#include <NewPing.h>

#include <EEPROM.h>

#define TRIG\_PIN 9

#define ECHO\_PIN 8

#define MAX\_DISTANCE 200  // Max sensor range in cm

NewPing sonar(TRIG\_PIN, ECHO\_PIN, MAX\_DISTANCE);

unsigned int emptyHeight = 40;  // Default until calibration

void setup() {

  Serial.begin(9600);

  // Load saved empty height from EEPROM

  EEPROM.get(0, emptyHeight);

  if (emptyHeight == 0 || emptyHeight > MAX\_DISTANCE) {

    emptyHeight = 40;

  }

  Serial.print("Loaded empty bin height: ");

  Serial.print(emptyHeight);

  Serial.println(" cm");

  Serial.println("Press Enter to calibrate bin as empty.");

}

void loop() {

  delay(2000);

  unsigned int distance = sonar.ping\_cm();

  // Check if Enter is pressed

  if (Serial.available()) {

    char c = Serial.read();

    if (c == '\n' || c == '\r') {

      if (distance > 0 && distance <= MAX\_DISTANCE) {

        emptyHeight = distance;

        EEPROM.put(0, emptyHeight);

        Serial.print("✅ Calibration complete. Empty bin height set to: ");

        Serial.print(emptyHeight);

        Serial.println(" cm");

      } else {

        Serial.println("⚠️ Calibration failed: distance out of range.");

      }

    }

    // Clear input buffer

    while (Serial.available()) Serial.read();

  }

  // Print current fullness

  Serial.print("Distance: ");

  Serial.print(distance);

  Serial.print(" cm");

  if (distance == 0 || distance > emptyHeight) {

    Serial.println(" - Status: Out of range");

  } else {

    int fullness = ((emptyHeight - distance) \* 100) / emptyHeight;

    fullness = constrain(fullness, 0, 100);

    Serial.print(" - Fullness: ");

    Serial.print(fullness);

    Serial.println("%");

        if (fullness >= 90) {

      Serial.println("Full");

    } else if (fullness >= 50) {

      Serial.println("Half-full");

    } else if (fullness > 0) {

      Serial.println("Low");

    } else {

      Serial.println("Empty");

    }

  }

  }

import serial

import time

import sys

import msvcrt  # Use msvcrt for non-blocking input on Windows

# Set up the serial port (adjust port as needed)

arduino = serial.Serial(port='COM4', baudrate=9600, timeout=2)

def read\_serial():

    """Function to read data from Arduino."""

    if arduino.in\_waiting:

        # Decode data while ignoring invalid characters

        return arduino.readline().decode(errors='ignore').strip()

    return None

print("Connected. Press Enter to calibrate, or 'q' to quit.")

try:

    while True:

        # Non-blocking input: Check if there’s any serial data

        data = read\_serial()

        # If there's data, print it to the console

        if data:

            print(data)

        # Check if Enter was pressed using msvcrt for non-blocking input

        if msvcrt.kbhit():  # Checks if a key is pressed

            user\_input = msvcrt.getch().decode()  # Get the keypress (as byte, then decode)

            if user\_input.lower() == 'q':  # Exit loop if 'q' is pressed

                break

            elif user\_input == '\r':  # If Enter is pressed, calibrate

                arduino.write(b'\n')  # Send Enter to Arduino (trigger calibration)

                print("Calibrating...")

        # Sleep to keep frequency of data reading consistent

        time.sleep(0.1)  # Adjust the delay based on Arduino's frequency

except KeyboardInterrupt:

    pass  # Handle KeyboardInterrupt exception

finally:

    arduino.close()  # Close the serial port

Distance: 108 cm - Status: Out of range

Distance: 17 cm - Status: Out of range

Calibrating...

✅ Calibration complete. Empty bin height set to: 16 cm

Distance: 16 cm - Fullness: 0%

Empty

Distance: 5 cm - Fullness: 68%

Half-full

Distance: 14 cm - Fullness: 12%

Low