SIT123 Data Capture Technologies

Final Project

Hannah Smith 216019732

Introduction

This project aims to create a smart home solution for carers of elderly people or people with dementia who are living independently. It was inspired by my grandmother, who often forgets to eat meals and cannot reliably report about how much she has eaten. This device monitors use of the kitchen and sends a daily report by wi-fi to the carer's email address. It also tweets whenever motion is detected in the kitchen. The carer can use this information to monitor whether the elderly person has used the kitchen to make food, and then make decisions accordingly. In order to create this device, I hypothesised that food preparation in the kitchen space could be accurately measured and reported using this device, which uses motion detection and a temperature/humidity sensor. The research questions needed to test this hypothesis are:

1. How many readings of "active" by the PIR motion sensor indicate food preparation?
2. What temperature and humidity patterns indicate food preparation?
3. How can this information be communicated accurately through ThingSpeak to the carer?

Approach

The project timeline is show below.

	Week 7	Week 8	Week 9	Week 10
Consult with tutor	🟧			
System design	🟥	🟥		
Hardware design and testing		🟨	🟨	
Software design and testing		🟩	🟩	🟩
Writing report			🟦	🟦

Hardware Development

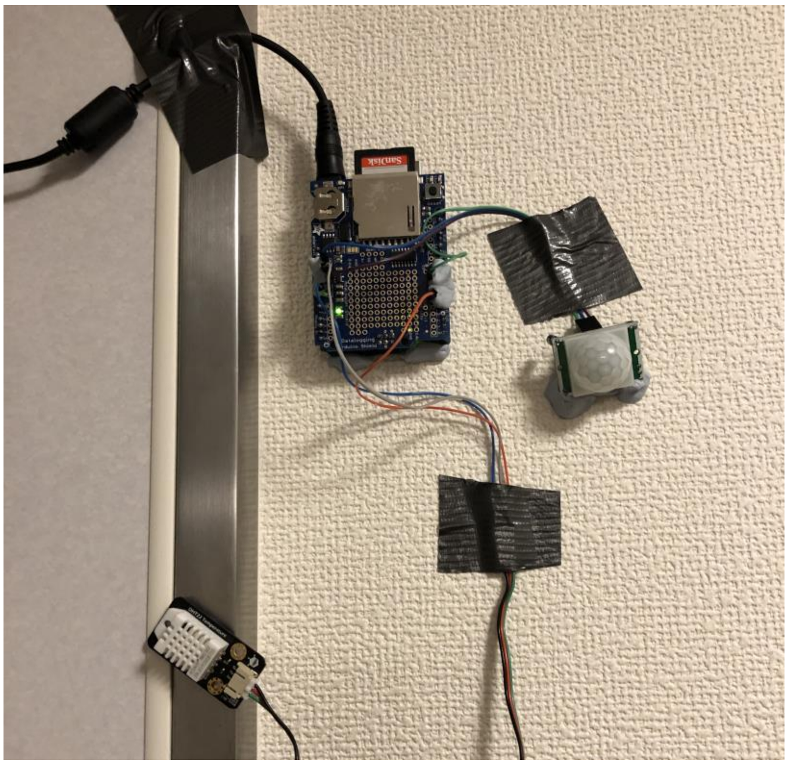


Figure 1

The initial hardware setup is shown in Figure 1. This was used to gather data from the sensors in order to determine thresholds. The data was read from the sensors and stored as a CSV file on the SD card using the attached datalogging shield. However, the datalogging shield did not have heads and it was difficult to get a reliable connection with the sensors.

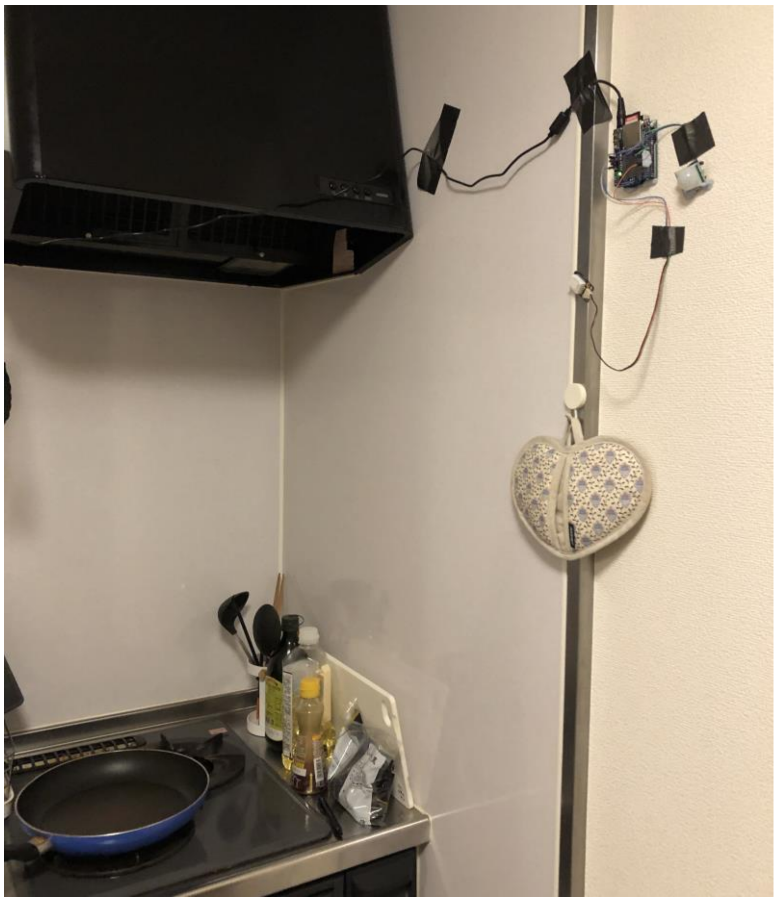


Figure 2

Figure 2 shows the location of the hardware in the kitchen. The temperature and humidity sensor are placed close to the stovetop in order to detect when it is used. The PIR motion sensor is directed towards the rest of the kitchen in order to detect movement near the stove and sink.

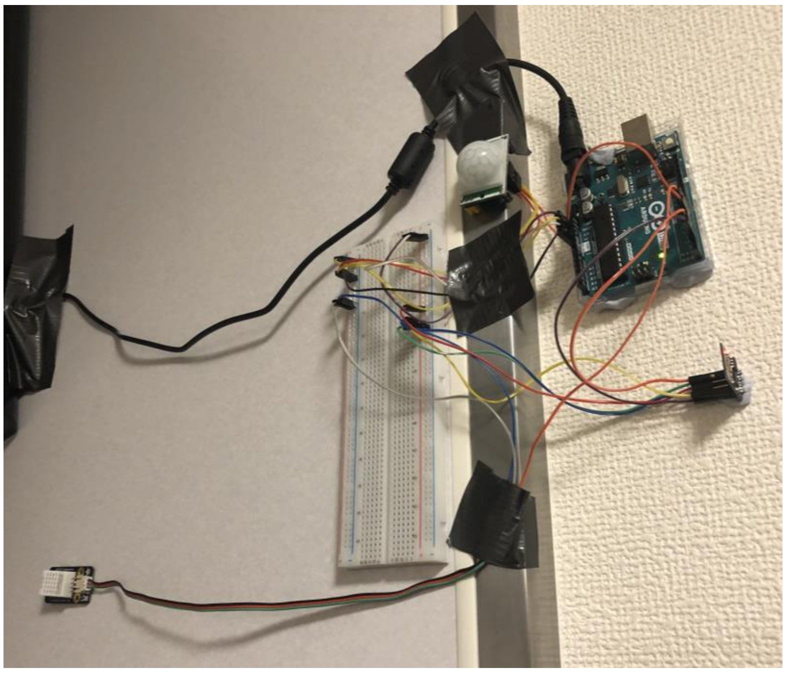


Figure 3

Figure 3 shows the second version of the setup. Initially, it was planned that the Arduino would have a datalogger attached which would be used to store, read and analyse the data. However, given the limited dynamic memory capacity it was determined that using an online service would be a simpler alternative. The datalogging shield was removed, and an ESP-01 Wi-Fi module. This was used to upload the sensor data to ThingSpeak. Adding in the Wi-Fi module necessitated the use of a breadboard.

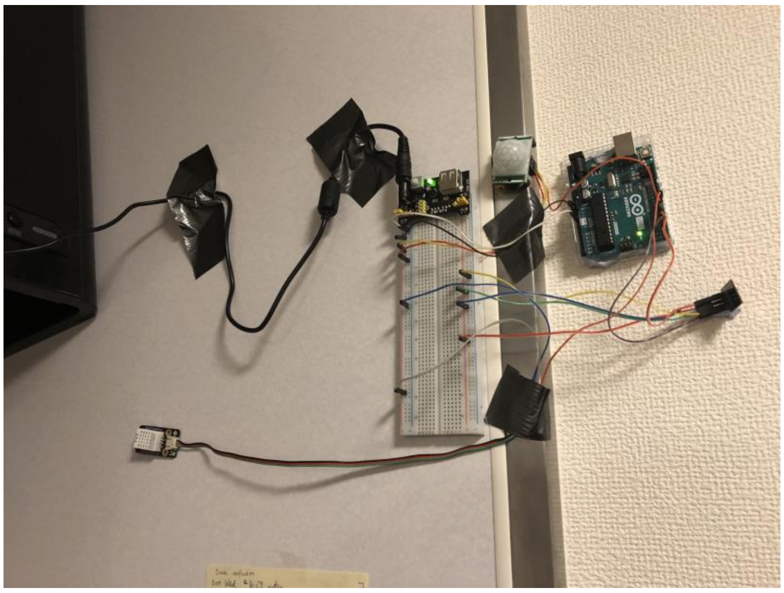


Figure 4

Figure 4 shows the final version of the device. Previously, the setup had problems with power. Due to the insufficient power, sometimes the sensors would stop retrieving data. A power supply was used in order to ensure regular power supply, and a voltmeter was used to select an appropriate supply from the AC adapter (which has variable voltage).

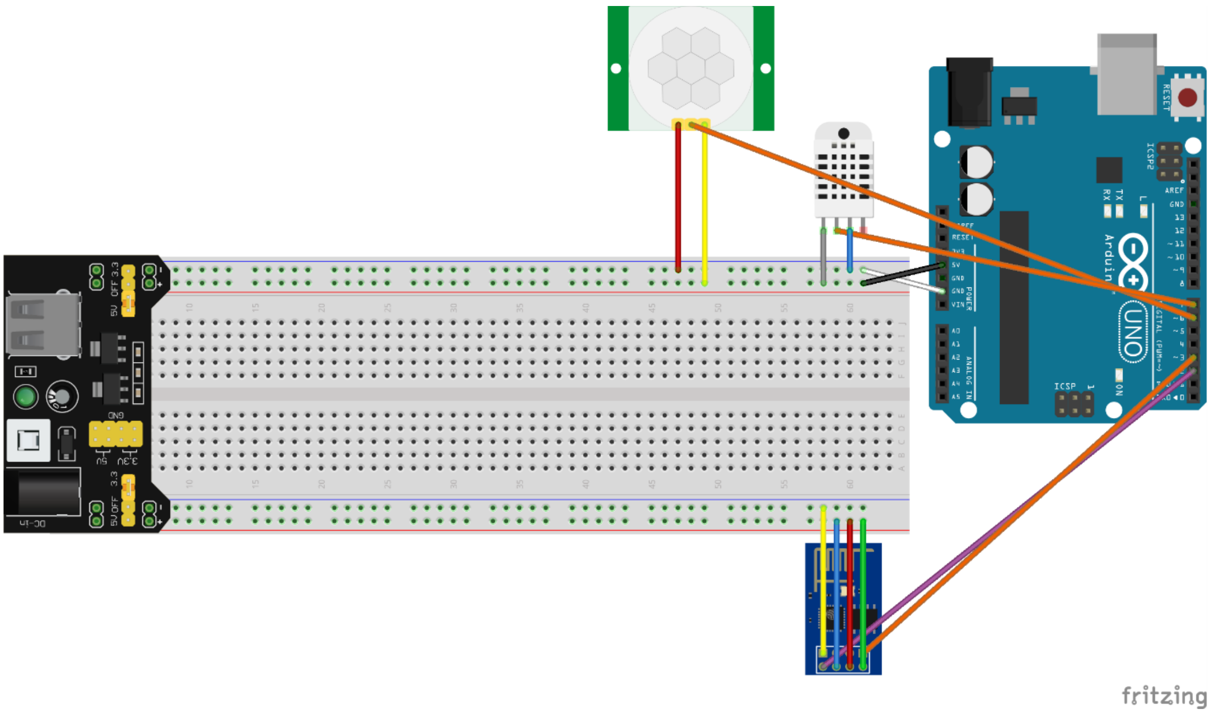


Figure 5

Figure 5 shows a circuit diagram of the final hardware setup. A voltmeter was used to test the 3V and 5V rail and determine if the setup had sufficient power.

Software Development

Arduino software

For each of the hardware iterations, a different Arduino program was written. The various components are described below in the order the were written. Some were removed as the project evolved, indicated in italics.

1. Read data from the PIR motion sensor and the temperature/humidity sensor simultaneously
2. *Logging the data as a CSV file that could be retrieved and analysed*
3. *Reading data from the CSV and analysing it in arrays*
4. *Sending emails from the Arduino using an SMTP email service*
5. Sending sensor data to ThingSpeak at 20 second intervals.

```
/* This program was created for my final project for SIT123
    Author: Hannah Smith
    Created: September, 18 2020
    Adapted from:
    Code for connecting to ThingsSpeak using the ESP-01 from
    https://create.arduino.cc/projecthub/neverofftheinternet */
/* ---------- Libraries ---------- */
#include <DHT.h>
#include <DHT_U.h>
#include <Adafruit_Sensor.h>
#include <SoftwareSerial.h>
```

```
/* ---------- Constants ---------- */
#define DHTPIN 7 // temperature humidity sensor
#define PIRPIN 6 // motion sensor
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
float motionReadings = 0;
int i = 0;
long startTime = 0;
long waitTime = 0;
SoftwareSerial ESP8266(2, 3); // Rx,  Tx
boolean relay1_st = false;
boolean relay2_st = false;
unsigned char check_connection = 0;
unsigned char times_check = 0;
boolean error;

/* ---------- Variables ---------- */
float h, t; //the initial values read for the humidity and temperature
int state; //the state of the motion sensor, recorded as 1 for active and 0
for inactive

void setup() {

  Serial.begin(9600);
  ESP8266.begin(9600);

  startTime = millis();

  pinMode(13, OUTPUT);

  //LED flashes to indicate the program is starting
  for (int i = 0; i < 5; i++) {
    digitalWrite(13, HIGH);
    delay(500);
    digitalWrite(13, LOW);
    delay(500);
  }

  dht.begin(); //Setup for temperature sensing
  pinMode(PIRPIN, INPUT); //Setup for PIR

  ESP8266.println("AT+RST");
  delay(2000);
  Serial.println("Connecting to Wifi");
  while (check_connection == 0)
  {
    Serial.print(".");
    ESP8266.print("AT+CWJAP=\"SSID\",\"PASSWORD\"\r\n");
    ESP8266.setTimeout(5000);
    if (ESP8266.find("WIFI CONNECTED\r\n") == 1)
    {
      Serial.println("WIFI CONNECTED");
      break;
    }
    times_check++;
    if (times_check > 3)
    {
      times_check = 0;
      Serial.println("Trying to Reconnect..");
    }
  }
}

void loop() {
  for (i = 0; i < 20; i++)
  {
    readMotionSensors();
    motionReadings += state;
    delay(1000);
  }
  motionReadings /= 20;
  waitTime = millis() - startTime;
  if (waitTime > 20000) {
    readTempSensors();
    writeThingSpeak();
    startTime = millis();
  };
  i = 0;
}

//Functions
void readTempSensors(void)
{
  h = dht.readHumidity(); //Read the data and store it as floats
  t = dht.readTemperature();
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" Humidity: ");
  Serial.println(h);
}

void readMotionSensors(void)
{
  if (digitalRead(PIRPIN) == HIGH) {
    state = 1; //an active reading is recorded as 1
    Serial.println("Motion detected!");
  }
  else {
    state = 0; //inactive
    Serial.println("No motion detected.");
  }
}

void writeThingSpeak(void)
{
  startThingSpeakCmd();
  String getStr = "GET /update?api_key=";
  getStr += "RC1FB3JL6A8FGZSN"; //API key for the ThingSpeak project
  getStr += "&field1=";
  getStr += String(t);
  getStr += "&field2=";
  getStr += String(h);
  getStr += "&field3=";
  getStr += String(motionReadings);
  getStr += "\r\n\r\n";
  GetThingspeakcmd(getStr);
  Serial.println(getStr);
}

void startThingSpeakCmd(void)
{
  ESP8266.flush();
  String cmd = "AT+CIPSTART=\"TCP\",\"";
  cmd += "184.106.153.149"; // api.thingspeak.com IP address
  cmd += "\",80";
  ESP8266.println(cmd);
  Serial.print("Start Commands: ");
  Serial.println(cmd);

  if (ESP8266.find("Error"))
  {
    Serial.println("AT+CIPSTART error");
    return;
  }
}

String GetThingspeakcmd(String getStr)
{
  String cmd = "AT+CIPSEND=";
  cmd += String(getStr.length());
  ESP8266.println(cmd);
  Serial.println(cmd);

  if (ESP8266.find(">"))
  {
    ESP8266.print(getStr);
    Serial.println (getStr);
```

```
      delay(500);
      String messageBody = "";
      while (ESP8266.available())
      {
        String line = ESP8266.readStringUntil('\n');
        if (line.length() == 1)
        {
          messageBody = ESP8266.readStringUntil('\n');
        }
      }
      Serial.print("MessageBody received: ");
      Serial.println(messageBody);
      return messageBody;
    }
    else
    {
      ESP8266.println("AT+CIPCLOSE");
      Serial.println("AT+CIPCLOSE");
    }
  }
```

ThingSpeak Code

In order to accurately report on the use of the kitchen, several apps had to be built using ThingSpeak's MATLAB capabilities.

1. First, the sensor is sent from the Arduino to ThingSpeak every 20 seconds (free accounts only allow updates every 15 seconds – the extra five seconds is to allow for a margin of error in the timing.
2. The temperature and humidity data is read every 20 seconds.
3. The motion data is read every second, and then an average is created for each 20 second block. Figure 6 shows an example of the data as it is received by ThingSpeak.

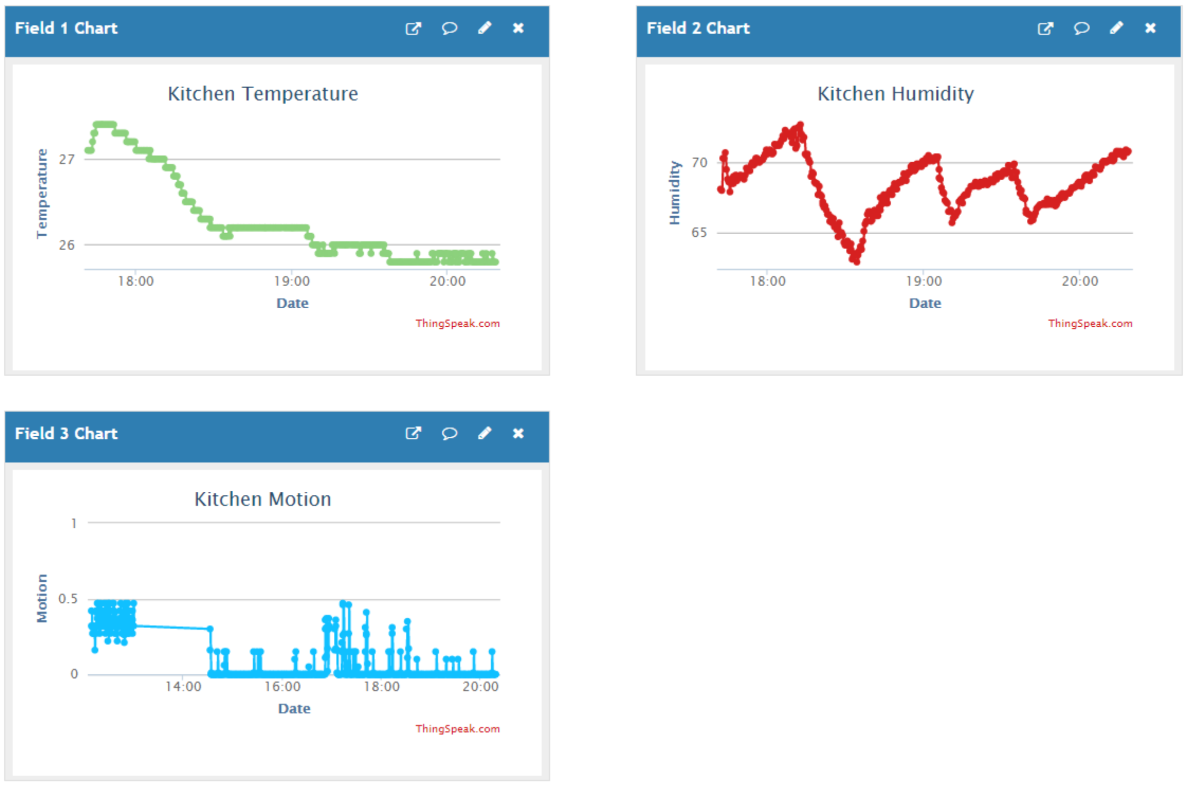
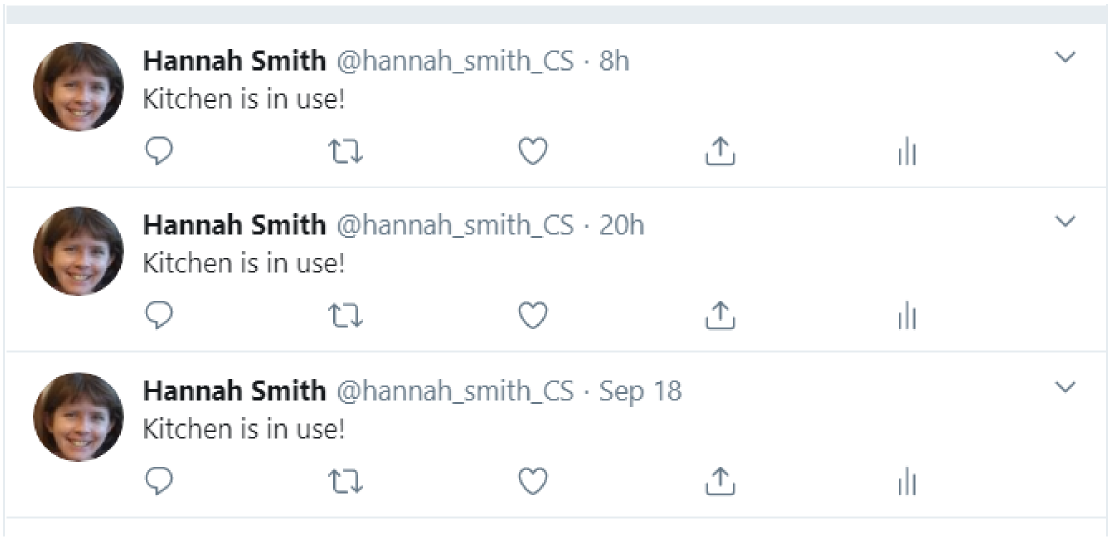


Figure 6

The Kitchen Motion data is measure by a ThingTweet app. This app checks every 10 minutes for a data point greater than 0.3 and then sends a Tweet to a designated Twitter account (see Figure 7). This is designed so the customer can use Twitter to check on their old person they are caring for, or perhaps even share the information with family.


Hannah Smith @hannah_smith_CS · 8h
Kitchen is in use!

Hannah Smith @hannah_smith_CS · 20h
Kitchen is in use!

Hannah Smith @hannah_smith_CS · Sep 18
Kitchen is in use!

Figure 7

Since the motion data is read every second, it is important not to send the tweets too often. To avoid overreporting the motion data, the data was collected into an average reading for each minute. Since ThingSpeak free accounts only allow for three data sources per channel, I wrote code to process an average for each minute and write it to another channel. Due to the limitation on update frequency, this second channel is updated every 10 minutes using the following MATLAB code.

```
% This code takes readings from the motion sensor every minute and
% averages them, to be used for detecting kitchen use.

% Store the channel IDs
readChannelID = 1145985;
writeChannelID = 1147888;

% Provide the write API key.
writeKey = 'GIN15GMFJMR8VYZE';

% Reads the latest 30 readings from the motion data. The data is read
% every 20 seconds so this will constitute 10 minutes. This script
% is triggered to run every 10 minutes
[motion, time] = thingSpeakRead(readChannelID,'Fields',3,'Numpoints',30);

% Divides the data into arrays for each minute
motion1 = motion(1:6);
motion2 = motion(7:12);
motion3 = motion(13:18);
motion4 = motion(19:24);
motion5 = motion(25:30);

%stripping any NaN values that will break the code
motion1 = motion1';
motion1 = motion1(~isnan(motion1))';
motion2 = motion2';
```

```
motion2 = motion2(~isnan(motion2))';
motion3 = motion3';
motion3 = motion3(~isnan(motion3))';
motion4 = motion4';
motion4 = motion4(~isnan(motion4))';
motion5 = motion5';
motion6 = motion5(~isnan(motion5))';

% Calculate the average motion for each minute
avgMotion1 = mean(motion1);
avgMotion2 = mean(motion2);
avgMotion3 = mean(motion3);
avgMotion4 = mean(motion4);
avgMotion5 = mean(motion5);

%Creating arrays with the average motion for each minute and a timestamp
a = [avgMotion1; avgMotion2; avgMotion3; avgMotion4; avgMotion5];
t = time([1,7,13,19,25]);

% Write the average to the channel
thingSpeakWrite(writeChannelID,'Values',a,'writekey', writeKey,
'TimeStamps',t);
```

Finally, an app was built in MATLAB to process the data and report it to the customer in a daily email sent at 8pm. This time would need to be adjusted if the user usually has dinner at a later time. In order to not overreport, each kitchen visit detected by the motion sensor triggers a state of “motionDetected” which remains in place until a 0 reading is detected indicating the person left the kitchen. Similarly, a spike in temperature creates a state of “spikeDetected” which remains until the temperature drops.

```
% This code reads the data from the kitchen sensors. At 8pm every day it
% reports on kitchen visits and whether there were any spikes in the
% temperature.

% Store the channel ID
sensorDataChannelID = 1145985;
motionDataChannelID = 1147888;

% Provide the ThingSpeak alerts API key.
alertApiKey = 'TAKT7Q6Q12RCP68CCA96C';

% Set the address for the HTTTP call
alertUrl="https://api.thingspeak.com/alerts/send";

% webwrite uses weboptions to add required headers.  Alerts needs a
% ThingSpeak-Alerts-API-Key header.
```

```
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key",
alertApiKey ]);

% Set the email subject.
alertSubject = sprintf("Daily Kitchen Report");

% Read the recent data.
[temp, time1] = thingSpeakRead(sensorDataChannelID,'Fields', 1,'Numdays',1);
[motion, time2] = thingSpeakRead(motionDataChannelID,'Fields',
1,'Numdays',1);

% Create an array to store the information about when the temperature spikes
indicating the kitchen is being used.
spikeTimes = [];

% Loops through the readings of temperature and looks for a jump in 3
degrees over five minutes
spikeDetected = false; %This is set as false until we detect a jump in temp
for i = 1:1:(length(temp)-15)
    baseTemp = temp(i); %Baseline to compare it with the readings 5 minutes
later
    counter = 0;
    sum = 0;
    if spikeDetected == false
        if ~(isnan(temp(i+13)))
            sum = sum + temp(i+13);
            counter = counter + 1;
        end
        if ~(isnan(temp(i+14)))
            sum = sum + temp(i+14);
            counter = counter + 1;
        end
        if ~(isnan(temp(i+15)))
            sum = sum + temp(i+15);
            counter = counter + 1;
        end
        averageTemp = sum / counter;
        if (averageTemp - baseTemp) >= 3
            spikeTimes = cat(1,spikeTimes,time1(i));
            spikeDetected = true; %If we find a jump in temperature, set to
true
        end
    else %spikeDetected remains true until there is a drop of 1 degree
detected
        if ~(isnan(temp(i+13)))
            sum = sum + temp(i+13);
            counter = counter + 1;
        end
        if ~(isnan(temp(i+14)))
```

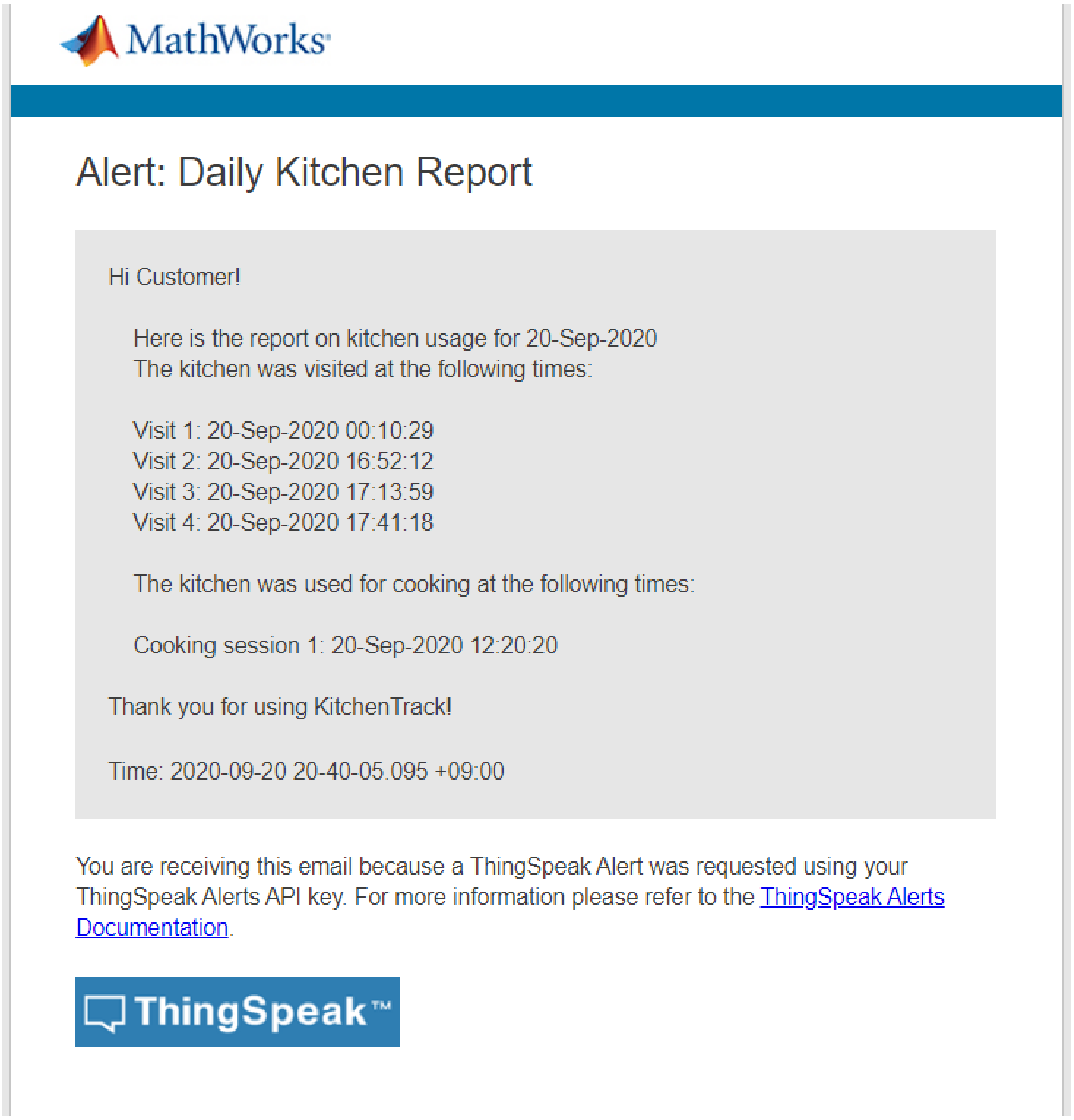
```
            sum = sum + temp(i+14);
            counter = counter + 1;
        end
        if ~(isnan(temp(i+15)))
            sum = sum + temp(i+15);
            counter = counter + 1;
        end
        averageTemp = sum / counter;
        if (baseTemp - averageTemp) >= 1
            spikeDetected = false;
        end
    end
end

% Create an array to check how often the kitchen was visited. This could
indicate shorter visits like to snack or wash up. It reads every 5 minutes
motionSpikes = [];
motionDetected = false;
for i = 1:1:length(motion)
    if motionDetected == false
        if motion(i) >= 0.2
            motionSpikes = cat(1, motionSpikes, time2(i));
            motionDetected = true;
        end
    else
        if motion(i) == 0
            motionDetected = false;
        end
    end
end

% Check to make sure the data was read correctly from the channel.
Date = today('datetime');
DateString = string(Date);
alertBody = 'Hi Customer!\n\n    Here is the report on kitchen usage for ';
alertBody = alertBody + DateString;
if isempty(spikeTimes) && isempty(motionSpikes)
    alertBody = alertBody +'\n    No data from the kitchen. Please check
your device';
end
if ~(isempty(motionSpikes))
    alertBody = alertBody + '\n    The kitchen was visited at the following
times: \n';
    for i = 1:length(motionSpikes)
        alertBody = alertBody + '\n    Visit ' + string(i) + ': ';
        alertBody = alertBody + string(motionSpikes(i));
    end
end
if isempty(spikeTimes)
```

```
    alertBody = alertBody + '\n    The kitchen was not used for cooking
today.';
else
    alertBody = alertBody + '\n\n    The kitchen was used for cooking at the
following times: \n';
    for i = 1: length(spikeTimes)
        alertBody = alertBody + '\n    Cooking session ' + string(i) + ': ';
        alertBody = alertBody + string(spikeTimes(i));
    end
    alertBody = alertBody + '\n\nThank you for using KitchenTrack!';
end
alertBody = compose(alertBody);
disp(alertBody);

 % Catch errors so the MATLAB code does not disable a TimeControl if it
fails
try
    webwrite(alertUrl , "body", alertBody, "subject", alertSubject,
options);
catch someException
    fprintf("Failed to send alert: %s\n", someException.message);
end
```



MathWorks

Alert: Daily Kitchen Report

Hi Customer!

Here is the report on kitchen usage for 20-Sep-2020
The kitchen was visited at the following times:

Visit 1: 20-Sep-2020 00:10:29
Visit 2: 20-Sep-2020 16:52:12
Visit 3: 20-Sep-2020 17:13:59
Visit 4: 20-Sep-2020 17:41:18

The kitchen was used for cooking at the following times:

Cooking session 1: 20-Sep-2020 12:20:20

Thank you for using KitchenTrack!

Time: 2020-09-20 20-40-05.095 +09:00

You are receiving this email because a ThingSpeak Alert was requested using your ThingSpeak Alerts API key. For more information please refer to the ThingSpeak Alerts Documentation.

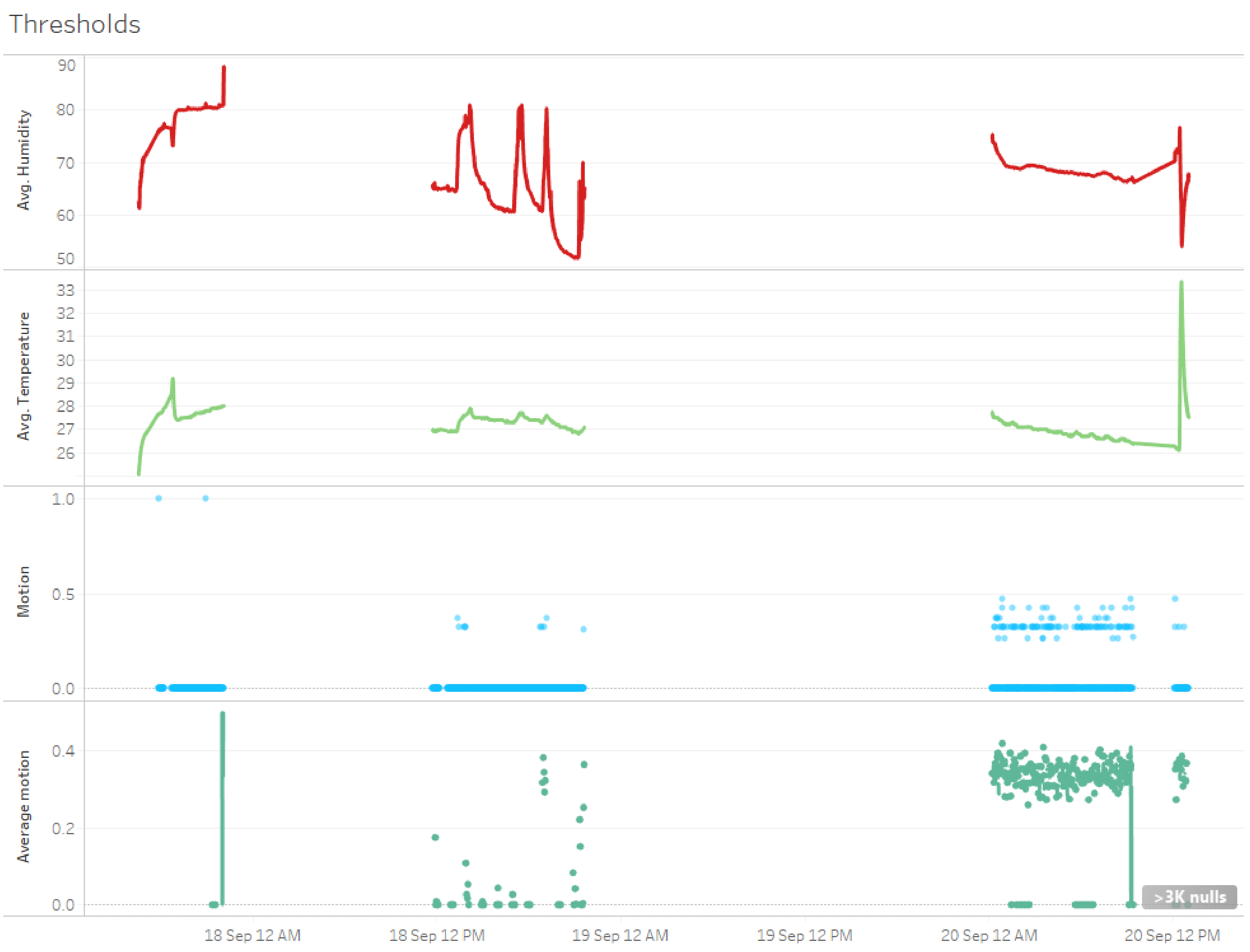
ThingSpeak™

Figure 8

Figure 8 shows an example of the message that the customer would receive every day.

Validating thresholds and sampling rate

The final stage of the project was to test the prototype system over an extended period. Previously, data collected had been unusable due to the hardware issues described above. In order to verify the data, kitchen visits and cooking were manually recorded using a pen and paper. Once the hardware was working with sufficient reliability the data was analysed in Tableau to determine an initial set of thresholds. Figure 9 shows this initial analysis. The gaps are null values due to the hardware issues. This initial analysis showed a spike in temperature, not humidity, when the stove was used. It also revealed that average motion per minute rather than the raw motion reading was a more accurate picture of the number of visits to the kitchen.



Humidity, temperature, motion and average motion used to determine thresholds [September 2020]

Figure 9

Based on this initial visual analysis, the following rules were created.

1. Humidity variation does not indicate cooking, but relates more to the environmental condition
2. Temperature spikes of 3 degrees or more indicate cooking on the stove
3. A subsequent drop of 1 degree will indicate the cooking has ceased and the cooking event is over
4. An average motion per minute of greater than 0.2 indicates the kitchen is being used
5. A subsequent drop to 0 indicates the kitchen visit has finished

Results

The results after another day of collection are displayed in Figure 10. This data was cleaned by removing time zone strings from the date stamps. Filtering revealed no anomalous numbers out of range. Figure 11 shows the ThingSpeak report from that day.

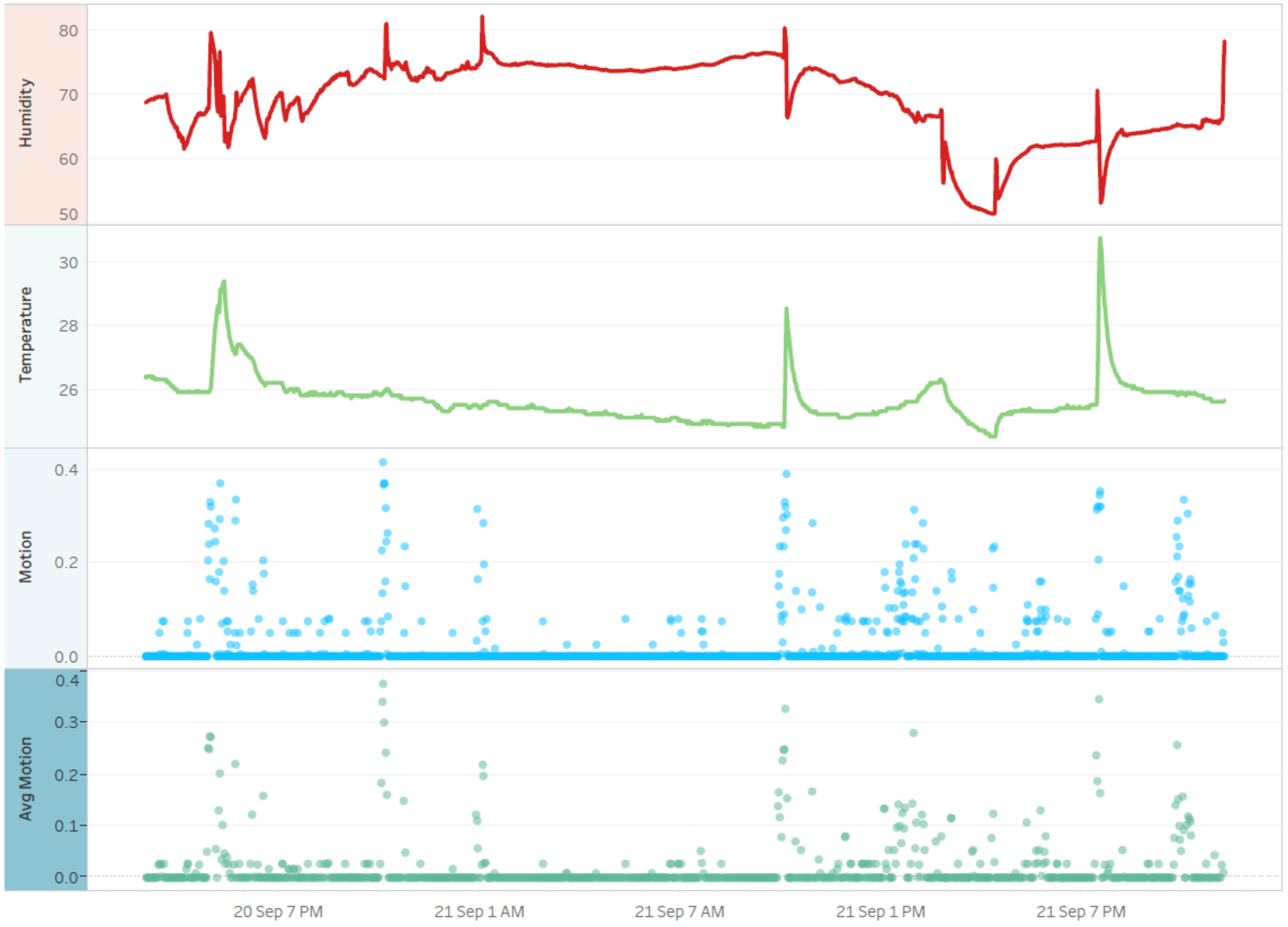
Looking at the graph, we can see two more spikes in temperature on the 21st of September at approximately 10 am and 7pm.

The kitchen visits recorded manually that day were:

Date	Time (approx.)	Activity
September 21	10am	Fry an egg
	2pm	Use the microwave
	7:30pm	Cooking on the stove

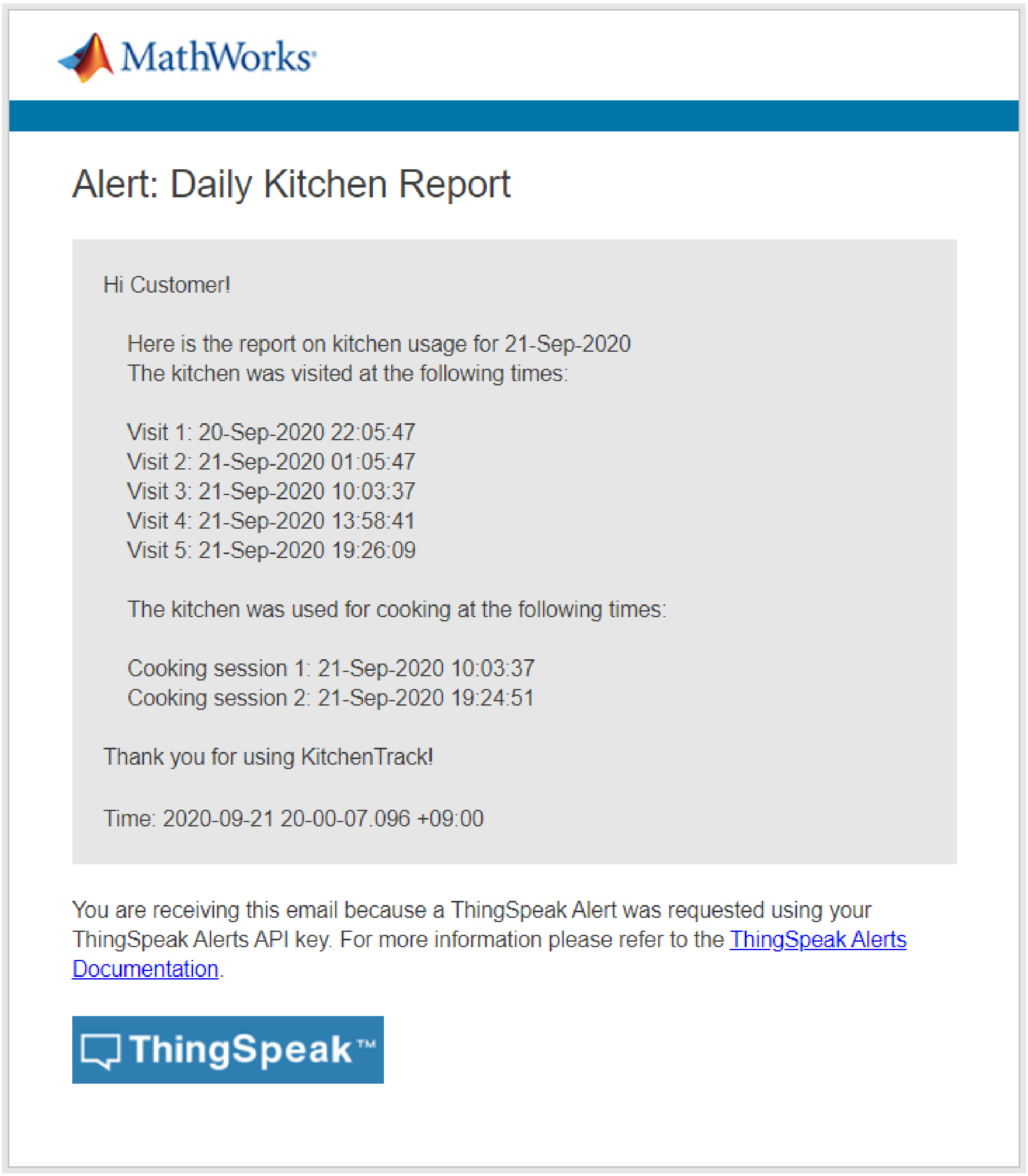
Rule 1 seems to be valid as there were many dips and spike in humidity unrelated to the cooking activities. For example, there is a spike at around 11pm on September 21st. We can see that the spikes in temperature spikes correspond with the times the stove was used, but not the microwave. There was a smaller temperature spike when the microwave was used but further testing would be needed to determine threshold for this. Rule 2 is validated by this data. Rule 3 also holds up as in both cooking events, the temperature returned to the previous baseline temperature quickly, dropping at least 1 degree within 5 minutes. We can also see that the average motion accurately depicts kitchen visits even when they do not use the stove, validating Rule 4. The motion detection also stopped so dual reports were not sent, validating Rule 5.

Results



Humidity, Temperature, Motion and Average Temperature by Minute [September 2020]

Figure 10



MathWorks

Alert: Daily Kitchen Report

Hi Customer!

Here is the report on kitchen usage for 21-Sep-2020
The kitchen was visited at the following times:

Visit 1: 20-Sep-2020 22:05:47
Visit 2: 21-Sep-2020 01:05:47
Visit 3: 21-Sep-2020 10:03:37
Visit 4: 21-Sep-2020 13:58:41
Visit 5: 21-Sep-2020 19:26:09

The kitchen was used for cooking at the following times:

Cooking session 1: 21-Sep-2020 10:03:37
Cooking session 2: 21-Sep-2020 19:24:51

Thank you for using KitchenTrack!

Time: 2020-09-21 20-00-07.096 +09:00

You are receiving this email because a ThingSpeak Alert was requested using your ThingSpeak Alerts API key. For more information please refer to the ThingSpeak Alerts Documentation.

ThingSpeak™

Figure 11

Discussion

Limitations

This was a preliminary test of a prototype, and therefore these results only reflect one day of data. Given the reliance of the device accuracy on temperature, it would need to be tested over a longer period of time in more extreme temperatures. Currently it is autumn where this data was recorded and the weather is very mild, with little variation between day and night. There is no artificial cooling or heating used in the kitchen, even though that could affect the results. More testing is needed to determine how the device will perform in different environmental situations.

Furthermore, the data collected does not give us any knowledge about how kitchen size affects the data – a larger kitchen may not be accurately measure with a single device. The device was able to be placed directly next to the stove in this kitchen, but different device placements will affect how the sensors read the temperature and humidity changes. Furthermore, food preparation does not necessitate the use of the stove and the use of other appliances and preparation spaces would likely affect the accuracy.

This means that the data collected will always have some degree of ambiguity, necessitating interpretation of the email data from the user. Specifically, they may not be able to understand what constitutes a “visit”. The device also only measures whether a person is in the kitchen or not, regardless of their activities. Someone visiting the kitchen for non-culinary purposes would still register as a visit. Someone preparing a salad for dinner would not register as cooking. In these situations, the results would not accurately reflect the real situation. Further development is needed to create a device that can measure different types of food preparation.

Challenges

The primary challenges in this project were accurately gathering and transmission of the sensor data. The preliminary testing of the threshold seems to reveal stove use at least can be accurately measured. However, the efficacy of the program used is entirely dependent on the reliable gathering of data. This device took multiple builds to be able to provide a stable collection of data. The project was always limited by the hardware, so I learned that it is important to complete the hardware development stage and then adapt the software to fit the limitations of the hardware.

Ethical concerns

This project required the completion of a Low Risk Ethics Application Form, submitted via email on the 7th of September during the project planning phase. This project used data gathered about humans, and therefore there are issues of handling private data. For the purposes of testing these thresholds, it was necessary to record my daily activities. If this were to be tested on a wide scale, that would require people to give private information about how they spend time at home and the layout of their kitchen. To prevent misuse, it seems necessary to deidentify data when it is uploaded, and to collect kitchen profiles anonymously as well.

If this device was a real product, it would pose an even greater risk to privacy, and so the security capabilities of ThingSpeak would need to be determined. An unsuitable server could mean unauthorised people have access the data. This data could potentially be used to track if someone is home or not in order to plan criminal activity. Therefore, it is essential that the release of the data be tightly controlled. The Twitter option would be done using private accounts, an optional. The email would need to be verified, and any customer accounts would need two-step authentication.

However, the greatest ethical concern is with regards to the target user. Given that the device is designed specifically for people with a mental impairment, it would be necessary to determine ethical standards regarding the use of this device. Although the Information Technology Professionals Association [1] has guidelines, they are quite vague about this specific area of health data. The Australian Human Rights Commission [2] published a report with specific proposals regarding technology and the disabled, but its target audience is policymakers and it is difficult to apply to this specific situation. Further research would be needed, but some preliminary ideas are that the device should in some way indicate its purpose – perhaps through the use of a LED screen that indicates when it is recording data. The device manual would also need to be written with the target user in mind, to make it easy for them to understand its purpose.

References

[1] Information Technology Professionals Association, “Code of Ethics,” 2020. [Online]. Available: https://www.itpa.org.au/code-of-ethics/. [Accessed 24 September 2020].

[2] Australian Human Rights Commission, “Human Rights and Technology,” Australian Human Rights Commission, Sydney, Australia, 2019.