Trideum

CHARLES SMITH, JUSTIN CARNEY, IAN ROBERSON

# Cyber Physical Systems

## 1. Overview

What are cyber physical systems (CPS)?

- Critical to the operations of major structures such as:
  - Oil pipelines
  - Traffic lights
  - Electric grids
- Critical to the operations of basic building systems such as:
  - Elevators
  - Temperature regulators
  - Secure access methods

## 2. Problems

Why do we care about cyber physical systems?

- Insecure CPS can lead to massive damage resulting in financial loss or loss of life, in extreme cases

How do we prevent this?

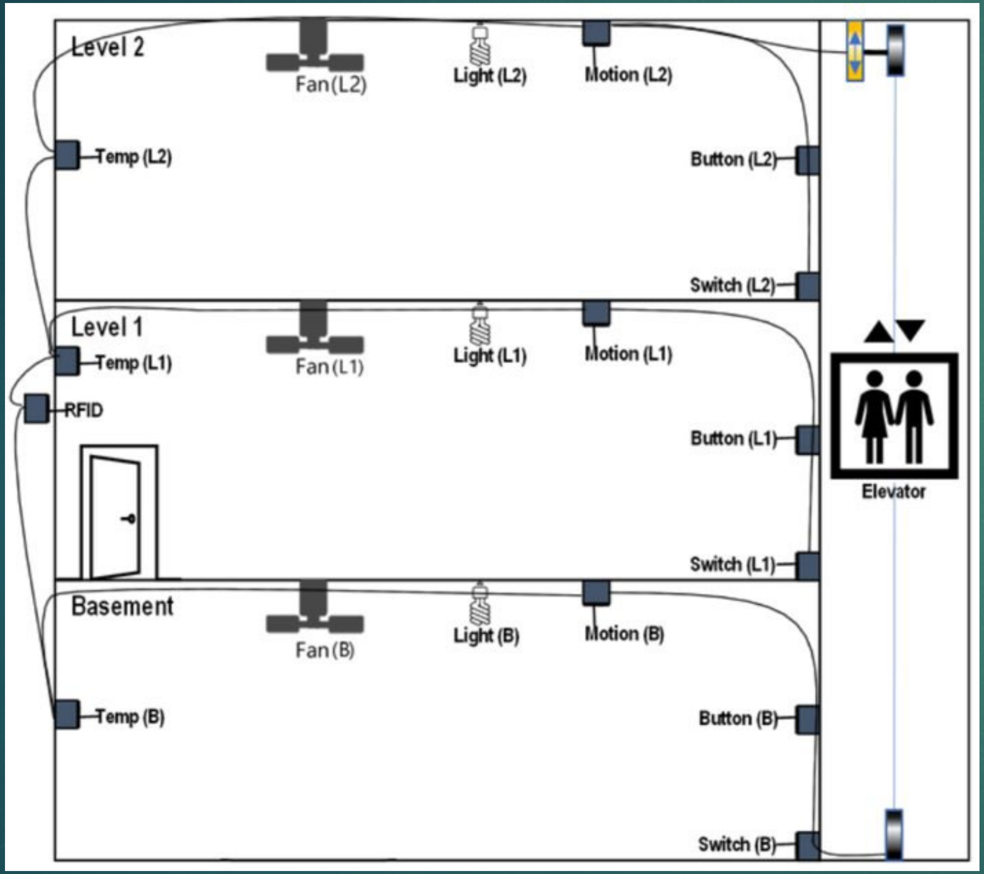
- Raise awareness of the importance of cyber security in the simplest of applications
- By making secure cyber physical systems of course!

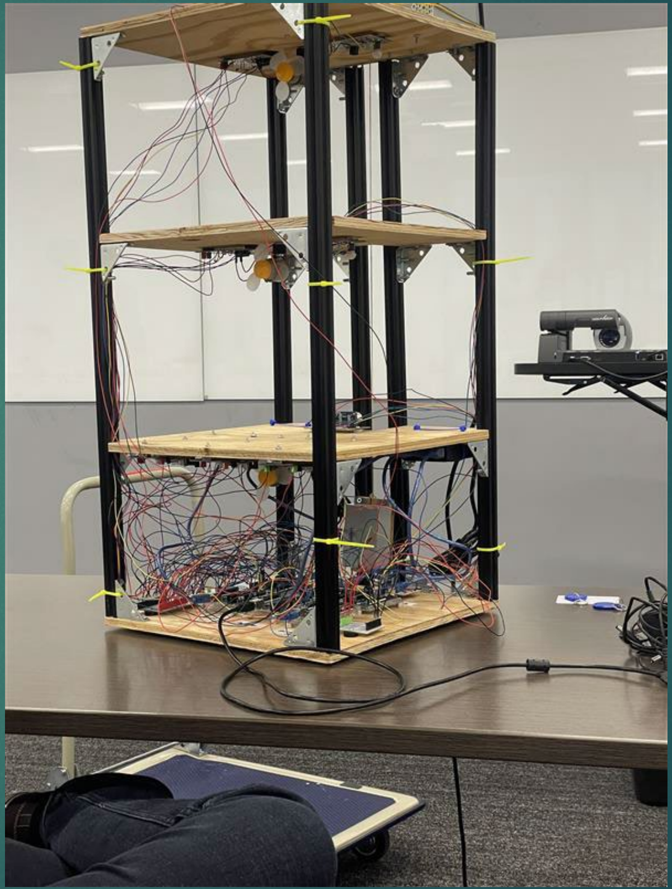
## 3. Goals

Design a building model with cyber physical systems and identify potential risks to the building.

- HVAC system
- Elevator
- RFID scanner
- Motion Sensor

# 4. Building Design





## 5. Solution Overview

A 3-story building model built with:

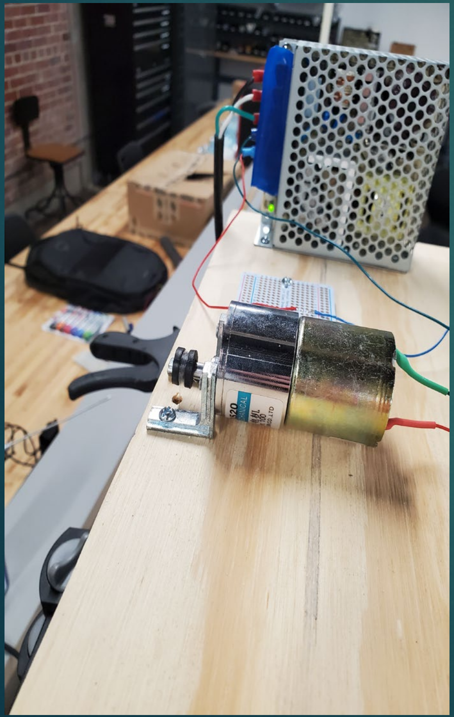
- 4 cut planks of plywood for the roofs and floors of each floor
- Aluminum rods to hold the building together
- Wires run through the poles to each floor's components

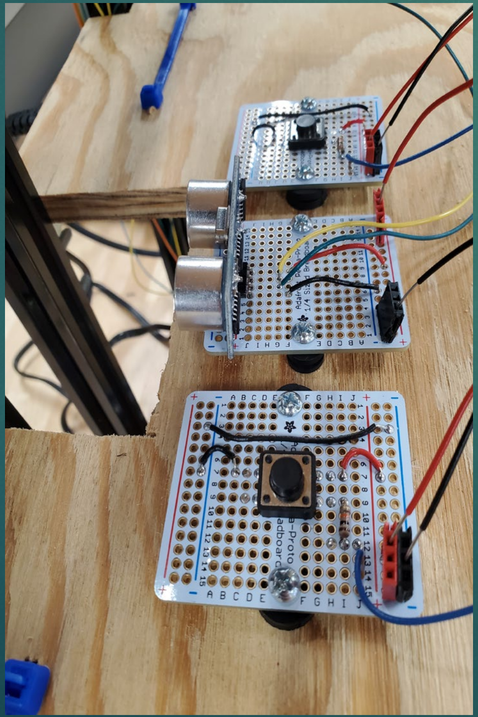
Physical systems controlled by:

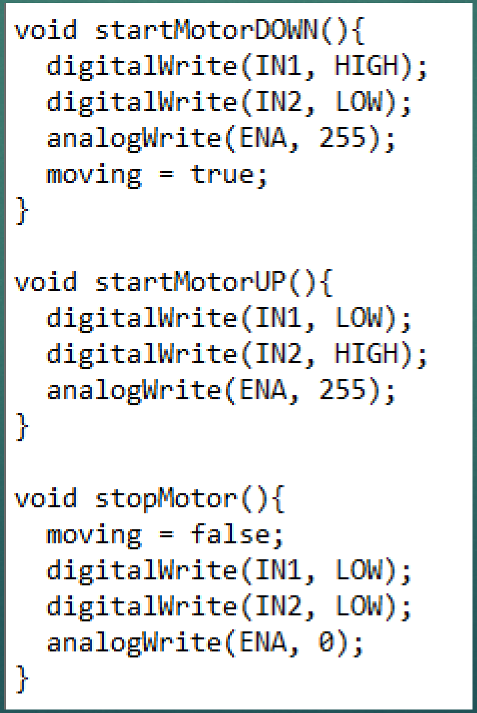
- 6 separate Arduinos 1 for each system (elevator, buzzer, motion sensor, HVAC, RFID, master)
- Status displayed with a UI written in Python

## 6. Elevator

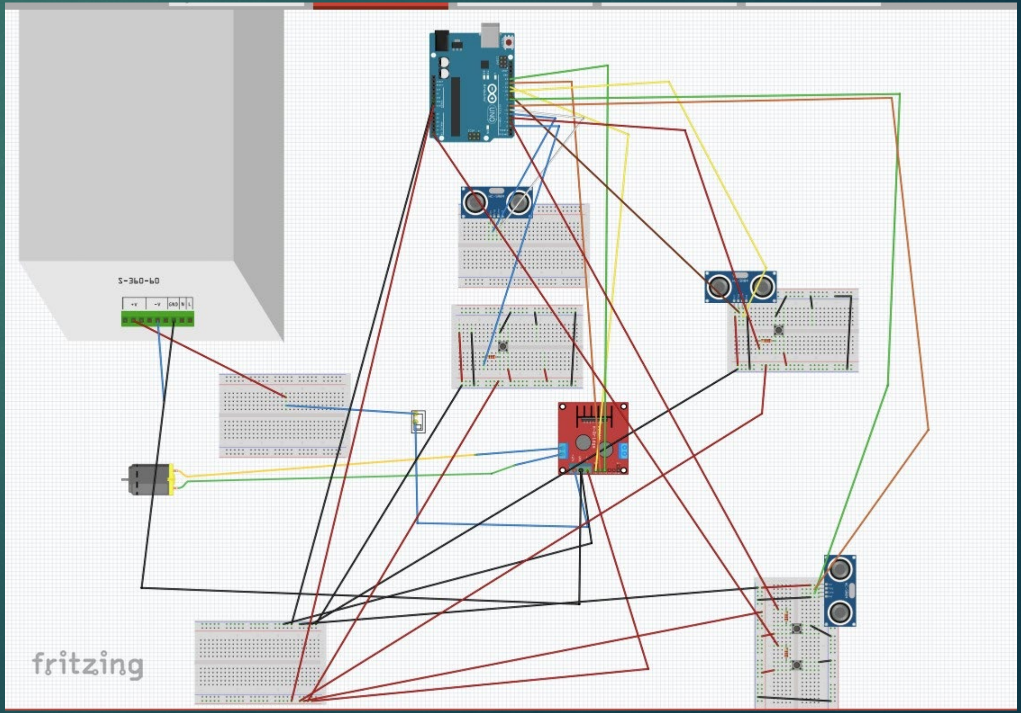
- Motor positioned on the top of the building with rope attached to the elevator
- Buttons on each floor used to call elevator to that floor
- Sensors detect stop the elevator when it has reached the requested floor
  - Stops elevator when blockage is detected
  - If 2 sensors are blocked an error is flagged and the elevator becomes inoperable
  - Arduino keeps track of which floor the elevator is on and sends that info to the master Arduino



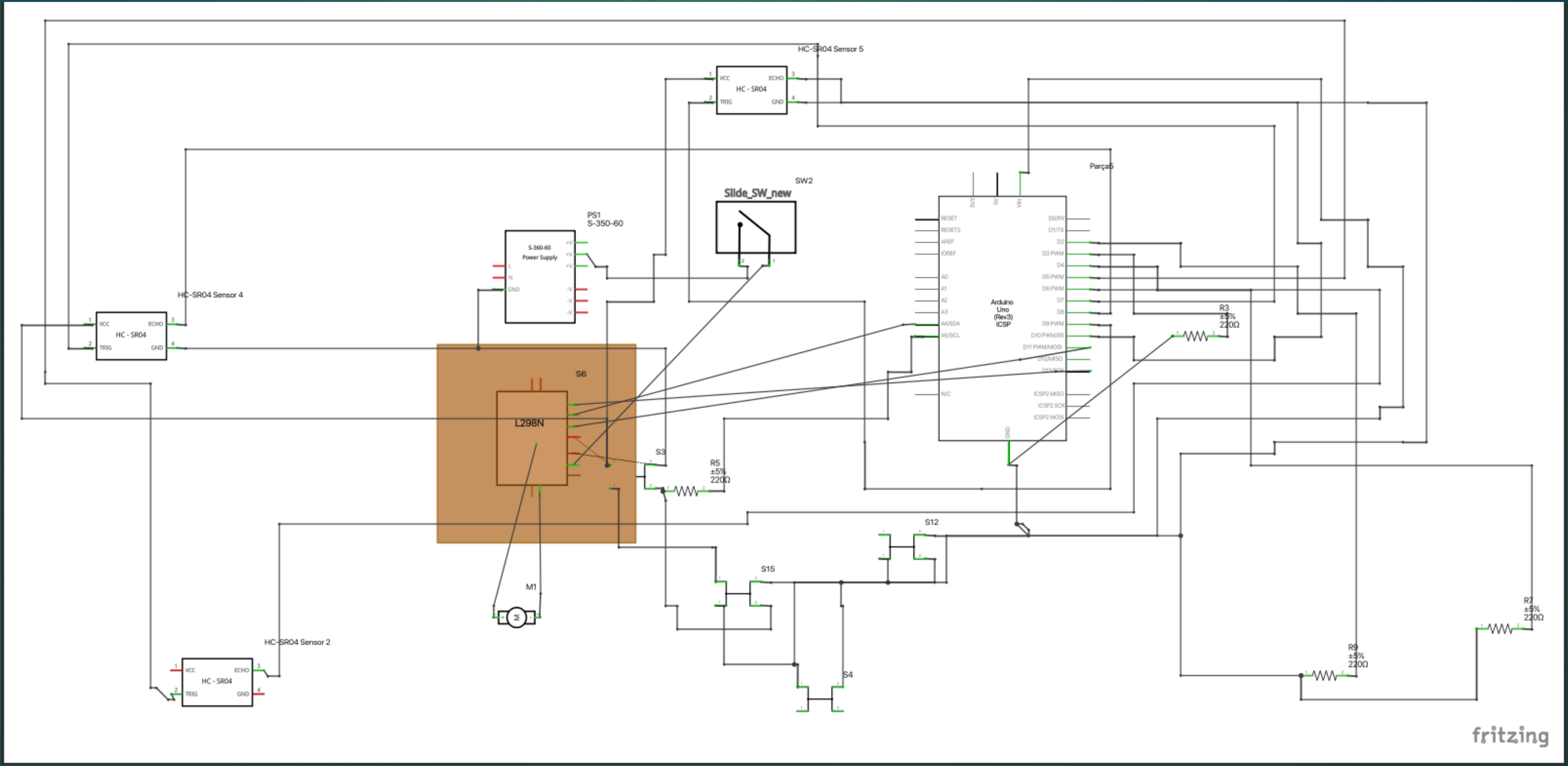




```
void startMotorDOWN(){
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  analogWrite(ENA, 255);
  moving = true;
}

void startMotorUP(){
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  analogWrite(ENA, 255);
}

void stopMotor(){
  moving = false;
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  analogWrite(ENA, 0);
}
```

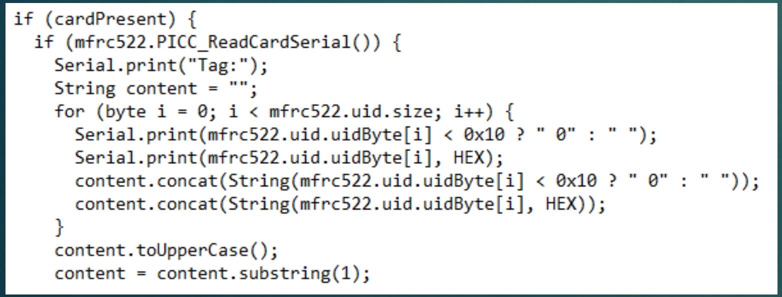


# 7. ELEVATOR MODULE

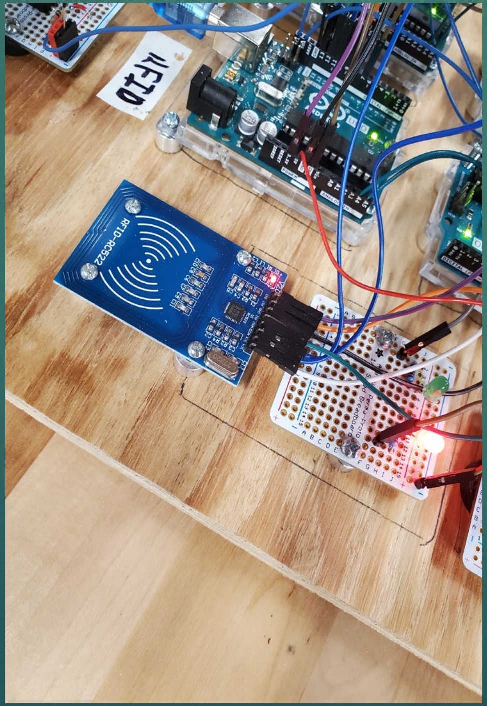


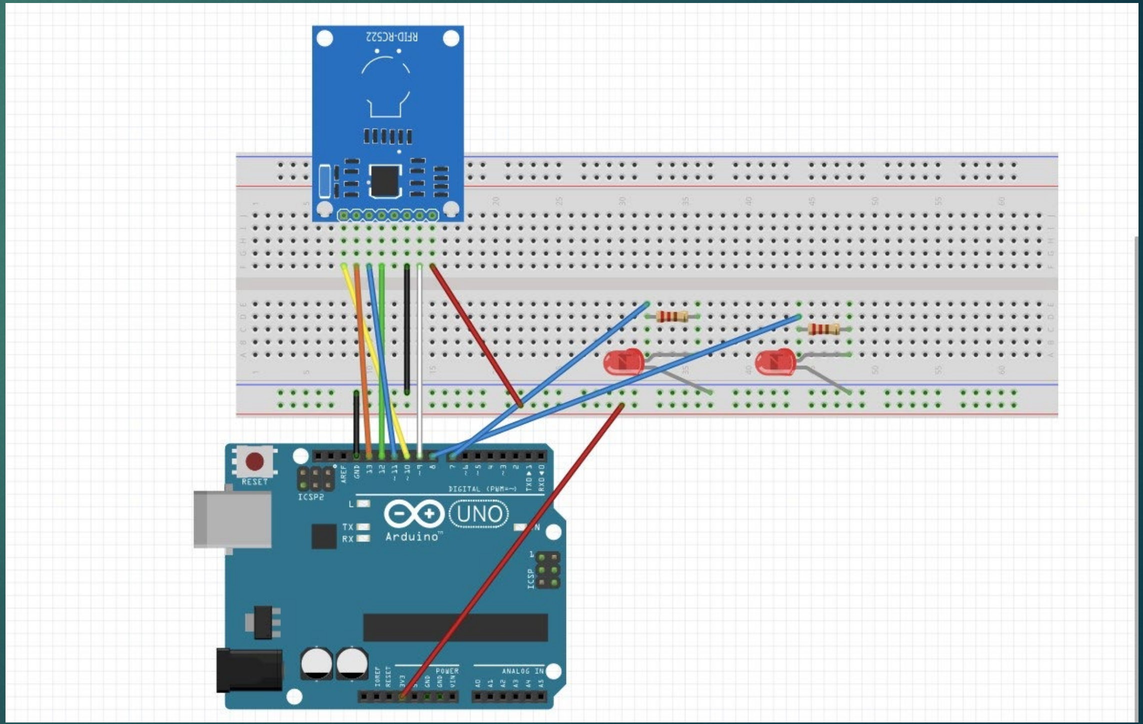
# 8. RFID

- Uses radio scanner to receive digital data from ID card
- If ID values are in the table, send corresponding name to the master

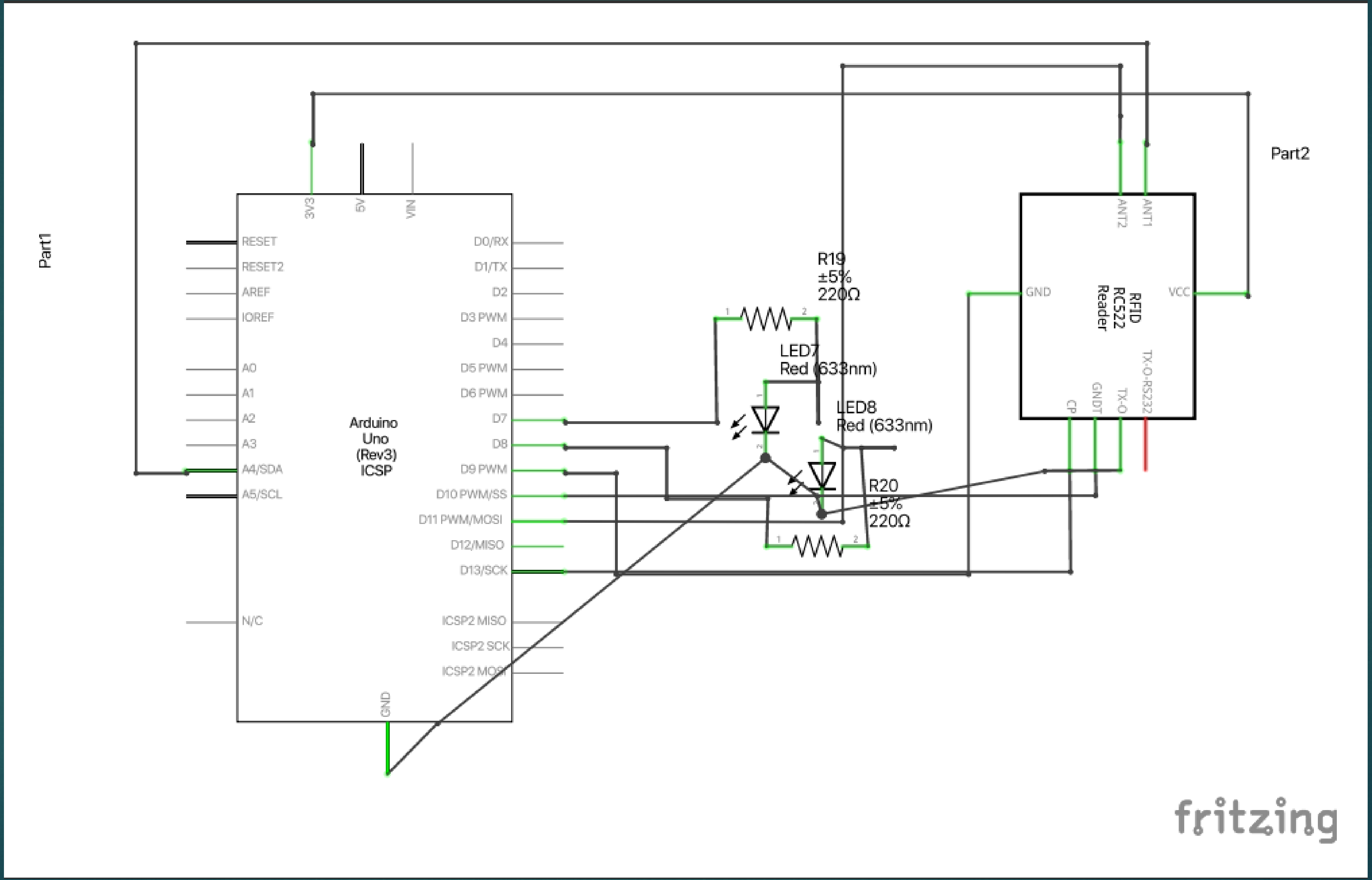


```
if (cardPresent) {
  if (mfrc522.PICC_ReadCardSerial()) {
    Serial.print("Tag:");
    String content = "";
    for (byte i = 0; i < mfrc522.uid.size; i++) {
      Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
      Serial.print(mfrc522.uid.uidByte[i], HEX);
      content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
      content.concat(String(mfrc522.uid.uidByte[i], HEX));
    }
    content.toUpperCase();
    content = content.substring(1);
```



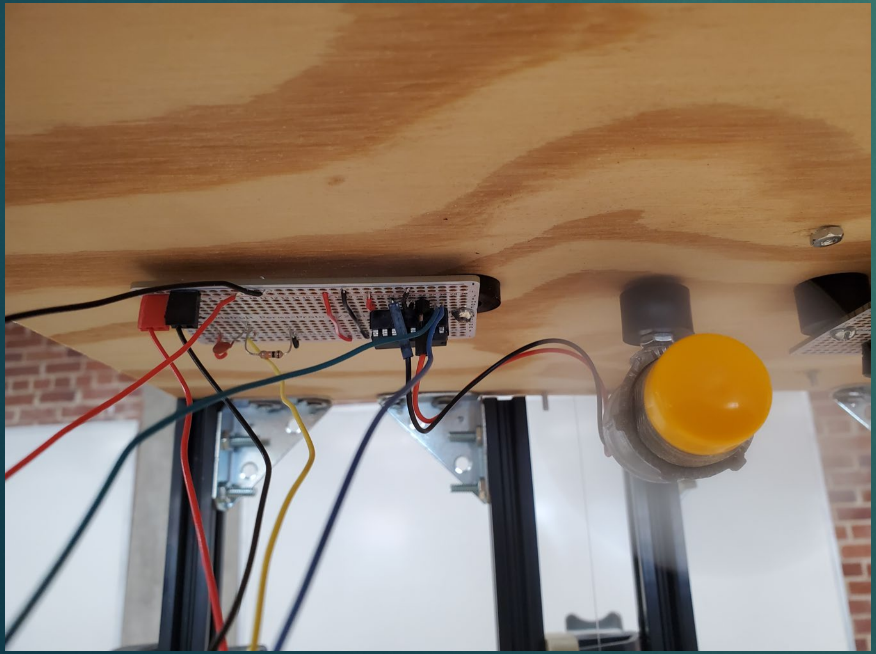


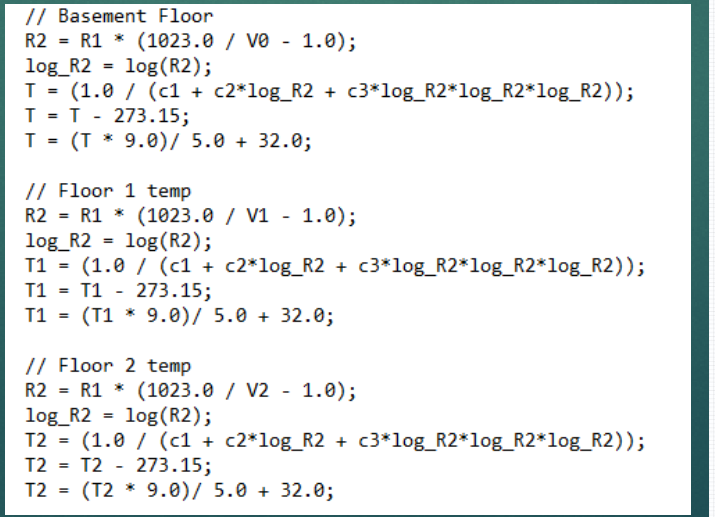
# 9. RFID MODULE



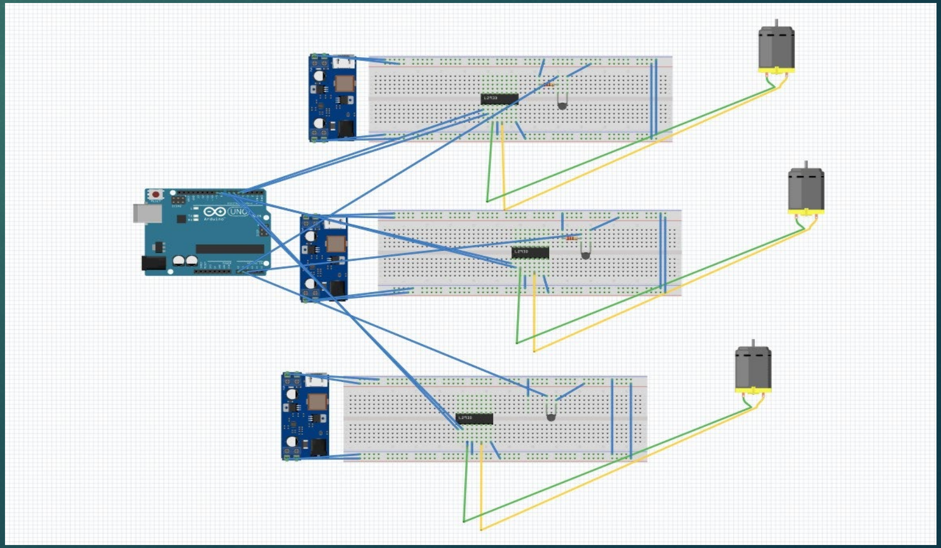
# 10. HVAC

- Uses thermistors to calculate temperature
  - Thermistors don't measure temperature directly; they change in resistance based on temperature
- If temperature gets too high, then the fans will spin
- Temperature thresholds are below 40 and above 90 degrees, if temperature reaches those values, an error will be sent to the UI

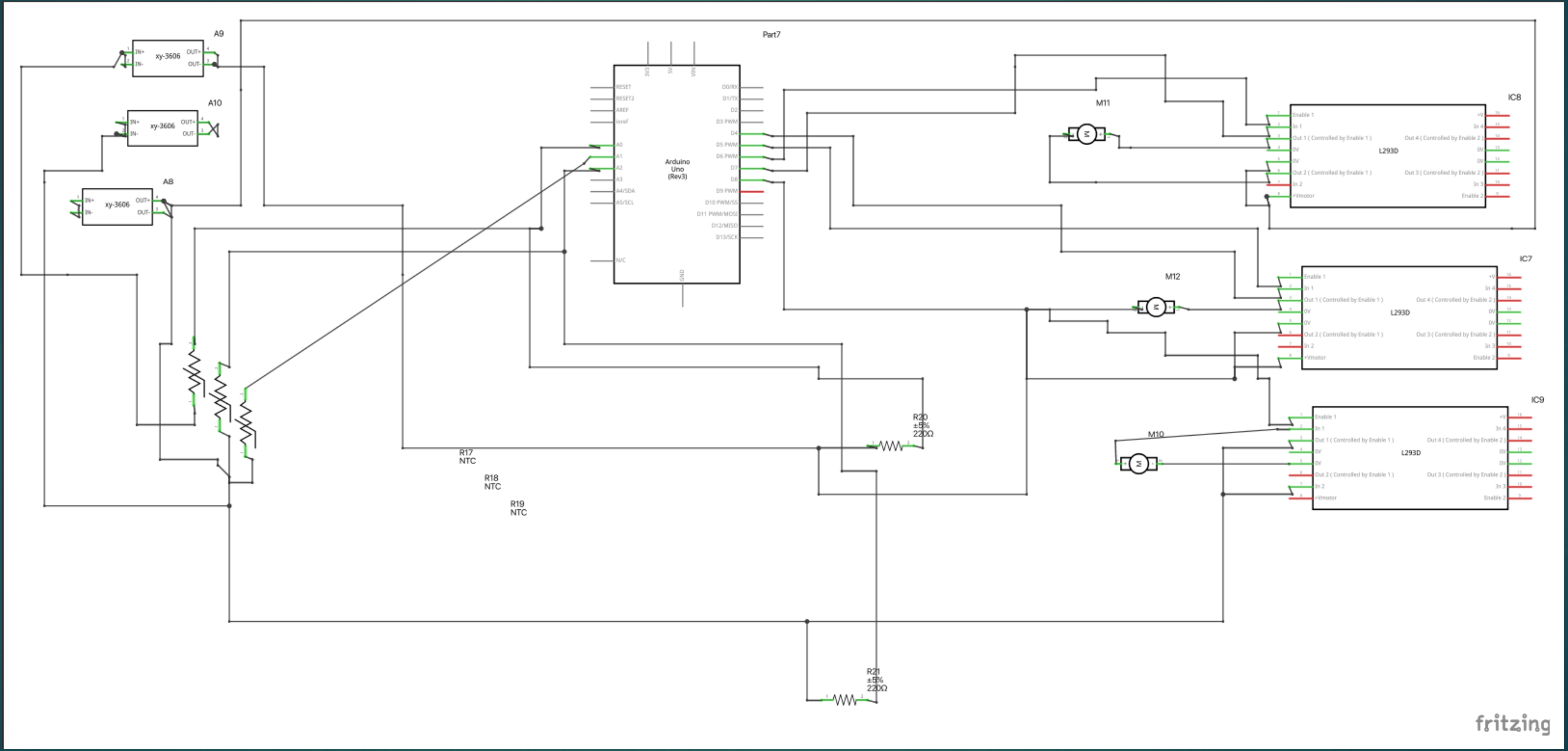




```
// Basement Floor
R2 = R1 * (1023.0 / V0 - 1.0);
log_R2 = log(R2);
T = (1.0 / (c1 + c2*log_R2 + c3*log_R2*log_R2*log_R2));
T = T - 273.15;
T = (T * 9.0)/ 5.0 + 32.0;

// Floor 1 temp
R2 = R1 * (1023.0 / V1 - 1.0);
log_R2 = log(R2);
T1 = (1.0 / (c1 + c2*log_R2 + c3*log_R2*log_R2*log_R2));
T1 = T1 - 273.15;
T1 = (T1 * 9.0)/ 5.0 + 32.0;

// Floor 2 temp
R2 = R1 * (1023.0 / V2 - 1.0);
log_R2 = log(R2);
T2 = (1.0 / (c1 + c2*log_R2 + c3*log_R2*log_R2*log_R2));
T2 = T2 - 273.15;
T2 = (T2 * 9.0)/ 5.0 + 32.0;
```

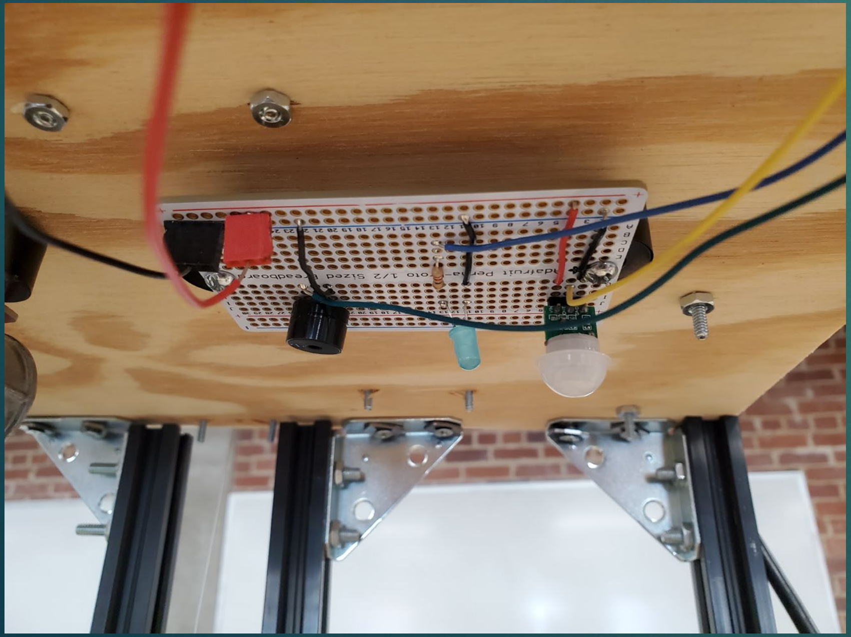


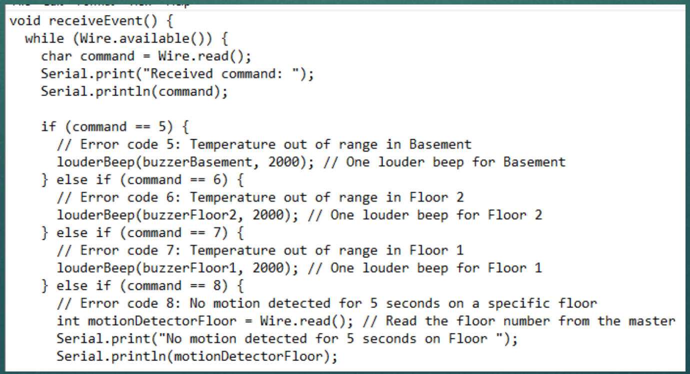
# 11. HVAC MODULE



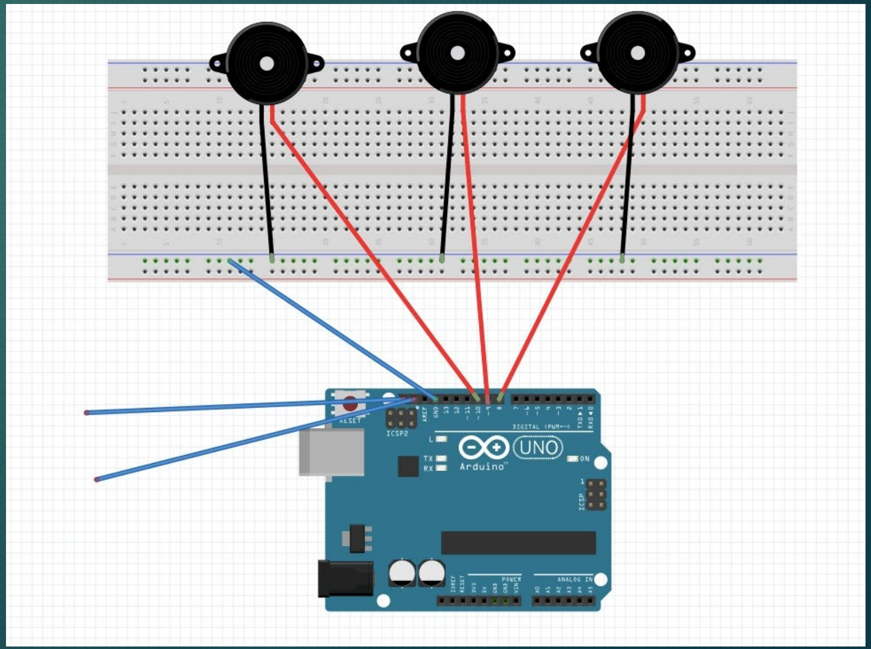
## 12. Buzzer

- Communicates with master Arduino to determine if problems are detected and if problems are detected then beep until problem is resolved

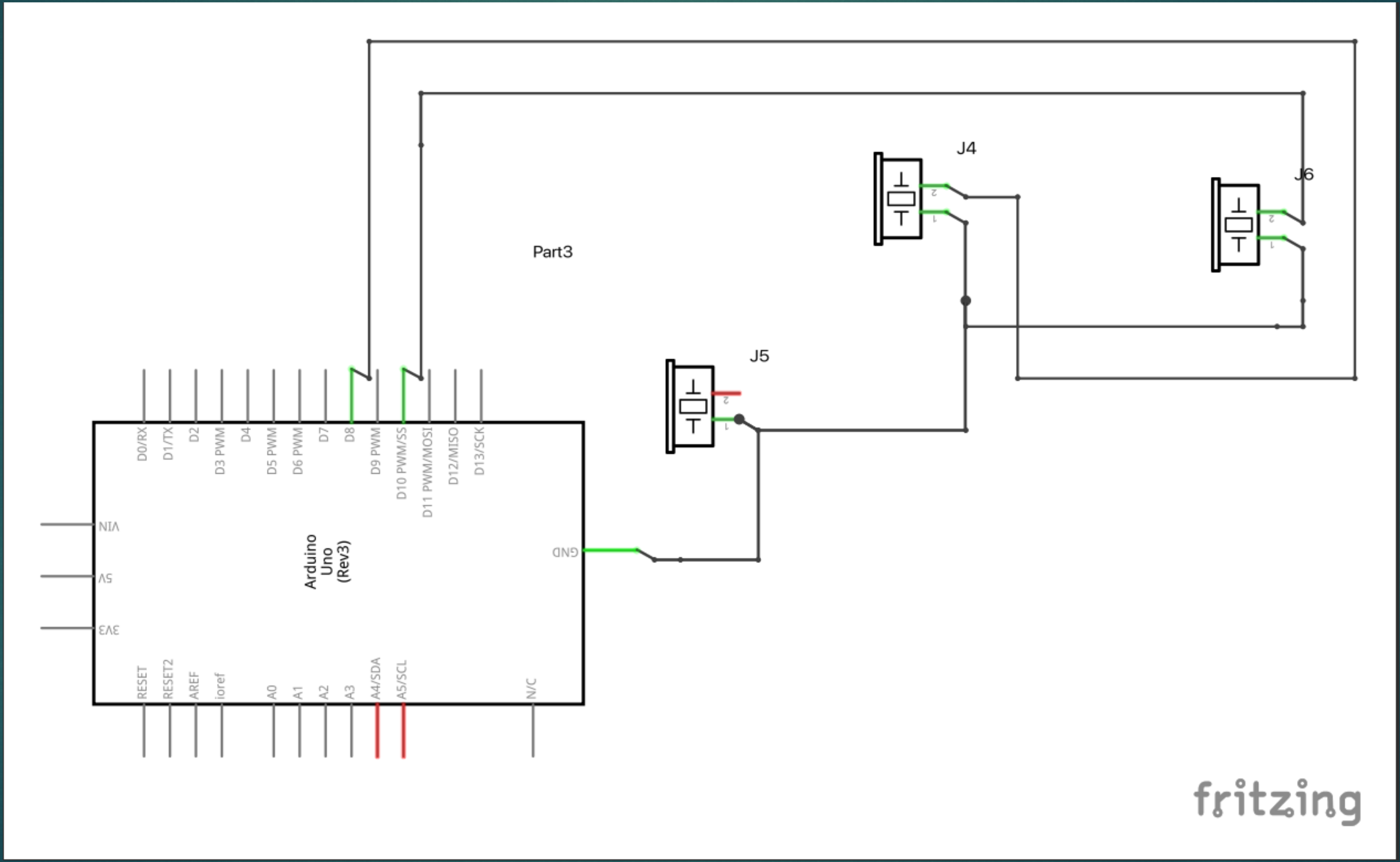




```
void receiveEvent() {
  while (Wire.available()) {
    char command = Wire.read();
    Serial.print("Received command: ");
    Serial.println(command);

    if (command == 5) {
      // Error code 5: Temperature out of range in Basement
      louderBeep(buzzerBasement, 2000); // One louder beep for Basement
    } else if (command == 6) {
      // Error code 6: Temperature out of range in Floor 2
      louderBeep(buzzerFloor2, 2000); // One louder beep for Floor 2
    } else if (command == 7) {
      // Error code 7: Temperature out of range in Floor 1
      louderBeep(buzzerFloor1, 2000); // One louder beep for Floor 1
    } else if (command == 8) {
      // Error code 8: No motion detected for 5 seconds on a specific floor
      int motionDetectorFloor = Wire.read(); // Read the floor number from the master
      Serial.print("No motion detected for 5 seconds on Floor ");
      Serial.println(motionDetectorFloor);
```

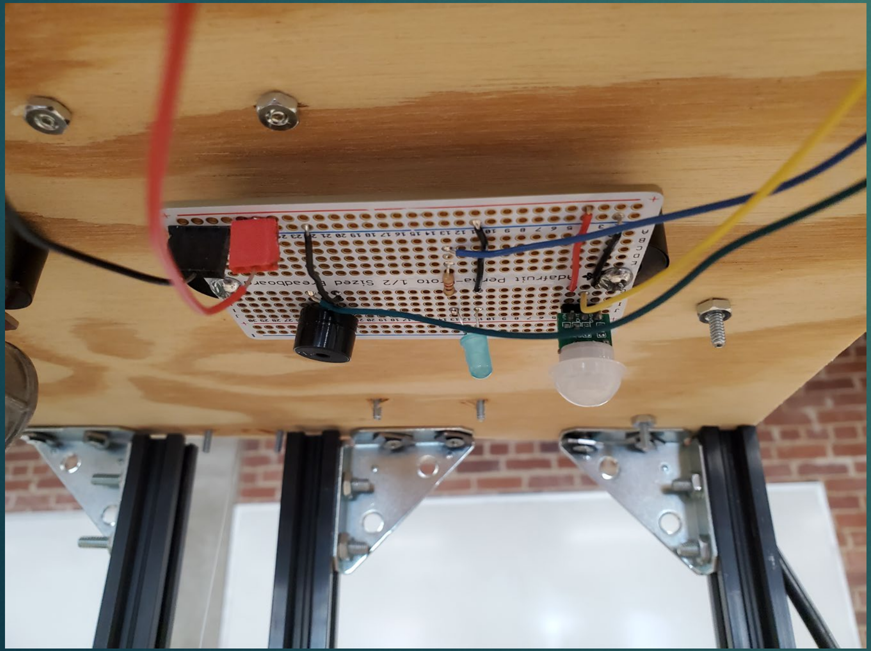


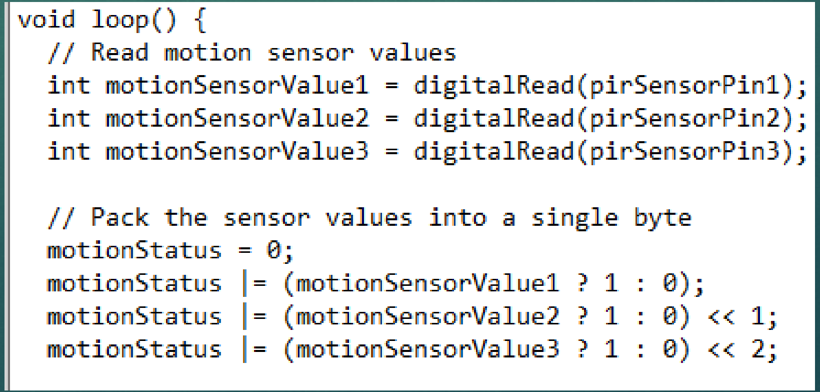
# 13. BUZZER MODULE



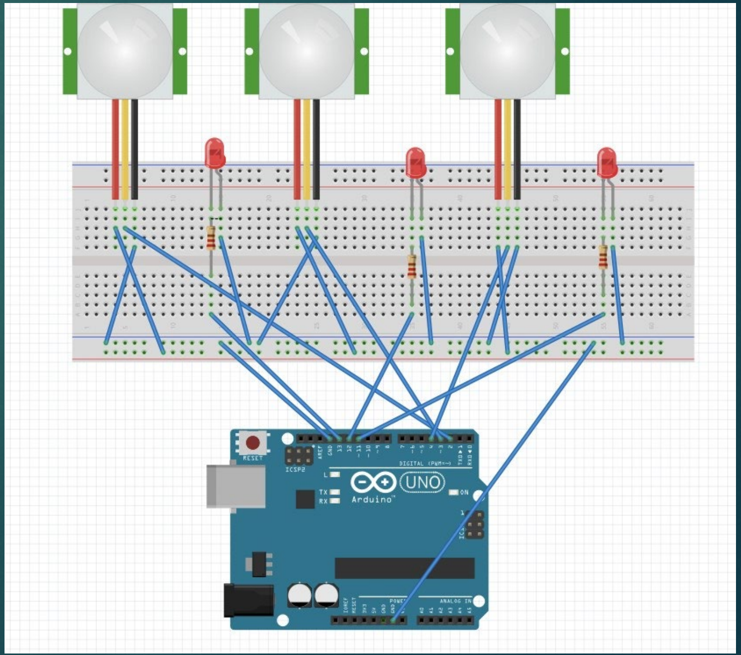
# 14. Motion Sensor

- Detects if there is motion on each floor and turns lights on when motion is detected
- If on After Hours, alarm will beep if motion is detected on that floor
- If on Public Hours, alarm will beep if no motion is detected



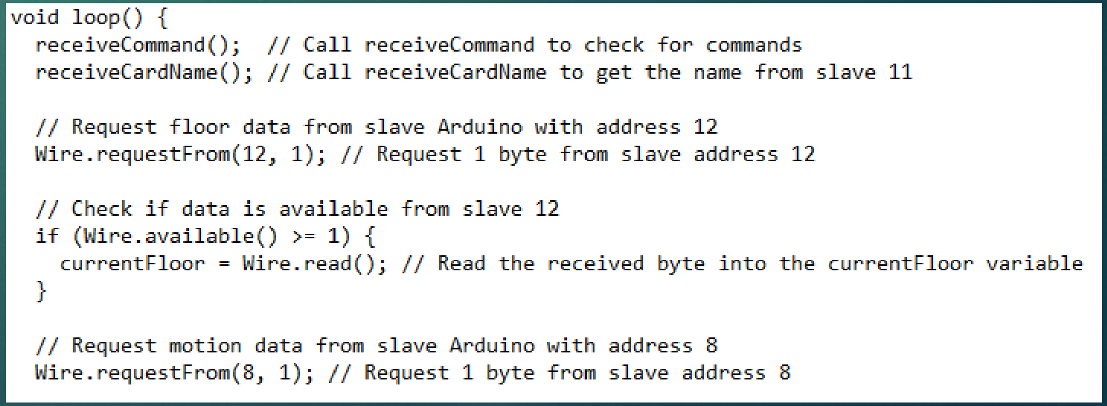


```
void loop() {
  // Read motion sensor values
  int motionSensorValue1 = digitalRead(pirSensorPin1);
  int motionSensorValue2 = digitalRead(pirSensorPin2);
  int motionSensorValue3 = digitalRead(pirSensorPin3);

  // Pack the sensor values into a single byte
  motionStatus = 0;
  motionStatus |= (motionSensorValue1 ? 1 : 0);
  motionStatus |= (motionSensorValue2 ? 1 : 0) << 1;
  motionStatus |= (motionSensorValue3 ? 1 : 0) << 2;
```



# 15. Master Arduino

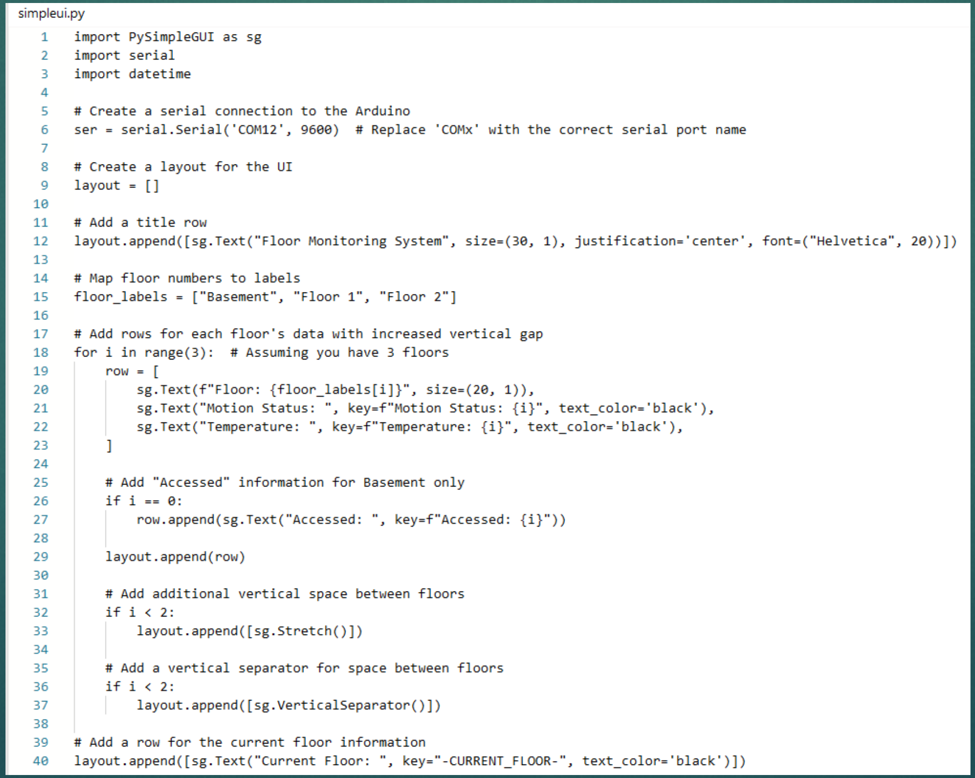
- Reads information from each system and passes it to the UI
- Utilizes Inter-Integrated Circuit (I2C) to communicate to 5 other Arduinos
- Monitors the state of each system
  - Reads the temperature of each floor
  - Reads the motion status of the elevator, which floor it is on, and if blockage is detected
  - Reads if there is motion on each floor



```
void loop() {
  receiveCommand();  // Call receiveCommand to check for commands
  receiveCardName(); // Call receiveCardName to get the name from slave 11

  // Request floor data from slave Arduino with address 12
  Wire.requestFrom(12, 1); // Request 1 byte from slave address 12

  // Check if data is available from slave 12
  if (Wire.available() >= 1) {
    currentFloor = Wire.read(); // Read the received byte into the currentFloor variable
  }

  // Request motion data from slave Arduino with address 8
  Wire.requestFrom(8, 1); // Request 1 byte from slave address 8
```

# 16. UI

- Written in Python
- Displays information passed to it from the master



simpleui.py

```python
import PySimpleGUI as sg
import serial
import datetime

# Create a serial connection to the Arduino
ser = serial.Serial('COM12', 9600)  # Replace 'COMx' with the correct serial port name

# Create a layout for the UI
layout = []

# Add a title row
layout.append([sg.Text("Floor Monitoring System", size=(30, 1), justification='center', font=("Helvetica", 20))])

# Map floor numbers to labels
floor_labels = ["Basement", "Floor 1", "Floor 2"]

# Add rows for each floor's data with increased vertical gap
for i in range(3):  # Assuming you have 3 floors
    row = [
        sg.Text(f"Floor: {floor_labels[i]}", size=(20, 1)),
        sg.Text("Motion Status: ", key=f"Motion Status: {i}", text_color='black'),
        sg.Text("Temperature: ", key=f"Temperature: {i}", text_color='black'),
    ]

    # Add "Accessed" information for Basement only
    if i == 0:
        row.append(sg.Text("Accessed: ", key=f"Accessed: {i}"))

    layout.append(row)

    # Add additional vertical space between floors
    if i < 2:
        layout.append([sg.Stretch()])

    # Add a vertical separator for space between floors
    if i < 2:
        layout.append([sg.VerticalSeparator()])

# Add a row for the current floor information
layout.append([sg.Text("Current Floor: ", key="-CURRENT_FLOOR-", text_color='black')])
```

## 17. Technical Difficulties

- Elevator sensors malfunctioning (taking the input of the buttons)
  - Resolved by placing them on separate boards
- Limited number of pins on each Arduino
- Bolts protruding the bottom of the building
- Aluminum rods too close together, prevents elevator from reaching 3rd floor

## 18. What Could Have Been Done Differently?

- Instead of using 6 separate Arduinos, one or two Arduino Megas could be used instead
- Raspberry Pis could be used in place of Arduinos

## Questions