Technical University of Denmark
Department of Electrical and Photonics Engineering

Harbor Foot

A project for

34315
Internet of Things
Application and Infrastructure Implementation

Submitted By

Student Name	Student ID
Eigil Albæk	s201030
Jonathan Ulrich	s203882
Lucas Brylle	s203832
Thomas Hegelund	s211899

Supervised By

Sarah Renée Ruepp

Submission Date: 11/05/2023

Contents

1 Introduction

Yachting is a common leisure activity and attractive way of going on vacation in Denmark, due to the extensive coastline and the numerous marinas reachable within a week of sailing from anywhere in the country. So far, most Danish marinas are yet to be modernized. That's where our solution Harbor Foot comes into play.

1.1 Pain points

When visiting external marinas by boat, recreational sailors rely on there being free spots available. Though some spots are designated guest spots, the best spots tend to be held by members of the local yachting club. It is seen as good practice, to make your own marina spot available for others to use, while you are away from your home marina for one or more nights. This is done by flipping a red sign to green and writing a return date on it, so sailors borrowing the spot know when to leave the spot. Though it is good practice to do this, it is often forgotten or neglected. This leads to a few pain points for recreational sailors:

- Free marina spots are often not made available to recreational sailors coming to stay the night, limiting the options for incoming guests.
- It is often some of the more lucrative spots that are not made available to guests.
- In large, busy, or tightly packed marinas, it can be a frustrating experience to look for an available spot as you have to go very close to an empty spot in order to determine whether or not it is marked as available.

These pain points lead to lost revenue for the marinas, hence guests being less likely to visit because of lack of overview.

Figure 1: The left image is the current system used in most of the harbors to track the state of a berth. To the right, our replacement of this system

2 Solution: Harbor Foot

This project attempts to address the aforementioned problems through our sensor system named *Harbor Foot*. The system is meant to replace the common red/green plastic signs indicating

the availability of a berth. The sensor system would be able to collect data on the occupation status of a berth in real-time and update it automatically as boats come and go. This flow of information can both be used to guide guests coming into a marina to an available spot as well as generally making more spots available by requesting sailors (e.g. through e-mail or app notification) leaving their home marina to make their berth available to others until a certain date. A system unit would also be able to display the return date of the berth holder which can be updated from afar. Thus spots can be made available both automatically and remotely.

2.1 Additional pain points

The same information used to improve the experience of recreational sailing can also be used for addressing some pain points of the marina administrators. Guests usually have to pay a fee for staying in a marina. It is the harbor master job, to spot new guests in the harbor and collect the marina fees. This is a job that could be made easier using the information gathered by Harbor Foot. Information about which boats have arrived on a given day can be integrated with the existing payment collection system making it quick and easy to get an overview of missing payments. Additionally, the administrators will gather more data about the usage of the marina over time, which can have intrinsic value in itself. Besides knowing how many boats come in on specific days, it will also be easy to track metrics such as: Which spots are the most popular, when guests and associated boats come and go, how long guests stay, etc.

2.2 Assumptions

A few assumptions have been made about why recreational sailors do not always make their berth available to use for others when going away for one or more nights:

- They forget to make their spot available.
- There is some friction associated with assigning a return date to a berth as it often involves physically applying a piece of painter's tape and writing on it with a marker.
- If the berth holder is uncertain about their return date, they risk coming back to an occupied spot.

The proposed solution solves all of the above by reminding the berth holder to update the spot availability, eliminating the need for physical artifacts like tape and markers and the action of applying them, and the option to change the homecoming date remotely.

Besides the assumptions mentioned above, it has also been assumed that the yachting community would welcome this innovation. It is furthermore assumed that it is technically feasible to make a reliable and cost-effective system that can withstand the relatively harsh environment found in Danish marinas throughout the year. Cost efficiency is especially important, since most harbors have a small budget, so to justify our product we need to hit a price range they're comfortable in for them to increase their harbor and revenue.

2.3 Instructions for use

There should be one Harbor Foot sensor for each berth in the marina. The sensor should be attached to the pier such that it faces the boat occupying the berth and the displayed information is readable from an incoming boat. The boat that usually belongs in the spot should be equipped with a chip or a device from which the sensor can detect the presence.

Once the boat that usually occupies the berth leaves the marina, the boat owner will be requested to select a return date via e-mail. If the owner does not respond, then the berth

remains marked as occupied through a red LED. If the owner does select a return date, the berth will be marked as free with a green LED, and the return date shown with a 4-digit 7-segment display. Whenever an incoming guest occupies the berth, this information will be updated on the server.

Recreational sailors coming into a marina as guests can view which spots are available online, through either a list view or a map. This will eliminate the stress of searching a busy marina for an available berth.

The harbor master can use the online information to keep track of where in the marina there are new guests that have not paid the marina fee. To help keep track of which guests have already paid the marina fee, the harbor master can register this information online. They would also be able to see if a guest has accidentally occupied an unavailable spot so they can help them find another. Upon the return of the spot holder, the system will recognize the presence of the correct boat, and no actions need to be taken.

2.4 Hardware and circuit design

In the design for the setup, we wanted to use as little space as possible, so we limited ourselves to only using one breadboard. This also made it important to plan out where every component and wire needed to be. the layout of the breadboard can be seen in figure 14.
For our setup, we used the Arduino Uno, which uses an ATMega328P as its microcontroller. The advantage of using the Arduino Uno for our design is the 23 general-purpose I/O pins which can be used for both digital input/output or analog input. The ATMega328P is an 8-bit microcontroller that uses 32KB of flash memory to store the program code even after being turned off. The ATMega328P setup on the Arduino board makes it easier to start a project and make a minimum viable product (MVP), because of the many functions. Through the use of nine pins, we connect the Arduino to a Shift register (21), four transistors (22), and a 4 digit 7 segment display (16). That has the function of showing a preset date. To detect if there is a low light level, EG at night, a photoresistor(20) will notice and send a signal to the Arduino to dampen the light levels of the display. The Arduino is also connected to a red and green LED. These LEDs are used to show if a berth is free, closed, or if an unwelcome boat is there. Besides this, the Arduino's receiving and transmitting (TX/RX) ports are connected to an ESP (18) so the signals from the ESP can say what the Arduino needs to do. The Arduino Uno also acts as a power source for the board and ESP through either a cable or battery.
The ESP is connected to the web server, where it will get its instructions from. It's also connected to the Ultrasonic sensor (19), which will detect if a boat is in the berth and send that signal back to the server. To connect the ESP with the boat device, the ESP is using a TX/RX 433 Mhz module (17) that will speak between the two devices. To improve the range of TX/RX module, we used a 173 mm wire to get a range of over 30 m. The range is most likely greater, but for the case of a boat leaving or entering a berth, we did not see a need for experimenting over greater distances.

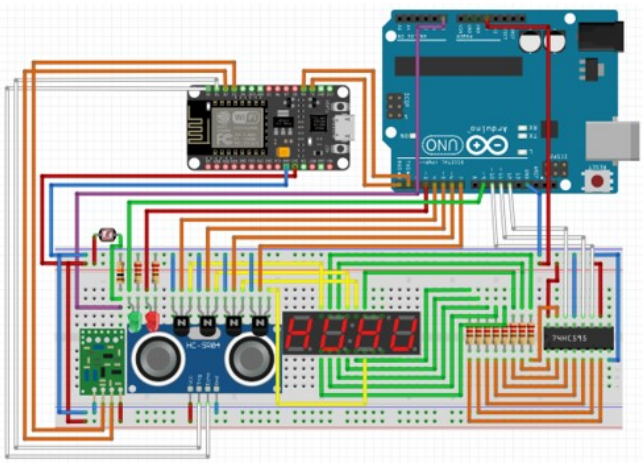

Figure 2: Overview of the circuit diagram of the setup on a breadboard for the berth module. For a bigger picture look in appendix 8.

Below, you will see the components used in the project. For pictures and links to the component, look in appendix 9.2.

Component	Amount used
Arduino Uno	1
ESP	2
4 digit 7 segment display	1
Transmitter and Receiver (TX/RX) module	1
Ultrasonic sensor	1
Photoresistor	1
Shift register (74HC595)	1
Transistor (PN2222)	4
Breadboard	1

2.5 Case design

For the case design, we needed a simple case that could split in two and at the same time show the LEDs, display, and give a hole for the antenna, photo-resistor, ESP, and Arduino input. To make the design we started by making some measurements of the breadboard, Arduino and ESP. We then used the program Fusion 360 to make a 3D model of the case. This was made easier by implementing a photo of the breadboard and components to give an idea of where to make the cutouts. With the help of online data sheets for the components we could size the cutouts correctly with extra room 1-2mm to ensure that the components would fit through. The case could then be 3D printed and fitted with the breadboard and components. It is designed so the Arduino Uno is under the breadboard, but can still be reached with cables through the case.

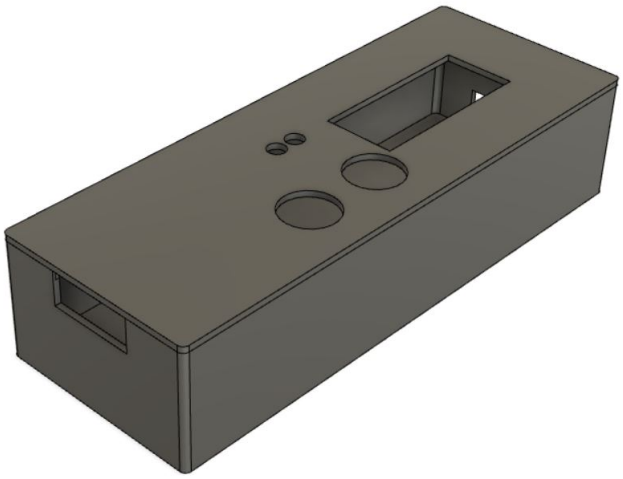

Figure 3: Here we see the case design in Fusion 360

2.6 Software design

The software was designed in 3 modules to realize our desired data flow displayed in 4.

Figure 4: The data-flow for our system

2.7 Communication protocol

To allow the different devices to talk seamlessly with each other, we developed our own communication protocol. It consists of 6 digits, each having a different function. The exact function of each digit can be seen in figure 5.

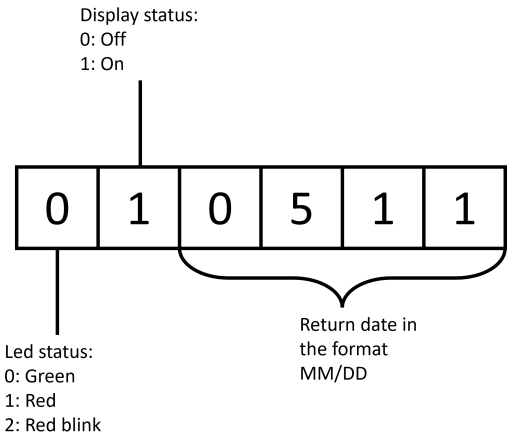

Figure 5: The return code we have created for our system.

2.7.1 Web server

The web server was built in Python using FastAPI for the back end, SQLite for the database, Starlette for the mediator, and plain old HTML, CSS, and Javascript for the front end. These frameworks and languages were chosen due to their simple nature, which allows for rapid development.

The user interface (UI) was designed using the mobile-first principle. This will in the future lead to better placement on Google, and a better experience for the majority of users seeing as mobile devices make up the vast majority of web traffic.
The UI consists of three main views. They can all be seen in figure 6. The left view is for the harbor master. Here the payment status of the berths occupied by guests can be updated. The view in the middle is for the guests. It has two styles: The graphical one in the figure, and a list view. The right view is for the berth holders. It allows them to update the return date of their berth, and see the occupation status of it.

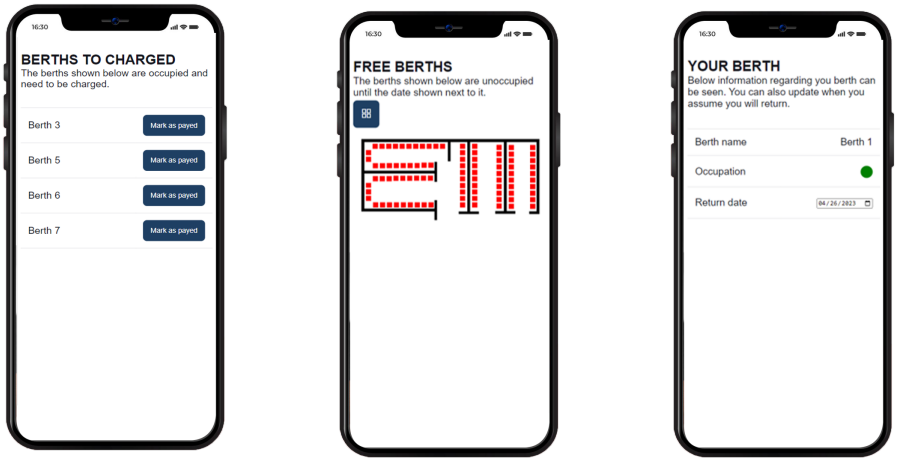

Figure 6: The different views of the user interface. The intended users are left: Harbor Master, middle: guest, and right: berth holder.

The last functionality of the web server is the API which allows the berth device to call it using a POST request. This is done using the endpoint api-url/update_berth_status. The request should contain two books: One for the occupation status of the berth and one describing whether it is the main boat that is occupying the berth. Furthermore, it should also contain the UUID of the berth making the request.
Using this data and the flow chart in fig 7 the return code, and actions the web server must take are calculated.

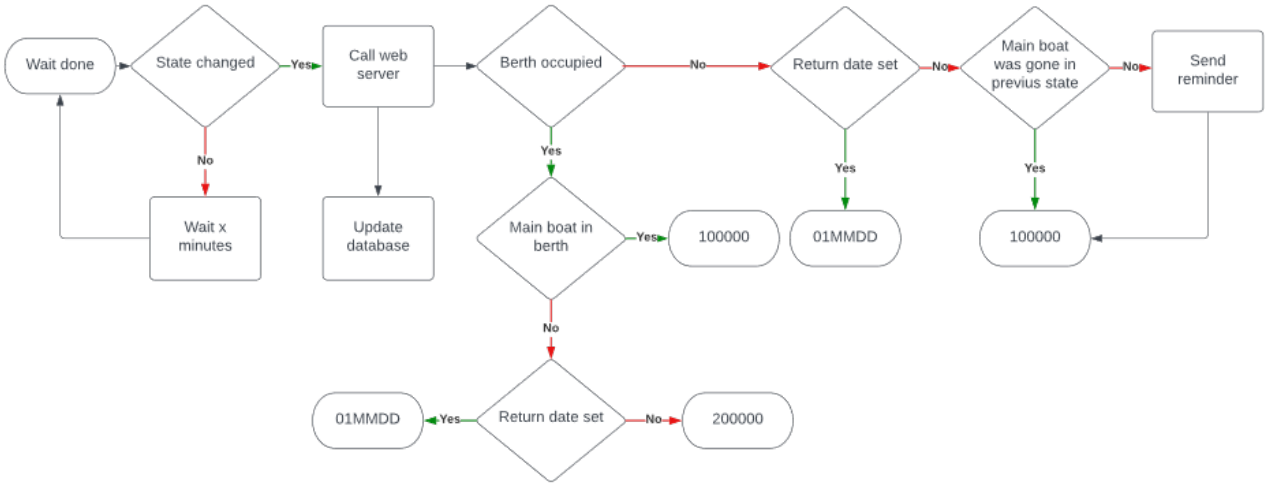

Figure 7: The flow chart used to determine the return code of the endpoint api-url/update_berth_status, and the actions the server must take.

2.7.2 ESP8266 code

The ESP8266 was coded while having it be the link between Uno and the web server in mind since it has the ability to connect to WiFi. At initialisation, the ESP connects to the local WiFi specified in the code. When a connection is established, it turns on the receiver for the RX/TX 433MHz Transmitter. The ESP starts each iteration by sampling its two boat sensors, the ultrasonic distance sensor and the RF 433MHz receiver. Afterward, it formats a JSON object containing the current value of the sensors and POSTS it to the web server. The server responds with a response code, that is passed through the ESP to the Uno by an RX/TX connection.

2.7.3 Arduino Uno code

The Arduino code is pretty straightforward, it receives the response code from the ESP, which determines the state of the 7-segment display and the LEDs. The Arduino also reads the input of our analog sensors, the photo-resistor, to update the brightness of the display.

3 Application Requirements

We have analysed some of the possible requirements our system would have, if it were to be deployed in the real world. As with many other IoT applications, it is essential that the device cost is low. Often this is important due to the sheer volume of devices to be deployed. However, in our case, there is a special emphasis on low cost, due to the fact that Danish marinas are typically not run with large budgets and high profit margins. Therefore, there is a strong need for the system to be affordable. The devices will have to be able to support bidirectional communication, though the application is not time critical and it does not need to support high datarates or large package sizes. For simpler hardware, the star topology could be a good network configuration, as devices do not need to be capable of forwarding information. The limited size of marinas and their relatively wall-free layouts also help facilitate this topology configuration. Most piers where the devices would be mounted tend to have power available, which would eleminate the need for batteries. One downside of the devices being mounted on a pier is the exposier to costal climate, which the devices would have to be resilient towards.

As marinas are typically close to towns and cities and are used by many people, the devices would also have to be able to tolerate interference from other network traffic in the area.

4 Tests and results

We tested all the functionalities of our system and they all succeeded. This includes an RF test, a test of the display, a test of the LEDs, the ultrasonic sensor, the communication between the web server and the berth device, and also the function of the photoresistor.
To test the LEDs, display, and photoresistor at the same time, we set a date for the berth holder's boat to return. This then showed that the display worked and the green LED lit up. When the photoresistor was covered by an object, we could see that the light dimed on both the display and LED. A challenge here is that the framerate of the display changes, which is a weakness. However, the framerate change isn't noticeable to the human eye and is only visible on video.

To test our ultrasonic sensor, we set the berth device to occupied so a red LED would light up. We then put a "boat" without the boat device on it, to show it was not the berth holder's boat. The LED would then start to blink red. This indicated that our ultrasonic sensor was working. However, we noticed that the sensor would not respond if the boat was too close, about 10-15 cm. This could be a weakness for the system.

One of the functionalities that has not been not fully tested, is the RF system. To test the signal strength we set up our berth module at one end of a 30-meter-long hall with a line of sight, meaning that there were no obstacles. We then moved the boat device through the hall and never lost signal, which could be seen by our LEDs turning from red to green. To make sure it wasn't an error, we disconnected the TX module from the boat device at the 30-meter mark, where the LEDs would change. To test it further we would, at the 30-meter mark, go through a side hall that had a thick brick wall where the signal quickly died out. This made it so we could conclude that we could receive a signal from a range of 30 meters with the system in the line of sight, which is decent. However, That might not be the case for a boat, since the sensor is to be expected to be on the inside of the boat. This could be a problem, but if we choose to roll out this product, we will look at other ways to implement the communication between berth holders' boats and berths.

5 Tecnical problems and challenges

One of the first obstacles we encountered was the number of pins the 4-digit 7-segment display (16) used from the Arduino Uno. In our first test of setting up the display, we used 12 pins, one for each of the pins on the display. We solved this problem by using a shift register (74HC595 (21)) and four transistors (pn2222 (22)). This made it possible to use only 9 pins instead of the 12 we used before.
Another challenge was the transmission, we spend a fair amount of time confused about why we were barely able to send a signal with the RF system. The error was in the antenna itself. It was not long enough, for the wavelength to be correctly propagated through the transmission line. To fix this problem, we ensured that our antenna length was long enough to hit the 1/4 wavelength of the signal, which is equal to around 17.3cm(17).

Where to have the logic of the system was another challenge. On the one hand, on-device logic would allow for less data transfer. This would allow the device to use less power, and be more secure. This would however make it more tedious to implement seeing as the berth device

is coded using C++. More importantly, it would also make it more difficult to make changes in the future since each device would have to be updated individually. We, therefore, choose to have the logic regarding the status of the system on the web server.

Lastly, how to communicate between the different devices presented a challenge. The requirements were that it should be simple and uniform for all parts of the system. We settled on the system with our own custom return code as described in section 2.7.

6 Future work

Throughout this project we have made a great MVP, however, there is still space for improvement. One of the first improvements would be to move the breadboard design over to a PCB. This would give us the ability to only use the number of functions we want and at the same time make the electronics take up less space. For this, it might also be interesting to look into a microcontroller with more cores, so everything can run on it instead of multiple microcontroller units.
While we are on the design, we would also like to look into how to make the case waterproof and more resistant to the unforgiving environment of the ocean.
We would also like to look into making a low-powered model. This will make it cheaper for the harbor to use the product. This could be done by changing the 7-digit display for a display using a tube and some motors. Alternatively, solar cells could be added to offset the power usage.
The user interface could also be expanded by adding payment directly on the web server, and the ability to reserve a berth. This would further improve the experience for the guest. Furthermore, the reminders which are currently sent via email could be sent via SMS or perhaps push notifications if the current web app is turned into a proper app.
Another task, for future work, is to make the system more secure. We can start by making users for the berth holds and harbor administrators. This would make tampering with the system harder. Furthermore, guests could also be required to sign up to reserve berths. Thereafter we can start by implementing bigger security solutions such as encryption. Lastly, proper authentication between the berth device and web server would also improve the security vastly.
We will also look into mounting solutions for the boat device. This can either be done by giving the boat owner a small box that they can put on their boat. For some boats, it might even be possible to implement it into the boat itself through software. Additionally, we would also like to look into ways of detecting the presence of the berth holder's boat which does not require power on the boat. I.e. a powerless boat device.

7 Conclusion

In conclusion, our project succeeded, everything we sketched out as ideas were realized. Our system is fully functional, and we're able to simulate a harbor with a boat entering and leaving. We were able to communicate with a boat at an open-air distance of 30 meters. This product is ready to be implemented in the harbors of Denmark, but before that is done, we have some adjustments we want done e.g. creating of a PCB for the circuit, encryption and UI-optimization. We have talked about realizing this project in the real world, since an IoT solution like this, is yet to be seen in the harbor industry and the current system is outdated and screams for an modernization.

8 Team reflections

In this project, we have learned a lot about product development and the use of IoT and microprocessors. None of us had ever touched an Arduino before, but we quickly learned it and now we all understand the value of being able to easily prototype an IoT solution. Planning-wise, we learned a great deal about structuring an IoT solution and an engineering project in general. We started out with a brainstorm specked with crazy ideas. We quickly narrowed it down and planned what to do, who did what, and when it should be done. This step allowed us to ensure that the majority of tasks could be done independently of others' progress which meant that our workflow did not end up following the waterfall model. We all agreed, that this is not something that we train enough as engineers. Usually, the scope of our projects is already given by the teachers, but it was fun to develop it ourselves this time.

9 Appendix

9.1 Who did what

	Eigil	Jonathan	Lucas	Thomas
Berth module	90%	10%	0%	0%
Boat module	10%	10%	80%	0%
Case design	10%	90%	0%	0%
Display	10%	90%	0%	0%
Communication (Boat to berth)	0%	0%	100%	0%
Communication (Berth to web server)	0%	0%	60%	40%
Webserver	0%	0%	0%	100%
User interface	10%	0%	0%	90%
Return code	100%	0%	0%	0%
State flow chart	10%	0%	40%	50%
Pressentation	25%	25%	25%	25%
Powerpoint	22.5%	22.5%	22.5%	32.5%
Report	25%	25%	25%	25%
Testing of device	25%	25%	25%	25%
Video demo editing	0%	100%	0%	0%

Table 1: Attribution table.

9.2 Hardware and case setup

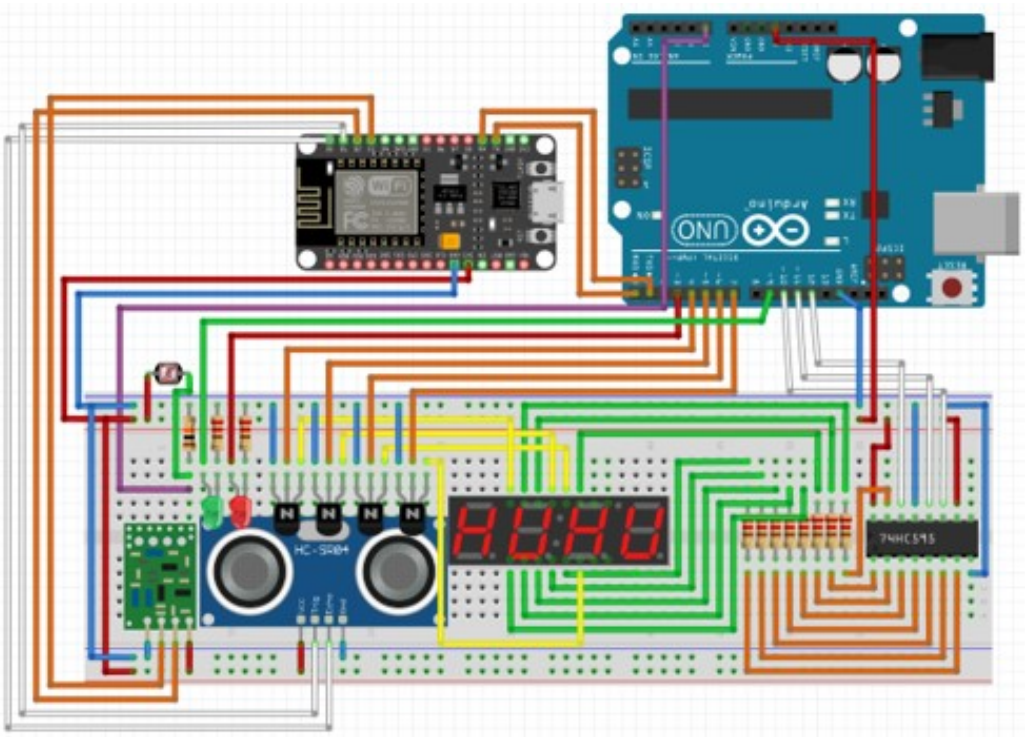

Figure 8: Overview of the circuit diagram of the setup on a breadborad for the berth module.

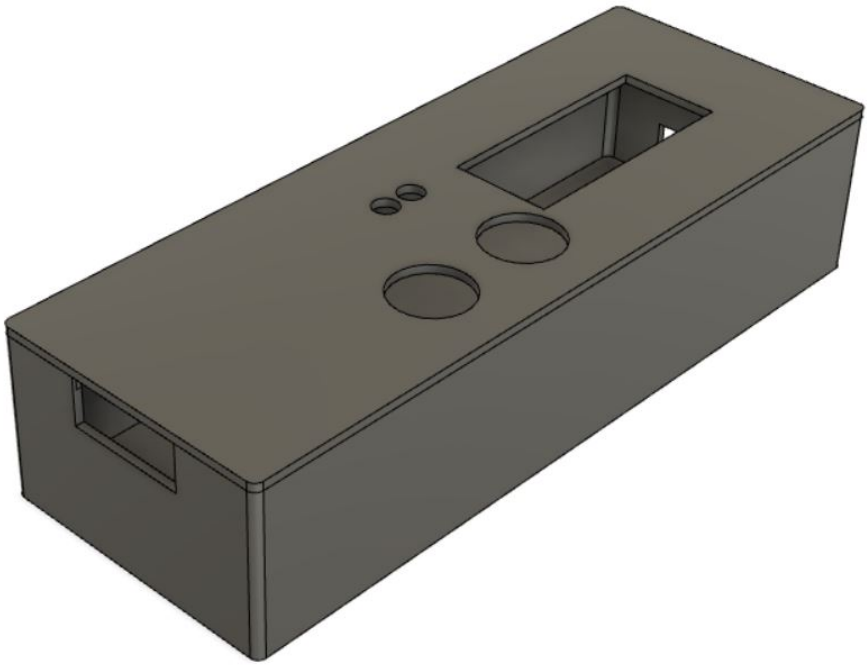

Figure 9: Case design

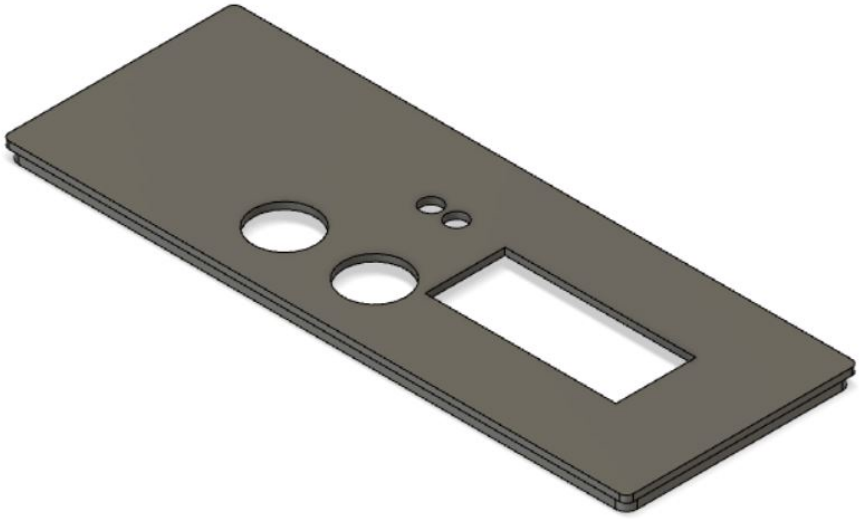

Figure 10: Case top part

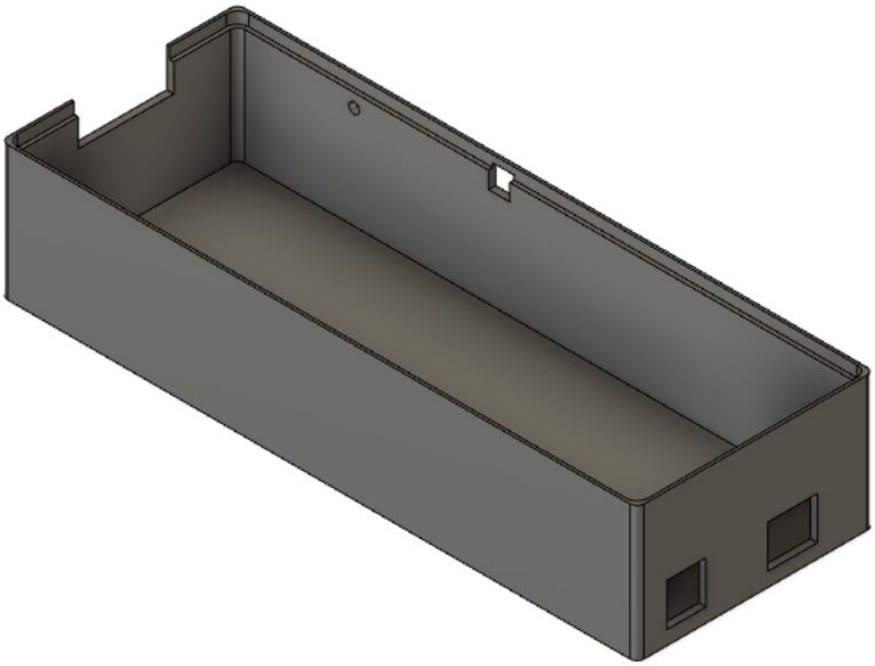

Figure 11: Case bottom part

Figure 12: Breadboard in the case. Note that the Arduino Uno is under the breadboard

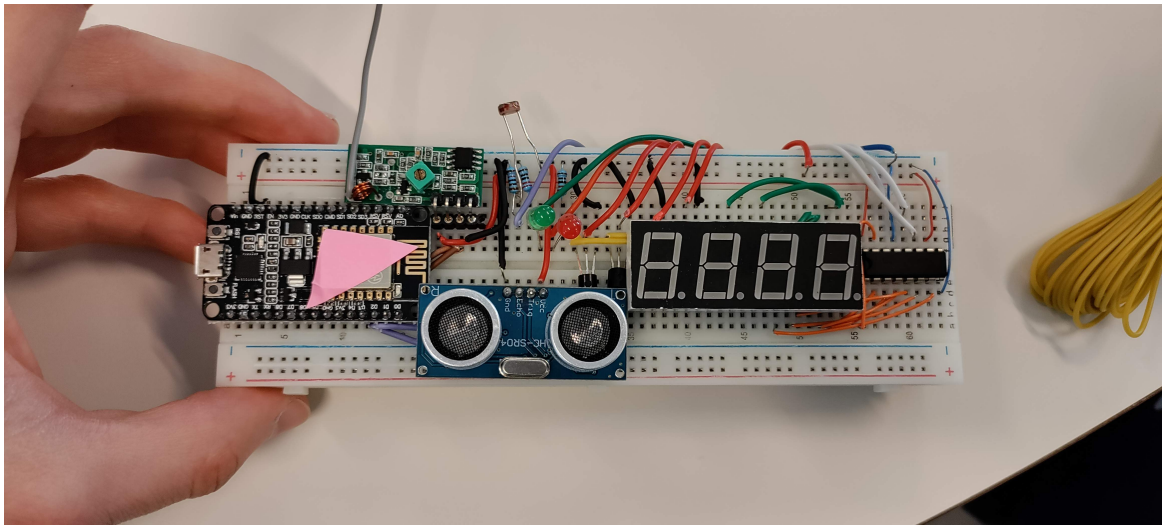

Figure 13: Breadboard without the case. Note that the Arduino Uno is under the breadboard

Figure 14: Breadboard in a closed case, displaying the date 30. April.

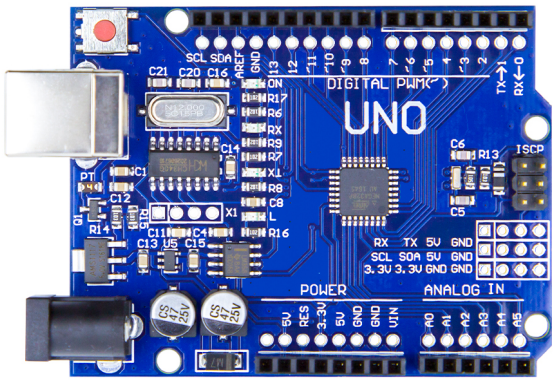

Figure 15: Arduino Uno, Link: `https://arduinotech.dk/shop/improved-version-uno-kit-starter-saet-rfid-learning`

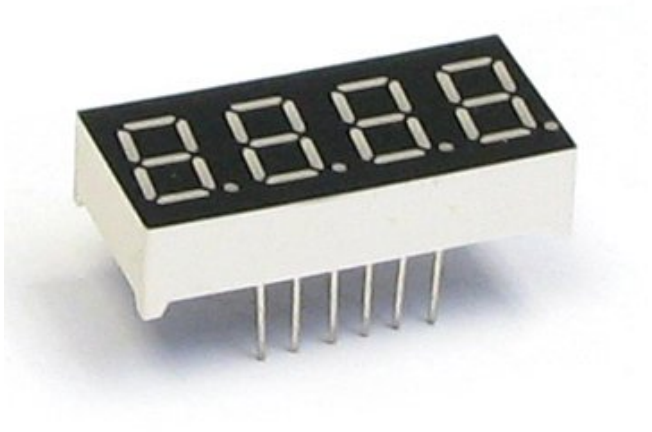

Figure 16: 4 digit 7 segment display, Link: https://arduinotech.dk/shop/4-bit-tube-led-display/?gclid=CjwKCAjw9J2iBhBPEiwAErwpeQRIq6FipWJxqatZR3FLcBUVyXZXR1maQ
kENM3ZbsiOLgQBoCW64QAvD_BwE

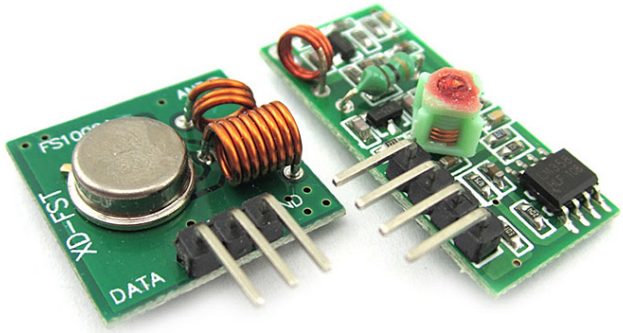

Figure 17: TX/RX module, Link: https://ardustore.dk/produkt/wireless-transmitter-and-receiver-433mhz-wireless-module?gclid=
Cj0KCQjw8e-gBhD0ARIsAJiDsaXT8Y7vpMVidYoPs-ckz5CNkP4GTcuhYPD7htKKuCu7XQiGAloPQ0AaAopUEALw
wcB&fbclid=IwAR0PC1OZ4nPPQmKhlVSXBl95XvKWufUSOHCUMTcql37gCPizVZuIL1OopA4

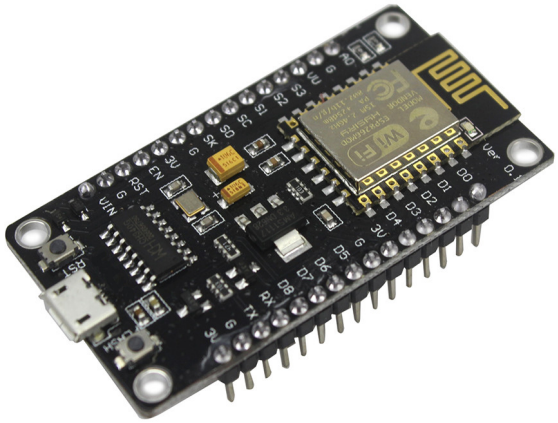

Figure 18: ESP, Link: https://ardustore.dk/produkt/nodemcu-esp8266-lua-30pin-ch340g-udviklingsboard?gclid=Cj0KCQjwmN2iBhCrARIsAG_G2i50bdxzWYVcG8TA4B7-yF54Shzy2nu6KtnFsY7gxj4g_X3Zm1BjsrYaAtBKEALw_wcB

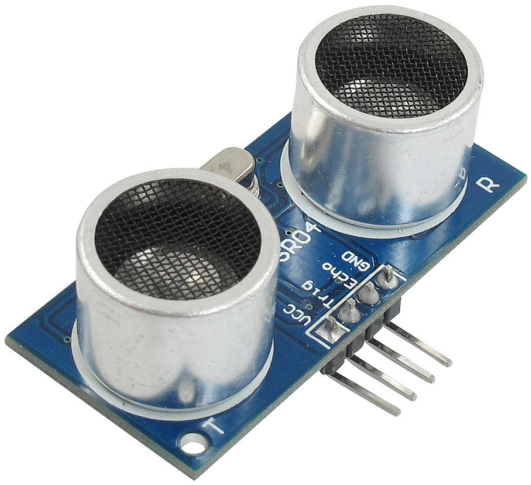

Figure 19: Ultrasonic sensor, Link: https://ardustore.dk/produkt/hc-sr04-module?gclid=Cj0KCQjwmN2iBhCrARIsAG_G2i75cSWGH73iLuDF-ELOWW5J_bQjO4LnLfdrQhQA_SrDkO5M2P3krIMaAvEhEALw_wcB

Figure 20: Photoresistor, Link: https://ardustore.dk/produkt/photosensitive-resistance-ldr-sensor?gclid=Cj0KCQjwmN2iBhCrARIsAG_G2i6xAk-rKROO6e_P0bjjjZkZhtefiC9N_r98caJucreTqvbHxyo0k0IaAg64EALw_wcB

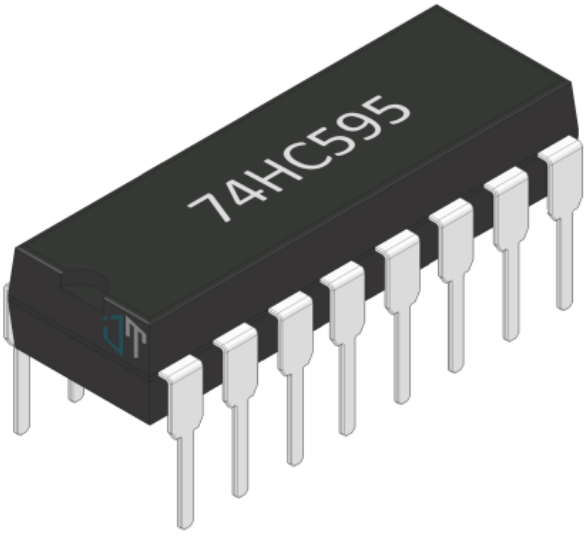

Figure 21: Shift register, Link: https://jentronic.dk/integrerede-kredslob/80-74hc595.html?gclid=Cj0KCQjwmN2iBhCrARIsAG_G2i6pCE9JLKLyRRmnddLU0wfUakEmxtsn0shstaMkhVaXaU1I40spnWYaAtQ0EALw_wcB

Figure 22: transistor pn2222, Link: `https://www.componentsinfo.com/pn2222/`

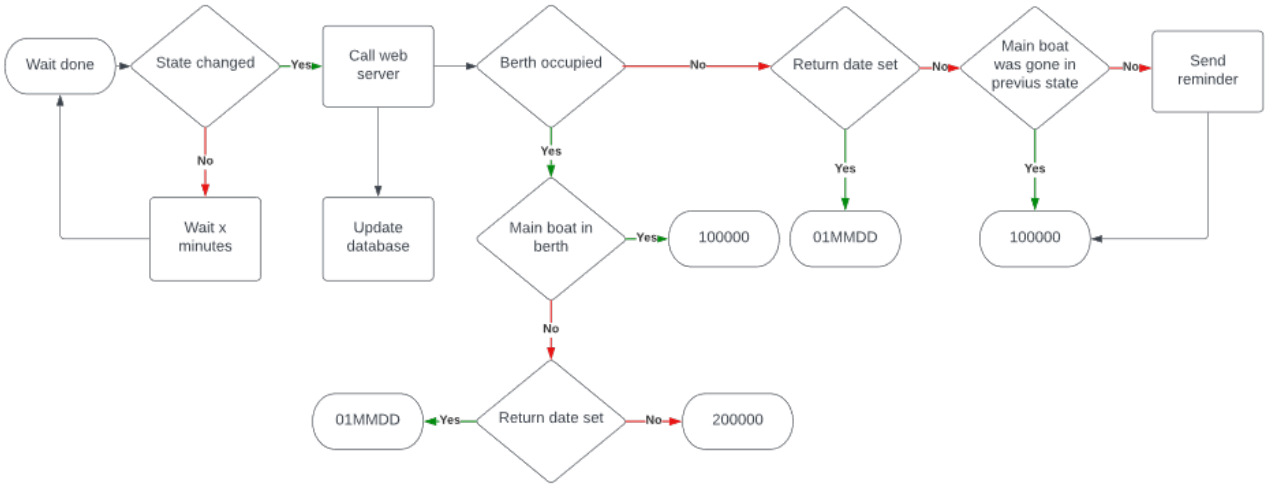

Figure 23: The flow chart used to determine the return code of the endpoint api-url/update_berth_status, and the actions the server must take.