Expertise
and insight
for the future

Leppävaara Certificate

Arsi Arola, Duc Vo, Mai Vu, Mikko Larke, Thang Tran

Kalaportti - Smart Device for Home Aquarium

Helsinki Metropolia University of Applied Sciences

Information Technology - Smart Systems

Internet of Things (Project)

Final Report

22 December 2020

Contents

List of Abbreviations

ASCII	American Standard Code for Information Interchange.
ESP-IDF	Espressif IoT Development Framework.
GND	Ground.
GPIO	General-purpose Input/Output.
HTTP	Hypertext Transfer Protocol.
I2C	Inter-integrated Circuit.
IDE	Integrated Development Environment.
IoT	Internet of Things.
LCD	Liquid Crystal Display.
LED	Light-emitting Diode.
MOSFET	Metal Oxide Semiconductor Field Effect Transistor.
MQTT	Message Queuing Telemetry Transport.
NTP	Network Time Protocol.
NVS	Non-volatile Storage.
op-amp	Operational Amplifier.
PWM	Pulse-width Modulation.
RPM	Revolutions Per Minute.
RTOS	Real-time Operating System.
SPI	Serial Peripheral Interface.
SSID	Service Set IDentifier.
UART	Universal Asynchronous Receiver/Transmitter.
URL	Uniform Resource Locator.

1 Introduction

This report is the final document for the Internet of Things (Project) course at Metropolia University of Applied Sciences. The course challenges students to develop a useful smart wireless product. Leppävaara Certificate team decides to continue building the **Kalaportti | Smart Device for Home Aquarium** project from the last Smart Systems and Wireless Communication course. The previous final report presents the features, functions, components plan for the project. For this report, the goal is to show the implementation and outcome of the final product.

Fishes in particular and pets in general are becoming crucial roles in human life and more thought of as family members or human best buddies. The increasing number of pets and more robust bonding leads to the growing demand for pet products and services. More money is spent on pet healthcare and well-being. [1]. Indeed, pet healthcare products not only help owners take care of their pets easier but also allow pets to live longer and more comfortably, preserving the human-animal bond.

Kalaportti | Smart Device for Home Aquarium embraces the above idea to satisfy the third most owned household pet in the world - fish [2] and its owner. Owning an aquarium has been shown to improve numerous aspects of human life. Fishes make owners live happier and healthier. However, guaranteeing that fishes live in a quality environment might be difficult and stressful since there are many factors to look after, such as pH levels, temperature, water levels, etc. Kalaportti device measures all essential and latest information, sends it to the owner's devices via Kalaportti Discord Bot or website. Kalaportti provides users the fish feeding feature. Kalaportti also aims to please the fish owner with custom lighting systems. Kalaportti ensures having fishes as pets is fun and enjoyable for anyone.

In conclusion, the team accomplished building the Kalaportti product using multiple sensors and actuators. The following chapters provide complete implementation details of the hardware and software parts. The last chapter reveals the team's problems during the developing process.

2 Hardware

2.1 Components

The used components for this stage of the project is not much different from the original plan. The two most considerable changes were that the team decided to exclude the pH sensor and the Liquid Crystal Display (LCD) display. The pH sensor costs more than 100€, even more than other components combined. Also, no suitable and cheap replacement could be found; thus, the pH monitor feature is not possible in this project. The LCD was removed since, during the development, the team recognized that the website already had enough and up-to-date information. In addition, there are other minor adjustments with other components too. The final choice for building a Kalaportti device is shown in table 1.

Type	Component	Name
Controller		ESP32 [3]
Sensors	Temperature sensor	DS18B20 [4]
	Water level sensor	Reed switches RI-03A [5] + magnet
Actuators & others	Water pump	MOT DIY61289P [6]
	Food feeder	Motor provided by the school + 3D printed drill
	LED light	10cm RGB LED stripe [7]

Table 1: List of final components for the Kalaportti project.

Another change compared to the initial proposal is the Light-emitting Diode (LED) light. The planned one is 1 meter long. The team realized that it is too much for a demo with a small water tank. Therefore, the shorter one, 10 centimeters RGB LED stripe, was chosen. This stripe is also waterproof.

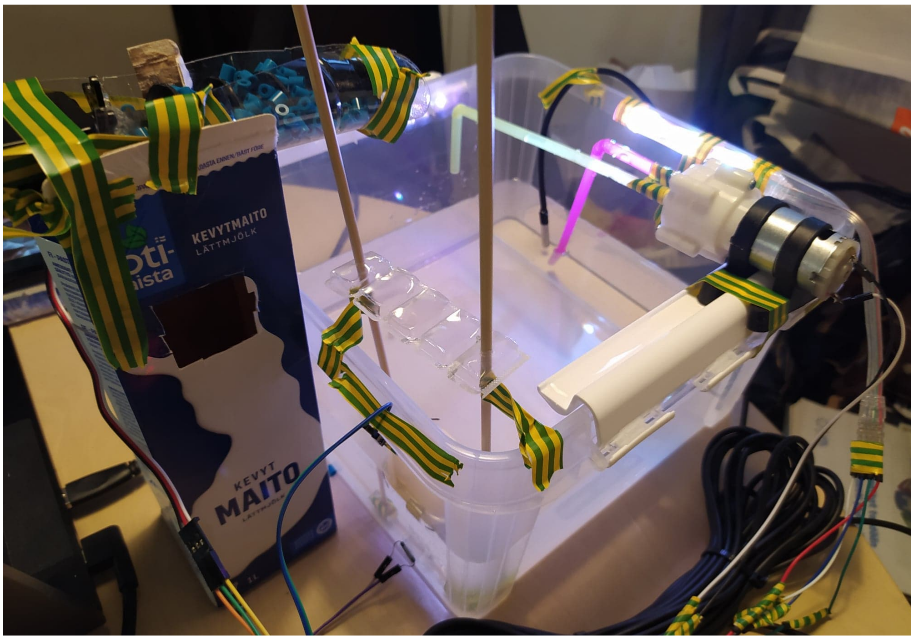

Figure 1: All components.

2.2 Temperature Sensor

To make the implementation process easier, the team used a premade header file created by a Github user called *feelfreelinux* [8]. This library provides all configuration and communication with one wire protocol. Users only need to do the following steps:

- Implement *ds18b20_init (int GPIO)* function to initialize the connection with General-purpose Input/Output (GPIO) pin where the sensor is connected.
- Implement *float ds18b20_get_temp(void)* function to get a float-type variable storing the last read temperature.
- Connect the temperature sensor data wire to the proper pin on the controller. Other Vin and GND wires are connected to the ground.

2.3 Water Level Sensor

The components set-up for the water level monitor is to have upper and lower reed switches outside the aquarium, and a magnet floating really near the reed switches but inside the fish tank. The upper switch is activated when the magnet gets close, alerting when the water level goes too high and vice versa with the lower one. In the perfect scenario, the idea can be achieved with a tube attached to the aquarium's inside wall to guarantee the activation with the switches, and inside the tube is the magnet sticks to a piece made from floating materials, to make the magnet floats with the water level. This setting would be great in real-life since it does not take much space in the aquarium and ensure no interference between the magnet and other activities.

However, the above architecture is challenging and takes time to perform since the team did not find a suitable size tube. Thus, for the proof of concept, the team decided to do a different set-up, as shown in figure 2. Instead of having a tube for guiding, two sticks are used to lock the magnet in place. A Kinder egg toy placeholder is used as the floating piece. Other options that were considered are styrofoam or air bubble sheets. However, the Kinder egg toy is perfect since it is thin and light and the inside is full of air. The magnet is glued outside the egg (figure 3). The set-up worked within some attempts.

The system monitors periodically for changes in reed switches status. If so, the system alerts the user about the difference. The program also uses a boolean variable to ensure the alert only resets when the user clears the water level alert manually. This setting makes sure the user notices the changes in the tank temperature.

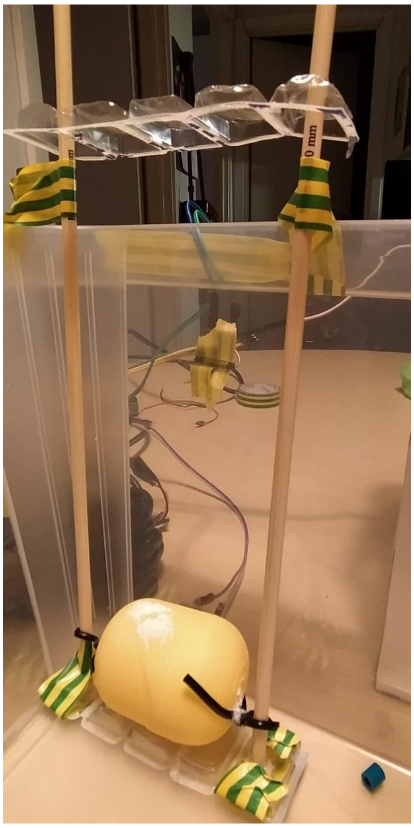

Figure 2: The water level monitor.

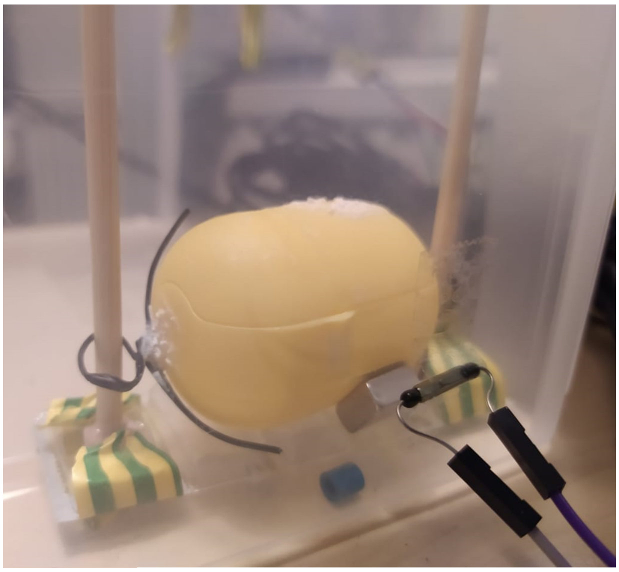

Figure 3: The magnet and the reed switch.

2.4 Water Pump

The water pump is attached to the top edge of the fish tank, as shown in figure 4. It has two small tunnels: one is for the water to come in, and the other is for the water to draw out. The team uses two straws as pipes, tied to the water pump tunnels with tape. The pink straw carries the water up. The water then drops through holes drifted along the green straw.

The water pump works with a simple DC motor that uses 12 volts to operate. In addition, the team wants to provide the user with the ability to turn on and off the water pump when needed. Therefore, an H-bridge is added to the system. An H-bridge obtains the high and low signals from the controller and acts as a switch to control the Vin of the motor. Thus, the water pump can be turned on and off as commanded.

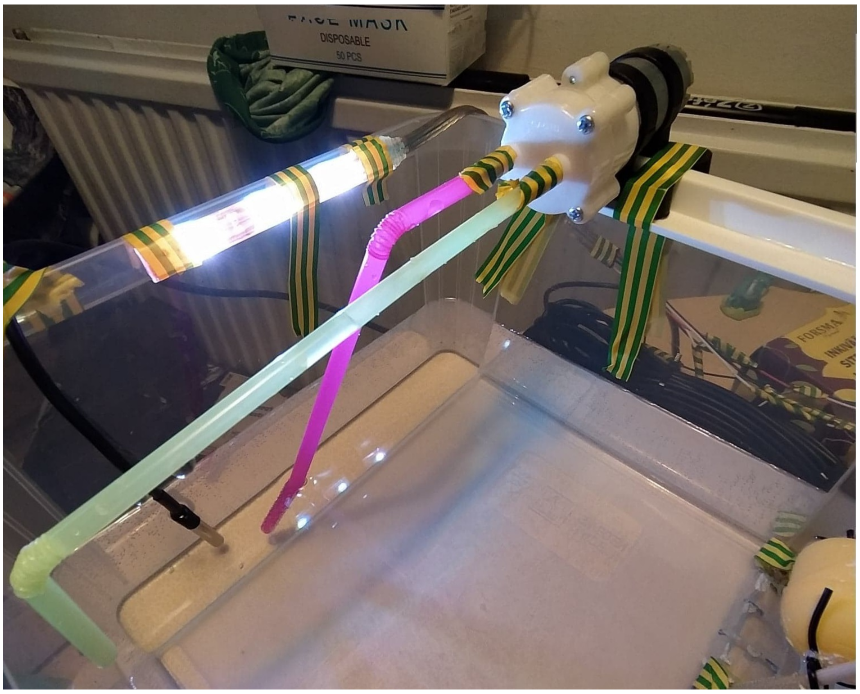

Figure 4: The water pump.

2.5 Food Feeder

The automated fish feeder (figure 5) utilizes a 360-degree servo motor with a long piece that looks like a drill or an auger. When the feeding time comes, the servo motor spins, causing the auger to twirl too. This motion moves the food forward along the pipe and drops it into the fish tank.

Figure 5: The fish feeder from above.

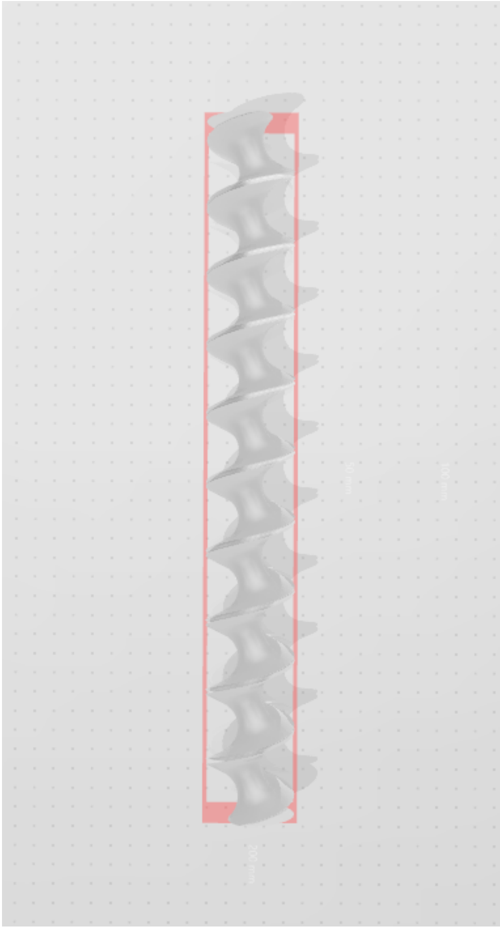

Figure 6: The auger 3D design.

The auger was the only 3D printed element of the project. Figure 6 shows the 3D design on the computer. The printing process was successful on the first try. Other parts, such as the tube and the stand, were creatively constructed with household items to assemble the feeder properly (figure 7). Everything was attached together by electric tape and glue.

Figure 7: The homemade fish feeder.

The fish feeder works thanks to the servo motor and the time given by the Network Time Protocol (NTP) server. The system checks once per second whether the current hour, minute, and second match the user's feeding time. If so, a binary semaphore flag is sent to the servo motor. The motor gets the signal and starts pushing food into the aquarium. To control the servo's rotation, it needs the Pulse-width Modulation (PWM) with different duty cycles. This is a part of Espressif IoT Development Framework (ESP-IDF) framework, a library called PWMC. The library allows the PWM frequency and duty cycle to be specified.

2.6 Modifications To Feeder Servo

In the components request at the beginning of the course, the project's motor was a continuous rotation model. However, to minimize the project cost, the team used the 180-degree servo motor available from the school storage. Thus, the main task for the food feeder to work as expected is to modify this motor to move 360-degrees continuously.

Examining the mechanism of the 180-degree servo motor (figure 8), there are three essential pieces, marking with the green, blue, and red circle. The green one indicates the physical motor. Its only function is to move the other gears. The blue circle points to the metal knob, connecting to the potentiometer. When rotating, it will have different voltage readings correlating to its position. This voltage is measured by a circuit that is at the bottom of the motor. Another gear, circled in red, is placed on top of the potentiometer. It attaches to and turns with the potentiometer. The gear also has a mechanical limit on the other side (figure 9) to physically limit the movement to a little over 180-degree.

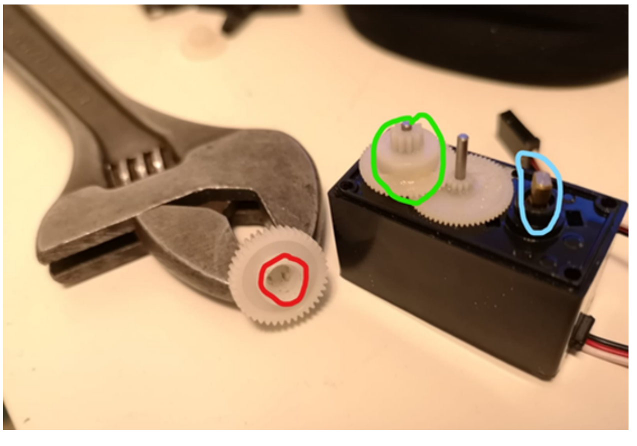

Figure 8: The servo motor with its pieces.

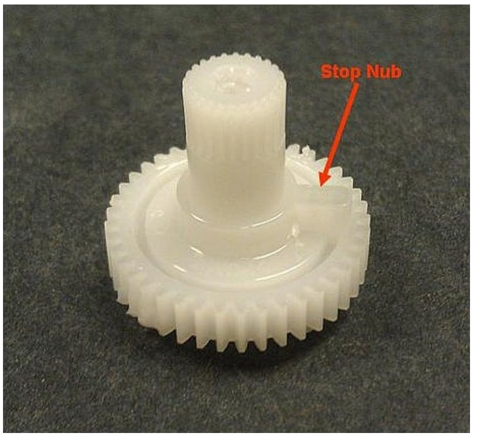

Figure 9: Physical stopper sof the servo.

The idea to modify this 180-degree servo is to make the potentiometer output constant. It can be done either by disconnecting it from the circuit, then replacing the connection with correct resistors, or by stopping the potentiometer nub from moving even when the motor is spinning. The team decided to do the second option. It can be done easily by removing the grooves and blocks of the gearing that stick to the potentiometer, then locating it with super glue. Removing the grooves was done with an electric drill. First, one drilled slowly with a relatively small drill bit. After that, increase the drill bit sizes and keep drilling until the grooves were gone. The blocks on the other side were removed with pliers and a sharp knife.

In the end, the idea was tested, and it did work smoothly as planned. However, the potentiometer seemed to move a little bit over time. Therefore, the team decided to add some super glue to make it last longer.

2.7 LED Light

As mentioned before, the project uses a 10 cm long RGB LED stripe. It sticks to the water tank's side by tape as in figure 10.

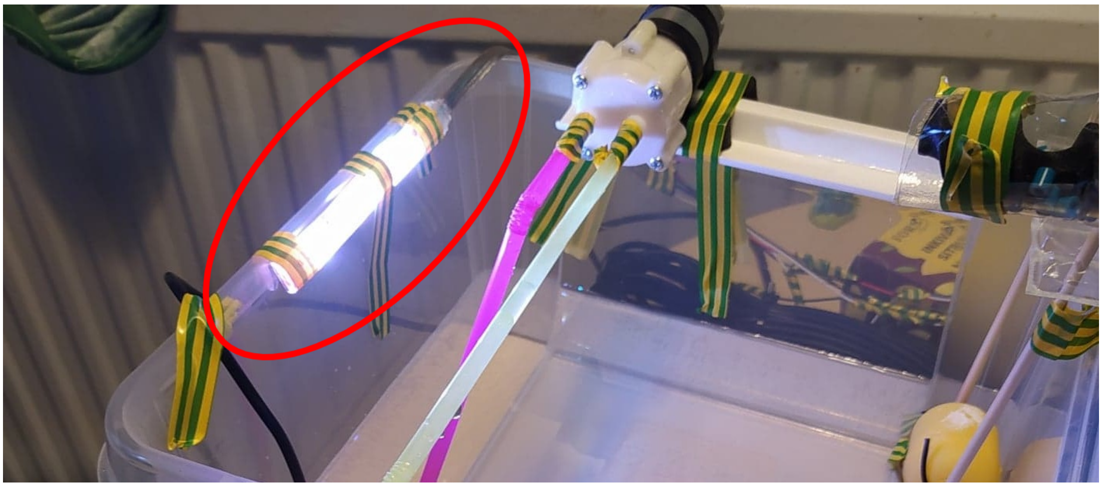

Figure 10: LED stripe circled in red.

The RGB stripe needs four separate 12 volts inputs: three for each primary color (red, green, and blue) and one for the voltage input. The voltage input can be connected directly to the power source. However, other colors need to be controlled with PWM so that each color can be modified. That is provided by LEDC library, which is included in ESP-IDF framework, to control PWM duty cycles of each color channel. The duty cycle is scaled from 0 to 255 for easy conversion of universal RGB codes.

2.8 Soldering

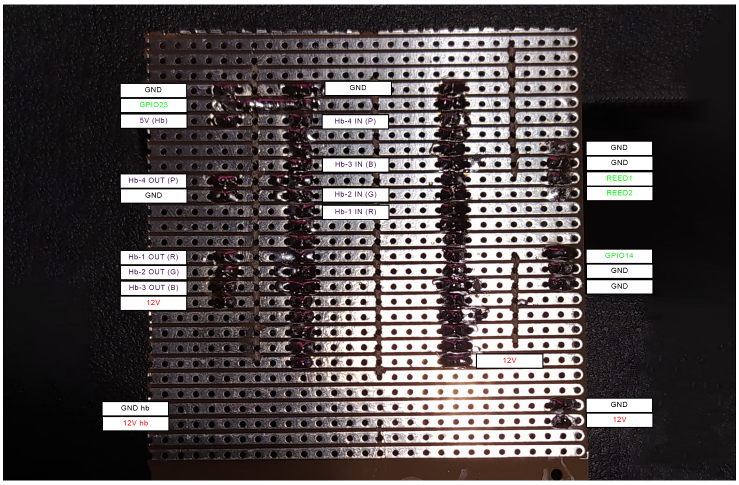

Figure 11: The soldering map.

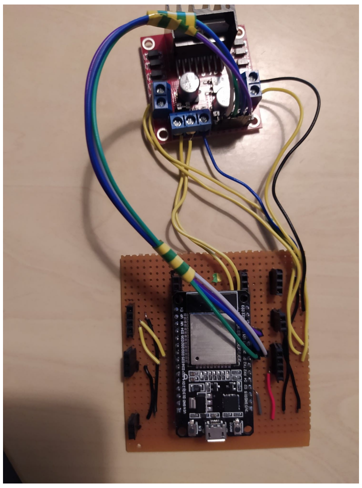

Figure 12: The final solder.

The device is built on a solderable breadboard with the female header for each component to ensure modularity and stability. By melting solder around the connection, two or more electronic parts are joined together. Soldering will create a strong electrical bond between the solderable board and the female header or pin header when it cools down because it is a metal alloy.

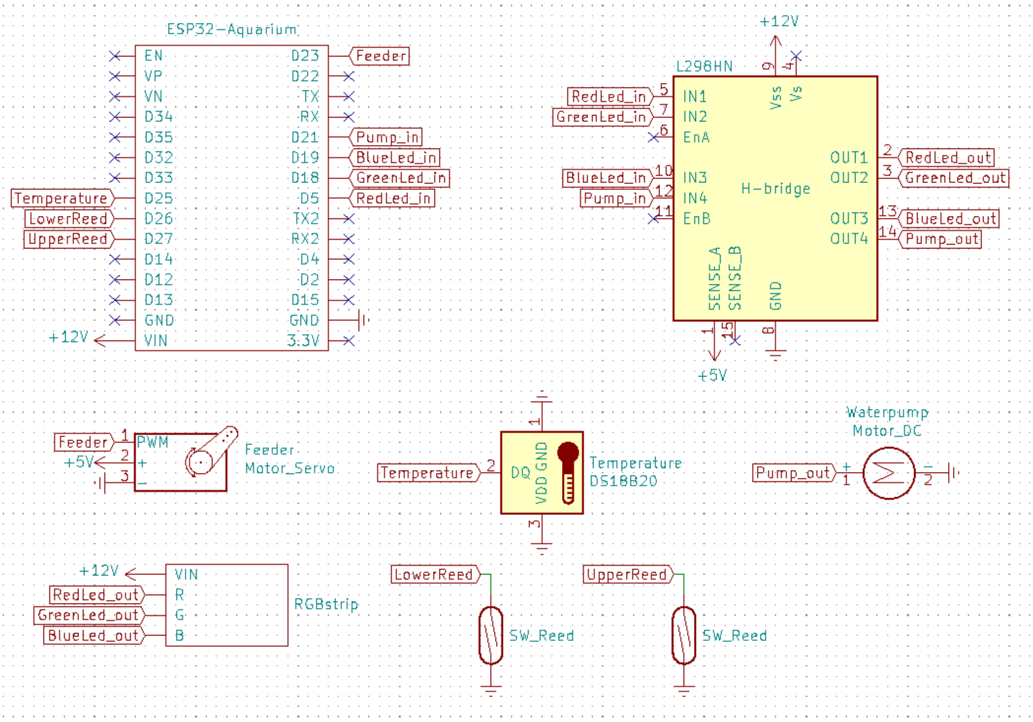

Figure 13: The pin layout.

3 Software

There are several possible options of programming languages, platforms, frameworks for programming on the ESP32. The most common, also the possibilities were carefully considered, were MicroPython, Arduino Integrated Development Environment (IDE), and ESP-IDF. Each one of them has its own advantages and disadvantages. Those are the criteria to examine and choose the most suitable option.

Firstly, MicroPython might be a good choice since it allows coding in Python 3 and provides a small subset of the Python standard library. Python is an easy and straightforward langue to work with. Compared to other programming languages such as C++, less code in Python is needed for the same task to be done. Despite the ease, there are some limitations when it comes to components. Some sensors in this project do not have existing code in Python, but higher chances in C/C++. It would be messy and painful to code them all from scratch. Additionally, only a few members have previous experience with Python. Especially, no member has tried Python with microcontrollers before. MicroPython might be an attractive option, but it is not too beneficial. Therefore, the team decided not to use MicroPython.

The second promising option is Arduino IDE since there are tons of tutorial online and ready-to-go code. However, it requires to work with Arduino IDE software, and the team has very few practices in this aspect. Besides, its programming language is similar to C but somewhat different. C/C++ are more widely used in real-life applications, and those are what the team has been doing during the last courses. Consequently, Arduino IDE is not favorable.

Lastly, ESP-IDF was considered because it is a platform coded in C, uses the C library, and also a cross-platform. One of the provided ESP-IDF libraries utilizes the FreeRTOS, which the team well-practiced and studied in the last period. The software itself is a command-line program. For building the code and flashing the chip, command lines can be used, which the team got used to quickly. In conclusion, ESP-IDF was the perfect option for the team process.

3.1 Software Flowchart

Figure 14 shows the software flowchart of the project. There are six threads: five of them directly connect to the components, and the other one is implemented to update the time for the food feeder task.

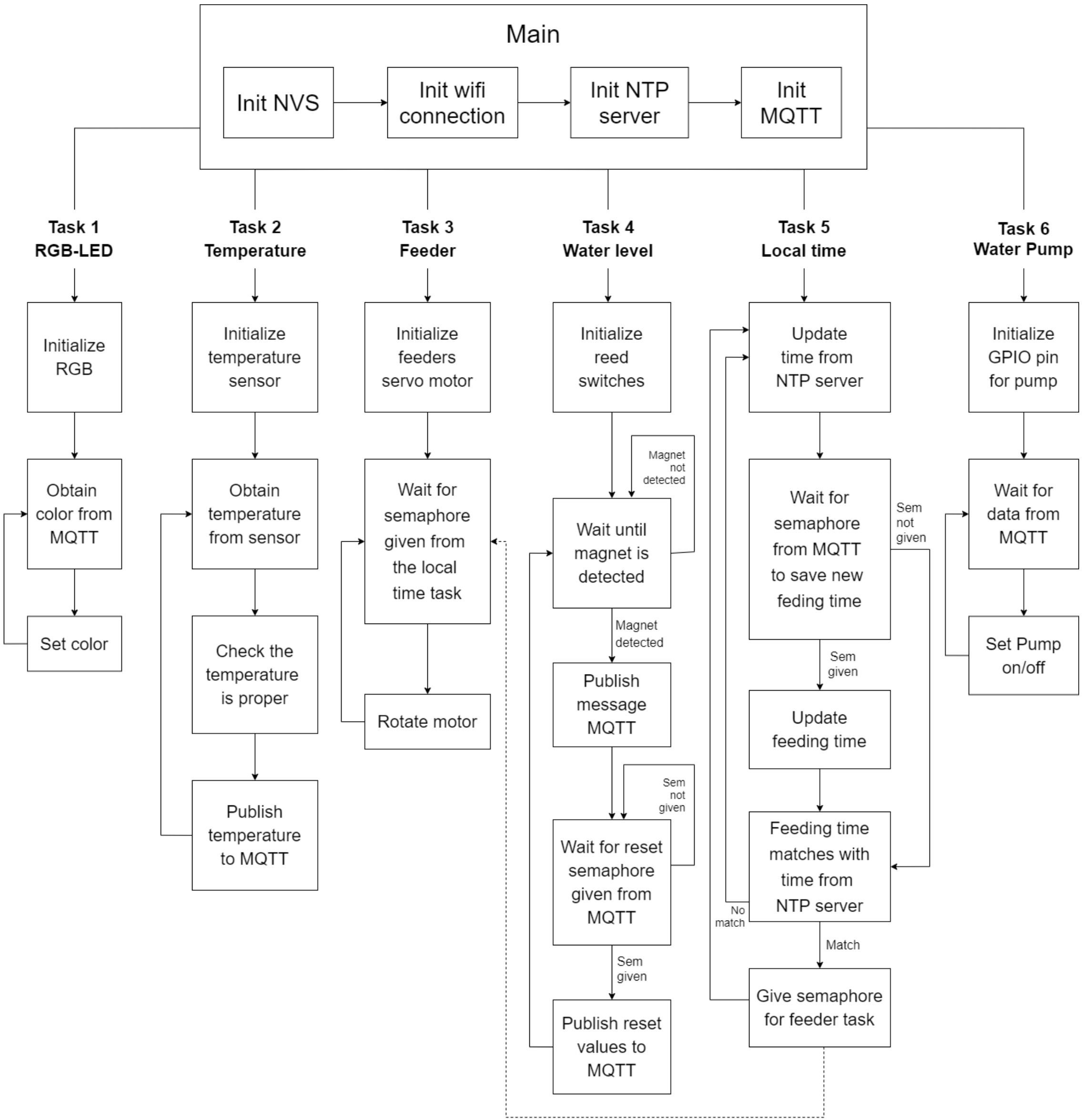

Figure 14: Software flowchart of the Kalaportti project.

3.2 Header Files

A header file is a file with extension .h, which contains C function declarations and macro definitions shared between several source files. FreeRTOS is a Real-time Operating

System (RTOS) for microcontrollers. It is used to separate parts of the software to different tasks that can run in parallel.

Name	Description
ds18b20	- Initialize one wire connection to sensor - Parse temperature reading from device
feederServo	- Initialize servo with appropriate PWM timer - Set rotation direction and speed by changing duty cycles
systemTiming	- Initialize connection to NTP server - Get the current time from the server
ConnectWifi	- Initialize Wifi connection - Connect to Wifi with the SSID and password
RGBcontroller	- Initialize red, green, and blue channle with appropriate PWM timer - Set color by universal HEX or RGB code - Change color instantly or by fade

Table 2: List of the header files.

3.3 Sending data to Internet MQTT

The smart aquarium uses Message Queuing Telemetry Transport (MQTT) protocol to send and receive data from the Internet. Kalaportti utilizes MQTT broker provided by Adafruit IO to use Uniform Resource Locator (URL) based dashboard, showing and sending data between dashboard and aquarium. Connection to the dashboard on a remote server is done with MQTT protocol and uses MQTT library provided by the ESP-IDF framework.

MQTT connection between ESP32 and Adafruit IO dashboard first needs to be authenticated by Adafruit IO username and user-specific active key. After the controller sends this information to the dashboard, it is available to use. Each element on the dashboard is updated by a feed generated inside Adafruit IO. These feeds can then be either published or subscribed by ESP32 to show updated values or changed variables from the program.

3.4 Adafruit IO Dashboard

Kalaportti uses Adafruit IO Dashboard as its web platform for data visualization and device control. This platform is chosen because it is free-to-use, and it meets all requirements for a basic student Internet of Things (IoT) project.

Adafruit IO Dashboard is not too complicated to use. Once Kalaportti is connected to the platform, the Adafruit IO Dashboard allows us to visualize data collected via sensors with graphs and charts. There are built-in buttons and switches to trigger particular features on our device. Furthermore, there are plenty of tutorials, well-documented client libraries, and ready-to-go code (for ESP32) available, simplifying the programming process.

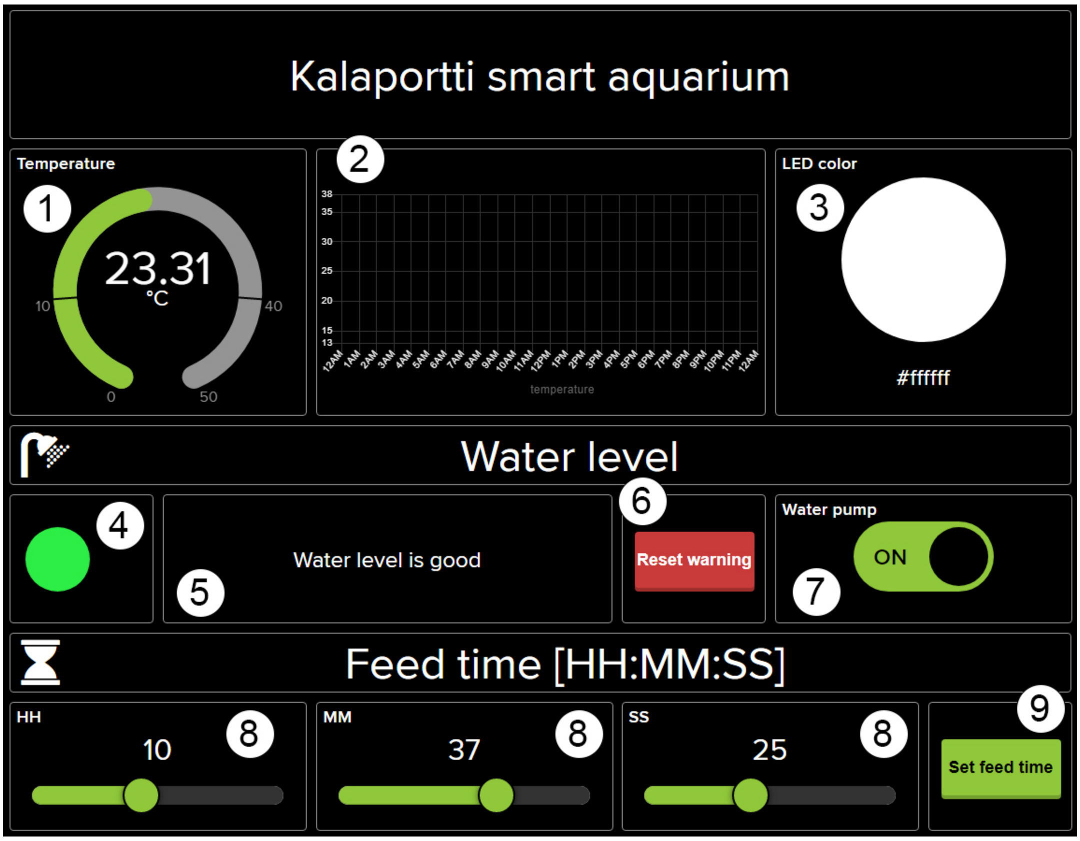

Figure 15: Adafruit IO dashboard.

Kalaportti Adafruit IO Dashboard has several blocks. Each block is implemented for a specific purpose. The blocks can be divided into 9 parts based on their functions, marked from 1-9 as in figure 15.

1. Gauge block, showing the current water temperature with warnings set to below 10 and over 40 degrees.

2. Line graph of the recorded temperature over time.
3. Color picker of aquarium LED color.
4. Indicator light for water level. Green is good, while red is bad.
5. Text box, showing the verbal output of the water level, whether the water level is too high, too low, or good.
6. Momentary button to reset water level warning.
7. Toggle button to turn the water pump on or off.
8. Number sliders system to modify the feeding time in 24h.
9. Momentary button to confirm the feeding time.

3.5 Discord Bot

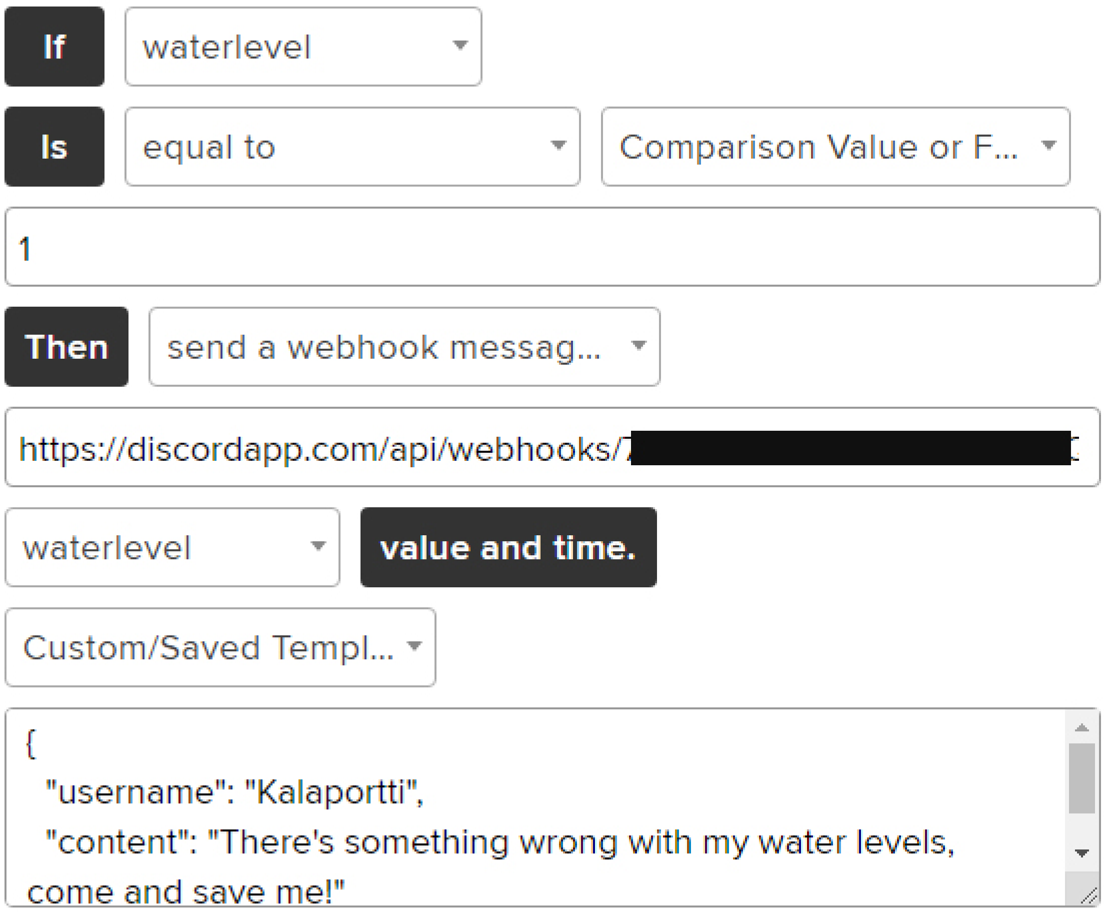

Figure 16: Webhook.

With Adafruit IO, webhooks and triggers of the published values from the dashboard's feeds can be used to inform the user. The available features right now are to alert when something is wrong with the water levels or temperatures. Smart aquariums webhook is directed to Discord Bot, so users can easily be notified of the aquarium problems.

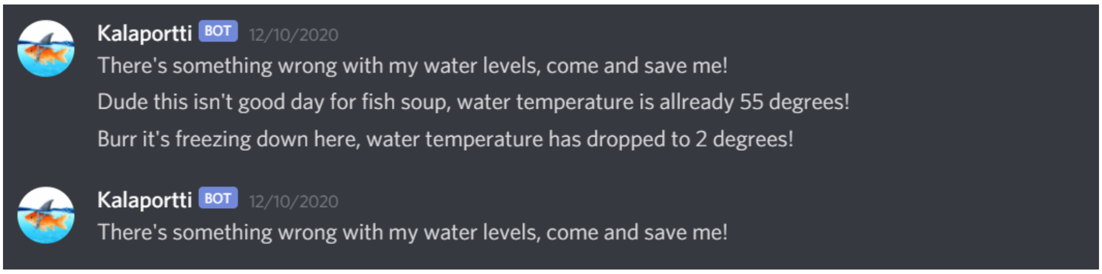

Figure 17: Discord Bot warning messages.

4 Problems

During the development of the project, the team encountered lots of difficulties. Fortunately, the problems were solved so that the project can be finished in time. Some of the major issues and their solutions are mentioned in table 3.

Issues	Solutions
Remote working environment due to the pandemic	Divide the works as much as possible. But in reality, the person holding the controller was the one to be able to properly develop at a time.
Temperature sensor did not return data on first attempts	Temperature sensor's datasheet shows that one wire protocol needs both GND and Vin connect to the ground. After fixing, the temperature data was transmitted properly.
Highly unstable connector for LED stripe wires	Use tons of electrical tape and delicacy while testing. For the final product, a propper adapter or a different led stripe model is needed.
Accidental short-circuiting of the original controller, and a different pin layout of the spare dev kits	Re-solder some of the pins to work with the new pin layout, and build the properly insulated case for the solder board.
Build the prototype using only household resources	Need creative minds and lots of tape.
Difficulty when building the website and working with RESTful HTTP requests	Use pre-provided dashboard and MQTT protocol instead of HTTP requests. For the final product, an experienced web developer might be needed.

Table 3: List of big issues and their solutions.

5 Discussion and Conclusion

The original plan in the previous course was to have 5 features: monitor water temperature, water level, pH level, modify the feeding time, the LED light. Also, the plan was to produce 2 convenient ways to show the data: by the LCD display and the website. The team managed to finish 4 out of 5 features (except for the pH level monitor) and the website. In replace of the LCD display, the Discord Bot was made to quickly alert the user. The substitute is reasonable since now most people use mobile phones daily. In conclusion, the team agreed that the Kalaportti project achieved the initial objective and happy with the results.

Still, some improvements can be carried out in this project. Some additional features discussed in the previous report, such as the ability to adjust the water level or to control the water temperature itself, were unfortunately not considered in this stage due to their extreme complexity. Other upgrades can be done in the future so that the final product can be commercially or widely used.

Firstly, the servo motor can be changed to the 360-degree one for a more stable operation. Secondly, for real fish aquariums, the LED light needs to be replaced with the longer one for better light performance. Most importantly, visually, the product needs to have a more suitable appearance. Due to the lack of time and proper tools, the project was built with household items. Plastic tubes are needed to replace the straws and to transform the design of the water level monitor. 3D printers might be used to make custom modes to fit in components. The website can be developed further to have a more attractive display.

In conclusion, Kalaporrti is a helpful device for fish owners. It reduces the pressure and the stress of having fish by helping users monitor necessary water conditions and provide additional features. The project was completed with cost-efficient electronic components. This IoT Project course has helped group members apply the hardware and IoT knowledge into practices. The team is thankful to receive enthusiastic support and constructive comments from the teacher. The project can be carried on in the Innovation Project course.

Bibliography

1 Hyman S. Investing In The Pet Care Industry. ProShare Advisors LLC; 2019. Available from: https://insight.factset.com/pet-care-industry-investing#:~:text=The%20pet%20industry%20spans%20veterinary,pet%20and%20pet%20supply%20stores.&text=The%20global%20pet%20care%20industry,as%20%24203%20billion%20by%202025. [cited October 13 2020].

2 Man’s Best Friend: Global Pet Ownership And Feeding Trends. Growth from Knowledge; 2016. Available from: https://www.gfk.com/insights/mans-best-friend-global-pet-ownership-and-feeding-trends [cited October 13 2020].

3 ESP32. Wikipedia;. Available from: https://en.wikipedia.org/wiki/ESP32 [cited December 18 2020].

4 One Wire Waterproof Temperature Sensor. Elecrow Bazaar; 2019. Available from: https://www.elecrow.com/wiki/index.php?title=One_Wire_Waterproof_Temperature_Sensor [cited December 18 2020].

5 Reed Switch. Farnell;. Available from: https://fi.farnell.com/comus/ri-03a/switch-reed-spst-no-0-5a-200vdc/dp/2409213?st=reedswitch [cited December 18 2020].

6 WATERPUMP WITH MOTOR 6-12VDC. PARTCO The Electronics Shop;. Available from: https://www.partco.fi/en/electromechanics/motors/dc_motors/20206-mot-diy61289p.html?search_query=Waterpump&results=1 [cited December 18 2020].

7 Waterproof IP65 LED stripe RGB 10cm. PARTCO The Electronics Shop;. Available from: https://www.partco.fi/en/leds/led-stripe/14579-led-n-vt-rgb.html [cited December 18 2020].

8 Simple library for single DS18B20 on ESP32. feelfreelinux - Github;. Available from: https://github.com/feelfreelinux/ds18b20 [cited December 18 2020].