Federal University of Technology – Paraná – Brazil – UTFPR
Academic Department of Electronics – DAELN
Academic Department of Informatics – DAINF
Computer Engineering
Integration Workshop 3 (ELEX22) – S71 – 2023/2

Technical Report
Mage Hand - Gesture Controlled Candy Vending Machine

João Felipe Sarggin Machado– joaomachado@alunos.utfpr.edu.br
João Guilherme Martins Silva – jsilva.2018@alunos.utfpr.edu.br
João Vítor Dotto Rissardi – joaorissardi@alunos.utfpr.edu.br
Pedro Sodré dos Santos – psantos.2000@alunos.utfpr.edu.br
Thiago de Mendonça Mildemberger – thimil@alunos.utfpr.edu.br

November 2023

Abstract

The advancement of electronics, information technology and robotics has made it possible for many processes to be automated. Therefore, the purpose of this project is to automate a process that until then remained mostly manual, the sale of candy in bulk. Mainly in shopping malls in large cities, there are several kiosks and stalls that sell candy in this way, which makes the process reasonably common. This, summed up with the fact that the process is pretty straight forward, creates enough potential for automation. Thus, this document describes Mage Hand, a candy in bulk vending machine that allows the customer to buy their candy automatically without the need for any prior requirements, such as a phone app, for example, and someone to attend him at the time. Furthermore, the machine still seeks to give the customer the same feeling as a manual purchasing procedure, where they gradually place the sweets in their bag, while preventing them from coming into contact with the candy before paying and also providing an attractive and fun interface that allows him to operate the machine from a distance just using hand gestures.

1 Introduction

Accordingly to the 2023 State of Treating yearly report [1], candy sales reached $42.6 billion in the total market in the year of 2022. Even though this number is boosted by inflation and still suffers from the effects of the COVID-19 pandemic, there is no denial that selling candy is still a very profitable market.

A not very big, but still considerable, share of this market is in selling candy in bulk, it being inside kiosks of shopping malls, or in another types of stores. This process of selling candy in bulk is quite straight forward. The customer sees all the candy types available and their prices, select some, slowly put them in a bag, pay for them, and finally leave.

Taking both this considerations into account, it is not hard to see that the process of selling candy in bulk can be automated, leading to improvements in general profit and client satisfaction, since it is not necessary to have an attendant to sell the candy in the process.

This is the main motivation behind the construction of Mage Hand, a vending machine capable of selling candy in bulk in an automatic way. However, it is important to notice that automating this process is not simply replicating all the steps mentioned above with a machine doing it instead of a person. There are some other co-factors that need to be taken into account to make the whole process fully fledged out.

That's why Mage Hand also have some secondary objectives, those being: keeping the candies inaccessible to the customer until his payment is confirmed, avoiding theft; allowing the customer to slowly pour the candy in a cup while he is seeing how much he is buying in real time; allowing the customer to reject an order, i.e, discard all the candies in his cup so a brand new order can start at any time; avoiding wasting good candy and guaranteeing all rejected candy can be put back in the machine; creating a fun and attractive interface that doesn't required the customer to have a previously installed app; having a good owner interface that allows the owner of the machine to customize it and at the same time receive feedback from it.

It is this objective of creating a fun and attractive interface that also adds another motivation for the client to place an order that is not simply buying candy that resulted in the idea of using hand gestures to control the machine from a distance, which ended up being it's biggest differential.

1.1 Solution Overview

With the purposes and context mentioned above in mind, it is possible to start describing the solution for the introduced problem. Therefore, Figure 1 presents a high level block diagram of the whole proposed system.

As we can see, the project is divided into 5 big blocks, with a Raspberry[2] PI 4 micro-controller as it's core. First of all, the Machine has a pretty complex mechanism to deliver the candies. That's why those have been abstracted here as only "Archimedes Screw" and "Candy cups", and will be described in more details later.

Before starting to unravel each component, it is important to say that the machine will be limited to only 2 different types of candy, and is part of it's domain restriction that these are necessarily small types of candies, like M&M's for example.

It is easy to perceive that some kind of storage will be needed to keep safe the candies that can be bought. Because of that, some sort of mechanism that removes candy from the storage and put them in a visible cup is also necessary.

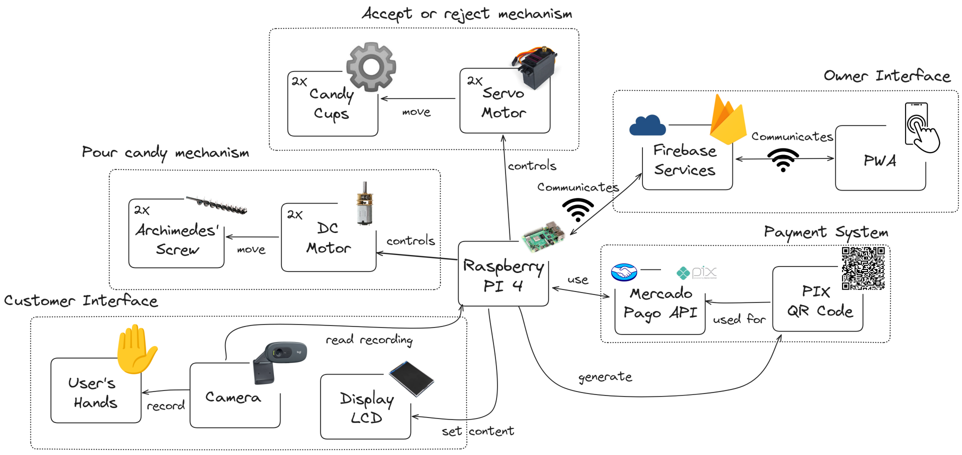


Figure 1: High level block diagram of the system. Source: The authors.

This mechanism will be composed of an Archimedes screw[3] that will carry the candies from the storage to the cup. This Screw will be turned by a N20 high torque micro DC Motor which will be activated in the right time by the Raspberry Pi. Two mechanisms of this kind will be needed since there are two types of candy.

At the same time, a different mechanism to retain the candies in the middle of a buy and then delivering them to the customer once he pays is also important. This mechanism will too be responsible for moving the candies to a secondary and inaccessible internal storage if the order is somewhat canceled. It will be composed of cups for holding the candies and servo motors controlled by the Raspberry Pi for turning those cups. There will be 2 cups, one for each candy, and they will be turned one way for accepting the order and another for rejecting it.

As for the payment system, the Mercado Pago API[4] will be used, since it provides all the structure needed for performing real payments and it is also for free. Outside of that, all that is required by the project part is to be able to communicate with the API, done by Rest architecture, and generate a proper QR code that allows the customer to pay for the order with the PIX system of his bank account.

The customer interface will be composed of a camera to record his hand in order to be possible to detect his hand gestures, and a LCD display to help guiding him throughout the whole process, indicating what gesture he needs to do next, what are the available candies, how much of it is available, how much it costs and how much the client is currently buying. The display will also show to the client the PIX[5] QR code at the end of the process.

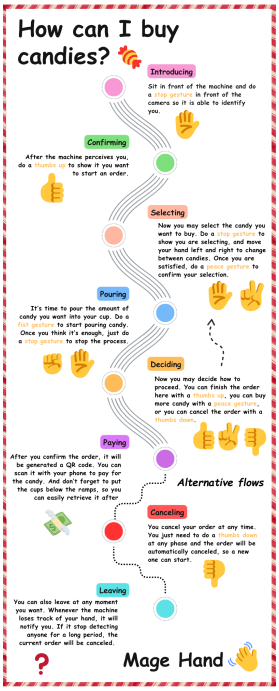
The real-time gesture recognition process will be key to the project. This is what will compose the fun and innovative interface mentioned and allow the customer to control the machine in a "touchless" way, i.e, from a certain distance without having to press any button or move any lever. This also means a gesture code will be necessary. The customer will need to do certain gestures at certain times. An overview of this code can be seen in Figure 2b.

The owner interface will be made in form of a Progressive Web App (PWA) [6]. This PWA will communicate with the Raspberry Pi using various Google Firebase services[7]. The interface will allow the owner to set the name, price and image of the candies he intends to sell, to see the history of all orders placed sorted by day, load his own Mercado Pago access token so he can direct all payments for his account and change his password, so no one else can have control over the machine except him. The app will also be needed in order for the owner to input how much candy he is replenishing into the machine, so it knows the current volume. That not being enough, the app will also provide feedback from the machine in form of notifications.

Finally, the Raspberry Pi 4 will be the heart of the project. It will be responsible for creating the communication channel with the owner app, controlling each electronic component (motors and display), using the Mercado Pago API to create real payments and most important, reading images from a camera and processing them in order to try to identify what and gesture the customer is making.

A quick view of the machine that was build based on this whole concept explained here can be visualized in Figure 2a.

(a) Final result overview



(b) Machine use instructions

Figure 2: Final machine overview and gesture code. Source: The authors

2 Project Specification

Since the specification of this project was done using the System Engineering methodology, it is quite long, comprising around 140 total requirements. Given that, it would be fitted to add all those requirements in a document of this proportion. That's why all the project specification, with all requirements, can be seen in the link: Project Requirements[1].

Outside of that, what can be done is give a quick explanation of the requirements tags. The project specification was divided into tags. Every requirement has at least one tag. The tags represent which part of the project the requirement refers to and of type is the requirement. These tags are: Functional and Non Functional (to specify the type of requirement); Mechanical, Hardware and Software; Candy Containers, Candy Collecting Mechanism and Candy Delivering Mechanism (to specify the stage/part of the physical structure); Introduction Phase, Selection Phase, Pouring Phase, Payment Phase and Finishing Phase (to specify the flow and the phase of the buying candy process); General (to see overall requirements); Owner App, Data Storage, User Interface and Gesture Recognition (to specify the part of software).

Besides the requirements, the project also has the anti-requirements, that are shown at the link: Project Anti-Requirements[2].

3 Development

Now it is possible to describe in more details the development of the project, that being the final mechanical design, electronic design and software design.

3.1 Mechanical Design

First, have in mind that the mechanical design presented here is the third iteration of the original design. The first iteration was discarded due to it having very low chances of working. The second was significantly improved over time, resulting in the third one that can be seen in Figure 3.

As it can be seen, the machine is divided into 5 sections from top to bottom: The Candy Storage, the Candy Collecting Mechanism, the Candy Delivering Mechanism, the Main Body and the Base. Each part will be described in more details ahead. Most parts of the machine were made using MDF wood, and acetate sheets were also used extensively to add refinements in the candy flow.

[1] Link to the Project Requirements: `https://magehand.notion.site/82f517d65f1b4d64b34600bdd95a5cea?v=10aed0ba55414a13bceab13bd03ee73b&pvs=4`

[2] Link to the Project Anti-Requirements: `https://magehand.notion.site/9736c05523f7402199e639f03c383ba7?v=8d61c0feaad647fa8d890325ae4a1928&pvs=4`

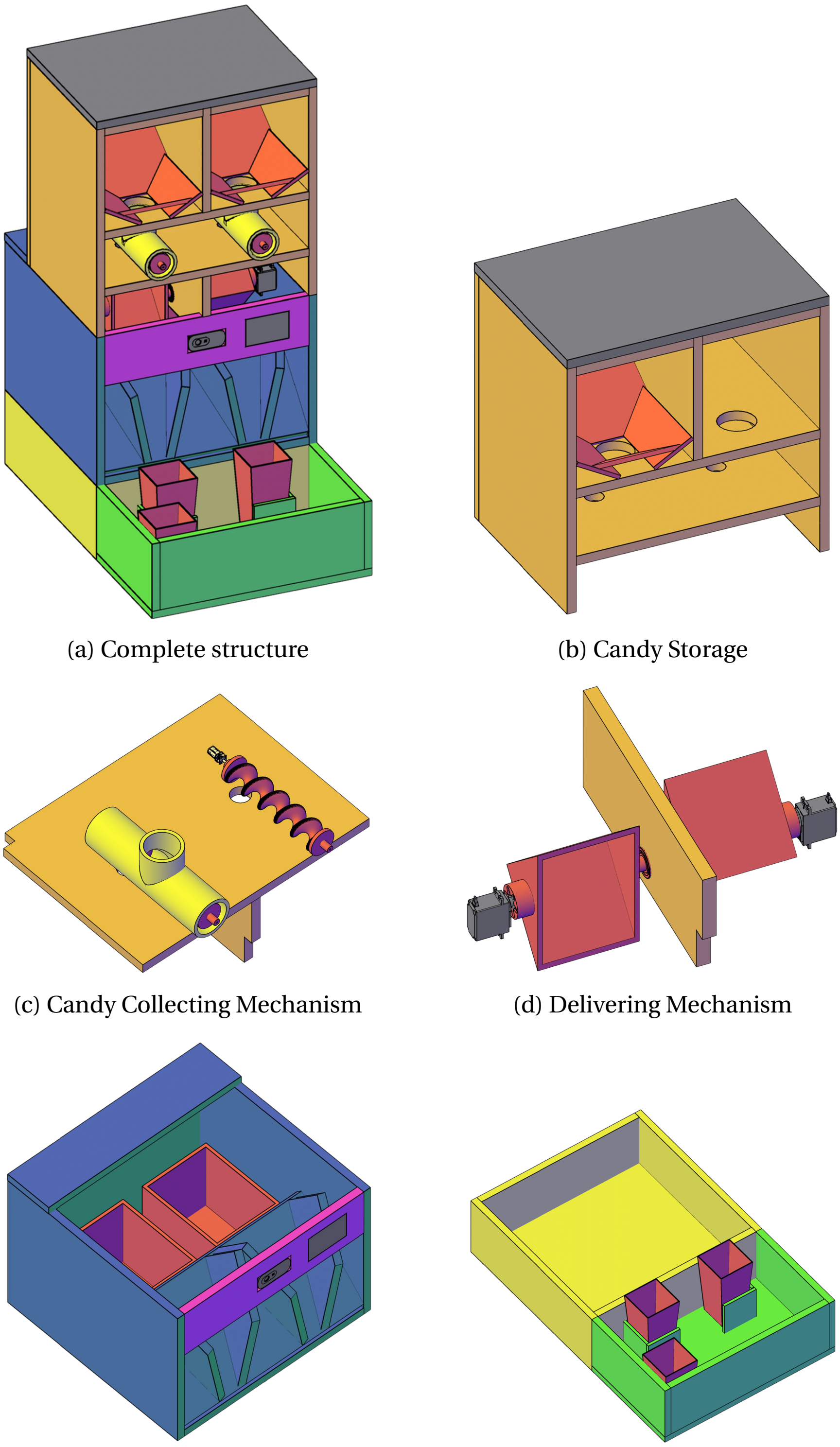
3.1.1 Candy Storage

The Candy Storage, which can be seen in Figure 3b, is basically composed of a enclosure made of MDF wood (in yellow), a lid locked by a padlock (in gray), a funnel also made in MDF (in orange) and an acrylic panel in the front. The purpose of this part is to hold all candies separately until someone decides to buy them. The acrylic is to make them visible to customer at all times. The lid is locked because only the machine owner should have access to the storage, so he can replenish it. At last, the funnel is for converging the candies into the bottom hole that ends up in the next part.

3.1.2 Candy Collecting Mechanism

The Candy Collecting Mechanism, visible in Figure 3c, is quite complex. It's main piece is an Archimedes Screw (Figure 4c) smelted in aluminum (in red). The Screw is settled inside a T PVC tube connector (in yellow). Covering the inside of the tube there is also an acetate layer to reduce friction between candies and the tube wall. To make it possible to turn the screw a DC motor is coupled in rearmost part of it. The front most part has an extension so that the screw is supported at both ends and can rotate in a fixed location. The motor is very tiny, and so can be fixated inside the tube by a 3D printed piece (Figure 4a).

The Archimedes Screw will be responsible for moving all the candies from the T connector entrance at the top to a hole made at the bottom a little behind when it rotates. The Tube will assure that the candies remain enclosed. Using this, with only the act of turning the DC motor on it is possible to move the candies from the storage, since they all will fall into the tube because of the funnel, to another opening that in fact in end up in the next mechanism.



(a) Complete structure

(b) Candy Storage

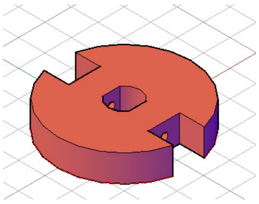
(c) Candy Collecting Mechanism

(d) Delivering Mechanism

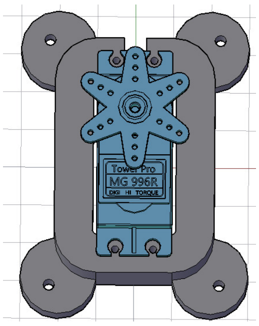
(e) Main Body

(f) Base

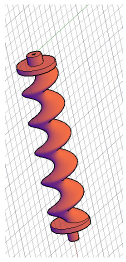
Figure 3: 3D view of the mechanical structure and mechanisms. Source: The authors



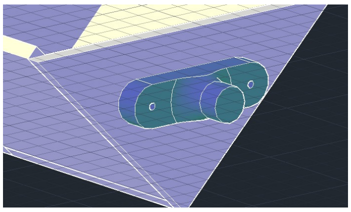
(a) DC motor support.



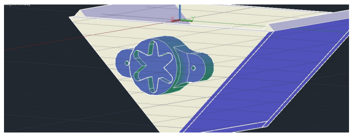
(b) Servo motor support.



(c) Archimedes Screw.



(d) Candy cups bearing support.



(e) Candy cups wall support.

Figure 4: 3D printed pieces. Source: The authors.

3.1.3 Delivering Mechanism

The cups seen in Figure 3d (in red) are the so called "Delivering Mechanism". They are also made in MDF and have a two layer coating inside it. One layer of EVA to cushion the candies when they fall into them from above, and one layer of acetate to reduce friction.

Reducing friction is so important because the main idea of this mechanism, which is also why the cups have this triangular format with 45° angles, is for the coupled Servo motors in the side (in gray) to be able to turn the cups full with candies 90° to one side when the order is "accepted" and 90° to another when it is "rejected". To allow this rotation, the cups are fixed to the middle wall (in orange) with bearings and the Servos are attached to the machine walls by 3D printed pieces (Figure 4d, Figure 4e and Figure 4b). With that in mind, it is possible to see that the were used servos because they provide precise control of the final position.

3.1.4 Main Body

The piece seen in Figure 3e is the main part of the structure. In it is possible to see the MDF piece (in pink) which hold both the camera and the display and both end destinies of the candies which are in the cups once it rotates. All blue parts are also made in MDF, with the front ramp being where the candies will fall when the payment is accepted. In the front of the ramps there will also be an acetate sheet to guarantee the candies are inaccessible and will always converge to the holes at the end. More acetate is used to enclose the candies in the front of the ramp hole and to cushion their fall from the cups. The red boxes behind are where the candies will fall when the order is somewhat rejected, being that an active rejection by the client, or simply the continuous lack of client detection.

This boxes assure that no candy is ever wasted, since the owner can easily open the back side of the machine, which is of course locked with a padlock, retrieve the boxes and put the candy back in the storage. In the back part, together with the boxes, will be fixed the project power strip. The motivations behind this power strip will be explained later. This also implies that the back lid will have a hole in where the power strip cable pass thought, allowing the machine to be powered by a single wall socket.

3.1.5 Base

Finally, the base comprises the last piece of the structure and can be seen in Figure 3f. It divide itself into two parts. The, per say, "basement" of the machine (in yellow) which is where the Raspberry, the Printed Circuit Board (PCB) and the computer power source are placed. This part has a hole in the top so the power cables can go thought to the power strip above. And the green part, which is completely open and accessible to the customer. It is where the customer can place the paper cups (in red) just below the ramps to receive his candy after the buy. The whole structure around the cups is for assuring that no candy ever falls into the floor even if they fall of the paper cups.

3.2 Electronic Design

The electronic components of the project are the 3.5” LCD display, the 2 N20 DC Motors, the 2 MG995 Servo Motors and the Logitech C270 camera. Since the Raspberry Pi 4 is equipped with several USB ports, the camera is connected directly to one of those. This is the reason the camera don’t show up in the Schematic of Figures 5, 6 and 7.

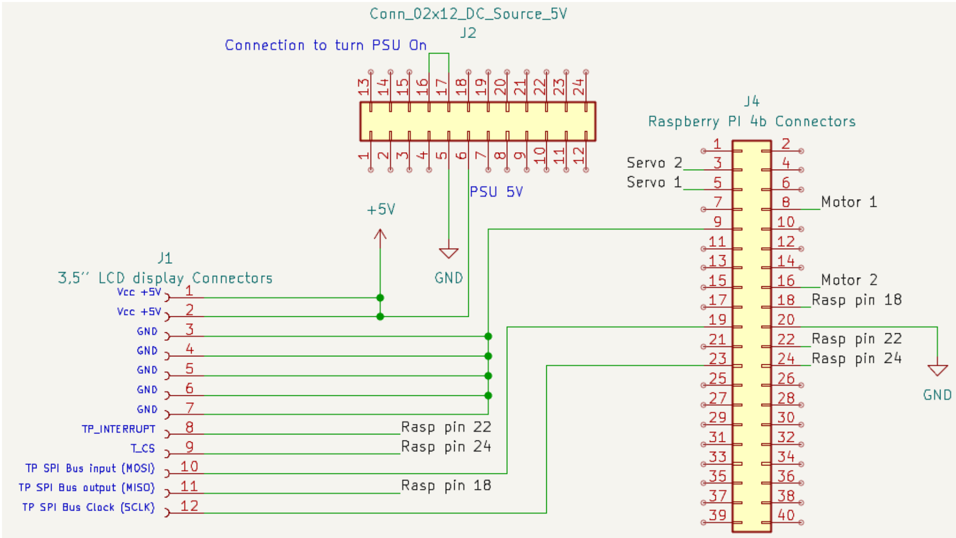


Figure 5: Electronic diagram: Raspberry Pi and Display. Source: The authors

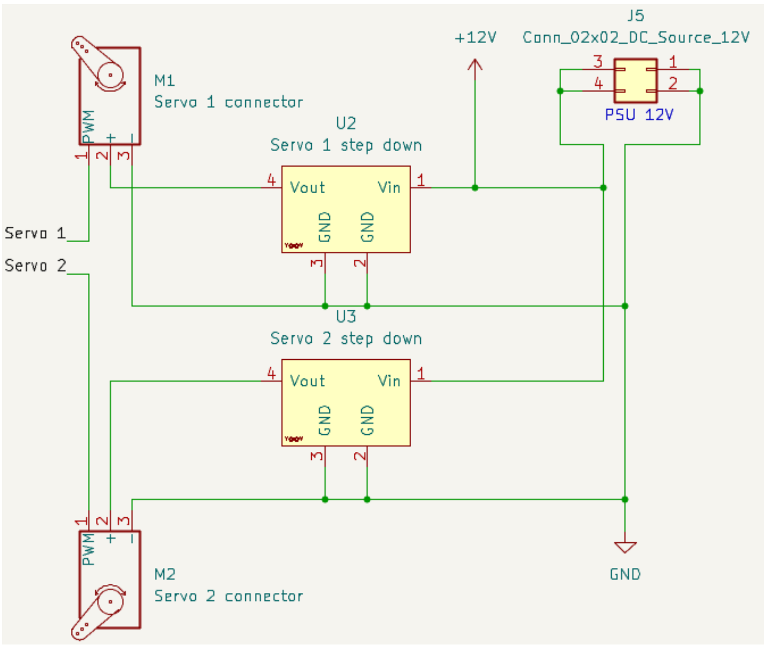


Figure 6: Electronic diagram: Servo Motors and DC-DC converters. Source: The authors

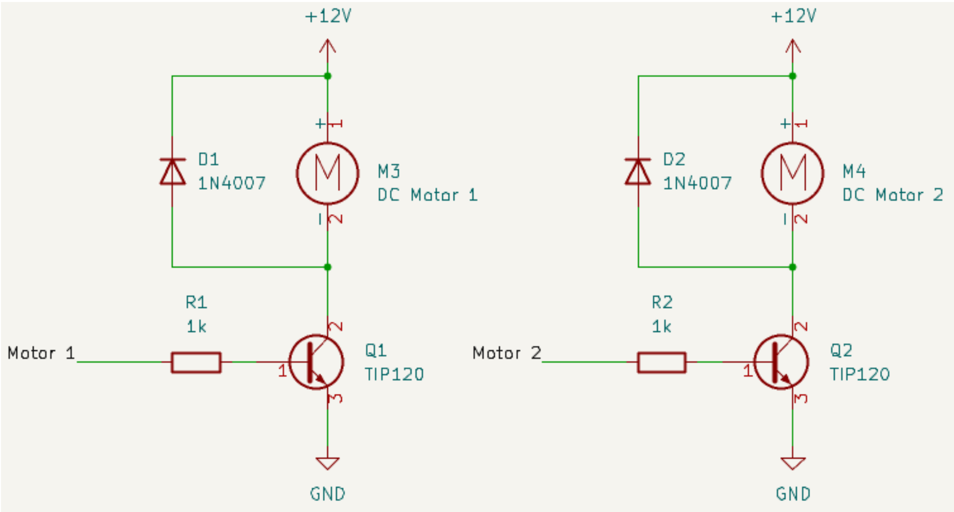


Figure 7: Electronic diagram: DC motors and their drivers. Source: The authors

Let's go step by step. As already mentioned, the power source that will be used for all the components, except the Raspberry Pi, will be a Computer Power Supply Unit (PSU). This source was chosen because it has 12V and 5V outputs, which are the two tension levels needed, and will also provide more than enough power and current for powering them all at the same time.

The DC motors are powered directly by the 12V of the computer source. Since they will turn the Archimedes Screws, and they are pretty heavy because they are made of metal, they need to have a lot of torque to also move all the candies inside the Screw mechanism. That's why some high torque DC motors with a reduction box were used. These motors are activated by a simple transistor driver circuit with a flyback diode, as can be seen.

The servo motors are powered though Step Down modules. These motors have to be quite strong as well, since they are moving the cups which can be full of candy. To use the maximum of their torque, it is important to power them with the maximum allowed voltage, which is 7 Volts. Since the computer source don't have 7V output, Step Down modules were used to reduce the 12V to the maximum 7 Volts. Here an important thing to mention is that for generating this PWM, the pigpio[8] pin factory will be used inside the gpiozero[9] library. This reduces jitter significantly, since pigpio also runs a daemon in background that allows for faster hardware-software integration.

The used display is a 3.5 inches Rpi Display, and so uses SPI communication[10] to receive information. It is important to notice that it is also a touch screen display. But, since this functionality won't be used, it will be deactivated. Thus, the most important thing to perceive is that the correct display pins need to be connected to one of the Raspberry Pi SPI interfaces.

Moving ahead, Figure 8 shows a power source diagram for all components, indicating all electric voltage levels.

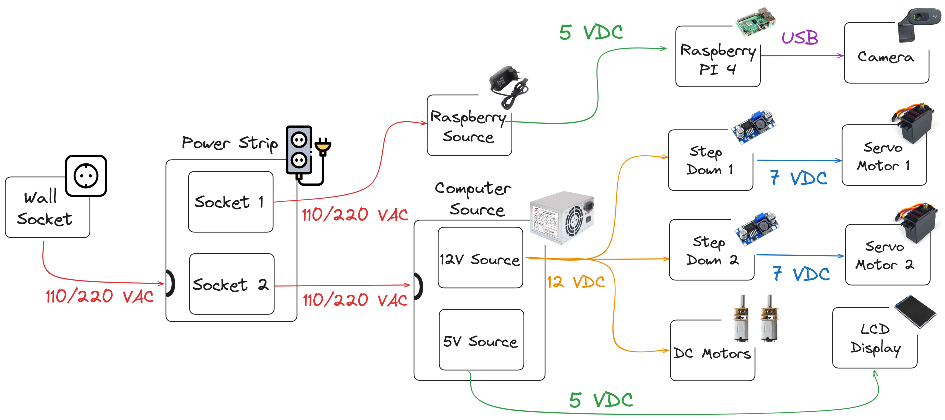


Figure 8: Power source diagram of all components. Source: The authors.

Here it is possible to see why a power strip was needed. The computer source can power up the whole circuit, but it's not being used for the Raspberry Pi. The reason behind that is that the Raspberry was a very essential and expensive component of the project. So, whatever could be done to prevent it from burning needed to be done. The main measure was to use the official Raspberry source, which is the topmost, guaranteed and recommended way of powering it up. So, since a single wall socket was the a requirement for the project, the power strip is a good addition.

3.3 Software Design

The software part of the project can be divided into two major parts. The firmware, which will run in the Raspberry and was made using Python[11] and the PWA, which will run in the cloud, and was made using Javascript and React[12] framework.

3.3.1 Firmware

The firmware have 3 basic functionalities, control the machine operation, do the image processing to identify gestures and communicate with the Firebase functions to integrate with the PWA.

The image processing was done using the MediaPipe[13] library. This library allows the downloading and running of a task that has 2 main purposes. First, identify a hand in a given image frame, and second, get the position of the hand joints within the frame. With that, we only need to translate the joint positions into a gesture label.

The first try was to use a pre-trained Neural Network with Tensorflow[14] to do that. This solution didn't stuck until the end because it was very costly from a processing point of view. Since this code was running in a embedded system, the frame rate became too low and the response time too high.

So, a better solution was implemented. A code was adapted from "Prasad9" Github [15] based on another one [16] by "andypotato" to perform calculations using the joints positions. This calculations result in the curling level, direction and orientation of each finger. Using those 3 values, the gesture being made can be deduced by inserting the expected configuration for each case. The joint tracking and gesture detection can be seen in Figure 9.

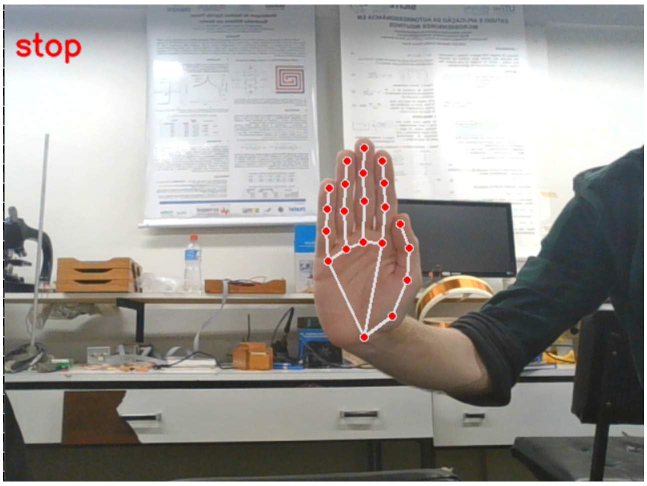


Figure 9: Process of gesture recognition. Source: The authors.

Having in mind the gesture recognition process, the machine operation can be more easily explained by the very simplified StateChart in Figure 10.

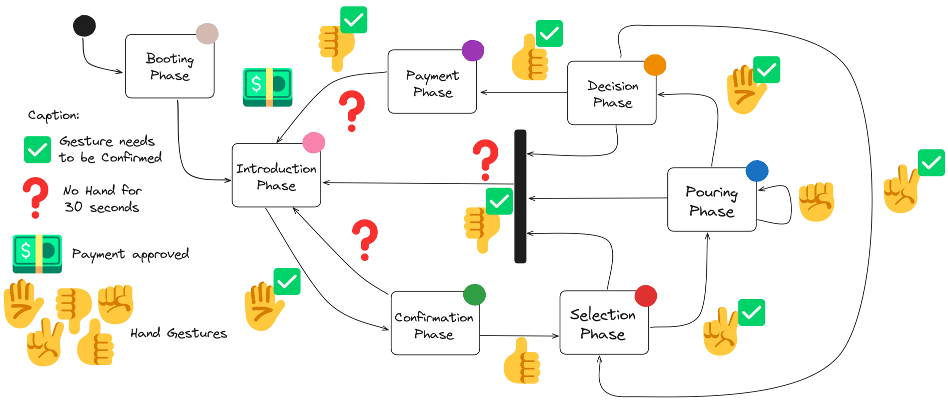


Figure 10: Statechart diagram of the firmware. Source: The authors.

Here it is possible to see that 5 different gestures were defined to operate the machine. Each phase looks for different gestures, and each gesture has different effects depending on the phase. Furthermore, if the joint positions don't fit any of the 5 configured gestures, it is classified as unknown, and can be used to perform some actions in some stages. If no hand is found within the frame, the customer is considered gone, and an alert message will appear at the display showing the customer was lost. If it remains this way for more than 30 seconds, the order is automatically rejected.

The diagram also shows that some gestures need to be confirmed. This is important because some of them result in very critical actions, and so it is important to be sure the customer is really wanting to do them, and don't do by accident.

In the booting phase, the machine tries to connect to the internet and reject any candies that are currently in the cups. In the introduction phase the machine loads the information from the database if any change is detected and waits for a Stop gesture. The confirmation phase is just to make sure the detected gesture is from someone who wants to place an order and so the machine waits for a Thumbs Up. The Selection phase is where the client will decide which of the 2 candies he will buy by using the absolute position of the hand and a Peace Gesture to confirm. The pouring phase allows the customer to pour the candy into the cup using a Fist gesture and stop it by using a Stop gesture. The amount of candy in the cup is calculated by using the time the motor was on. The decision phase is where the client will decide if he wants to buy more candy, finish the order here, or reject it. Finally, the payment phase is where the PIX QR code will be shown and the Mercado Pago API will be used.

If, in any phase the customer makes a Thumbs Down or leaves for more than 30 seconds, the system rejects the order and go back to the Introduction Phase. If by any chance the machine loses power during the payment phase, it will automatically accept any order after it boots, i.e, it will deliver the candies to the customer. This is necessary because there is the possibility that the client payed for the candy and the response just didn't arrive in time.

3.3.2 Progressive Web App (PWA)

As already said, the owner app was implemented in form a of PWA, which is essentially a web page that has some specific requisites that make it able to be installed in a cell phone. Then, it will appear in the phone with an icon that looks exactly like a real app.

The app also is fully integrated with Firebase, which is a Cloud service provider whose main function is to be a Database. So, all functionalities that use Firebase and will be mentioned in sequence are being held in the Cloud. In fact, the main app itself is hosted within Firebase in the Cloud. In this form, Firebase provides a url so anyone anywhere can download and use the app.

The main functions of the app can be seen in the use case diagram of Figure 11.

The login is important because one of the main ideas is to have multiple machines, so some form of guaranteeing that the owner will only be able to connect to his machine is needed. The login username and password will be given to the

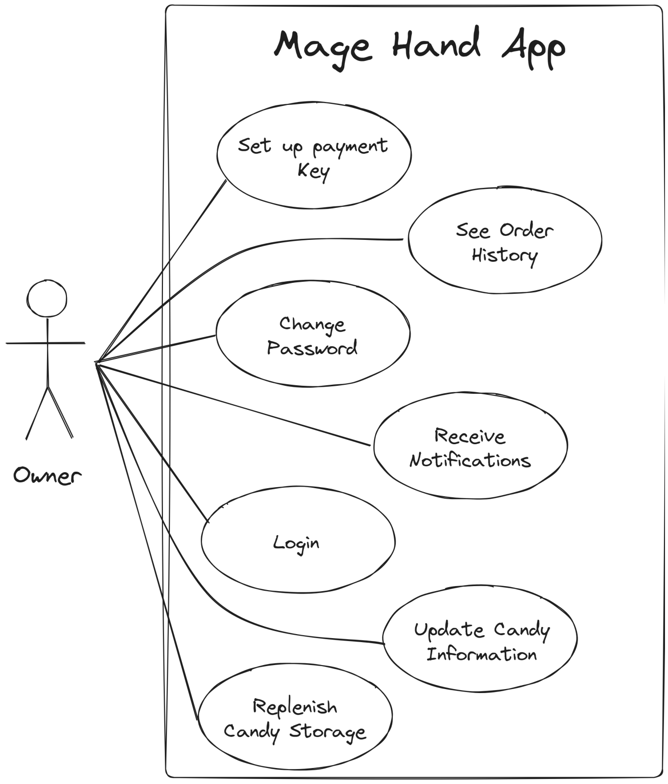


Figure 11: Use Case diagram of the app. Source: The authors.

owner together with the machine. That's why the owner also have the possibility of altering his password, to ensure no one can access his machine. This login process is handled using the Firebase Authentication Service.

The app also need to allow the owner to change the candies names, prices and also images to his own need. Moreover, the app will provide functionalities to permit the owner to change his Mercado Pago Token, along with instructions on how to do so, and input the replenished amount whenever he need to put more candy in the storage. This last part is essential since the machine need to know how much candy there is in it so it can discount it properly. Along with this, the owner can see the order history of each day, which contain the status of the order, the amount of each candy bought and the final price. All these functionalities are implemented and integrated with the Firmware using the Firebase Realtime Database to store text data, and Firebase Storage to store the candy images.

At last, the app permits the owner to receive feedback notifications from the machine as well. There are three types of notifications. One for warning when the machine just went offline, one warning for when the machine just went online and one warning when one of the machine storages is empty and needs replenishment. The notifications where implemented using Firebase Cloud Messenger and Firebase Cloud Functions. Basically, the machine keeps pinging a variable at the Firebase Realtime Database while it is online. A Cloud Function inside Firebase is awaken time to time to verify this variable and then deduce the machine status, as well as the storage situation. If this function notices anything different, it send a notification to notification token registered to the machine owner.

All PWA screens can be better visualized at Figure 12.

4 Results

The results of the Mage Hand project were satisfactory. All the requirements were fulfilled, after the requirement modifications according to the Mechanical Redesign.

The Mage Hand is able to automatic sell candies, with a fun and innovative user experience, allowing the customer to reject and at the same time, not waste products; it is easily customizable for each owner; it don't let the customer reach the candies before payment and allow him to see how much candy is available.

A picture of the full project can be seen in Figure 13. Also, the link with the final video presentation can be seen in Mage Hand - Final Video[3]. All the process of the project development can be see in the blog Mage Hand - Blog[4].

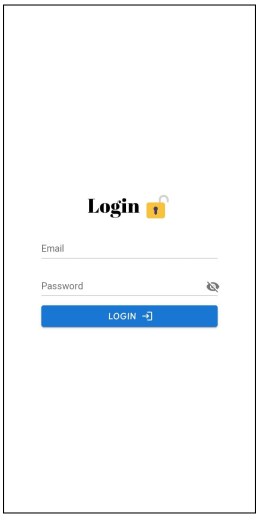
4.1 Budget

Table 1 contains the list of materials used to build the Mage Hand Machine. The initial budget preview of the project was R$ 1005.05, but since many mechanical parts had to be redesigned or replaced, the project spent plus R$ 233.44 of the R$ 250.00 Risk Budget foreseen in the beginning. The Risk Budget was spent in 2 DC Motors (R$98.44), in 2 metal Archimedes screws (R$ 20.00), 2 PVC tubes (R$ 50.00) and one acrylic piece (R$65.00).

Overall, the budget was inside the expected: R$1238.49 of R$ 1250.00.

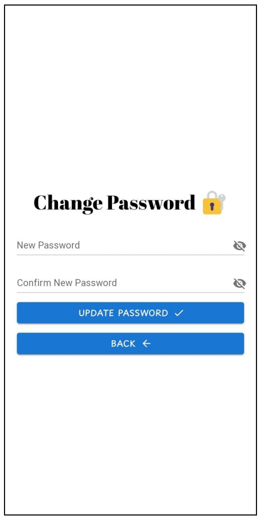
[3]Link to Final Video: `https://www.youtube.com/watch?v=aE3ZgukRHk4`

[4]Link to Blog: `https://magehand2.notion.site/Mage-Hand-2175e10a08164b84a041c308a4688ef4?pvs=4`

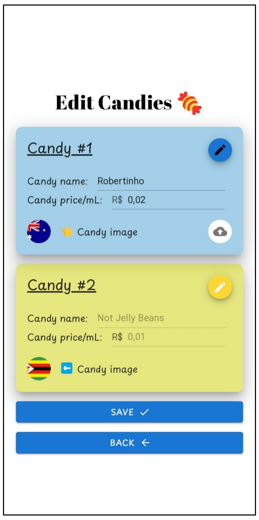


(a) Login Screen.

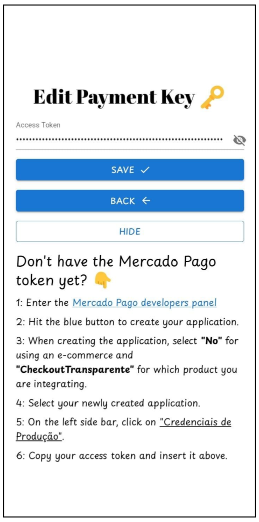
(b) Main Menu.



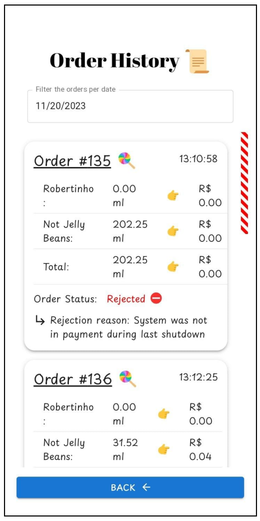
(c) Change Password.



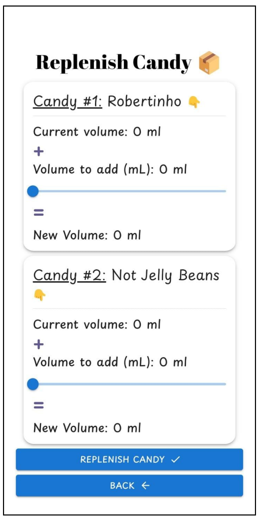
(d) Edit Candies.



(e) Edit Mercado Pago Key.



(f) Order History.

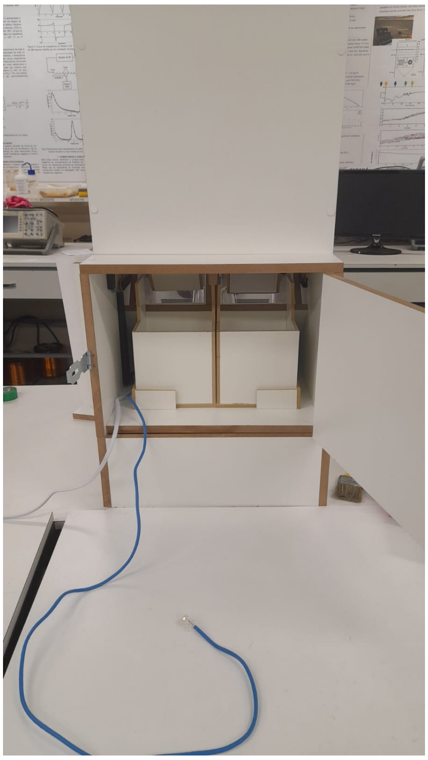


(g) Replenish Candy.

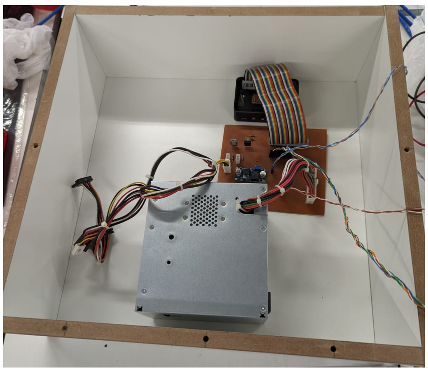
Figure 12: Complete PWA Screens. Source: The authors.

(a) Front View.

(b) Back View Closed.



(c) Back View Opened.



(d) Internal Basement View

Figure 13: Complete Machine View. Source: The authors.

Initial Budget	
Name	Price
Servo Motor 2x	R$100.00
Camera	R$150.00
Raspberry Pi4	R$314.15
LCD Display 3.5”	R$100.00
MDF Boards	R$50.00
PVC Transparent Board	R$39.90
ABS Filament for 3D printing	R$40.00
Wiring/Screws/Nails/etc	R$50.00
PCB (Phenolite Board)	R$10.00
Power Source	R$50.00
Shipping of components	R$100.00
Total Initial Budget	**R$1005.05**
Risk Budget	
Name	Price
2 DC Motors	R$98.44
2 Metal Archimedes’ Screws	R$20.00
2 PVC Tubes	R$50.00
Acrylic Piece	R$65.00
Extra MDF	R$100.00
Total Risk Budget Used	**R$333.44**
Total Budget	
Total Initial Budget	R$1005.05
Total Risk Budget Used	R$333.44
Total Budget	**R$1338.49**

Table 1: Project Budget

4.2 Schedule

In the project Schedule, all the tasks were planned to be accomplished in 6 deliverables, starting from September 17^{th} to November 16^{th}, over 8 weeks. The deliverable are defined as following: Deliverable 1 - Mechanical Design ; Deliverable 2 - Mechanical Project and Electronic Design ; Deliverable 3 - Electronic Project, Full Mechanical Structure and Software design ; Deliverable 4 - Software Project and Partial Integration ; Deliverable 5 - Full integration part I ; Deliverable 6 - Full integration part II.

The number of planned hours was of 423h, but the total number of worked hours was of 541h, 28% over the estimated. The main problems faced were in Deliverable 3, where most of the mechanical problems showed up. Also, Deliverable 3 and 4 were the ones where the most delayed hours for the project development happened. Figure 14 shows visually the number of planned and worked hours and the number of delayed hours per Deliverable.

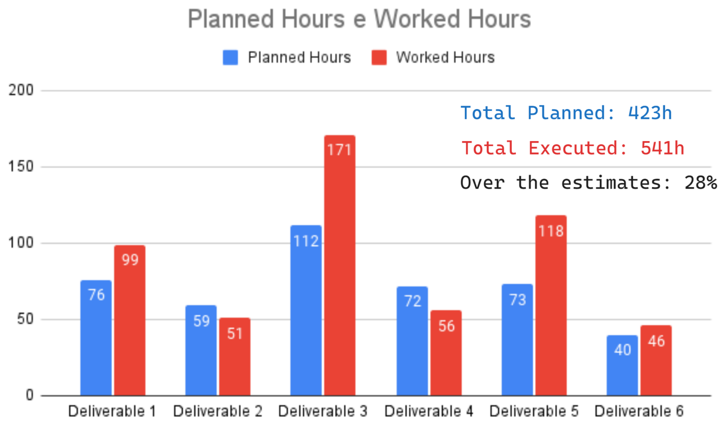


Figure 14: Hours Planned x Hours Worked on each deliverable. Source: The authors.

5 Conclusions

Finally, at the end, it is possible to draw some conclusions about the final result and development of the project. First of all, it is very important to say that the team faced a lot of mechanical issues, mainly in design. This is due to the almost complete lack of knowledge in this field by all team members, who were much more familiarized with hardware and software, reason why there were so little problems in those areas which were also quickly fixed. The mechanical design needed to be rebuilt radically at the start, and after that, unraveling the mechanisms also proved to be a tricky task. The team was only able to achieve such level of success thanks to a lot of external help.

Outside that, 2 more considerations about the development are valid. First, the team now fully comprehend the importance of a solid risk response plan, since almost all of the listed risks at the begging really happened. Second, the team also only started to produce better results at the end, when the teamwork was finally growing, showing the importance of knowing how to work properly as a team.

Now, regarding the final results, they are very satisfactory. All that was planned was able to be accomplished in a solid way and the machine ended up being really fun to use. The interface went through a lot of test involving no team members (to reduce bias) and so could be refined very well. Nevertheless, some improvements may be still considered. The method for measuring the candy amount by the time the DC motor turned is not as precise as it would be nice to be, so a kind of scale would be necessary. The size of the display is also no optimal, since too much information need to be shown at certain part, a bigger display would be better. Some quick fixes at the structure could be switched to more durable solutions. And at last, several sensor could be added to detect failure in motors and also the amount of candy in the storage.

Acknowledgments

We would like to express our sincere gratitude to the individuals who made this project possible. Without their support and guidance, this work would not have been achievable. We want to acknowledge Edival Ribeiro Chaves and Mister Meger for providing the metal Archimedes Screw, Bruno Bock, Mario Alfredo and Prof. Miraldo for giving some very valuable mechanical insights, Profs Fabro and Heitor who gave serious feedbacks and tips during the whole process, Matheus Kunnen and Ricky for lending their 3D printers and time, Prof Pichorim and Prof Caio for letting us using the BIOTA lab at the university, Franco Barpp, Jhonny Krystian and Toledo for being present for most of the duration, Prof Rubens for helping with electronics doubts and most important, all the people at the wood workshop here at the UTFPR, specially Chico, Vinícius, Natália and Gabriela Mazi, who helped us with all the wood work and without whom the project would certainly not be possible.

References

[1] NCA & 210 Analytics. https://candyusa.com/wordpress/wp-content/uploads/2023/02/Bite-Sized-Candy-Insights-eFinal-022723.pdf.

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

[3] Archimedes Screw. https://en.wikipedia.org/wiki/Archimedes%27_screw.

[4] Mercado Pago API. https://www.mercadopago.com.br/developers/pt/reference.

[5] Banco Central. https://www.bcb.gov.br/estabilidadefinanceira/pix.

[6] PWA. https://web.dev/explore/progressive-web-apps.

[7] Firebase. https://firebase.google.com/?hl=pt-br.

[8] pigpio. https://abyz.me.uk/rpi/pigpio/.

[9] gpiozero. https://gpiozero.readthedocs.io/en/latest/.

[10] SPI. https://en.wikipedia.org/wiki/Serial_Peripheral_Interface.

[11] Python. https://www.python.org/.

[12] React. https://react.dev/reference/react.

[13] MediaPipe. https://developers.google.com/mediapipe.com/.

[14] Tensorflow. https://www.tensorflow.org/?hl=pt-br.

[15] Prasad9. https://github.com/Prasad9/Classify-HandGesturePose.

[16] andypotato. https://github.com/andypotato/fingerpose.