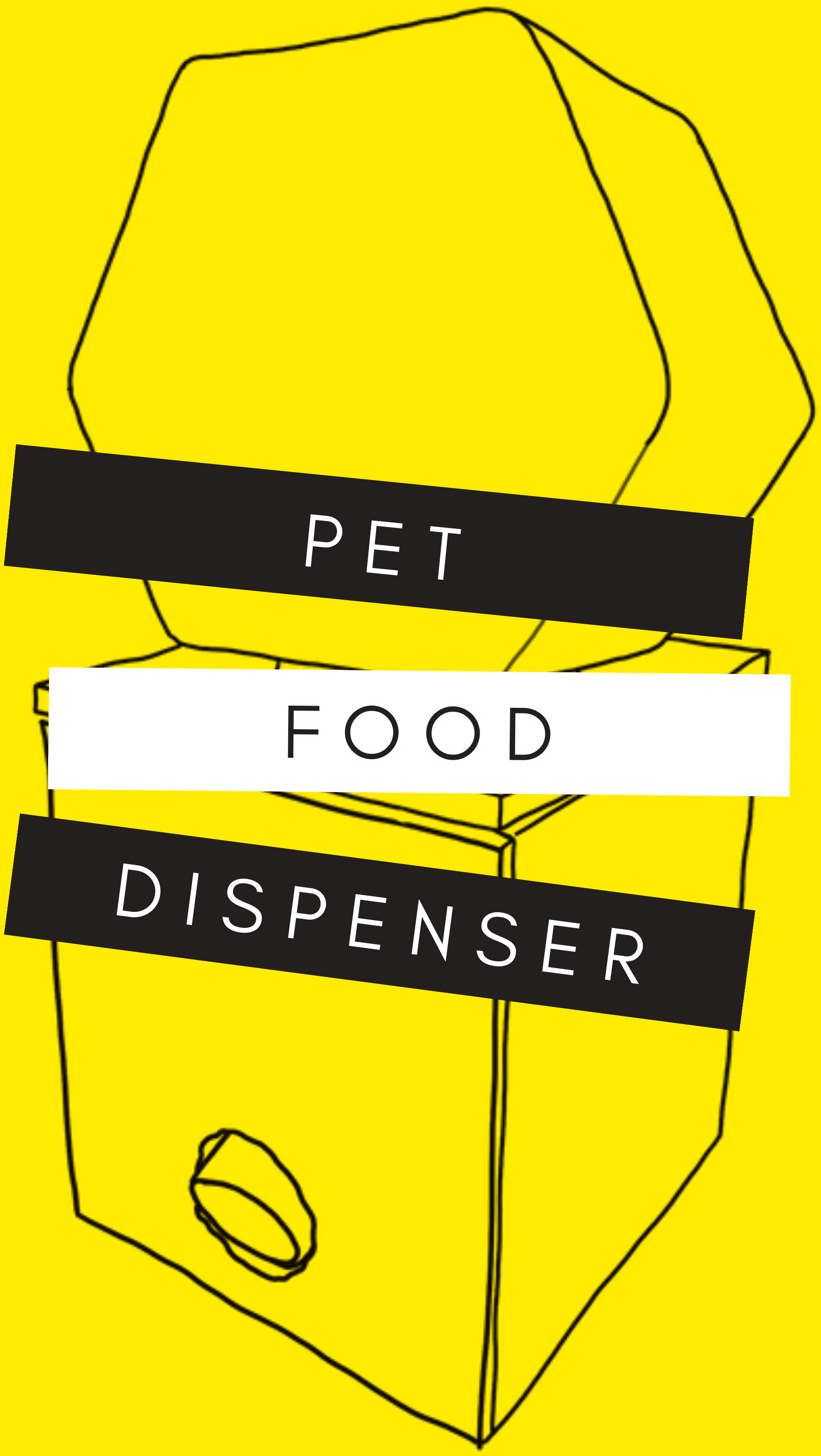
ME 102B
FINAL PROJECT REPORT

OPTION 1:
PHYSICAL PROTOTYPE



PET FOOD DISPENSER

BY CAITLYN LEE

SUBMISSION DATE
DECEMBER 8, 2020

Description of Product

This is Bumi; my sister Sidney's new puppy. We think she's a German Shepard/Border Collie/Labrador Retriever Mix and she's currently five months old. Bumi is an energetic and clumsy dog who's going through a growth spurt and eats about 5 cups of food per meal. As she ages, she will start to eat less and less food so I thought this would be a good opportunity for me to build a food dispenser for Bumi. I designed this system so that Sidney could walk over to the food dispenser during her Bumi's mealtime, set the amount of food she wants to feed her, and dispense the food with the push of a button.

Figure 1: Bumi, user of this product

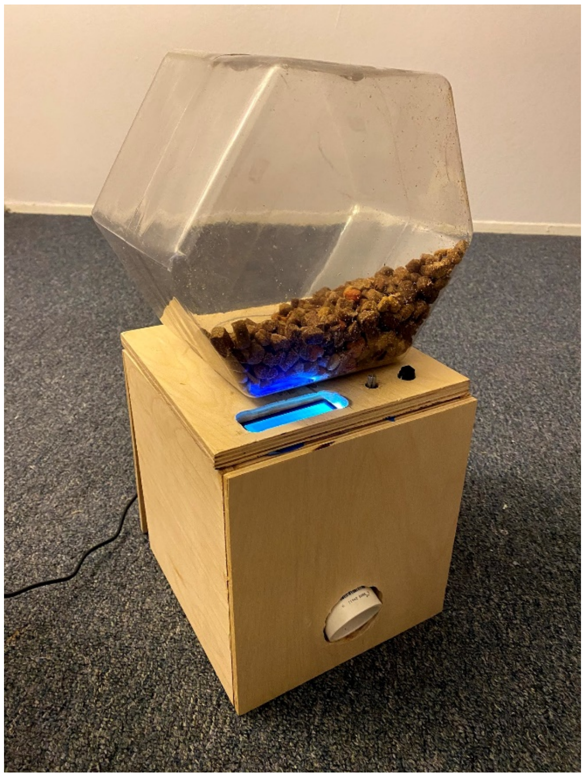


Figure 2: Entire pet food dispenser assembly

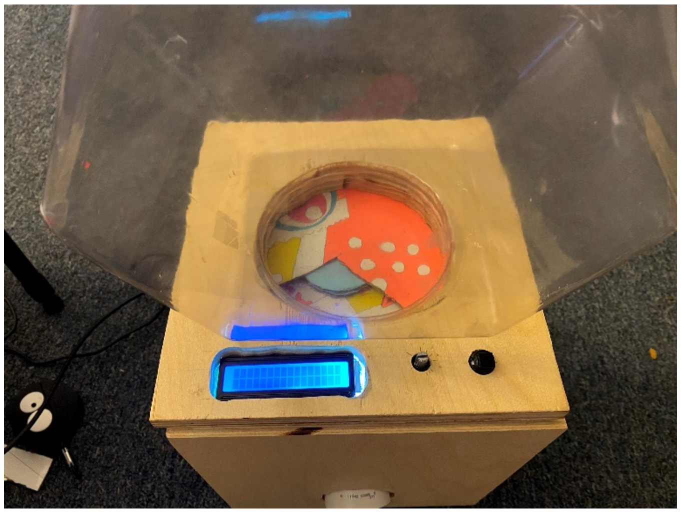


Figure 3: Interfaces for user include an LCD display, potentiometer, and pushbutton

For a video example of the device's operation, visit: https://youtu.be/foRIBS4j0dE

Housing/Food Dispensing Mechanism:

Housing: The wooden box contains the mechanism to dispense food into Bumi's food bowl and hides the circuit inside. The clear, hexagonal plastic container holds the dog food and can be removed from the wooden box.

Food Dispenser: A continuous servo motor pivots a cardboard wheel about its center. The wheel has a hole which allows food from clear plastic container to flow into the wooden box, into a paper funnel and into a PVC pipe which dispenses the food into Bumi's bowl. The servo motor is mounted in a metal can using cardboard and hot glue. The servo horns are attached to the wheel and placed onto the servo axis. To provide space for the circuit, the servo and metal can assembly are mounted on a wooden platform using wooden spacers.

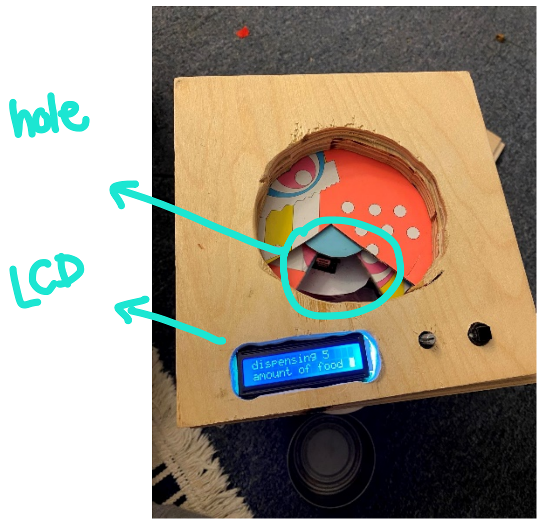


Figure 4: Dispensing food with wheel hole open

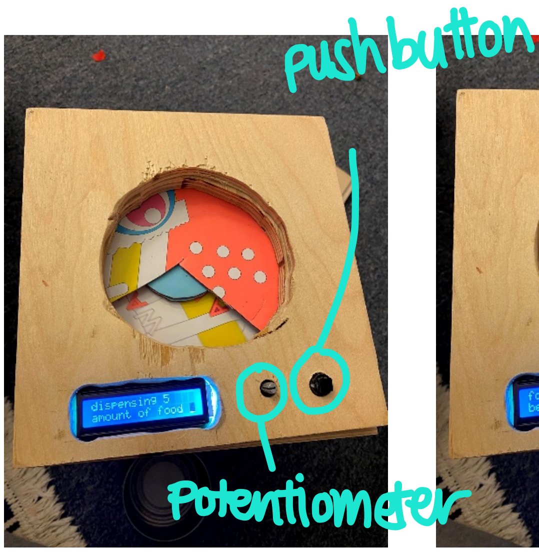


Figure 4: Dispensing food with wheel hole closed

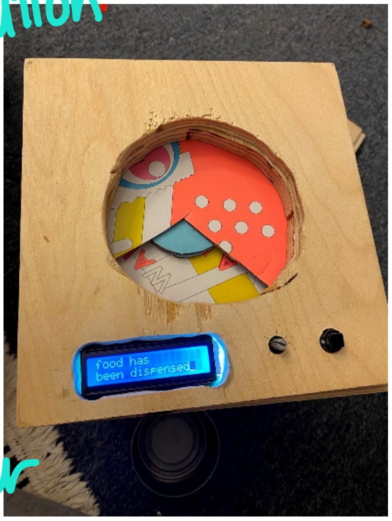


Figure 6: LCD display lets user know that the food is done being dispensed

Circuit Diagram (microcontroller, sensor, actuator, interface)

The main goal of this project was to learn how to use new components. The main components of this circuit includes: (1) a continuous rotation servo motor, (2) reed switch, (3) LCD display.

1. Continuous Servo Motor: I bought this motor (Pololu 1248 Continuous Rotation Servo Motor 5.1 kg-cm 6V, 11" Leads, 1.56" x 0.81" x 1.65" Size) for $13. This is a standard sized servo with two ball bearings on the output shaft to reduce friction and comes with a built-in H-Bridge driver that that controls the direction and speed with an input signal. A power supply and Pulse Width Modulation (PWM) signal is needed to control the motor. A pulse is sent to the motor every 20ms (50 Hz) and the pulse width varies between 1 – 2 ms. A pulse of 1.5 ms is the neutral point where the motor stops moving and can be adjusted using the little trim potentiometer. Any control signal above or below 1.5 ms is proportional to the difference between the command position and the neutral point position. Thus, the control signal sets the speed and direction. I added a 1K potentiometer (PDB181-E420K-102B Linear Potentiometer) from the lab kit so the amount of food released could be adjusted proportionally to the resistance being provided by this component. I also added a pushbutton switch (Single Pole Single Throw

Tactile Pushbutton) from the lab kit so the user can signal the product to dispense the food. A 4.7 kΩ resistor is used to pull-up the pushbutton signal and prevent a floating state signal.

2. Reed Switch: Since I misunderstood what a continuous servo motor was, (I assumed I could precisely control the position), I realized that I had no way of understanding where the hole in the dispensing plate was. Therefore, I added a reed switch (Reed Switch – Insulated) for $2 and a magnet (Magnet Square – 0.125") for $1. The magnet is in the spinning food dispensing plate and every time the plate rotates and releases some food, the magnet passes by the reed switch and the microcontroller keeps track of the number of times the plate spins. This controls the amount of food that is dispensed.
3. LCD Display: I bought this LCD display (SunFounder I2C 1602 Serial LCD Module Display 16x2) for $9. The LCD screen notifies the user that the button to dispense food is pressed and when the food is finished dispensing.

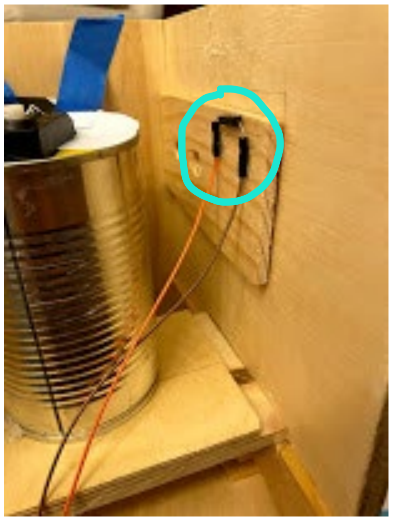


Figure 7: Reed Switch used as a position sensor of the spinning wheel

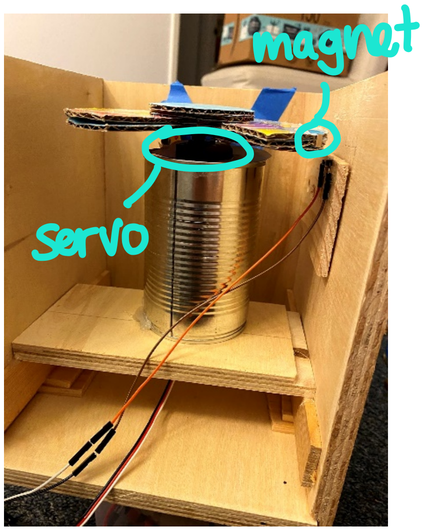


Figure 6: Wheel mounted onto servo motor with a magnet embedded (silver cube on right side of wheel)

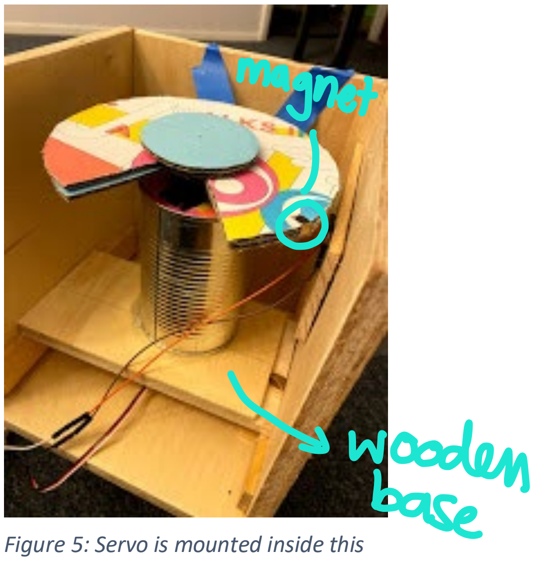


Figure 5: Servo is mounted inside this metal can which lies on a woodent base levitated by wooden spacers

In addition to these parts, I used temporary LEDs for debugging purposes. They weren't used for the final product, but were useful when writing the code and understanding how to use each component. I used the lab kit's 5V 2A switching power supply and female 12V DC power jack adapter connector. This powered the servo motor and LCD display. I used the lab kit's Lithium Ion Polymer Battery (3.7V 290mAh) to power the ESP32 when it is not powered by the USB and laptop. Ideally, the microcontroller should be powered through the 5V wall adapter so there isn't a need to recharge the lithium

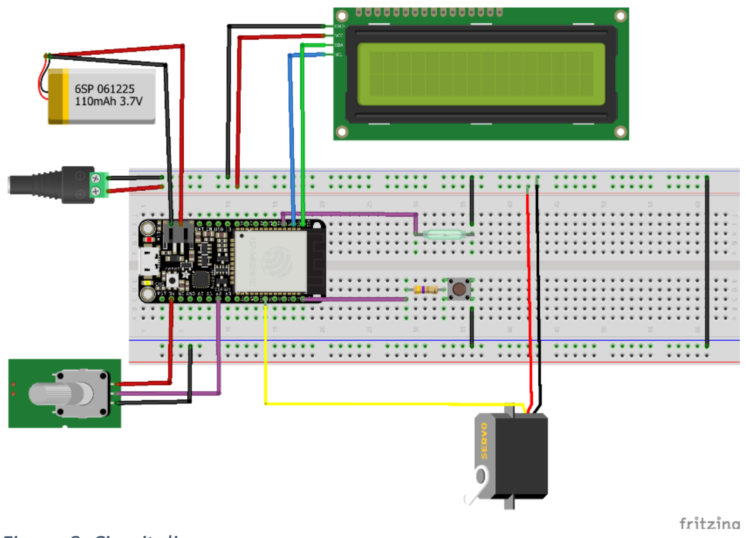


Figure 8: Circuit diagram

ion battery. I believe this can be done by routing the power into the V_{USB} pin of the ESP32. However, this could be risky because the microcontroller was not designed to do this and would essentially power the board by back powering the USB port. This could confuse or damage the circuit.

Finite State Machine Diagram & Demonstration

This is a simple machine since it is only "ON" when the pushbutton is pressed. Notice, there is no event that turns the machine back off. Once the pushbutton is pressed, the machine will perform all its services and then automatically turn itself "OFF."

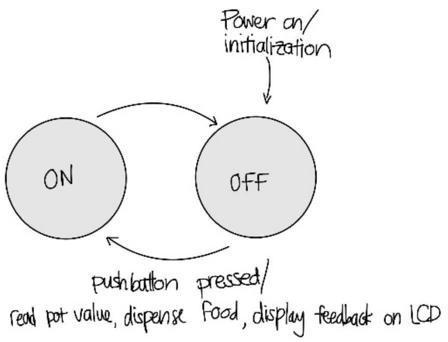


Figure 9: Finite state machine diagram

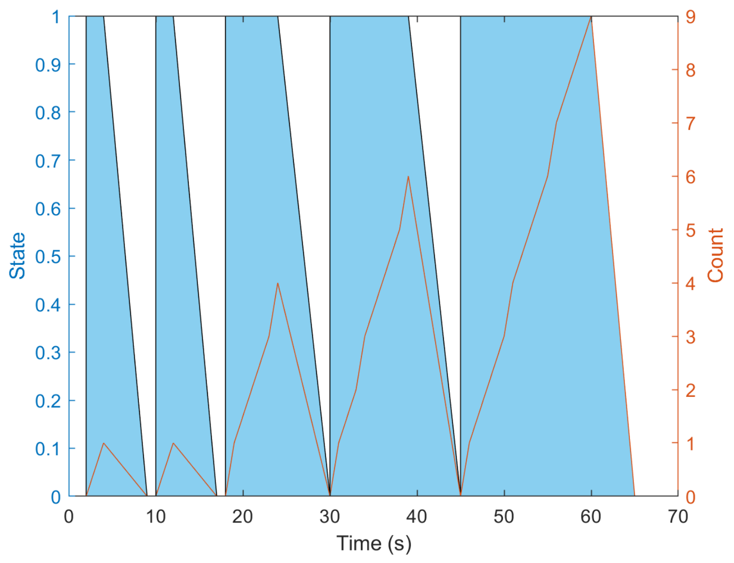
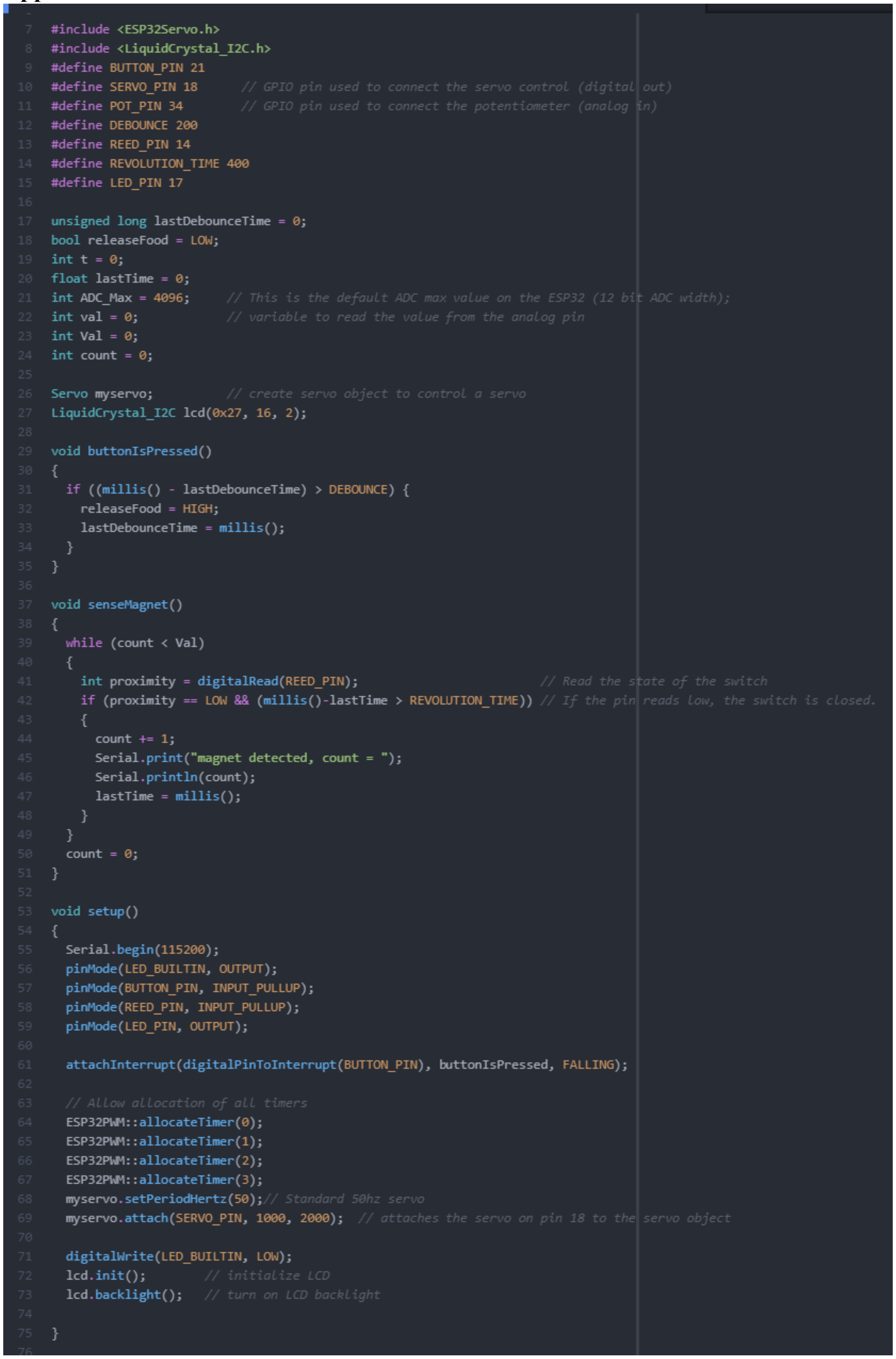


Figure 10: State 1 is the shaded blue region, State 0 is the white region. Number of counts (of revolustions) increase linearly because the servo motor spins at a constant speed.

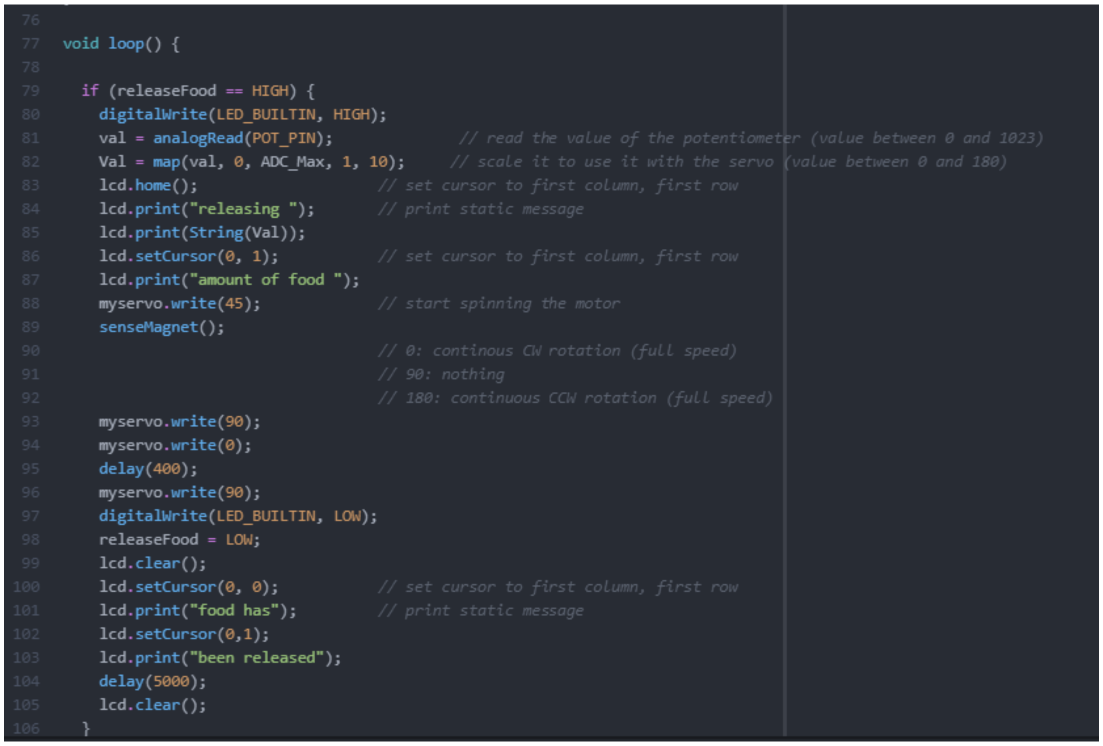
For the complete Arduino IDE code, see Appendix I.

When the pushbutton is pressed, the microcontroller reads the value of the potentiometer, displays the amount of food it was told to dispense on the LCD screen, and turns on the servo motor to start releasing food. The reed switch is used to count the number of times the food dispensing plate spins. When the number of counts is the same as the value from the potentiometer, the motor stops spinning. The LCD screen then lets the user know that the food dispensing is complete.

Appendix I: Arduino Code



```
#include <ESP32Servo.h>
#include <LiquidCrystal_I2C.h>
#define BUTTON_PIN 21
#define SERVO_PIN 18      // GPIO pin used to connect the servo control (digital out)
#define POT_PIN 34        // GPIO pin used to connect the potentiometer (analog in)
#define DEBOUNCE 200
#define REED_PIN 14
#define REVOLUTION_TIME 400
#define LED_PIN 17

unsigned long lastDebounceTime = 0;
bool releaseFood = LOW;
int t = 0;
float lastTime = 0;
int ADC_Max = 4096;     // This is the default ADC max value on the ESP32 (12 bit ADC width);
int val = 0;            // variable to read the value from the analog pin
int Val = 0;
int count = 0;

Servo myservo;          // create servo object to control a servo
LiquidCrystal_I2C lcd(0x27, 16, 2);

void buttonIsPressed()
{
  if ((millis() - lastDebounceTime) > DEBOUNCE) {
    releaseFood = HIGH;
    lastDebounceTime = millis();
  }
}

void senseMagnet()
{
  while (count < Val)
  {
    int proximity = digitalRead(REED_PIN);                          // Read the state of the switch
    if (proximity == LOW && (millis()-lastTime > REVOLUTION_TIME)) // If the pin reads low, the switch is closed.
    {
      count += 1;
      Serial.print("magnet detected, count = ");
      Serial.println(count);
      lastTime = millis();
    }
  }
  count = 0;
}

void setup()
{
  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(BUTTON_PIN, INPUT_PULLUP);
  pinMode(REED_PIN, INPUT_PULLUP);
  pinMode(LED_PIN, OUTPUT);

  attachInterrupt(digitalPinToInterrupt(BUTTON_PIN), buttonIsPressed, FALLING);

  // Allow allocation of all timers
  ESP32PWM::allocateTimer(0);
  ESP32PWM::allocateTimer(1);
  ESP32PWM::allocateTimer(2);
  ESP32PWM::allocateTimer(3);
  myservo.setPeriodHertz(50);// Standard 50hz servo
  myservo.attach(SERVO_PIN, 1000, 2000);  // attaches the servo on pin 18 to the servo object

  digitalWrite(LED_BUILTIN, LOW);
  lcd.init();        // initialize LCD
  lcd.backlight();   // turn on LCD backlight

}
```



```

void loop() {

  if (releaseFood == HIGH) {
    digitalWrite(LED_BUILTIN, HIGH);
    val = analogRead(POT_PIN);              // read the value of the potentiometer (value between 0 and 1023)
    Val = map(val, 0, ADC_Max, 1, 10);     // scale it to use it with the servo (value between 0 and 180)
    lcd.home();                   // set cursor to first column, first row
    lcd.print("releasing ");      // print static message
    lcd.print(String(Val));
    lcd.setCursor(0, 1);          // set cursor to first column, first row
    lcd.print("amount of food ");
    myservo.write(45);            // start spinning the motor
    senseMagnet();
                                  // 0: continous CW rotation (full speed)
                                  // 90: nothing
                                  // 180: continuous CCW rotation (full speed)
    myservo.write(90);
    myservo.write(0);
    delay(400);
    myservo.write(90);
    digitalWrite(LED_BUILTIN, LOW);
    releaseFood = LOW;
    lcd.clear();
    lcd.setCursor(0, 0);          // set cursor to first column, first row
    lcd.print("food has");        // print static message
    lcd.setCursor(0,1);
    lcd.print("been released");
    delay(5000);
    lcd.clear();
  }
```