Federal University of Technology of Paraná – UTFPR
Academic Department of Electronics – DAELN
Academic Department of Informatics – DAINF
Computer Engineering
Integration Workshop 3 (EEX23) – S71 – 2024/1

Technical Report

Cat Amusement and Tap Station

Augusto Carvalho de Freitas augustofreitas@alunos.utfpr.edu.br
David Luiz Becker de Souza davidsouza@alunos.utfpr.edu.br
Erika Burei Alves erikaburei@alunos.utfpr.edu.br
Henrique Nelson da Silva Vanin henriquevanin@alunos.utfpr.edu.br
João Mathias Ravazzi Martins joaomartins.2000@alunos.utfpr.edu.br
Willian Becker de Souza willian.1998@alunos.utfpr.edu.br

July 2024

Abstract

Many cats suffer from health problems caused by not drinking enough water. The main organ affected is the kidneys, which play the important role of filtering waste and toxins from the blood. Thus, a lack of water not only leads to chronic dehydration, but can also decrease kidney function and increase the risk of crystals and kidney stones forming. Based on the solution to this serious problem, Cat Amusement and Tap Station was created, a solution that addresses feline hydration challenges by combining water supply with interactive engagement. Designed to meet cats demanding preferences, this multi functional device features a two-liter water storage capacity, filtration and cooling mechanisms to ensure fresh, appealing water. Integrated interactive elements such as laser toys, a ball launcher and a snack dispenser, remotely controllable by pet owners, enhance the hydration experience. Real-time video monitoring allows remote observation and interaction, promoting a strong bond between owner and pet. Thus, the Cat Amusement and Tap Station not only offers a solution for felines water intake, but also improves their overall health and brings them closer to their owner.

1 Introduction

Cats are notoriously independent creatures and can often be a bit finicky when it comes to their preferences, especially regarding drinking water. However, the lack of adequate water intake can lead to a series of kidney problems in cats, posing a serious concern for pet owners.

To help prevent kidney problems related to inadequate water intake, cat owners should ensure their pets have access to fresh, clean water at all times.

The Cat Amusement and Tap Station was designed to provide a suitable solution to the problem in question. Storing up to two liters of water, the device

not only keeps the water moving, but is also capable of filtering and cooling it, details that are of great importance for attracting cats to water. In addition, the project includes other types of attractions, such as laser toys and a ball launcher, as well as a snack dispenser, which can be controlled remotely by the guardian. The device also has an audio system, where predefined sounds and even sounds recorded by the user can be played.

Also, with a view to the need for monitoring and the importance of interaction between the pet and its guardian, the device has a real-time video system where, even when far away, the owner can monitor and interact with their kitten.

1.1 Overview

As already mentioned, Cat Amusement and Tap Station is a multifunctional solution to feline health problems. Featuring modules such as a water fountain, laser, camera, snack dispenser and ball launcher, the project turns drinking water into a fun and attractive environment for cats. In addition, it brings the pet closer to its owner, as well as better monitoring of the cat when the owner is away.

To develop this solution, the project was divided into modules: snack dispenser, ball launcher, water fountain, camera and laser, communication, mobile app and electronics.

The following block diagram shows the interaction between all the parts.

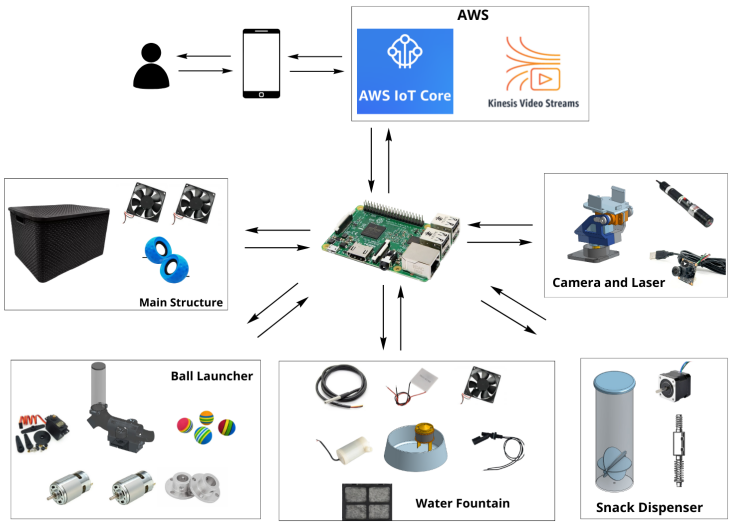

Figure 1: Block diagram of the project

As can be seen in Figure 1, the main component is a Raspberry Pi 3 single-board computer, which interacts with and controls all the modules. The water fountain was developed to provide good quality water for the feline, so a filter,

cooler and pump are included in the module. The camera and laser are attached together, as it is necessary to have a view of where the laser is pointed. The support that attaches them has the possibility of movement, which is controlled via the app by the user.

The ball launcher is responsible for promoting movement in the cat by launching a ball with a chime, while the snack dispenser releases treats. These can occur at times programmed by the user via the app or by a manual command.

The user can visualize their kitten, interact with it and feed it remotely, all via a mobile app. The communication between the mobile app and the physical device is done via an Amazon AWS server.

2 Project Specification

The project specification was based on three types of requirements: functional requirements that correspond to the functions of the project, non-functional requirements that are the characteristics and anti-requirements that describe what the project does not propose to do.

The following are some of the main requirements of the project.

2.1 Important Requirements

- **FR1**: The device shall have a drinking fountain (where the pumped water will come out), located at the top of the reserve tank.
- **FR7**: There must be an opening to give the cat access to the water in the reserve tank.
- **FR57**: The app will have a screen that displays functionalities related to ball lancher, snack dispenser and the water fountain to the user.
- **FR58**: The app will have a screen that allow the user to control the camera, the laser, the ball launcher and the audio player remotely.
- **FR62**: The app will alow the user to schedule the ball launcher and the snack dispenser and the audio player.
- **FR64**: The app will show message status about the device's operation (ex: temperature below or above certain degree, some scheduled action that has been made).
- **FR12**: The device must activate the cooling element if the measured water temperature is above 25°C.
- **FR13**: The device must deactivate the cooling element if the water temperature drops below 18°C if the cooling element is active.
- **FR23**: The camera image can be viewed via the app.

- **FR26**: The laser can be moved by the user through the app.
- **FR30**: The device shall be able to play sounds.
- **FR37**: The device must launch balls at least 1m in distance.
- **FR48**: The device must be able to release portions of snacks.
- **FR56**: The system should allow the user to interact with the functionalities through an app.
- **FR74**: The device shall have a backup battery for the Raspberry Pi fitted in a battery shield.
- **FR77**: The system must send commands over the internet to control the features.

3 Development

This section deals with the development of the project. It is divided into the modules that were used to develop the project, so that all the team members could work in parallel. Thus, the mechanical, hardware and software parts were divided into: main structure, water fountain, snack dispenser, ball launcher and support for the camera and laser. In addition, one of the team members was directly involved with the electronics part of the project and another with the development of the application.

3.1 Mechanics

The first part of the development concerns the mechanical part. Several surveys were carried out to choose the materials used, as well as the components themselves. Some of the components were adapted from other objects that had other functions, as was the case with the general structure, which used to be an organization box. In addition, many modules required specific components, such as the ball launcher, which was entirely modeled and produced using 3D printing. This development will be discussed in detail below.

3.1.1 Water fountain

The design of the fountain structure was made so that all the requirements were met. The main part is the inner circumference, which is the structure responsible for the water flowing into the tank. As defined by the requirements, at the base of the top of the fountain there are small holes, which allow the water to pass into the tank. The base of this circumference is open so that the water can move throughout the tank. Another important point is that the structure has been designed so that the cat can have access both to the water coming out

of the top of the fountain and directly into the tank, allowing the animal to drink even when there is no power.

Figure 2 shows a 3D render of the water fountain.

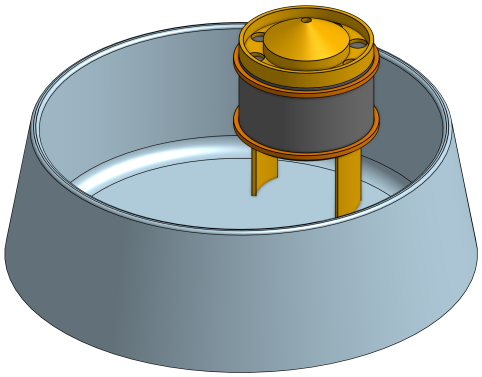

Figure 2: 3D render of the water fountain

A standard cat water bowl was chosen as the water container, as it already met the size specifications. Cuts were made in this container to integrate it with the other components of the module, such as the hole for the water level sensor. Materials were used to seal off the water around the cuts.

3.1.2 Snack dispenser

A cereal dispenser previously obtained by the team was used as the base of the structure. To be able to rotate the axis and release the items, a stepper motor with a shaft coupler were used, which is controlled by the APP, allowing the user to choose the routine for releasing the snacks.

One measure that was taken to further prevent snacks from getting stuck was to add a funnel that avoids creating too much weight on top of the paddle wheel.

Figure 3 shows the 3D render of the snack dispenser structure.

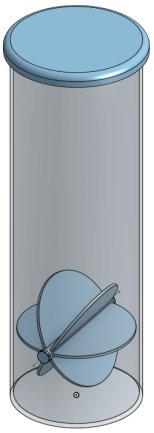

Figure 3: Snack dispenser - 3D

3.1.3 Ball launcher

The module's structure was 3D printed and contains a compartment to store the balls that contains internal rattles to attract the cat's attention, a servo motor to serve as a lock preventing the balls from falling out of the compartment, two DC motors connected to 2 wheels with a rubber material to have more friction and be more effective when launching the balls.

The DC motors are 12V and are connected to a H bridge and are controlled by a PWM module, to adjust the rotation speed of the motors. The user can choose to interact with the ball launcher via the APP and choose to release a ball anytime, or create a schedule to release the balls at specific moments of the day. Figure 4 shows the 3D render of the ball launcher model.

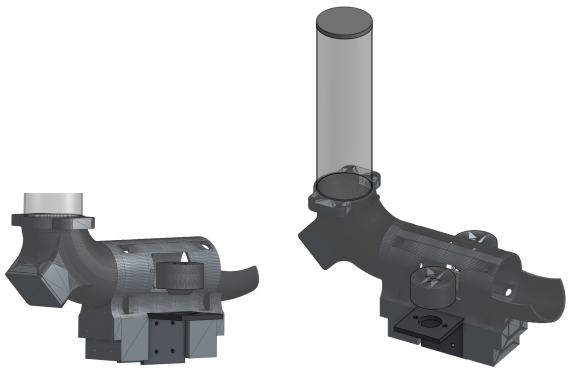

Figure 4: 3D render of ball launcher model.

3.1.4 Camera and laser movement support

The support used for the camera and laser came with four separate pieces that, when assembled, create a robust structure for movement both horizontally and vertically.

This support was developed to be used with two micro servos, one at the base is responsible for horizontal movement (side to side) and the other at the top is responsible for horizontal movement (up-down). Furthermore, the support has space to fix the camera and laser on its support piece.

Instead of 3D printing our own structure, a similar structure was purchased and used. Figure 5 shows the structure assembled with the two servos.

Figure 5: Structure for the camera and laser.

3.1.5 Main structure

As mentioned, the general structure was adapted from an organization box. The positioning of the components was defined and the cuts and holes in the structure were made. The part of the structure most at risk of getting wet was also sealed off.

With regard to positioning, there were some subtle changes so that positions could be found where all the parts of the project could perform their functions effectively. This was the case with the ball launcher, which was initially going to be on the side of the structure, but as the project progressed it became clear that it would be best to place it at the top so that it could launch the balls as expected. Other modules, such as the water fountain, remained in the same place as initially defined.

In addition, another aspect that was considered relevant when choosing the positioning of the modules, as well as their correct functioning, was the size and preference of the project's clients, the cats. Thus, characteristics such as the height of the cats were observed, and it was concluded that the height is on average 23cm-25cm. This is relevant because it was found that the ideal height for a cat to drink water depends on a number of factors, such as the size of the cat's breed, the shape of the bowl and the cat's drinking habits. In general, the water

should ideally be at the height of the cat's chest or elbows. This allows them to drink without having to bend down or squeeze, which can cause discomfort and even digestive problems.

Based on all the studies carried out, the best positioning was then defined, seen in Figure 6.

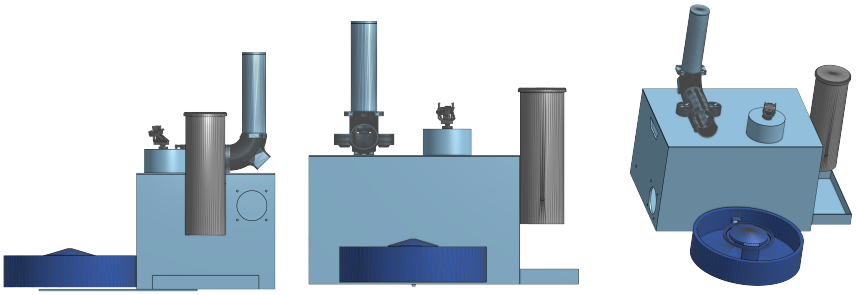

Figure 6: 3D render of the main structure from different angles

Also, the general structure was made to avoid the cat from touching the internal components inside the box, and thinking in the air cooling system we used two fans, one for intake and one for exhaust.

Finally, the final assembly can be seen in the Figure 7.

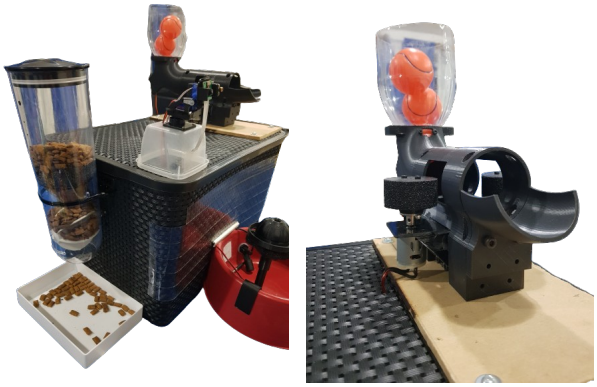

Figure 7: Final assembly.

3.2 Eletronics

The Figure 8 shows the full electronic schematic of the project.

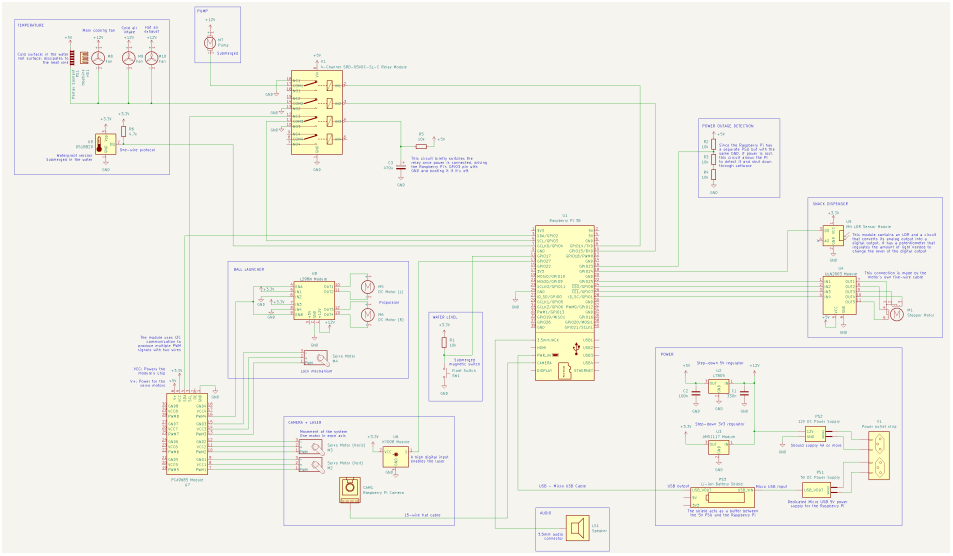

Figure 8: Full electronic schematic

The project is centered on a Raspberry Pi 3, which is responsible for controlling all the peripherals and handling the data flow between them and the cloud server. The pinout was chosen based on the characteristics of the GPIO pins and of the components used, which will be explained further.

Since the Raspberry Pi 3 works with 3.3V logic and can be harmed by higher voltages, all components that can operate with 3.3V are connected as such. The stepper motor and the servo motors require 5V, so the correspondent regulator supplies them power. The circuit is powered by a 12V DC supply, while the Raspberry Pi has its own dedicated 5V supply and a battery buffer in between. The battery used is a Li-ion 3.7V battery on a shield that steps up its voltage to 5V. Both power supplies are connected to a power strip.

As the Raspberry Pi is powered by a dedicated supply with a battery buffer, once the power is disconnected, the computer will continue working while the value read by the correspondent pin changes from 3.3V to 0V. This allows the Raspberry Pi to detect the outage and safely shut down before its battery runs out. This is the reason for the battery, to avoid abrupt power offs that can corrupt the OS.

The Raspberry Pi 3 does not have enough PWM channels to handle all the components that require it, a PCA9685 module was chosen, as it uses I2C communication to control up to 16 independent PWM channels. Connected to it are the two servo motors responsible for the movement of the camera and laser, the servo motor of the lock mechanism of the ball launcher and the h-bridge that controls the DC motors of the ball launcher. This h-bridge is an L298N module, with both directions permanently set and the velocities controlled by the same PWM signal. The laser is controlled by a single digital wire, while the camera is

not represented as it is plugged to a dedicated slot on the Raspberry Pi, not to the GPIOs.

The circuit for the snack dispenser is composed of a stepper motor, driven by an ULN2003 module, and an LDR module to detect the level of snack. If it detects light, meaning the snack is below a pre-established threshold, the digital output of the module will indicate so. The water level is detected by a float switch with a pull-up resistor. The chosen thermometer communicates with the Raspberry Pi through the one-wire protocol, which is fixated on GPIO4 in the board, so the component must be wired to this pin.

There are four relays, each of the first three enabling one of the indicated components. The fourth is responsible for booting the Raspberry Pi once power is supplied. If powered off by software, the Raspberry Pi has two ways of booting again: by reconnecting the power or by applying 0V to the GPIO3 pin. However, this pin is used by the I2C, so the relay controls this switching. For a few moments after power is restored, the GPIO3 pin will be connected to ground and will boot the Raspberry Pi. After that, the relay switches the pin to the I2C pin.

Many components use a printed circuit board as a centralized hub to handle its connections with each other. Since most cannot be directly soldered to the board, it has pin headers to which wires can be soldered and then routed to the components.

3.3 Software

In this section, the functioning of the system's communication architecture is be explained in the first two subsections. This architecture is used to operate features such as video streaming and remote control of system modules.

3.3.1 General Communication Architecture

- **Video Streaming**: The system uses the Raspberry Pi camera to stream video, allowing it to be viewed through an app via an internet connection. This is achieved using AWS's Kinesis Video Streams service, which enables real-time video transmission via the WebRTC API. This API allows for point-to-point connections to exchange audio and video with minimal delay. Communication occurs between network points, with a master node discovering connected viewer points, and viewer nodes interacting only with the master node. The architecture also includes signaling channel and TURN/STUN channel concepts. The overall architecture can be seen in Figure 9.

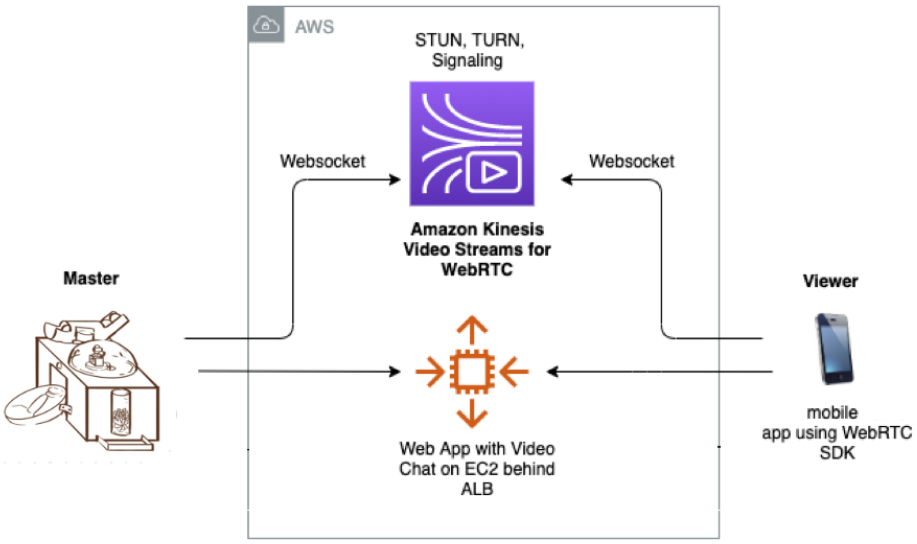

Figure 9: Communication architecture for video streaming

  - **TURN Channel**: A TURN (Traversal Using Relays around NAT) channel is essential in WebRTC-based systems when devices are behind firewalls or NAT networks, making direct communication difficult. The TURN channel acts as an intermediary, aiding data transmission when peer-to-peer communication is not possible. It functions like a server, receiving data from one device and retransmitting it to another. If device A cannot connect directly with device B, device A sends data to the TURN channel, which then forwards it to device B, enabling efficient communication even when direct connections are blocked.
  - **Signaling Channel**: A signaling channel for WebRTC is an essential component to facilitate communication between participants in a WebRTC session. While WebRTC itself focuses on transmitting real-time data such as audio and video, the signaling channel is responsible for coordinating and controlling communication between clients. This channel performs functions such as: session negotiation, establishing and closing connections, and managing events and errors.

- **Remote Control of Features**: In addition to video streaming, the system allows the user to remotely control (via the internet) features such as: positioning the laser and camera, launching balls with the ball launcher, releasing portions and snacks with the dispenser and playing audio. For such functionalities, the AWS Iot Core service is used, which uses the MQTT communication protocol. Figure 10 shows the communication architecture for remote control of functionalities.

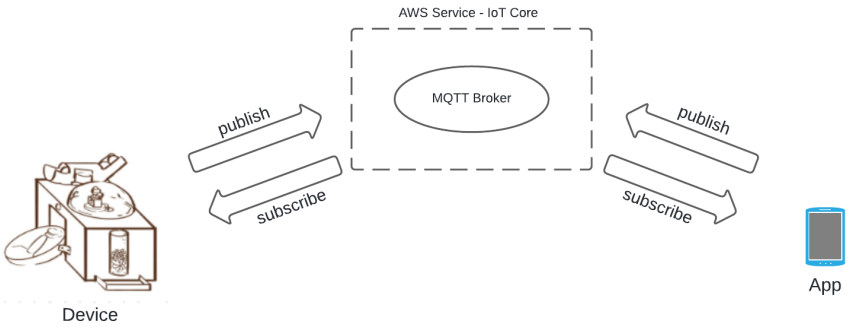

Figure 10: Communication architecture for remote control

MQTT is a lightweight messaging protocol widely used for IoT applications, operating on a topic, publisher, and subscriber basis. In the MQTT Broker (in this case, on AWS), several topics (logical paths) are established, each corresponding to a different functionality. Devices can publish to or subscribe to these topics. When a publisher updates a topic, all subscribers to that topic receive the new information. For example, a water temperature sensor can publish its readings to a "water temperature" topic on the broker, allowing the app subscribed to this topic to remotely view the sensor's temperature measurements.

3.3.2 Communication Module Design

- **Firmware**: The raspberry is subscribed to MQTT topics that are used by the app to send messages and will publish to topics to send messages to the app. On the left we have the main flow where sensors are read and messages are sent to the app. On the right we have the response function when a message arrives from the app.

- **App**: The app's communication module (to control the project's functionalities) is implemented through a class called MQTTClient. This class encapsulates the MQTT client used to communicate with the AWS broker. The class has the basic attributes for connection establishment and MQTT message exchanges. The most important methods are on_message and publish.

  The on_message method will be triggered every time a topic (to which the app is subscribed) changes in the broker. This way, when the solution's firmware makes a publication on the cats/alertWaterLevel topic, for example, this method will be automatically executed and the app will be able to perform the appropriate actions.

The publish method is used to publish to an MQTT topic, this way the firmware will be notified of the change and will receive the message sent by the app. For example, the app might post to the cats/launchBall topic to notify the firmware that the user wants to launch a ball with the Ball Launcher.

3.3.3 Audio Module Design

- **Firmware**: The firmware must also handle the audio module. The tasks to be performed were divided into three procedures: playSound, deleteFileSound and includeFileAudio. These functions will be called in the MQTT callback function (when a publication occurs in a topic to which the device is subscribed).
- **App**: The app must be able to play audio on the smartphone, give commands to play audio on the device, delete audio recorded by the user and allow the recording of new audio. Figure 6 shows the flowchart of all procedures that will be used to carry out such actions. The idAudio that appears in some functions is generated in the button click event. The called publish functions publish messages to MQTT topics (send messages to the device).
- **Audio file synchronization**: The app allows the user to play audio on the device and record new audio. For this to be possible, the audio recorded in the app must be sent to the device and there must be a way to synchronize the audio present in the system (audio inserted in the app must be inserted in the Raspberry and audio deleted in the app must be deleted in the Raspberry). The topics cats/fileAudio and cats/delAudio are used for audio synchronization.

3.3.4 Schedule Communication Design

Communication of the system's scheduling function was implemented through two MQTT topics: cats/setSchedule and cats/delSchedule. The Raspberry is subscribed to these topics and the app is the publisher of these topics.

Every time the user sets a schedule in the app (for example, releasing a portion of snacks at 3:00 pm) the app will publish it in the set topic and Raspberry will receive the information, configuring the functionality for the specified time. The format of the message to be sent will be <time># <functionality>, for example "15:00#dispenser". With this, Raspberry will be able to interpret the information and set the functionality. When the user deletes a booking record, the app will publish it in the deletion topic and Raspberry will receive the information and delete the booking. The message format will be the same as in the set case: <time>#<functionality>.

3.3.5 Water fountain software designs

This section shows the designs of all the software related to the water fountain, such as pump shutdown, temperature control and water level warning.

- **Pump shutdown**: The device prevents the pump from turning on when the water level is below the safe level. In addition the software detects when the water level passes the minimum and turn off the pump if it is on.

  The code flow first sends a request to start pumping water, the water level sensor is checked and then depending on whether the water is at a level below or above the determined level, the pump is switched on or off.

- **Water level warning**: The device checks via the water level sensor whether the container with water is at a lower level than the one determined, and if it is, it sends an alert to the application. The check is carried out every ten minutes.

- **Temperature control**: The device maintains the water temperature between 18ºC and 25ºC. The check takes place every five minutes, a temperature sensor is used and its firmware is programmed to activate or deactivate the cooling system.

3.3.6 Laser and Camera control

The movement of the structure containing the laser and camera involves not only the movement of the motors, but also the communication between the Raspberry and the application, since the movement information is sent from the application to the topic in the MQTT broker, in which the Raspberry is subscribed. There are two forms of movement for this structure, one of which corresponds to the standard movement, which occurs when the laser is on but has not been moved by the user for more than a second. This movement takes place only horizontally, so that the servo responsible for movement in this direction moves from the center by 5 degrees to each side, alternately, while the vertical position remains as it was. The other form of movement occurs when the user sends commands via the app, where the movement information received is processed so that the servos move appropriately.

3.3.7 Interface Software

- **Interface functionalities design**:

  The app was developed in React Native using the open-source tool Expo for faster, more robust application and native components that is easier to manipulate. Expo is an open-source platform for making universal native apps for Android, iOS, and the web with JavaScript and React. For the database connection, it is implemented locally in the mobile device and using the free library SQLite because its more simple and direct for the application usage.

  SQLite is a database engine written in the C programming language, it is a library that software developers embed in their apps. In specific, the software was implemented using a library called react-native-sqlite-storage that is a plugin developed for mobile applications.

  Finally, The application also provides a track record of actions and notifications received by the device, such as high temperature levels or low water levels in the fountain.

4 Results

The results of the Cat Amusement and Tap Station were satisfactory. We spent several months working on the project and producing each module so that, in the end, we could integrate several functionalities in one place. All the mandatory requirements were met, as well as some of the optional requirements.

During development, some adaptations were made so that the project could meet what was required. That said, the difficulties in the integration encountered will be presented below.

- **Mechanics**: The mechanical parts corresponding to the water fountain, the laser and camera structure and the ball launcher worked correctly when integrated with the other modules in the project. The only remaining mechanical problem is with the snack dispenser, which generates some snack crashes while it is releasing them. This problem has been greatly mitigated by the software, which rotates the motor in both directions so that the stuck snacks can be released.
- **Electronics**: During the integration of the electronic part, some problems were detected which caused delays for this deliverable. The main one was a short circuit on the board, which caused the 5V regulator not to work, so it was necessary to correct this part of the circuit. Later, it was noticed that this same regulator was unable to supply

enough current to the peltier, causing problems in this module of the project, to solve this problem was necessary to adapt the circuit placing a separate 5V supply to power only the peltier.

There were also problems with the H-bridge power supply, responsible for powering the ball launcher's DC motors, which requires around 2 amps for each channel. The source's output wires burned out because of the high current flowing through them, making it necessary to use thicker wires to supply the H-bridge.

With these problems solved, it was possible to integrate all the electronics into the project.

– **Software**: With regard to software integration, some modules, with the exception of the water fountain, such as the ball launcher, the snack dispenser, and the camera and laser structure, required adaptations for them to function correctly. In addition, the predefined audio software was also implemented and works well. The biggest problem found was with the video streaming software.

Although there were difficulties during the development of the project, the team managed to overcome them. In the end, all the mandatory requirements were completed. In addition, 57,14% of the optional requirements were also implemented. Some statistics in this regard are shown in the figure below.

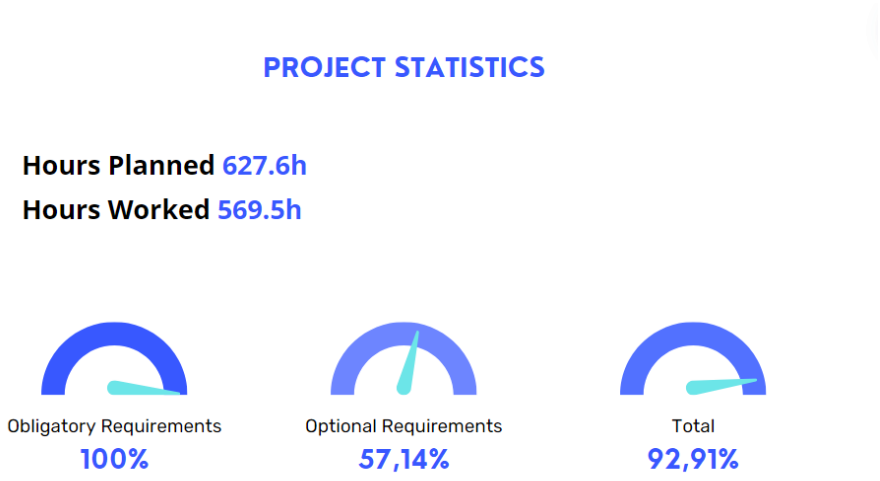

Figure 11: Project statistics

The entire development of the project was recorded on the following website: C.A.T.S.

Images of the final state of the project have already been shown in the explanations of the modules in section 3. Below, Cat Amusement and Tap Station actually being used.

Figure 12: C.A.T.S. being used

4.1 Budget

The initial budget for the project was R$1190.87, but because of unforeseen circumstances, the total rose to R$1344.14. The final amount includes more powerful motors to fit the needs of the project.

4.2 Schedule

The project began in April and finished at the end of June. There were deliveries during this time:

- **Deliverable 1**: Mechanical design;
- **Deliverable 2**: Full Mechanical project. Electronic design;
- **Deliverable 3**: Full electronic project. Software design;
- **Deliverable 4**: Software project;
- **Deliverable 5**: Mechanics, hardware and software integration;
- **Deliverable 6**: Overall integration and functional tests.

In the first deliverables, the team managed to complete all the tasks proposed in the schedule, so the on-time deliverables were: deliverable 1, deliverable 2 and deliverable 3.

From deliverable 4 onwards there were delays, which amounted to 13 hours for deliverable 4, 15 hours for deliverable 5 and 20 hours for deliverable 6. The reason for the delays was the high complexity of the modules and the difficulty in integrating them.

Figure x shows the estimated hours + 30% and also the actual hours worked on each deliverable.

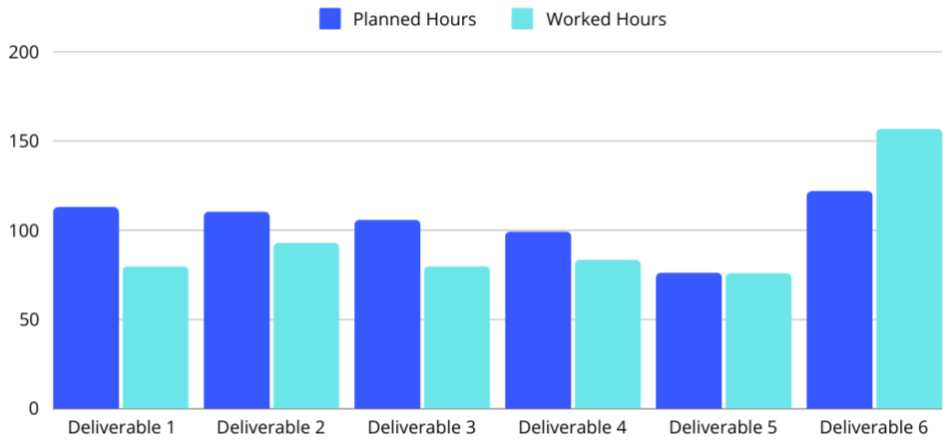

Figure 13: Hours Planned x Hours Worked one ach deliverable

5 Conclusions and FutureWork

5.1 Conclusions

The development of the Cat Amusement and Tap Station provided learning in various sectors, as the project includes completely different modules and in the end they were all integrated in one place.

Despite all the difficulties encountered over the course of the days, the team managed to overcome them through a lot of effort and help from friends. The main stumbling block was video streaming, which was implemented efficiently after more than 50 hours working on the functionality.

It's worth noting that the weeks leading up to the start of the project were essential, as it was through them that all the stages of the project were planned in detail. In addition to defining the requirements, setting the timetable and dividing up the tasks, the risk plans also played a key role in the project's success, since they already defined the measures to be taken in the event of incidents.

Finally, getting the Cat Amusement and Tap Station project up and running with all the functionalities correctly exceeded all expectations. Every

stage of planning was essential for success. As was every stage of repair and improvement.

5.2 Future Work

To further improve the project, some of the possible improvements are:

- Use rechargeable batteries instead of a direct connection to the power supply;
- Standardizing the colors of the project;
- Identification of cats.

Acknowledgments

Our honest gratitude to everyone who helped develop the Cat Amusement and Tap Station.

Especially to Rene Lopes for printing and helping with the Ball Launcher's structure, to Thiago Henrique Medeiros for helping with the video streaming implementation.

References

[1] C.A.T.S.. https://www.notion.so/caatss/C-A-T-S-69b36da9a9d64c878644f96304fe3b71.

[2] Raspberry. https://www.raspberrypi.com/.

[3] OnShape. https://cad.onshape.com/.

[4] International Cat Care. "Understanding Feline Chronic Kidney Disease". https://icatcare.org/advice/chronic-kidney-disease/.

[5] Petlove. "Como fazer meu gato beber água?". https://www.petlove.com.br/dicas/como-fazer-meu-gato-beber-agua.

[6] Petz. "O que fazer quando gato não bebe água?". https://www.petz.com.br/blog/pets/gatos/gato-nao-bebe-agua/.