AUTOMATED PET CAGE

By

Avram Fouad
Christina Hejny
Saurav Kumar

Final Report for ECE 445, Senior Design, Spring 2022
TA: Hanyin Shao

04 May 2022
Project No. 16

Abstract

Our group created an automated cage for owners of pet rodents. The system allows for the user to manually enter the time they would like the cage to open. Once this time is reached, the cage automatically opens and allows the pet to get its necessary exercise and enrichment time. After the pet is outside the cage for at least 2 hours, the cage automatically closes once the pet returns and a feeder dispenses a treat for the pet to devour. The only user effort required by our device is for the user to enter the desired time to open the cage. Despite being unable to use our fabricated PCB, our project was still a success and accomplished the requirements mentioned in later sections.

Contents

1. Introduction

1.1 Problem

According to a survey done by Mott Poll Report, 63% of families in 2019 without pets said that their top reason for not keeping one was that it requires too much hassle [1]. Keeping a pet rodent requires plenty of care and responsibility just like with any other pet. It is recommended that a pet rodent receives at least one hour of time outside its cage everyday [2]. Unfortunately, many people have work or school commitments that prevent them from keeping a pet because they wouldn't be available at home during the day to let out their pet and feed them. Additionally, some existing pet owners find it extremely inconvenient to have to manually let their pet out of the cage and back in.

1.2 Solution

To solve this problem, we created an automated rodent cage. The device is mounted on top of the cage and includes a feeder, a pressure sensor, and an ultrasonic sensor. The user is first prompted to enter the current time and then prompted to enter the desired time for the cage to open. Once this specified time is reached, the cage door will open letting the pet out. Our solution utilizes a compact servo motor to rotate the door along its hinge to fully open and close it. The servo motor is governed by our control unit that keeps track of real time to determine when to open the door. The desired time that was entered by the user is also being displayed using an LCD screen. In addition, there are "open" and "close" buttons that immediately open and close the door in case of any emergency.

Once the time entered by the user is reached, the cage will open for at least 2 hours and then begin the closing process. Once the cage has been open for exactly 2 hours, the device will "beep", letting the rodent know it's time to re-enter the cage. Once the rodent has re-entered the cage, it will step on a thin-film pressure sensor pad. Upon activating the pressure sensor, the cage door will begin to close. Once the door has fully closed, the feeder will dispense a treat to the pet. The idea behind the "beep" and feeder system is to train and reinforce the rodent to enter the cage every day (similar to a skinner box). The pet can be easily trained to understand the "beep" means to re-enter the cage and get a treat.

1.3 High level Requirements

- The device must be able to properly control the orientation of the cage door. This includes fully opening and fully closing, as well as stopping if something blocks the doorway.
- If the cage has been open for at least 2 hours [+/- 1 minute], the door must close within 3 seconds of the thin film pressure sensor activating. This is to ensure that the rodent does not leave the cage before the door closes.
- The cage should not close during the designated 2 hour open-cage period under any circumstance unless the emergency "close" button is pressed. This strict requirement will guarantee that the rodent is never locked out of the cage.

1.4 Visual Aid

Figure #1: Front View of Cage

1.5 Block Diagram

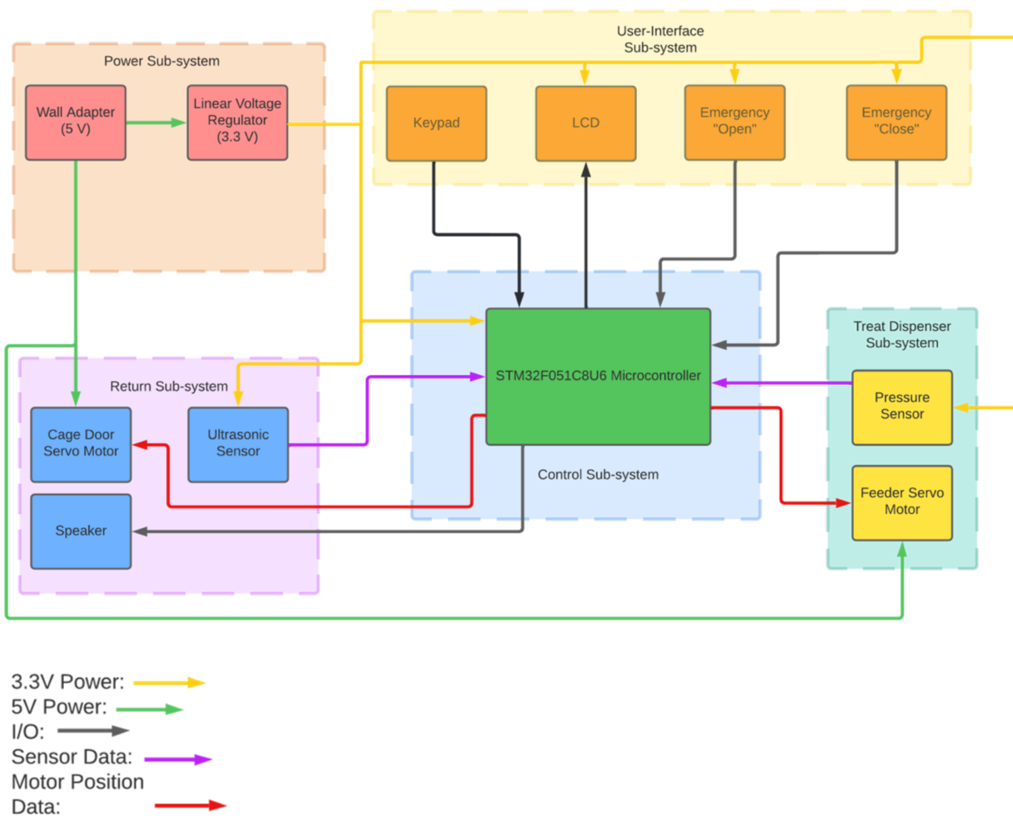

Figure #2: Block Diagram

The Power sub-system provides power to all other sub-systems via 3.3V and 5V outputs.   The User-Interface sub-system provides the time data input by the user via a keypad and displays the specified time on an LCD display. The Return sub-system operates the cage door, via a servo motor, an ultrasonic sensor and produces the “beep”. The Treat Dispenser sub-system checks if the rat has returned to the cage and dispenses the treat to it via the feeder when required. The Control sub-system consists of just a microcontroller. It processes all the input and sensor data and outputs the motor position data for the Return and Treat Dispenser sub-systems. It also outputs I/O voltages to the LCD display and speaker.

2. Design

2.1 Power Sub-System

The power sub-system consists of a 5 V power supply that plugs into a standard US wall outlet. The input to the power supply is an AC voltage source with a frequency of 60 Hz and an RMS voltage around 120 V. The power supply converts this to a DC source with a voltage of 5 V. The 5 V output is used to directly power the servo motors for the feeder and the cage door. According to the servo motor datasheet, the motors can experience current spikes of up to 3 A when they are active [3]. Additionally, only one motor is active at a given time. Therefore, we can calculate the maximum power drawn by each motor using Equation #1. It should be pointed out that the power supply we chose has a current rating of 5 A in order to meet the current rating of the motor.

$$P_{motor} = V * I = (5\ V) * (3\ A) = 15\ W \quad \text{(Equation \#1)}$$

The power sub-system also consists of a linear voltage regulator which is used to step down the 5 V output of the power supply to 3.3 V. The 3.3 V output of the regulator is then used to power most of our other components such as our LCD and Pressure sensor. Figure #3 displays the final schematic of our power sub-system. The voltage regulator we used has a range of output voltages that can be selected by the resistor and capacitor values in the schematic. Therefore, these values were chosen from the datasheet to give us a 3.3 V output [4].

In our initial design, we planned on utilizing a 3.7 V Lithium-ion battery to power our device. A buck-boost converter would then be used to step up the 3.7 V to 5 V in order to power the motors. Additionally, the same voltage regulator would be used to step down the voltage to 3.3 V in order to power the rest of our components. However, our final design proved to be much more efficient and less complex because we could get rid of the buck-boost converter entirely and power our motors with the 5 V supply. Furthermore, by getting rid of the battery, we didn’t have to worry about designing a battery charging circuit. Since most pet owners keep their cages at home where a standard wall outlet is present, we didn’t see any reason to keep the battery as our power source.

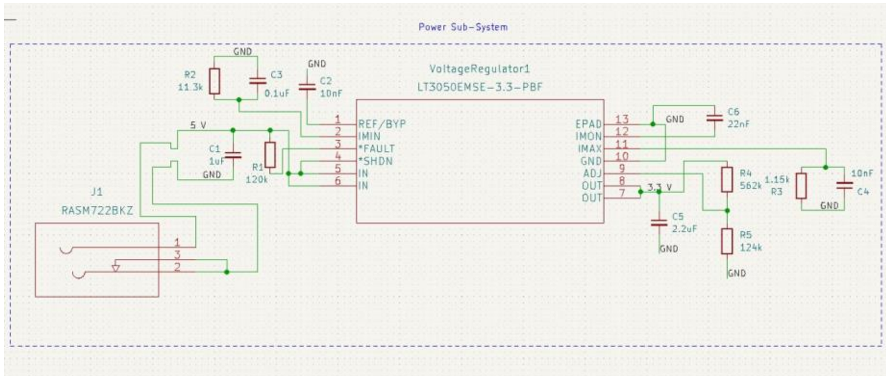

Figure #3: Power Sub-System Circuit Schematic

2.2 User-Interface Sub-System

The user-interface sub-system consists of a 4x3 keypad and an LCD. The 4x3 keypad is first used to set the real time. The user will simply enter 4 digits (the first 2 digits represent the hour while the last 2 represent the minutes). Then, to confirm the real time, the user will press the "#" key. The real time will then be displayed on the LCD. The control sub-system will keep track of time and ensure that the minutes and hours are correctly incremented. To reset the real time input, the user can press "*". Once the user has set the real-time, the LCD prompts the user to select a time to open the cage. This is done in the same manner as entering the real time. For example, pressing "*" will clear the selected time while pressing "#" will confirm the time entered. After the user confirms the desired time to open the cage, the LCD will display this value. It should be pointed out that the user still has the option to clear/reset the desired open time even after they confirm their original selection. Furthermore, the user has the option to immediately open or close the cage door in case of any emergency event. If the user presses the character "*" followed by "#", the cage door will immediately close if it was previously open and vice versa.

In our initial design, we planned on using a 4-digit seven segment display to show the desired open time selected by the user. However, after wiring the display and writing some code on Arduino IDE, we found that the display needed constant refreshment of data even if we wanted it to display the same value indefinitely. Therefore, the microcontroller could not constantly refresh the data while performing other functions at the same time. As a result, we decided to use an LCD instead. This proved to be a much better option as the LCD did not require constant refreshment to display the same text. Furthermore, the LCD allowed us to display more than 4 characters of text. This enabled us to prompt the user for input in addition to displaying their selected time to open the cage. Figure #4 displays the final circuit schematic for this sub-system.

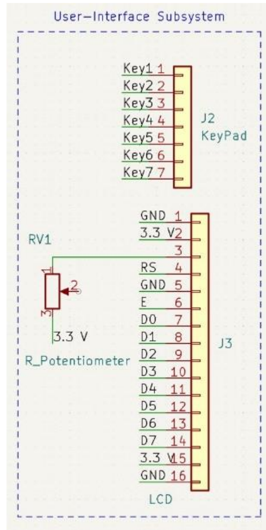

Figure #4: User-Interface Sub-System Circuit Schematic

2.3 Control Sub-System

The control sub-system is composed of an STM32F051C8U6 microcontroller. The microcontroller contains a real-time clock module which is calibrated using a 32.768 MHz external crystal. The control sub-system is responsible for correctly processing all of the user input data. We wrote code in Arduino IDE that was able to correctly scan the user input from the Keypad. Once the correct characters are scanned, the microcontrollers sends data to the LCD such that the real time and selected open time entered by the user on the keypad can be displayed. After the user initializes and confirms the real time, the microcontroller keeps track of the time by correctly incrementing the hour and minutes. Additionally, after the user enters a desired time to open the cage, the microcontroller checks whether the real time and the open time are equal. If they are not equal, the microcontroller simply continues to keep track of real time. Once the open time is reached, the microcontroller sends a PWM signal to the cage door servo motor which opens the door to let the pet out. After the cage door opens, the microcontroller continues to keep track of real time.

If the pet comes back to the cage before being outside for at least 2 hours, the cage door will still remain open and the feeder won’t dispense any food. This will serve as an indication for the pet to return outside the cage and get more exercise. Once the cage has been open for exactly 2 hours, the microcontroller will send a pulse to the speaker which will play a “beep” sound. This is used to notify the pet to come back to the cage. The “beep” sound is played in 15-minute intervals until the pet comes back inside the cage and steps on the pressure sensor. After the pressure sensor is activated, the microcontroller sends a PWM signal to the cage door motor which closes the door. Shortly after, the microcontroller sends another PWM signal to the feeder motor which dispenses food for the pet to devour.

The control sub-system also takes in distance values from an ultrasonic sensor placed by the cage door. If the door is in the middle of closing and the ultrasonic detects any object within 18 cm of the door, the microcontroller immediately sends a pulse to the cage door which reverses the direction of the door. This is done to ensure that the door never closes on the pet and causes any injury. In addition, if the emergency button is pressed at any point, the microcontroller immediately sends a PWM signal to the cage door motor which opens the cage door if it was previously closed and vice versa. Figure #5 displays the circuit schematic for the control sub-system.

Unfortunately, our control sub-system did not work as intended because we were unable to program the microcontroller. We realized that in our PCB design we did not use the correct pins to program the STM32. We thought that we could use the SPI protocol/pins to program the microcontroller but it turns out that it can only be done using the Serial Wire Debug pins. This is further elaborated in the verifications section. We ended up replacing the microcontroller and PCB with an Arduino Mega Dev Board. Fortunately, we were able to demonstrate complete functionality with the dev board. This will also be elaborated in the verifications section.

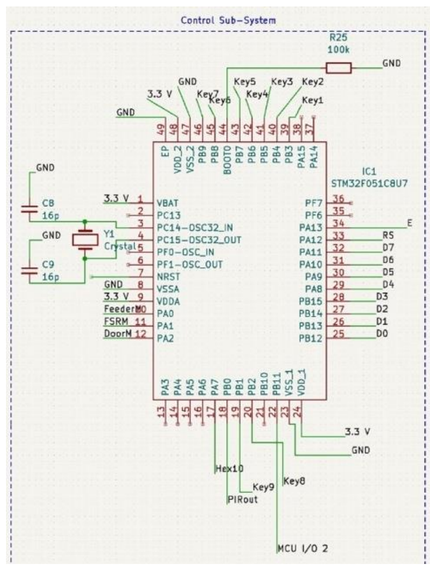

Figure #5: Control Sub-System Circuit Schematic

2.4 Treat-Dispenser Sub-System

The treat-dispenser sub-system includes a thin-film pressure sensor and a servo motor for a feeder. This sub-system is responsible for dispensing food once the pet returns to the cage and steps on the pressure sensor after being outside the cage for at least 2 hours. The pressure sensor acts as a force-sensitive resistor (FSR). With no force applied on the pad, the equivalent resistance is in the Megaohms. However, as more and more force is applied, the resistance decreases at an exponential rate. The pressure sensor is used in a voltage divider circuit and is placed in series with a 470 kΩ resistor as shown in Figure #6. With no force applied on the pressure pad, the resistance is orders of magnitude greater than 470 kΩ. Therefore, the voltage across the pressure sensor will be almost exactly 3.3 V (input voltage of the voltage divider circuit). As the force applied increases, the resistance of the pressure pad will quickly decrease along with the measured voltage until it reaches 0 V. The voltage across the FSR can easily be calculated as follows:

$$V_{FSR} = (3.3\ V)\frac{R_{FSR}}{R_{FSR}+470\ k\Omega} \qquad \text{(Equation \#2)}$$

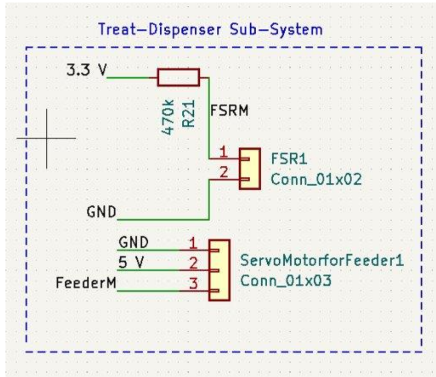

Figure #6: Treat-Dispenser Sub-System Circuit Schematic

The pressure sensor we used did not have any tables or graphs included in the datasheet that showed the relationship between the equivalent resistance and force applied. Therefore, we collected data for the FSR voltage vs weight applied ourselves using test weights as shown in Table #1. It is clear that once the weight applied exceeded 30 g, the FSR voltage began to dramatically decrease. Figure #7 displays a plot of the data with an exponential fit. Equation #3 gives the equation for the exponential fit.

Weight Applied (g)	FSR Output Voltage (V)
30	3.29
35	0.71
50	0.21
70	0.10
100	0.045
150	0.026

Table #1: FSR Voltage vs Weight Applied

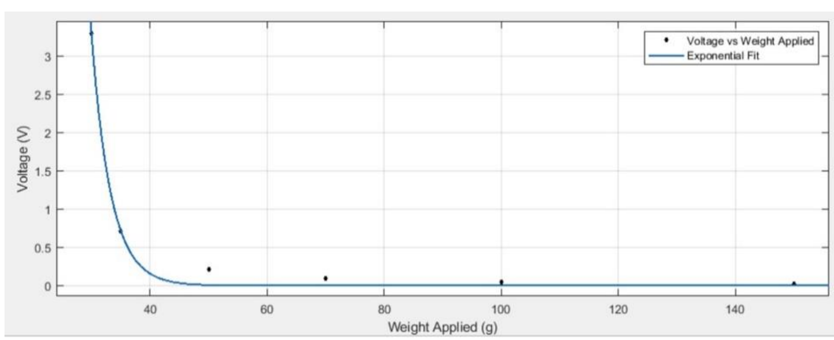

Figure #7: FSR Voltage as a Function of Applied Weight

$$V_{FSR} = (3.024x10^4)e^{-0.3042*Weight} \qquad \text{(Equation \#3)}$$

Finally, once we collected data on the FSR voltage, we had to decide on a threshold voltage for the FSR to activate the feeder. The feeder should only be activated for a sufficiently low FSR voltage since that would mean the pet is stepping on the pad. We wouldn’t want a false alarm scenario where the feeder is activated when only pieces of food are laying on the pressure pad. We decided that the feeder should not dispense food for weights below 50 g. According to Table #1, this would correspond to a threshold voltage of 0.21 V. However, we ended up choosing a threshold voltage of 0.16 V to be safe. Since the typical adult rat is 250-300 g, only about 15-20% of its full body weight would need to be applied to the pad to activate the feeder. Additionally, since a rat only eats 20 grams of food a day, droplets of food would not be heavy enough to ever activate the feeder [5].

2.5 Return Sub-System

The return sub-system consists of a speaker, an ultrasonic sensor, and a servo motor for the cage door. This sub-system is responsible for ensuring that the pet returns to the cage in a safe and secure manner. Once the pet has been outside the cage for exactly 2 hours, the speaker will play a “beep” sound used to notify the pet to come back inside the cage. The “beep” is repeated in 15 minute intervals until the pet comes back inside the cage and activates the pressure sensor. At this point, the control system sends a PWM signal to the servo motor to close the cage door. The ultrasonic sensor is used to calculate the distance of the nearest object while the door is closing. If any object is within 18 cm of the door, the control system will send a pulse to reverse the direction of the cage door movement. This is done to ensure that the cage door never closes on the pet.

Figure #8 displays the circuit schematic for this sub-system. The microcontroller output for the speaker is first amplified using a BJT test circuit given by the speaker datasheet [6]. Our design initially utilized a PIR sensor to detect any movement near the cage door. The output of the sensor would then be amplified by an Op-Amp. However, after wiring the PIR sensor with the Op-Amp and writing sample code, we noticed that the output values we were getting from the sensor were mainly noise and did not change at all even if we created movement in front of the sensor. As a result, we replaced the PIR sensor with an ultrasonic sensor which proved to be very beneficial. This small change allowed us to obtain accurate distance measurements that we could use in our code to prevent the cage from closing on the pet. This will be further elaborated in the verifications section.

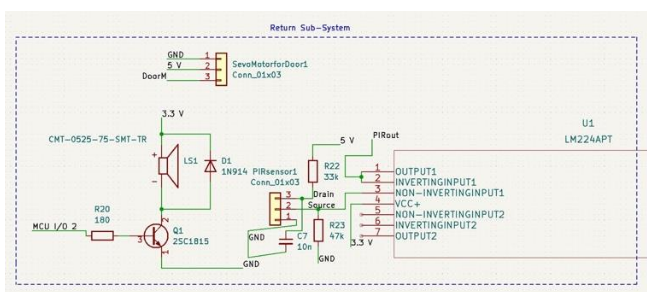

Figure #8: Return Sub-System Circuit Schematic

3. Design Verification

The finished device functions exactly as designed. During the final demo, the device was able to show all functions and in the correct sequence on a pet cage. Videos 1, 2, and 3 in Appendix B show how the device works and verifies that the complete product works as intended. Most of the features of the device are shown in the videos except for the clear function, emergency open and close button, and that the pressure sensor will not trigger when debris is on it. The finished product can verify almost all requirements set out by the design document. The control sub-system did not meet the requirements due to issues with programing the microcontroller. A further explanation of this is found in section 3.3. However, the project is an overall success. With extra time and further work on the project, the device would be able to meet all requirements with high confidence. The requirements and verification tables for each sub-system can be found in Appendix A.

3.1 Power Sub-system

The power sub-system successfully meets all the requirements set out in the design document, which can be found in Table 1 of Appendix A. To generalize the requirements, the sub-system must output a 5 V line to the other sub-systems and be able to handle up to 3 A of current. It must also output a 3.3 V line that can handle up to 300 mA of current. Both requirements were met and verified in their respective subsections below.

3.1.1 5 Volt Power Line

For the 5 V power line, the specific requirement is that it "must provide 5V +/- 8% for up to 3 A of current." This gives the sub-system a voltage range between 4.6 V to 5.4 V. Within this range, all components that use 5 V will function properly on the device. This includes the two servo motors and the ultra-sonic sensor. The 3 A current limit was decided based on the needs of the servo motors. At times, the servo motors may experience a sharp peak in current of up to 3 A [3]. If the system could not provide enough current during a spike, the device might shut off temporarily and reset.

To verify this requirement, we followed the procedure outlined in Table 1 of Appendix A. The final resistance value of the potentiometer was found to be 1.69 Ω to reach the 3 A of current. Figure #9 shows the oscilloscope reading of both the 5 V and 3.3 V power lines under the current conditions with close up of the voltage measurements. The yellow line represents the 5 V line's voltage. As seen in figure #9, the yellow line is completely straight with almost no variation in voltage. The output voltage was measured to be 5.098 V. This fits well within the range that is required for the device. The 5 V power line successfully meets the requirement.

Figure #9: Oscilloscope reading of both 5 V and 3.3 V outputs of power sub-system.

3.1.2 3.3 Volt Power Line

For the 3.3 V power line, the specific requirement is that it "Must be able to provide 3.3V +/- 6% for up to 300 mA of current." This gives the sub-system a voltage range between 3.102 V to 3.498 V. Within this range, all components that use 3.3 V will function properly on the device. This includes the microcontroller, pressure sensor, and LCD. The 300 mA current limit was decided by adding up the max current requirements for all components rounded up. This way, the device will be able to supply the max current to all components at the same time without temporarily shutting off and resetting.

To verify this requirement, we followed the procedure outlined in Table 1 of Appendix A. The final resistance value of the potentiometer was found to be 10.94 Ω to reach the 300 mA of current. The 3.3 V line is shown in figure 1 as the green line. As seen in figure #9, the green line is completely straight with almost no variation in voltage. The output voltage was measured to be 3.204 V. This fits well within the range that is required for the device. The 3.3 V power line successfully meets the requirement.

3.2 User-Interface Sub-system

The user-interface sub-system successfully meets all the requirements set out in the design document, which can be found in Table 2 of Appendix A. To generalize the requirements, the keypad must properly send the data input by the user to the microcontroller. The LCD must display the correct opening time input by the user. Finally, the “*” button must clear the set opening time after which the user must be able to enter a new desired opening time. All three requirements were met and verified in their respective subsections below.

3.2.1 Keypad Functionality

For the keypad’s functionality, the specific requirement is that “The microcontroller must correctly process the time sequence entered on the keypad by the user.” It is very important to the functionality of the device that the keypad correctly relays data to the microcontroller.
To verify this requirement, we followed the procedure outlined in Table 2 of Appendix A. The code added to display the keypads out is shown in Figure #10. It’s just a simple print statement that prints the variable that stores the opening time entered by the keypad. Figure #11 shows the serial monitor output of the added code. We start the device and program the software with the two prints statements. We then set the time to an arbitrary time. Then we entered “1”, “2”, “5”, “0”, “#” as outlined in the verification procedure. The monitor prints the correct opening time that was just entered. This verifies that the sub-system fulfills the requirement.

```
Serial.print("Time of opening:");
Serial.print(timeinput);
```

Figure #10: Added code that displays the opening time in the serial monitor

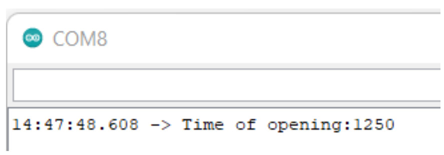

Figure #11: Output of the serial monitor displaying the correctly entered opening time.

3.2.2 Correctly Displaying Time

For the LCD’s functionality, the specific requirement is that “The LCD must be able to display the selected time input processed by the microcontroller.” This is done incase the user accidentally inputs the wrong time; they would be able to see the error and then correctly input the desired time. The user might also forget what time they set the device to open, so the LCD will also serve as a reminder to the user.

To verify this requirement, we followed the procedure outlined in Table 2 of Appendix A. Figure #12 shows pictures of the LCD displaying the correct information at different stages of the user inputs. After powering on the device, we input the sequence “1”, ”2”, “0”, “0”, “#” on the keypad. The LCD display then correctly displayed the current time entered that is used to set the real time clock. This is shown in the bottom left picture in Figure #12. Then we input the sequence “0”, “1”, “0”, “0”, “#” on the keypad. The LCD then currently displayed the correct desired opening time that we just set. This is shown in the top right picture in Figure #12. This verifies that the LCD can display both the current time set by user and then display the opening time for the rest of the devices use and that it meets the requirement.

Figure #12: Pictures of LCD at the various stages of user inputs.

3.2.3 Clearing Time

The user-interface sub-system must also have a clear input in case the user accidentally inputs the wrong time or simply wants to change the opening time for whatever reason they have. The specific requirement is that "The '*' button must be able to reset the selected 'open' time."

To verify this requirement, we followed the procedure outlined in Table 2 of Appendix A. This verification was carried out immediately after the verification in Section 3.2.2. Therefore, we had already set the current time to 12:00 and the opening time to 1:00. After this, we pressed the "*" button on the keypad. The LCD updated to show boxes where the opening time would be. The boxes alert the user that the opening time has been cleared and needs to be re-entered. Note that if you clear the opening time and do not set a new time, the device will simply not open at any time. The bottom right picture in Figure #12 shows the updated display with the cleared time. After clearing the opening time, we were able to input another 4-digit time, press "#", and the display then correctly showed the new opening time. This verifies that the sub-system correctly clears the opening time and fulfills the requirement.

3.3 Control Sub-system

The control sub-system did not meet any of the requirements set by the design document. We were unable to correctly program the microcontroller with our code, which means that it did not function at all. We were initially confused on which pins of the microcontroller were used to program the code onto the chip due to our inexperience with using them in custom hardware designs. We initially, and wrongly, assumed that we could use the SPI pins and designed the first PCB to use those pins with the programing cable. After soldering the PCB and failing to program the microcontroller, we learned that our specific microcontroller, the STM32F0, can only be programmed using the Serial Debug Wire (SDW) pins. The SDW uses different pins than the SPI, so we were unable to salvage the initial PCB order. For the second PCB order, we moved the pin assignments around so that the programing wires were connected to the SDW pins. By the time the PCB arrived and was soldered, there was very little time left before the demo. We were unable to get an ST-Link in time, which is a cable that connects from a computer to the SDW pins and programs the microcontroller. We also learned that the STM32F0 cannot be programmed with the Arduino IDE which all the code was written in.

We fully understand the issues with the sub-system and how to fix them for later versions of the product. In a third PCB order, we would re-design the control sub-system to use an ATMEGA microcontroller since it's compatible with Arduino IDE. We would also have time to acquire the ST-Link. With these changes and a few weeks of time, we are confident that the control sub-system will function properly and meet all the requirements. For the purposes of the final demo and for verifying our code and the rest of the sub-systems, we used an Arduino Mega dev board in place of our microcontroller. Note that the Arduino Mega uses the ATMEGA microcontroller and functions similarly to the proposed design changes. The dev board is used to verify the requirements for this sub-system to show that the code written for the microcontroller properly functions.

3.3.1 Real Time Clock

The control sub-system must be able to keep track of real time in order to determine when to open the cage door at the desired time. It must also be accurate to ensure that over weeks of use, the device doesn't slowly move out of sync to the point where the cage opens at a completely different time than the desired time. The specific requirement is that "The control sub-system must be able to set and keep track of the real time ± 1 minute."

To verify this requirement, we followed the procedure outlined in Table 3 of Appendix A. Code was added to the software to print the current every time the microcontroller updates the time. Figure #13 shows the code added to the software to print the time to the monitor. The clock was set on the device and then we waited an hour. Figure #14 shows the start of hour and Figure #15 shows the end of the hour. The real time clock completes one hour in 59 minutes and 57.596 seconds. This means that it has an error of 2.404 seconds per hour. The real time clock has a daily error of 57.696 seconds per day, meaning that the user must resync the time every 2 days to keep within the allowed 1-minute error. This verifies that the software fulfills the requirement. However, since the actual microcontroller was unable to be used, the overall sub-system does not fulfill the requirement.

```
Serial.print("Current time: ");
  Serial.print(hours);
  Serial.print(":");
  Serial.println(minutes);
```

Figure #13: Code added to print the current time in the serial monitor

```
COM8

15:11:02.743 -> Current time: 15:11
15:12:02.719 -> Current time: 15:12
15:13:02.664 -> Current time: 15:13
15:14:02.600 -> Current time: 15:14
15:15:02.560 -> Current time: 15:15
15:16:02.526 -> Current time: 15:16
15:17:02.486 -> Current time: 15:17
15:18:02.448 -> Current time: 15:18
15:19:02.404 -> Current time: 15:19
15:20:02.380 -> Current time: 15:20
15:21:02.333 -> Current time: 15:21
15:22:02.297 -> Current time: 15:22
15:23:02.252 -> Current time: 15:23
15:24:02.214 -> Current time: 15:24
```

Figure #14: Serial monitor showing start of test.

```
COM8

15:58:00.840 -> Current time: 15:58
15:59:00.806 -> Current time: 15:59
16:00:00.777 -> Current time: 15:60
16:01:00.733 -> Current time: 15:61
16:02:00.698 -> Current time: 16:2
16:03:00.652 -> Current time: 16:3
16:04:00.597 -> Current time: 16:4
16:05:00.569 -> Current time: 16:5
16:06:00.544 -> Current time: 16:6
16:07:00.502 -> Current time: 16:7
16:08:00.436 -> Current time: 16:8
16:09:00.394 -> Current time: 16:9
16:10:00.368 -> Current time: 16:10
16:11:00.339 -> Current time: 16:11
```

Figure #15: Serial monitor showing end of test

3.3.2 Interpreting Pressure Sensor Data

The control sub-system must only check the data from the pressure sensor after the two-hour opening period has finished. This is so the rodent's two hours of daily enrichment is not cut short by the rodent trying to get food too early. The specific requirement is that "The microcontroller should only check if the pressure sensor is triggered after the two hours timer has finished."

To verify this requirement, we followed the procedure outlined in Table 3 of Appendix A. After the cage opened, we placed a 250 gram weight on the pressure sensor. Upon placing the weight on the pressure sensor, the cage did nothing. For the full 2 hours, the device was unaffected by the weight on the sensor. At the 2 hours mark, the device beeped to signal for the rodent to come back to the cage. Immediately after the beep finished, the cage door closed and the feeder dispensed food. This shows that the control sub-system did not read the pressure sensor data until after the 2-hour period. This verifies that the code fulfills the requirement. However, since this was carried out using a dev board, the overall sub-system does not meet the requirement.

3.3.3 Generating "Beep"

The control sub-system must generate an 800 Hz "beep" to signal to the rodent that it is time to re-enter the cage, step on the pressure sensor, and receive its treat. This "beep" must be generated 2 hours after opening the cage. The specific requirement is that "The control sub-system must send a digital high signal to the speaker 2 hours $\pm$ 1 minute after the programmed time." The "beep" should then occur within a range of 1 hour and 59 minutes after opening to 2 hours and 1 minute after opening.

To verify this requirement, we followed the procedure outlined in Table 3 of Appendix A. Figure #16 shows the serial console in Arduino IDE. The first timestamp shows the cage door open at 9:52 pm. Then two hours later at 11:52 pm, the second timestamp shows the "beep" generated by the control sub-system. The third timestamp shows a second "beep" 15 minutes later. Incase the rodent does not hear the first "beep", the control sub-system will play "beeps" on 15-minute intervals until the pressure sensor is triggered. The difference in time between the cage opening and the beep was 1 hour 59 minutes and 55 seconds. This fits within the range of the requirement and verifies that the code meets the requirement. However, since this was carried out using a dev board, the overall sub-system does not meet the requirement.

Figure #16: Serial Console in Arduino IDE showing timestamped prints for the times specific functions occur.

3.4 Treat Dispenser Sub-system

The treat dispenser sub-system successfully meets all the requirements set out in the design document, which can be found in Table 4 of Appendix A. To generalize the requirements, the pressure sensor must be sensitive enough to detect a 250-500 gram rodent while also not trigger if any force below 50 grams is applied to the sensor. This is based upon the average weight of a rat. A rodent cage might also have food or debris lying around that could affect the sensor. The below 50-gram weight requirement assures that the pressure sensor will only be triggered when the rat re-enters the cage and steps on the sensor. The feeder must also dispense food as a treat for the rat.

3.4.1 Dispensing Food

The feeder within the treat dispenser sub-system must be able to dispense food as a treat for the rat. The treat is used to incentivize the rodent to come back into the cage and reinforce its training. The specific requirement is that "The feeder should be able to dispense at least 4 grams of food." The 4-gram requirement was set somewhat arbitrarily. It ensures that the rodent will receive a proper amount of food as a treat.

To verify this requirement, we followed the procedure outlined in Table 4 of Appendix A. We let the feeder dispense food into a bowl, then we placed the bowl on a scale that was already zeroed out with the bowl on it. Figure #17 shows a picture of the first trial. Table #2 shows the result of five dispenses.

The average weight of food dispensed by the feeder is 4.12 grams. This is above the minimum requirement of 4 grams. Therefore, the feeder meets the requirement for the sub-system.

Figure #17: Picture of one of the weigh-ins for the food dispensed by the feeder.

Trial	Weight (grams)	Meets Requirements
1	4.31	Yes
2	4.12	Yes
3	3.57	No
4	4.10	Yes
5	4.52	Yes
Average	**4.12**	**Yes**

Table #2: Data from the feeder weigh ins

3.4.2 Sensing the Rodent

The pressure sensor must be able to sense a 250-500 gram rodent and trigger to close the cage and dispense food. This requirement is key to the function of the device since it tells the device that the rat has entered the cage and the closing process can begin. The specific requirement is that “The sub-system must be sensitive enough to detect a 250-500 gram rodent.”

To verify this requirement, we followed the procedure outlined in Table 4 of Appendix A. We ran through the sequence of the device several times with various weights on top of the pressure sensor. Table #3 shows the data from each run through under various weights. For each trial, as soon as the device beeped the pressures sensor was triggered, the cage closed, and food was dispensed by the feeder. We also ran through the sequence using a rat. We had the rat step on the pressure sensor as it would do to use the device. The device triggered using just the rat’s weight as well. This verifies that the pressure sensor meets the requirement.

Weight (grams)	Does the pressure sensor trigger?
200	Yes
250	Yes
350	Yes
450	Yes
500	Yes
Rat	Yes

Table #3: Data from 250 g pressure sensor test

3.4.3 Sensing Weight Limit

The pressure sensor should not trigger if there is less than 50 grams of weight on the sensor. This 50-gram limit was chosen due to the weight of the food used for rats. A pellet of rat food is typically around

3-5 grams. With multiple pellets and other possible objects on the sensor, a 50-gram weight limit would ensure that the sensor does not accidentally trigger before the rodent enters the cage.

To verify this requirement, we followed the procedure outlined in Table 4 of Appendix A. Similar to the verification process for section 3.4.2, we placed weights on the pressure sensor while running through the sequence of the cage. Table #4 shows the data from the tests. For weights 50 grams and under, the pressure sensor does not trigger. For weights above 50 grams, the pressure sensor does trigger. During the final demo, we placed a few pellets of food onto the pressure sensor while running through the device's sequence. With the pellets on the sensor, the device did not trigger. It was only until I lightly pressured the sensor that it triggered. This verifies the sub-system met the requirement.

Weight (grams)	Does the pressure sensor trigger?
30	No
35	No
50	No
70	Yes
100	Yes
150	Yes

Table #4: Data from 50 g pressure sensor test

3.5 Return Sub-system

The return sub-system successfully meets all the requirements set out in the design document, which can be found in Table 5 of Appendix A. The return sub-system must control the orientation of the cage door to open and close the cage. The cage must also open and close within 5 seconds. This is to ensure that the pet cannot trigger the pressure sensor and then quickly escape, locking themselves out of the cage. While the cage door is closing, there is potential for an object in the doorway to get pinched especially if the cage door closes quickly. To make the cage door safe, the ultra-sonic sensor must detect objects in the doorway and stop the motor from fully closing the door.

3.5.1 Motor Orientation

The servo motor must be able to fully open or fully close the cage door. This is one of the most important functions of the cage since the main objective of the device is to let the pet out. For the motor attached to the door, at 0° orientation the cage door is fully closed. At 180° orientation, the cage door is fully open. The specific requirement is that "The servo motor must rotate between positions 0° and 180° within +/- 3° with commands from the microprocessor." If the sub-system fulfills this requirement, then the cage door will properly open and close.

To verify this requirement, we followed the procedure outlined in Table 5 of Appendix A. Figure #18 shows three pictures of the cage door with the servo motor attached. The first picture is of the cage door closed before the opening time. It is in the 0° orientation which is marked with a piece of tape. The second picture is of the cage door after opening. It successfully opens to 180° orientation as the rotor is parallel to the tape. The third picture is of after the cage door closes. It successfully returns to the 0° orientation as the rotor is parallel to the tape. Note that for the second and third picture, the cage door was taken off the rotor to make it more visible. This verifies that the sub-system fulfills the requirement.

Figure #18: Pictures showing the angle of the rotor in open and close positions

3.5.2 Motor Time Restraint

The servo motor must be able to open and close the cage quickly. For opening the cage door, this doesn't matter. However, for closing the cage door, it is important to close the cage door before the rodent could potentially escape. The specific requirement is that "The servo motor must completely open or close the cage door, from 0° to 180° and 180° to 0°, within 5 seconds +/- 0.5 second as long as the rodent is not in the way of the cage." 5 seconds was chosen as the time restraint. The servo motor could easily close the door within this time frame while, at the same time, fast enough that the rodent could not make it outside the cage before the door closes. If the motor can open and close the cage within the time restraint, then there is a very low chance of the pet rodent escaping and becoming locked out.

To verify this requirement, we followed the procedure outlined in Table 5 of Appendix A. Using a stopwatch, we timed the cage door opening and closing. Table #5 shows the data from this test. Both the time of opening and closing are within 5 seconds. Note that the motor closes the door much quicker than 5 seconds and much quicker than a rodent would be able to escape. This verifies that the sub-system fulfills the requirement.

Duration of Opening (seconds)	Duration of Closing (seconds)
4.24	1.73

Table #5: Times it takes the cage door to open and close

3.5.3 Ultra-Sonic Sensor

As a safety concern, the cage door should not close if there is an object in the doorway. Without any kind of sensor near the door, the rodent could get pinched when the cage door closes. The specific requirement is that "If the Ultra Sonic sensor detects the rodent within "18" cm of the cage door, the door should stop closing immediately." If the ultra-sonic sensor can fulfill the requirement, then there is a low chance of something getting hurt by the door closing and the device is safer for the pet.

To verify this requirement, we followed the procedure outlined in Table 5 of Appendix A. All three videos in Appendix B show the procedure in action and verify that the cage door re-opens and doesn't close until the object is removed from the doorway. We run through the device's sequence. While triggering the pressure sensor to close the door, we put our hand in the way of the doorway. When we do this, the cage door re-opens. After removing the hand, the door closes again. We also tested this with a pet rat. We put the rat near the doorway and triggered the pressure sensor. The door began to close but stopped and reopened when the rat moved near the doorway. When the rat moved away from the doorway, the cage door closed properly. This verifies the sub-system fulfills the requirement.

4. Cost and Schedule

4.1 Cost Analysis

The average salary of an Electrical Engineering graduate from the University of Illinois at Urbana-Champaign was $79,714 in the 2018-2019 school year [3]. This is equivalent to an hourly pay of about $38.32. We plan on working for an average of 12 hours each week. Since there are 14 weeks in a semester, we can calculate the total labor cost for each person.

Labor Cost = (Hourly Pay) * (Average Hours per Week) * (Total Weeks in Semester)

Labor Cost = ($38.32) * (12 hours/week) * (14 week) = $6,437.76

Name	Hourly Pay	Total Hours	Total Cost
Saurav Kumar	$38.32	168	$6,437.76
Christina Hejny	$38.32	168	$6,437.76
Avram Fouad	$38.32	168	$6,437.76
			$19,313.28

Table #6: Labor Cost

4.2 Parts

Name	Description	Manufacturer	Part #	Quantity	Cost	Ext Cost
Keypad	COM-14662 KEYPAD - 12 BUTTON	SparkFun Electronics	1568-1856-ND	1	$4.50	$4.50
Pressure Sensor	RP-C18.3-ST THIN FILM PRESSURE S	DFRobot	1738-SEN0294-ND	2	$5.00	$10.00
Micro controller	ARM Microcontrollers - MCU Mainstream Arm Cortex-M0 Access line MCU 64 Kbytes of Flash 48 MHz CPU, motor co	STMicroelectronics	511-STM32F051C8U7	1	$4.28	$4.28

7-Segment Display	1001 DISP 7SEG 0.56" QUAD WHT 14DIP	Adafruit Industries LLC	1528-1513-ND	1	$4.95	$4.95
Lithium Battery	BATTERY LITHIUM 3.7V 2.5AH	Adafruit Industries LLC	1528-1840-ND	1	$14.95	$14.95
DC-DC converter	DC DC CONVERTER 5V 5.6W	Pololu Corporation	2183-2123-ND	1	$10.61	$10.61
Voltage Regulator	IC REG LINEAR 3.3V 100MA 12MSOP	Analog Devices Inc.	LT3050EMSE-3.3#PBF-ND	1	$4.72	$4.72
USB battery charger	LI-ION LI-POLYMER CHARGER BOARD	Adafruit Industries LLC	1528-1833-ND	1	$6.95	$6.95
Servo Motor	SERVOMOTOR 6V FEETECH HI-TORQUE	Pololu Corporation	2183-3426-ND	2	$19.45	$38.90
Speaker	BUZZER MAGNETIC 3V 5X5MM SMD	CUI Devices	102-CMT-0525-75-SMT-TR-ND - Tape & Reel (TR)	2	$2.03	$4.06
Infrared Sensor	MOTION SENSOR	Murata Electronics	490-11915-ND	1	$3.07	$3.07
Crystal	CRYSTAL 32.768KHZ 9PF SMT	Fox Electronics	631-FK135EIWM0.032768-T3TR-ND - Tape & Reel (TR)	1	$1.49	$1.49
Subtotal						**$108.48**
Quoted Machine Shop Labor Hours	3 days	8 hours per day	56 $/hr			$1344.00
Quoted Team Labor Hours						$19,313.28
Total						$20,765.76

Table #7: Parts Cost

4.3 Schedule

Week	Saurav Kumar	Christina Hejny	Avram Fouad
2/21	Complete Design Document, Meet with Machine Shop to discuss Feeder Implementation, Start PCB Design, Create Flowchart of overall system, Prepare for Design Review, Order Parts	Complete Design Document, Meet with Machine Shop to discuss Feeder Implementation, Start PCB Design, Select New Microcontroller, Prepare for Design Review, Order Parts	Complete Design Document,Meet with Machine Shop to discuss Feeder Implementation, Start PCB Design, Order 1st round of parts, Prepare for Design Review, Order Parts
2/28	Finalize PCB Design and submit for Review, Take a look at code for generating speaker sound and Test on Breadboard	Finalize PCB Design, Simulate Pressure Sensor Resistance with her Rat, Hand over the Cage to Machine Shop	Finalize PCB Design, Test Power System on Breadboard,
3/7	Work on necessary changes to PCB if needed, Write code for Microcontroller to correctly gather input time from Keypad	Solder Parts onto PCB, Unit Test Servo Motor for the door and take measurements ono rotation angle	Write Code for Microcontroller to display user input time on 7-segment Display, Debug Power Sub-System if necessary
3/14	Spring Break	Spring Break	Spring Break
3/21	Write Code to read Analog Voltage at Ouput of FSR voltage divider, Debug any troublesome Sub-Systems	Finalize code for correctly opening and closing Cage Door, Debug any troublesome sub-systems	Begin writing code for Feeder to dispense food Order any remaining parts needed and fix any mistakes in PCB Design
3/28	Begin Wiring any required sub-systems together and enclose wires with Conduit Protective covering,	Make any necessary changes to cage design if needed, Begin training the rat on how to properly use our solution if necessary	Unit test Feeder Sub-System and ensure that it dispenses food at the correct time
4/4	Start testing overall design with everything together, take videos of results as well as any quantitative measurements, conduct any troubleshooting if required	Start testing overall design with everything together, take videos of results as well as any quantitative measurements, conduct any troubleshooting if required	Start testing overall design with everything together, take videos of results as well as any quantitative measurements, conduct any troubleshooting if required
4/11	Prepare for mock demo, conduct last-minute changes if needed	Prepare for mock demo, do conduct last-minute changes if needed	Prepare for mock demo, conduct last-minute changes if needed
4/18	Prepare for presentation	Prepare for presentation	Prepare for presentation
4/25	Work on final paper	Work on final paper	Work on final paper
5/2	Finalize lab notebook and teamwork evaluation.	Finalize lab notebook and teamwork evaluation.	Finalize lab notebook and teamwork evaluation.

Table #8: Schedule

5. Conclusion

5.1 Accomplishments

We successfully completed all our high-level requirements and were able to demonstrate complete functionality of all project components. First, the servo motor can fully open and close the cage door. Second, if the cage has been open for at least 2 hours [+/- 1 minute], the door closes within 3 seconds of the thin film pressure sensor being activated. Lastly, the cage door does not close during the designated 2 hour open-cage period under any circumstance unless the emergency "close" button is pressed.

The power sub-system can output a 5 V line to the other sub-systems with up to 3 A of current. It is also able to output a 3.3 V line with up to 300 mA of current. For our user interface sub-system, the keypad can successfully communicate the time entered to our control unit. The clear button functions properly and allows the user to reenter the desired time. The LCD can display the correct opening time set by the user. For the treat dispenser sub-system, the pressure sensor is sensitive enough that a 250–500 gram rodent would easily activate it while also not being triggered by weights below 50 grams. Finally for the return sub-system, the servo motor is able to fully open and close the cage door, the speaker beeps after the two-hour break period is over and in 15-minute intervals until the pet comes back. Our ultrasonic sensor can detect if the pet is too close to the door and stop the door from closing.

5.2 Uncertainties

One of the major challenges that faced us was that we couldn't program our microcontroller. When we first designed our PCB, we designed it with the intention to use SPI pins, which we later realized isn't the correct pin assignment for our specific microcontroller. Our STM32F0, can only be programmed using the Serial Debug Wire (SDW) pins. We designed and ordered a new PCB as soon as we discovered the problem but by the time the new PCB arrived and was soldered, there was very little time left before the demo.

5.3 Ethical Considerations

Our most important goal in this project was to make sure we are creating a safe environment for our pet to use. One of the first safety related problems we had to address was to make sure the pet doesn't chew on any of the wires used. To resolve that issue, we made sure to cover all wires, accessible to the pet, with steal conduits. The second issue would be the owner's failure to train the pet on how to use the feeder which could result in the pet undereating. To resolve that we spent two weeks training the pet on how to step on the pressure sensor after hearing the beep to get its treat. We will also make sure to inform anyone who uses our project in the future of that issue. IEEE code I1 states how we must "hold paramount the safety, health, and welfare of the public" [7]. On the other hand, we wanted to make sure the food in the dispenser's storage isn't accessible to the pet, to avoid the scenario where the pet finds it and overeats. To tackle this problem, we placed the dispenser in a high corner so that the pet has no access to the dispenser storage. Finally, to make sure the pet doesn't get injured by the movement of the cage door, we added an ultrasonic sensor that sends a signal to our control unit to open the door back up if it detects an object near it.

5.4 Future Work

We think our project has a lot of room for improvement and has the potential to be a great commercial product. Our project was built for a pet rat as it is the only pet that we had access to. Since rats are very smart it made it easy to train our pet in a short period of time. Although our project was created for a pet rat, we think our project is scalable to a lot more pets like cats, dogs and even birds. Furthermore, our project was built for only one pet to use at a time, but we think it could be modified to incorporate more than one. If we were to continue the project past the time frame of this class, we would attach RFID tags to the pet collars. This would potentially allow for more than one pet to use the cage.

References

[1] "Family Pets: Pros and Cons for Kids." National Poll on Children's Health, https://mottpoll.org/reports/family-pets-pros-and-cons-kids.

[2] Christy. "Diet and Exercise for Pet Rats." All Our Paws, 28 May 2016, http://allourpaws.com/rodents/diet-and-exercise-for-pet-rats/.

[3] "Pololu - FEETECH High-Torque Servo FS5115M," www.pololu.com. https://www.pololu.com/product/3426

[4] LT3050 - 100ma, Linear Regulator with Precision ... - Analog Devices. https://www.analog.com/media/en/technical-documentation/data-sheets/3050fa.pdf.

[5] "Rat Care Guidelines." [Online]. Available: https://www.in.gov/boah/files/Rat-Care.pdf

[6] "CMT-0525-75-SMT-TR." CUI Devices, https://www.cuidevices.com/product/resource/3dmodel/cmt-0525-75-smt-tr.

[7] IEEE, "7.8 IEEE Code of Ethics," Institute of Electrical and Electronics Engineers, ieee.org, 2022. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed Feb. 3, 2022]

Appendix A: Requirement and Verification Table

Table 1   Power Sub-System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. Must be able to provide 3.3V +/- 6% for up to 300 mA of current.	1A. Connect a potentiometer in series with the 3.3 V output. 1B. Measure the output voltage and current of the linear regulator using an oscilloscope while adjusting the potentiometer knob. 1C. Check whether the output voltage is within 3.3 V +/- 6% for currents up to 300 mA	Y
2. Must provide 5V +/- 8% for up to 3 A of current.	2A. Connect a potentiometer in series with the 5 V output. 2B. Repeat step 1B. 2C. Check whether the output voltage is 5 V +/- 8% for currents up to 3A.	Y

Table 2 User-Interface Sub-System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. The “*” button must be able to reset the selected “open” time.	1A. First enter an arbitrary time like 12:30 by pressing the 1, 2, 3, and 0 buttons on the keypad. 1B. Press the “*” button on the keypad. Check whether the “open” time previously shown on the LCD is cleared.	Y
2. The microcontroller must correctly process the time sequence entered on the keypad by the user.	2A. Wire each pin of the keypad to a separate I/O pin in the microcontroller. 2B. Create an array in software that takes in values from the microcontroller I/O pins being used for this test. 2C. Press an arbitrary key sequence of 1, 2, 5, 0. This represents an “open” time of 12:50. Check the serial monitor to see if the array matches the key sequence that was pressed.	Y
3. The LCD must be able to display the selected time input processed by the microcontroller.	3A. Correctly wire the LCD according to its datasheet. 3B. In software, create code that prints the array of characters entered by the user onto the LCD. 3C. Repeat step 2C but verify the contents of the LCD instead.	Y

Table 3 Control Sub-System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. The microcontroller should only check if the pressure sensor is triggered after the two hours timer has finished.	1A. Place a 250 g weight on the pressure sensor before the two hours are done and make sure the door doesn't move and the feeder doesn't release any food.	N
2. The control sub-system must send a digital high signal to the speaker 2 hours ± 1 minute after the programmed time.	2A. Set a desired time for the cage to open and ensure the speaker beeps two hours after the door opens.	N
3. The control sub-system must be able to set and keep track of the real time ± 1 minute.	3A. Set up the STM 32 RTC in software. 3B. Create variables for the current hour and minute from the RTC. 3C. Write code to print out the hour and minute variables on the LCD. 3D. Check whether the hour and minute variables on the LCD match the real hour and minute. Also ensure that the hour and minute variables increment correctly.	N

Table 4 Treat Dispenser Sub-System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. The feeder should be able to dispense at least 4 grams of food.	1A. Enter an arbitrary “open” time using the keypad. 1B. Once 2 hours after the “open” time has been reached, place a 300 g weight on the pressure sensor. 1C. If the feeder correctly dispenses the food, measure the weight of the food using a scale.	Y
2. The sub-system must be sensitive enough to detect a 250-500 gram rodent.	2A. Once again enter an arbitrary “open” time using the keypad. 2B. Once 2 hours after the “open” time has been reached, place a 250 g weight on the pressure sensor. Confirm whether the cage door closes as it should. 2C. Repeat steps 2A and 2B using heavier weights up to 500 g.	Y
3. The feeder should not dispense food at all if any force below 50 grams (+/- 3 g) is applied to the sensor.	3A. Repeat step 2A. Once 2 hours after the “open” time has been reached, place a 50 gram weight on the pressure sensor. 3B. Observe whether the feeder dispenses food. If it doesn’t, the system has worked correctly. 3C. Repeat steps 3A and 3B for various weights between 0-50 grams.	Y

Table 5 Return Sub-System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. The servo motor must rotate between positions 0° and 180° within +/- 3° with commands from the microprocessor.	1A. Set the time on the device so that the cage door will open soon. Verify that the position of the motor is at +/- 3°of 0° using a protractor. 1B. Once the cage door opens, verify that the position of the motor is at +/- 3°of 180° using a protractor. 1C. When the cage door closes, verify that the position of the motor is back to +/- 3°of 0° using a protractor.	Y
2. The servo motor must completely open or close the cage door, from 0° to 180° and 180° to 0°, within 5 seconds +/- 0.5 second as long as the rodent is not in the way of the cage.	2A. While completing verification 1, measure the time from when the motor starts rotating to when it finishes in position 180°$_n$ using a stopwatch. 2B. As the cage door closes, measure the time from when the motor starts rotating to when it finishes in position 0° using a stopwatch.	Y
3. If the Ultra Sonic sensor detects the rodent within "18" cm of the cage door, the door should stop closing immediately. This is to prevent the door from coming into contact with the rodent.	3A. Wire the Ultra Sonic sensor according to the datasheet and connect it to the appropriate microcontroller pin. 3B. Set any "open" time for the cage to open. Wait until the cage opens. Once the cage is open, wait for exactly 2 hours. 3C. After 2 hours, apply a force using your finger to the pressure sensor. While the door closes, put your hand close to the cage door and verify that the door stops closing.	Y

Appendix B: Video Links

1. Video shown in final presentation showing complete functionality
   - https://youtu.be/I75W1cO5MV8
2. 2 hour time lapse of cage during the open period
   - https://drive.google.com/file/d/1RFkL_KfqPr5GjiJd2fBYe2zcww1PpldC/view?usp=sharing
3. Extra Credit Video with Explanation
   - https://drive.google.com/file/d/1F9i_qANDCb-G4FNcn5CIoQiKQSKwsfrC/view?usp=sharing

Appendix C: Extra Figures

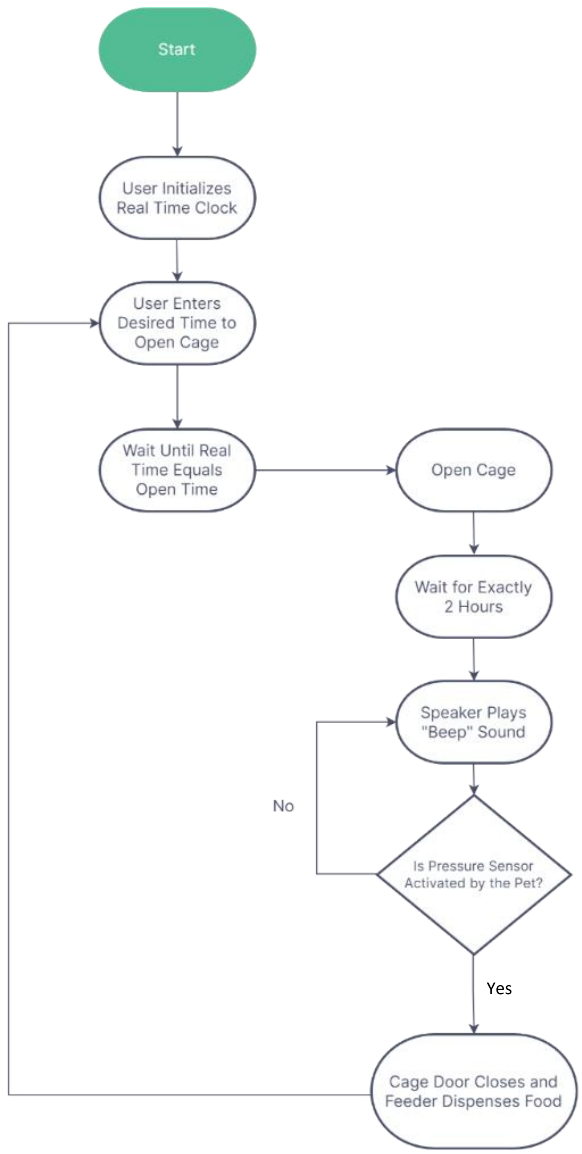

Figure #20: Flowchart of Overall Design