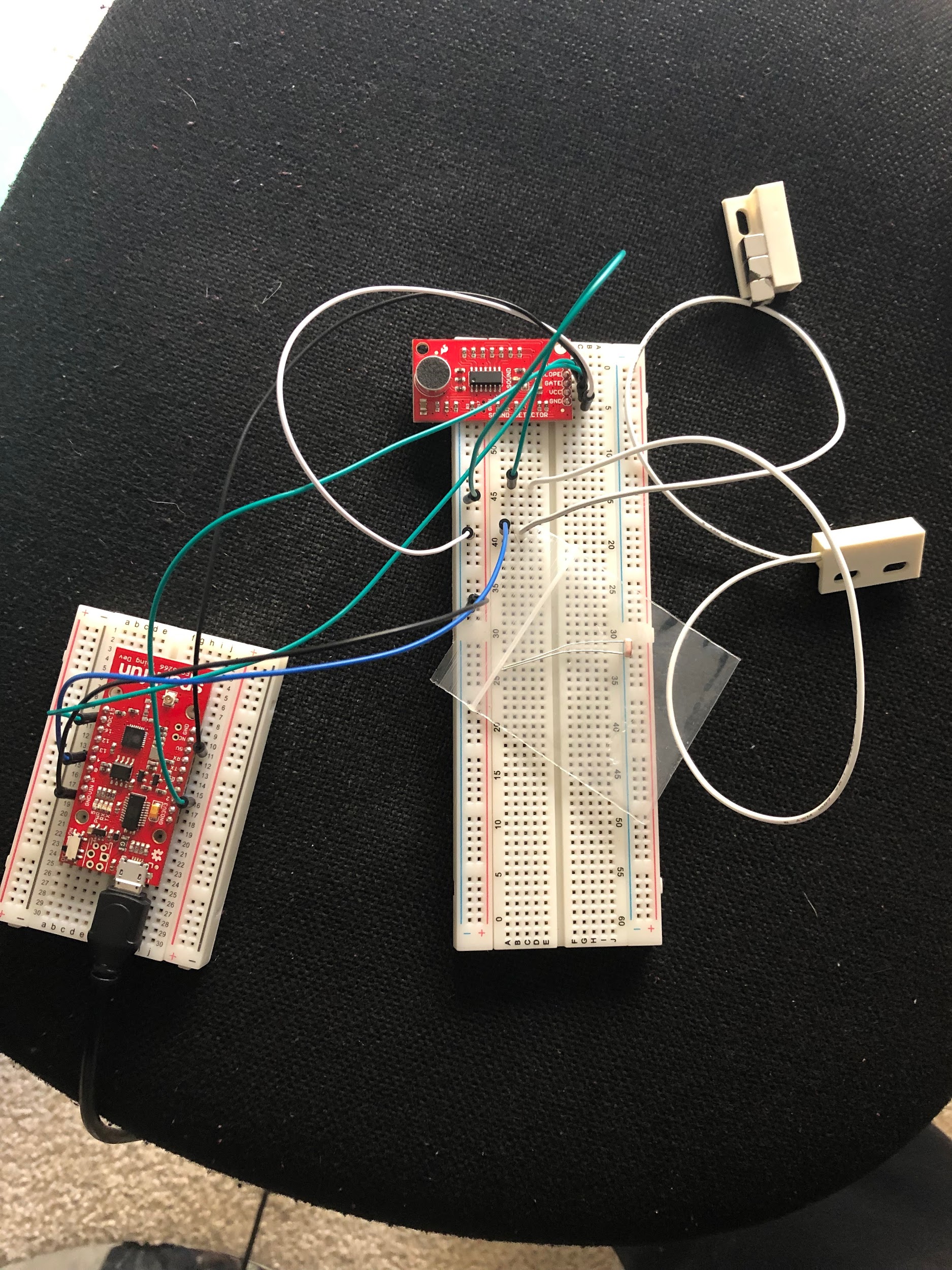
**Karan Khosla, 91725095**

**Xiao Zhang, 78369457**

**LOCK YOUR DOOR PLEASE!**

**FINAL REPORT**

Picture of project



Overview

The goal of our project is to complete a smart-space device that can detect if a door is locked or not and remind user to lock it by sending emails from the cloud.

Hardwares and Devices

1. Sparkfun ESP8266 WiFi Board x1
2. Sound sensor x1
3. Magnetic Door Switch Set

Fulfillments

1. Using sensors to determine whether a user has closed a door without locking it
2. Send the data to Azure Iothub, using Azure Function Apps to trigger a function that sends a email reminder to the user.

**Challenges**

1. Distinguish the sound

When we started to work on the project, we attempted to use a sound sensor to memorize the amplitude of different sounds and distinguish them in the code, so we can use sounds to check whether a door is closed and locked. But then we realized that the sound detector can only distinguish the amplitude of sound, which is not enough for us. So we are using an extra set of Magnetic Door Switch Set, which just tells us whether a door is closed. With the help of the Switch set, we are able to use the sound sensor to determine whether a door is locked after closing.

2. The sound detector malfunctioned one day before the demo so we had to change how our logic works. We are now using the magnetic door switch set to simulate the door closing and locking as well.

3. The app we are using to send the data to the cloud was not able to sustain a stable wifi connection while we were sending our data. So, we had to modify our logic a little to make it work.

**Work done(earlier)**

**Phase 1: Calibration: This phase lasts for 15 seconds.**

We are calibrating the locking sound using our sound detector. We are using a calibration technique to distinguish between two sounds. We use the **ratio of max\_amplitude to the difference of times of the sound wave to get the ratio** as described in the picture. We assume that other sounds have a different ratio and hence might be ignored by our logic.

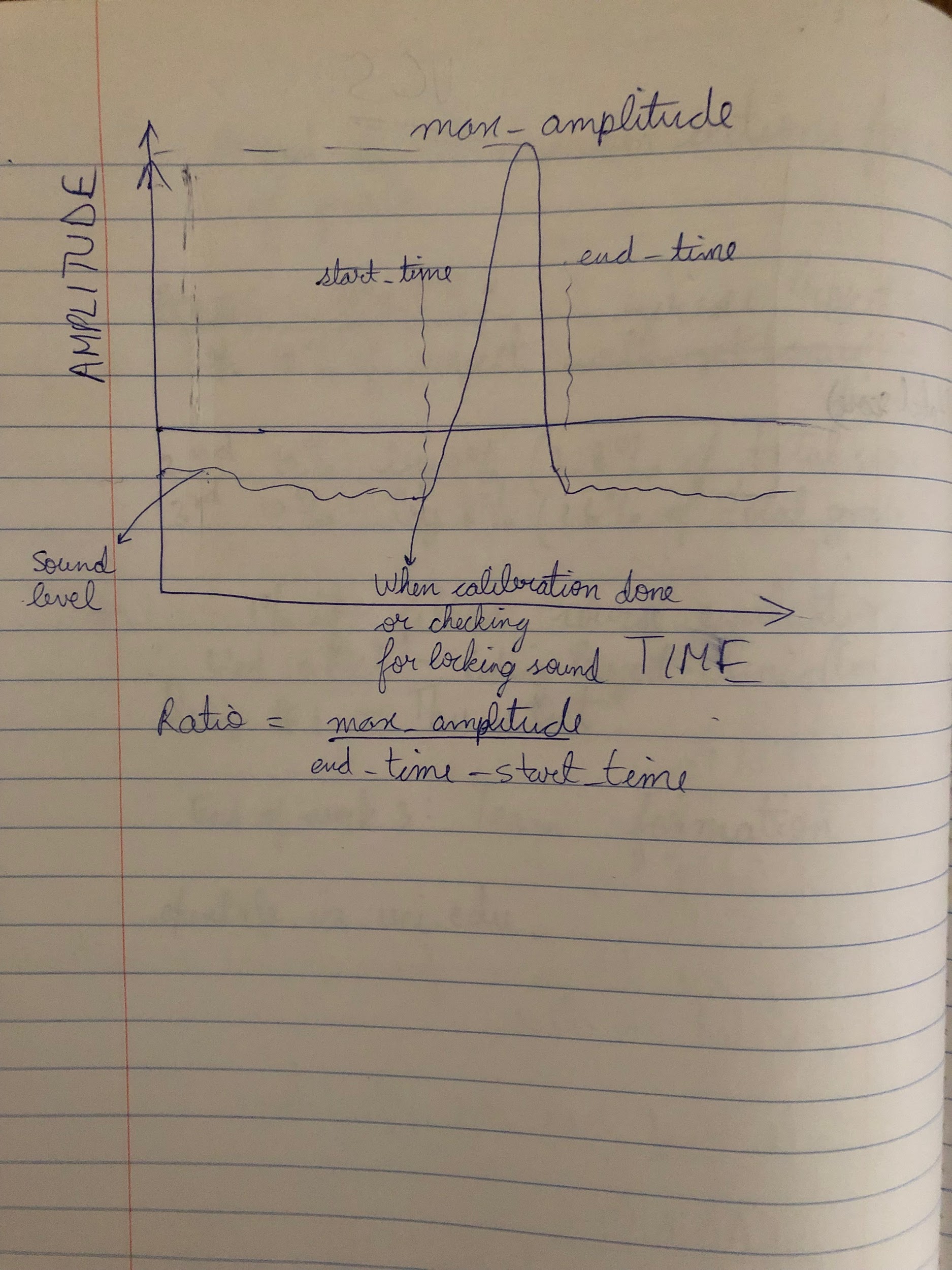
**Calibration code:**

soundValue = analogRead(PIN\_ANALOG\_IN);  
 getParams();  
 if(end\_t > start\_t && timer\_f == false)  
 {  
 updateRatio();  
 resetParams();  
 }

-getParams gets the max\_amplitude and the two times used to get the ratio.

-updateRatio updates the ratio during calibration and ignores any anomaly data that the sensor gets. Thus, after 15 seconds, we take the average of all the ratios obtained and set it as the ratio of the locking sound.

void getParams()  
{  
 if(soundValue > back && timer\_f == false)  
 {  
 timer\_f = true;  
 start\_t = millis();  
 }  
 if(timer\_f == true && soundValue < back)  
 {  
   
 end\_t = millis();  
 timer\_f = false;  
 //frequency = max\_amp/(end\_t-start\_t);  
 }  
 if(timer\_f == true)  
 {  
 max\_amp = max(soundValue, max\_amp);  
 }  
 delay(10);  
We get the final value of the ratio in the variable **ratio\_t.**

****

**Phase 2: Check for lock after door closed.**

After calibration, we check for if door is locked after it closed. If it is within a set time frame, then no notification is sent. Otherwise, a notification is sent to the cloud, which sends the user an email.

We use the magnetic door switch set to determine whether the door is locked or not. If it comes close, then the circuit is complete, which will happen when a door closes. We set a timer when the door closes to check whether the door is locked within that time frame.

**Code for Phase 2:**

Code to check if door is closed or not:

state = digitalRead(door\_sensor);  
 if(state == LOW && !door\_closed) // door closed  
 {  
 Serial.println("Door closed");  
 notif\_t = millis();  
 door\_closed = true;  
 delay(4000);  
 }

If we get a sound with ratio ranging from (ratio\_t - 3, ratio\_t +3), then we state that the door is locked and enable the door\_locked flag to disable sending notifications to the cloud.

getParams();

float ratio = (max\_amplitude)/(time\_end - time\_start);

if(ratio > ratio\_t -3 && ratio < ratio\_t + 3)

{

door\_locked = true;

}

Else we then check if time after door closes is greater than 15 secs and no door locked yet

if(!door\_locked && millis()-notif\_t > 15000)

{

send\_f = true; // send data flag

Notif\_t = millis(); // reset the last mail sent timer.

}

//If send\_f is true, send the data to the cloud

if(send\_f && !messagePending && messageSending)  
 {  
 Serial.println("Sending data to the cloud");  
 char messagePayload[MESSAGE\_MAX\_LEN];  
 readMessage(messageCount, messagePayload);  
 sendMessage(iotHubClientHandle, messagePayload);  
 messageCount++;  
 send\_f = false; // disable send\_f flag so device doesn’t keep on sending notif to the cloud  
 delay(interval);  
 }  
 IoTHubClient\_LL\_DoWork(iotHubClientHandle);

**Phase 3: Sending e mail to the user.**

We are using a IOTHub trigger based Azure function to send the email to the user. The Azure function listens on the IOTHub and whenever it receives a message, it uses SMTPClient to send a mail.

Code for sending email in Azure function

string fromEmail = ""; // email of the user who sends the mail  
 string toEmail = // send email to whom  
 int smtpPort = 587;  
 bool smtpEnableSsl = true;  
 string smtpHost = //your smtp host  
 string smtpUser = // your smtp user  
 string smtpPass = // your smtp password  
 string subject = "Door unlocked!";  
 string message = "Your door is unlocked. Remember to lock it please";

//Configuring the mail delivery.  
 MailMessage mail = new MailMessage(fromEmail, toEmail);  
 SmtpClient client = new SmtpClient();  
 client.Port = smtpPort;  
 client.EnableSsl = smtpEnableSsl;  
 client.DeliveryMethod = SmtpDeliveryMethod.Network;  
 client.UseDefaultCredentials = false;  
 client.Host = smtpHost;  
 client.Credentials = new System.Net.NetworkCredential(smtpUser, smtpPass);  
 mail.Subject = message;

//Check if mail is sent or not?

mail.Body = message;  
 try {  
 client.Send(mail);  
 log.LogInformation("Email sent");}

Word done (working)

Unfortunately, last night the sound detector was spewing garbage data and so my logic stopped working. So, I used the door switch set to also detect if the door is locked or not. The door closing check is the same but the door locking is now checked with the door switch set again as the switch set can also detect the proximity of the lock near the wall and say that the door is locked or not.

Otherwise, the Phase 3 is the same.