usually used as the graph’s axes, where valence is the described positivity or negativity of the emotion and arousal the intensity of the emotion that is being experienced[21][23]. An example of such a graph can be seen in figure 1.

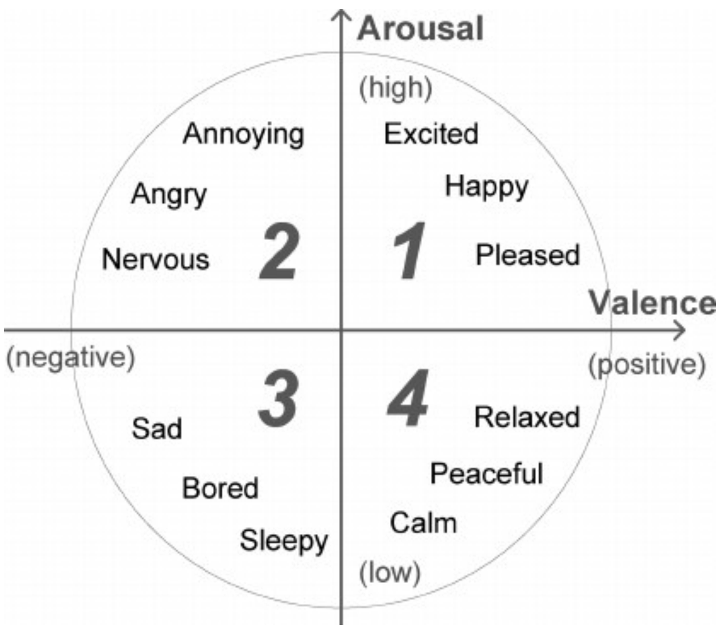


Figure 1: Dimensional emotional model [24]

The type of metrics from different sensors relate to what type of emotions have been explored in a literature review looking into the use of different sensors for recognising emotions[38]. One of the results was a graph giving a full overview of different types of sensors, and how often they were used for recognising different types of emotions. This was visualised in a Sankey diagram, which can be found in appendix A. When looking at the diagram, the most common used sensors regard the use of GSR, in full Galvanic skin response, and cardiovascular measurements, such as HR, PPG, HRV and ECG.

As mentioned in chapter 1, introduction, the goal is to use smart wearables to collect the data. For this the use of GSR and cardiovascular sensors works well, as these are usually applied in the form of wearable devices. Their function and the physiological states these sensors can measure does differ. However, Gasparini et al.[31] and Saganowski et al.[32] state that combining sensors to create a multimodal approach can affect which physiological states can be measured, combined with the fact it increases accuracy. This combination of sensors then can be integrated in a wearable to create a system which can accurately measure several types of emotions.

Cardiovascular sensors is a group of sensors which could be applied in such combination, as it has many different metrics of measurements, from simple heartbeats per minute, expressed as HR(heart-rate), to ECG(electrocardiogram) or PRV(pulse rate variability). All these different techniques provide different information regarding the heart and the pulmonary system. According to Yuda et al. [33], it is of great importance to be aware of the fact that there are many differences. Not only the difference in what physiological data is measured, but care should also be taken if the chosen sensors fits the application and can do unobtrusive measurements in the application environment.

The physiological states that can be categorized from cardiovascular measurement depend on the type of cardiovascular measurement done, which has been stated in the previous bit, relating to the research done by Marc. Shu et al. [25]. They mention that the simplest form, heart rate, can provide basic information on arousal. When looking at more specific emotions, slow variations of PRV can be connected to anger, while fast variations of the PRV can be connected to happiness or a relaxed feeling during work [33]. A full list of different types of cardiovascular measurement and the emotions that can be recognized based on these measurements according to the table from Yuda et al. [33] can be found in appendix B. Looking at this table, it can be concluded that the broad spectrum of different cardiovascular measurements means most emotions can be measured accurately when selecting the right sensor.

Another category of sensors, EDA sensors, also known as GSR (Galvanic Skin Response) sensors, are based on changes in the sympathic nervous system. The sympathic nervous system can cause changes to the skin conductance. This is done using sweat glands stimulated due to changes in the sympathic nervous system, which makes the skin moist resulting in the skin to be more conductive [7]. There is a one-to-one relation between sweat-production and GSR measurements[8]. Therefore, GSR can provide insights in the changes in the sympathic nervous system. As wearable technology can be applied close to areas where these skin changes can be measured most accurately, GSR sensors are very suitable for wearable applications.

GSR offers general information regarding the physiological state of a person. Basu et al. [8] mentions GSR can offer insight in arousal and valence. This is affirmed by Gasparini et al. [31] and Shu et al. [25]. However, GSR on its own cannot provide individual emotions classification options, as more data from other sensors is needed in order to achieve this.

There are two different parts of GSR, with different outcomes and thus conclusions that can be made based on them. The tonic part, which is the slow changing part based on the skin conductance level, and strongly depends on the subjects' physiological characteristics. Next to the tonic part, there is the phasic part, which is the rapid changing part, based on skin conductance response and changes due to specific stimuli [31]. When applying GSR sensor technology in a wearable, this difference can influence the resulting physiological state that is recognized and should therefore be taken into account.

There are three key advantages for both GSR and cardiovascular measurement for emotion recognition. First of all, the use of physiological data-based emotion recognition is less affected by the influence of external factors[7]. Secondly, both sensors can be found in consumer electronics, like smartwatches for example. This increases the chance of access to those sensors, as well as the fact that people are more accepting to smartwatches, as they are used to wearing one [32]. However, it is important to keep in mind that not all cardiovascular measurements can be done by wearables. Finally, this integration into products like smartwatches also means the products can be used for application outside a lab environment[7].

Next to these common benefits, GSR measurements come with the benefit of having been applied often for emotion recognition applications. It has therefore become a common standard, which means that information regarding the use and possible outcomes have been abundantly described. According to Feng et al.[34], this abundance of information makes the use of GSR very accessible.

There are three common disadvantages when using EDA or cardiovascular measurements. The captured physiological signals contain noise and might have problems with baseline wander. This means proper pre-processing is needed in order to get reliable results [34], [32],[8]. This makes that implementation of algorithms and enough computing power are constraints to the application of the sensors. Next to this is the second issue that is connected to external application. The lab-environment, in which background noises are fairly limited are not the best location for obtaining genuine emotion [32]. Therefore, application outside the lab, where genuine emotion stimulating events occur are preffered, which come with more noise and require a mobile setup. A third issue is the similarity of data which can be caused by different factors of the human body. Hovsepian et al. [35] point out that physiological arousal indicative of stress response can easily be mystified by movements of limbs, changes in posture, and physical activity. In this case it referred to GSR measurements, however, Wijasena et al.[7] point out this also applies for cardiovascular measurements. Next to that, placing the sensor at the wrong spot on the human body heavily influences the reliability of the measurement. This means that great care should be taken when placing the sensor, analysing the data, and extra data might be required to properly classify the data into separate emotions.

Next to these common disadvantages, GSR has an extra disadvantage, as signal processing of the GSR signal is key for a proper result, as the two different parts of GSR data need to be interpreted separately. That is why the raw data from the GSR needs to be processed in order to split the signal in its tonic and phasic component. Due to those two parts having a different cause [31], both signals have a different influence on the final result of the emotion recognition.

A key disadvantage that only applies to cardiovascular sensors, is the issue of the different types of cardiovascular sensors having a different influence on the final resulting emotion measurements. All these different types come with different ways of using, and different places at which those sensors can be applied. This results in some cardiovascular sensors being less unobtrusive than others, while accuracy might be improved [32].

When looking at smart wearables that are outfitted with the necessary sensor technology for emotion recognition, there is one major issue regarding most of commercially available wearables. As has been found by Kammoun et al.[26], a major challenge is getting the data in real time from the wearable, as most of them have their own secure platform which only allows retrieval afterwards via their API, or do not have the necessary hardware to pass the data in real time to a cloud based server. This makes that most commercially available wearables are not suitable for real-time data physicalization

This is also what was found when looking into the use of the Empatica E4. This is a medical grade research wearable, especially developed to allow researchers to use physiological data during their research. However, when the device was tested during this research, it was found the Empatica E4 had no way to retrieve the data in real time from the wearable. It could only be accessed after a session with the wearable was complete.

2.3 Potential application areas

A final key part of the project are the potential application areas. The use of near-real time, easy to understand data that can be interacted with in an intuitive way has many different possibilities, and in this chapter, several of those will be further elaborated upon.

As has been mentioned in chapter 2.1, Liana[20] has been specifically designed as a self-reflection tool. According to their research, there is an increasing trend in the use of feedback devices that are based on data provided by wearables. Whereas the use for Liana is in the category of sports, another area where the interest into self-reflection is growing is stress. According to Peake et al.[28], there is an exploding increase in the amount of wearables offering features to monitor stress. These wearables will inform the user about their stress levels in the form of graphs and data, and some might even give some advice. However, as has been mentioned previously in chapter 2.1, people with no background in data science can interpret data that is presented to them in a different, physical modality much easier. That is where the use of a physicalization of dynamic data can offer a solution, providing broader access to the use of reliable data to do reflective tasks, may it be self-reflection or someone else providing reflection on the data and giving advice to the person wearing the wearable.

Another case in which the use of easier access to data regarding someone's physiological state might help is in the field of product development and testing. A case in which this has been applied is collecting data through a wearable and adapt a VR(Virtual Reality) environment to prevent someone falling ill due to cybersickness[29]. However, product-developers are not always fully trained in using physiological data as a way to deduct the participant's physiological state. This might however be needed to find out how someone is really feeling during the interaction with one of the products. So a physicalization that can give a product-developer easy access to a reliable form of data on which argumentation could be based would be helpful. Especially in the field of HCI, such a tool might offer researchers a new form of evaluating their product based on reliable data. This is also stated in the research done by Alfaras et al.[30], in which they were able to use bio-data in order to make new discoveries regarding the designing of a product.

A final example of an application area for the use of physicalization of dynamic data could be in situations where people are not able to communicate how they really feel, but in which it might be of great importance to be aware of their physiological state, in order to alter the response to them or the environment around them. An example could be the previously mentioned VR environment, where a developer might want to change the environment if a person starts to feel really uncomfortable. A completely different application might be in situations for people with autism, who might not be able to communicate how they really feel, but the caretaker or family members might want to know, in order to change their response to the person with autism or alter the environment that person is in at that moment.

These are just some of the potential application areas, and many more can be found, and might even be found after the system is fully operational.

Chapter 3 – Methods and Techniques

In this chapter, the methods and techniques will be explained that are used to develop and test the wearable and physicalization. The core method used during the full process is the Creative Technology Design process[4]. Using the Creative Technology Design Process means there will be 4 phases: ideation, specification, realisation, evaluation. Each of those phases will contain their own methods and techniques to gain a satisfactory result and a good starting point for the next phase. A graphical overview of the steps can be found in figure 2 below, with elaboration on each of the different phases.

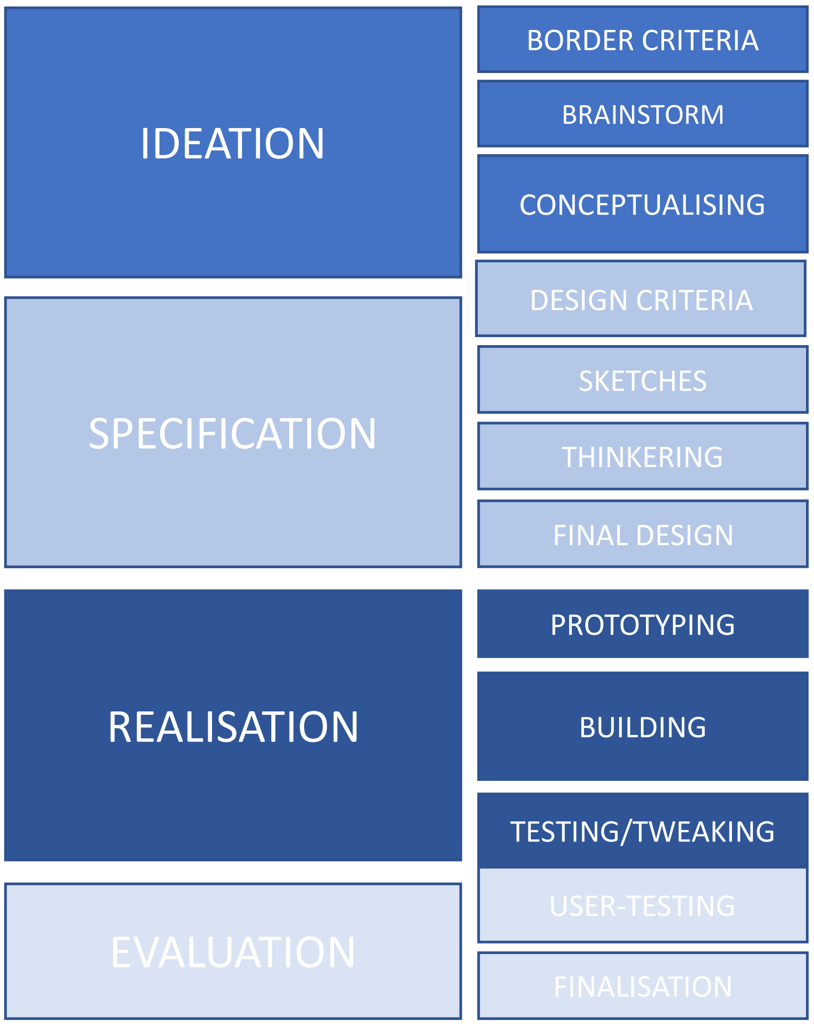


Figure 2: Graphical overview of the full project process

3.1: Ideation phase

In the ideation phase, the main technique that will be used is the mind map techniques. In this technique, different mind maps will be created in which all steps, from the first brainstorming to the final product specification will be further elaborated upon. This in order to create a clear overview between the different stages of the ideation, and also to see where connections could be seen. It can provide an overview in one large picture, providing the possibility for a researcher to look back at previous steps, being able to directly involve previously gained knowledge in the next steps. This phase will start with setting a few bordering criteria, a set of points to which the design should adhere, but are not strict yet. They provide the researcher with a first set of points from which this phase can start. Then, a broad brainstorm session regarding the topics of physical means, as well as digital techniques to display data will be conducted. Based on this, a first round of concepts will be generated, in which the possible different ways explored earlier will be combined with some of the first set of design criteria in order to come up with a load of rough concepts regarding the physicalization. During this concepting phase, inspiration by looking at previous products and other sources of inspiration will be used to create a broad spectrum of concepts. During another session, a final concept will be selected, and a round of creating actual designs for the physicalization based on the selected concept will be done. Again, inspiration from external sources will be used in order to provide inspiration for several of the designs. The conclusive part of this phase will be a rough design of the full product. In this design, several factors have already been considered, based on the bordering criteria that have been set at the beginning of this phase.

3.2: specification phase

As mentioned in the previous phase, this phase will start with a rough design. With this design, a set of new criteria is being developed, further delving into the specifications the installation should adhere to. This will also involve the wearable. Afterwards, a full list of criteria to which the installation should adhere is developed and a final design and component list of the installation is made. In this phase, some prior testing with components in order to check their function will be done, to see if they meet the requirements and thus can be used in the final design. This phase will conclude a final design for both the wearable and the physicalization itself, as well as an overview of how the internal structure of the full system will work. This internal structure will become the backbone of the system, also known as the pipeline.

3.3: Realisation phase

When the final design has been developed in the previous phase, and most of the criteria have been met, the installation will be built. In this phase, this will start with developing the wearable and the structure of the pipeline that connects the wearable and the physicalization wirelessly and in near-real time. As this still has to be fully tested, this phase will start with creating a prototype of the pipeline, which will then be adapted until a working system is created. This working version will then be integrated into the rest of the system. Next to that, the physicalization will be built combined with the final version of the watch. All components will be tested separately and will be integrated in stages, to ensure that all components will work seamlessly together. Staged integration will also ensure that issues which arise in certain components of the full system can be handled separately, and thus saving time developing the final product. Another benefit is the fact that the final product will consist of several components, making future improvements or redevelopment of a single component possible. The result of this part of the process is a fully working system which can be used for a user-study as part of the evaluation process.

3.4: Evaluation phase

The final stage of the development is the evaluation. In this stage, the final product will be tested with participants in the form of a user study. The full design of the study will be described in chapter 7-Results. The outline of the research will consist of participants interacting with the installation, while the wearable will be gathering data. All stages of the interaction and the performance and functionality of the system will be tested. Based on the user study, as well as the findings during the several stages of development, the final results of the system will be presented.

Chapter 4 – Ideation

The goal of the ideation phase is to design the full product. However, in order to get to the best result possible, several steps have been taken to ensure all areas of the product have been explored, and each decision made as part of the design can be supported by proper argumentation. These steps include setting the bordering criteria, brainstorm, and conceptualising.

4.1: bordering criteria

As part of creating a starting point, the first step that has been taken is creating a mind map in which several criteria have been put into as a guide for the next steps. It should be noted that this part of the ideation process is mainly focussed on the physicalization itself. So the criteria set out only apply for the physicalization. This mind map with the criteria can be seen in figure 3.

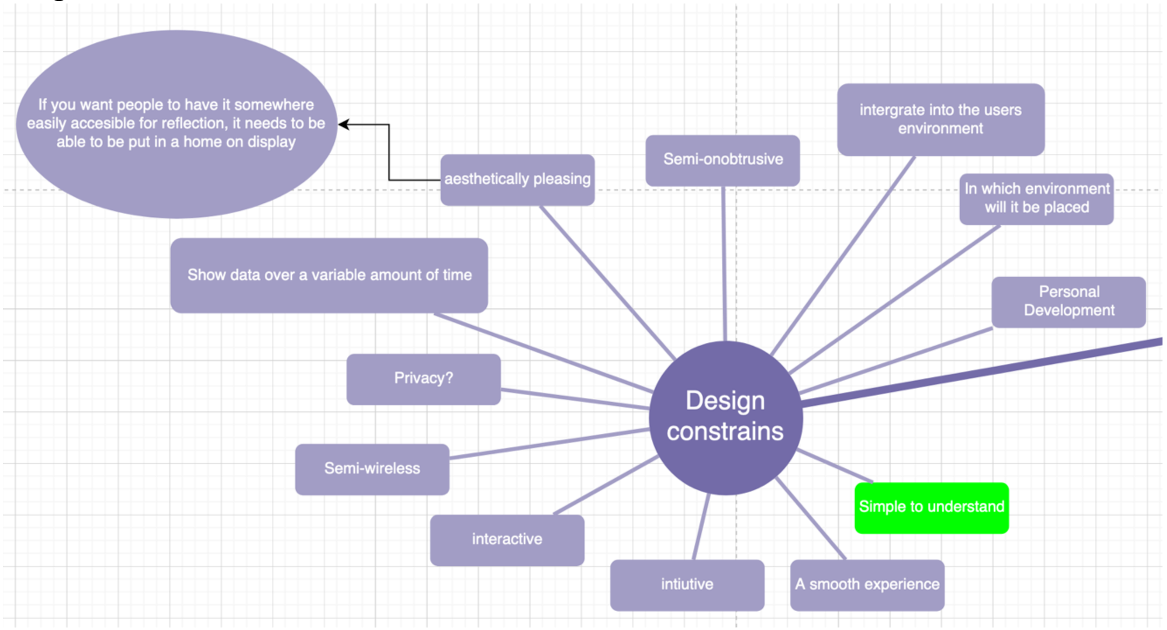
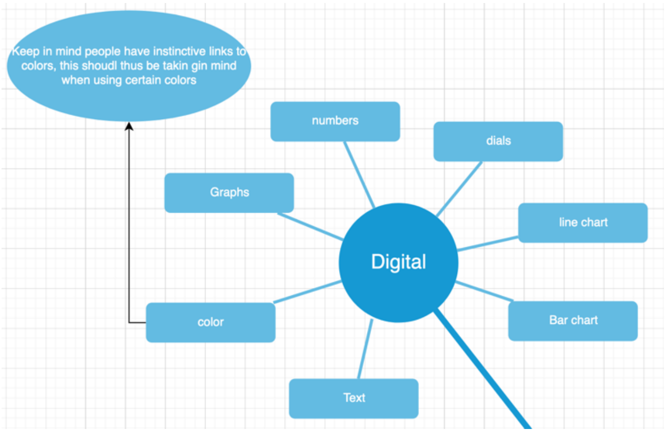

If you want people to have it somewhere easily accesible for reflection, it needs to be able to be put in a home on display
aesthetically pleasing
Semi-onobtrusive
intergrate into the users environment
In which environment will it be placed
Show data over a variable amount of time
Personal Development
Privacy?
Design constrains
Semi-wireless
Simple to understand
interactive
intiutive
A smooth experience

Figure 3: Design criteria mind map

These criteria have been developed after having talks with the supervisors combined with points which have been found during the background research. One of the key points, which has also been marked in green, is the fact that is should be simple to understand. As also has been mentioned during the background research, a physicalization should be able to provide a new way of understanding the data, making it so more people can understand. Especially in this topic where the data could actually offer useful information to improve someone's lifestyle or could provide insights that could lead to changes to a prototype in the process of developing a new product, a system which can be used and understood by most people is a key concept. Another key point is the integration into someone's environment, as this installation could become part of someone's daily life. That is why the aesthetically pleasing criteria is also part of the mind map. Another reason integration in the home environment is key is due to another factor that can be found in the mind map, privacy. Having a system which can show someone's personal data might not always be wanted by the user, so creating a system which will blend into the environment around the person would make it less obvious to someone unaware of the product what it is actually showing. This gives the user of the product control over their own data, which is a key point in privacy.

4.2: Initial brainstorm

With the bordering criteria set, it is important that all the possibilities related to physicalizing data are explored, as this could lead to more concepts generated during the next phase, as well as making sure there is a complete picture of what a physicalization could entail. This is done by looking at physical means, as well as digital means. The results of this initial brainstorm can be seen in figure 4 and figure 5 below.


Keep in mind people have instinctive links to colors, this shoudl thus be takin gin mind when using certain colors
numbers
dials
Graphs
line chart
Digital
color
Bar chart
Text
Figure 4: Digital modalities mind map

In figure 4 on the left, the results of the brainstorm regarding the digital means can be seen. There are both abstract terms and more concrete terms in the mind map. Each of those could be a means to present data. As can be seen, it is very broad, which is done intentionally, in order to not limit the conceptualization in the next phase. Something that should be noted is the fact that with the physical modality colour, it should be taken into consideration that colours can have an instinctive meaning[5], and thus care should be taken when colours are chosen for the use in an application.

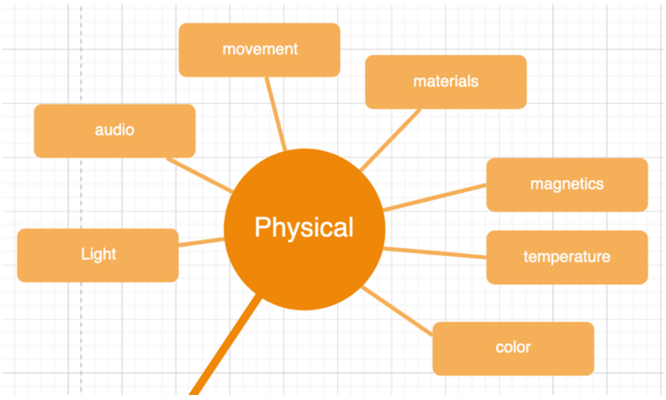
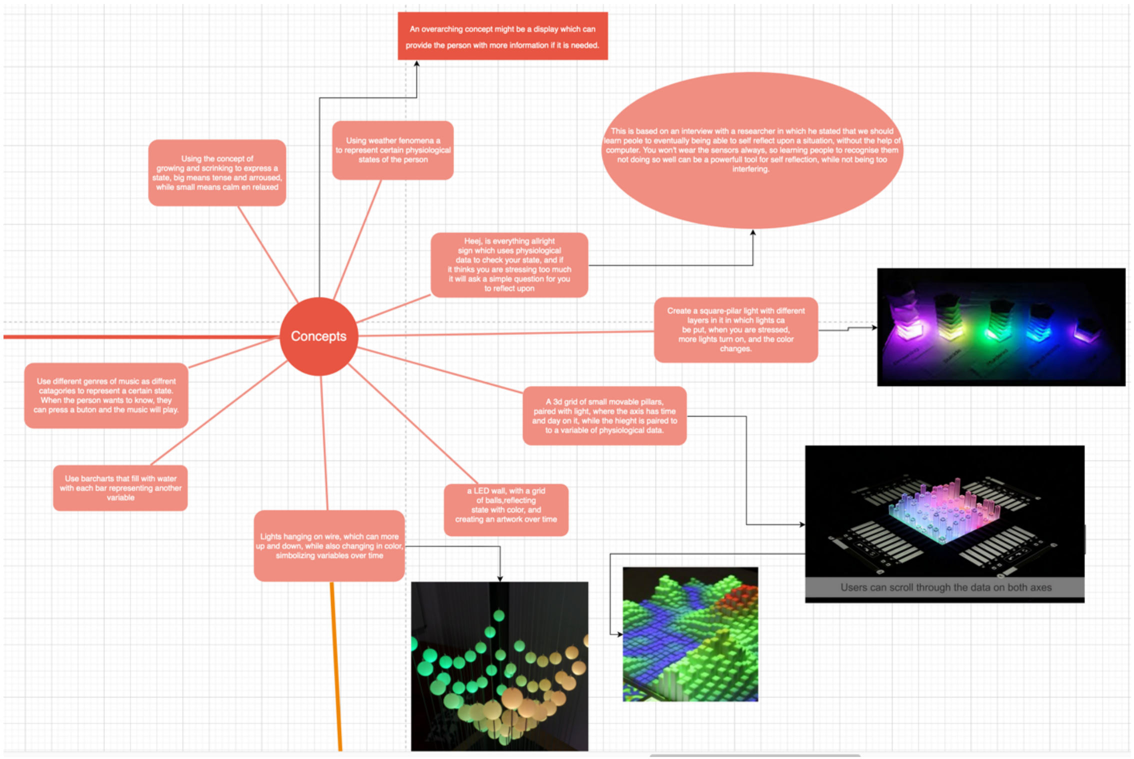


Figure 5: Physical modalities mind map

Next to the digital means, physicalization is mostly regarding physical means. Figure 5 shows the result of a brainstorm regarding the applicable physical means for the final product. Most of these terms are again very broad, but this is done for the same reason as mentioned before. This should provide a basis for the conceptualisation phase, but should not be a limiting factor.

4.3 Conceptualising

The mind maps in figure 4 and 5 as well as the bordering criteria from figure 3 form the basis for the creation of concepts. This was again done in the form of a brainstorm session, in which some criteria were taken, combined with some physical or digital means, in order to generate concepts. In this phase, inspiration from external sources like other physicalization that have been developed, but also ordinary things were used in order to come up with multiple extra concepts. Also, an interview was conducted with someone doing research into conveying the message of stress to a person. This all resulted in the mind map which can be seen in figure 6.


An overarching concept might be a display which can provide the person with more information if it is needed.

Using the concept of growing and scrinking to express a state, big means tense and arroused, while small means calm en relaxed

Using weather fenomena a to represent certain physiological states of the person

This is based on an interview with a researcher in which he stated that we should learn peole to eventually being able to self reflect upon a situation, without the help of computer. You won't wear the sensors always, so learning people to recognise them not doing so well can be a powerfull tool for self reflection, while not being too interfering.

Heej, is everything allright sign which uses physiological data to check your state, and if it thinks you are stressing too much it will ask a simple question for you to reflect upon

Concepts

Create a square-pilar light with different layers in it in which lights ca be put, when you are stressed, more lights turn on, and the color changes.

Use different genres of music as diffrent catagories to represent a certain state. When the person wants to know, they can press a buton and the music will play.

A 3d grid of small movable pillars, paired with light, where the axis has time and day on it, while the hieght is paired to to a variable of physiological data.

Use barcharts that fill with water with each bar representing another variable

a LED wall, with a grid of balls,reflecting state with color, and creating an artwork over time

Lights hanging on wire, which can more up and down, while also changing in color, simbolizing variables over time

Users can scroll through the data on both axes

Figure 6: Initial concepts mind map

In figure 6, several short descriptions of concepts were written down. Some including physical means like water, others including digital means such as light and thus colour. Some combining those two. The pictures that can be seen were inspiration sources for the concept described. Some of the concepts might be a bit more elaborately described, this is due to this concept already being more clearly imagined. As mentioned, not all criteria set out in the beginning are adhered to in each of the concepts, which is no issue, as further specification will happen in the next phase of the process, in which several round of revising would be done.

Figure 6 formed the basis of a longer discussion in order to find the best one as the basis for the next phase. In this, all concepts were discussed, with the help of the criteria mentioned earlier, as well as some more explanation when needed. During this meeting, the decision was made that the concept in which lights were used that would change colours based on changes in the data would be chosen. This had several reasons. First of all, it would be something which could be integrated into an environment fairly easily, as everyone needs a light around them, and most people have light both for practical reasons, but also for decorative reasons. This could offer possibilities to create a unique design. Furthermore, previous experiences with creating physicalization from both the researcher as well as the supervisor was used to dismiss concepts. An example was the water concept, due to the reason that it is very challenging to make sure something is watertight. The choice for light could also mean that light can change to protect a user's data, targeting the criterium of privacy. This would mean making different modes in the design. The factor of light, and especially colour having a meaning means that people intuitively can link factors to colour. This would make it easier to understand data when it is linked to colour. For example, if there is a high spike in a certain physiological state, the colour could change to red if the spike is not preferable. As people intuitively link red to something bad[5], this could be useful in making the product easier to understand.

With the selection of the concept complete, the next step would be another brainstorm session, as the selected concept needed to be realised in a product design. For the phase, again a mind map has been made, which can be seen in figure 7.

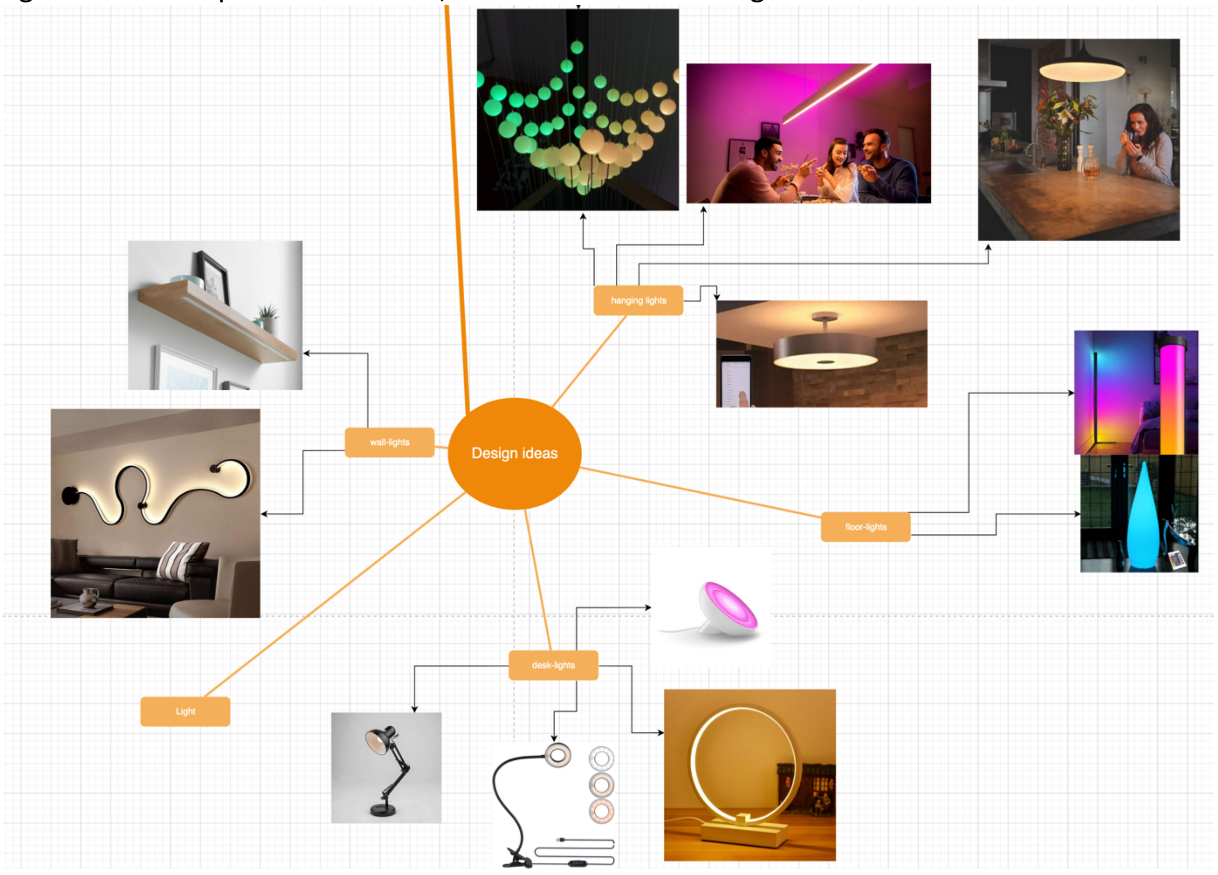


Figure 7: Design ideas brainstorm mind map

Similar to the first round of conceptualising as can be seen in figure 6, inspiration from outside has been used as a basis for this mind map, resulting in a mood board. Different forms of light product have been explored, resulting in several first sketches. These can be seen below in figure 8.

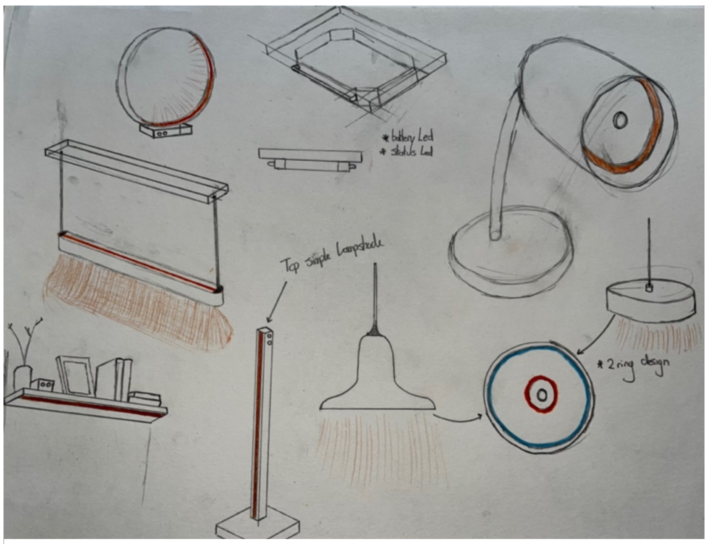


Figure 8: initial design ideas sketches

In each of those sketches, the red ring or strip is the LED ring or strip which will use colour to display data. The different types of light from the mood board have been converted to the concept in this case. With this set of new sketches, another discussion was held in which the designs were discussed with the supervisor, considering the previously set criteria, and looking at what would become the final rough design. The final choice fell on the ring-light which can be seen in the top left corner. This design was chosen for several reasons. First and foremost, the fact that the shape of the light itself is round and has a top and bottom, making it fairly easy to map the light indication, in this case a datapoint, to a timestamp. This is because the shape is similar to a clock. The second reason was the fact that the physicalization would fit in multiple different environments. It could be set on a desk in an office, or around the house. But it could also be used as a wall decoration, with a small shelve in the circle to put something like a plant on. This would make the factor of integrating it into someone's natural environment easy. This would then also make it less obvious and would thus blend in easier. Both factors help with protecting the privacy of the person's whose data might be on display, another factor that was considered. Also, the installation looked quite aesthetically pleasing, with a sleek, modern design that could serve multiple purposes. All these reasons combined resulted in the ring light becoming the design choice of the physicalization.

4.4: conclusions

As can be seen above, this phase set the first steps towards the final full system, mostly focussing on the physicalization, as this was the part of the system where there was a lot of freedom to create something unique. The reason why the wearable part is not included in this part is due to the design of the wearable being very limited. Limitations like the electronics choice and the fact the wearable had to be as small as possible. This and the rest of the design stages of the wearable will be discussed further in the next chapter.

In the process of designing the physicalization part of the system, it started with a black canvas, in which first some borders were created in the form of bordering criteria. From that point, a broad brainstorm was done to make sure a good basis of knowledge was created from which concepts for the design could be created. These were created in the next step, together with the bordering criteria. After discussing the best concept with the help of those criteria, a new round of gathering information and inspiration for generating designs from the previously selected concept was completed. From the resulting mood board, some initial sketches were made. With the help of those sketches, a final design was chosen, which will be further specified in the next phase.

Chapter 5 – Specification

This chapter describes the details of the physicalization and the wearable to acquire physiological data required for the physicalization. The goal of this phase is to further develop that sketch into a full design and component list with the help of a set of design criteria, which will be set during this chapter with the help of the initial set of criteria developed in chapter 4. The pipeline, which has already been briefly mentioned, will also be further specified in this chapter. This will regard the way data is processed, transmitted and stored between the two component, and will form the backbone on which the full system relies.

5.1: The physicalization

The first component of the full system is the physicalization itself. Based on the result from chapter 4, this will become a light-ring that can be used as decoration as well. For the physicalization, there are several criteria the final design should adhere to.

- **The physicalization should be intuitive and easy to use**
  This is a key point in the operation of the system. As has been discussed earlier, the full setup should be able to provide insights without having to understand complex data or the need to take complex steps. That is why the operation of the physicalization itself should also be simple.
- **The data should be easily mappable to a time-stamp**
  As the user will also need to be able to see changes over time in the data, it is of key importance that the data can be analysed over time. The physicalization should make this possible.
- **The user should have the control over data, so the physicalization should offer multiple modes for the user, for both the types of data it will display, as well as an off mode**
  As mentioned in the previous part as well, a key point should be that the user can turn the system off in order to not show their personal data. The system should offer a way to easily achieve this with the physicalization. Next to that, all other modes of displaying the data should be directly available from the physicalization.
- **The physicalization should easily fit in the environment of the user**
  Another aspect is the integratability into someone's environment. As explained earlier, for reflection the data is needed more often, and integration into the environment can offer this, making someone aware of the data, and thus it can provoke a user to act upon it. Next to that, this criterium is to make sure it can be used in multiple environments, in a home environment just as a decorative light for on the wall or on a cupboard, or in the office on a desk or even as part of a research lab.
- **The physicalization should offer a smooth experience**
  This is related to both the use of the installation, as well as the inner workings. The system should operate autonomously in the environment with little need to interfere. Furthermore, the system should be developed in such a way that it will automatically start and set the installation up, as well as make sure if something in the code goes wrong, it tries to reboot. It should also just be 1 power wire, so that there is no hassling with cables.
- **The user should be able to understand and interpret data**

The goal is to make complex data more accessible, so that should therefore also be a core value to which the installation should adhere.

With this set of criteria, and the rough design that was generated in chapter 4, a 3d design was made. This can be seen in the pictures in figure 9 below.

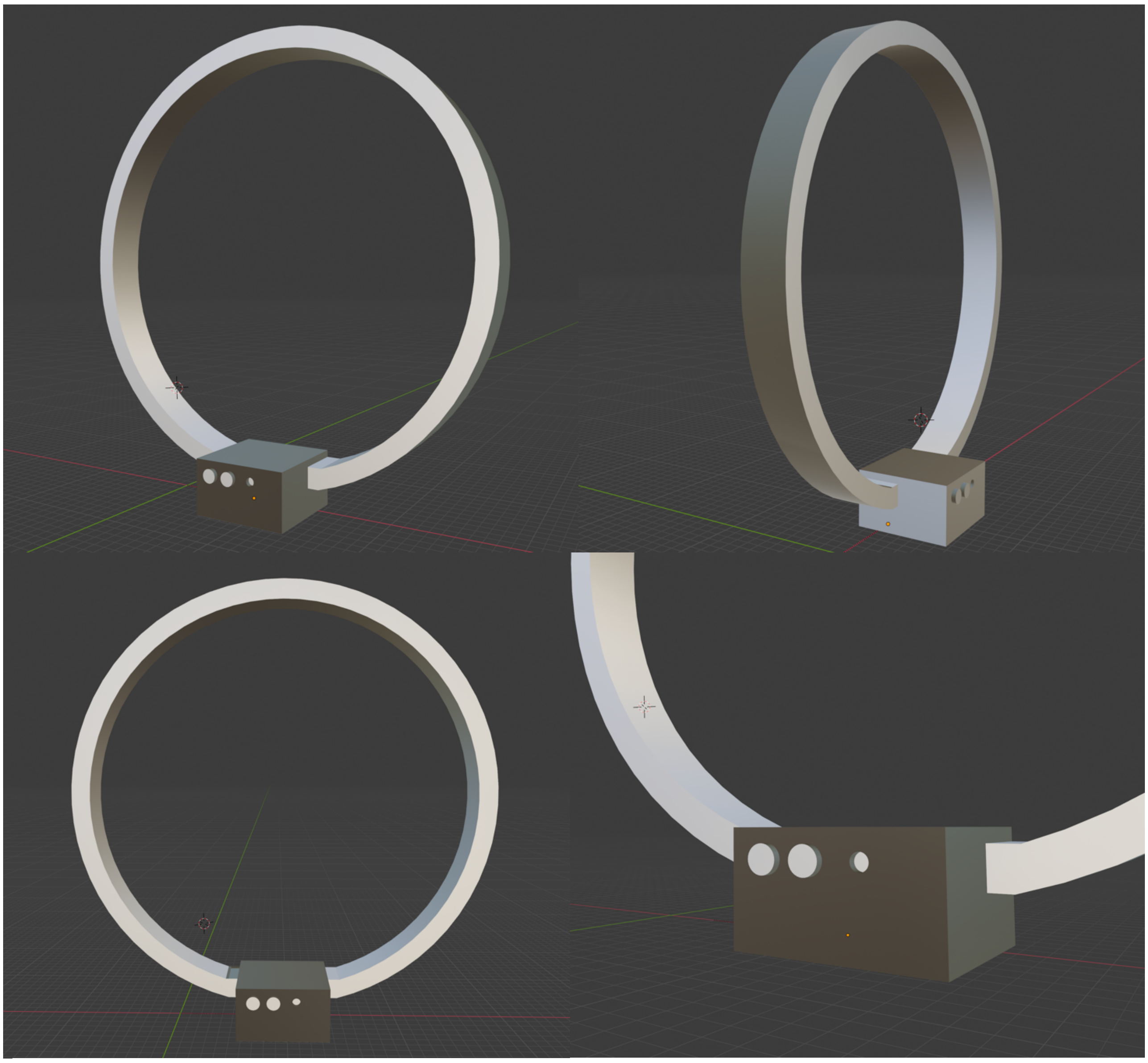


Figure 9: First design renders

With this more-detailed design, some changes have been made from the first rough design. First of all, the base has been enlarged to provide enough room for all the components that it needs to house. Furthermore, the ring now enters the base on the side instead of laying on top. This is so the wires can be easily thread through into the base, making connecting them easier. Also, the base now offers two buttons and a status LED. This status LED will provide the user with an indication in which mode the installation is in.

The next step in the specification of the physicalization was to create a list of components that needed to be used in the system. This list would also be based on the criteria set out, as well as the availability of the components. In Figure 10 below, each component is specified, with a short explanation of what the purpose of it is in the physicalization

Component	Purpose
Warm White COB LED strip	This is the strip which will be turned on when the data representation is turned off, as it provides the installation with a mode in which it can also function as a normal decorative lamp. This component will be placed on the inside in the ring, creating a warm glow on the object on the platform in the middle, as well as some light for the surroundings.
WS2812B pixel strip	This is an LED strip which will be fitted into the channel inside the ring to provide the colour and position of the specific datapoints. The pixel density of the strip is 60 LED's per metre, with a total of 90 LED's fitted inside the physicalization. This would yield a maximum accuracy of one datapoint per 40 seconds.
5v 10A Meanwell power supply	This is one of the two power supplies in the system, this one being the main supply, as it provides the power for the WS2812B LED's as well as the main computer fitted inside the physicalization. The 10A supply was chosen based on the maximum load that could be needed for the LED strip, which is 5.4 amps, combined with the 3 amps needed for the main computer. That leaves 2 amps of overhead power, which should be a sufficient margin.
24v 3A Meanwell power supply	This power supply is there to provide the power for the warm white COB LED strip. The LED strip will need a bit more then 1A when fully lit, so again, there is a margin of 2A, which is plenty.
Raspberry Pi 3B+	This is the main computer of the full installation, chosen because it would offer a small, capable computing solution that could be programmed with python, and offers simple GPIO integration to control the LED's as well as take inputs from switches. The computer does have a graphical interface that can be accessed,

	and is useful when bug fixing the system, but can also run without being connected to any output display. It will also offer plenty of computing power in case AI or a resourceful algorithm is integrated into the physicalization.
Relay board	For the control of the COB LED strip by the raspberry, a simple solution using a relay to cut power to the power supply was chosen. To do this, a relay is needed.
24v 3A LED strip dimmer	As the COB LED strip only needs to provide light in the off mode, a simple dimmer was put in between the power supply and the strip to limit the output amount of light to make sure it would resemble a decorative light not radiating too much light out. To do this, a simple in line dimmer was chosen.
2X on-off toggle switch	For the final design, the whole physicalization will be controlled by two toggle switches. This deviates from the previously presented design, but has been found to be the best solution as will be explained later.

Figure 10: Physicalization component table with explanation

With the component list ready, a final design could be produced with takes into account the specific dimensions of the components listed above, to make sure they would fit properly inside the physicalization. This final design can be seen in figure 11 below, and the design drawing can be found in appendix C.

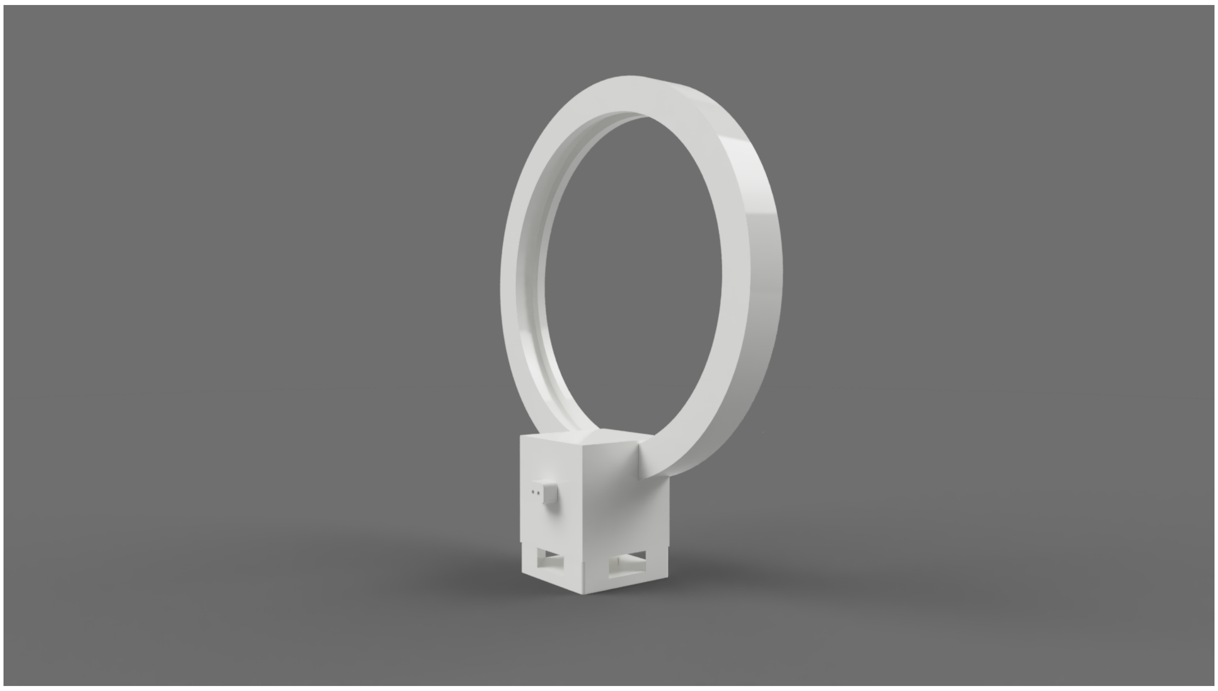


Figure 11: Final design physicalization

5.2: The data physicalization pipeline

Before talking about the wearable, another part of the installation which needs to be properly specified is the pipeline. This connects all the components together and makes sure it becomes one system. This "pipeline" consists of several protocols and digital systems used to make the physicalization work as designed. These are mostly focussed on the processing, transmitting, and storing of data. It also defines the coding languages used, as well as specifying the development platform for the different components. This could therefore, as has been mentioned in the introduction of chapter 5, be seen as the backbone of the system. However, due to the general outline used in the pipeline, it will also form a blueprint for future dynamic physicalisation, as a starting point for development.

To create such a pipeline, a set of criteria to which the pipeline should adhere is needed. These are listed below with a short explanation of why those are important parts.

- **Full autonomous operation**
  The system should start working as soon as the physicalization and wearable are booted up, and no intervention should be needed in order to get the system working. Everything should start automatically and if something is not working, it should automatically retry.
- **The system should be based on a cloud-based database as a storage medium**
  This is to keep data safe in case some part of the system fails. Then the data is still accessible. It also offers online access to the raw data of the wearable.
- **The pipeline should provide the possibility for a full wireless system**
  As the system consists of a wearable which should be able to function in a persons' daily life, it should not be tethered to a physicalization. The person should have the freedom to move and do their thing.
- **The pipeline should make it possible to get a near-real time physicalization of the data, so a constant stream of data is required**

As the goal is to provide easy access to the data right when something is happening, to be able to adjust upon it, the pipeline should make this possible. Also, quick pull and push commands for the data should be possible.

- **The chosen solution should be implementable into Arduino code as well as Python**

This is due to these two being the languages which will be used for the wearable and the physicalization.

- **The real-time data processing and mapping should be possible**

Relating to a previously mentioned point, the system should be designed in such a way that the data can be real-time processed in order to make sure it can be instantly displayed as soon as it is acquired. Also, it should store metadata like timestamps to make sure the data can be properly mapped to the mapping metric chosen.

With these criteria, a more specified version of the system can be developed. In figure 12.1 a simplified outline of the pipeline can be seen.

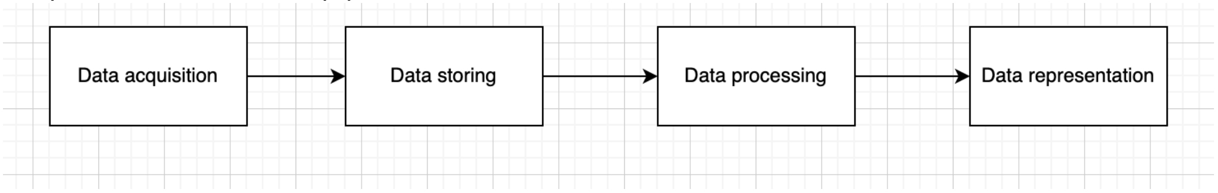


Figure 12.1: Pipeline outline flowchart

This basic outline of the pipeline can be further specified, of which the result can be seen in figure 12.2. Each component in the pipeline has be specified with a colour, where red is the wearable, green the webhost and blue the physicalization. The steps in each of those components have also been elaborated a bit more upon.

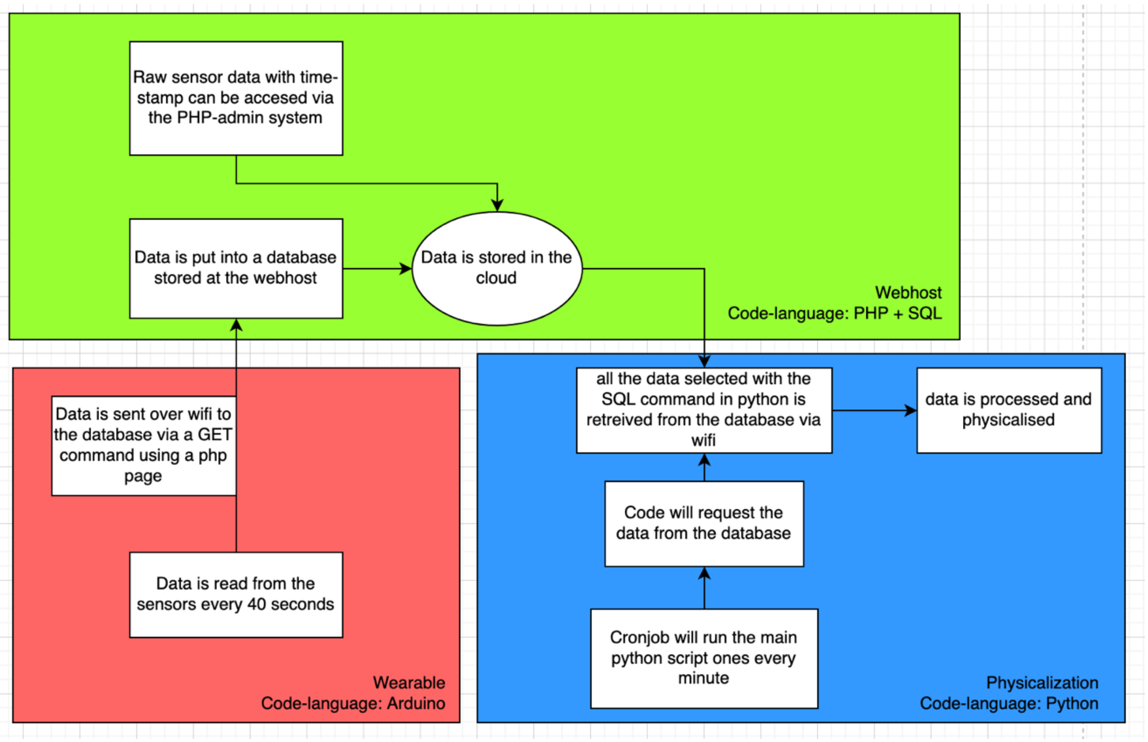

Raw sensor data with time-stamp can be accesed via the PHP-admin system
Data is put into a database stored at the webhost
Data is stored in the cloud
Webhost
Code-language: PHP + SQL
Data is sent over wifi to the database via a GET command using a php page
Data is read from the sensors every 40 seconds
Wearable
Code-language: Arduino
all the data selected with the SQL command in python is retreived from the database via wifi
data is processed and physicalised
Code will request the data from the database
Cronjob will run the main python script ones every minute
Physicalization
Code-language: Python

Figure 12.2: Pipeline explanatory flowchart

As can be seen in picture 12.2, most of the system is developed, instead of using existing components. For example, the database. This system could be made with already existing plugins. This has even been tested, by making use of IFTTT[6] in combination with Webhook, a service which connects the wearable with IFTTT, and Google Sheets, an existing tool for storing datasets. However, this system proved not to meet the criteria, and especially the criterium of near-real time. As IFTTT only accepts a certain amount of requests per minute, the system would never be able to properly upload the sensor data. This will in turn affect the accuracy of the system, as more data means a better picture of someone's physiological state can be created. This will be discussed later in chapter 8 as well. Next to that, the data would then be stored on an out of our control system, which in terms of privacy is also not favoured. Furthermore, choosing to develop our own simple solution meant it can be adapted specifically to our needs.

Another large benefit this pipeline has is the fact that the wearable and the physicalization don't have to be on the same network in order to function. This means data can also be retrieved in external locations, while it is still wirelessly and in near-real time physicalized at the location of the physicalization.

In this form, the data is also never stored on one of the devices, and only kept in the cloud. This is because the physicalization will every single minute request all the data again, from the start point set at the beginning, to the current time of requesting the data. This means that all the data will again be processed by the algorithm and will then be turned into colours and put on the right spot. The data in this case will never be stored on one of the devices keeping it safe for the user. This does however have an effect on the physicalization not being as quick, due to all the raw data having to be processed again. It has to be seen in testing how this affects the performance of the installation.

A final point is the full autonomous operation. This can be guaranteed due to the use of cronjob, an automatic scheduler for Linux based systems. Cronjob will make sure the program will run once every minute and will keep doing this for as long as the system is powered. The downside is that it can only go up to a frequency of 1 minute, not faster.

5.3: the wearable

The final piece of the installation is the wearable, that is providing the data from physiological data for the installation. As mentioned earlier, this had not been taken into consideration in the ideation phase, as there were several constraints this wearable should already adhere to. Another consideration was using an existing wearable as part of the installation. The solution chosen for the wearable part will have to adhere to a certain set of criteria:

- **Full autonomous operation**
  Just like the other parts of the system, this wearable needs to work fully autonomous. This means turning the device on and it should start working and uploading the data automatically.
- **The system should measure the Heart-rate and GSR data**
  These two metrics are needed in order to calculate Valence and Arousal. These two metrics are also fairly easy to measure on the wrist.
- **The system should work fully wirelessly**

This entails two parts, the part of the system uploading the data to the cloud, while also relating to the power of the system needed to be provided by a battery. This means that the system needs a small onboard computer that can provide a WIFI-connection, as this is part of the pipeline. The other part, the battery, should be integrated into the wearable, and needs to provide enough power to make the wearable run for a substantial amount of time, at least 90 minutes.

- **The wearable should have some sort of communication method to transmit its status**

This could be a display or simple LED which can give the battery and processing status of the wearable, as well as other important metrics for the user.

- **The wearable should be able to provide the sensor readings in real time**

This is an important criterium, as the goal of the physicalization is to provide near-real time data from the physiological signals and put this on display.

- **The wearable should be as small as possible**

This is a criterium to make sure the wearable can be comfortably worn on the wrist during the measurements.

5.3.1: Existing products

These criteria are the basis for the wearable that should fit the rest of the system. The first part in further specifying the wearable is looking at existing solutions. Then one certain product comes up, which is the Empatica E4(See figure 13)

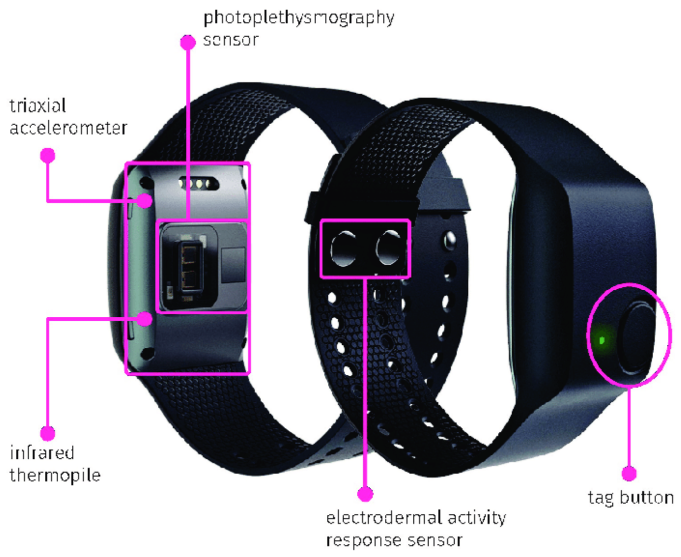

photoplethysmography sensor
triaxial accelerometer
infrared thermopile
electrodermal activity response sensor
tag button

Figure 13: The Empatica E4

This product has all the sensors that are required as specified in the criteria. These sensors are also medical grade, meaning they can provide reliable and accurate data. In order to see if it would fit the project, the wearable has been rented and then tested, together with the software provided with the wearable. After a short testing session, the wearable proved to work properly. However, it could not adhere to one of the key criteria, providing the data in real-time. The system could only upload the data after a session of measurements was completed. This is also in line with previous finding in the background research. Most commercial wearables could not provide their data in real time.

5.3.2: sensors

This meant that a custom wearable needed to be developed. This came with a certain number of challenges. The first part of developing the wearable was getting the sensors to do the measurements. Due to time-limitation and limited access to resources, this meant looking into commercially available sensors that could be integrated into Arduino code. For the GSR this meant there were only a few options, and the one available at the moment was a GSR sensor produced by GROVE.

The heartrate sensor proved to be a bit more challenging. There are multiple different types of sensors available. A few that were available had been ordered in order to test their capabilities. However, soon some challenge arose as heart sensors are particularly difficult to get working reliable and accurate, as they need to be applied at a specific part of the wrist and with specific pressure in order to avoid it being too loose[7]. Also, to get proper data, a lot of processing is needed[8], because the raw data from the sensor would need patented algorithms that can turn the data into a proper heart-rate reading.

With these challenges in mind, a solution was found in a wearable heart-rate sensor which could be connected to an Arduino. The was the GROVE finger-clip Heart-rate sensor. This sensor would provide a stable and pre-processed signal, and already has a case around it which could be worn on the wrist.

5.3.3: the final wearable

With these sensors found, the next phase was to sort out the rest of the components that needed to be included in the wearable. As mentioned in the criteria, it needed to be as small as possible. This resulted into the following component list in figure 14 below.

Component	Purpose
GROVE GSR sensor	A sensor which provides an analog signal for the GSR. In this case this will be related to skin resistance
GROVE finger clip heart-rate sensor	The sensor which measure the Heart-rate via PPG, and sends the processed HR value over I2C
WEMOS S2 mini	The main computing board small enough for a wearable, but powerful enough to offer all the flexibility needed in terms of functionalities of the wearable.
250 MAH Lipo	A battery that provides the power to the wearable
WS2812B LED	A single LED to give the user the status of the wearable, as well as giving battery voltage by slowly changing colour from green(full battery) to red(empty battery)
Set of resistors	Needed to create a voltage divider which could be used to measure battery voltage with the WEMOS S2 mini, in order to give the user data regarding the battery status

Figure 14: Wearable component table with explanation

With the list of the components sorted out, they needed to be fitted into a final design. Due to the heart-rate sensor already stuck into a case, this would provide a basis to put an add-on on top, which would house the rest of the components. The design of this add-on can be seen in figure 15 below.

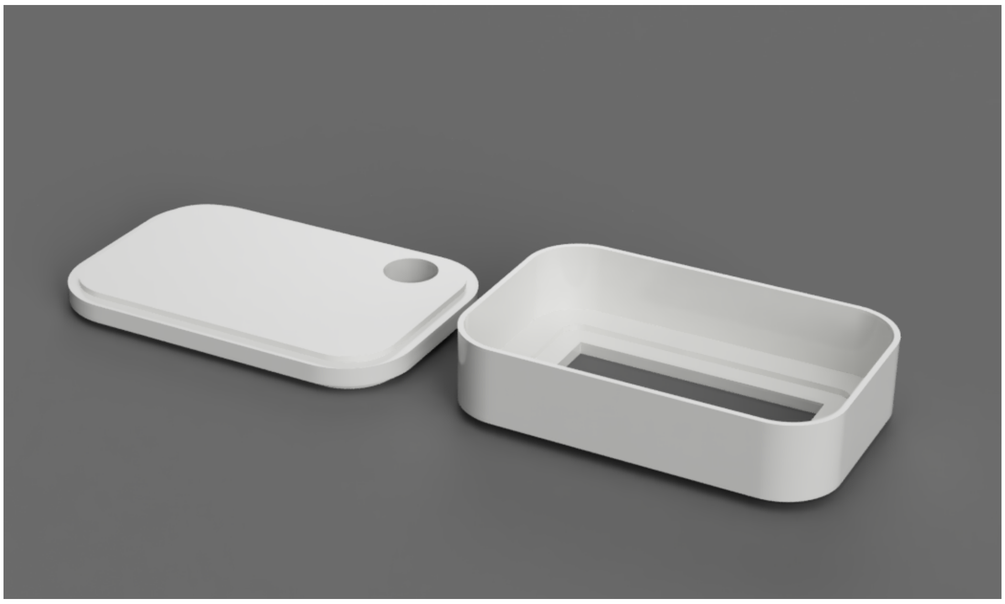
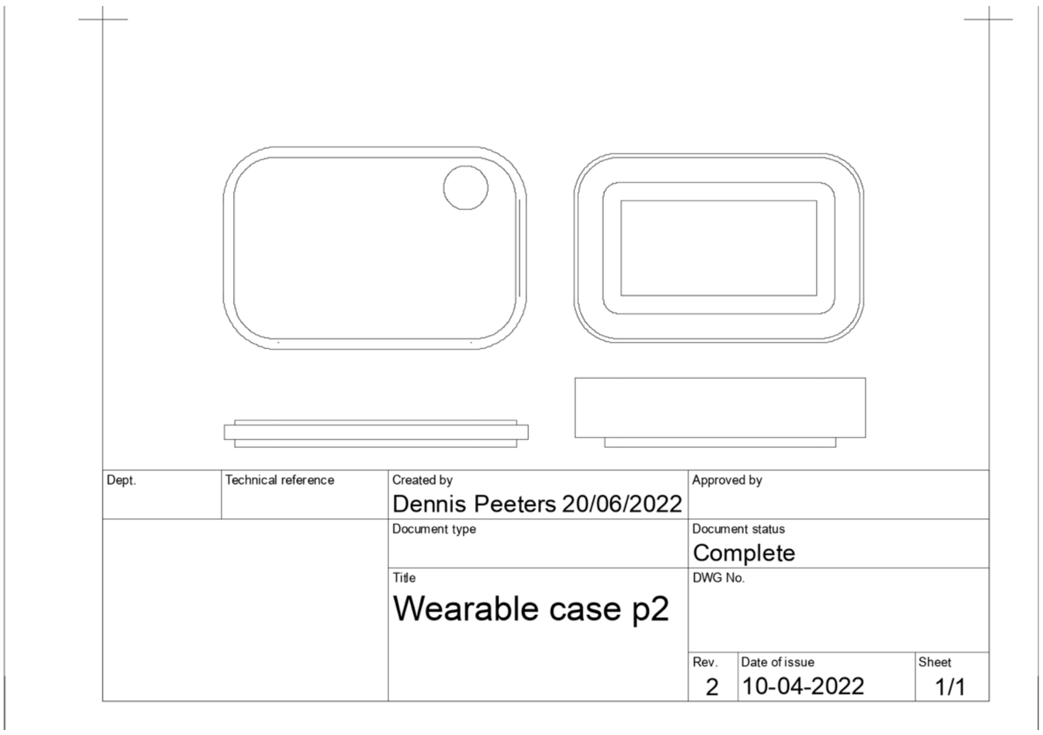


Figure 15: Final design wearable add-on

Next to the final design, there is also a technical drawing. This is generated based on the 3d design as can be seen above. A small example can be found in figure 16. A full file can be found in appendix D, in which the drawing is up to scale.



Dept.	Technical reference	Created by Dennis Peeters 20/06/2022	Approved by
		Document type	Document status Complete
		Title Wearable case p2	DWG No.
			Rev. 2 | Date of issue 10-04-2022 | Sheet 1/1

Figure 16: Preview of the technical drawing of the wearable

Chapter 6 – Realisation

With all three parts of the system now fully specified, the goal of this chapter is to provide detail on the hardware development of the full system. Building and software details as well as the challenges that arose during the building of the project will be further elaborated upon. Certain decisions that altered the design due to those mentioned challenges will also be briefly mentioned. This chapter will be split up in the three parts of the system again, the physicalization itself, the pipeline, and the wearable. In these parts, the different aspects of challenges, software and hardware will be separate parts. It should be noted that this chapter will start with the wearable and the pipeline, as those were first fully developed before the physicalization was completed

6.1: Wearable

The wearable is responsible to acquire and transmit the data. As has been mentioned earlier, this role of data provider could not be realised with an existing device, so a new device had to be developed. In the previous chapter of specification, the core components as well as the design of the watch have been specified. In this chapter, these components are brought together, and integrated in a fully working wearable.

6.1.1: Hardware

Step one in this process was to gather all the separate components from the list specified above. These need to be merged into an electrical circuit which would fit the small case of the wearable. To do this, a 2d realistic sketch, as well as a circuit diagram have been made to further clarify the component layout and connections between them. These can be seen in figure 17 and figure 18 below

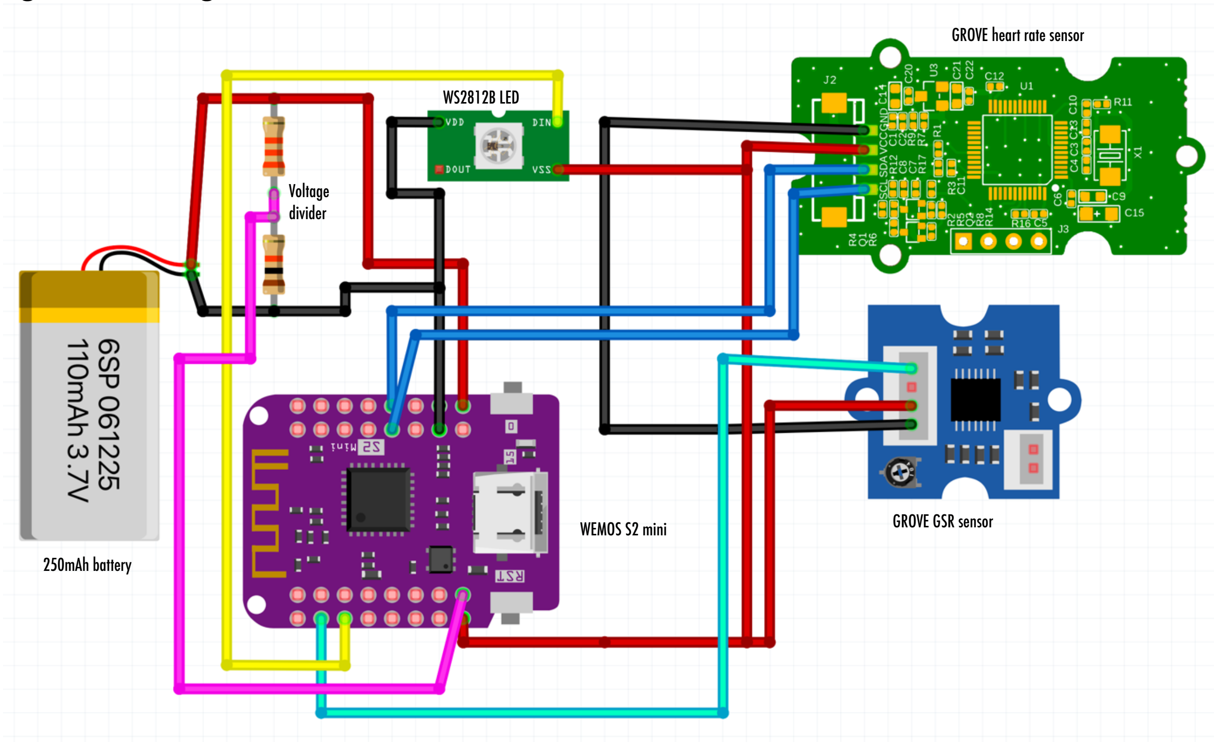


Figure 17: Graphical representation of the circuit diagram wearable

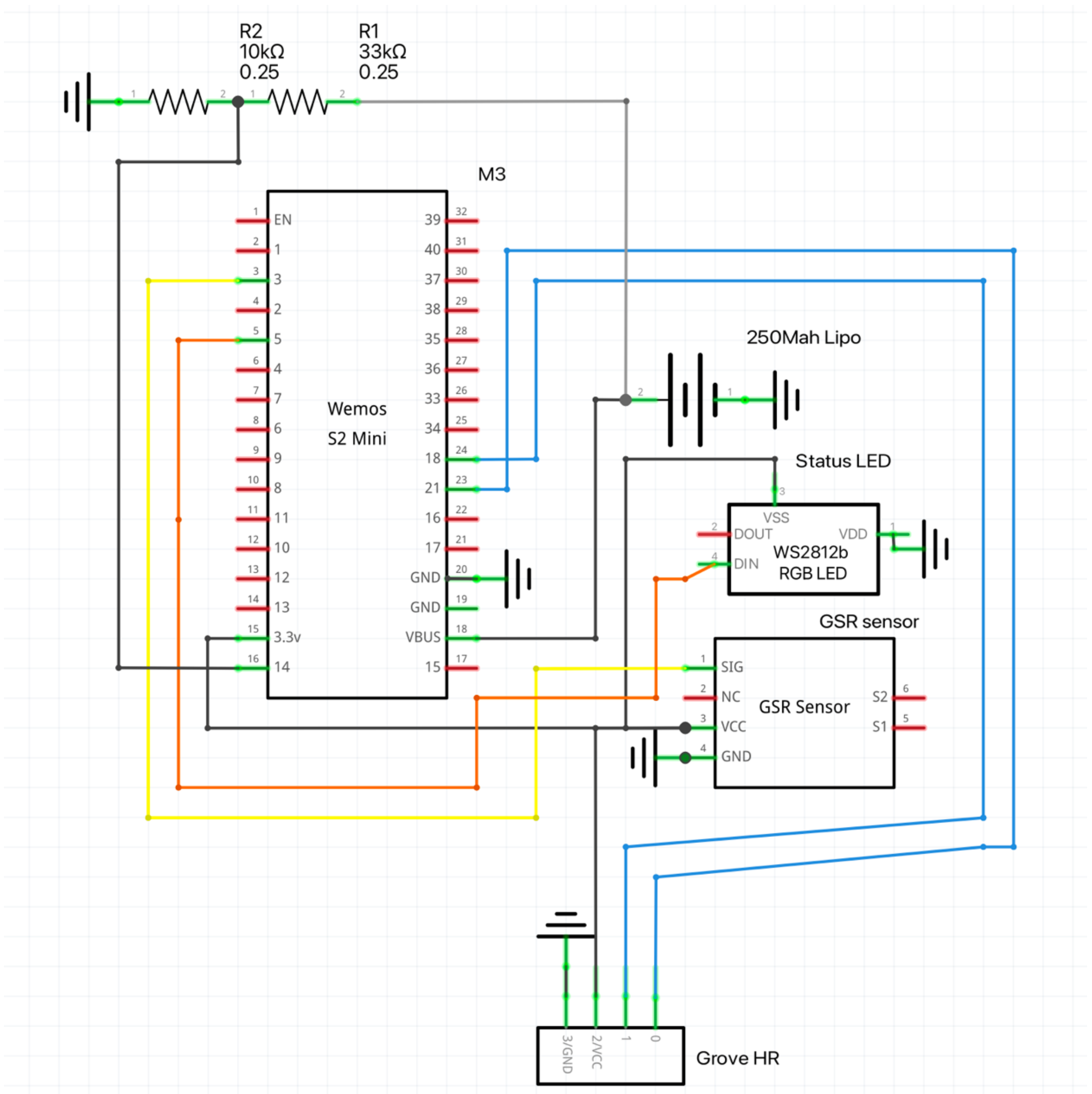


Figure 18: Technical circuit diagram wearable

Something that should be noted is that a few components seen in these diagrams do not reflect the real components. This has to do with the software used to design the drawings. This is based on a component library, and some specific components were not available. The GROVE heart-rate board has been self-developed and added to the library, but for the other components, a similar replacement has been found. For example the lipo-battery. This has only a capacity of 250mAh, not 110mAh from the picture.

A few things to point out that are of interest are the voltage divider, which provides the main-board with the reference measurement point to check the voltage of the battery. This will be used to give the user some more information regarding when to charge the battery for example.

Another question when looking at the design is why the choice was made to use the WEMOS-S2 mini. This has several reasons. First of all, the ESP-32 S2, which is the computing chip on the WEMOS-S2 mini, offers superior performance compared to Arduino boards or older ESP based boards. The computing chip has plenty options to get data in, like full I2C(a 2-line communication protocol) support and internal 12-bit ADC's(analog to digital converters) providing more accurate measurements from the GSR and battery levels. It also provides WIFI and Bluetooth options to connect the board wireless to the internet or other device. These features help in achieving the previously set out goals. However, another benefit of this newer computing chip with plenty inputs and processing power is future development. This chip would be capable to run the algorithm that turns raw heart-rate data from the sensor into an actual heart rate reading. This is now done by a separate computing board inside the wearable. This option could provide the next iteration with more accurate and useful data to start with. Next to that, extra components could easily be added to the wearable, as there are plenty of I/O pins still free that can take in data from sensors. The last two reasons for choosing this board in particular, were its tiny size relative to the capable computing power and the proper support and plenty resources that can be found on the internet.

With the circuit diagram finalized, the first version of the watch could be made. The previously seen design is printed with the help of the 3d-printer, and could, with a bit of filing some edges, be fitted on top of the existing case for the GROVE heart rate sensor. This all resulted in the wearable which can be seen in figure 19, figure 21 and figure 22 below.

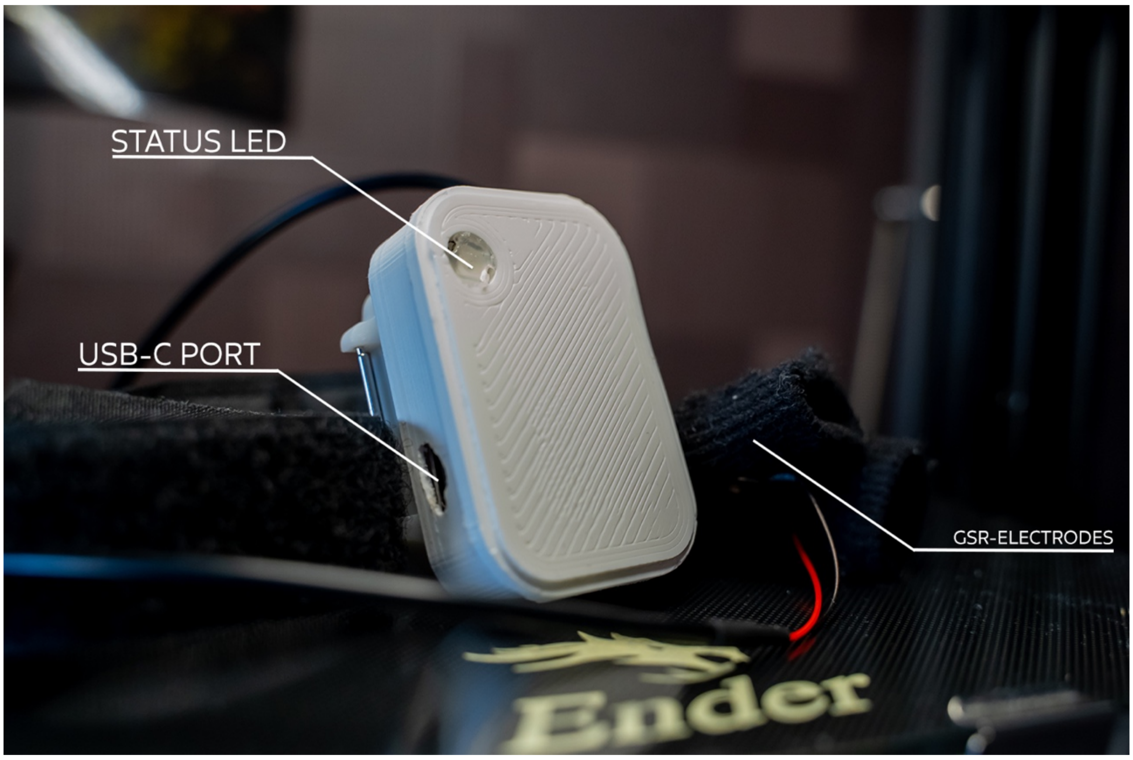


Figure 19: Wearable prototype

This is the wearable, completed and closed. It is not turned on at this moment, hence the LED is off. In the design that was printed by the 3d printer it was forgotten to add an USB-C port access hole on the side. This was partly due to the final internal layout not being completely clear, and afterwards it was found unnecessary to print a whole new for just this small point. Thus, a hole has been made after the printing. In the picture above the three core parts of the wearable can be seen, the status LED telling the user the amount of battery left in the device. It does this by using a colour fade which very slowly goes from green to red. A visualisation of what the colour would look like can be seen in figure 20 below.

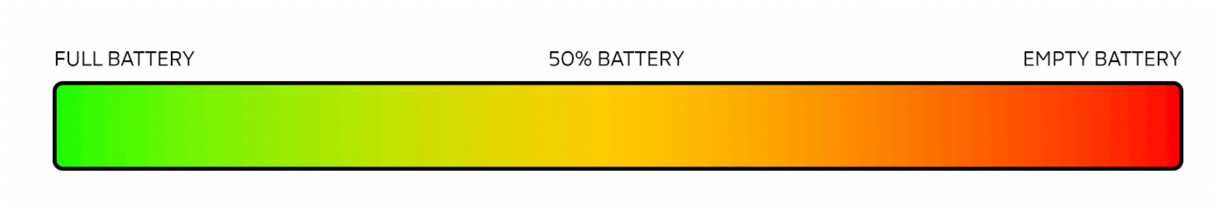


Figure 20: battery indicator colour representation

Next to that, the LED will also inform the user about when a data-package is sent to the server. It does this by blinking blue.

The GSR electrodes that can be seen in picture 19 are two cloth socks, with a small metal disk in them. These are supposed to be put around two fingers of the same hand, with the metal touching the finger. This is part of the rest of the circuitry of the GROVE GSR measurement board.

The USB-C port that can be seen on the side is at the moment just to upload the code onto the controller inside the wearable. However, the idea is to later also use this as the connector for charging. This has however not been implemented in this version of the wearable yet.

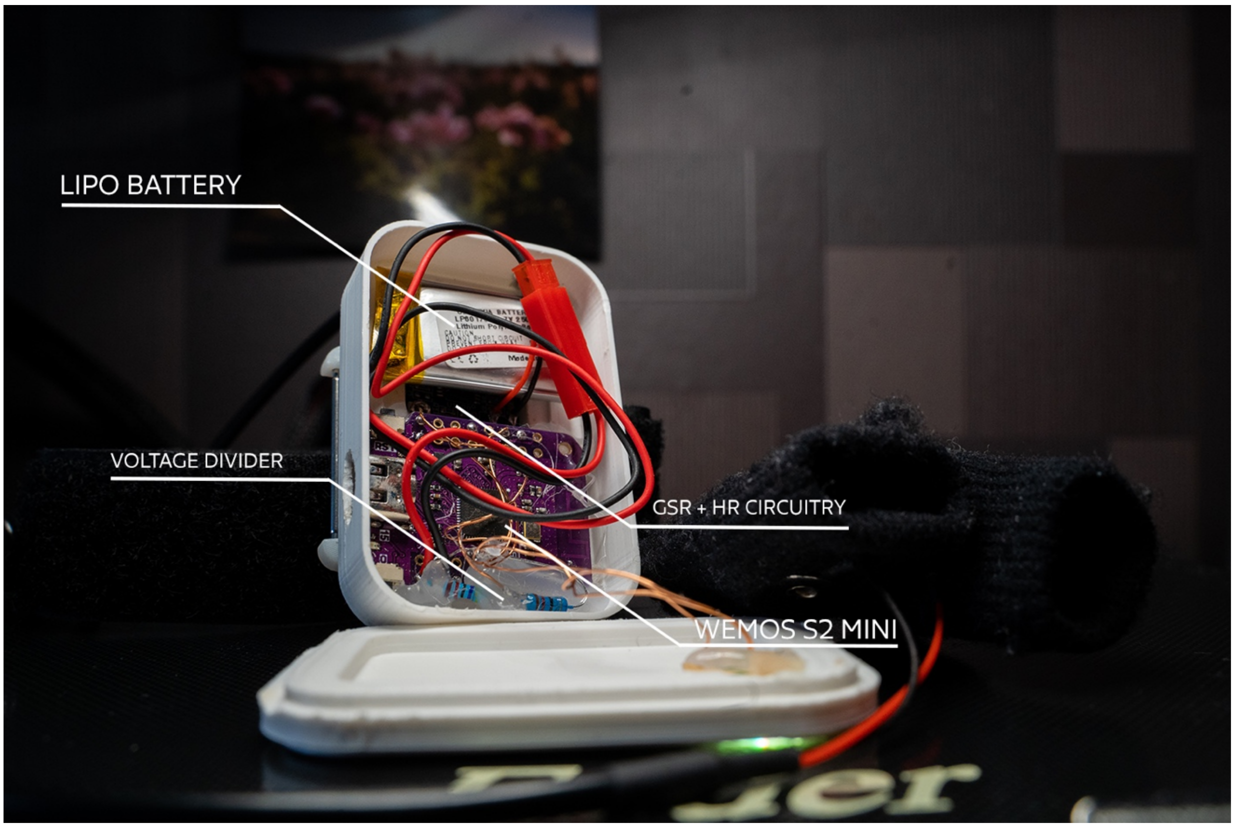


Figure 21: Wearable prototype inside

As can be seen in figure 21 above, the inside of the wearable is not as sleek as the outside. Due to the limited amount of space, there was not a lot of room to properly put everything, so most parts are stacked on top of each other, connected with really tiny enamel copper wire. However, when closed off, this cannot be seen. The battery is connected with the board with a large red connector. This has a reason, which will be explained in the challenges chapter 8 later on.

The GSR and HR circuitry can be found underneath the battery and WEMOS-S2 mini. In order to make them fit, the connectors have been taken off, and the wires have been directly soldered on the pads on the PCB of both sensors. The GSR sensor has been placed in such a way, that in between the battery and WEMOS S2 mini there is a small space where the user can access the potentiometer that is on the GSR sensor board. This potentiometer can be used to calibrate the GSR sensor for the environment it is in. This could be automated in a later stage, by using a digital potentiometer IC. However, in this version of the wearable this has not been done.

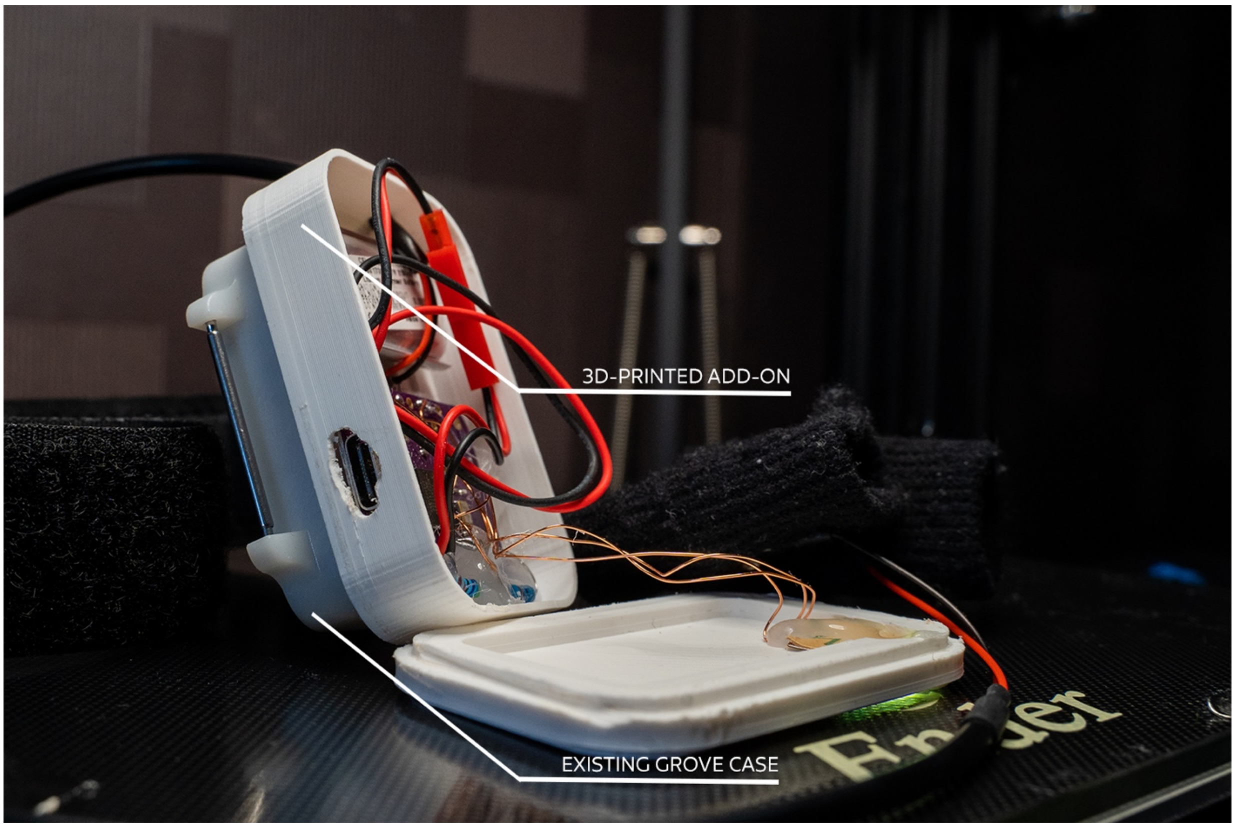
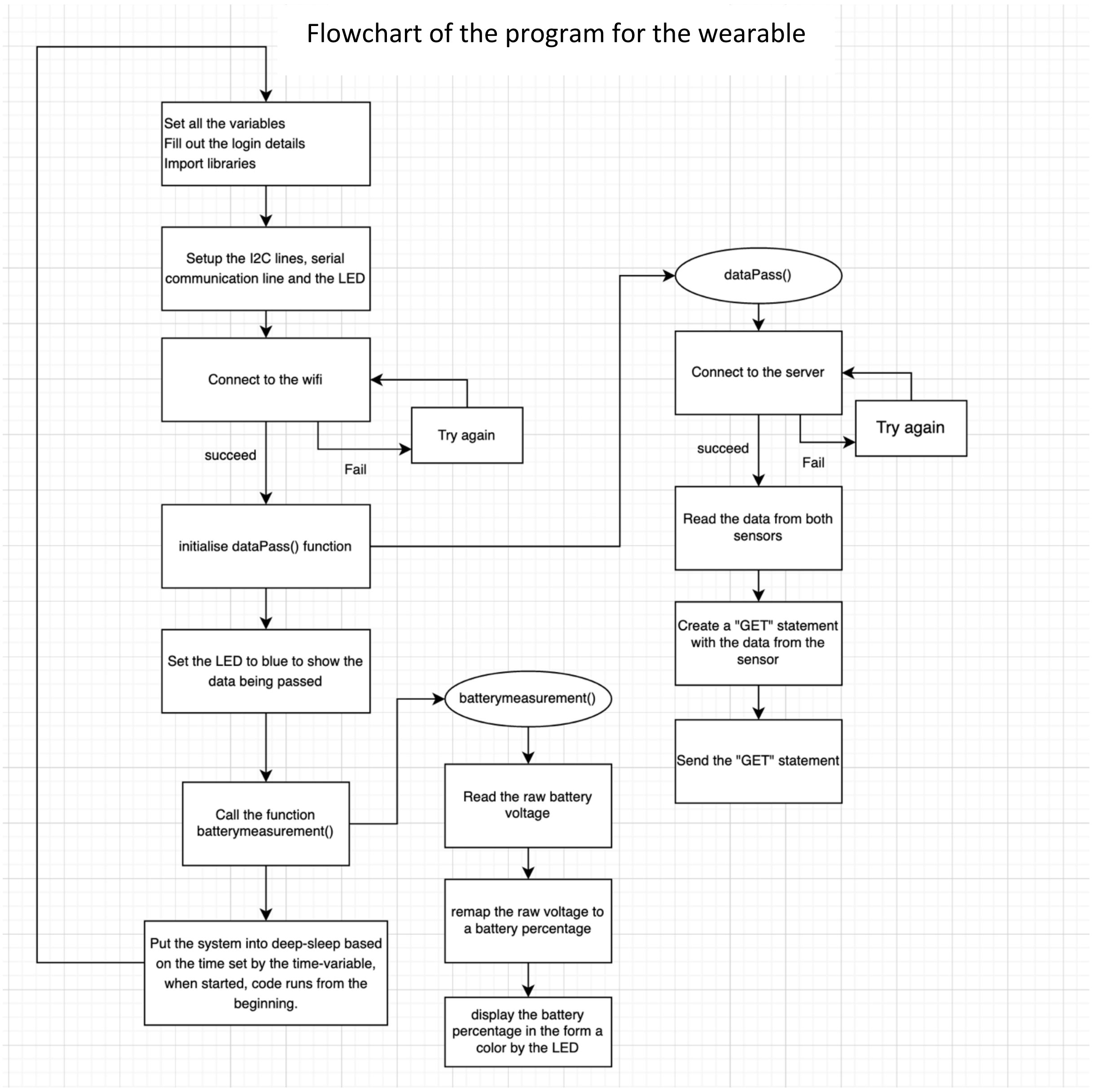


Figure 22: Wearable prototype side profile

Figure 22 above shows the wearable from a side angle, where the existing GROVE hear-rate sensor case can be seen, merged with the 3d-printed add-on that has been shown in the specification chapter.

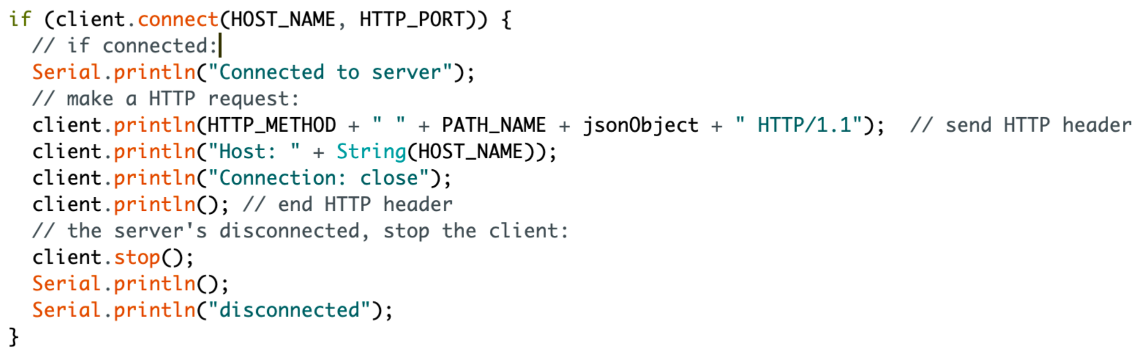
6.1.2: Software

In order to get all the hardware working, software needed to be developed. For the wearable the Arduino platform has been chosen, as it is known by the researcher, and has already example code for both the sensors. The full code can be found in appendix E. However, to simplify it a bit, and make it more accessible to people with less understanding of code, a flowchart has been made to give an overview of the code that has been written for the wearable. This flowchart can be found below in figure 23.


Flowchart of the program for the wearable
Set all the variables
Fill out the login details
Import libraries
Setup the I2C lines, serial communication line and the LED
Connect to the wifi
Try again
succeed
Fail
initialise dataPass() function
Set the LED to blue to show the data being passed
Call the function batterymeasurement()
Put the system into deep-sleep based on the time set by the time-variable, when started, code runs from the beginning.
dataPass()
Connect to the server
Try again
succeed
Fail
Read the data from both sensors
Create a "GET" statement with the data from the sensor
Send the "GET" statement
batterymeasurement()
Read the raw battery voltage
remap the raw voltage to a battery percentage
display the battery percentage in the form a color by the LED
Figure 23: Wearable code flow-chart

As can be seen in the flowchart above, the code consists out of a main part, and two functions. One is to gather and pass the data to the server, while the other concerns the battery measurements. Something that should be noted is that the wearable shuts off most functions after completing the measuring and sending. It will go into deep sleep mode for 20 seconds, after which the code will run again. This is done for a reason. The wearable has a very small battery, while it has two fairly power-hungry processors working inside. The one inside the heart-rate sensor cannot be tweaked with code, but the main processor, the WEMOS S2 mini, can be. So in order to achieve a run-time of about 90 minutes, the main processor will go into deep sleep in which it only needs very little power. This is a compromise, which will be elaborated more upon in the discussion chapter 8.

The part of passing the data to the server will also be elaborated upon a bit more, as this also ties in with the rest of the installation and in particular the pipeline of the installation.



```
if (client.connect(HOST_NAME, HTTP_PORT)) {
  // if connected:
  Serial.println("Connected to server");
  // make a HTTP request:
  client.println(HTTP_METHOD + " " + PATH_NAME + jsonObject + " HTTP/1.1");  // send HTTP header
  client.println("Host: " + String(HOST_NAME));
  client.println("Connection: close");
  client.println(); // end HTTP header
  // the server's disconnected, stop the client:
  client.stop();
  Serial.println();
  Serial.println("disconnected");
}
```

Figure 23: Code snippet regarding data passing to the server

The code that has been used to pass the data to the server can be seen in figure 25. A HTTP (HyperText Transfer Protocol) request was created by using the “GET” method. Although it might not be the most secure and reliable way to communicate the data, due to the lack of time and the ease this could be implemented, the GET method was chosen. However, a more elaborate discussion can be found in the discussion chapter 8. The jsonObject, an attribute that can be seen in the code, is the part that contains the data from the sensors. In case it cannot connect directly, it will try this snippet of code 5 more times. If it still fails after the 5th time, the data will not be uploaded to the database. The Serial.println() parts that can be seen in the code are there as part of the development of the wearable but will not have any function for the user of the product. These are to give a more elaborate status on what is happening to the developer.

As mentioned before, the full code for the wearable can be found in appendix E. In there, comments regarding more specific parts have been placed in order to clarify certain parts.

6.1.3: Challenges

The development of the wearable came with some challenges. There were many minor issues which could be easily resolved. There were however also some more major challenges, which will be discussed below. Some of them have been resolved, while others still need looking into. These will also be discussed in the discussion chapter 8 later in the report.

The first major challenge which arose was the issue of the data-parsing to the server. This has already been discussed in the specification chapter. This issue will be raised here as well, as the system is currently running on the researchers own webhosting, which is a paid service. However, this solution is not a sustainable solution, and a proper data-storage server with the right privacy settings should be sourced at the university. The solution of using the own server works, yet is temporary and will probably be terminated after the project is completed, as the login credentials will also give someone access to the other things hosted on the server.

The second challenge is the issue with the battery management. Previous iterations of the ESP-32 boards always included a battery management circuit which could provide charging via the USB-port located on the board, combined with the proper circuitry to charge a lithium cell without causing fire, explosions and other issues related to lithium battery charging. However, this ESP32-S2 board does not offer these charging functionalities. For this, a temporary solution has been found. The lithium cell chosen for in the wearable has been specifically selected as it has some of the protection-circuitry inside, preventing it from blowing up or catching fire while also protecting the cell during charging to make sure the lifespan of the cell is kept normal. The charging of the battery is currently done with an external charger, requiring the wearable to be opened and the battery to be fully disconnected and connected to a separate connection of the charger. The charging circuitry was too large to fit the wearable, so this solution has been chosen as a temporary fix. When a redevelopment of the wearable takes place, this topic of charging should be addressed again, in which the circuitry gets integrated into the wearable, and one port can be used to upload the code and charge the wearable, so the wearable does not have to be opened anymore. This would also create more space, as the connector is now taking up a lot of space in the wearable.

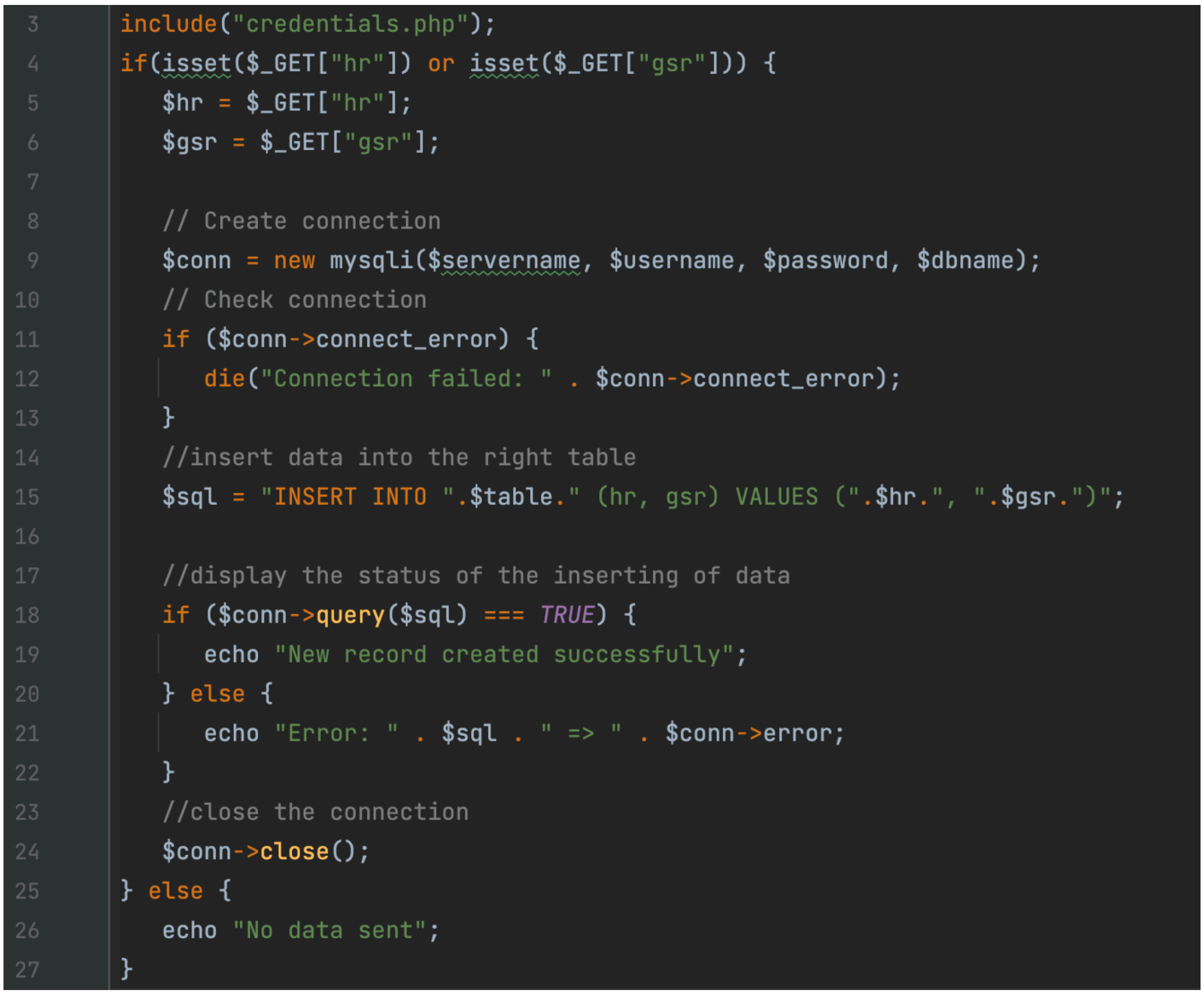
6.2: The pipeline

A key part of the full installation was designing and implementing the "pipeline" that would connect both the wearable and the physicalization together, enabling a near-real time representation of data. The outline of this has been specified in chapter 5.2, and the diagram that was developed during this specification phase can be found in figure 12.2.

With this outline of the system, the whole pipeline could be implemented. A major factor in how it would work was based on the choices made in chapter 6.1, the wearable. When the choice was made to use a self-owned server, the way the pipeline would work was significantly altered, and this also meant more parts needed to be developed from the ground up to get a fully working system. Something that should be mentioned is that the pipeline only consists out of a bit of code, which is the PHP-webpage that receives the data and adds it to the database. The pipeline itself is mainly the structure that ties the two components together and could be seen as the backbone of the full system, outlining how especially the data will be processed, transmitted, and stored in the full installation. The structure itself has already been specified, and the implementation is also part of the wearable and physicalization realisation. This part of the chapter will therefore give an explanation of what is happening on the server side, and discuss the challenges that arose during the development. As this also ties in with the development of the wearable and physicalization itself, those components will be mentioned separately under the wearable and physicalization parts of chapter 6: realisation. This means this pipeline part of chapter 6 will be relatively short.

6.2.1: Software

There is a lot of software involved in the pipeline of the full system. As mentioned in the introduction of this part, the software regarding the connections of the wearable to the server, as well as the connections from the physicalization and the inner workings of both components will be discussed in their own parts of chapter 6. The first part that will be discussed in this chapter is the webpage to which the data of the wearable is sent via the "GET" method as discussed in chapter 6.1. The simple PHP code that is used can be seen in figure 26 below.



```
3   include("credentials.php");
4   if(isset($_GET["hr"]) or isset($_GET["gsr"])) {
5       $hr = $_GET["hr"];
6       $gsr = $_GET["gsr"];
7
8       // Create connection
9       $conn = new mysqli($servername, $username, $password, $dbname);
10      // Check connection
11      if ($conn->connect_error) {
12          die("Connection failed: " . $conn->connect_error);
13      }
14      //insert data into the right table
15      $sql = "INSERT INTO ".$table." (hr, gsr) VALUES (".$hr.", ".$gsr.")";
16
17      //display the status of the inserting of data
18      if ($conn->query($sql) === TRUE) {
19          echo "New record created successfully";
20      } else {
21          echo "Error: " . $sql . " => " . $conn->error;
22      }
23      //close the connection
24      $conn->close();
25  } else {
26      echo "No data sent";
27  }
```

Figure 26: PHP-webpage code

The webpage is hosted by a webhosting provider. As can be in line 3 of the code, there is a credentials file which is needed in order to gain access to the database. This has not been included into this report for privacy reasons, but all that it needs is the servername, username, password, and the name of the database. The first thing the code does, as can be seen in line 4, is that it waits until a request comes in. When a request is received from the wearable, it creates a new connection and tries to connect to it, as can be seen in line 9 and 11 in the picture above. When this succeeds, the data that came from the wearable will be inserted into the right table in the database(line 15 in the picture above). Afterwards, it will close the connection and wait for a new incoming request.

When following the data in this pipeline, the next phase is the database itself. This is also managed by the webhosting provider. The software to access the database is in this case PhpMyAdmin, and in there the data is stored in a table. As mentioned previously, the data received is heartrate and GSR. However, as soon as the data is stored in the database, a timestamp is added. This is needed in order the physicalise the data later. A simple representation of the database can be found in figure 27 below.

Timestamp	**hr**	**GSR**
(YYYY-MM-DD HH:MM:SS)	(heart rate value)	(skin conductance value)

Figure 27: Table representation of the database

The timestamp is stored in a standard format, making it compatible with the python library in charge of pulling the data from the server.

The code responsible for the conversion of these raw GSR and HR values into valence and arousal will be further discussed in chapter 6.3.2

6.2.1: Challenges

The nature of the challenges that would arise in the development of the "pipeline" are mostly also related to the wearable or the physicalization. An example is the previously mentioned temporary solution of using the webserver hosted by the researcher himself. Due to this challenge, the whole process as has been described in detail in this chapter and chapter 5.2 only fully applies to the temporary solution. However, the structure defined in figure 12.2 still applies to an eventual other solution. This is due to the benefits described in chapter 5.2.

Another challenge which has been discussed earlier but should be mentioned again is the fact that the system needs to be able to cope with a large amount of data request. This is for now not a major issue, but might become important in future iteration, which will be discussed in the discussion chapter 8.

Another challenge is the fact of how the system communicates if the data is actually stored. The user never receives feedback on one of the devices if the data is really there. Although methods have been implemented for developers to see if this really happens, there is no feedback loop implemented towards to user.

6.3: Physicalization

Final step in the realisation of the full system is the physicalization. This has therefor also been the last part that was built. The physicalization, as mentioned before, is the part of the installation that takes the data and processes it. It turns the heartrate and GSR data into valence and arousal attributes and represents those by using colour and light. In the previous chapter of specification, a component list was presented, combined with 3d renders and a technical drawing. In this chapter, these components are brought together, the structure is built, and some code is written to tie everything together into a working system.

6.3.1: Hardware

Similar to the development of the wearable, the first step in realising the physicalization was creating the circuit diagrams to connect all the components listed in the specification above. The components needed to be placed in such a way that all function would be integrated, while also fitting in the small box below the physicalization. Also, enough room was needed to make sure the heat produced by the power supplies and the raspberry pi would not cause issues when the system would be running for a longer time. The result of these diagrams, both in full colour as in a more technical layout can be found in figure 28 and figure 29 below.

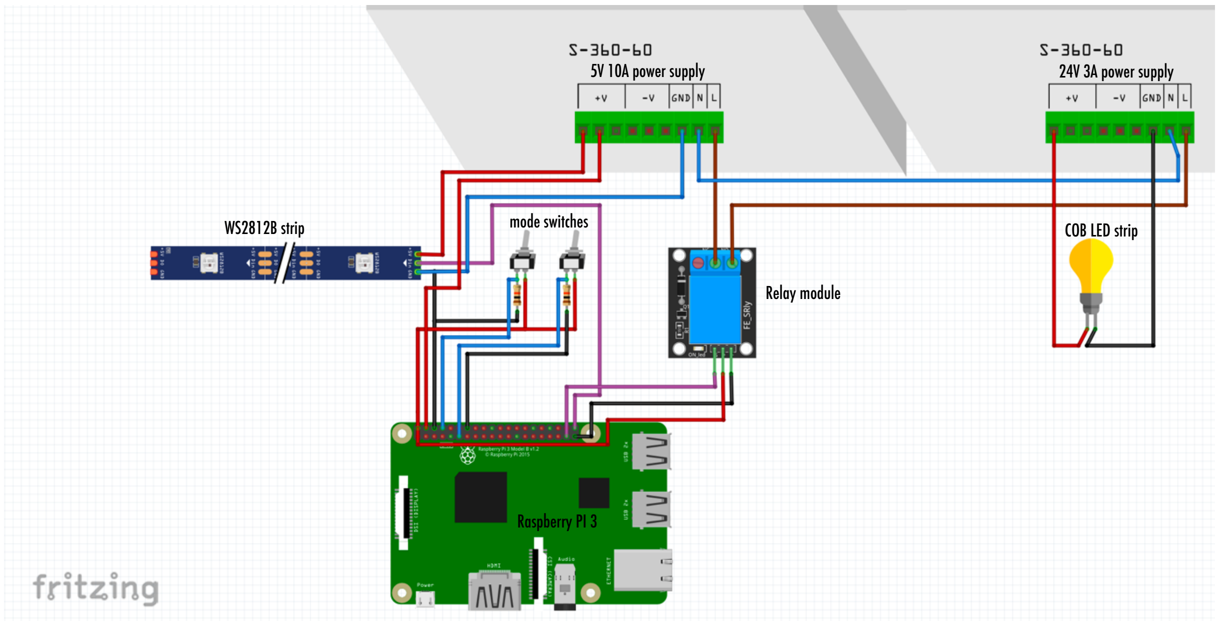


Figure 28: Graphical representation of the circuit diagram physicalization

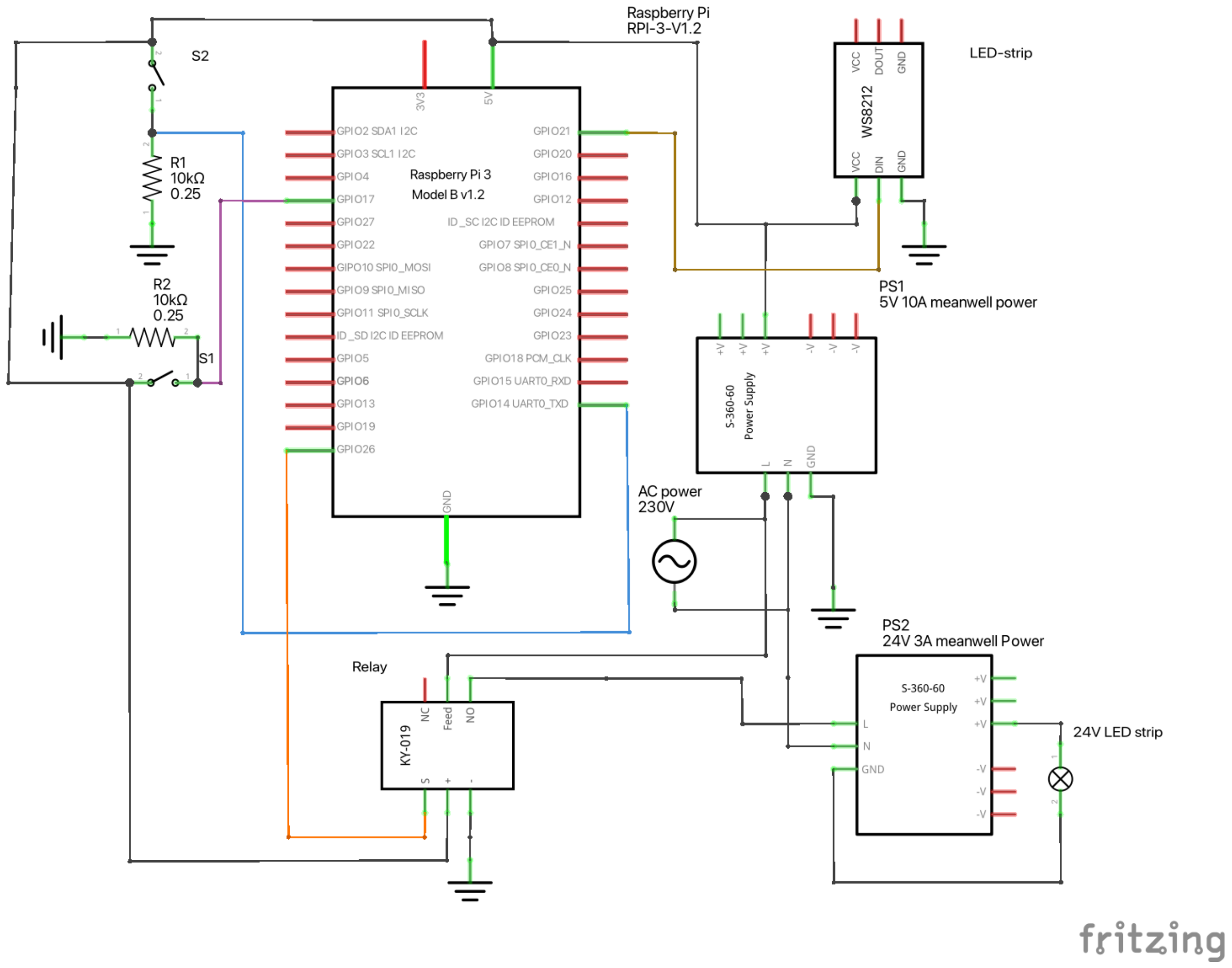


Figure 29: Technical circuit diagram physicalization

As was the issue with the diagrams of the wearable, some components that were used for the diagrams are not the actual component, but just placeholders to represent the actual ones. In these diagrams these are the S-360-60's in figure 28. The one on the left is the 5v

10A Meanwell power supply, while the one on the right is the 24v 3A Meanwell power supply. In figure 29 these have been marked with extra labels. Furthermore, the Bulb in figure 28 represents the 24V COB LED strip that is used inside the ring. The rest of the components are the correct part as listed in the specification.

With the circuit diagram ready, the rest of the physicalization could be built. The whole thing is a combination of 3D-printed parts, combined with a wooden structure. As can be seen in figure 30 below, the base of the system, as well as the small switch-holder in the front is made with a 3d printer. This to get the perfect fit for the component which should slot into them, as well as integrating mounting points in them so it could be securely attached to the wooden structure without coming apart.

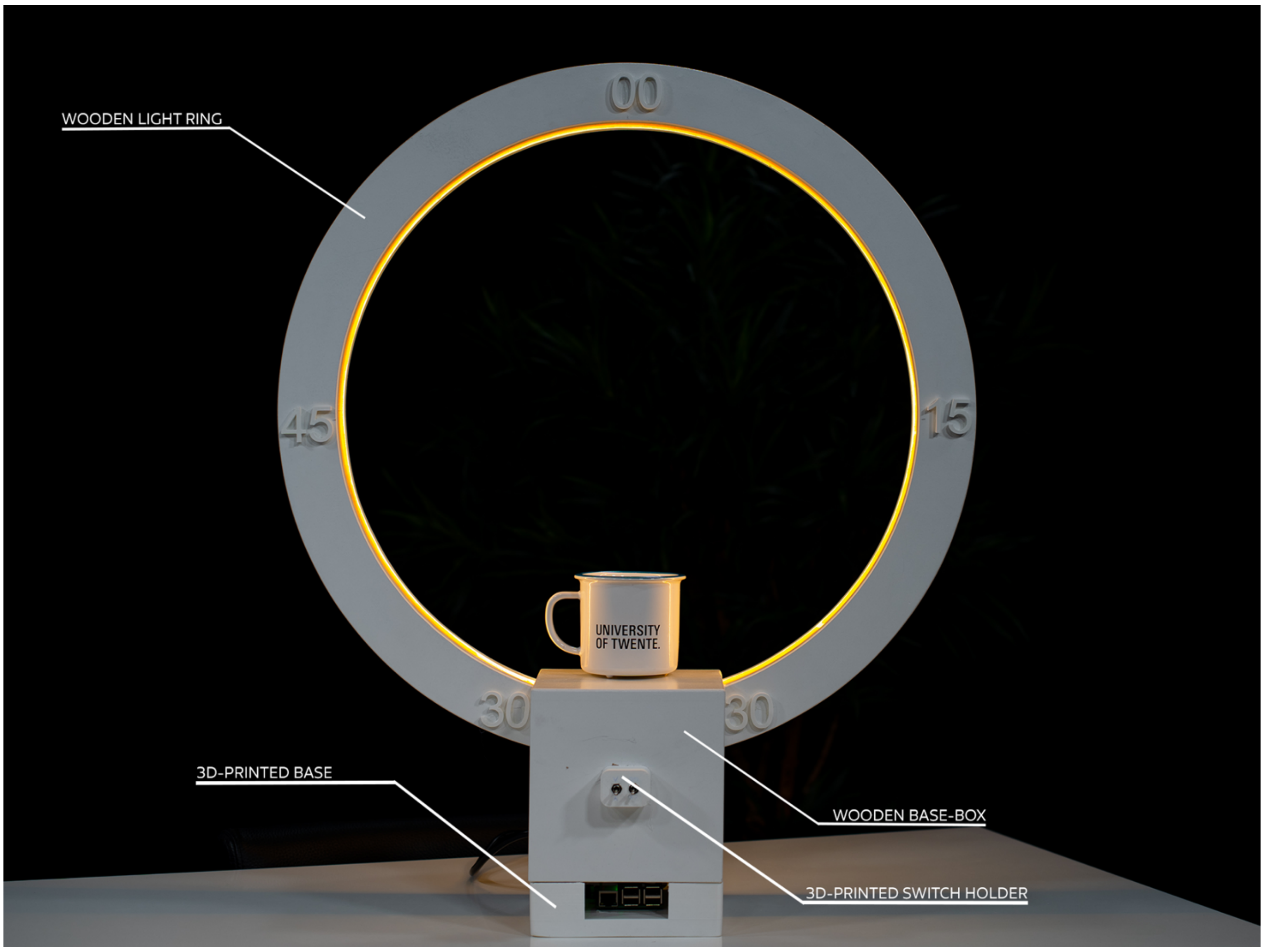


Figure 30: Physicalization frontal view

Another benefit of creating a 3D-printed base is the fact that cooling can now be improved. Another part that is 3D-printed, are the numbers on the ring itself. These have been added later, for which the reasoning will be explained in the challenges part of this chapter.

The wooden ring together with the base box, as can be seen in the picture above, have been made out of several types of wood. Most prevelant are the types MDF and bend-plywood. This is special type of plywood which can be used to make round shapes. The first step was to use a CNC-router to cut out the two wooden rings out of the MDF board. Together with the bend-plywood, the round shape of the circle could be created, as can be seen in figure 31 below

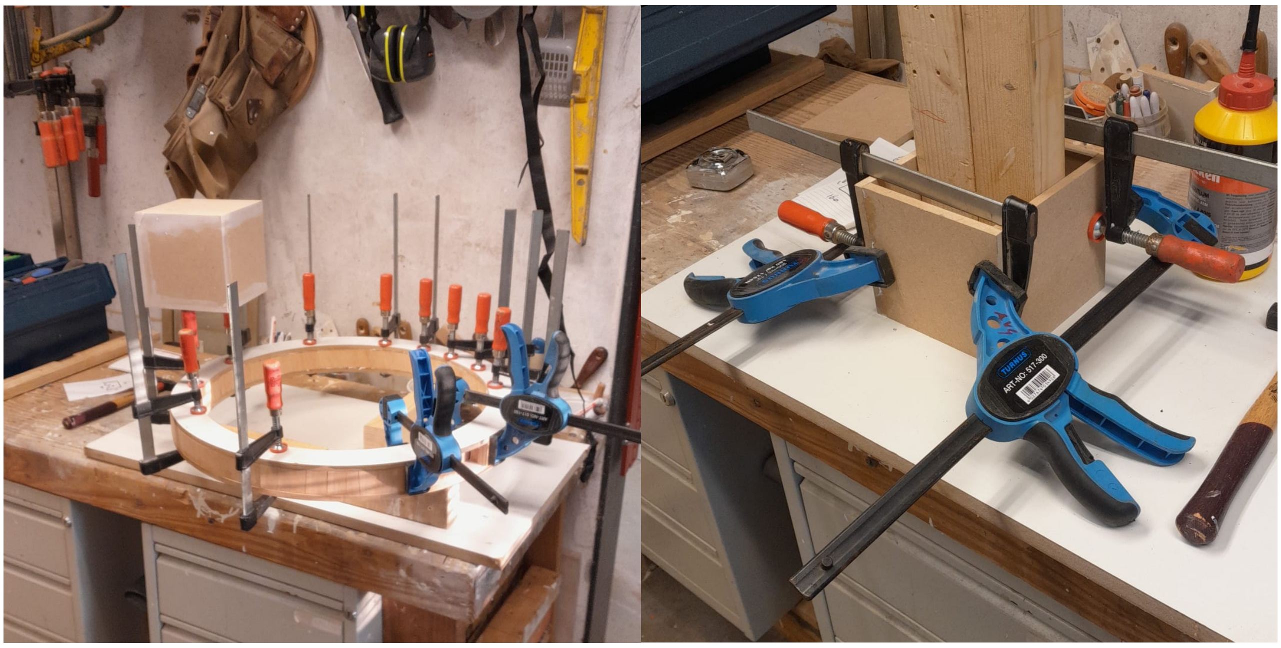


Figure 31: Pictures of the building process part 1

Also, the wooden box was made out of MDF. After the glue had set, the two shapes were painted white, as can be seen in figure 32.

After both of the shapes were fully painted, they were merged together into one large shape, which can be seen in figure 32.

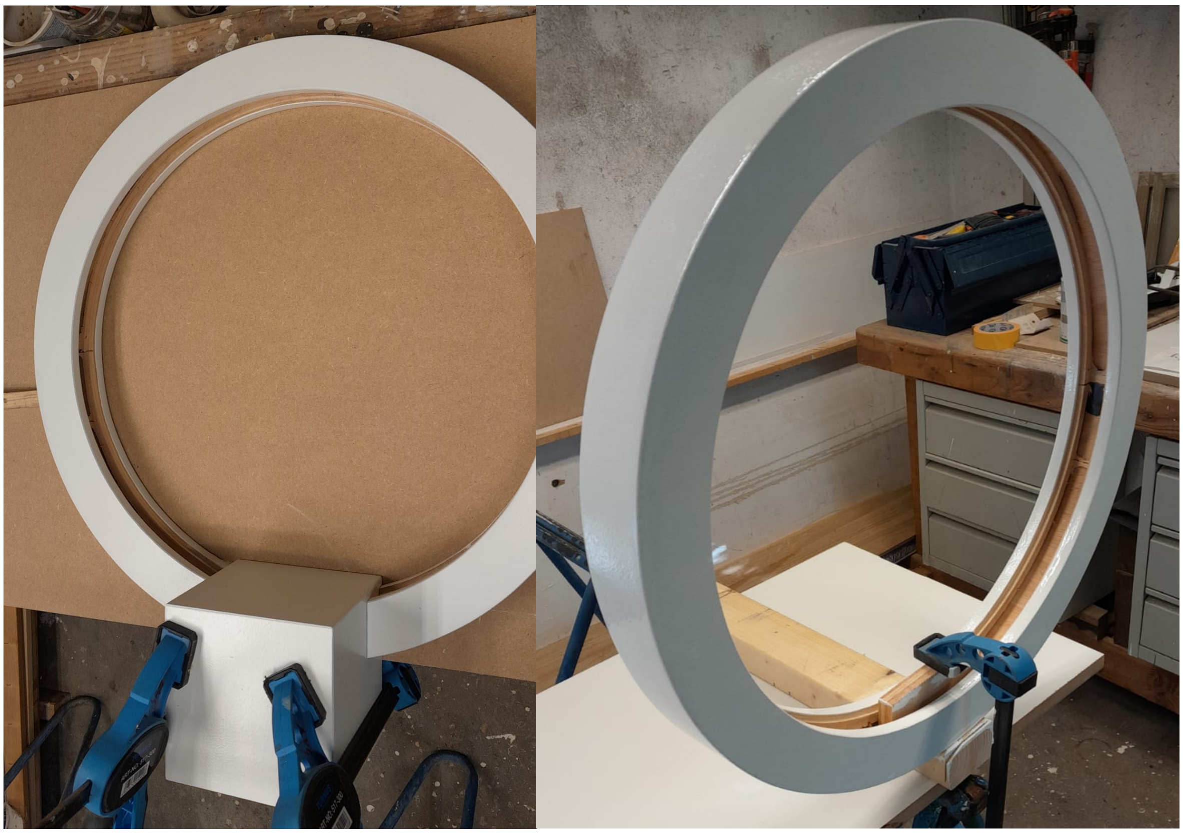


Figure 32: Pictures of the building process part 2

With the wooden structure complete, the next phase was to start fitting the electronics into the ring. The way these are fitted can be seen in figure 33 below.

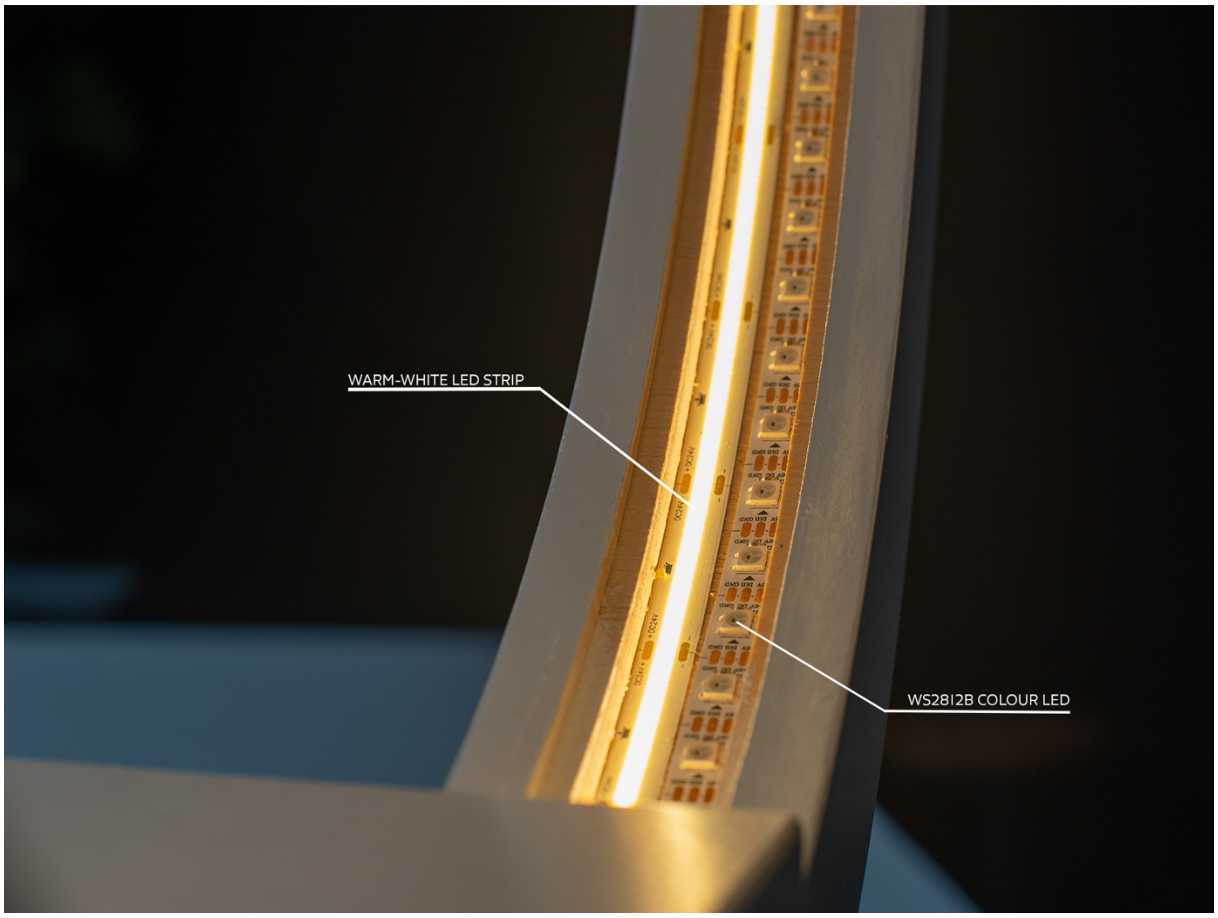


Figure 33: LED strips placements in the wooden ring

Afterwards, the rest of the electronics were fitted into the base of the physicalization according to the circuit diagrams that have been provided earlier in this chapter.

The purpose of the WS2812B colour LED ring is to be able to display valence or arousal, based on the position of the switches. For this certain colours have been selected.
The physicalization will display the arousal and valance level that a person is experiencing using colours of the LED lights. The colour coding scheme is represented in Figure 34 and 35

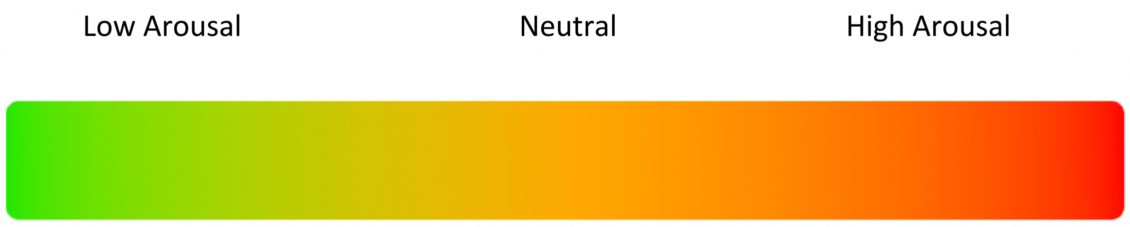


Figure 34: LED colour representation for arousal

Looking at figure 34 when an LED is orange, it means the arousal levels of the person are fairly neutral. Green is an indication of low arousal levels and red an indication of high arousal levels When switching the physicalization in Valence mode, the colours of the system will change to a new spectrum, from blue to yellow, as can be seen in figure 35. Valence is a measure between positive a negative, where blue means the value is a positive valence, while a negative valence is indicated by a more yellow colour. The colour in the middle, white, means a neutral state.

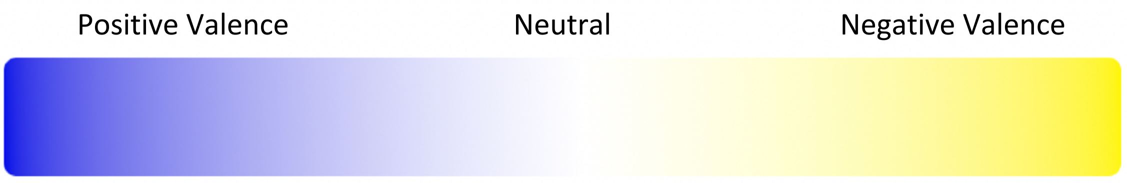


Figure 35: LED colour representation for valence

For the LED's, these more contrasting colours have been chosen in order to really get a clear distinction between the different states. Another choice that was made is that arousal goes from green to red, where in the middle the mix of the colours creates orange. This is because arousal is fairly linear from a low to a high level. However, for the valence measure, the choice was made to create three states, positive, negative, and neutral. For this, three colours (from blue to white and from white to yellow) have been chosen, making it clearer there are three states.

The switches on the front, as seen in figure 36, can be set to several modes.

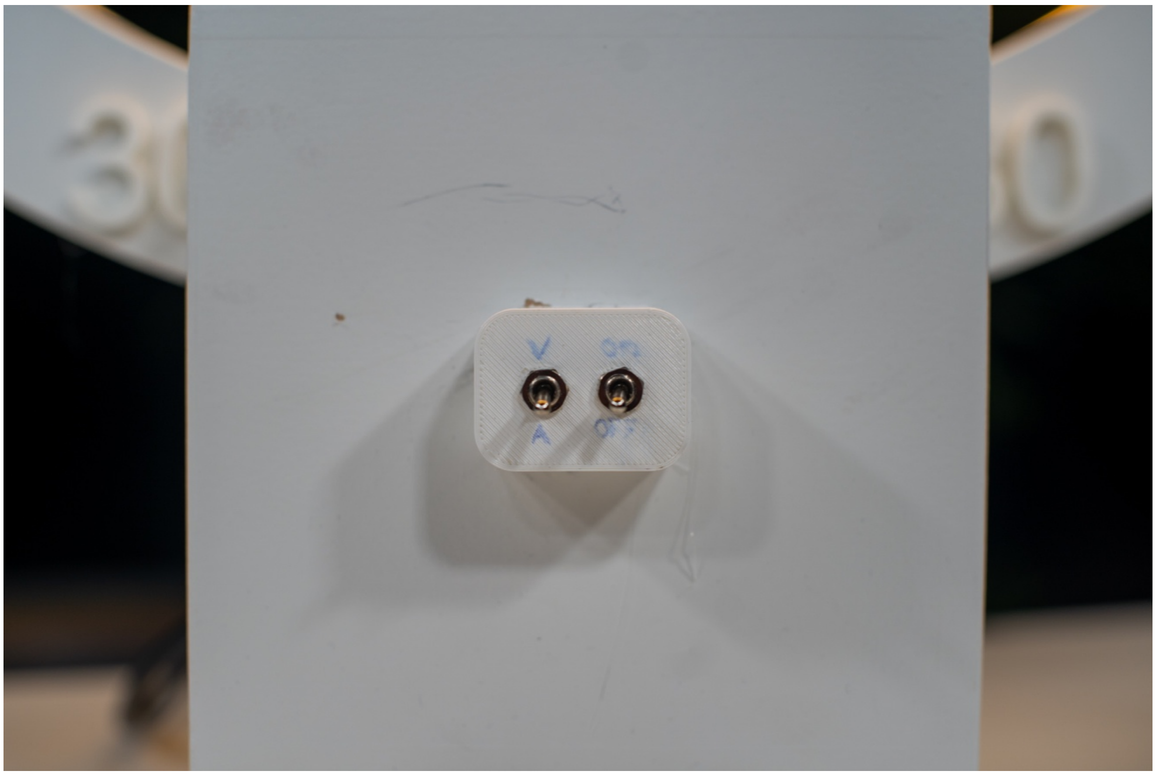


Figure 36: Frontal view of the control switches

There are two switches. The one on the right will control the on and off of the representation of data. So switching it to off means the warm-white LED's will turn on, and the physicalization will form a simple decorative wall or desk light. When the switch is set to on, the warm-white LED strip will turn off, and the system will start pulling data form the database. In this mode, the switch on the left can be used as well. Switching this switch into V-mode will process the data according to the Valence algorithm, while switching it into A-mode will make sure the data is processed according to the arousal algorithm.

6.3.2: Software

With the hardware installed and working, the next phase is to provide the system with the right software. As mentioned earlier, the system uses a raspberry pi to do the computing. This means the code will be developed using the Python coding language. This was chosen as it easily integrates with the raspberry and the researcher is already familiar with the language, making it easier to develop the relative complex software needed for the physicalization. Like with the wearable, a simplified schematic representation of the software has been made in order to get a better understanding of the program. This can be seen in figure 37 below(next page due to size).

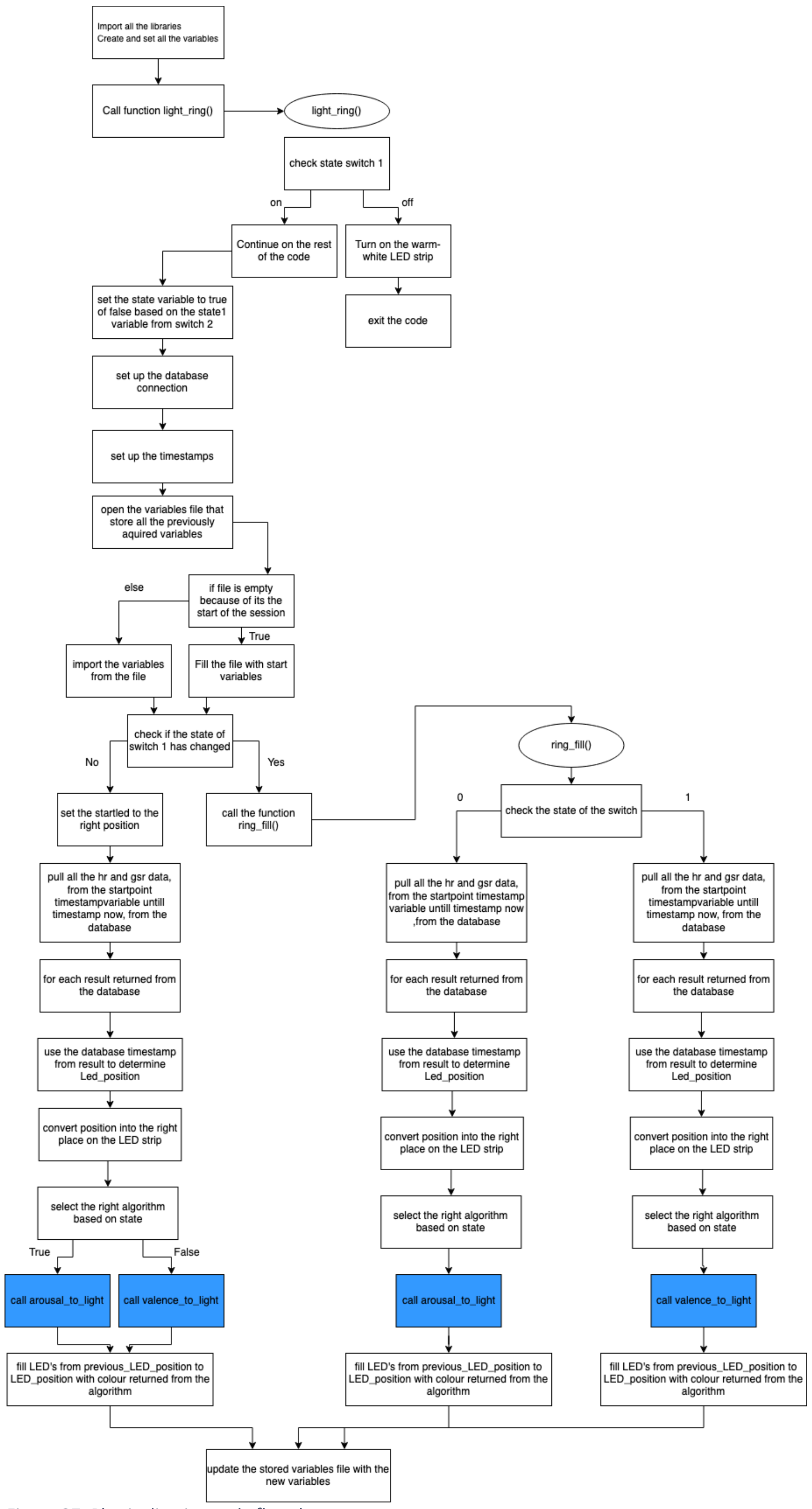

Import all the libraries
Create and set all the variables
Call function light_ring()
light_ring()
check state switch 1
on
off
Continue on the rest of the code
Turn on the warm-white LED strip
exit the code
set the state variable to true of false based on the state1 variable from switch 2
set up the database connection
set up the timestamps
open the variables file that store all the previously aquired variables
if file is empty because of its the start of the session
else
True
import the variables from the file
Fill the file with start variables
check if the state of switch 1 has changed
No
Yes
set the startled to the right position
call the function ring_fill()
ring_fill()
check the state of the switch
0
1
pull all the hr and gsr data, from the startpoint timestampvariable untill timestamp now, from the database
pull all the hr and gsr data, from the startpoint timestamp variable untill timestamp now ,from the database
pull all the hr and gsr data, from the startpoint timestampvariable untill timestamp now, from the database
for each result returned from the database
for each result returned from the database
for each result returned from the database
use the database timestamp from result to determine Led_position
use the database timestamp from result to determine Led_position
use the database timestamp from result to determine Led_position
convert position into the right place on the LED strip
convert position into the right place on the LED strip
convert position into the right place on the LED strip
select the right algorithm based on state
select the right algorithm based on state
select the right algorithm based on state
True
False
call arousal_to_light
call valence_to_light
call arousal_to_light
call valence_to_light
fill LED's from previous_LED_position to LED_position with colour returned from the algorithm
fill LED's from previous_LED_position to LED_position with colour returned from the algorithm
fill LED's from previous_LED_position to LED_position with colour returned from the algorithm
update the stored variables file with the new variables

Figure 37: Physicalisation code flowchart

In the code a lot of duplication can be seen. One of the functions can be found in similar forms 3 times, which makes the code inefficient. This is due to a lack of time and a change that had to be made to the code last minute due to it not working. This will be further discussed in the challenges chapter 8 later. Another remarkable thing are the blue boxes for the arousal_to_light() function and the valence_to_light() function. These will be later elaborated on in more detail, as they form an important part of the software, thus an in-detail explanation with pseudocode will be provided. The full code can also be found in appendix F, which will have comments in it in order to better understand the different parts of the code.

The code, as can be seen in figure 37, does not run in a loop, and thus will finish as soon as the calculations have been done. In order to run the code more often, a program called CronJob was used to make sure the code would run every single minute. The disadvantage of this is that the code will now only run once a minute, resulting in the changes a user would make with a switch only being visible after one minute. This is a limitation which will be discussed later as well. CronJob also makes sure that as soon as the raspberry gets powered and the operating system is booted, the code as seen above is automatically started, and will afterwards run every single minute until the system is shut down again. This also makes that if the system runs into an error due to a tiny mistake, it will try again after a minute, and it will keep trying.

Another part CronJob does is wiping the storedvariables.txt file when the system boots up. This is to make sure the system will start with a new set of measurements and won't request and process old data. All these functions run automatically, without the user noticing. When the system is provided with power, it will take approximately 2 minutes before everything is up and the first datapoints are being processed if the system is immediately switched on after boot-up. When it is powered up in the off state, the warm-white LED ring will turn on 1 minute after power has been provided, and it will take an additional 1 minute after the switch is set to on before the first datapoint are put in the LED ring.

The operating system on the main computer, the raspberry pi itself, is RaspberrypiOS, a fully graphical operating system with a normal desktop and file system. As the physicalization itself is still in development, a graphical accessible interface makes making changes on the system, as well as testing out and installing different assets a lot easier.

The first key point of notice in the main code is the storedvariables.txt system. As the code does terminate when it is done, and starts from the beginning again every minute, it is necessary to store certain variables from previous sessions, like the time the installation was turned on as the start time, the start position of the first datapoint, previous heartrate and GSR values for example. The latter values are needed in order to do difference calculations for the algorithm, which will be explained later. This stored variables file saves those variables from being deleted, serving as a storage location. Some of the variables in this file will be updated after every time the code is executed. Others will not be updated after they have been initialised during the first time the code is executed after power-up. At the moment, there are more variables stored then that are needed for proper operation. This is again due to a last-minute change, which will be explained in the challenges later.

The algorithms for the valence and arousal calculations are another key point in the full system. Something that should be noted is that the algorithm is based on the theory provided by [25] and research done in the background research. The valence and arousal functions take in the current heartrate and GSR values and the previous heartrate and GSR values. These will with the help of case-based reasoning be turned into a RGB-value, which can then be displayed by the LED's. Both algorithms have a set of cases to against which the can compare the gathered heart rate and GSR value. The first example is the pseudocode for the arousal_to_light() function. Something that has been discovered during the writing of the pseudocode is the fact that there are duplicate cases in both the arousal_to_light() as well as the valence_to_light() functions. This does not affect the working, but it might look messy.

```
Function arousal_to_light(Current values, previous values)
    if there is no change or a small rise heart rate compared to the previous value
        if there is a small drop or no change in GSR compared to the previous values
            if heart rate and gsr are within the set limits
                redvalue = (((heart_rate * (255 / HR calibration max     value)) + (255 - (gsr
* (255 / GSR calibration max value)))) / 2
             greenvalue = (((255 - (heart_rate * (255 / HR calibration max value))) + (gsr *
(255 / GSR calibration max value))) / 2)
                if redvalue > 255:
                    greenvalue = greenvalue - (redvalue - 255)
                    redvalue = 255
                if greenvalue < 0:
                    greenvalue = 0
                elif redvalue < 0:
                    redvalue = 0
                return color value in the form of RGB(redvalue, greenvalue, 0)
    elif the heart rate is between 50 and 88 and the gsr value is within normal limits:
        redvalue = ((((heart_rate * (255 / HR calibration max value)) – HR correctionvalue)
+((255 - (gsr * (255 / GSR calibration max value)) - GSR_correctionvalue))) / 2)
                greenvalue = (((255 - (heart_rate * (255 / HR calibration max value))) + (gsr
* (255 / GSR calibration max value))) / 2)
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        return color value in the form of RGB(redvalue, greenvalue, 0)
    elif the heartrate is higher than 88 and the gsr is lower than 100
        redvalue = (((heart_rate * (255 / HR calibration max value)) + (0.5 * (255 - (gsr *
(255 / GSR calibration max value))))) / 1.5)

        greenvalue = (((255 - (heart_rate * (255 / HR calibration max value))) + (0.5 * (gsr *
(255 / GSR calibration max value)))) / 1.5)
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        return color value in the form of RGB(redvalue, greenvalue, 0)
    elif the heartrate is between 50 and 88 and the gsr value is between 100 and 400
        redvalue = ((((heart_rate * (255 / HR calibration max value) - HR_correctionvalue)) +
(0.5 * ((255 - (gsr * (255 / GSR calibration max value))) - GSR_correctionvalue))) / 1.5)
        greenvalue = (((255 - (heart_rate * (255 / HR calibration max value))) + (0.5 * (gsr *
(255 / GSR calibration max value)))) / 1.5)
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        return color value in the form of RGB(redvalue, greenvalue, 0)
    elif there is a small drop or no change in gsr value compared to the previous values
```

```
        if check if the heartrate is higher then 88 and the gsr lower then 100
            redvalue = (((0.5 * (heart_rate * (255 / HR calibration max value))) + (255 - (gsr
* (255 / GSR calibration max value)))) / 1.5)
            greenvalue = (((0.5 * (255 - (heart_rate * (255 / HR calibration max value)))) +
(gsr * (255 / GSR calibration max value))) / 1.5)
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        return color value in the form of RGB(redvalue, greenvalue, 0)
    elif the heartrate is between 50 and 88 and the gsr value is between 100 and 400
        redvalue = ((0.5 * (((heart_rate * (255 / HR calibration max value)) -
HR_correctionvalue)) + (255 - ((gsr * (255 / GSR calibration max value)) -
GSR_correctionvalue))) / 1.5)
        greenvalue = (((0.5 * (255 - (heart_rate * (255 / HR calibration max value)))) + (gsr
* (255 / GSR calibration max value))) / 1.5)
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        return color value in the form of RGB(redvalue, greenvalue, 0)
    elif the heartrate is higher then 88 and the gsr lower then 100
        redvalue = (((heart_rate * (255 / HR calibration max value)) + (0.5 * (255 - (gsr *
(255 / GSR calibration max value))))) / 1.5)
        greenvalue = (((255 - (heart_rate * (255 / HR calibration max value))) + (0.5 * (gsr *
(255 / GSR calibration max value)))) / 1.5)
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        return color value in the form of RGB(redvalue, greenvalue, 0)
    else:
        return a black value, so (0,0,0)
```

The first case, which will be taken as an example here, applies to the situation in which the input is an overall slowly rising heart rate and a slowly lowering GSR value compared to the previously monitored heart rate and GSR value. As has been mentioned in the state of art previously, this is a sign that someone is in general being a bit more aroused. To be sure of this, the system checks if the person has a heartrate above a certain threshold, in this case set at 88 and the GSR value is below a certain threshold, in this case 100. If both these cases are met, this means the person is aroused and the input from the database is used to calculate the intensity of this arousal. So if the heart rate is much higher than the threshold, and the GSR value much lower than the threshold, the value will be extremely red, while if it is just above the threshold of 88 for heart rate, and just below the threshold of 100 for GSR, the colour will not be a red, as some green will be mixed in

This can be seen for each of the cases defined for arousal. The formula for colour calculations is sometimes slightly altered, in order to get a more accurate result. As the heart rate is low and the GSR is high, the system should show a very green value. The formulae, as well as the values of HR calibration max value, GSR calibration max value, HR_correctionvalue and GSR_correctionvalue are used to compensate for this. These values can be tweaked in order to improve the working of the algorithm.

As can also be seen in de pseudocode above, if the system cannot put the values received from the database into a certain case, it will return an RGB value of 0,0,0, which will make sure the LED stays off. Something that should be noted is that during the background research, it has been found that the best results would be achieved by using a proper AI that has been trained on a dataset, an example could be an SVM classifier. However, due to the lack of time and a training dataset, the choice was made to implement this case-based system instead.

Next to the arousal_to_light() function, there is also the valence_to_light() function. The pseudocode for this can be found below

```
Function valence_to_light(Current values, previous values)
    if the difference between heart rate and previous heart rate and gsr and previous gsr is
within the limits of the coefficients used
        if the heart rate and gsr is within normal human limits
            bluevalue = ((255 - ((abs(difference between previous and current heart rate) *
(130 / HR_Valence_devcoefficient))) + (255 - (abs(difference between previous and current gsr)
* (150 / GSR_Valence_devcoefficient)))) / 2)
            yellowvalue = (((abs(difference between previous and current heart rate) * (130 /
HR_Valence_devcoefficient)) + (abs(difference between previous and current gsr) * (150 /
GSR_Valence_devcoefficient))) / 2)
            if bluevalue > 255:
                yellowvalue = yellowvalue - bluevalue
                bluevalue = 255
            elif yellowvalue < 0:
                yellowvalue = 0
            elif yellowvalue > 255:
                yellowvalue = 255
            return colour value in the form of RGB(yellowvalue, yellowvalue, bluevalue)
        else:
            return colour value as black in the form(0, 0, 0)
    elif the difference between heart rate and previous heart rate and gsr and previous gsr is
outside the limits of the coefficients used
        if the heart rate and gsr is within normal human limits
            bluevalue = (((abs(Difference between previous and current heart rate) * (130
/HR_Valence_devcoefficient2)) + (abs(Difference between previous and current gsr) * (150 /
GSR_Valence_devcoefficient2))) / 2)
            yellowvalue = ((255 - ((abs(Difference between previous and current heart rate) *
(130 / HR_Valence_devcoefficient2))) + (255 - (abs(Difference between previous and current
gsr) * (150 / GSR_Valence_devcoefficient2)))) / 2)
            if bluevalue > 255:
                yellowvalue = yellowvalue - bluevalue
                bluevalue = 255
            elif yellowvalue < 0:
                yellowvalue = 0
            elif yellowvalue > 255:
                yellowvalue = 255
            return colour value in the form of RGB(yellowvalue, yellowvalue, bluevalue)
            else:
                return colour value as black in the form(0, 0, 0)

    else:
        if the heartrate and gsr value are within human limits
            if Difference between previous and current heart rate > HR_Valence_devcoefficient:
                bluevalue = (((0.5 * (abs(Difference between previous and current heart rate)
* (255 / HR_Valence_devcoefficient2))) + (1.5 * (abs(Difference between previous and current
gsr)) * (255 / GSR_Valence_devcoefficient))) / 2)
                yellowvalue = (((255 - (0.5 * (abs(Difference between previous and current
heart rate) * (255 / HR_Valence_devcoefficient2)))) + (1.5 * (abs(Difference between previous
and current gsr)) * (255 / GSR_Valence_devcoefficient))) / 2)
                if bluevalue > 255:
                    yellowvalue = yellowvalue - bluevalue
                    bluevalue = 255
                elif yellowvalue < 0:
                    yellowvalue = 0
                elif yellowvalue > 255:
                    yellowvalue = 255
                return colour value in the form of RGB (yellowvalue, yellowvalue, bluevalue)
            elif Difference between previous and current gsr > GSR_Valence_devcoefficient:
                bluevalue = (((1.5 * (abs(Difference between previous and current heart rate)
* (255 / HR_Valence_devcoefficient2))) + (0.5 * (abs(Difference between previous and current
gsr)) * (255 / GSR_Valence_devcoefficient))) / 2)
                yellowvalue = (((255 - (1.5 * (abs(Difference between previous and current
heart rate) * (255 / HR_Valence_devcoefficient2)))) + (0.5 * (abs(Difference between previous
and current gsr)) * (255 / GSR_Valence_devcoefficient))) / 2)
                if bluevalue > 255:
                    yellowvalue = yellowvalue - bluevalue
                    bluevalue = 255
                elif yellowvalue < 0:
                    yellowvalue = 0
                elif yellowvalue > 255:
                    yellowvalue = 255
                return colour value in the form of RGB(yellowvalue, yellowvalue, bluevalue)
             else:
                return colour value as black in the form (0, 0, 0)
        else:
            return colour value as black in the form (0, 0, 0)
```

When using the first case as an example again, it will only apply if the changes in heart rate and GSR compared to the previously measured values are very small, and thus not a lot of activity in both is measured. This lack of activity is an indicator of a negative valence, as has been explored in the background chapter. Before the values are used to calculate the colour output, it is first checked if they are within limits, to make sure false readings are not used for calculating colours. These values are for now set at a minimum heart rate of 50 and a maximum GSR value of 400. When the values from the database are in range, the values are used in a formula to calculate colour. Again, like for the arousal function, the formula is sometimes slightly altered to make sure it properly applies for the situations. It also uses variables which can be set by the developer for the formula. The important difference is that for the valence calculations, it will use the difference value, to see if there is a lot of activity, while for arousal, it uses the raised state, so the higher heart rate and lower GSR as an indicator for arousal.

As has been mentioned for arousal, the valence function has the same system that when values cannot be classified in one of the cases, the system will set the RGB value to 0,0,0.

When the colour is returned, and used to colour the LED’s, there is another check to make sure that the value that is returned can be used as an RGB value. If the algorithm returns nil, the system will make sure the LED is set to an RGB value of 0,0,0, meaning it will be black. This to make sure the code will not crash when something unpredicted has happened and the algorithm returns something else then a RGB value.

6.3.3: Challenges

The development of the physicalization came with several challenges, some of them have already been mentioned. Not all of them will be discussed, as some were minor and were easily fixed. However, some could not be fixed, or it took a bit more effort in order to find the proper solution.

The first issue which was discovered was the brightness of the warm-white LED strip in the system. When powered up, it turned out to be too bright for the decorative lamp setting. The solution would be to integrate a dimmer into the design. As the electronics to design a dimmer circuitry which could be controlled by the raspberry were not available in time, a temporary solution was chosen which was an inline dimmer that was placed inside the installation, and would be set once to the desired brightness, but would require someone to take apart the installation to alter the dimmer setting. This inline dimmer was placed between the power supply and the warm-white LED strip.

Another issue that was found during the initial testing of the system was that it was difficult to read the timestamps from the ring, as there were no indications to guide someone. This was solved by adding 3d-printed numbers, as can be seen in picture 38 below.

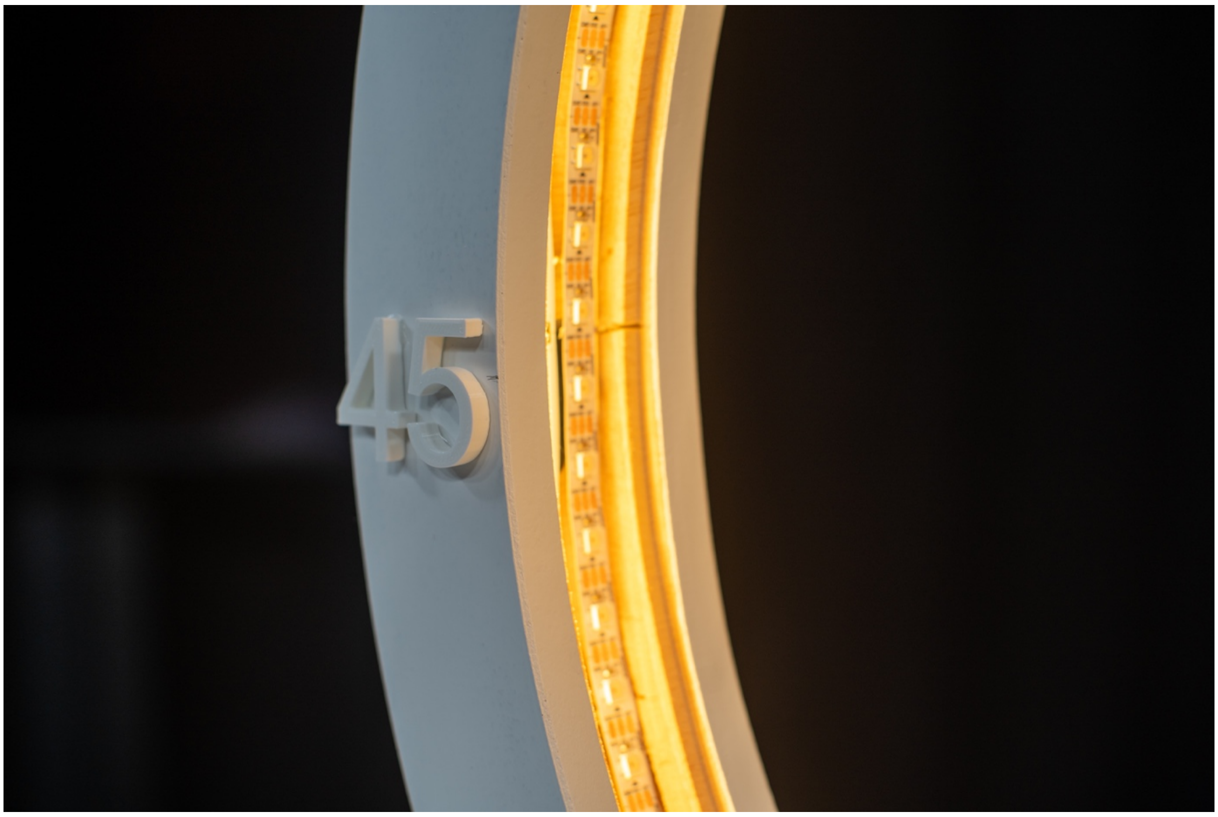


Figure 38: Close-up of added numbers onto the physicalization

However, the major challenges came from the software part of the installation. As has been mentioned before regarding figure 37, there are some duplicate pieces of code in the software. This is because the initial idea of the software was to have a separate way of updating the LED ring in case the state of the switch did not change. As it is fairly inefficient to recalculate all the values from the startpoint of the installation when the state has not changed, the initial plan was to only use the algorithm for the new incoming value, and store the previously gathered colour values in the stored variables file. This also explains why there are multiple extra variables and lists stored in the storedvariables.txt file. These were for this old function. However, it was found that this was not working, and the choice was made due to the lack of time to use the same system which was used when the state did change, as this seemed to work much better. This however meant that some parts of the code were duplicated in other functions. These however should in a later version of the system be removed and one function should be created. This will be elaborated on further in the discussion, as there are more parts of the code that need revision.

The final challenge during the development was the different inner workings of the library that controlled the colour LED strip. The library updates all the LED’s, even if only one should get a colour. The others will then be turned to black. This is different to how the same library works on the Arduino plaform, where only the LED that needs to be updated in the strip will be updated. The others will stay on their previously assigned colour. The python library will then first turn all the LED's to black, after which only the LED that is sent the colour to will get a colour. This is another reason why the first idea for updating of the LED's when the status would not change didn't work properly.

Chapter 7 – Results

7.1: User study

With the first prototype ready, the next phase was to do a user study. The goal of the user study is to find out how effective and useful is the proposed physicalization as a tool to convey a person's physiological state?

7.1.1.: Variables:

The Independent Variables that are involved in the user study are the type of emotion on display and the duration of the emotion on display. With these are the dependent variables, in this case the ease of use, ease of understanding, the attractiveness of the installation, the satisfaction of the user in regards of the installation, accuracy and the usefulness of the physicalization.

7.1.2: Study design:

The study includes 2 participants in pairs in each session, participant A, who will be exposed to both valence and arousal stimuli and thus provides data which can be physicalized. the other participant, participant B, observes the physicalization and take notes. The clips used to stimulate arousal and valence measures are randomised in order.

7.1.3: Participants

5 couples were recruited for the user testing, aged between 19 and 25. However, the age and experience regarding physiological state is not important. They all had a high-school standard of English, as the study is done in English. The recruitment was done via Facebook, WhatsApp and via word of mouth.

7.1.4 Tasks

As the goal of the study was to find out how effective and usable the proposed physicalization is, the tasks should account for that. The target was to have someone interact with the installation, while someone else provides the data that will be interacted with at the same time. This, as the concept is to be able to provide near real-time physicalisation of the data that is being captured by the wearable. The concept of interaction is very broad here, as the person can use all the different function in order to complete an observation. This goal of this observation is to see if someone can accurately and easily predict someone's physiological state by interacting with the physicalization. The tasks of observing entails looking at the installation and noting down the valence and arousal value at certain timestamps. These timestamps should be deducted from looking at the installation itself. As part of the goal for the user test is to find out the ease of use, the person is not told a lot about the installation prior to interacting with the installation but e provided with some reading material stating how the system works, in case there might be questions regarding the interaction with the physicalization. The other participant will be tasked with watching a set of 4 movie clips, selected from a database.

7.1.5: Procedure

The research had 5 sessions with each session having 2 different people, whereas those 2 people were acquainted with each other.

Each session took 45 minutes, and started with a short introduction into the research and the working and explanation regarding the installation was done. In this phase, both participants, which will be called participant A and participant B from this point onwards, were asked if they have read the ethical information brochure, if not they were asked to do this at that moment or it can be read to them. Afterwards, they were asked to sign the ethical consent letter. After this had been done, the participants were asked to choose a role for the research. Both roles will be further explained below.

The role of participant A was to go through the experience of several stimuli, and was therefore the data provider for the physicalization. They were asked to go to another room, adjacent to the room in which they started. In there, they first got an explanation about what was going to happen. They got some extra explanation regarding what was going to happen during the experiment, and were again reminded regarding their right, and what to do if they wanted to stop. If they understood, and were able to ask all their questions, they were asked to wear the wearable. In this room, there was a place to sit in front of a screen. On this screen, a set of different clips of movies were shown to stimulate arousal or valence in the participant. During a session, they were subjected to 2 different stimuli, valence and arousal, from which both they got two separate stimuli, so this means they saw 4 video clips. In between, there was some time to relax again and to have a conversation with the researcher. Caution was taken that the conversation with the researcher would be about a topic which would not influence the reading, as certain conversation topics might also work as a stimuli for a person.

The video's used as the stimuli were based on two out of the several scientific databases that have been developed as a tool for researchers to induce emotions in participants. These databases are: FilmStim [9] and Chieti Affective Action Videos (CAAV)[10]. These databases were selected as they have been used in multiple scientific studies regarding the induction of emotions to recognise them with physiological sensors. Other databases which could be used are the IAPS (The International Affective Picture System) [11], CEVS(China's Standard Emotional Video Stimuli Materials Library) and MAHNOB [12]. However, these do need special permissions to gain access to them.

If the user at any moment during the watching of the video's wanted to stop, the image and audio feed was cut immediately, no questions asked. If the user completed the session or stopped early, a short debrief would follow. This part of watching and debriefing took approximately 20 minutes, in which 4 clips of approximately 2 to 4 minutes each were shown. The clips and thus the connected physiological state, in this case arousal and valence, were mixed up between different participants.

The role of participant B was to observe the physicalization itself in the other room. They were also provided with some reading material regarding the installation(appendix G) to further inform them. This was done because part of the goal was also to find out the ease of learning with the installation. At this moment, they also could ask questions if they had any. As soon as participant A started with the watching, participant B was asked to start observing and interacting with the installation. Participant B was asked to write down the physiological states that could be observed, combined with the time the participant B could connect to it. Also, any conclusions based on what could be seen needed to be written down as well. It was explained that this was important for the next phase. They could use pen and paper provided by the researcher to write everything down. In the physicalization, two different modes were present. The valence and the arousal mode. Next to that, the participants could turn on or off the installation. After participant B was done observing the physicalization, they were asked to fill out the adapted version of the USE questionnaire[13], which can be found in appendix H.

After both participants were finished with their individual parts, they were asked to come together and discuss what just had happened based on the findings of participant B. This was done under observation of the researcher who took notes. Participant B was asked in this case to tell about an observation at a certain point, where participant A could then respond if this was indeed the case, and if not, what it should have been at that point. These discussions were done to check the agreement between the participants based on the observations made by B. This paired with how participant A felt and how participant B perceived together with the intended affect targeted by the video. This took 10 minutes.

In the end, both participants were debriefed regarding the research and thanked for participating. They would again be informed about their rights regarding participating.

The full procedure is summarised in the table(figure 39) found below, setting out the general steps for both of the participants.

Participant A	**Participant B**
Short introduction	Short introduction
Consent	Consent
Separate explanation about the emotion induction process	Reading of written instructions of the installation
Emotion induction by video part 1(Valence/Arousal)	First observation part with the physicalization
Emotion induction by video part 2(Valence/Arousal)	Second observation part with the physicalization
Short debrief regarding the previous experience	Filling out questionnaire based upon system usability scale questionnaire
Discussion with participant B based upon participant B's notes	Discussion with participant A based upon previously made observations
Wrap-up	Wrap-up

Figure 39: Table with step-by-step research outline

7.1.6: Hypothesis:

Due to the shape and light, the user of the installation understands the relative abstract concepts of valence and arousal based on a person's real physiological state better, combined with observing the installation and can create meaningful discussion points related to the observations. The installation was an easy to learn experience, and the user is satisfied with the installation, as it was very usable.

7.2: User study results

The amount of participant for the user study was very limited, as mentioned previously, there were only 5 pairs, so in total 10 participants. It must be noted that this means no conclusions can be made based on the results that will be presented below. However, some preliminary statements regarding the 5 results could be made. This user study could be a good starting point as a pilot for more testing, but based on the results other steps could be taken as well.

The goals of this part of chapter 7 is to discuss the results according to the goal set out at the beginning of chapter 7. So, the result will be split up in the effectiveness of the system and the usefulness of the system. A third part will be dedicated to the response gained from the discussion at the end of each research. In this, participants made several statements and shared their thoughts regarding the installation.

The data gained during the research can be found in appendix I. This consist out of 6 tables in which the data from each user study regarding the observational part is collected, and a final table containing the results from the USE questionnaire for each session with an overall average score.

Each table that can be found in appendix I, with the exception of figure i.6 in appendix I, consist out of several categories, as can be seen in figure 40 below.

timestamp	**hr**	**gsr**	**arousal noted**	**valence noted**	**gender dataprovider**	**video playout**	**Clip arousal/valence score**

Figure 40: Example of the categories in the table

In this overview, the first category is timestamp. This is the timestamp that is stored on the database, and all the other points have been matched to this timestamp. The second and third category hr and gsr are the raw data collected by the wearable. This is also stored on the database, and directly imported into the table. The arousal noted category is the arousal observations made by the participant. The participants were not told in which form to write down this observation, so some of the tables might look different than others, as some participants scored the levels from the light with numbers, while others used terms like high and low. The next category is valence noted. Like in the arousal noted, these are the observations of the participant observing and again participants were not told how to write it down. So here the filled in values might also differ per table. It should be noted that for both the arousal noted and valence noted category, there might be empty spots in the table. This is when people could not read data from the installation, and left that part of the their form empty. Gender data provider is the category which states if the person of the wearable was male, female or non-binary. Video playout provides information when videos were started and stopped. The final category provides information on what emotion the video that is played would induce, based on the data provided by the video database[9].

For the table that provides information on the USE questionnaire scoring by the observing participants, a scale from 1 to 7 was used, as can be seen in figure i.6 in appendix I.

Notes and observations made by both the participants and the researcher will be discussed. However, due to privacy reasons, they will not be provided as a separate raw source in the appendix.

7.2.1 efficiency of the system

The efficiency in case of the physicalization entails the speed in which the data would be available to the observer. The raw data available in the figure i.1 to i.5 in appendix I do not provide the exact timestamps of when the data reaches which step in the full installation. However, based on comments during the discussions and observing the inner workings during the first 5 user tests, it could be stated that the efficiency of the system is fairly low, as three of the 5 groups mentioned the waiting between the updating of the system would prevent them from quickly getting an overview of both states, and thus it would take them some time to gain a complete picture of someone's physiological state. One participant mentioned that it would observe one metric, valence or arousal, for a few minutes, and then would switch and would start updating the other metric from the previous point of switching onwards.

7.2.2 usefulness of the system

The usefulness of the system entails a lot of different aspects, so these will be further split up in chapter 7.2.2. The topics discussed will be the accuracy, the data representation itself and the scores from the USE questionnaire.

7.2.2.1 accuracy

The figures i.1 to i.5 in the appendix I will provide a good overview of the accuracy of the system, especially when looking at the arousal and valence noted categories. The system does not perform well in terms of accuracy. In two cases, study 2(figure i.2) and 3(figure i.3), no valence data would be presented by the physicalization. After some checking, it turned out the data would actually reach the physicalization, however, it could not be classified by the algorithm in one of the cases programmed in. In all the other studies, with the exception for study 4, a lot of the data that would come into the system could not be classified by the system in one of the cases, thus the light at that timestamp would stay off, when looking at the notes of the researcher and the observations of the participant. When looking at the data that was classified, there is only one datapoint in the 5 studies that is classified properly. The rest of the data was misclassified when looking at the observations of the participants and the notes of the researcher regarding the physicalization and comparing them to the emotional induction score of the video clip played at the timestamp. Another issue was a misplaced sensor in the study number 1(figure i.1), where the heart-rate readings were completely off and consistently high, which did not reflect reality at that point.

Something that should be noted is the fact when the light would not turn on, participants that would do the observations were slightly confused and thought the system did not work. This confusion was amplified if multiple datapoints would come back in black.

7.2.2.2 data representation

The topic of data representation has several angles. The angle from the USE questionnaire will be discussed in the next part. The data-representation was also mentioned by the participants observing the installation. During two user tests, it was mentioned that it was hard to get a proper overview, because the system could only display one data stream at the time. They suggested maybe adding another LED strip in order to be able to get both data streams(valence and arousal) in the same picture. Another comment that was made regarding the LED's was the fact that it was sometimes hard to distinguish between the neutral state and the state just next to this. It was mentioned that this made the accuracy of reading off the data more difficult. Another fact that was mentioned is the fact that the learning curve to intuitively be able to see how a person is feeling was quite steep, and a suggestion was made that maybe using different colours to represent different emotions would make it easier, so red would be anger and blue would be relaxed as an example. This would leave out the part of the gradient. A comment that was added to this suggestion was the fact that the current way of representation would be more effective if you want to read of multiple metrics from the system, thus gaining more knowledge.

4 of the 5 persons observing also noted that they would have to get quite close to be able to see a person's physiological state and connect it to the right timestamp. They would not be able to observe it properly from a further distance. This could be seen as a disadvantage, but in terms of privacy, it means it takes more time to deduct someone's personal data.

7.2.2.3 USE questionnaire

Although the system might not have always done what it was supposed to do, people overall scored it as being very useful, with an average of 5 out of 7 points, according to figure i.6 in appendix I. Especially the simplicity of operation and the possibility to quickly recover from mistakes was noted as very beneficial. One participant mentioned that the minimalistic design made it very suitable for the task at hand, as it would not be distracting. Something the installation did not score very well on was the fact a person could use it without instructions. However, this is also intentionally, as the privacy of the user is guaranteed. Someone not knowing about the installation would not be able to deduce any information from the physicalization.

Another beneficial point was the fact that the installation was very easy to learn about, as can be seen in the scoring of the ease of learning category in figure i.6 in appendix I. This means that if a person is willing to use it, they can quickly understand the inner workings and can interact with it to gain more knowledge about the other person wearing the wearable.

When looking at the satisfaction in regard to the product, people were overall satisfied. However, not a lot of people had a use for the physicalization themselves. Considering this physicalization was developed with a few more specific use cases in mind, these results are not alarming. The rest of the satisfaction metrics were an above average, which means the observers were happy with using it.

7.2.3 additional observations

During the user tests, a lot of additional comments regarding the installation were made. Some of these have already been mentioned in the previous parts, other will be mentioned below.

First of all, 3 of the 5 participants that did the video watching mentioned that the videos in the database were really dated, and they mentioned this might influence someone’s respond to the videos. One of the participants also mentioned that if you might already have seen the clip, or if you know the actor and do not like it, you might be biased, and the emotional response to the video clip might also be different. These are all factors that could influence the way you experience the video clips.

Then there was an issue raised by one of the participants that did the observing. This regarded the switch in the physicalization that would select the mode, as this sometimes got stuck in a neutral position which would break the code, and nothing would be physicalised. This would automatically fix itself once a minute had passed, but it meant the physicalization would not do anything for 2 minutes.

A final observation made was the fact that due to the amount of LED’s, in this case 90 LED’s, it would become an issue determining the right state at the right moment. Because sometimes two LED’s would present 1 minute, while at other points, 1 LED would represent 1 minute. It was mentioned by several participants that this would be confusing.

Chapter 8 – Discussion & Future Work

In this chapter, the goal is to reflect upon the project, see where the shortcomings are, and discuss what could be done in future work. As this project was meant to be an exploration of developing a near-real time data physicalization system, there is a list of points of attention which would need further looking into or should be changed in order to improve the full system.

The first part of the installation that would need adaption for the future would be the algorithm inside the physicalization. As has been mentioned before, the algorithms that take in the Heart rate and GSR data are at the moment case-based. However, as has been seen during the user testing, this leads to a lot of classification mistakes. In order to improve accuracy and performance related to the algorithm, implementing an AI, of which an example could be a SVM classifier type of system, could be considered. Implementing such a system would however require a large existing database of data regarding emotions and physiological data from the sensors in order to train the AI. However, when choosing to only change the algorithm with for example calibration and new cases would be another option next to implementation of an AI as a solution to improve the accuracy.

Another factor that would help the accuracy when looking at the software would be proper calibration steps for the sensor each time it switches user. As has been mentioned previously, every individual has slightly different base-readings of heart rate and GSR levels. Thus, calibrating for these differences when the person wears the wearable for the first time would solve issues regarding these differences when applied in processing of the data. This could be variables in code that would be set during a calibration period at the start of a session or a system where a certain mode in the system is activated to do the calibration.

Next to the parts that could be altered with rewriting the software of the system, there are other factors playing a part in the accuracy and usability of the system. A first point is the sensors currently used are consumer-based sensors, and especially the Heart-rate sensor, just produces a value for heart rate. Both the GSR and Heart rate sensor have no medical clearance or certification, so their accuracy cannot be proven without proper testing. Furthermore, in order to do proper emotion classification, it would be much more useful to use HRV instead of HR, which cannot be provided by the current sensors. Even better would be using the RAW data from the sensors themselves, without being processed. Then, multiple different metrics could be taken and combined. This could then be used to create a reliable system with a much higher accuracy. This is also backed up by the previously discussed background section regarding the use of heart rate sensors. Also, take into consideration that sensors produce noise, and this should then also be countered by the new sensors with proper code.

A final point that would significantly improve the accuracy and usability of the system would be increasing the data-rate from the wearable and make sure all the data is used for processing. This has been mentioned during the realisation phase, however, implementation was not done due to battery issues and limiation with the wearable, as well as the way code was implemented on the raspberry. It does not take away that the current implementation of only taking a snapshot each minute compromises the accuracy in a significant way. Problems like reading-errors have suddenly a much larger impact on the resulting colour. Also, the small sudden changes in someone's physiological state might only be present for under a minute, meaning the system would not register them. The advice is to increase the update rate to at least once a second, preferable even faster, and combine this with a method of data sampling in order to filter out reading errors. A method that could be suggested is using windowing, however, other methods could be used as well. Also, on the raspberry side, all these datapoints should be taken in and classified, from which an average for that specific minute could be calculated. In this way, the accuracy of the results would significantly increase. It also opens up possibilities to make a snapshot mode, in which the full ring will display just one colour, but update this colour each second according to the reading coming in. This could offer people a real-time insight into someone's physiological state.

Next to these improvements of the accuracy and usability of the system, there are also parts of the code that could be altered to improve efficiency. Especially when starting to implement the previously mentioned future improvements, efficiency becomes key in order to keep all the processing on the small computer inside the physicalization. A first improvement that was already planned for the system but could not be implemented in the code would be only processing the new incoming data, and store the already processed colours on the system, so it would be able to quickly recall them and only process the new data. As mentioned before in the realisation, it has actually been tried, however, due to the lack of time this function was never fully realised. Parts of it have however been left in the code. The major issues that were found during the trail of implementing it were the fact the library for the LED's was programmed to first wipe all the LED's colour data and then only apply the newly added data. This meant the LED's itself could not store the data and using the storedvariables.txt would not properly solve this issue.

Another point regarding the implementation of the code, which already has been briefly mentioned, is the fact that the system currently uses Cronjob to keep the code running. However, a major limitation is the minimum interval time of 1 minute. This would not work if the previously mentioned points need to be implemented. A new system that would update the code according to the incoming data would make the code run according to the data instead of relying on a fixed time interval.

A final adaptation that should be considered when improving the backend of the system, is the implementation of a more secure way to upload and retrieve the data from the database. This would be done by using a method called prepared statements. These would make sure the privacy of the user is more guaranteed and would make certain implementation in light of further expanding the system easier to accomplish.

Next to these accuracy and usefulness related points of improvement, there are also other aspects that need to be discussed. The first one was already briefly mentioned during chapter 7, but the user tests need to be expanded, and include a more diverse range of data providers, where data can be retrieved from different genders, different ethnicities and different ages in order to properly be able to conclude on the inner workings of the system. However, a choice could be made to first redesign the algorithms in order to get those fully working, after which a new round of user testing with a larger group would be initiated. This might result in a more accurate overview of the potentials of the system, as well as a clear starting point for further development.

Then there are also several hardware related points that could be discussed, of which some have already been mentioned during the realisation and user testing phases. First of all, the battery management system of the wearable needs to be revised in order to make some of the previously mentioned points even possible. Next to that, it would greatly improve the user experience, as charging would not mean taking the full watch apart anymore.

During the user testing some issues were mentioned regarding the physicalization, which were the issues related to the readability and the operation. Some of those would be fixed when implementing the previously mentioned instant update system, while others would mean adapting the amount of LED’s and maybe even the placement of a diffuser over the LED’s. Something that should be kept in mind though is the fact people really liked the simplicity of the operation of the full system, which is a major benefit in making the system fully accessible to a broad audience.

In this chapter, the largest points on discussion with a focus on what could be done for the future have been discussed. These are mainly software related and focus on improving the accuracy of the system. As is the opinion of the research, this should be seen as the first point of attention. Next to the points in this chapter, some other points have been raised, especially during the realisation phase. Next to the points related to the product, a separate track that is of great influence on the project is the fact that different gender and ethnicities all have slightly different ways of how physiological state is presented through bio signals. Although the basic theory is common, these different ways should be explored further, in order to gain a better understanding and applying the knowledge to make the product applicable in a broader setting.

Chapter 9 – Conclusion

The goal of this research was to find the challenges and opportunities in developing an interactive and intuitive way of representing human affect using dynamic data. Based on the background research, this topic had still little research relating to it. Therefore, this research would be an exploration and would result in the development of a dynamic data physicalization with the use case of physiological data, resulting in several final conclusions due to the broadness of the topic.

Based on the research done it can be proven that it is possible to create a dynamic data physicalization. This prove is in the physicalization combined with the wearable that has been created, which succeed in the goal to provide an easy to use and thus accessible way to intuitively interact with data and gain knowledge, as can be proven when looking at results during the user studies.

A second part of the research was developing a pipeline, which can be seen as a blueprint for future developments of physicalizations of dynamic data. The resulting flowchart has been developed based on the steps taken and discoveries made during the development of the physicalization created in this project, as well as the background research done at the start of the research.

A final part in the conclusion of the research involves the challenges that were found in the background research, as well as during the development, relating to both specific issues with physicalization of physiological data, as well as general issues relating to the physicalisation of real-time dynamic data.

In the final stages of the research, the user testing and the resulting discussion provide an overview of especially the challenges and future work that is needed in order to further explore the topic of physicalization of dynamic data. However, this research provides a basis from which multiple different deeper explorations into subcategories of physicalization of dynamic real-time data can be conducted.

References

[1]. L. Santamaria-Granados, M. Munoz-Organero, G. Ramirez-Gonzalez, E. Abdulhay, and N. Arunkumar, "Using deep convolutional neural network for emotion detection on a physiological signals dataset (AMIGOS)," IEEE Access, vol. 7, pp. 57–67, 2019.

[2]E.-H. Jang, B.-J. Park, M.-S. Park, S.-H. Kim and J.-H. Sohn, "Analysis of physiological signals for recognition of boredom pain and surprise emotions", J. Physiol. Anthropol., vol. 34, no. 1, pp. 25, 2015.

[3] C. Sedikides and E. G. D. Hepper, "Self-improvement: The motive to self-improve," Soc. Personal. Psychol. Compass, vol. 3, no. 6, pp. 899–917, 2009.

[4] Mader A.H. and Eggink W., "A Design Process For Creative Technology", Proceedings of the 16th International conference on engineering and product design education, Enschede, The Netherlands, 2014, pp. 568-573

[5] N. Palombi, "10 color meanings: the psychology of using different colors," Webflow, 11-Nov-2021. [Online]. Available: https://webflow.com/blog/color-meanings. [Accessed: 14-Jun-2022].

[6] Ifttt.com. [Online]. Available: https://ifttt.com/explore. [Accessed: 06-Jul-2022]

[7] H. Z. Wijasena, R. Ferdiana, and S. Wibirama, "A survey of emotion recognition using physiological signal in wearable devices," in 2021 International Conference on Artificial Intelligence and Mechatronics Systems (AIMS), 2021

[8] S. Basu et al., "Emotion recognition based on physiological signals using valence-arousal model," in 2015 Third International Conference on Image Information Processing (ICIIP), 2015.

[9] "FilmStim," Uclouvain.be. [Online]. Available: https://sites.uclouvain.be/ipsp/FilmStim/film.htm. [Accessed: 29-Jun-2022].

[10] A. Di Crosta et al., "The Chieti Affective Action Videos database, a resource for the study of emotions in psychology," Sci. Data, vol. 7, no. 1, p. 32, 2020.

[11] "International affective picture system (IAPs)," Umass.edu. [Online]. Available: https://www.umass.edu/research/guidance/international-affective-picture-system-iaps. [Accessed: 29-Jun-2022].

[12] S. Koelstra, "HCI tagging database," Mahnob-db.eu. [Online]. Available: https://mahnob-db.eu/hci-tagging/accounts/login/?next=/hci-tagging/search/. [Accessed: 29-Jun-2022].
[13] "USE Questionnaire: Usefulness, Satisfaction, and Ease of use," Garyperlman.com. [Online]. Available: https://garyperlman.com/quest/quest.cgi?form=USE. [Accessed: 29-Jun-2022].

[14] J. Zhao and A. V. Moere, "Embodiment in data sculpture: A model of the physical visualization of information," in Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts - DIMEA '08, 2008.

[15] D. Lockton, D. Ricketts, S. Aditya Chowdhury, and C. H. Lee, "Exploring Qualitative Displays and Interfaces," in Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '17, 2017.

[16] K. Sauvé, D. Potts, J. Alexander, and S. Houben, "A change of perspective: How user orientation influences the perception of physicalizations," in Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, 2020.

[17] Y. Jansen et al., “Opportunities and Challenges for Data Physicalization,” in Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI ’15, 2015.

[18] B. Lee, E. K. Choe, P. Isenberg, K. Marriott, J. Stasko, and T.-M. Rhyne, “Reaching broader audiences with data visualization,” IEEE Comput. Graph. Appl., vol. 40, no. 2, pp. 82–90, 2020.

[19] G. Calvert, C. Spence, and B. E. Stein, The Handbook of Multisensory Processes. MIT Press, 2004.

[20] D. Menheere, E. van Hartingsveldt, M. Birkebæk, S. Vos, and C. Lallemand, “Laina: Dynamic Data Physicalization for Slow Exercising Feedback,” in Designing Interactive Systems Conference 2021, 2021, pp. 1015–1030.

[21] Arora, A., Chakraborty, P., Bhatia, M.P.S. Problematic Use of Digital Technologies and Its Impact on Mental Health During COVID-19 Pandemic: Assessment Using Machine Learning (2021) 348, pp. 197- 221.

[22] 12th International Conference on Intelligent Human Computer Interaction, IHCI 2020 (2021) 12615 LNCS, 942 p.

[23] 12th International Conference on Intelligent Human Computer Interaction, IHCI 2020 (2021) 12616 LNCS, 942 p.

[24] T. Zhang, A. El Ali, C. Wang, A. Hanjalic, and P. Cesar, “CorrNet: Fine-grained emotion recognition for video watching using wearable physiological sensors,” Sensors (Basel), vol. 21, no. 1, p. 52, 2020.

[25] Shu et al., “A review of emotion recognition using physiological signals,” Sensors (Basel), vol. 18, no. 7, p. 2074, 2018.

[26] A. Kammoun, R. Slama, H. Tabia, T. Ouni, and M. Abid, “Generative Adversarial Networks for face generation: A survey,” ACM Comput. Surv., 2022.

[27] E4 wristband,” Empatica. [Online]. Available: https://www.empatica.com/research/e4/. [Accessed: 30-Jun-2022].

[28] J. M. Peake, G. Kerr, and J. P. Sullivan, “A critical review of consumer wearables, mobile applications, and equipment for providing biofeedback, monitoring stress, and sleep in physically active populations,” Front. Physiol., vol. 9, 2018.

[29] G. Li et al., “Multimodal biosensing for vestibular network-based cybersickness detection,” IEEE J. Biomed. Health Inform., vol. 26, no. 6, pp. 2469–2480, 2022.

[30] M. Alfaras et al., “From Biodata to Somadata,” in Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, 2020.

[31] F. Gasparini, M. Giltri, and S. Bandini, “Discriminating affective state intensity using physiological responses,” *Multimed. Tools Appl.*, vol. 79, no. 47–48, pp. 35845–35865, 2020.

[32] S. Saganowski, “Bringing emotion recognition out of the lab into real life: Recent advances in sensors and machine learning,” Electronics (Basel), vol. 11, no. 3, p. 496, 2022.

[33] E. Yuda, T. Tanabiki, S. Iwata, K. Abe, and J. Hayano, “Detection of daily emotions by wearable biometric sensors,” in 2019 IEEE 1st Global Conference on Life Sciences and Technologies (LifeTech), 2019.

[34] H. Feng, H. M. Golshan, and M. H. Mahoor, “A wavelet-based approach to emotion classification using EDA signals,” Expert Syst. Appl., vol. 112, pp. 77–86, 2018.

[35] K. Hovsepian, M. al'Absi, E. Ertin, T. Kamarck, M. Nakajima, and S. Kumar, "CStress: Towards a gold standard for continuous stress assessment in the mobile environment," *Proc. ACM Int. Conf. Ubiquitous Comput.*, vol. 2015, pp. 493–504, 2015.

[36] G.Ö.Z.E.L. Shakeri et al., "RadioMe: Challenges During the Development of a Real Time Tool to Support People with Dementia", 2021.

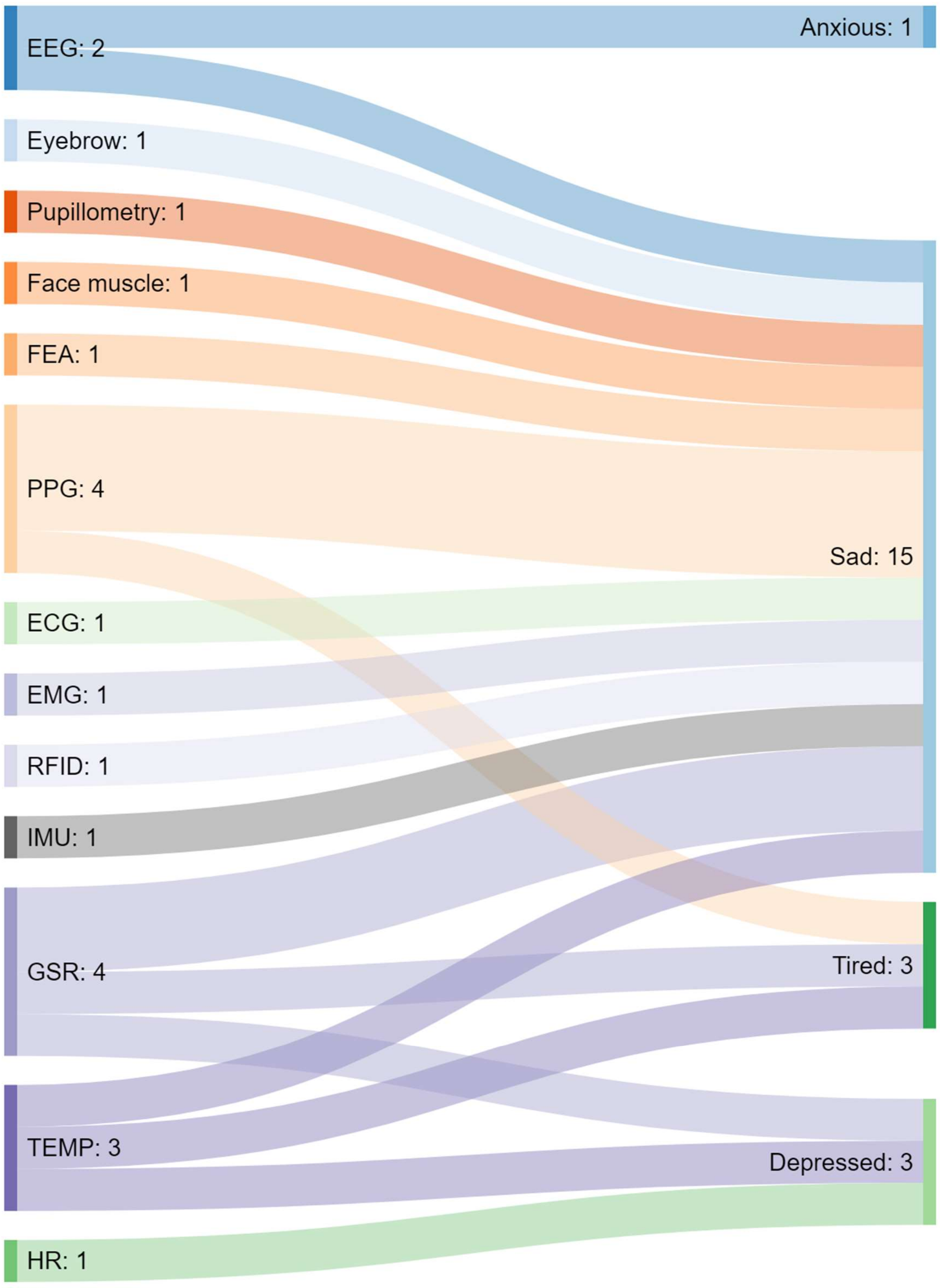
[37] R. van Loenhout, C. Ranasinghe, A. Degbelo, and N. Bouali, "Physicalizing sustainable development goals data: An example with SDG 7 (affordable and clean energy)," in CHI Conference on Human Factors in Computing Systems Extended Abstracts, 2022.

[38] Fuentes Bongenaar, M.S. "Towards a taxonomy of quantitative emotion measurement using wearables for inclusive user experience research," University of Twente, 2022.

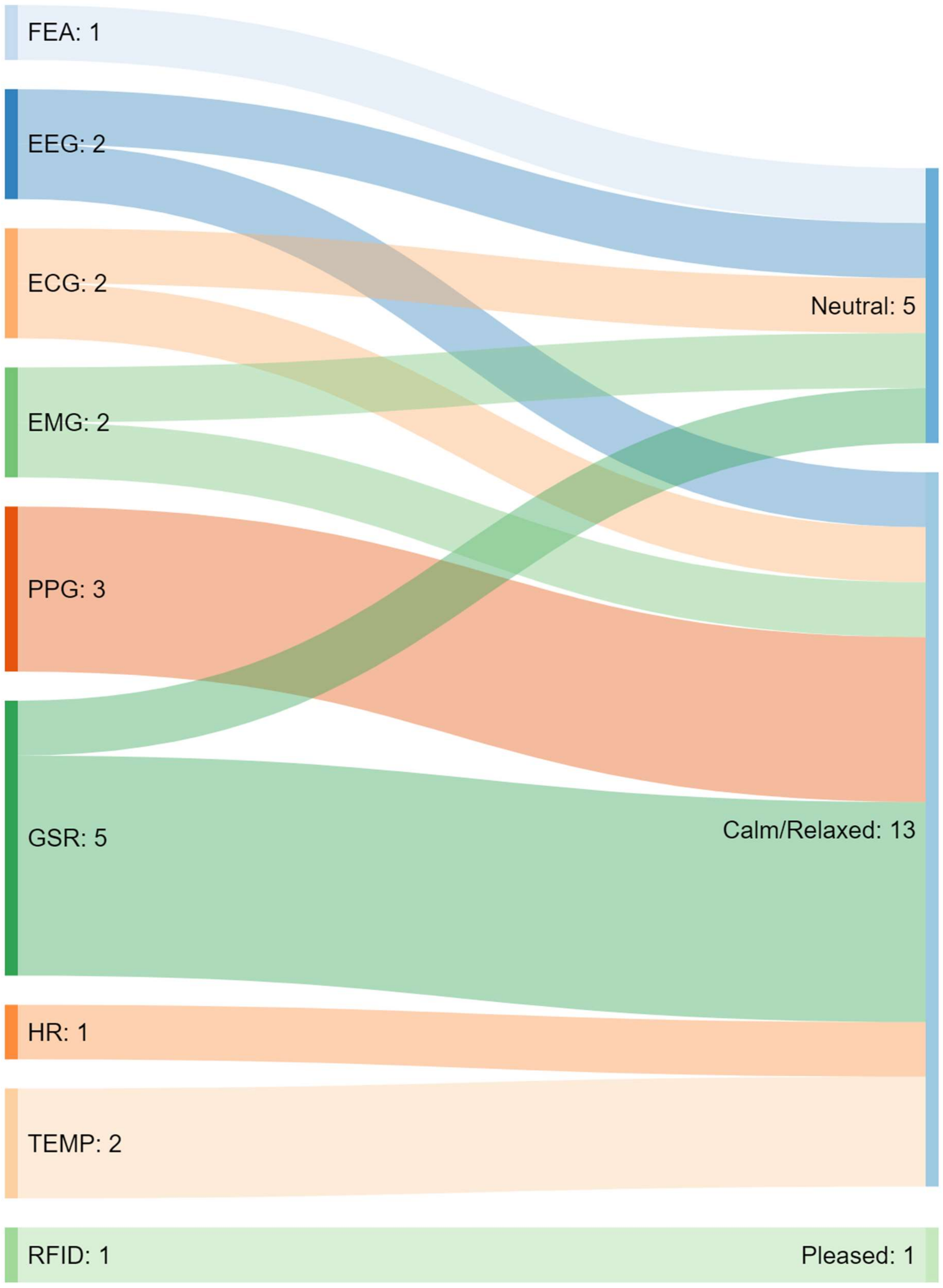
Appendix A through I

Appendix A: Sankey Diagram

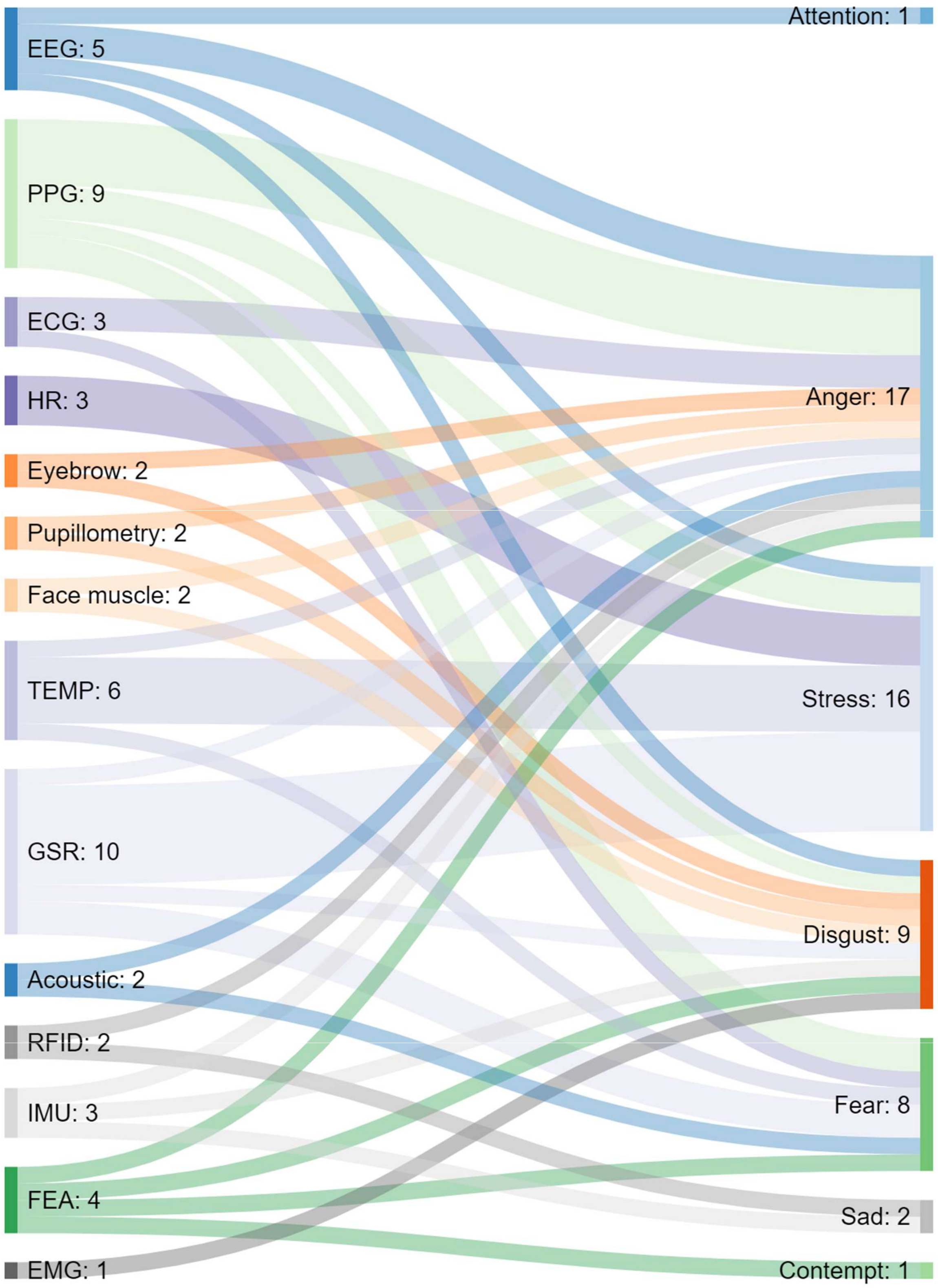
Source: [38]



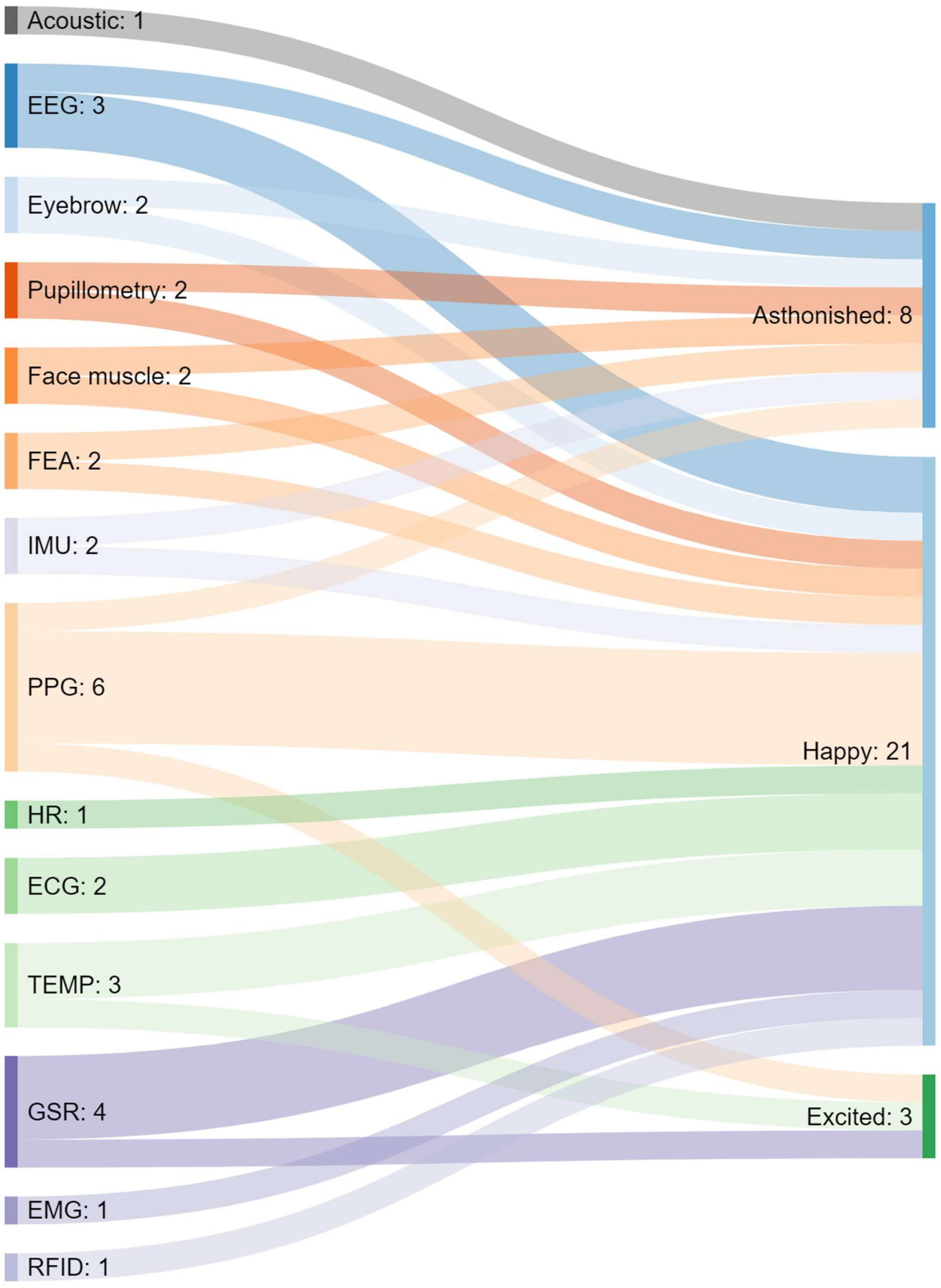
Sankey diagram showcasing the positive valence/positive arousal emotions and the sensors used to measure them.[38]



Sankey diagram showcasing the positive valence/positive arousal emotions and the sensors used to measure them.[38]



Sankey diagram showcasing the positive valence/positive arousal emotions and the sensors used to measure them.[38]



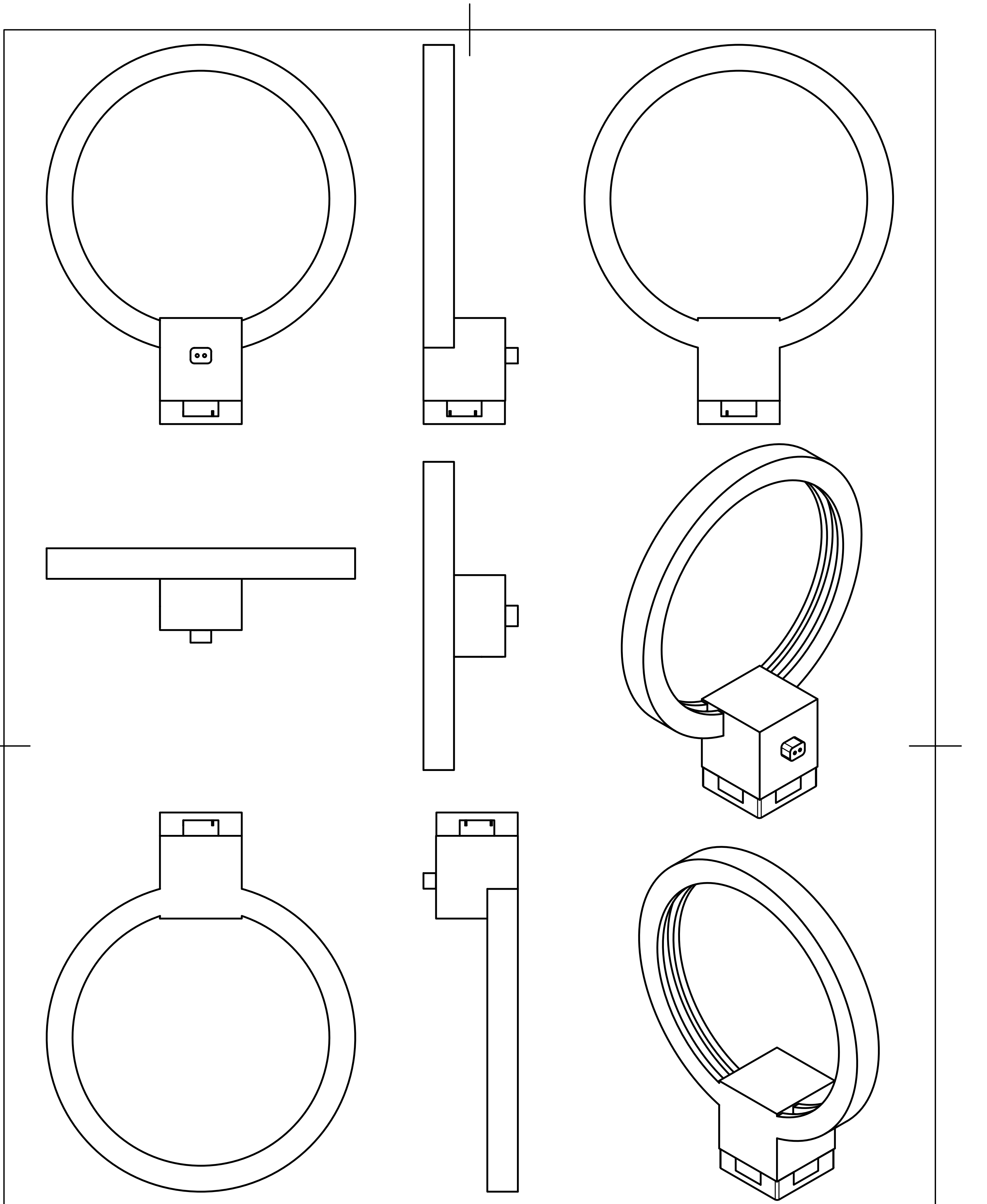
Sankey diagram showcasing the positive valence/positive arousal emotions and the sensors used to measure them.[38]

Appendix B: The relationship between emotions and physical features[33]

	Anger	Anxiety	Embarrassment	Fear	Amusement	Happiness	Joy
Cardiovascular							
HR	↑	↑	↑	↑	↑↓	↑	↑
HRV	↓	↓	↓	↓	↑	↓	↑
LF		↑		(--)		(--)	
LF/HF		↑			(--)		
PWA				↑			
PEP	↓		↓	↓	↑	↑	↑↓
SV	↑↓	(--)		↓		(--)	↓
CO	↑↓	↑	(--)	↑	↓	(--)	(--)
SBP	↑	↑	↑	↑	↑--	↑	↑
DBP	↑	↑	↑	↑	↑--	↑	(--)
MAP			↑	↑	↑--	↑	
TPR	↑			↓	↑	↑	(--)
FPA	↓	↓		↓	↓	↑↓	
FPTT	↓	↓		↓		↑	
EPTT		↓		↓		↑	
FT	↓	↓		↓	(--)	↑	
Electrodermal							
SCR	↑	↑		↑	↑		
nSRR	↑	↑		↑	↑	↑	↑
SCL	↑	↑	↑	↑	↑	↑--	(--)
Respiratory							
RR	↑	↑		↑	↑	↑	↑
Ti	↓	↓		↓--	↓	↓	
Te	↓	↓		↓		↓	
Pi	↑			↑		↓	
Ti/Ttot				↑	↓		
Vt	↑↓	↓		↑↓	↑↓	↑↓	
Vi/Ti						↑	
Electroencephalography							
PSD (α wave)	↑	↑		↓	↑	↑	↑
PSD (β wave)	↓				↑		
PSD (γ wave)				↓	↑	↑	↑
DE (average)	↑	(--)		↓		↑	↑
DASM (average)	(--)			↑	↓	↓	↓
RASM (average)	↑			↑		↓	

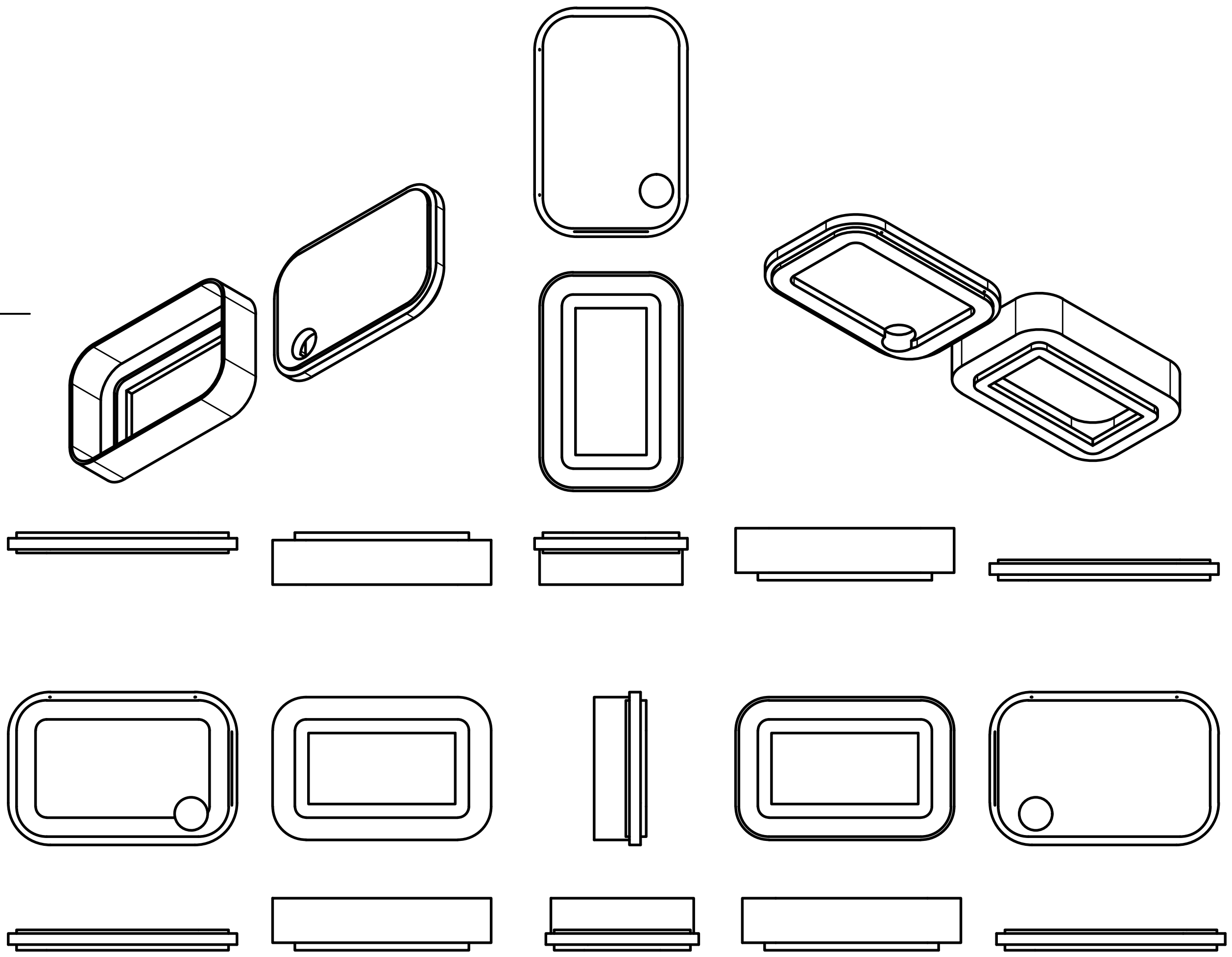
Note.* Arrows indicate increased (↑), decreased (↓), or no change in activation from baseline (−), or both increases and decreases in different studies (↑↓).

Appendix C: Technical drawing physicalisation



Dept.	Technical reference	Created by Dennis Peeters 04/04/2022	Approved by
		Document type	Document status Complete
		Title Main physicalisation Scale: 1:10	DWG No.
			Rev. / Date of issue / Sheet 1/1

Appendix D: Technical drawing wearable add-on



Dept.	Technical reference	Created by Dennis Peeters 10/04/2022	Approved by
		Document type Technical design	Document status Complete
		Title WATCH design 1 scale 1:2	DWG No.

Rev.	Date of issue	Sheet
		1/1

Appendix E: Full program for Wearable

```
/*
  Code written by Dennis Peeters, May 2022
  Based on an example provided by Rui Santos, details at https://randomnerdtutorials.com
*/
// Import all the required libraries
#ifdef ESP32
#include <WiFi.h>
#else
#include <ESP8266WiFi.h>
#endif
#include <Wire.h>

//set all the variables to the right startvalues
// Replace with your SSID and Password
const char* ssid     = "REPLACE";
const char* password = "REPLACE";

//Webserver related information
int    HTTP_PORT   = 80;
String HTTP_METHOD = "GET";
char   HOST_NAME[] = "dennispeeters.nl";
String PATH_NAME   = "/ph.php";

// Deep sleep variables to save power
uint64_t uS_TO_S_FACTOR = 1000000;  // Conversion factor for micro seconds to seconds
uint64_t TIME_TO_SLEEP = 20; //20 second deepsleep timer

//Setup I2C communication lines for HR sensor
#include <Wire.h>
#define I2C_SDA 21
#define I2C_SCL 18

//Setup the status LED
#include <Adafruit_NeoPixel.h>
#define PIN        5
#define NUMPIXELS 1
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
#define DELAYVAL 500

void setup() {
  //setup communication channels and LED
  Serial.begin(115200);
  Wire.begin(I2C_SDA, I2C_SCL);
  pixels.begin();
  delay(2000);  //delay to prevent LED from crashing
  initWifi();   // Establish a Wi-Fi connection with your router
  dataPass();   // Read sensors and make an HTTP request to the server
  delay(20);
  pixels.clear(); // Set all pixel colors to 'off'
  //Blink the color blue to show sending is complete
  for (int i = 0; i < NUMPIXELS; i++) { // For each pixel...
    pixels.setPixelColor(i, pixels.Color(0, 0, 150));
    pixels.show();
    delay(DELAYVAL); // Pause before next pass through loop
  }
  //Read the battery voltage and change colour of the LED accordingly
  batterymeasurement();

#ifdef ESP32
  // enable timer deep sleep
  esp_sleep_enable_timer_wakeup(TIME_TO_SLEEP * uS_TO_S_FACTOR);
  Serial.println("Going to sleep now");
  // start deep sleep
  esp_deep_sleep_start();
#else
  // Deep sleep mode
  Serial.println("Going to sleep now");
  ESP.deepSleep(TIME_TO_SLEEP * uS_TO_S_FACTOR);
#endif
}

void loop() {
  // sleeping so wont get here
}
// Establish a Wi-Fi connection with your router
void initWifi() {
  //Try to connect to the network
  Serial.print("Connecting to: ");
  Serial.print(ssid);
  WiFi.begin(ssid, password);
  int timeout = 10 * 4; // 10 seconds
  //Try loop to keep trying to connect to the internet
  while (WiFi.status() != WL_CONNECTED  && (timeout-- > 0)) {
    delay(250);
    Serial.print(".");
  }
  Serial.println("");

  //Print connection status
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("Failed to connect, going back to sleep");
  }
  Serial.print("WiFi connected in: ");
  Serial.print(millis());
  Serial.print(", IP address: ");
  Serial.println(WiFi.localIP());
}

// Read sensors and make an HTTP request to the server
void dataPass() {
  WiFiClient client;   //set the wifi system
  int retries = 5; //set the amount of retries to upload the data
  //Keep trying to connect to the server
  while (!!!client.connect(HOST_NAME, 80) && (retries-- > 0)) {
    Serial.print(".");
  }

  // set all for the reading of the sensors
  int HR = 0;
  Wire.requestFrom(0x50, 1);    // request 1 bytes from slave device
  Serial.print("heart rate sensor:");
  while (Wire.available()) {        // slave may send less than requested
    unsigned char c = Wire.read();   // receive heart rate value (a byte)
    Serial.println(c, DEC);         // print heart rate value in serial line for development purposes
    HR = c; //set the HR variable to the received heart rate

  }
  int GSR = analogRead(3); //analog reading of the GSR signal
  int remapGSR = map(GSR, 0, 8200, 0, 1023); // remap the signal to an different level
  Serial.print("GSR value:"); //print value in the command line for development purposes
  Serial.println(remapGSR);

  // Pass the data into a string
  String jsonObject = String("?hr=") + HR + String("&gsr=") + remapGSR;

  if (client.connect(HOST_NAME, HTTP_PORT)) {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    client.println(HTTP_METHOD + " " + PATH_NAME + jsonObject + " HTTP/1.1");  // send
HTTP header
    client.println("Host: " + String(HOST_NAME));
    client.println("Connection: close");
    client.println(); // end HTTP header
    // the server's disconnected, stop the client:
    client.stop();
    Serial.println();
    Serial.println("disconnected");
  }
  else {// if not connected:
    Serial.println("connection failed");
  }
  int timeout = 5 * 10; // 5 seconds
  while (!!!client.available() && (timeout-- > 0)) {
    delay(100);
  }
}

void batterymeasurement() {
  //battery voltage reading
  int nVoltageRaw = analogRead(14); //read the raw voltage from the voltage divider
  int remapVoltageRaw = map(nVoltageRaw, 0, 8200, 0, 1023); //remap the voltage reading
  float fVoltage = (float)remapVoltageRaw * 0.0045; //calculate the actual voltage of the
battery
  Serial.println(fVoltage);
  float fVoltageMatrix[22][2] = {
    {4.2,  100},
    {4.15, 95},
    {4.11, 90},
    {4.08, 85},
    {4.02, 80},
    {3.98, 75},
    {3.95, 70},
    {3.91, 65},
    {3.87, 60},
    {3.85, 55},
    {3.84, 50},
    {3.82, 45},
    {3.80, 40},
    {3.79, 35},
    {3.77, 30},
    {3.75, 25},
    {3.73, 20},
    {3.71, 15},
    {3.69, 10},
    {3.61, 5},
    {3.27, 0},
    {0, 0}
  };

  int i, perc, redvalue, greenvalue;

  perc = 5;
  redvalue = 0;
  greenvalue = 0;
```

```
 //turn the voltage into a percentage reading, where 3.3V is 0 percent and 4.2V is 100%, the
range of a LiPo battery cells safe operation
  for (i = 20; i >= 0; i--) {
    if (fVoltageMatrix[i][0] >= fVoltage) {
      perc = fVoltageMatrix[i + 1][1];
      break;
    }
  }
  //map the percentage to color, where full green is 100% and full red is 0%
  redvalue = map(perc, 0, 100, 150, 0);
  greenvalue = map(perc, 0, 100, 0, 150);
  pixels.clear(); // Set all pixel colors to 'off'

  //set the pixels to the color that was determined to be the battery status
  for (int i = 0; i < NUMPIXELS; i++) { // For each pixel...
    pixels.setPixelColor(i, pixels.Color(redvalue, greenvalue, 0));
    pixels.show();
    delay(DELAYVAL); // Pause before next pass through loop
  }
  delay(20);
}
```

Appendix F: full program physicalization

```
# Code written by Dennis Peeters, may 2022
# based on background research done during the GP and examples provided by
a TCS student

# Import libraries
import datetime
import RPi.GPIO as GPIO
import board
import time
import neopixel

# set the hardware pins
LED_amount = 90
GPIO.setmode(GPIO.BCM)
pixels = neopixel.NeoPixel(board.D21, LED_amount, brightness = 0.6,
auto_write = True, pixel_order = neopixel.GRB)
from turtle import pd

# import libraries regarding data retrieval from the serverl
import mysql.connector
import pytz as pytz

# set the algorithm calibration values
HR_maxvalue = 130
GSR_maxvalue = 250
HR_correctionvalue = 98
GSR_correctionvalue = 130
HR_Valence_devcoefficient = 4
GSR_Valence_devcoefficient = 30
HR_Valence_devcoefficient2 = 35
GSR_Valence_devcoefficient2 = 120
previousHR = 76
previousGSR = 150
previousHR1 = 76
previousGSR1 = 150

# Create a LED color buffer
LED_buffer = []
for x in range(0, (3 * LED_amount)):
    LED_buffer.append(0)
print(LED_buffer)

# Set the hardware pins of the RPI to the right mode
GPIO.setup(14, GPIO.IN)
GPIO.setup(17, GPIO.IN)
GPIO.setup(26, GPIO.OUT)

# Read switch states and connect them to a variable
state2 = GPIO.input(17)
state1 = GPIO.input(14)
print(state1)
print(state2)

# Setup of the relay to control the warm-white light
relay = GPIO.output(26, 1)

# Function to check if the switch is in the off-mode, if it is, turn the
warm-white LED's on and exit the code
def light_ring():
    if state2 == 1:
        GPIO.output(26, 1)
        pixels.fill((0,0,0))
        exit()
    elif state2 == 0:
        GPIO.output(26, 0)

# Run the previously discribed function
light_ring()

# Change the mode selector in a boolean
if state1 == 1:
    state = True
else:
    state = False

# Connect to the database, replace with the right login information
mydb = mysql.connector.connect(
    host="dennispeeters.nl",
    user="*********",
    password="********",
    database="*********"
)

# further setup of the database and timestamps
mycursor = mydb.cursor()
timezone = pytz.timezone("UTC")
now = datetime.datetime.now().replace(microsecond=0)
one_minute_ago = now - datetime.timedelta(minutes=1)

# Code to set the variablefile to hold values which should be kept after
finishing the code
variablefile = open("storedvariables.txt", "r+")
if variablefile.read() == "": # Check if the file is empty, if so, fill
with a number of base values
    if int(now.minute) > 30:
        sb = int(((now.minute) - 30) * (LED_amount / 60))
    else:
        sb = int(((now.minute) + 30) * (LED_amount / 60))
        print(sb)
        variablefile.write(str(state) + "," + str(sb) + "," + str(now) +
"," + str(sb) + ",70" + ",200")
    for x in range(0, 3 * LED_amount):
        variablefile.write("," + str(LED_buffer[x]))
        variablefile.close()
    if int(now.minute) > 30:
        LED_position = int(((now.minute) - 30) * (LED_amount / 60))
    else:
        LED_position = int(((now.minute) + 30) * (LED_amount / 60))
        print(LED_position)
        variablefile2 = open("storedvariables.txt", "r")
        print(variablefile2.read())
        variablefile2.close()
    for y in range(0, LED_amount):
        pixels[y] = (100, 100, 100)
        time.sleep(2)
    for y in range(0, LED_amount):
        pixels[y] = (0, 0, 0)
        time.sleep(1)
else: # Read the values present in the file and connect them to variables
to be used in the code
    variablefile = open("storedvariables.txt", "r")
    temp_variable = variablefile.read().split(",")
    file_state = temp_variable[0]
    print(file_state)
    last_LED = temp_variable[1]
    startpoint = temp_variable[2]
    startled = temp_variable[3]
    previousHR = int(temp_variable[4])
    previousGSR = int(temp_variable[5])
    print(len(temp_variable[6]))
    for x in range(0, (3 * LED_amount)):
        LED_buffer[x] = int(temp_variable[(x + 6)])
    print(LED_buffer)
    print(last_LED)
    print(previousHR)
    print(previousGSR)
    variablefile.close()

# function used to fill the ring with new data when switching modes
def ring_fill(variable, startpoint, current_time):
    LED_position = startled
    # Pull the data from the database
    mycursor.execute(f"SELECT * FROM data WHERE timestamp > '{startpoint}'
AND timestamp < '{current_time.strftime('%Y-%m-%d %H:%M:%S')}'") # Read
data from the database
    myresult = mycursor.fetchall()
    previousHR = 76
    previousGSR =150
    for x in myresult: # For each of the results from the database
        if variable == 0: # if the mode is set to arousal mode
            previous_LED_position = int(LED_position)
            print(f"Date: {x[0]}, HR: {x[1]}, GSR: {x[2]}") # print data
for development purposes
            # next if-statement used to match timestamp to right LED
position
            if x[0].minute > 30:
                LED_position = ((x[0].minute) - 30) * (LED_amount / 60)
            else:
                LED_position = ((x[0].minute) + 30) * (LED_amount / 60)
            LED_color = arousal_to_light(x[1], x[2], previousHR,
previousGSR) # request data to be processed by the arousal algorithm to
determine colour
            # set new previous values for the next round of algorithm
calculations
            previousHR = x[1]
            previousGSR = x[2]
            # Counter set to fill the right amount of LED's due to mismatch
in amount of LED's and 60 minutes
            counter = int(LED_position) - previous_LED_position
            if LED_color == None: # failsafe to prevent issues with
algorithm function
                LED_color = (0, 0, 0)
            # fill the LED's with the right colour
            for y in range(0, counter):
                for z in range(0, 3):
                    LED_buffer[(3 * (previous_LED_position + y)) + z] =
LED_color[z]
                pixels[previous_LED_position + y] = LED_color
                if (int(previous_LED_position + y + 1)) < LED_amount:
                    pixels[int(previous_LED_position + y + 1)] = (100, 100,
100)
                time.sleep(0.1)
        elif variable == 1: # if the mode is set to valence mode
            previous_LED_position = int(LED_position)
            print(f"Date: {x[0]}, HR: {x[1]}, GSR: {x[2]}") # print data
for development purposes
            # next if-statement used to match timestamp to right LED
position
            if x[0].minute > 30:
                LED_position = ((x[0].minute) - 30) * (LED_amount / 60)
            else:
                LED_position = ((x[0].minute) + 30) * (LED_amount / 60)
            LED_color = valence_to_light(x[1], x[2], previousHR,
previousGSR)# request data to be processed by the valence algorithm to
determine colour
            # set new previous values for the next round of algorithm
calculations
            previousHR = x[1]
            previousGSR = x[2]
            # Counter set to fill the right amount of LED's due to mismatch
in amount of LED's and 60 minutes
            counter = int(LED_position) - previous_LED_position
            # fill the LED's with the right colour
            for y in range(0, counter):
                if LED_color == None: # failsafe to prevent issues with
algorithm function
                    LED_color = (0, 0, 0)
                for z in range(0, 3):
                    LED_buffer[(3 * (previous_LED_position + y)) + z] =
LED_color[z]

                pixels[previous_LED_position + y] = LED_color
                if (int(previous_LED_position + y + 1)) < LED_amount:
                    pixels[int(previous_LED_position + y + 1)] = (100, 100,
100)
                time.sleep(0.2)

# Function to turn the raw data from the server into the right colour based
for the arousal mode
def arousal_to_light(heart_rate, gsr, x, y):
    # set the imported values to variables used in the function
    previousHR = x
    previousGSR = y
    if (heart_rate - previousHR) <= 6 and (heart_rate - previousHR) >= 0: #
check the raw values against a certain case
        if (gsr - previousGSR) >= -30 and (gsr - previousGSR) <= 0:
            if heart_rate >= 88 and gsr <= 100:
                # calculation to convert the raw data into the colour using
values to adjust to the case
                redvalue = (((heart_rate * (255 / HR_maxvalue)) + (255 -
(gsr * (255 / GSR_maxvalue)))) / 2)
                # calculation to convert the raw data into the colour using
values to adjust to the case
                greenvalue = (((255 - (heart_rate * (255 / HR_maxvalue))) +
(gsr * (255 / GSR_maxvalue))) / 2)
                # if statements to prevent out of bound variables, and
adapt accordingly
                if redvalue > 255:
                    greenvalue = greenvalue - (redvalue - 255)
                    redvalue = 255
                if greenvalue < 0:
                    greenvalue = 0
                elif redvalue < 0:
                    redvalue = 0
                # return the colour values
                return (int(redvalue), int(greenvalue), 0)
    elif heart_rate >= 50 and heart_rate < 88 and gsr <= 400 and gsr > 10:
# check the raw values against a certain case
        # calculation to convert the raw data into the colour using values
to adjust to the case
        redvalue = ((((heart_rate * (255 / HR_maxvalue)) -
HR_correctionvalue) + (
            (255 - (gsr * (255 / GSR_maxvalue)) - GSR_correctionvalue))) /
2)
        # calculation to convert the raw data into the colour using values
to adjust to the case
        greenvalue = (((255 - (heart_rate * (255 / HR_maxvalue))) + (gsr *
(255 / GSR_maxvalue))) / 2)
        # if statements to prevent out of bound variables, and adapt
accordingly
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        # return the colour values
        return (int(redvalue), int(greenvalue), 0)
    elif heart_rate >= 88 and gsr <= 100: # check the raw values against a
certain case
        # calculation to convert the raw data into the colour using values
to adjust to the case
        redvalue = (((heart_rate * (255 / HR_maxvalue)) + (0.5 * (255 -
(gsr * (255 / GSR_maxvalue))))) / 1.5)
        # calculation to convert the raw data into the colour using values
to adjust to the case
        greenvalue = (((255 - (heart_rate * (255 / HR_maxvalue))) + (0.5 *
(gsr * (255 / GSR_maxvalue)))) / 1.5)
        # if statements to prevent out of bound variables, and adapt
accordingly
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        # return the colour values
        return (int(redvalue), int(greenvalue), 0)
    elif heart_rate >= 50 and heart_rate < 88 and gsr <= 400 and gsr > 100:
# check the raw values against a certain case
        # calculation to convert the raw data into the colour using values
to adjust to the case
        redvalue = ((((heart_rate * (255 / HR_maxvalue) -
HR_correctionvalue)) + (
                    0.5 * ((255 - (gsr * (255 / GSR_maxvalue))) -
GSR_correctionvalue))) / 1.5)
        # calculation to convert the raw data into the colour using values
to adjust to the case
        greenvalue = (((255 - (heart_rate * (255 / HR_maxvalue))) + (0.5 *
(gsr * (255 / GSR_maxvalue)))) / 1.5)
        # if statements to prevent out of bound variables, and adapt
```

```
accordingly
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        # return the colour values
        return (int(redvalue), int(greenvalue), 0)
    elif (gsr - previousGSR) >= -30 and (gsr - previousGSR) <= 0: # check
the raw values against a certain case
        if heart_rate >= 88 and gsr <= 100:
            # calculation to convert the raw data into the colour using
values to adjust to the case
            redvalue = (((0.5 * (heart_rate * (255 / HR_maxvalue))) + (255
- (gsr * (255 / GSR_maxvalue)))) / 1.5)
            # calculation to convert the raw data into the colour using
values to adjust to the case
            greenvalue = (((0.5 * (255 - (heart_rate * (255 /
HR_maxvalue)))) + (gsr * (255 / GSR_maxvalue))) / 1.5)
            # if statements to prevent out of bound variables, and adapt
accordingly
            if redvalue > 255:
                greenvalue = greenvalue - (redvalue - 255)
                redvalue = 255
            if greenvalue < 0:
                greenvalue = 0
            elif redvalue < 0:
                redvalue = 0
            # return the colour values
            return (int(redvalue), int(greenvalue), 0)
    elif heart_rate >= 50 and heart_rate < 88 and gsr <= 400 and gsr > 100:
# check the raw values against a certain case
        # calculation to convert the raw data into the colour using values
to adjust to the case
        redvalue = ((0.5 * (((heart_rate * (255 / HR_maxvalue)) -
HR_correctionvalue)) + (
                    255 - ((gsr * (255 / GSR_maxvalue)) -
GSR_correctionvalue))) / 1.5)
        # calculation to convert the raw data into the colour using values
to adjust to the case
        greenvalue = (((0.5 * (255 - (heart_rate * (255 / HR_maxvalue)))) +
(gsr * (255 / GSR_maxvalue))) / 1.5)
        # if statements to prevent out of bound variables, and adapt
accordingly
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        # return the colour values
        return (int(redvalue), int(greenvalue), 0)

    elif heart_rate >= 88 and gsr <= 100: # check the raw values against a
certain case
        # calculation to convert the raw data into the colour using values
to adjust to the case
        redvalue = (((heart_rate * (255 / HR_maxvalue)) + (0.5 * (255 -
(gsr * (255 / GSR_maxvalue))))) / 1.5)
        # calculation to convert the raw data into the colour using values
to adjust to the case
        greenvalue = (((255 - (heart_rate * (255 / HR_maxvalue))) + (0.5 *
(gsr * (255 / GSR_maxvalue)))) / 1.5)
        # if statements to prevent out of bound variables, and adapt
accordingly
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        # return the colour values
        return (int(redvalue), int(greenvalue), 0)
    elif heart_rate >= 50 and heart_rate < 88 and gsr <= 400 and gsr > 100:
# check the raw values against a certain case
        # calculation to convert the raw data into the colour using values
to adjust to the case
        redvalue = (((heart_rate * (255 / HR_maxvalue) -
HR_correctionvalue)) + (
                    0.5 * ((255 - (gsr * (255 / GSR_maxvalue))) -
GSR_correctionvalue)) / 1.5)
        # calculation to convert the raw data into the colour using values
to adjust to the case
        greenvalue = ((255 - (heart_rate * (255 / HR_maxvalue))) + (0.5 *
(gsr * (255 / GSR_maxvalue))) / 1.5)
        # if statements to prevent out of bound variables, and adapt
accordingly
        if redvalue > 255:
            greenvalue = greenvalue - (redvalue - 255)
            redvalue = 255
        if greenvalue < 0:
            greenvalue = 0
        elif redvalue < 0:
            redvalue = 0
        # return the colour values
        return (int(redvalue), int(greenvalue), 0)
    else: # if the value cannot be placed into one of these cases, the LED
will stay off, by using color black
        return (0, 0, 0)


# Function to turn the raw data from the server into the right colour based
for the valence mode
def valence_to_light(heart_rate, gsr, x, y):
    # set the imported values to variables used in the function
    previousHR = x
    previousGSR = y
    HR_Difference = heart_rate - previousHR
    GSR_Difference = gsr - previousGSR
    if (HR_Valence_devcoefficient >= (heart_rate - previousHR) >= (0 -
HR_Valence_devcoefficient)) and (
            (0 - GSR_Valence_devcoefficient) <= (gsr - previousGSR) <=
GSR_Valence_devcoefficient):  # check the raw values against a certain case
        if heart_rate >= 50 and gsr <= 400: # Check if the values are in
the normal range of heart rate and GSR, to filter reading errors
            # calculation to convert the raw data into the colour using
values to adjust to the case
            bluevalue = ((255 - ((abs(HR_Difference) * (130 /
HR_Valence_devcoefficient))) + (
                    255 - (abs(GSR_Difference) * (150 /
GSR_Valence_devcoefficient)))) / 2)
            # calculation to convert the raw data into the colour using
values to adjust to the case
            yellowvalue = (((abs(HR_Difference) * (130 /
HR_Valence_devcoefficient)) + (
                    abs(GSR_Difference) * (150 /
GSR_Valence_devcoefficient))) / 2)
            # if statements to prevent out of bound variables, and adapt
accordingly
            if bluevalue > 255:
                yellowvalue = yellowvalue - bluevalue
                bluevalue = 255
            elif yellowvalue < 0:
                yellowvalue = 0
            elif yellowvalue > 255:
                yellowvalue = 255
            # return the colour values
            return (int(yellowvalue), int(yellowvalue), int(bluevalue))
        else: # if the value cannot be placed into one of these cases, the
LED will stay off, by using color black
            return (0, 0, 0)
    elif ((heart_rate - previousHR) > HR_Valence_devcoefficient or
(heart_rate - previousHR) < (
            0 - HR_Valence_devcoefficient)) and (
            (0 - GSR_Valence_devcoefficient) > (gsr - previousGSR) or (gsr
- previousGSR) > GSR_Valence_devcoefficient):  # check the raw values
against a certain case
        if heart_rate >= 50 and gsr <= 400: # Check if the values are in
the normal range of heart rate and GSR, to filter reading errors
            # calculation to convert the raw data into the colour using
values to adjust to the case
            bluevalue = (((abs(HR_Difference) * (130 /
HR_Valence_devcoefficient2)) + (
                    abs(GSR_Difference) * (150 /
GSR_Valence_devcoefficient2))) / 2)
            # calculation to convert the raw data into the colour using
values to adjust to the case
            yellowvalue = ((255 - ((abs(HR_Difference) * (130 /
HR_Valence_devcoefficient2))) + (
                    255 - (abs(GSR_Difference) * (150 /
GSR_Valence_devcoefficient2)))) / 2)
            # if statements to prevent out of bound variables, and adapt
accordingly
            if bluevalue > 255:
                yellowvalue = yellowvalue - bluevalue
                bluevalue = 255
            elif yellowvalue < 0:
                yellowvalue = 0
            elif yellowvalue > 255:
                yellowvalue = 255
            # return the colour values
            return (int(yellowvalue), int(yellowvalue), int(bluevalue))
        else: # if the value cannot be placed into one of these cases, the
LED will stay off, by using color black
            return (0, 0, 0)

    else:
        if heart_rate >= 50 and gsr <= 400:  # check the raw values against
a certain case
```

```
            if HR_Difference > HR_Valence_devcoefficient:
                # calculation to convert the raw data into the colour using
values to adjust to the case
                bluevalue = (((0.5 * (abs(HR_Difference) * (255 /
HR_Valence_devcoefficient2))) + (
                        1.5 * (abs(GSR_Difference)) * (255 /
GSR_Valence_devcoefficient))) / 2)
                # calculation to convert the raw data into the colour using
values to adjust to the case
                yellowvalue = (((255 - (0.5 * (abs(HR_Difference) * (255 /
HR_Valence_devcoefficient2)))) + (
                        1.5 * (abs(GSR_Difference)) * (255 /
GSR_Valence_devcoefficient))) / 2)
                # if statements to prevent out of bound variables, and
adapt accordingly
                if bluevalue > 255:
                    yellowvalue = yellowvalue - bluevalue
                    bluevalue = 255
                elif yellowvalue < 0:
                    yellowvalue = 0
                elif yellowvalue > 255:
                    yellowvalue = 255
                # return the colour values
                return (int(yellowvalue), int(yellowvalue), int(bluevalue))
            elif GSR_Difference > GSR_Valence_devcoefficient:
                # calculation to convert the raw data into the colour using
values to adjust to the case
                bluevalue = (((1.5 * (abs(HR_Difference) * (255 /
HR_Valence_devcoefficient2))) + (
                        0.5 * (abs(GSR_Difference)) * (255 /
GSR_Valence_devcoefficient))) / 2)
                # calculation to convert the raw data into the colour using
values to adjust to the case
                yellowvalue = (((255 - (1.5 * (abs(HR_Difference) * (255 /
HR_Valence_devcoefficient2)))) + (
                        0.5 * (abs(GSR_Difference)) * (255 /
GSR_Valence_devcoefficient))) / 2)
                # if statements to prevent out of bound variables, and
adapt accordingly
                if bluevalue > 255:
                    yellowvalue = yellowvalue - bluevalue
                    bluevalue = 255
                elif yellowvalue < 0:
                    yellowvalue = 0
                elif yellowvalue > 255:
                    yellowvalue = 255
                # return the colour values
                return (int(yellowvalue), int(yellowvalue), int(bluevalue))
            else: # if the value cannot be placed into one of these cases,
the LED will stay off, by using color black
                return (0, 0, 0)
        else: # if the value cannot be placed into one of these cases, the
LED will stay off, by using color black
            return (0, 0, 0)

# Check if the mode switch has been changed compared to the previously
recorded state
if file_state != str(state):
    if state is True:
        ring_fill(0, startpoint, now)
    elif state is False:
        ring_fill(1, startpoint, now)
else: # if the mode has not changed(This code is, as explained in the
report, similar to the ring_fill function)
    LED_position = startled
    # Pull the data from the database
    mycursor.execute(f"SELECT * FROM data WHERE timestamp > '{startpoint}'
AND timestamp < '{now.strftime('%Y-%m-%d %H:%M:%S')}'")
    myresult = mycursor.fetchall()
    previousHR = 76
    previousGSR =150
    for x in myresult:  # For each of the results from the database
            previous_LED_position = int(LED_position)
            print(f"Date: {x[0]}, HR: {x[1]}, GSR: {x[2]}") # print data
for development purposes
            # next if-statement used to match timestamp to right LED
position
            if x[0].minute > 30:
                LED_position = ((x[0].minute) - 30) * (LED_amount / 60)
            else:
                LED_position = ((x[0].minute) + 30) * (LED_amount / 60)
            #Check the mode in order to select the right algorithm
            if state is True:
                LED_color = arousal_to_light(x[1], x[2], previousHR,
previousGSR) # request data to be processed by the arousal algorithm to
determine colour
            elif state is False:
                LED_color = valence_to_light(x[1], x[2], previousHR,
previousGSR) # request data to be processed by the valence algorithm to
determine colour
            # set new previous values for the next round of algorithm
calculations
            previousHR = x[1]
            previousGSR = x[2]
            # Counter set to fill the right amount of LED's due to mismatch
in amount of LED's and 60 minutes
            counter = int(LED_position) - previous_LED_position
            if LED_color == None: # failsafe to prevent issues with
algorithm function
                LED_color = (0, 0, 0)
            # fill the LED's with the right colour
            for y in range(0, counter):
                print(previous_LED_position + y)
                for z in range(0, 3):
                    LED_buffer[(3 * (previous_LED_position + y)) + z] =
LED_color[z]
                print(LED_color)
                print(LED_buffer)
                pixels[previous_LED_position + y] = LED_color
                if (int(previous_LED_position + y + 1)) < LED_amount:
                    pixels[int(previous_LED_position + y + 1)] = (100, 100,
100)
                time.sleep(0.1)

# Open the file that stores variables after the code has finished
variablefile = open("storedvariables.txt", "r+")
variablefile.truncate(0) # Wipe the file
variablefile.close()
# re-open the file that stores variables after the code had finished
variablefile = open("storedvariables.txt", "r+")
# Fill the code with the right set of updated variables
if int(now.minute) > 30:
    sb = int(((now.minute) - 30) * (LED_amount / 60))
else:
    sb = int(((now.minute) + 30) * (LED_amount / 60))
variablefile.write(str(state) + "," + str(sb) + "," + startpoint + "," +
startled + "," + str(previousHR) + "," + str(previousGSR))
for x in range(0, 3 * LED_amount):
    variablefile.write("," + str(LED_buffer[x]))
print("update complete") # Print statement for development purposes
variablefile.close()
```

Appendix G: Reading material user test explanation physicalization

Instructions Physicalisation

Controls

When looking at the device, you will see two switches. One is labelled with on-off. This is there to turn on the display of the data. In the off position, it will turn the physicalisation in a general light. The second switch is labelled with Arousal and Valence. This is to switch between the two different variables that can be displayed

! IMPORTANT!: The system will only update the lights and check the position of the switches, so if it is set to valence or arousal mode for one and set to on or off for the other, ones every minute, so it can take up to 1 minute before the physicalisation will update its lights if you put one of the switches in a different position.

Reading the physicalisation.

The round shape of the physicalisation is chosen with a specific reason. It has been chosen to resemble a clock, making that data can be mapped to certain minutes. Figure 1 below is an illustration of how data can be mapped.

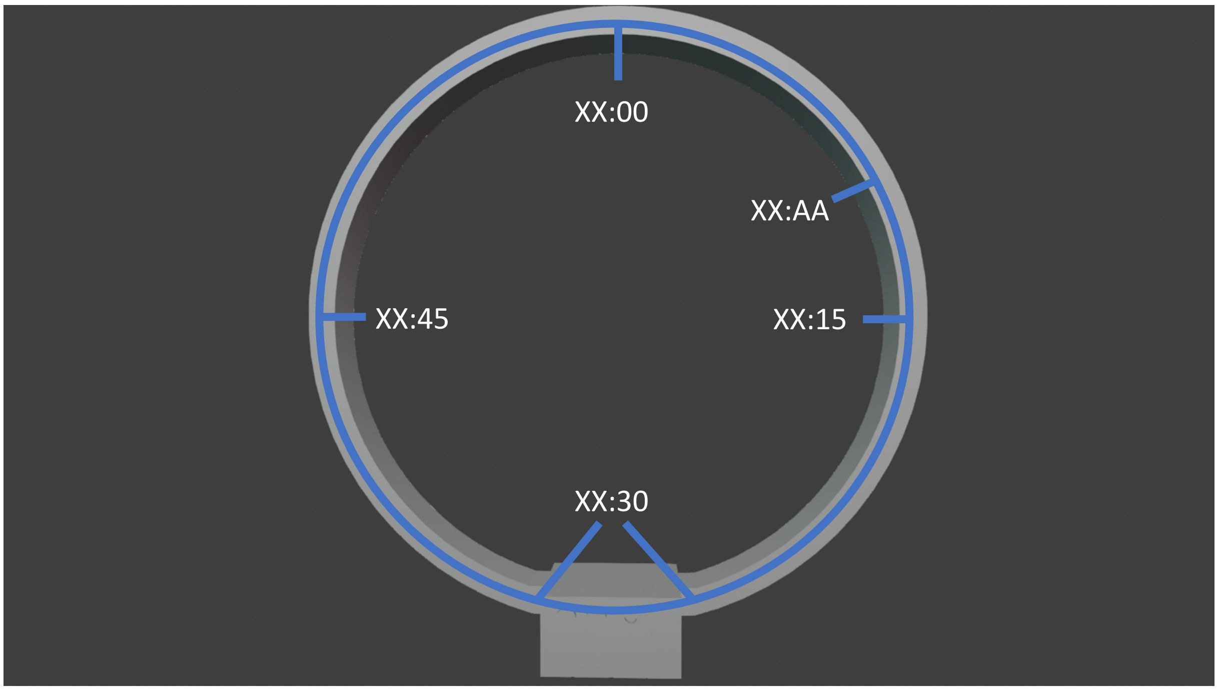


Figure 4

To give a more concrete example of that. When the LED’s at point XX:AA are turned on , it means that at that point, the data is from the timestamp XX:AA, where XX is the current hour, and AA is 10 past, so at XX:10 the data is …. The XX in figure 1 is the current hour as described earlier.

The meaning of the data

The physicalization will display the arousal and valance level that a person is experiencing using colour of the LED lights. The coding scheme is represented in Figure 2 and 3.

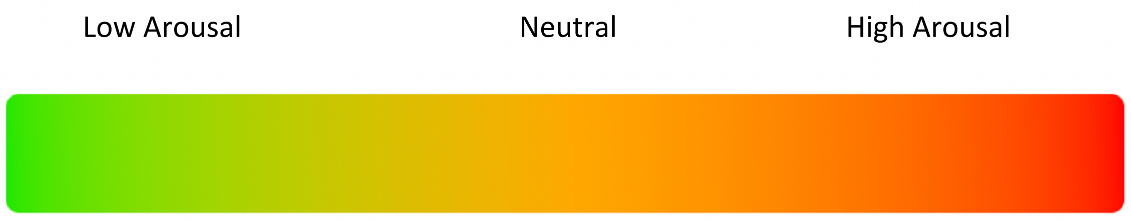


Figure 5

Looking at figure 2, when an LED is orange, it means the arousal levels of the person are fairly neutral. Green is an indication of low arousal levels and red an indication of high arousal levels When switching the physicalisation in Valence mode, the colours of the system will change to a new spectrum, from blue to yellow, as can be seen in figure 3. Valence is a measure between positive a negative, where blue means the value is a positive valence, while a negative valence is indicated by a more yellow colour. The colour in the middle, white, means a neutral state.

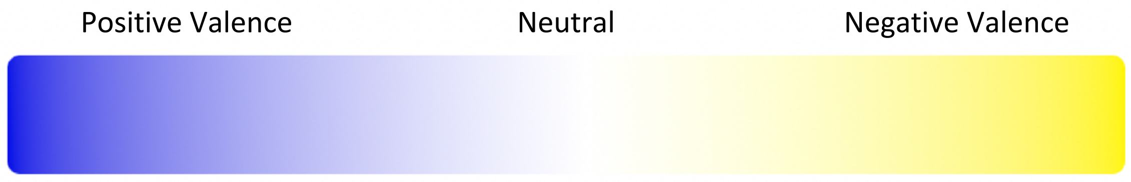


Figure 6

Both Valence and arousal could then be used to get a rough indication of the emotion someone is experiencing. This can be done by using the representation that can be seen in figure 4.

! IMPORTANT!: When an LED is not turned on, the system has detected a false reading which cannot be mapped to a colour.

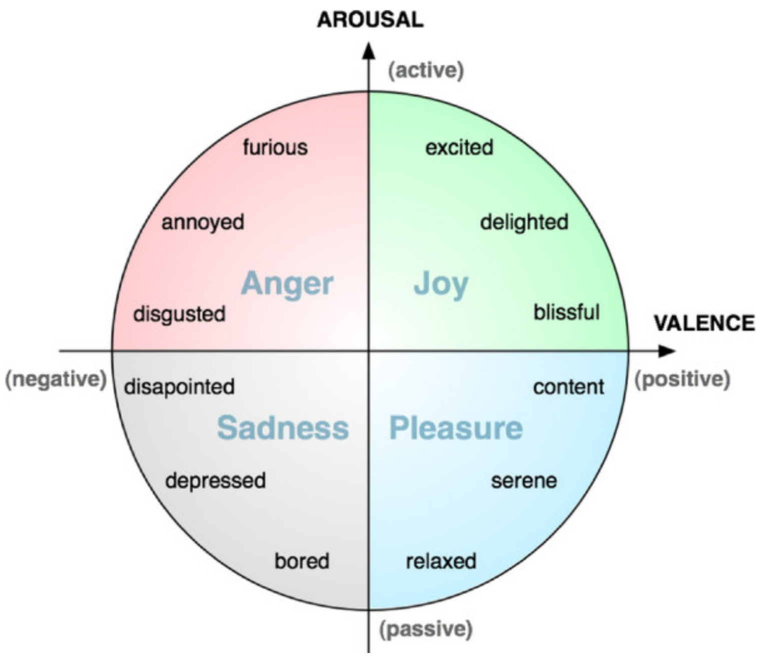


Figure 7

Taking notes
Your task now is to observe the physicalisation in front of you. Please write down the timestamp you think a certain datapoint has been added, based on the current hour and the minutes passed (according to the explanation above). Furthermore, write down the state of valence(positive, negative, neutral) or arousal(high, low, neutral). Finally, try to write down the emotion you think can be connected to it.

Troubleshooting
You might experience issues during the tests with the system. For example, it stops loading further. If this happens, please switch it to the other measure-mode (Valence/Arousal) from the one you are currently in and wait 1 minute. The system will automatically retry to process the data.

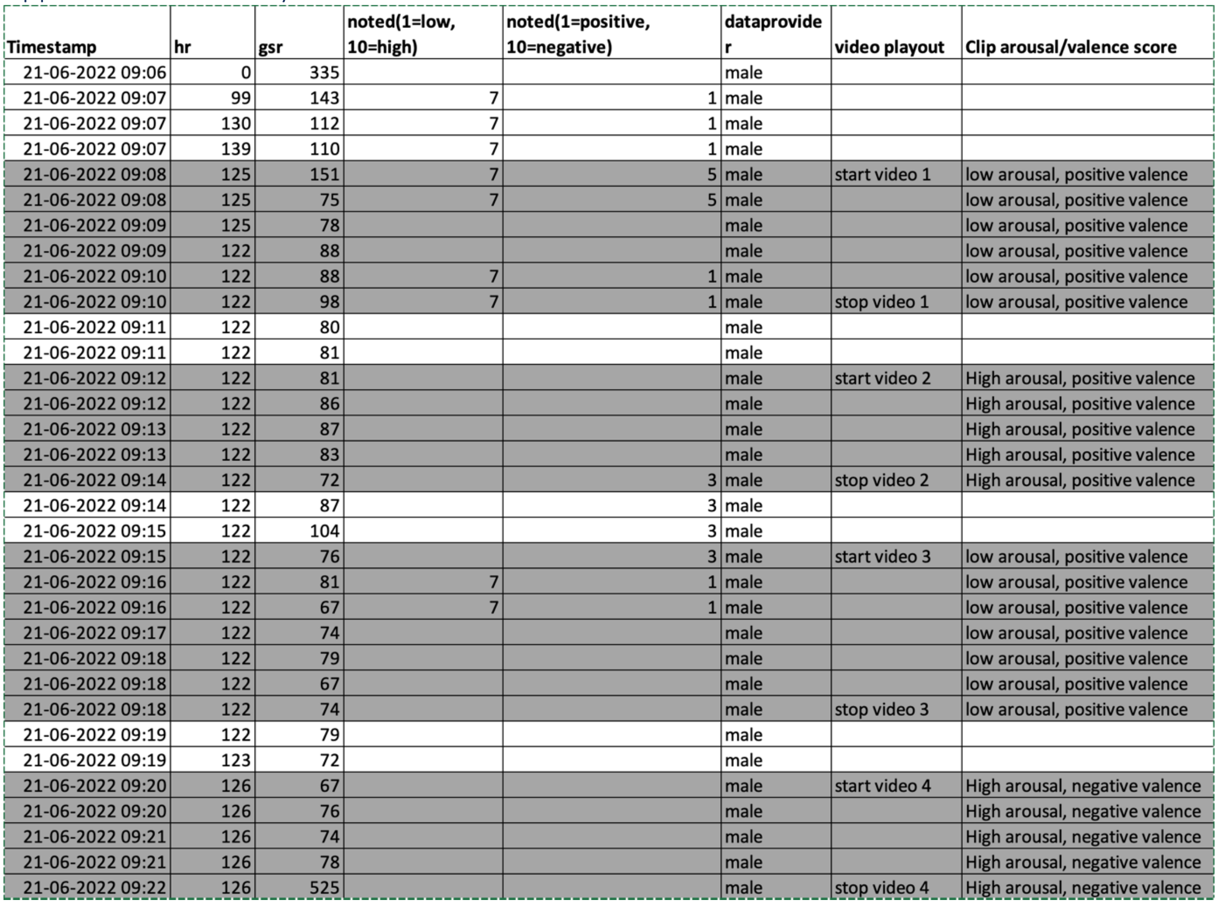
Appendix H: USE questionnaire for user study

USE Questionnaire

USEFULNESS		**1**	**2**	**3**	**4**	**5**	**6**	**7**	
It is useful.	strongly disagree								strongly agree
It does everything I would expect it to do.	strongly disagree								strongly agree
EASE OF USE		**1**	**2**	**3**	**4**	**5**	**6**	**7**	
It is easy to use.	strongly disagree								strongly agree
It is simple to use.	strongly disagree								strongly agree
It is user friendly.	strongly disagree								strongly agree
It requires the fewest steps possible to accomplish what I want to do with it.	strongly disagree								strongly agree
It is flexible.	strongly disagree								strongly agree
Using it is effortless.	strongly disagree								strongly agree
I can use it without written instructions.	strongly disagree								strongly agree
I don't notice any inconsistencies as I use it.	strongly disagree								strongly agree
Both occasional and regular users would like it.	strongly disagree								strongly agree
I can recover from mistakes quickly and easily.	strongly disagree								strongly agree
I can use it successfully every time.	strongly disagree								strongly agree
EASE OF LEARNING		**1**	**2**	**3**	**4**	**5**	**6**	**7**	
I learned to use it quickly.	strongly disagree								strongly agree
I easily remember how to use it.	strongly disagree								strongly agree
It is easy to learn to use it.	strongly disagree								strongly agree
I quickly became skillful with it.	strongly disagree								strongly agree
SATISFACTION		**1**	**2**	**3**	**4**	**5**	**6**	**7**	
I am satisfied with it.	strongly disagree								strongly agree
I would recommend it to a friend.	strongly disagree								strongly agree
It is fun to use.	strongly disagree								strongly agree
It works the way I want it to work.	strongly disagree								strongly agree
It is wonderful.	strongly disagree								strongly agree
I feel I need to have it.	strongly disagree								strongly agree
It is pleasant to use.	strongly disagree								strongly agree

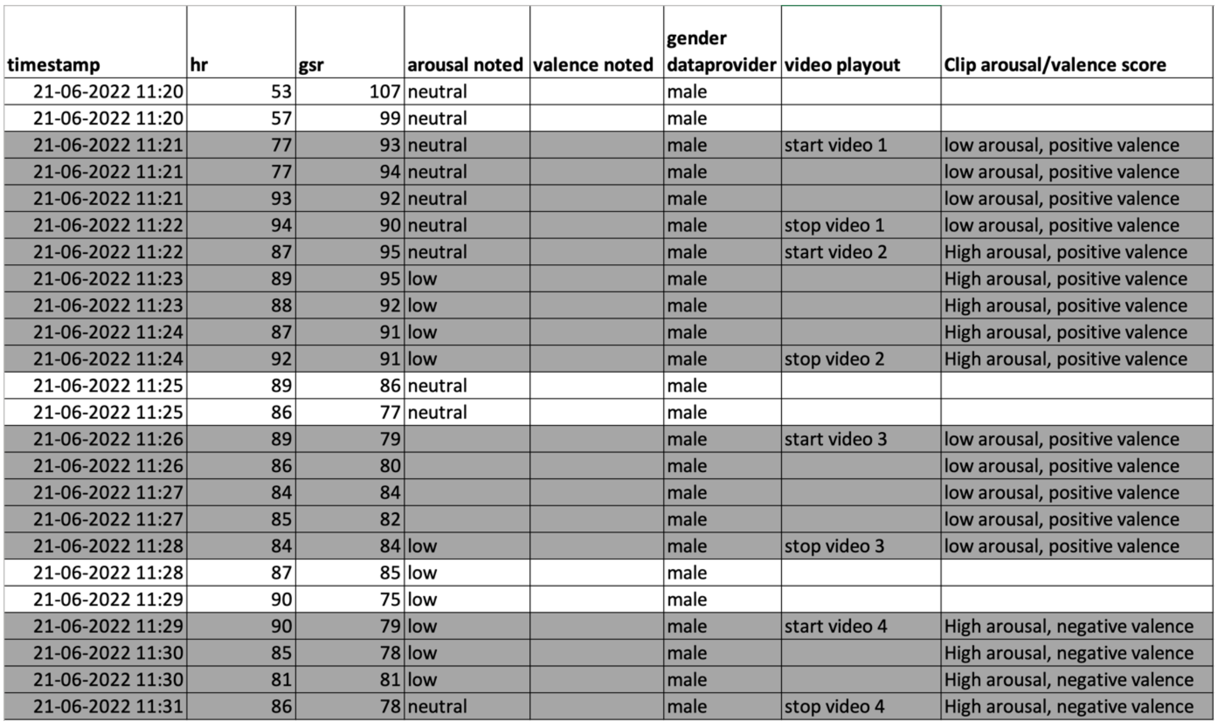
Comments regarding the interaction with the physicalisation:

Appendix I: User study results



Timestamp	hr	gsr	noted(1=low, 10=high)	noted(1=positive, 10=negative)	dataprovider	video playout	Clip arousal/valence score
21-06-2022 09:06	0	335			male		
21-06-2022 09:07	99	143	7	1	male		
21-06-2022 09:07	130	112	7	1	male		
21-06-2022 09:07	139	110	7	1	male		
21-06-2022 09:08	125	151	7	5	male	start video 1	low arousal, positive valence
21-06-2022 09:08	125	75	7	5	male		low arousal, positive valence
21-06-2022 09:09	125	78			male		low arousal, positive valence
21-06-2022 09:09	122	88			male		low arousal, positive valence
21-06-2022 09:10	122	88	7	1	male		low arousal, positive valence
21-06-2022 09:10	122	98	7	1	male	stop video 1	low arousal, positive valence
21-06-2022 09:11	122	80			male		
21-06-2022 09:11	122	81			male		
21-06-2022 09:12	122	81			male	start video 2	High arousal, positive valence
21-06-2022 09:12	122	86			male		High arousal, positive valence
21-06-2022 09:13	122	87			male		High arousal, positive valence
21-06-2022 09:13	122	83			male		High arousal, positive valence
21-06-2022 09:14	122	72		3	male	stop video 2	High arousal, positive valence
21-06-2022 09:14	122	87		3	male		
21-06-2022 09:15	122	104		3	male		
21-06-2022 09:15	122	76		3	male	start video 3	low arousal, positive valence
21-06-2022 09:16	122	81	7	1	male		low arousal, positive valence
21-06-2022 09:16	122	67	7	1	male		low arousal, positive valence
21-06-2022 09:17	122	74			male		low arousal, positive valence
21-06-2022 09:18	122	79			male		low arousal, positive valence
21-06-2022 09:18	122	67			male		low arousal, positive valence
21-06-2022 09:18	122	74			male	stop video 3	low arousal, positive valence
21-06-2022 09:19	122	79			male		
21-06-2022 09:19	123	72			male		
21-06-2022 09:20	126	67			male	start video 4	High arousal, negative valence
21-06-2022 09:20	126	76			male		High arousal, negative valence
21-06-2022 09:21	126	74			male		High arousal, negative valence
21-06-2022 09:21	126	78			male		High arousal, negative valence
21-06-2022 09:22	126	525			male	stop video 4	High arousal, negative valence

Figure i.1: User study round 1



timestamp	hr	gsr	arousal noted	valence noted	gender dataprovider	video playout	Clip arousal/valence score
21-06-2022 11:20	53	107	neutral		male		
21-06-2022 11:20	57	99	neutral		male		
21-06-2022 11:21	77	93	neutral		male	start video 1	low arousal, positive valence
21-06-2022 11:21	77	94	neutral		male		low arousal, positive valence
21-06-2022 11:21	93	92	neutral		male		low arousal, positive valence
21-06-2022 11:22	94	90	neutral		male	stop video 1	low arousal, positive valence
21-06-2022 11:22	87	95	neutral		male	start video 2	High arousal, positive valence
21-06-2022 11:23	89	95	low		male		High arousal, positive valence
21-06-2022 11:23	88	92	low		male		High arousal, positive valence
21-06-2022 11:24	87	91	low		male		High arousal, positive valence
21-06-2022 11:24	92	91	low		male	stop video 2	High arousal, positive valence
21-06-2022 11:25	89	86	neutral		male		
21-06-2022 11:25	86	77	neutral		male		
21-06-2022 11:26	89	79			male	start video 3	low arousal, positive valence
21-06-2022 11:26	86	80			male		low arousal, positive valence
21-06-2022 11:27	84	84			male		low arousal, positive valence
21-06-2022 11:27	85	82			male		low arousal, positive valence
21-06-2022 11:28	84	84	low		male	stop video 3	low arousal, positive valence
21-06-2022 11:28	87	85	low		male		
21-06-2022 11:29	90	75	low		male		
21-06-2022 11:29	90	79	low		male	start video 4	High arousal, negative valence
21-06-2022 11:30	85	78	low		male		High arousal, negative valence
21-06-2022 11:30	81	81	low		male		High arousal, negative valence
21-06-2022 11:31	86	78	neutral		male	stop video 4	High arousal, negative valence

Figure i.2: User study round 2

timestamp	hr	gsr	arousal noted	valence noted	gender dataprovider	video playout	Clip arousal/valence score
21-06-2022 13:39	91	104			female		
21-06-2022 13:39	92	100			female		
21-06-2022 13:40	89	81			female		
21-06-2022 13:40	98	82			female	Start video 1	High arousal, positive valence
21-06-2022 13:41	112	83			female		High arousal, positive valence
21-06-2022 13:41	104	74			female		High arousal, positive valence
21-06-2022 13:42	103	41			female	unexptected break	
21-06-2022 13:42	94	80			female		
21-06-2022 13:43	101	74			female		
21-06-2022 13:43	93	74			female		
21-06-2022 13:44	102	72			female	resume video	High arousal, positive valence
21-06-2022 13:44	99	73			female		High arousal, positive valence
21-06-2022 13:44	99	73			female	stop video 1	High arousal, positive valence
21-06-2022 13:45	102	42	neutral		female	start video 2	low arousal, positive valence
21-06-2022 13:45	99	72	neutral		female		low arousal, positive valence
21-06-2022 13:46	100	72			female		low arousal, positive valence
21-06-2022 13:46	104	72			female		low arousal, positive valence
21-06-2022 13:47	105	73			female		low arousal, positive valence
21-06-2022 13:47	99	56			female		low arousal, positive valence
21-06-2022 13:48	112	72			female		low arousal, positive valence
21-06-2022 13:48	109	71			female		low arousal, positive valence
21-06-2022 13:49	108	72			female	stop video 2	low arousal, positive valence
21-06-2022 13:49	101	72			female		
21-06-2022 13:50	102	71	neutral		female		
21-06-2022 13:50	103	72	neutral		female	start video 3	low arousal, positive valence
21-06-2022 13:51	104	70			female		low arousal, positive valence
21-06-2022 13:51	104	27			female		low arousal, positive valence
21-06-2022 13:52	104	70			female		low arousal, positive valence
21-06-2022 13:52	107	70			female	stop video 3	low arousal, positive valence
21-06-2022 13:53	100	70	neutral		female		
21-06-2022 13:53	103	73	neutral		female	start video 4	High arousal, negative valence
21-06-2022 13:54	106	69			female		High arousal, negative valence
21-06-2022 13:54	109	70			female	stop video 4	High arousal, negative valence

Figure i.3: User study round 3

timestamp	hr	gsr	arousal noted	valence noted(1=positive, 10=negative)	gender dataprovider	video playout	Clip arousal/valence score
22-06-2022 13:52	1	643	low	5	male		
22-06-2022 13:53	0	631	low	5	male		
22-06-2022 13:53	52	465	low	5	male		
22-06-2022 13:54	59	324	low	5	male	start video 1	low arousal, positive valence
22-06-2022 13:54	81	288	low	5	male		low arousal, positive valence
22-06-2022 13:55	77	259	low	5	male		low arousal, positive valence
22-06-2022 13:55	70	274	low	5	male		low arousal, positive valence
22-06-2022 13:56	68	260	low	5	male	stop video 1	low arousal, positive valence
22-06-2022 13:56	72	234	low	5	male		
22-06-2022 13:57	71	239	low	4	male	start video 2	High arousal, positive valence
22-06-2022 13:57	74	253	low	4	male		High arousal, positive valence
22-06-2022 13:58	72	271	low	1	male		High arousal, positive valence
22-06-2022 13:58	72	308	low	1	male		High arousal, positive valence
22-06-2022 13:59	73	319	low	4	male		High arousal, positive valence
22-06-2022 13:59	74	310	low	4	male	stop video 2	High arousal, positive valence
22-06-2022 14:00	73	291	low	4	male		
22-06-2022 14:00	70	294	low	4	male	start video 3	low arousal, positive valence
22-06-2022 14:01	80	266		2	male		low arousal, positive valence
22-06-2022 14:01	70	303		2	male		low arousal, positive valence
22-06-2022 14:02	71	343	low	5	male		low arousal, positive valence
22-06-2022 14:02	71	368	low	5	male		low arousal, positive valence
22-06-2022 14:03	70	382	low	2	male		low arousal, positive valence
22-06-2022 14:03	75	391	low	2	male		low arousal, positive valence
22-06-2022 14:04	78	391	low	2	male		low arousal, positive valence
22-06-2022 14:04	78	386	low	2	male		low arousal, positive valence
22-06-2022 14:05	75	364		2	male	stop video 3	low arousal, positive valence
22-06-2022 14:05	74	288		2	male		
22-06-2022 14:06	70	273		2	male	start video 4	High arousal, negative valence
22-06-2022 14:06	70	257		2	male		High arousal, negative valence
22-06-2022 14:07	68	280	low	3	male		High arousal, negative valence
22-06-2022 14:07	71	267	low	3	male	stop video 4	High arousal, negative valence
22-06-2022 14:07	75	135	low	3	male		

Figure i.4: User study round 4

timestamp	hr	gsr	arousal noted	valence noted	gender dataprovider	video playout	Clip arousal/valence score
24-06-2022 09:25	58	158			male	start video 1	High arousal, positive valence
24-06-2022 09:25	77	258			male		High arousal, positive valence
24-06-2022 09:26	73	209			male		High arousal, positive valence
24-06-2022 09:26	61	264			male		High arousal, positive valence
24-06-2022 09:27	73	272			male		High arousal, positive valence
24-06-2022 09:27	79	243			male	stop video 1	High arousal, positive valence
24-06-2022 09:28	67	145	low	neutral	male	start video 2	low arousal, positive valence
24-06-2022 09:28	87	122	low	neutral	male		low arousal, positive valence
24-06-2022 09:29	88	114			male		low arousal, positive valence
24-06-2022 09:29	78	158			male		low arousal, positive valence
24-06-2022 09:30	79	243	low	positive	male		low arousal, positive valence
24-06-2022 09:31	81	259			male		low arousal, positive valence
24-06-2022 09:32	85	220			male	stop video 2	low arousal, positive valence
24-06-2022 09:32	73	159			male		
24-06-2022 09:33	67	94			male	start video 3	low arousal, positive valence
24-06-2022 09:33	76	62			male		low arousal, positive valence
24-06-2022 09:34	77	133	rising	negative	male		low arousal, positive valence
24-06-2022 09:34	78	113	rising	negative	male		low arousal, positive valence
24-06-2022 09:34	66	66	rising	negative	male		low arousal, positive valence
24-06-2022 09:35	72	81			male		low arousal, positive valence
24-06-2022 09:35	73	66			male	stop video 3	low arousal, positive valence
24-06-2022 09:36	83	72	low-neutral	positive-neutral	male		
24-06-2022 09:36	81	54	low-neutral	positive-neutral	male	start video 4	High arousal, negative valence
24-06-2022 09:37	79	71			male		High arousal, negative valence
24-06-2022 09:37	75	85			male		High arousal, negative valence
24-06-2022 09:38	78	122	low-neutral	neutral-negative	male		High arousal, negative valence
24-06-2022 09:38	75	165	low-neutral	neutral-negative	male	stop video 4	High arousal, negative valence
24-06-2022 09:39	86	69			male		
24-06-2022 09:39	85	52			male		

Figure i.5: User study round 5

Questions/**Catagory**							
usefulleness	User 1	user 2	user 3	user 4	user 5		average
It is useful.	4	4	6	5	6		5
It does everything I would expect it to do.	4	1	2	5	5		3,4
ease of use							
It is easy to use.	4	6	7	4	6		5,4
It is simple to use.	5	6	7	5	6		5,8
It is user friendly.	5	6	7	5	4		5,4
It requires the fewest steps possible to accomplish what I want to do with it.	5	6	7	4	5		5,4
It is flexible.	-1	-1	-1	-1	-1		
Using it is effortless.	5	6	7	4	3		5
I can use it without written instructions.	1	3	2	3	3		2,4
I don't notice any inconsistencies as I use it.	2	1	3	5	2		2,6
Both occasional and regular users would like it.	6	6	4	6	3		5
I can recover from mistakes quickly and easily.	6	2	4	5	7		4,8
I can use it successfully every time.	4	1	3	5	6		3,8
ease of learning							
I learned to use it quickly.	6	6	4	7	5		5,6
I easily remember how to use it.	6	6	7	7	5		6,2
It is easy to learn to use it.	6	6	6	7	5		6
I quickly became skillful with it.	6	6	6	7	3		5,6
satisfaction							
I am satisfied with it.	5	2	5	5	5		4,4
I would recommend it to a friend.	5	2	4	5	5		4,2
It is fun to use.	6	2	6	6	5		5
It works the way I want it to work.	5	1	3	4	6		3,8
It is wonderful.	4	4	4	5	5		4,4
I feel I need to have it.	5	3	1	2	2		2,6
It is pleasant to use.	5	3	4	4	5		4,2

Figure i.6: Results USE questionnaire