The modernised shower

IxD4 - Project group: cs-24-IxD-4-01

Aalborg University
Interaction design/IxD

Interaction design/IxD
Aalborg University
https://www.aau.dk

Title:
The modernised shower

Theme:
Physical design in the bathroom

Project Period:
01/02-2024 to 29/05-2024

Project Group:
cs-24-ixd-4-01

Participant(s):
Aske Secher Poulstrup
Emil Vigand Kimman
Jacob Hesselby Andersen
Rasmus Bredal Schultz

Supervisor(s):
Sander De Jong

Copies: 0

Page Numbers: 54

Date of Completion:
29th of May 2024

Abstract:

The theme of this project was "Physical Design within the bathroom".
The objective was to create a physical interactive prototype through an iterative process. The prototype for this project resulted in a customisable shower experience, that aims to make the experience of taking a shower more tailored for the individual user. The project started with a brainstorming session, followed by a Moscow model for setting prototype requirements. From the Moscow model and related work, it became clear what were necessary features for making a tailored experience. After the Moscow model, lo-fi sketches were developed and further refined as hi-fi sketches. The final concept was a physical prototype working through presets on a wall-mounted touchscreen, with customisable features like water pressure, temperature and height adjustment. By using these attributes, the experience can potentially achieve a more modernised shower experience.

Preface

This report is devised in the spring of 2024 from the 1st of February through the 29th of May. This fourth-semester project is a part of the Interaction Design bachelor at Aalborg University (AAU).

The topic for this project is "Physical Design within the bathroom".

We would like to thank our Supervisor, Sander De Jong, who provided guidance and counselling throughout the project period. We would also like to thank Timothy Merrit for helping us with questions about the physical design process.

Throughout this project the following AI-tools have been used: Chatgpt (“Chatgpt” n.d.), Github co-pilot (“Github Co-pilot” n.d.) & Goblintools (“Goblintools Formalizer” n.d.). These tools have been used for rephrasing and rewriting sentences and code.

1. Introduction

The scope of this project was to design and construct an interactive physical prototype, where the theme for the prototype was "Designing for the bathroom". The prototype had to be functional and would therefore be a demonstration of the project group's technical, programming and designing skills.

In the constantly changing world of digital experiences, user engagement and satisfaction stand as clear focus points for businesses striving to stay ahead. As the digital world expands, the demand for tailored and personalised interactions has become increasingly widespread. Addressing this demand, customisation and personalisation have emerged as powerful tools to enhance user experiences and foster meaningful engagements .

The article "Customisation and Personalisation" by the Nielsen Norman Group (schade, n.d.), focuses on the intricacies of these concepts and their roles in shaping modern digital interactions. In this project, we will explore the development of a prototype that showcases modernisation through customisation and personalisation.

Personalisation is done by the user interacting with the system. Developers make the system identify the individual user and deliver the content, experience or functionality they desire. Personalisation can be done either individually or at a group/audience level. This project will focus on individual personalisation (schade, n.d.).

Customisation is enacted by the user. A system may offer opportunities for the user to customise or make changes to their experience. This could be that the user has specific needs for their configuring layouts, content or system functionality. Customisation may also involve moving or prioritising actions in an interface. This could be a layout change for a better experience (schade, n.d.).

Modernisation is the process of adapting something to modern needs or habits (dictionary, n.d.). This includes the customisation and personalisation of products and digital solutions to best meet the preferences of users. The practice of tailoring to users needs, is widespread in modern technologies and products, as there is an increasing demand from users for this feature (schade, n.d.).

While researching a theme for this project the project group were brainstorming new products. This included unseen opportunities for needs users have in the bathroom that haven't been met yet. After a period of brainstorming through different methods, there was a change in perspective. Instead, the focus became making a more individual experience of already existing experiences.

Then came the idea of a modernised shower. A shower system with the ability to customise certain aspects of the shower, and by presets making it a customised experience. Based on the motivation, a problem statement has been developed:

How can we develop a physical prototype that modernises the experience of taking a shower?

Related work
In order to gain insight into existing work regarding interactivity in a shower setting, we will examine how other products in this field have been developed. The project group came across the company "Kohler" which specialises in interactive furniture for kitchen and bathroom environments. Kohler's focus is the individuals well being supported by good design and an innovative process (Kohler, n.d.).

The first item from Kohler, that the project group pondered upon was the "DTV+". A digital interface with four controllable elements for giving the user a true multi-sensory shower experience. These four elements are water, steam, lighting and music. Every element is completely customisable to the user's needs and is controlled by a touchscreen interface, as seen in Figure 1.1. The interface also has a function that can store up to six user-definable presets, so each user's favourite setting can be saved for future use. The interface also displays the current temperature, desired temperature, time of day and which preset is chosen. Underneath the display are three touch buttons which are on/off, temperature down and temperature up. These buttons are colour-coded for easier understanding, and the interface is made for wall mounting both in and out of the shower.

Figure 1.1: Kohler DTV+ model

Another product the project group looked at from Kohler was the "Anthem". A digital control display that makes the controls of the users costume shower be right at their fingertips. With a sleek and minimalistic display it lets the user configure and control three different shower functions to their desire. These shower functions are temperature and water pressure for creating a personalised experience for each use. The controller also have a built-in eco mode that lets the user conserve water, and then remind the user of their mindful water usage. The display and more feature highlights are shown on Figure 1.2.
Kohler sells the Anthem on its built-in, preconfigured hydrotherapies, which all are designed to enhance personal well-being with experiences that either relax, invigorate or deliver a sense of escape (Kohler, n.d.). The built-in modes are: Wake Up, Warm Up, Sleep Simple, Shine/Tone, and Cool Down. Like the DTV+ the Anthem also have an option for presets however the Anthem have ten presets. As a part of these relaxing experiences the Anthem also offers a variety of different shower heads to purchase with it.

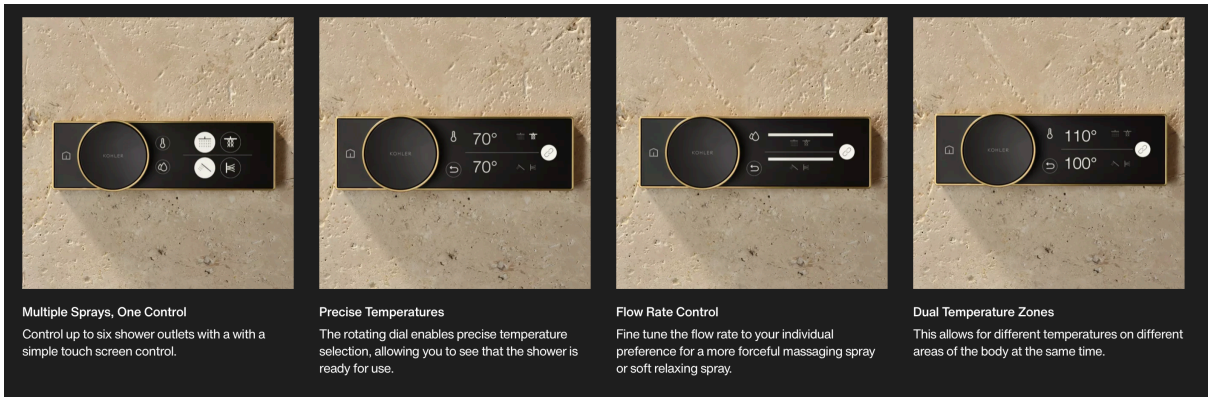


Figure 1.2: Kohler Anthem model

2. Method

This Chapter will outline the principles of customisation and personalisation and lay the foundation of the project's idea generation for the prototype.

2.1 Personalisation and customisation

Businesses strive to crater experiences for the individual customer. Two phenomena that have gained recognition are customisation and personalisation. While often used interchangeably, these concepts possess unique characteristics and implications for both consumers and businesses (schade, n.d.).

2.1.1 Customisation

Customisation revolves around empowering consumers to tailor products or services according to their unique preferences and requirements. It involves providing users with a range of options and features that they can adjust to suit their individual needs. Customisation enables users to exert control over their experiences, fostering a sense of ownership and satisfaction. An example is Nike's customisable sneakers, which allow consumers to select colours, materials, and even add customisations like initials or symbols. By offering such options, Nike not only enhances the utility of its products but also taps into consumers' desire for self-expression and individuality (schade, n.d.).

2.1.2 Personalisation

On the other hand, personalisation hinges on leveraging data and technology to deliver tailored experiences to individual users. By analysing user behaviour, preferences, and demographic information, businesses can anticipate users' needs and present them with relevant content or recommendations (schade, n.d.).
An example of this is Netflix's personal recommendations for users. Without any input from the user taking an active decision, Netflix recommends series and movies depending on watch history and watch time (Medium, n.d.).

2.1.3 Comparison

While both Customisation and personalisation aim to enhance user experiences, they differ in their approach and execution:
User control
Customisation empowers users to actively participate in shaping their experiences by providing them with options and features to customise. In contrast, personalisation

operates in the background, leveraging data and algorithms to deliver tailored experiences without requiring explicit user input (schade, n.d.).

Scope of adaptation

Customisation primarily focuses on adapting the product or service itself, allowing users to tweak its features or attributes. personalisation extends beyond the product to encompass the entire user journey, including content recommendations, marketing messages, and user interfaces.

Level of automation

Customisation often involves manual processes wherein users make explicit choices to customise their experiences. Personalisation relies on automation and algorithms to analyse data and deliver tailored experiences at scale, often in real-time.

Intent and context

Customisation is driven by users' explicit preferences and choices, reflecting their immediate needs or desires. In contrast, personalisation operates based on implicit cues and historical data, aiming to anticipate users' future needs and preferences proactively.

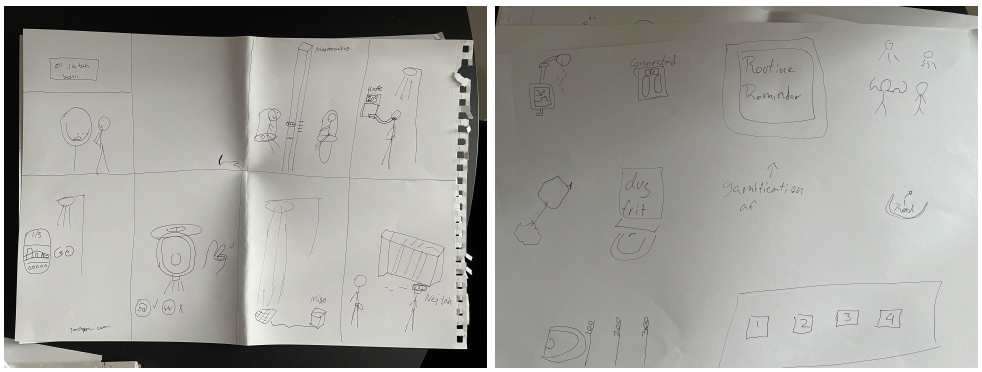
2.2 Idea generation

Generating ideas is a crucial aspect of the initial phase of a new project, as this is where all ideas can be explored. Various methods can be utilised for idea generation, such as the techniques of Crazy 8 and the Method 635, which have been applied by the project group to encourage diversity in ideas.

2.2.1 Crazy 8

The project group used a brainstorming technique known as "Crazy 8" to initiate the process of generating solutions for the bathroom (Google, n.d.). The initial iteration of Crazy 8 involved each member generating ideas without limitations, as can be seen in Figure 2.1a.

The second iteration of Crazy 8 was done by limiting it to specific subjects to narrow the idea generation. One session had the subject of public bathrooms and one session with private bathrooms, with focus on the gamification of public bathrooms. The Crazy 8's can be seen in Figure 2.1b.



(a) First iteration of Crazy 8 (b) Second iteration of Crazy 8

Figure 2.1

During the second iteration, the project group identified three ideas that showed potential for further development, leading to the creation of the third iteration of Crazy 8. The three ideas: A public-themed sensing bathroom, shower games, and foot-controlled restroom games. The public-themed idea had examples of being deployed at amusement parks, where customising the design of the restroom based on its location within the amusement park. The second idea was karaoke in the bath with a screen on the wall displaying the lyrics of a song and a microphone attached to it. The third idea was a game that could be played on the ground below the toilet seat to keep its users entertained while sitting on the toilet. This idea was iterated on having a practical use in both the public and private bathrooms. After having iterated on the ideas using the Crazy 8, we will use another idea generation method called Method 635.

2.2.2 Method 635

We utilised Method 635 ("Method 635" n.d.), to further explore our main ideas, but the method was slightly altered to be more relevant to our project and group dynamic. As there were only four group members and four different overall ideas that needed potential solutions. Each person would write three solutions to each idea, before passing it on to the next person. This is different from how the method is intended, as we wrote the title of four specific ideas and iterated three potential solutions to each of those ideas. The different ideas consisted of: A game that could be played on the floor with your feet in a public bathroom, a karaoke game that can be played in the shower, a customisable shower experience to tailor for the individuals and a bathroom experience which focuses on the different human senses, which would improve the overall bathroom experience.
Each time the paper was passed on to the next person, each solution would be further iterated upon, giving new ideas which weren't discussed previous to the exercise. It can be seen how each person further iterated upon the previous idea in the Figure 2.2

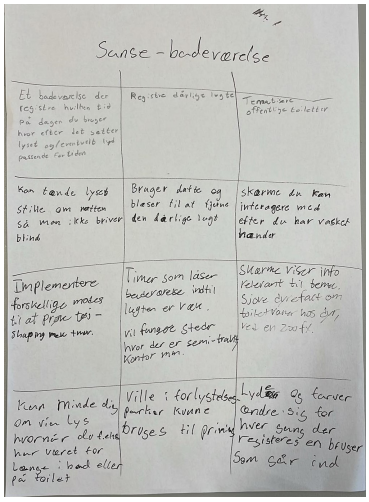


Figure 2.2: Method 635

This exercise helped figure out which ideas wouldn't work in the projects scope and narrowed down the idea, which gave room for discussion. The best solution the project group agreed upon was the customisable shower experience, as it offered the possibility for an interesting project subject. The customisable shower experience had the appropriate level of difficulty for the project scope and could fulfil the study plan requirements for this semester. The next step will include outlining the priorities for implementation of features for the shower, using the MoSCow method.

2.2.3 MoSCoW

For this project, the project group has used the MoSCoW model to establish an understanding of prioritisation when developing the prototype. Because of the length of the project only spans within a singular semester, prioritisation is an important part of the process. The MoSCoW model, the project group has made, can be seen in Figure 2.3.

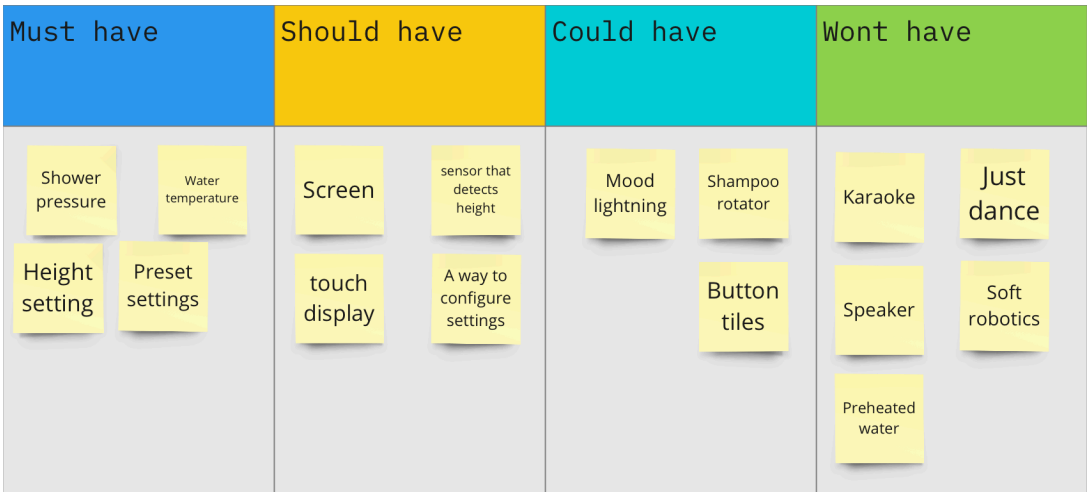


Figure 2.3: MoSCoW model

Must have
In the MoSCoW prioritising model, must have requirements are essential for the prototype's functionality. When considering the customisation of the shower, these essential requirements may include:

- **Control of water pressure**
- **Adjustments to the shower head height**
- **Reading and displaying water temperature**
- **Having presets for profiles**

These requirements are based on the basic attributes of a shower. This is water control, temperature and height of the shower head. These attributes are essential for the prototype to operate and function as a standard shower. Given that the prototype enables users to customise their shower experience, it is a "Must have" to have the options for presets.

The control of water pressure includes the capability to adjust the pressure of the water, similar to a standard shower. It is intended to serve as a primary setting that can be preset to a desired pressure level, which can be adjusted.
Adjustments to the height of the shower head will be a key feature of the prototype, as individual preferences for height may differ. This functionality will allow users to preset their desired height, eliminating the need for manual adjustments before each shower.
When showering, there may be variations in temperature preferences among household members. Therefore, the ability to set a preset temperature will improve the customisation of the shower experience.
The option to utilise presets will be essential for customising the shower experience. This feature will eliminate the requirement for manually adjusting the shower settings, allowing for tailored experiences for each user.

Should have
These requirements cover the useful but not necessary aspects of the prototype. These requirements may be important, but they are not crucial for the prototype's functionality. These requirements will be:

- **A display**
- **Sensor to detect height**
- **Touch screen**
- **A way to configure settings**

These requirements are not essential for the function of the prototype, but should help the interaction between the shower customisation and the user. The display is merely there to help visualise their preferences and make for easy access for the individual to reconfigure as well as modernise the bath experience.

As the prototype will utilise presets to configure the shower to the individual's preferences, users will have the option to adjust these presets through a screen displaying the settings. The display is not a critical part for prototype functionality, but it will enhance the visualisation of user preferences.
A sensor will allow the shower head to automatically configure according to the user's height, when creating a profile or otherwise taking a shower without using a profile. A height sensor is not needed for the main function of the prototype.

Utilising displays to present data, such as temperature and water pressure, is commonly effective. However, in this prototype, enhancing the capability to view and customise presets could be achieved by integrating a touch screen interface. While buttons could serve as an alternative, a visually intuitive touch display could optimise the overall user experience, as adjusting preset settings might prove cumbersome with buttons. As it is a necessity for the prototype to include preset options, there needs to be a method available for configuring these presets.

Could have
These requirements are the nice-to-have features. They are not necessary for the core function of the prototype, and have a smaller impact if left out, these include:

- **Mood lightning**
- **Shampoo rotator**
- **Button tiles**

For a sensory experience, the project group talked about the shower having a light setting so the user could enjoy different sensory experiences. This feature could be a future addition to the prototype, but for the customisable experience, this is not a "must have" feature.

A shelf for shampoo and other bath appliances, that can rotate depending on which person takes a shower. The shampoo rotator allows a servo to rotate a cylinder placed in the wall. This cylinder is split up into three chambers. Each chamber allows the different users their individual shower products. This feature falls outside the current scope of the project, which primarily focuses on enhancing the overall showering experience within the shower enclosure.

As a play feature and for easy configuration, the project group discussed making the bathroom tiles interactive. The idea was to make the tiles in the shower touch-sensitive so they could act as buttons. It was decided that tiles could be

an interesting interaction, but since bathroom tiles are different from bathroom to bathroom, the prototype should include push or turn buttons

Won't have
Will-not-have requirements are all expectations for what the final prototype will not feature. A great benefit of this category is preventing scope creep.

- **Karaoke**
- **Just dance**
- **Speaker**
- **Soft robotics**
- **Preheated water**

Several of these ideas fall outside the scope of the project, however, they were part of the idea generation phase and therefore included to show the decision-making process for the project.

The idea was to implement a microphone on top of the shower head and have song lyrics on the display so that users could sing karaoke in the shower. This could be a fun prototype, but it may increase the water budget for a household since a shower would take significantly longer.
Implementing a mat and arrows on the floor could make it possible for users to play the game "Just Dance" via the display. Dancing and taking a shower increases both water use and the risk of injury, which is not a risk the project group wanted to take.
Incorporating a speaker in the shower, so that users could enjoy music. Since a speaker is already widely used in other rooms as well as bathrooms the project group didn't see the necessity of integrating a speaker in the project.
Utilising soft robotics for texture and button feel. The project group wanted to work with soft robotics, but also without forcing it into the prototype. The idea of implementing soft robotics seemed to be more inconvenient than useful.
Utilising the adjustable water temperature and presets so that the user could enjoy their preferred temperature as soon as they turn on the water. The prototype for this solution would need a hot water chamber that would need to be kept hot all hours of the day. This solution is not sustainable and therefore a "won't have".

2.3 Sub-conclusion

In this chapter the principles of customisation and personalisation were explained in depth, as well as the requirements for the prototype have been made. This started the overall conceptualisation of the physical prototype.

3. Design

This chapter will discuss various iterations in the design of the user interface for the screen. It will begin with a Lo-Fi design, followed by an explanation of the idea generation process. Subsequently, the design process will progress to a Hi-Fi stage, where each page of the user interface will be examined in detail. Finally, a visualisation in 3ds Max will be presented to demonstrate the potential appearance of the completed product.

3.1 Key focus points

A shower is made up of three key focus points: the height of the shower head, the water temperature, and the water pressure. At present, users are responsible for adjusting and manually setting the temperature on the mixing valve, as well as controlling the water pressure. This can lead to varying shower preferences among individuals within a household.
In order to facilitate the customisation of shower settings, an interface will be introduced to allow individuals in the household to personalise their shower experience. This feature will include the ability to save preference settings for different attributes of the shower. Each user will have the option to create a personalised profile to store their unique preferences. The interactive interface will enhance the user's ability to adjust and save their desired shower presets.

3.2 Concept of prototype functionality

The idea behind the shower is meant to be a personalised and customisable experience. A wall-mounted display will be attached somewhere in the shower, where the user can adjust the different settings and create a profile with their personal preferences regarding shower height, water temperature and water pressure. When a user enters the shower, their profile can be selected on the menu screen. From their profile, the shower collects a set of pre-entered data and adjusts itself according to those. From the temperature setting, a valve lets in different volumes of cold and hot water to mix and match the desired temperature. The water pressure will be adjusted by the valve controlling how much and how quickly the water is let into the shower hose. How these key functionalities will be achieved, will be explained in further detail later.

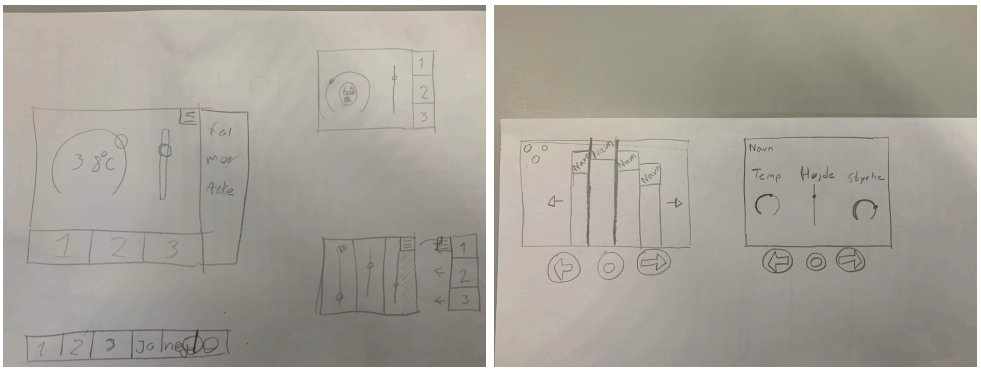
In order to provide a clearer explanation of the concept, an AI-generated image has been created to aid in visualising the idea, as seen in Figure 3.1.

Figure 3.1: Picture of shower concept generated by Bing Copilot (Copilot, n.d.).

3.3 Lo-Fi iterations

In the project group, Lo-Fi sketches have been used to develop the layout and design of the graphical user interface on the display. At this point Lo-Fi sketches have been utilised instead of Hi-Fi sketches, as they are more efficient at rapidly mapping out and exploring numerous design concepts without prematurely committing to a finalised idea in the early stages of ideation. Hi-Fi sketches are typically reserved for fully fleshed out concepts which require more time and detail.

In Figure 3.2a, some initial display layout designs featuring sliders in various configurations for convenient adjustment of shower settings. The initial version allows users to select a profile and customise temperature and water pressure settings.
In Figure 3.2b, the concept is to have the display only show the configurations and utilise buttons for navigation. Two additional iterations are shown in the Figure.



(a) Lo-Fi sketch one (b) Lo-Fi sketch two

Figure 3.2

3.4 Hi-Fi iterations

In Figure 3.3 one of the group members made their solution for the prototype UI. This member tried to see if it was possible to have everything on one page while maintaining a minimalistic look of the overall design. This included having the six users' presets and the adjustable parameters on one page and having a simple colour layout. Having too many interactable sites leads to a longer interaction, and since the display acts as the shower mixing valve, a short interaction could be a beneficial factor for the overall user experience.

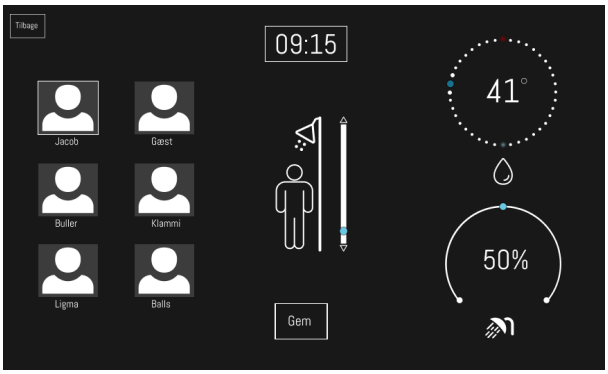


Figure 3.3: Hi-Fi sketch one

The iteration on Figure 3.4 focuses on accessibility with buttons. Instead of only focusing on possibilities with a touch screen, the project group wanted to explore potential features with buttons. In this sketch, three buttons have been implemented under the screen. These buttons control if the water pressure should go up or down, as well as if the shower should be turned on/off.
The display itself has a user icon in the top-left corner for a way to return to the user selection screen. It also displays the time in the top-middle, and then different sliders for the three focus attributes: Temperature, water pressure and height.

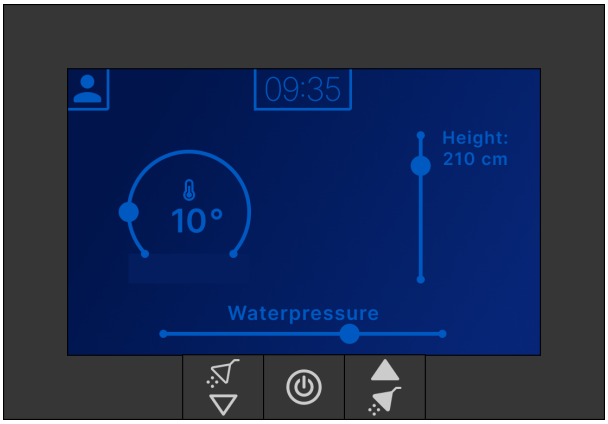


Figure 3.4: Hi-Fi sketch two

In Figure 3.3, the focus of the iteration was to include all options onto one page to

simplify the selection process and reduce the need for multiple pages. A challenge arose with the limited screen space for displaying sliders, which could make it difficult for users to accurately adjust water pressure and temperature. As a result, a new iteration was developed where the sliders were made more prominent and easier to manipulate on the small screen, as depicted in Figure 3.4. The final Hi-Fi iteration, shown in Figure 3.5, incorporates both the improved sliders and profile selection features from the previous iterations.

Figure 3.5: Hi-Fi sketch three

3.5 Final design

After iterating on the design, the project group started creating the final design in Figma.

3.5.1 Startscreen

Figure 3.6 shows the final iteration of the start screen. This is the screen the user will see when the device has been idle. The screen is made by a gradient background that sets the colour scheme for the rest of the final design pages. On this screen specifically is a red/orange button indicating to the user that the device needs to be turned on. The purple water droplet is only there for aesthetic purposes.

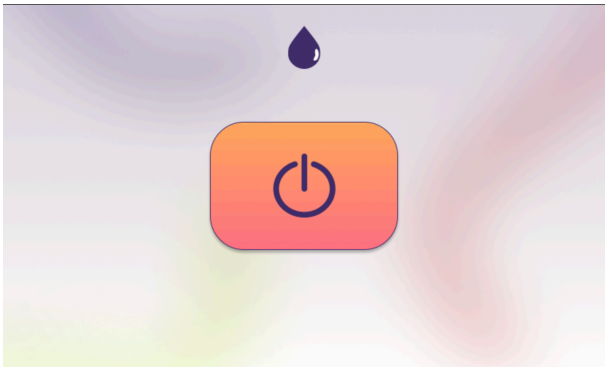


Figure 3.6: The final programmed version of the start page

The power-on button is large in size and makes use of Fitt's Law (Yablonski, n.d.). Fitt's Law is a description of how touch targets should be large enough for users to accurately click them. Since the power-on button is the only button on the start screen, the project group wanted to make sure that the button was easy to click.

3.5.2 Profile selection

Figure 3.7 shows the final iteration of the profile-select screen. After pressing the power-on button on Figure 3.6 the user will end up on this page. On this page, each family member will have a user-specific account. The Figure shows four preset profiles, an additional button for adding more users and a "Continue as guest"-button. The "Continue as guest"-button is intended for users who normally don't live in the household where the prototype is mounted.

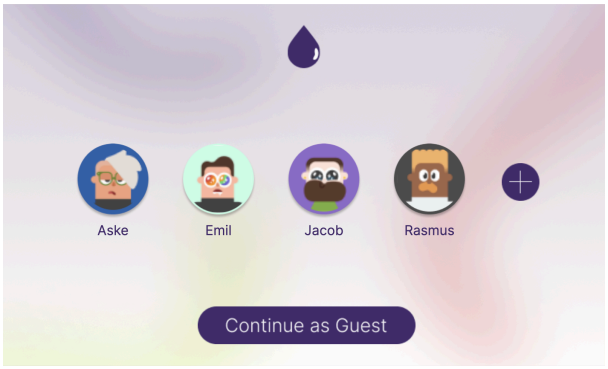


Figure 3.7: The final programmed version of the profile-select page

Jakob's law states that users spend most of their time on other websites than yours(Yablonski, n.d.). Therefore the profile selection screen is heavily inspired by the Netflix profile selection. Both how the users are placed and how the add-user button is. The difference is the Continue as guest option, which was a necessity for this prototype.

3.5.3 Mainpage

Figure 3.8 shows the final iteration of the profile-mainpage screen. After selecting either a preset user or the "Continue as guest"-button as seen in Figure 3.7, the user is led to their personal page. This page includes sliders for each of the customisable features such as Temperature, water pressure and height adjustment. The system will remember their settings and automatically set the three attributes to the user's desired setting. The project group decided to implement a variation of sliders so it would be easier to differentiate between them.

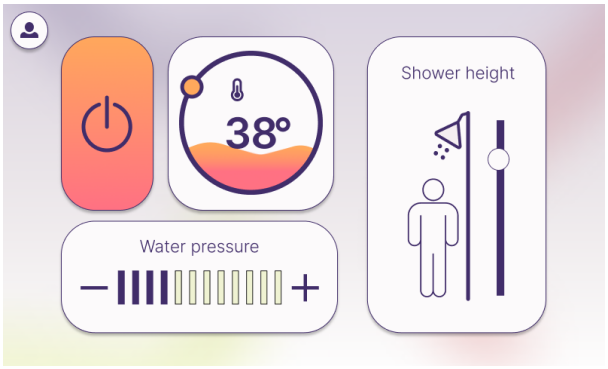


Figure 3.8: The final programmed version of the main page

For the different sliders, the design had to make it clear to the users that these are three separate sliders. Therefore the design accommodates the Law of common region (Yablonski, n.d.). The law of common region states that elements tend to be perceived as grouped if they share an area with a clearly defined boundary. The three sliders all have a white box around them to make this boundary clear for the user.

,

3.5.4 3D rendering

To visualise the finalised design of the prototype a 3D rendering has been created in 3ds Max, as can be seen in Figure 3.9, 3.10 and 3.11. This visualisation is merely a proof of concept and doesn't have the additional features as will be discussed in future work 5.4.3.

Figure 3.9: 3D rendering of concept

Figure 3.10: 3D rendering of concept 2

Figure 3.11: Close up of display screen

3.6 Sub-conclusion

Throughout this chapter, the focus has been on the project's prototype and the iterative process that led to the development of both Lo-Fi and Hi-Fi versions. This

resulted in the final user interface design, which will serve as a foundation for the upcoming section.123

4. Construction

This chapter will go through the prototype, including the mechanical parts that are incorporated, as well as the creation process for the integrated 3D-printed components. Furthermore, the database and Arduino code will be thoroughly explained.

4.1 Acquiring the parts for the prototype

The project requires parts for the prototype, these can be seen in Figure 4.1 and Figure 4.2. To accurately measure the temperature of the water in the shower, a temperature sensor designed to be submerged in water is required for this project. As part of the project concept, a screen displaying the preference settings of the shower and allowing for configuration adjustments was required. To control the water pressure two water pumps are needed, one for the hot water, and the other for the cold water. For the control of the height of the shower head, a servo motor was needed. All these parts were purchased from the website Amazon.de.

(a) The MASUNN Ds18B20 thermometer, (Amazon, n.d.[c])

(b) The Elecrow 5 inch display, (Amazon, n.d.[b])

Figure 4.1: Pictures of the thermometer and the 5-inch display

(a) The Akoczon water pump, (Amazon, n.d.[a])

(b) The Miuzei servo motor, (Amazon, n.d.[d])

Figure 4.2: Pictures of the Water pump and the servo motor

4.2 Programming

In this section, we will elaborate on the various components and discuss how the four key attributes outlined in the MoSCoW model: Temperature, water pressure, height adjustment, and preset settings as mentioned in Section 2.2.3, will be incorporated into the final iteration. Additionally, a detailed examination of the Arduino and its programming to enable these functionalities will be provided.

4.2.1 Coding process

The initial plan was to utilise the Arduino display screen to showcase the user interface of the prototype. However, complications arose during the setup of the screen, prompting a decision to explore alternative options. Instead of proceeding with the display screen, it was determined that a computer screen would be used to present the UI components.

Issues with the Arduino display screen included a multitude of different products, libraries, and packages combined into one overly complex product. This complexity, coupled with inadequate documentation for the Arduino, posed challenges in getting it to function properly. Official tutorials were tailored for different Arduino ESP32 screens, which did not align with the purchased ESP32 screen despite being advertised on the official website.

A significant amount of time was dedicated to establishing a connection between the Arduino and the computer in order to modify and upload the code. The code in itself were not the problem, but the display screen's connection to the computer was. It was ultimately successful only after connecting the Arduino to a power bank for external power, instead of the computer. This solution was not mentioned in any tutorials or forums for the Arduino.

The decision to switch the UI to a computer

As mentioned in the latter Section, the display screen showed more issues the more we got to work with it. Consequently, a decision was reached to replace the Arduino screen with a user interface that will be displayed on a computer screen using HTML, CSS, and Javascript. The method of communication between Arduino and Javascript became a pressing concern when considering the switch in UI display strategies. Fortunately, the integration of the Johnny-Five platform allows for communication between Javascript and microcontrollers such as the Arduino.
Along with Johhny-five, communications from Javascript to Arduino can be accomplished with the Firmata protocol on the Arduino. Firmata makes it so that receiving data from software on the host computer is easily obtained (Arduino, n.d.). The project group used the StandardFirmataPlus which is an example that can be found

in the Arduino IDE. The StandardFirmataPlus code makes the host computer able to receive all calls from written software.

However, the Johnny-five platform requires communication to occur through an internal port of the computer, while the UI will be hosted on a local server. The solution for this would be to integrate Express.js as a communicator between the two systems. However, because of the project timeline, the project group has determined that the most effective approach for creating a working prototype is to split it into two parts: a computer-based UI and a physical prototype to showcase its functionalities.
The final iteration of the code used for the attributes of this project will be presented in the upcoming sections.

4.2.2 Temperature

In order for the prototype to function properly and effectively manage temperature control, a temperature sensor is required to accurately determine the temperature of the water. The project group has chosen to use a digital temperature sensor, for this purpose. The sensor being used can be seen in Figure 4.3

Figure 4.3: picture of the Sensor that is being utilised

The sensor will be positioned after the point where the hot and cold water converge and mix, in order to provide the user with real-time temperature readings, this can be seen in Figure 4.6. In Figure 4.4 the setup for the sensor on the Arduino can be seen.

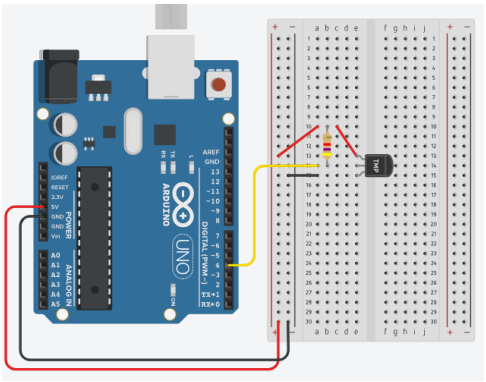


Figure 4.4: The setup and wiring to the Arduino Uno Board

As can be seen in Code .0.1 When wanting to use the thermometer in this project, the library Dallas temperature and OneWire need to be used as this supports the thermometer. Firstly the libraries are implemented with the "#include" statement and the thermometer is initiated with the "#define" statement. In the loop function, the "sensors.requestTemperatures();" function is called that requests the temperature and prints it afterwards with "printTemperature(insideThermometer);".

The "(insidethermometer)" argument is later used to regulate the temperature of the water that will be coming out of the water hose. The variable "targetTemp" is initiated as an empty float to be used in the if-statement. Depending on what button is pressed the "targetTemp" gets assigned the value from one of "targetTemp1" to "targetTemp4" values that has been made from the beginning. These variables are assigned a temperature.
A test with cold water have been made and can be seen in Figure 4.5. A cup was filled with bottled water from the fridge, and the temperature sensor was submerged into it, and displayed the accurate temperature for the container in the code terminal.

(a) Shows testing of the temperature sensor with cold water

```
Temp C: 16.00Requesting temperatures...DONE
Temp C: 15.50Requesting temperatures...DONE
Temp C: 14.50Requesting temperatures...DONE
Temp C: 14.00Requesting temperatures...DONE
Temp C: 14.00Requesting temperatures...DONE
Temp C: 13.50Requesting temperatures...DONE
Temp C: 13.00Requesting temperatures...DONE
Temp C: 13.00Requesting temperatures...DONE
Temp C: 12.50Requesting temperatures...DONE
Temp C: 12.50Requesting temperatures...DONE
Temp C: 12.00Requesting temperatures...DONE
Temp C: 12.00Requesting temperatures...DONE
Temp C: 12.00Requesting temperatures...DONE
Temp C: 11.50Requesting temperatures...DONE
```

(b) Results from the thermometer

Figure 4.5

4.2.3 Water pressure

One of the primary functions of a shower is to control the flow of water and adjust the water pressure from the shower head. In order to replicate the water pressure from an external source, a water pump, as mentioned in Section 4.1, will be utilised to facilitate the water pressure.
Incorporating the configuration outlined in Figure 4.6, two water pumps will be strategically positioned - one at the hot water source and one at the cold water source. The pumps will be submerged in the water to effectively pump water. This setup will enable precise control over both water pressure and temperature.

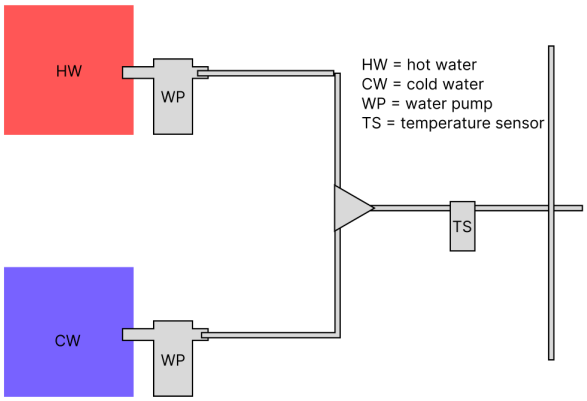


Figure 4.6: Diagram of setup

As for the Arduino setup, Figure 4.7 shows how the two water pumps displayed connected with an external power source in this figure shown as 6V of power but in practice will be 12V. The four buttons are shown to imitate the four profiles that can be selected. The MOSFET regulates the amount of power sent to the pumps to control the water pressure. The MOSFETS are connected to pins 10 and 11. This is specifically done because they Support PWM which allows for the regulation of power flow.

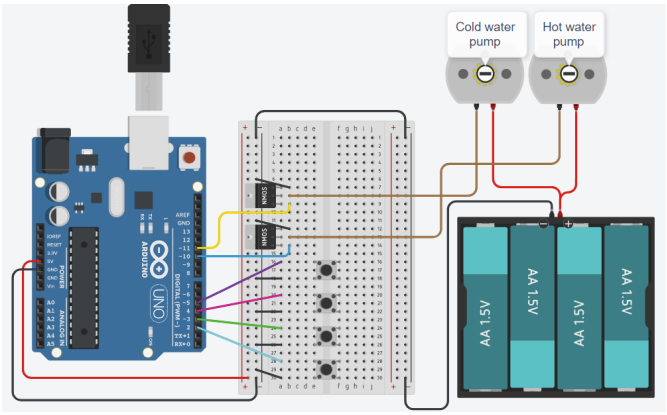


Figure 4.7: Arduino diagram of water pressure

As can be seen in Code .0.1, the water pressure is calculated together with the temperature of the water. The water pressure as can be seen in the code, goes from 0 to 255 as this is the value that the Arduino ranges from. The float "Ratio" is going to be used for calculating the speed of the two pumps. To assign a value to "Ratio", it starts out with "(Targettemp - coldTemp)", and then minuses "coldTemp" to "hotTemp" which is 45 degrees Celsius. After doing so, the "(Targettemp - coldTemp)" gets divided by 45. This ensures a decimal in between 0 and 1 that is assigned to the "ratio".

4.2.4 Height adjustment

This setup was made with five buttons. The five buttons are supposed to represent four profiles and a reset button so that when a button is pressed, the preset height for a profile will be initiated. Each button is connected to the same servo motor, which is then locked in the desired position as can be seen in Figure 4.8.

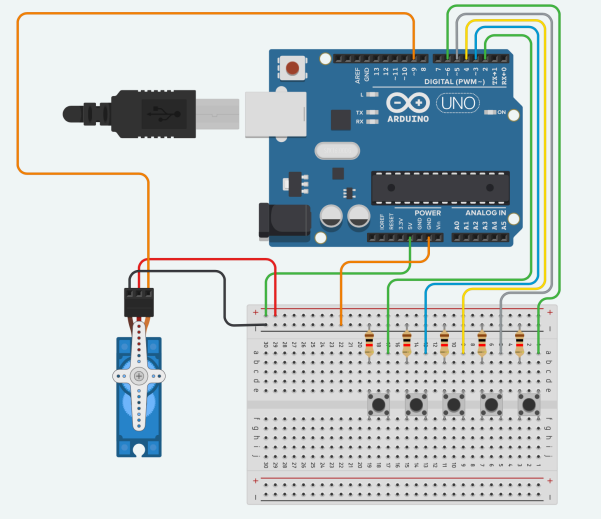


Figure 4.8: Circuit diagram of height adjustment

As can be seen in Code .0.1, To be able to use the servo library, it will have to be included in the code with "#include <Servo.h>". The servo is initiated with the "Servo myservo;". This enables the use of "myservo.write();" which calls the servo and rotate the value(degrees) in the parentheses. Within the library, the range of motion is limited to 180 degrees, which is less than the 270 degrees supported by the servo. This may restrict the ability to demonstrate optimal adjustments, but the functionality remains consistent with that of the final prototype.

4.2.5 Profile presets

This Section will discuss the initial iteration of the code being developed for managing profiles and individual preferences for a shower experience. Since the Arduino screen got scrapped, the code will be running on a computer. This means the code went from C++ to JavaScript.

Initial code

For the initial concept of the prototype, the plan was to have the prototype run on an Arduino display and communicate with Arduino boards that control various functions of the shower. This involved the development of a database management system using C++ for the Arduino boards. To gain a better understanding of how the code will be structured, a pseudo-code has been created as seen in Figure 4.9. It is seen that it is split into three sections. The display showcases various attributes of a profile, including ID, height, temperature, water pressure and name. The View component accesses profile data by utilising getter methods based on the profile ID. The controller allows users to modify and set different attributes as needed.

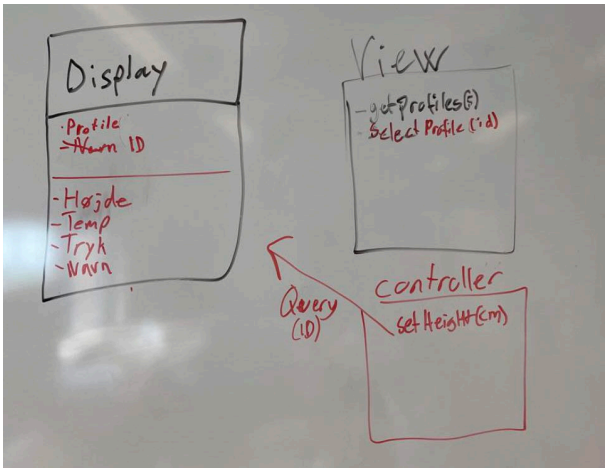


Figure 4.9: Pseudo code for understanding code structure

The C++ code that first got developed was a version where preset profiles were created and could be pulled with the corresponding ID as seen in Code .0.2.

Selecting profiles

As seen in the Code .0.4 we display how we utilise existing profiles, restricting users from adding new profiles and instead requiring them to utilise the preset profiles. In Code .0.3 it is shown that a class called "profile" has been created, and specific variables and their data types are being defined. A constructor is created to set up the profiles with profile ID, height, temperature, water pressure and the name the profile belongs to. Then setter method is created to set values for the profile and the corresponding getter method to pull the data.
In Code .0.4 the main function is portrayed. Two profiles have been preset, with all the variables. The method allows for the user to input a profile ID, that ID will go through a loop checking for corresponding values of height, temperature, water pressure and name and print that data. If the ID is not found, an error message will occur saying that profile x does not exist.

The code used for the project

The Code went back to the version with presets, due to complications with the Arduino display. This meant that the UI would be displayed on a computer screen instead of the Arduino display. Since the code got altered to fit the solution without the display, the decision to make presets for profiles was made. The Javascript code for the different aspects of the UI can be seen in Code .0.6.

4.3 Physical construction

This Section will delve into the physical construction of the prototype's features and its iterations.

4.3.1 Height adjustment

After the MoSCoW model in Section 2.2.3, the height adjustment became a "Must have" requirement for the prototype. The group therefore started iterating on how this could be made. A brainstorming process, which ended up in the sketches seen in Figure 4.10.

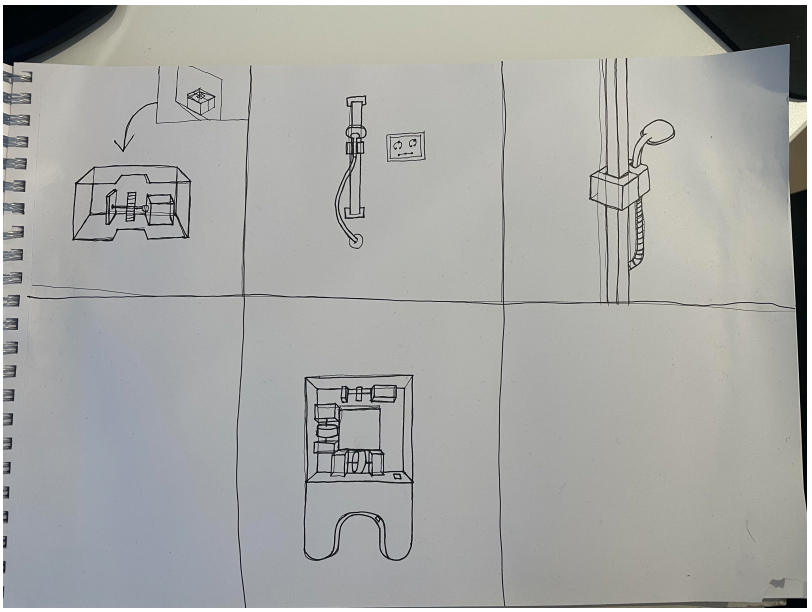
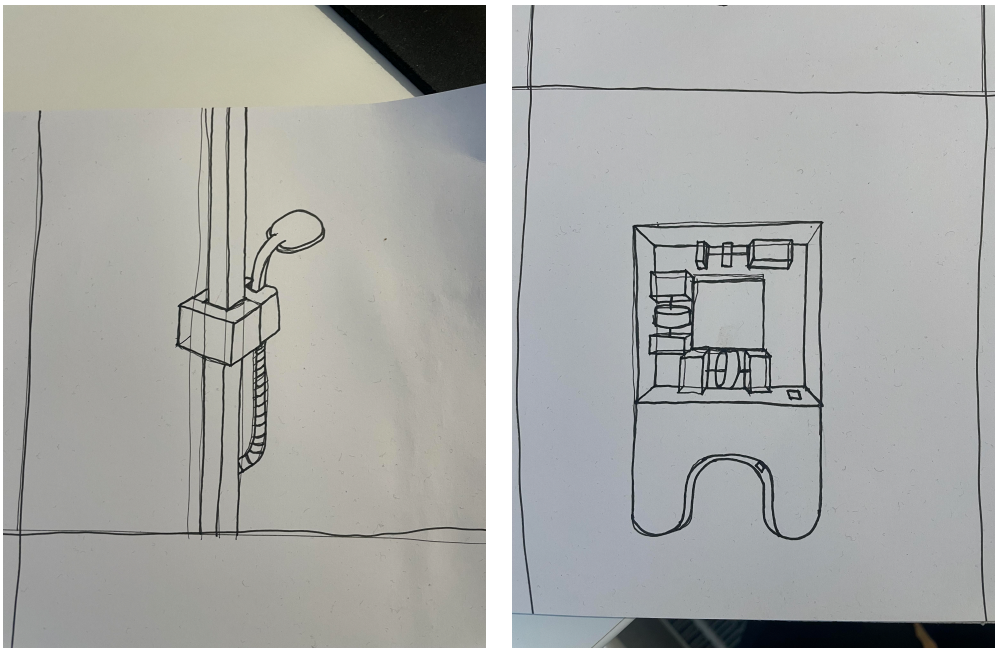


Figure 4.10: Different sketches of height adjustment

Figure 4.11a shows how the shower head is mounted on a box that slides up and down on a vertical pole. The project group were discussing whether it should be the shower head moving or the whole vertical pole moving up and down. For aesthetic looks the project group decided to go with the version where the box needed to be controlled by a servo motor.

The decision about using a servo motor ended up in the sketch shown in Figure 4.11b. This figure shows how the servo is mounted opposite of the shower head holder. The project group had the idea of having two support wheels placed on the opposite side of the servo and adjacent to it.



(a) Sketch of height adjustment (b) Sketch of inside the box

Figure 4.11

3D print creation

The box sketched in the previous Section 4.3.1 will be modelled in the 3D program called 3ds Max. Afterwards, the box can be 3D printed in plastic. The wires can be hidden inside the box and run alongside the shower hose to the Arduino.
In Figure 4.12, the placement of the servo and the support wheels within the box, are depicted. Prior to advancing with the prototype, various test prints will be conducted to assess feasibility. Additionally, the dimensions of the print must align with the size of the servos and wheels, which will be determined after observing the physical print.

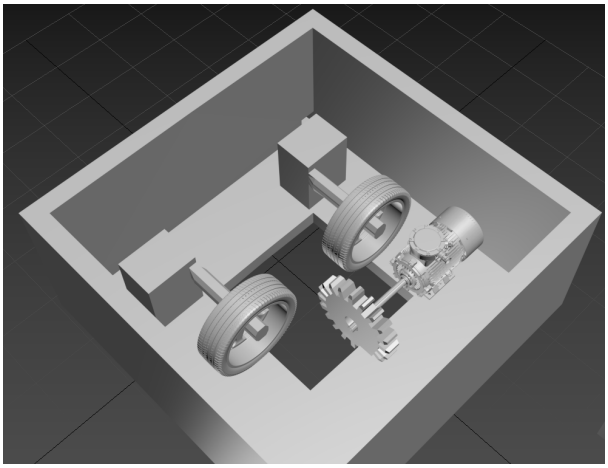
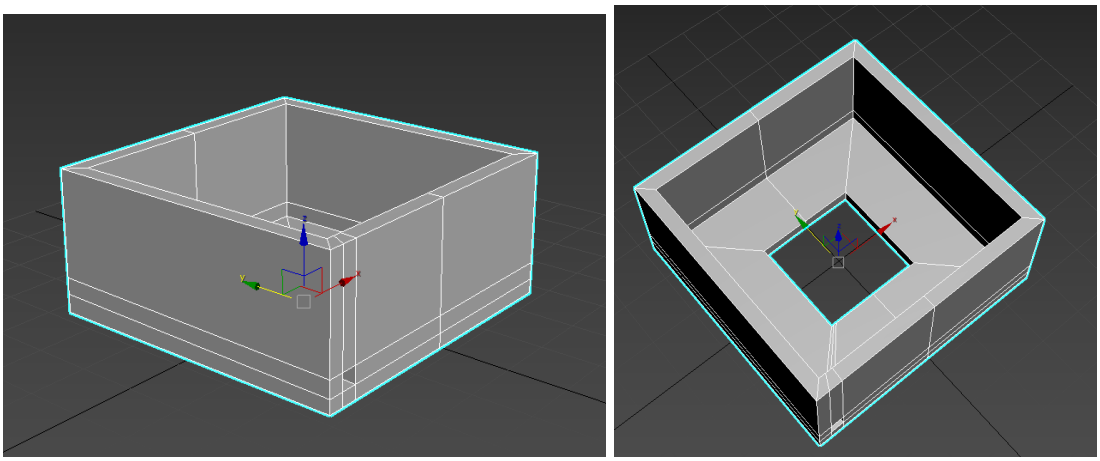


Figure 4.12: Visualisation of how the parts will fit

First iteration

In the first iteration of the 3D print, the box is comprised of two elements: a designated hole for the vertical pole intended for holding the box to the shower head, and small holes located in the bottom left and right corner on one of the sides of the box for wire passage, as seen in Figure 4.13 a and b.



(a) (b)

Figure 4.13: (a) First iteration 3D box, (b) First iteration 3D box from above

In this iteration of the box, a lid has been implemented to securely close the box and conceal the wires and servo motor. The box features a hollow border where the lid will fit, with a reversed segment that securely clicks into place. This design eliminates the need for glueing the box together, providing easier access when needed, during the construction of the prototype, as can be seen in Figure 4.14. The box will be printed in a smaller version than the original, as to test if the lid's clicking mechanic would work with the box. The reason behind this is simply because of the time it takes to print the fully sized box, compared to the smaller version. Additionally, the

logistics of utilising the 3D printer during the project timeline presented challenges due to the limited availability of the printers. Therefore, it was crucial to ensure that the closing mechanism of the box was fully functional before proceeding with printing the full-sized version.

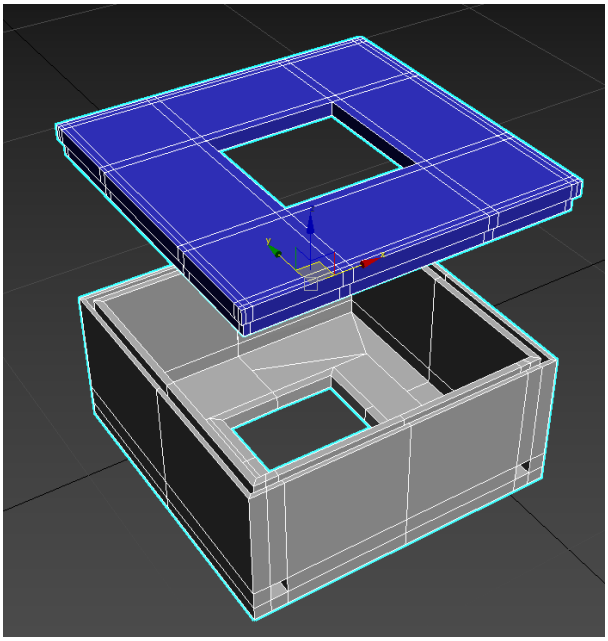
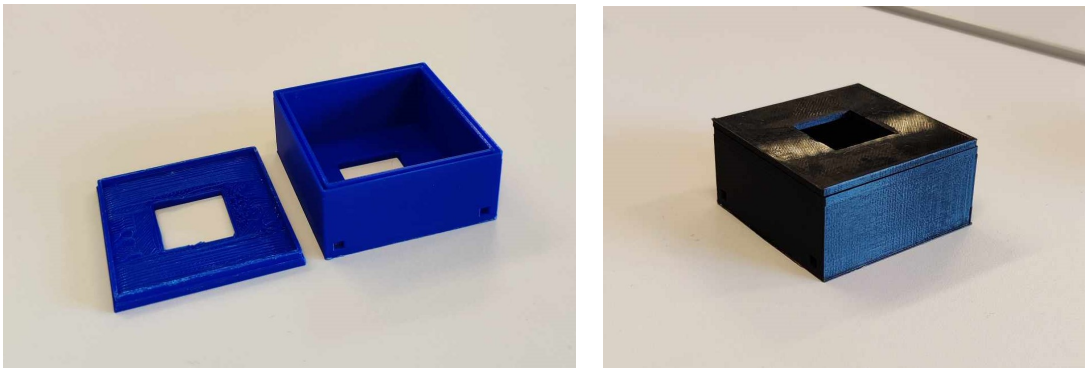


Figure 4.14: First iteration 3D before printing

After the first iteration was printed, a few mistakes had been made. Firstly the hollow border on the box, where the lid needs to click into place, were too narrow and couldn't fit unless forcefully pressed. Furthermore in the setup file, prior to initiating the printing process for the lid, it was incorrectly positioned on the print plate. The narrow borders that clicked into place on the box was oriented downwards. As a result, the lid did not receive optimal support and subsequently misprinted, as can be seen in Figure 4.15a. For the next iteration, the hollow border on the box would need to be properly fitted to the lid and more considerations to how the box would be printed.

After successfully re-printing the box, it was determined that proceeding with creating a full-sized model was feasible, as the lid functioned properly, as can be seen in Figure 4.15b.



(a) First iteration printed (b) Re-print of the first iteration

Figure 4.15

Prior to advancing to the next iteration of development for the box, an overview of the servo and support wheels that will be integrated into the box will be provided.

Servo construction

The servo used for this project have a rotation angle of 270 degrees, which meant that the total amount of cm the servo could move the vertical pole was limited. For increasing this distance the project group made the servo iteration as shown in Figure 4.16. This mechanism was created using a small gear connected to a larger gear. The small gear interfaced with the servo gear, thereby enabling the grey gear to achieve a range of motion exceeding the default 270 degrees.

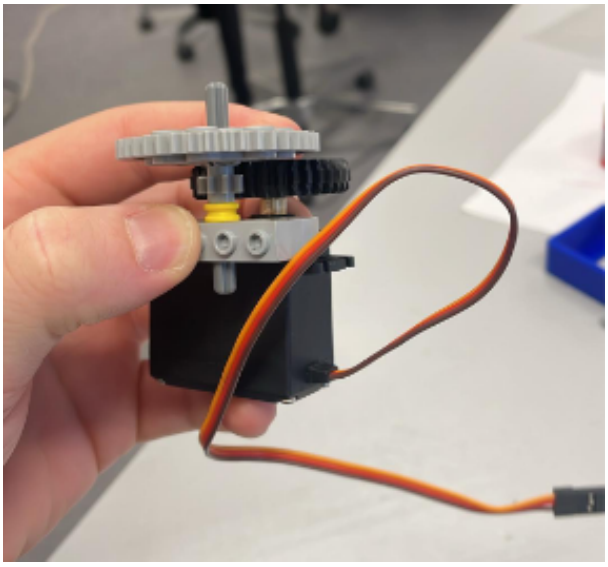
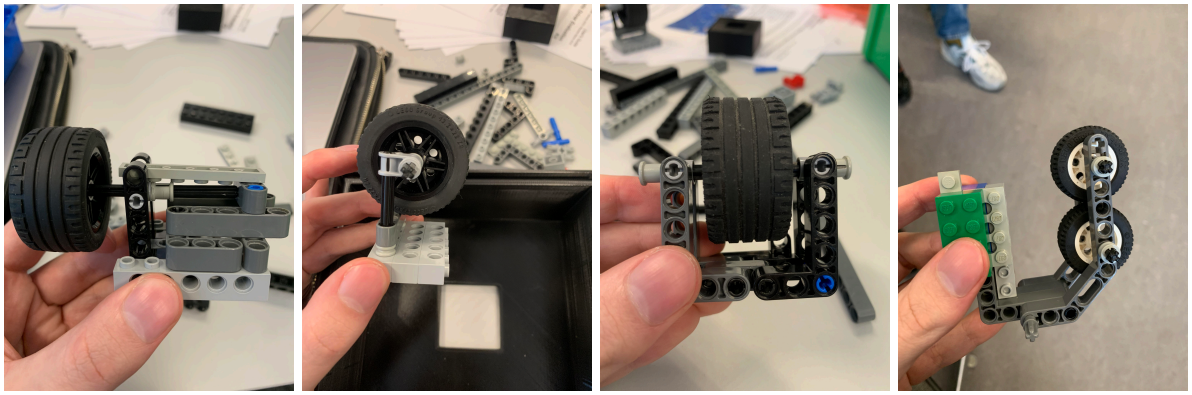


Figure 4.16: Servo with Lego gear

Support wheel construction

As an idea for stabilisation of the servo motor, the project group decided to have a support wheel on the opposite side of the vertical pole to counteract the weight that the servo will place on it. This wheel was built with a Lego Technic Wheel and extra Lego parts.
The first iteration of the wheel is shown in Figure 4.17a. The project group iterated further to make it more compact.
The second iteration of the support wheel is shown in Figure 4.17b. This iteration was made to be more compact than the previous, with the wheel placed in between two Lego axle shaft rods, instead of the wheel support on the side of the wheel for more optimal space usage.
However, this wheel was unstable when pressure was applied to it. Therefore it was reinforced with more Lego bricks. The next iteration is shown in Figure 4.17c.
The group had initially concentrated on creating support wheels using a large Technic wheel. After some consideration, it was decided to experiment with making support wheels using a pair of smaller wheels, as seen in Figure 4.17d. The goal was to enhance the wheel's ability to apply pressure across a larger surface area. Unfortunately, this design proved to be ineffective as the smaller wheels elevated the structure, requiring unnecessary space.



(a) First wheel iteration (b) Second wheel iteration (c) Third wheel iteration (d) Fourth wheel iteration

Figure 4.17

Second iteration

After printing the box, it became evident that it was larger than necessary for the final design. This sizing was intentional, as it would allow for testing how the various components and parts would fit inside. In Figure 4.18, a visual depiction of how the box stores the parts can be seen.

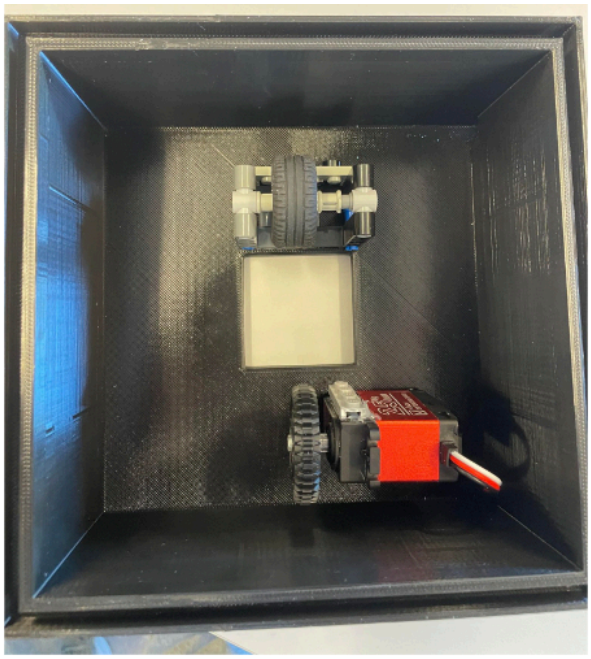


Figure 4.18: Box with parts inside

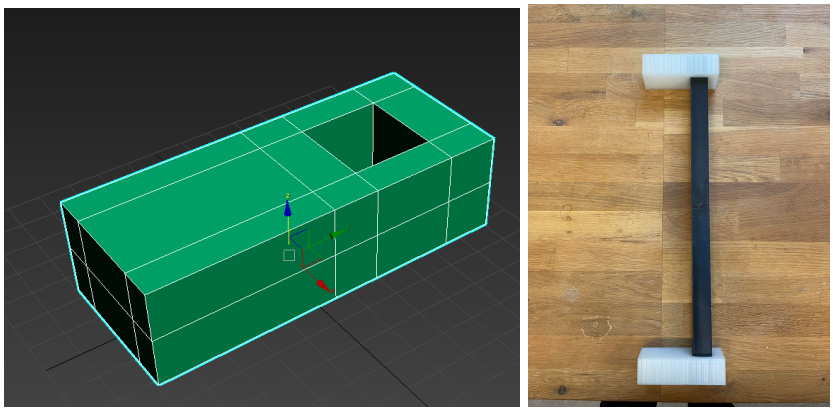
3D gear rack

The vertical pole will be made into a 3D gear rack properly fitting the gear on the servo motor, which will control the height of the shower. The servo gear will be adjacent to the gear rack, which will translate rotation of the gear into vertical movement on the gear rack. The gear rack will hold the box, which will hold the shower head. This setup can be seen in Figure 4.19.

Figure 4.19: Vertical adjustment gear rack

Additionally, a wall mount component will be designed in 3ds Max to securely affix the gear rack to the completed prototype. This component will be 3D printed, as

illustrated in Figure 4.20a. This component must ensure there is a six cm gap between the wall and the hole in the box. This clearance is necessary for the proper functioning of the height adjustment system, allowing it to move vertically without hitting the wall. The component will be affixed to the wall of the prototype structure, followed by securing it to the gear rack, which can be seen in Figure 4.20b.



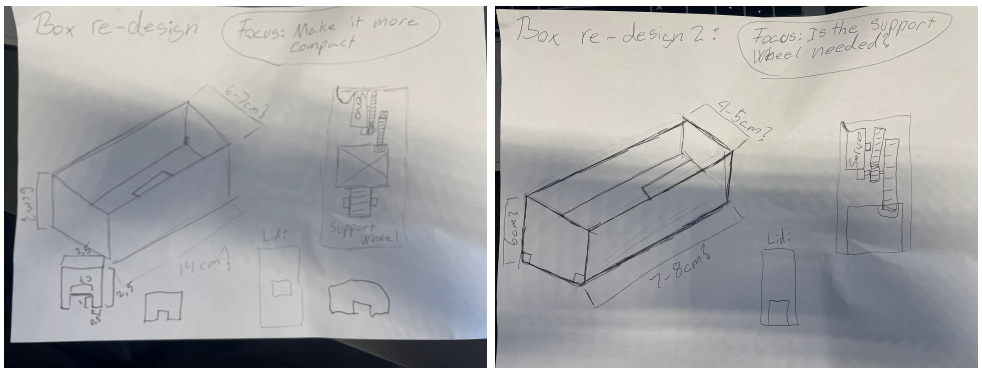
(a) Wall mount component

(b) Wall mount component printed

Figure 4.20

Re-design of box-print

After successfully printing the larger-sized box in which the servo motor would be placed, the project group can begin to look for potential improvements. The design had excess empty space, as the only necessary components were the servo motor and a support wheel. From this, the project group created another iteration, as seen in Figure 4.21a.
In Figure 4.21a, there was made a sketch of how the box would look with the support wheel. The project group explored other options as where a support wheel is not needed. The sketch of this iteration can be seen in Figure 4.21b, to try to improve the space usage within the box even further.

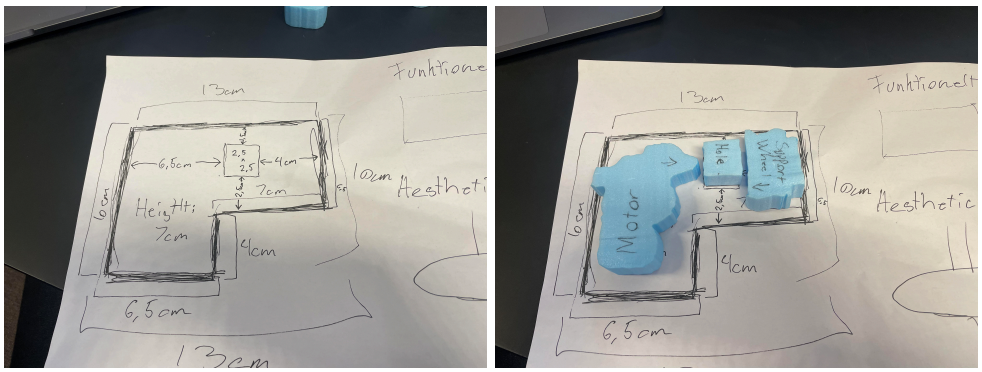


(a) Box with less space

(b) An iteration of the box without the support wheel

Figure 4.21

After deciding to include the support wheel, we revisited the previous design of the box and optimised the utilisation of space to accommodate all parts accordingly, as seen in Figure 4.22 a. The project group created wirecuts based on the dimensions of the servo and the support wheel. This allowed us to accurately visualise and position the components, enabling us to create a detailed drawing with precise measurements and gain a better understanding of the scale, as seen in Figure 4.22 b.



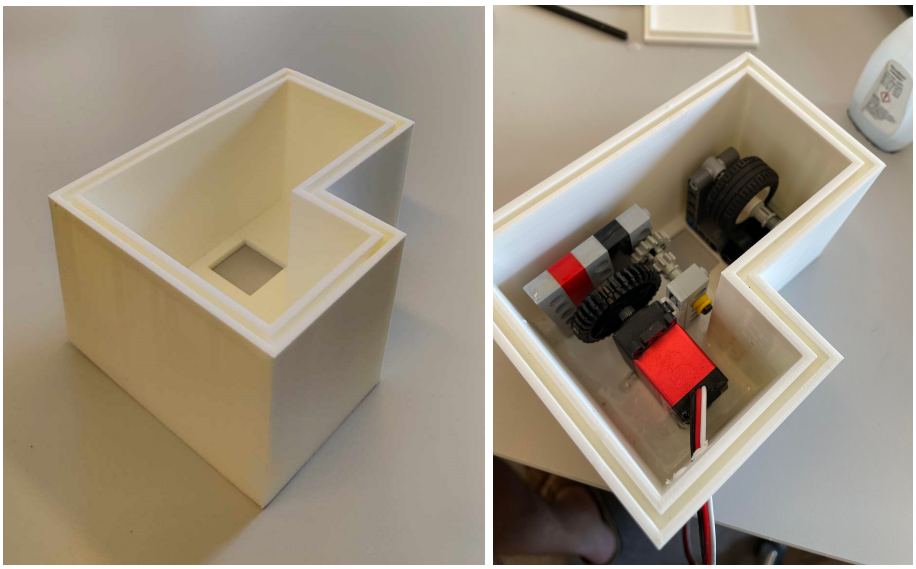
(a) Fourth Iteration sketched

(b) Wirecuts used for measuring components in the box

Figure 4.22

Final iteration

After modelling the box in 3ds Max with the measurements given by the sketch, the box was printed, as can be seen in Figure 4.23. Next the gear rack will be implemented, to adjust the height.

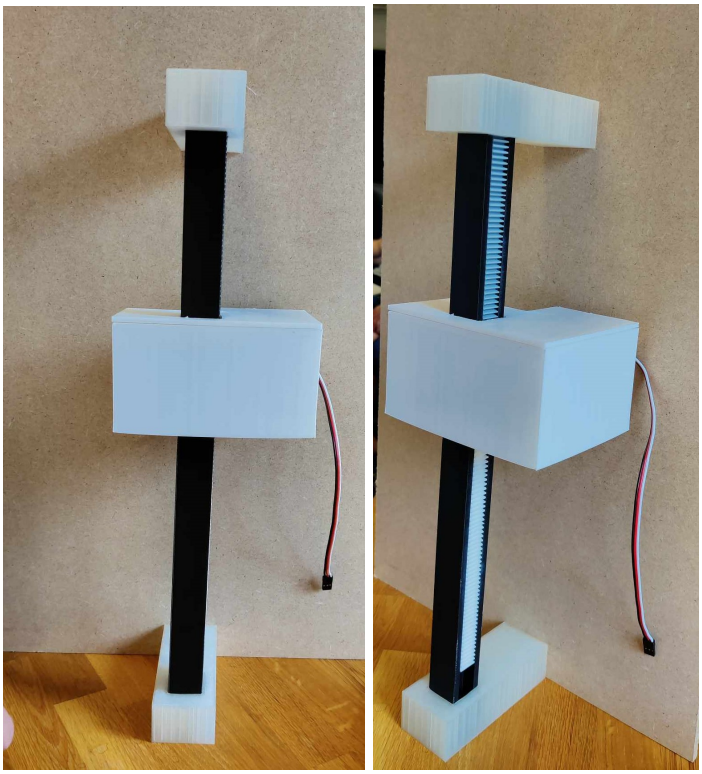


(a) Final Iteration box print

(b) Iteration with components inside

Figure 4.23

The project group experienced a problem with the servo mechanism. The servo utilised in this project is rated for a torque of 25 kilograms, however, since we attached another gear this power was decreased. The servo could not lift the box when vertically aligned, because of its own weight, as seen in Figure 4.24.



(a) Height adjustment segment

(b) Different angle of segment

Figure 4.24

The solution to the servo issue was to remove the extra gear on the servo. This resulted in a significantly smaller range of motion but increased the torque for the servo.

4.4 Sub-conclusion

After facing obstacles with the Arduino display screen and making adjustments to the projects direction regarding the UI, the prototype showed proof of concept for the customised shower experience. The assembly of the various 3D components was linked to the purchased parts and connected with the Arduino to enable the pumps, servos, and thermometer to operate. The final prototype of the shower can now be observed in Figure 4.25.

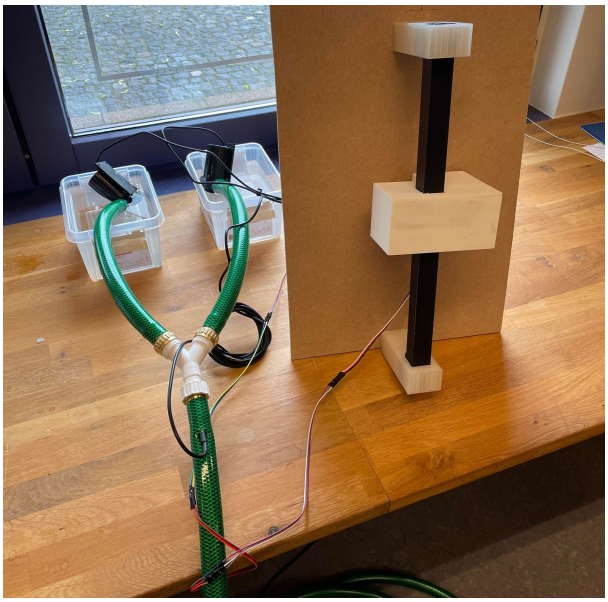


Figure 4.25: Final iteration of prototype

5. Discussion

In this chapter, we will examine how the competitors of other shower systems and the prototype developed by the project group align with the elements of personalisation and customisation discussed in Section 2.1. Additionally, the limitations encountered during the development of the prototype will be addressed. Suggestions and ideas for future work will also be provided.

5.1 Personalisation and customisation

When evaluating related work and the project prototype in comparison to the concepts of Personalisation and Customisation, as mentioned in Section 2.1, it becomes clear that certain products outperform others in meeting these principles.
In Section 1 the project mentions similar solutions to the project thesis. These solutions were all made by the company, Kohler. It would therefore be interesting to evaluate and discuss how the related work matches the principles of personalisation and customisation, and how our prototype matches or differs from these alternatives.

Regarding the concepts of customisation and personalisation, the project group aimed to incorporate these principles into the prototype developed throughout the project. While the prototype draws inspiration from two Kohler products, it also showcases unique attributes. The prototype have the preset functionality just like both the DTV+ and Anthem, and also makes it possible to adjust the shower's water pressure and temperature through the display. The Kohler products, especially the Anthem, have functions that the prototype does not have, such as music and steam. The project's prototype also differs in features, such as the implementation of the self-adjusting shower head, which is not found in any of the Kohler products. When focusing on the principles of customisation and personalisation, the prototype is very customisable. The project's prototype does have a similar concept as the Kohler products since it is more a customisable product than a personalised one. Solutions for how to balance these two concepts will be mentioned later in future work.

The first product mentioned in Section 1 was the DTV+, a digital control with a configurable colour touch screen. According to the article "Customization vs Personalization" by NN Group, (schade, n.d.), personalisation involves the system knowing the users preferences by their actions such as search or purchase history. In the DTV+ the only thing close to such feature is that it is possible to have up to six user-definable presets for saving the users favorite settings. However this is more of a customisable feature, since the user makes the adjustments themselves. The DTV+ have five different customisable features where each can be set to one user-specific desire by the presets. If features like spa, lightning or music have a memory function, from last time a bath was taken by the user, the DTV+ is not only customisable but also per-

sonalised. But information about this could not be located on the website.
Anthem is the second product mentioned in Section 1, a digital controller with sets of sensoric shower experiences. As mentioned with the DTV+, the Anthem also have more customisable features than personalised features. However, the Anthem have more customisable features than the DTV+, since the Anthem can control both water temperature and water pressure. As mentioned in Section 1, the anthem includes more sensoric experiences and built-in modes. From the website it could not be concluded, but since these modes had names like: Wake up, sleep simple and Cool down, it could have a functionality of remembering patterns of the user. According to the article from The NN Group, this would be an example of personalisation, since the device adapts to the users needs and therefore makes their experience more unique (schade, n.d.).

5.2 Displays in wet environments

One potential challenge that may arise during the implementation of the prototype is that touch screens may not function optimally when users have wet hands. This issue has been extensively debated within the group, leading to the exploration of several potential solutions. One suggestion was to position the screen behind the shower, so that water would not hit the screen. The user would still have wet hands, but this could still improve the usage of the screen. Alternatively, incorporating a touch screen with a water repellent surface could also address this issue. Furthermore the users may not interact with the screen after stepping in the shower, as they already have their preferred temperature, pressure and height settings set to their individual preset. Therefore this may not be a problem at all. To be sure whether this would be a defining problem for the project, user tests would have to be run. This could uncover any potential issues relating to the display screen and wet hands.

5.3 Target demographic

Throughout this project, identifying the target demographic has proven to be challenging due to its broad nature and applicability to a wide range of individuals. The initial concept of the shower was to be customised to meet the unique needs of each user, thus appealing to a diverse audience. Ultimately, the target demographic was determined to be individuals living in shared households or a collective where one bathroom is shared, this is a category that encompasses a wide range of users. The project group deliberated on potential target groups and concluded that families could benefit from the prototype. Especially considering the varying height requirements and other adjustments often needed in households with children. After a specific target demographic is defined, user testing would be necessary to evaluate the user interface's usability for families, potentially having the need for a child-friendly mode. An explanation of the user testing process will be provided later.

5.4 Limitations

In this chapter, we will highlight the limitations encountered during the project, and explore potential ideas for future work and tasks that could not be completed within the project's timeline.

5.4.1 Arduino display screen

The decision to use a touch display screen for the prototype proved problematic due to compatibility issues with the ESP32 Arduino display. Limited documentation and tutorials for the specific model further complicated troubleshooting efforts, resulting in significant time usage attempting to make the display screen work, but the effort did not prove to be worthwhile. After assessing the project group's limited experience with implementing a subpar Arduino unit, it was found to be excessively time-consuming and challenging. While theoretically suitable for the project, its complexity far exceeded what was necessary for the task at hand and was therefore not suitable for the project's scope.

5.4.2 Conversion of code for the Arduino

While members of the project group were troubleshooting the functionality of the Arduino display screen, other group members were focused on establishing a profile database for the user interface. Unfortunately, this task proved to be time-consuming, only to be abandoned when the decision was made to forego the display screen concept. Consequently, the code initially written in the Arduino programming language C++ had to be converted to Javascript in order to accommodate the user interface being displayed on a computer screen instead.

5.4.3 Future work

In order to assess the viability of the prototype as a product, user testing would be necessary. One potential approach could involve deploying the prototype in real-life settings with the demographic mentioned in Section 5.3. Participants would be asked to use the prototype for a designated period of time, keeping a detailed log of their experiences including ease of use, time saved, and overall satisfaction. As it is difficult to ethically record bathroom usage, a personal log book for each participant would be more ideal than other forms of observation in the bathroom. This data would then be used in a follow-up interview, during which participants' feedback would be discussed and incorporated into potential modifications to the prototype.

Additional features could be added such as a sensor for automatic adjustment of height instead of manual input, mood lightning, a waterproof touch display screen and other features discussed in Section 2.2.3. For households with children, the user interface could be tailored to be more child-friendly by simplifying buttons and

incorporating a more playful and colourful design. As the shower currently only offers customisation options without personalisation features, as explained in Section 5.1, a possible solution to this could be a self-adjusting temperature functionality. A program could be developed to track user preferences for temperature adjustments during showers and automatically adjust the temperature accordingly for the chosen user profile, the next time that user showers.

First thing to work on, would be getting the touch display screen to function and the missing UI options, such as creating a profile and setting up the presets. A suggestion of how the profile setup UI could look like, could be the following: The user can either choose their profile or set up a new profile as can be seen on Figure 5.1. In Step 1 as seen on Figure 5.2, the user then chooses a profile picture from the different illustrations. Then in Step 2 as can be seen on Figure 5.3, a picture has been chosen and the user now has the option to either save and continue or choose another picture. In Step 3 as can be seen on Figure 5.4, a keyboard appears, where the user can enter their name for the profile and save it. Lastly in Figure 5.5, the user now enters the setup page, where the height, temperature and water pressure can be set and saved for future shower use.

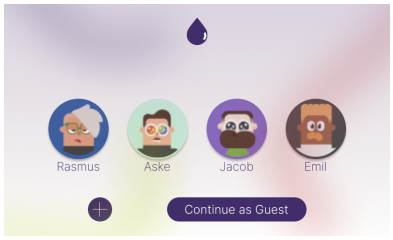


Figure 5.1: Choose Profile

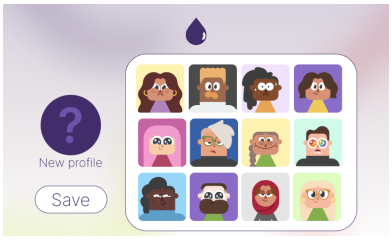


Figure 5.2: New Profile Step 1

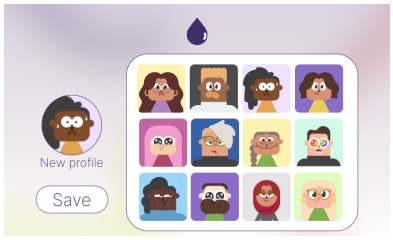


Figure 5.3: New Profile Step 2

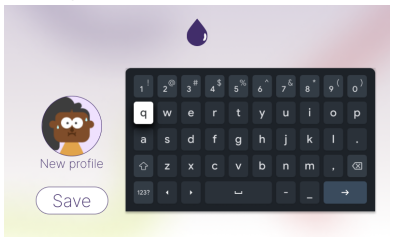


Figure 5.4: New Profile Step 3

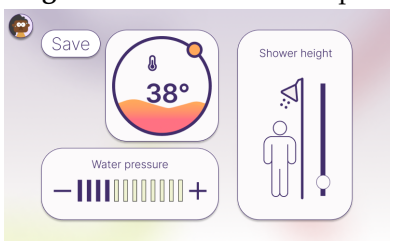


Figure 5.5: New Profile Step 4

Still some UI is missing, although the code has been written in Code .0.5. Within the project's timeline some complications with the code occurred and it was decided that implementing the UI for creating a profile, had to be postponed for future work. Currently, the prototype utilises a garden hose as a substitute for a traditional shower head and hose. This choice was made for logistical reasons, as purchasing a shower head would be an unnecessary expense and not essential to the overall concept of the shower prototype. Additionally, the incorporation of a shower head is not a customisable feature that aligns with the project objectives.
Furthermore, the practicality of integrating a shower head poses challenges, as it would add additional weight to the servo system already strained by the existing

components. If the prototype were to be developed into a marketable product, a shower head would need to be included and modifications made to support its installation. This would involve creating a 3D printed part to securely hold the shower head in place, allowing for the concealment of wiring and other potential mechanics needed for a more aesthetically pleasing design.

5.5 What did we learn?

As previously discussed in this chapter, this project has faced numerous challenges and obstacles, largely due to a lack of experience in physical prototyping in this manner. The group's basic knowledge with electronic devices proved challenging, especially when working with different components, that needed to fit together. Despite these challenges, this project served as a valuable learning opportunity, allowing the group to progress from minimal knowledge of basic electronics to a more comprehensive understanding.
The iterative nature of this project was more pronounced than in previous projects. It was a learning experience to have expectations and plans in place, only to realise that they were not feasible. Throughout the project, there was a continual discrepancy between expectations and the reality of what could be achieved. That reality was often something out of our control, like software not working as intended and misleading spec sheets with overall bad documentation. Throughout the course of understanding the functionality of the Arduino board in conjunction with other project components, various solutions were experimented with, but proven unsuccessful. This involved the use of the Johnny-Five framework and other software solutions that theoretically should have been compatible but were not supported for our specific project requirements. In conclusion, we discovered that successfully integrating these components proved to be challenging without prior knowledge of their functionality.

5.6 What would we do differently?

As the iterative process became more prominent in this project, the group should have been more adept at adjusting their expectations and approach when faced with challenges, which surfaced along the process. Our expectations should have been adaptable to the challenges that arose during the project. While our initial goals may have been sufficiently ambitious compared to our experience, it is important for the project to be flexible and adjust as unforeseen obstacles arise.

6. Conclusion

The problem statement for this project is:

How can we develop a physical prototype that modernises the experience of taking a shower?

Upon reviewing existing solutions for enhancing the shower experience, it was concluded that personalisation and customisation could potentially contribute to modernising a shower. With knowledge of Kohlers products, as discussed in Section 1, our prototype was inspired by their concepts. By integrating individual user preferences through a UI on a screen, there is potential to enhance the traditional, non-digitalised shower experience. This approach represents a promising step towards modernising the bathroom environment.

The shower prototype developed in this project focuses on essential attributes of a shower, such as height adjustment, water pressure, and temperature. These attributes have been integrated into a UI that allows for individual user profiles with custom presets. This concept aims to modernise the shower experience through customisation and personalisation.
As discussed in Section 1, modernising a product could involve integrating customisation and personalisation. This allows the product to be tailored to each individual user. The prototype demonstrates potential for offering a modernised experience, through personalisation and customisation, as its features are intended to meet this criteria. Though personalisation would come as future work, as the timeline for the project would not allow for extra features as such, this is discussed in Section 5.4.3. In order to confirm this conclusively, user testing would be necessary to evaluate the prototype's ability to achieve this goal.
Despite various setbacks during the development process, the prototype remains incomplete. However, this presents opportunities for further exploration and improvement.

Bibliography

Amazon (n.d.[a]). "Akozon Water Pump". In: *Last visited 21-05-2024* (). URL: https://www.amazon.de/-/en/Akozon-Circulation-Brushless-Temperature-Resistance/dp/B07GT1DNXL/ref=sr_1_73?crid=2FR7AFM95BAOM&dib=eyJ2IjoiMSJ9.9oFb7P8w-wZGcCyMYohOdenI0R22JgXuK_6zxZahHtCsMjuZTwHxaNSBfrLx-inqm4B4o9OU3rH7LBlBfzVAUTTtqULo99Nc1EJzWoNrnxvh09AZis_jobxxOisrOgJ8b0NTZRcb3im1gxFxiZCILOpTPGz6akczF5nC-77tDuDeTkoNSAKr5cqvryOuDDvl-IVXgOuLdy3lXlh8gPLdAPvSGqUnFbCH53UdUZVByTfUuyndedxNSljc7KE.MhY52M81FhOrF-nZ5lkI63gwKoHZ0_---oI0a8awa9A&dib_tag=se&keywords=arduino+water+pump&qid=1710329530&sprefix=arduino+water+%2Caps%2C101&sr=8-73.

— (n.d.[b]). "ESP32 Display, 5.0 Inch". In: *Last visited 21-05-2024* (). URL: https://www.amazon.de/-/en/Display-Touchscreen-Monitor-Compatible-Arduino/dp/B0CD1H72HJ/ref=sr_1_3?crid=2EOIK5B83J7LJ&dib=eyJ2IjoiMSJ9.vFd_C3KHVwflIJVsHDtLu3rJX5sQZyhF15kNrP5ceuuRNHBhCEp8ZWujPIYKIKn-SLjsa0WdRs04s3QvsoPfNMZMyJZj7pmdmRrkidSijnBjrRj62XcsXFdGe9iqz3DHidh3ZSNXAXl5eS3M49Id7P3bMNrRV_Lgks8BTEb1zVeOgtPPSDktXD95i5FobVWWNqO0D53J3M3XsDCRIDbypzSL_deQtGmc7Cz8v470sU.ppptMrLfjRnV4I0e-OGcnHI7D52WggfaGWyg5g-XInU&dib_tag=se&keywords=elecrow+5+inch+display&qid=1716267958&sprefix=elecrow+5+inch+display%2Caps%2C103&sr=8-3.

— (n.d.[c]). "MASUNN Ds18B20 thermometer". In: *Last visited 21-05-2024* (). URL: https://www.amazon.de/-/en/MASUNN-Ds18B20-Digital-Temperature-Waterproof/dp/B07QFNWH6Y/ref=sr_1_6?crid=3W436KD6IEJYD&dib=eyJ2IjoiMSJ9.-iqEynl4bmx_7jbiOrXU26Iv2xeg8tPJXr7B2VCCqBhLyLV6aFZFEJHodHfNW71Gp2w2mjfuhWrqGNvmLJVG0NbkfVDXYG-2DTTyGaFaZIStxgGwRceaMR8hXuXmiQN56hxMDY77jLndI249eC9ENg6sP3PKYm-CFcovKG2qN0_rasiZn-Io.-lIuGTtcZzz43OpfAeApWqxQbACqvO4v8xDBV1pDLwo&dib_tag=se&keywords=DS18B20%2BTemperatur%2BSensor%2B%E2%80%93%2B2%2BMeter&qid=1710328801&sprefix=ds18b20%2Btemperatur%2Bsensor%2B2%2Bmeter%2Caps%2C154&sr=8-6&th=1.

— (n.d.[d]). "Miuzei Servo Motor". In: *Last visited 21-05-2024* (). URL: https://www.amazon.de/-/en/Digital-Waterproof-Arduino-Aeroplane-Steering/dp/B0BZ4DFHRR/ref=sr_1_7?crid=3NHGHOQEXEFTL&dib=eyJ2IjoiMSJ9.epF-rs6SB-8vDTM9WDUCkx5z-i4znTKNyaPoPpT444xxRtDm4xjQyQBfd2NbH-N20jvseEnBFcHJy6FTfwESA8GxIewPQIzV3SFf5MBm14PyZ_EcKK_Qy6AtXxxfe7ZilLHA_w_uciTggC8aStI0EpaFsQak_k9Rrh-Ps1Po8fbsQ0IE_G87EY7wv4PDuWWs6d6o4I7ezSsq7enFHvz1E7Ky2AZbPPCwtwOBSYYQLoLqaPxRWmKxxFn2Z9PJhjooqrWqOWCJMRpV9ZIl0PxMgsLwRD8yzvmoIxiDVq0Ef1tKQo&dib_tag=se&keywords=servo%2Bmotor&qid=1710397332&sprefix=servo%2Bmotor%2Caps%2C89&sr=8-7&th=1.

Arduino (n.d.). "Firmata". In: *Last visited 05-05-2024* (). URL: https://www.arduino.cc/reference/en/libraries/firmata/.

"Chatgpt" (n.d.). In: *Last visited 27-05-2024* (). URL: https://chatgpt.com/?oai-dm=1.

Copilot (n.d.). "Bing.com". In: *Last visited 21-05-2024* (). URL: https://www.bing.com/chat.

dictionary, Oxford (n.d.). "modernisation definition". In: *Last visited 28-05-2024* (). URL: https://www.oxfordlearnersdictionaries.com/definition/american_english/modernize.

"Github Co-pilot" (n.d.). In: *Last visited 28-05-2024* (). URL: https://github.com/features/copilot.

"Goblintools Formalizer" (n.d.). In: *Last visited 27-05-2024* (). URL: https://goblin.tools/Formalizer.

Google (n.d.). "Crazy 8's". In: *Last visited 14-03-2024* (). URL: https://designsprintkit.withgoogle.com/methodology/phase3-sketch/crazy-8s.

Kohler (n.d.). "Kohler landing page". In: *Last visited 05-14-2024* (). URL: https://www.kohler.com.

Medium (n.d.). "Netflix personalisation recomendations". In: *Last visited 24-05-2024* (). URL: https://medium.com/@shizk/case-study-how-netflix-uses-ai-to-personalize-content-recommendations-and-improve-digital-b253d08352fd.

"Method 635" (n.d.). In: *05-03-24* (). URL: https://www.designmethodsfinder.com/methods/method-635n.

schade, Amy (n.d.). "Customization vs. Personalization in the User Experience". In: *Last visited 05-14-2024* (). URL: https://www.nngroup.com/articles/customization-personalization/.

Yablonski, Jon (n.d.). "Lawsofux". In: *Last visited 05-16-2024* (). URL: https://lawsofux.com.

Appendices

.0.1 Code for the Arduino

```
#include <OneWire.h>
#include <Servo.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 9

const int button1 = 2; // This button is for profile Aske
const int button2 = 3; // This button is for profile Emil
const int button3 = 4; // This button is for profile Rasmus
const int button4 = 5; // This button is for profile Jacob
const int pump1 = 10; // this pump is a dc 12v water pump for cold water
const int pump2 = 11; // this pump is a dc 12v water pump for hot water
Servo myservo; // create servo object to control a servo
const float targetTemp1 = 37.0;
const float targetTemp2 = 33.0;
const float targetTemp3 = 35.0;
const float targetTemp4 = 38.0;
const float coldTemp = 5.0;
const float hotTemp = 50.0;

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
DeviceAddress insideThermometer;


void setup(void)
{ pinMode(button1, INPUT);
  pinMode(button2, INPUT);
  pinMode(button3, INPUT);
  pinMode(button4, INPUT);
  pinMode(pump1, OUTPUT);
  pinMode(pump2, OUTPUT);
  myservo.attach(12);

  Serial.begin(9600);
  Serial.println("Dallas Temperature IC Control Library Demo");
  Serial.print("Locating devices...");
  sensors.begin();
  Serial.print("Found ");
  Serial.print(sensors.getDeviceCount(), DEC);
  Serial.println(" devices.");
  Serial.print("Parasite power is: ");
```

```
  if (sensors.isParasitePowerMode()) Serial.println("ON");
  else Serial.println("OFF");
  if (!sensors.getAddress(insideThermometer, 0))
  Serial.println("Unable to find address for Device 0");
  Serial.print("Device 0 Address: ");
  printAddress(insideThermometer);
  Serial.println();
  sensors.setResolution(insideThermometer, 9);
  Serial.print("Device 0 Resolution: ");
  Serial.print(sensors.getResolution(insideThermometer), DEC);
  Serial.println();
}


void printTemperature(DeviceAddress deviceAddress)
{
  float tempC = sensors.getTempC(deviceAddress);
  if(tempC == DEVICE_DISCONNECTED_C)
  {
    Serial.println("Error: Could not read temperature data");
    return;
  }
  Serial.print("Temp C: ");
  Serial.print(tempC);
}

void loop(void)
{
  Serial.print("Requesting temperatures...");
  sensors.requestTemperatures();
  Serial.println("DONE");
  printTemperature(insideThermometer);
}
void printAddress(DeviceAddress deviceAddress)
{
  for (uint8_t i = 0; i < 8; i++)
  {
    if (deviceAddress[i] < 16) Serial.print("0");
    Serial.print(deviceAddress[i], HEX);
  }
  float targetTemp;
  if (digitalRead(button1) == HIGH) {
    targetTemp = targetTemp1;
    myservo.write(25);
```

```
  } else if (digitalRead(button2) == HIGH) {
    targetTemp = targetTemp2;
    myservo.write(90);
  } else if (digitalRead(button3) == HIGH) {
    targetTemp = targetTemp3;
    myservo.write(135);
  } else if (digitalRead(button4) == HIGH) {
    targetTemp = targetTemp4;
    myservo.write(180);
  } else {
    digitalWrite(pump1, 0);
    digitalWrite(pump2, 0);
    return;
  }

  float ratio = (targetTemp - coldTemp) / (hotTemp - coldTemp);
  int hotSpeed = ratio * 255;
  int coldSpeed = (1 - ratio) * 255;
  digitalWrite(pump1, coldSpeed);
  digitalWrite(pump2, hotSpeed);
}
```

.0.2 Profile code in C

```
class Profile {
private:
    int id;
    double height;
    double temperature;
    double pressure;
    std::string name;

public:
    Profile(int id, double height, double temperature, double pressure,
    const std::string& name) :
        id(id), height(height), temperature(temperature),
        pressure(pressure), name(name) {}

    void setId(int newId) { id = newId; }
    void setHeight(double newHeight) { height = newHeight; }
    void setTemperature(double newTemperature)
    { temperature = newTemperature; }
    void setPressure(double newPressure)
```

```
    { pressure = newPressure; }
    void setName(const std::string& newName) { name = newName; }

    int getId() const { return id; }
    double getHeight() const { return height; }
    double getTemperature() const { return temperature; }
    double getPressure() const { return pressure; }
    const std::string& getName() const { return name; }
};

int main() {
    Profile profiles[] = {
        Profile(1, 185, 37.5, 80, "Emil"),
        Profile(2, 180, 36.8, 100, "Aske")
    };

    int inputId;
    std::cout << "Enter the ID to view profile: ";
    std::cin >> inputId;

    bool found = false;
    for (const Profile& profile : profiles) {
        if (profile.getId() == inputId) {
            found = true;
            std::cout << "Profile Information:" << std::endl;
            std::cout << "ID: " << profile.getId() << std::endl;
            std::cout << "Name: " << profile.getName() << std::endl;
            std::cout << "Height: "
            << profile.getHeight() << " cm" << std::endl;
            std::cout << "Temperature:
            " << profile.getTemperature() << " řC" << std::endl;
            std::cout << "Pressure: "
            << profile.getPressure() << " hPa" << std::endl;
            break;
        }
    }
```

.0.3 Profile selection

```
class Profile {
    constructor(id = 0, height = 0, temperature = 0,
    pressure = 0, name = "") {
        this.id = id;
```

```
        this.height = height;
        this.temperature = temperature;
        this.pressure = pressure;
        this.name = name;
    }

    setId(newId) { this.id = newId; }
    setHeight(newHeight) { this.height = newHeight; }
    setTemperature(newTemperature)
    { this.temperature = newTemperature; }
    setPressure(newPressure) { this.pressure = newPressure; }
    setName(newName) { this.name = newName; }

    getId() { return this.id; }
    getHeight() { return this.height; }
    getTemperature() { return this.temperature; }
    getPressure() { return this.pressure; }
    getName() { return this.name; }
}
```

.0.4 Made profile

```
const profiles = [
    new Profile(1, 185, 37.5, 80, "Emil"),
    new Profile(2, 180, 36.8, 100, "Aske")
];


const readline = require('readline-sync');
const inputId = parseInt(readline.question
("Enter the ID to view profile: "));

let found = false;
for (const profile of profiles) {
    if (profile.getId() === inputId) {
        found = true;
        console.log("Profile Information:");
        console.log("ID: " + profile.getId());
        console.log("Name: " + profile.getName());
        console.log("Height: " + profile.getHeight() + " cm");
        console.log("Temperature: "
        + profile.getTemperature() + " řC");
```

```
        console.log("Pressure: " + profile.getPressure() + " hPa");
        break;
    }
}

if (!found) {
    console.log("Profile with ID " + inputId + " does not exist.");
}
```

.0.5 Preset settings

```
const readline = require('readline-sync');


const userProfile = new Profile(0, 0, 0, 0, "");


const newId = parseInt(readline.question("Enter ID: "));
userProfile.setId(newId);

const newHeight = parseFloat(readline.question("Enter height (cm): "));
userProfile.setHeight(newHeight);

const newTemperature = parseFloat(readline.question
("Enter temperature (řC): "));
userProfile.setTemperature(newTemperature);

const newPressure =
parseFloat(readline.question("Enter pressure (hPa): "));
userProfile.setPressure(newPressure);

const newName = readline.question("Enter name: ");
userProfile.setName(newName);

// Displaying the added profile
console.log("\nProfile Information:");
console.log("ID: " + userProfile.getId());
console.log("Name: " + userProfile.getName());
console.log("Height: " + userProfile.getHeight() + " cm");
console.log("Temperature: " + userProfile.getTemperature() + " řC");
console.log("Pressure: " + userProfile.getPressure() + " hPa");
```

.0.6 Main page

```
// Water Pressure

document.addEventListener('DOMContentLoaded', function() {
    const bars = document.querySelectorAll('.bar');
    const plusButton = document.getElementById('plusButton');
    const minusButton = document.getElementById('minusButton');

    let waterPressure = 10;


    for (let i = 0; i < waterPressure; i++) {
        bars[i].style.backgroundColor = '#FFA85C';
    }

    plusButton.addEventListener('click', function() {
        if (waterPressure < bars.length) {
            bars[waterPressure].style.backgroundColor = '#FFA85C';
            waterPressure++;
        }
    });

    minusButton.addEventListener('click', function() {
        if (waterPressure > 0) {
            waterPressure--;
            bars[waterPressure].style.backgroundColor = '#EFF2D2';
        }
    });
});

document.addEventListener('DOMContentLoaded', function() {
    var slider = document.getElementById("mySlider");
    var output = document.getElementById("sliderValue");
    output.innerHTML = slider.value;

    output.innerHTML = slider.value;

    slider.oninput = function() {
        output.innerHTML = this.value;
    }
    function setSliderValues(profileId) {
```

```
        var profile = profiles[profileId]
        document.getElementById("MySlider").value = profile.height;
    }
//round slider

$("#roundSlider").roundSlider({
    sliderType: "min-range",
    handleShape: "round",
    width: 8,
    radius: 90,
    value: 37,
    startAngle: 90,
    max: "45",
    min: "15",
    step: "1",
    handleSize: "+20",
    tooltipFormat: function(args) {
        return args.value + "řC";
    }
});
});
```