Image Recognition Expiration Date Tracker

By

Kevin Choi

Jonathan Jacobson

Vaibhav Makar

Final Report for ECE 445, Senior Design, Spring 2021

TA: Bonhyun Ku

05 May 2021

Project No. 66

Abstract

This paper outlines a device used to remind users of expiring food in their refrigerators, resulting in decreased food waste. When a user wishes to insert new food into their refrigerator, the item is scanned by this device using computer vision and automatically added to a cloud database accessible from an Android application. The device monitors this database every time the refrigerator door is open and alerts the user of possibly expired food through both the application and a buzzer located on the device. The database can also be modified by the application, allowing the user to create or correct the entries as necessary. This project resulted in a working prototype.

Contents

1. Introduction

For the busy consumer it may be difficult to remember what food is in the fridge, and because of this food is wasted either through spoilage or through buying multiple of the same item. The presence of some reminder system might be able to lower this food waste.

Our solution to this problem was to create a device that sits inside a refrigerator and scans food items, adding them to an automatically generated timer list that will alert a user when an item is expired. The device connects to a mobile application that will assist in keeping the user up to date on how long food has been in their refrigerator.

This report will contain the engineering and design decisions made when creating the Image Recognition Expiration Date Tracker—mainly designing the device's architecture in terms of subsystem organization and communication along with construction or programming of the individual components. In the next chapter, there will be an in-depth description of the design. In the third chapter, there will be a discussion of the methods of verification used, the fourth an analysis of costs associated with the project, and a concluding chapter outlining the results and accomplishments of this project.

1.1 Background

Approximately 30-40% of the food supply in the United States is wasted every year. This amount of food waste is the equivalent of about 160 billion dollars [1]. On top of this, an estimated 48 million Americans get some form of food poisoning every year [2]. Ways to prevent these may include consumers being more aware of the types and amounts of food they buy, being conscious to only buy the food they plan on eating, but this has shown not to be enough.

Our design would require minimal user interaction and eliminate the need to memorize what's in the refrigerator. This system will lower food waste and offer greater user satisfaction as it could reduce grocery spending.

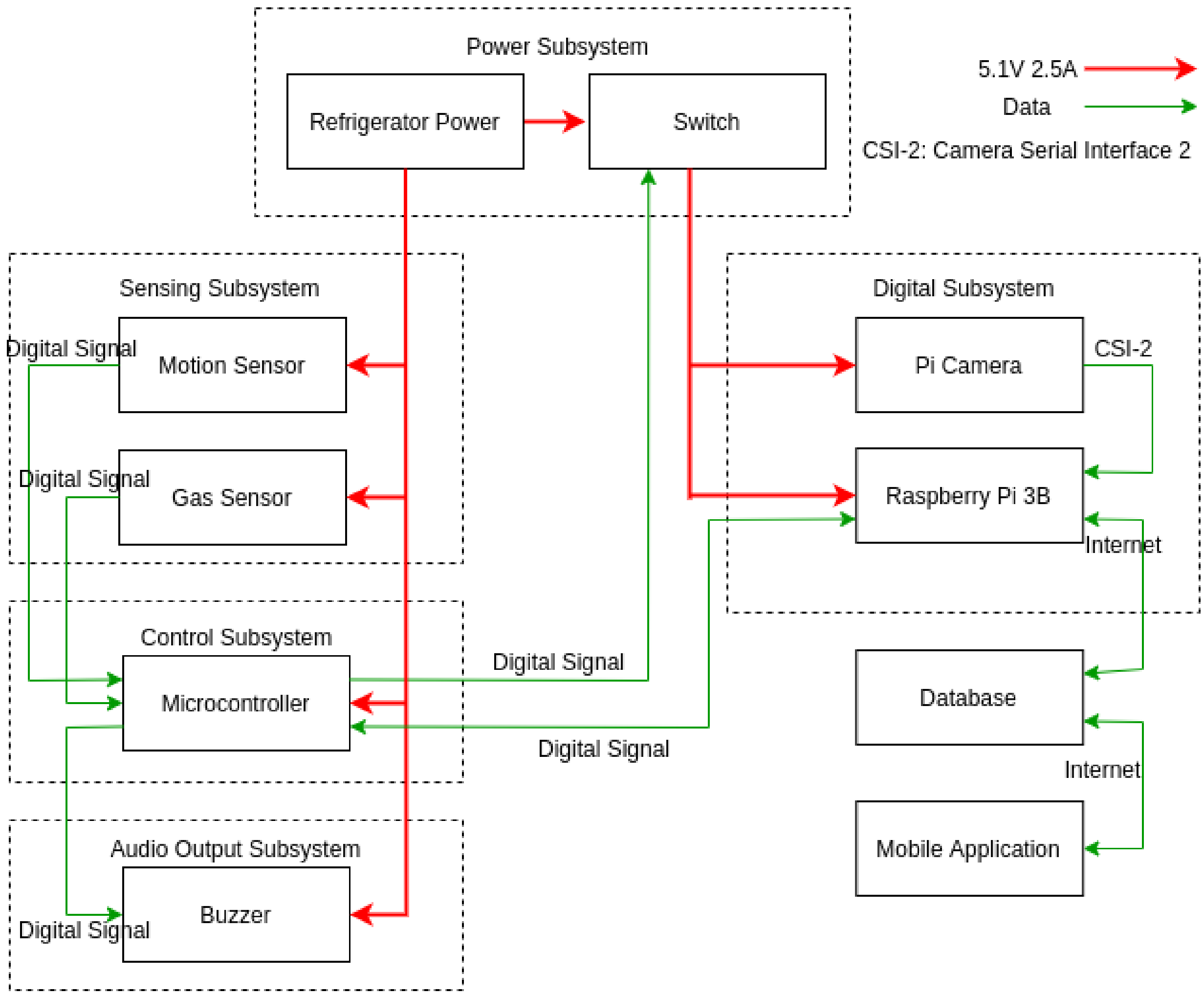


Figure 1. Block Diagram of the device.

2. Design

The major components of this design include the control board, the computer vision program, and the mobile application. In this section, we will detail the design of all three.

2.1 Control Board

The control board is built around the microcontroller, which connects to the sensors and alarms along with the Raspberry Pi 3B+ through a 5V USB power connection. The control board also takes the input voltage and steps it down to the 5V required for the other components.

2.1.1 Power Subsystem

The power subsystem contains the circuits needed to step down the input voltage to 5V and to control the power provided to the Raspberry Pi.

Due to our power most likely coming from the refrigerator itself, we assumed that the input voltage would be somewhere around 12-15V DC. This led us to choose a voltage regulator that took in a range of values from 7V to 35V and could step it down to 5V. We also needed to be able to run a large current through this device, as it would be powering the entire system, including the Raspberry Pi which had a recommended current capacity of 2.5A for our power supply.

Eventually, we settled on using the LD1085V50 voltage regulator as it met these requirements, with the ability to pass through a maximum of 3A.

This subsystem also includes the circuit required to pass the 5V power line to the Raspberry Pi. We needed to find a way to take a digital signal from the microcontroller, which is a low-current 5V signal, and use it to open a path from the main 5V line to the Raspberry Pi. To do this we designed a circuit taking advantage of the switch functionality of a high-power p-channel MOSFET (SUP53P06-20).

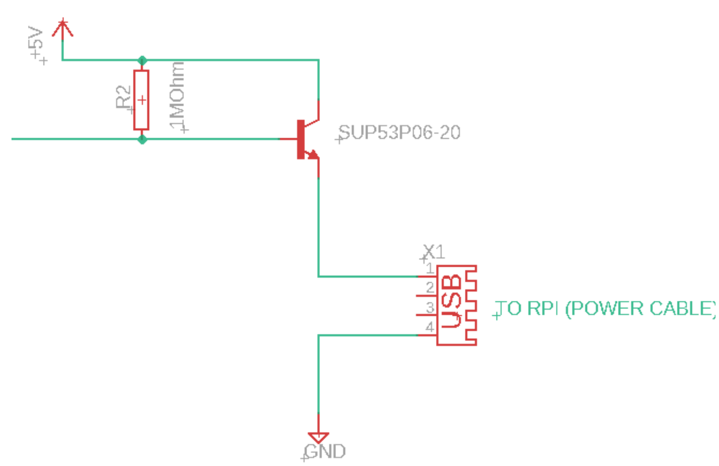


Figure 2. Raspberry Pi Power circuit. The signal from the left is a digital signal from the microcontroller.

2.1.2 Sensor Subsystem

The control board also connects and monitors the various sensors built into the device. These include the door sensor, the gas sensor, and a pin reading a digital signal from the Raspberry Pi to determine when a timer has expired.

The door sensor is a simple switch that is in the ON state when the door is opened, and in the OFF state when the door is shut. This sensor allows the microcontroller to know when the door is opened so that it can power on the Raspberry Pi.

Figure 3. Door switch. This switch is ON when the switch is not being pressed, and OFF when pressed.

The MQ-137 gas sensor is an ammonia gas sensor designed around the material Tin (IV) Oxide. When ammonia gas comes into contact with the Tin (IV) Oxide, the conductivity of the material increases. This causes an increase in the voltage across the sensor pins [3][4]. We originally chose this component because it was within our price range, had an accuracy rating high enough for our purposes, and was readily available. However, we failed to take into account that the Tin (IV) Oxide needs to be heated up in order to have this property. The sensor itself does contain a heating element, but due to the fact that we are placing this device into a refrigerator it does not work correctly. During testing we also faced the issue that the sensor takes over 24 hours to heat up to the operating temperature. Going forward this sensor would need to be replaced with one with a different method of operation.

Figure 4. The MQ-137 gas sensor. This sensor uses the conductive variability of Tin (IV) Oxide to operate.

2.1.3 Buzzer Subsystem

The buzzer subsystem consists of a simple circuit designed to use minimal external components. A digital pin connected to this circuit cycles rapidly from HIGH to LOW, and is amplified by a transistor. This varying signal is sent to the buzzer and the sound is made. To create the loudest sound possible without additional amplification, the frequency of the digital pin changing was made to match the resonant frequency of the buzzer.

In the future this subsystem might be expanded to include a voice alert, so that the user is told exactly what food has expired, and where it might be. For now, our system relies on the user checking the Android application to see which food item has been marked possibly expired.

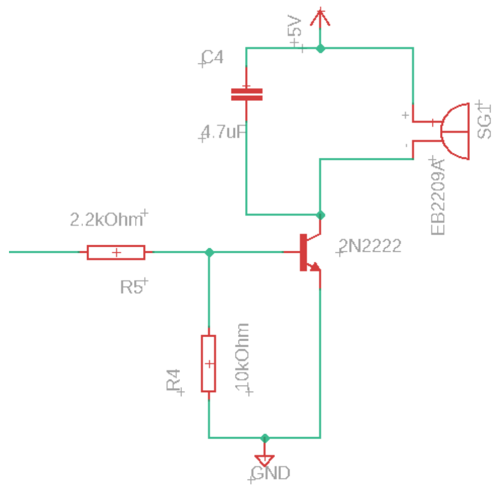


Figure 5. Buzzer circuit schematic. The signal from the left is the digital signal from the microcontroller.

2.2 Computer Vision

Raspberry Pi 3B+ takes charge in computer vision, which classifies the object with its camera and updates the entry to the database.

2.1.1 Image Classification

The image classification system utilizes TensorFlow Lite (tflite), the low resource version of the open-source software library for machine learning, TensorFlow. The interpreter core of tflite, responsible for executing the model in our python script, uses less than 300 kb of memory with a model size of 4 MB. Upon the detection of the door opening, the Raspberry Pi boots up and runs the script, which includes the authentication to the database, enabling the signal output to GPIO ports for signaling the buzzer, the activation of its attached camera, and the load of the tflite interpreter core [5]. The entire boot-up process takes about 20 seconds in the Raspberry Pi’s terminal mode, allowing the concept of scanning food items on every door opening.

Once the script is loaded, the Raspberry Pi starts scanning the object by taking pictures every 0.2 seconds. Unlike most of the simple classification scripts that use streaming footage (video), our script uses the image port of the Raspberry Pi to get the highest quality image available. Afterward, it processes the image and classifies it based on its pre-trained tflite interpreter core. It updates this information on the database after a set of validations to minimize false classifications.

2.1.2 Database

The database entries include three categories: the current time when the scanning happens, the food item classified, the estimated expiration date. We used the expiring durations of food items inside a refrigerator based on web sources [6][7] and specifically used the lower bounds of the durations since our system cannot account for the different settings of refrigerators. The entries are checked upon every Raspberry Pi boot-up to identify any expired food items. The script checks the third column of entries, the previous "current time" plus the expiring durations, to determine if any of them are earlier than the current time. If there are any, the script changes the font color of the entry for increased visibility and signals the buzzer.

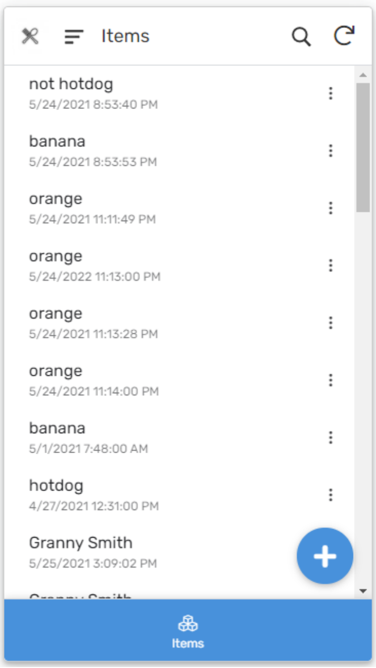
Creation date	Item Category	Expiration date
2021-04-26 20:53:40	not hotdog	2021-05-24 20:53:40
2021-04-26 20:53:53	banana	2021-05-24 20:53:53
2021-04-26 23:11:49	orange	2021-05-24 23:11:49
2021-04-26 23:13:16	orange	2022-05-24 23:13:00
2021-04-26 23:13:28	orange	2021-05-24 23:13:28
2021-04-26 23:14:00	orange	2021-05-24 23:14:00
2021-04-27 7:48:37	banana	2021-05-01 7:48:00
2021-04-27 12:29:49	hotdog	2021-04-27 12:31:00

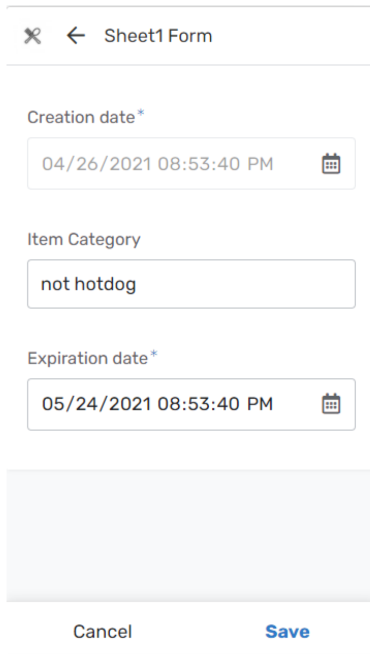
Figure 6. Database including expired items.

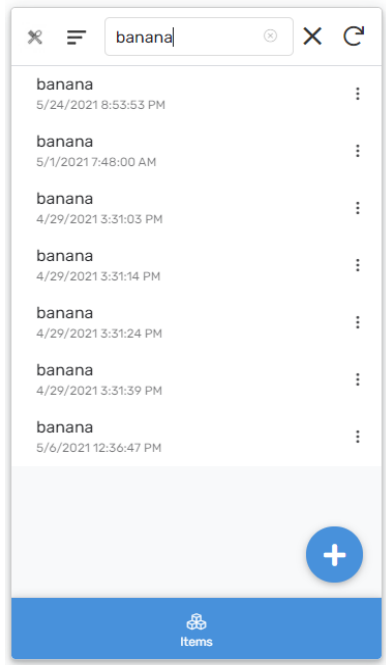
2.3 Mobile Application

Alongside the Control Board and Computer Vision subsystems, we designed a mobile application to allow the user easy access to the items stored in the spreadsheet.

The application was developed for Android mobile phones, and was developed using React Native, a framework based on Javascript. It connects to the sheet using the Google Sheets API, and allows the user to make changes to and view the spreadsheet in real time through a wrapper. It supports functionality to search for, add, remove, and edit items as well as their expiration dates. Users cannot edit Creation date as the timestamp serves as a unique identifier within the database.







Figures 7, 8, and 9. Screenshots from the mobile application.

3. Design Verification

To verify that our design was functioning correctly, we created both individual tests and diagnostic boot-up codes that alerted us whether a block was working. We made tests for the control board, image classification system, and database/application software.

3.1 Control Board Design Verification

To verify the control board, we designed some diagnostic boot-up code to help us see what was going on during the board's operation. This code tested the power circuits, sensors, and alarm circuits.

3.1.1 Power Subsystem

For the power subsystem, we needed to verify that it steps down the input power to 5V and provides stable enough voltage to power the board. To do this, we plugged the board into a 9V wall adapter and measured the voltage coming from the regulator with a voltmeter. This voltage was 4.99V which was what we needed. To test whether the power was stable enough to run the board, we wrote code that made the buzzer continuously buzz on startup. This test was also successful.

To test the USB power circuit, we wrote another portion of the start-up code to send power to the Raspberry Pi for 1 minute after the buzzer stopped. This way, we could have a voltmeter constantly measuring the output pins of the USB and see how quickly the voltage changed and whether it was a constant 5V. This test was successful as we measured the voltage of 4.99V for the entire minute, dropping to low levels otherwise. We then tried this experiment again with the Raspberry Pi connected, and the boot-up was successful. This experiment showed that the Raspberry Pi was able to draw the needed current for operation.

3.1.2 Sensor Subsystem

As mentioned previously, to test the boot-up code, we had a section where the buzzer started on boot-up. We tested the door sensor by having the buzzing continue until the door sensor was pressed, stopping the buzz. This test proved that the door sensor was functioning.

To test the gas sensor, we attempted to create a closed environment with rotting food in it to see if the sensor accurately detected the presence of ammonia gas. Unfortunately, we were not getting any results from this test. Upon deeper digging into the sensor's datasheet, we discovered that the sensor takes over 24 hours of heating to get to the operational state. After realizing this, we turned our focus to other portions of the project.

3.1.3 Buzzer Subsystem

With our code triggering the buzzer on boot up, it was immediately apparent that the buzzer was operational. We ensured that we ran the buzzer at its resonant frequency during the design by changing the frequency incrementally and choosing the value that provided the most acceptable output.

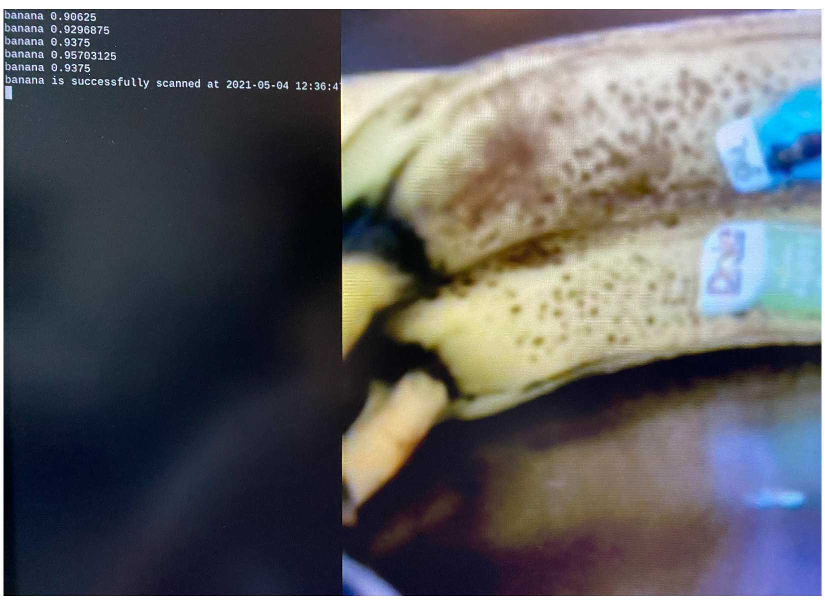
3.2 Computer Vision Design Verification

To verify the computer vision, we divided the process into image classification and database update to test each and merge later. After unit-testing each, we tried the whole process replicating the user behavior to determine its functionality.

3.2.1 Image Classification

We spent the most time verifying image classification as we thought it was the most vital determinant of our project's success. The testing steps involved building a TensorFlow model, converting it to a tflite model, and running it with live footage from the Raspberry Pi.

The classifiable objects include food items and several miscellaneous household items, as the scanning continues regardless of the presence of food items. By having a more comprehensive range of classifiable objects, we could reduce false classifications by giving the interpreter more options to classify and later check if it is a food item to validate. We made a validation step of Raspberry Pi detecting the same food item five consecutive times with higher than 80% accuracy and tested it with several food items. The result was successful as it could classify most of the food items in the list with higher than 90% accuracy.



2021-04-27 15:40:26	tomato soup	2021-04-30 15:40:00
2021-05-04 12:36:47	banana	2021-05-06 12:36:47

Figure 7. Computer Vision Demonstration.

3.2.2 Database Update

The verification for the database included authentication, tokenization for easier access later, and updating of entries. We used the mobile application to access this database and found out that the testing script could successfully write an entry.

The combining test was also successful as the device correctly detected the food items and wrote the accurate entry to the database.

3.3 Mobile Application Design Verification

Verification of the Mobile Application was fairly simple, as it only involved connecting it to the spreadsheet and testing all possible user operations. We were able to verify that the user was able to add, remove, and edit items from the list. Furthermore, we were able to verify that all these changes were also conveyed over to the spreadsheet. We also tested to ensure that new items scanned from the Pi also showed up in the application. Another aspect of the application we tested was the search function, which we confirmed only displayed results that matched the user's inputted search criteria.

4. Costs

The costs of this project were minimal. We tried to choose low-cost, readily available components to make our product accessible to as many people as possible.

4.1 Parts

Part	Manufacturer	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
MQ-137	Zhengzhou Winsen Electronics Technology Co., Ltd	34.95	34.95	34.95
Raspberry Pi Camera Module V2	Raspberry Pi Foundation	24.45	24.45	24.45
PIC18F25K83-E/SP Microcontroller	Microchip Technology	2.95	2.95	2.95
EMX-7T05SP Magnetic Buzzer	East Electronics	2.24	2.24	2.24
			Total	64.59

Table 1. Parts Cost

4.2 Labor

Rate based on the ECE undergraduate employee rate.

Employee	Rate ($)	Hours worked (Hrs)	Labor Cost ($)
Jonathan	15.75	40	1,575.00
Kevin	15.75	40	1,575.00
Vaibhav	15.75	40	1,575.00
		Total	4,725.00

Table 2. Personnel Labor Cost

5. Conclusion

5.1 Accomplishments

This project accomplished creating a device that would be able to scan food items and add them to an automatically generated timer list. The device successfully connects to the mobile application that informs the user which food is expired.

5.2 Uncertainties

The uncertainties related to this project include data about its utility. We attempted to create a product that needed minimal user interaction, simplifying user interfaces wherever we could. However, a large part of the failures of similar products is that they are too complex for the user to remember to do each time. We hope to make this a seamless experience by keeping our device in the refrigerator and automatically starting the system when the door opens.

5.3 Ethical considerations

Ethical considerations of this project include increased human reliance on technology. One of the automatically generated expiration dates may be wildly off. Our manual-input feature can remedy this, but if the user is not aware that the food has expired, this device could give them a false sense of security in the food they're eating. It would be imperative to market this device as a helpful reminder service and not a substitute for good judgment.

5.4 Future work

Future work with this project could include expanding the device to documenting food throughout the entire kitchen. If this were a countertop device, it would be possible to scan food items going into places other than the refrigerator. This approach could further eliminate food wastage.

Also, with more cameras and a more powerful processor, it might be possible to get interior views of the entire refrigerator, eliminating the need for users to scan the food items by hand. There could be a system in place that determines how long food has been in the fridge by checking the contents of the entire fridge every day. This method would lead to decreased user interaction, possibly creating an easier-to-use system.

References

[1] P. S. Mead, L. Slutsker, V. Dietz, L. F. McCaig, J. S. Bresee, C. Shapiro, P. M. Griffin, and R. V. Tauxe, “Food-Related Illness and Death in the United States - Volume 5, Number 5-October 1999 - Emerging Infectious Diseases journal - CDC,” *Centers for Disease Control and Prevention*, 15-Dec-2010. [Online]. Available: https://wwwnc.cdc.gov/eid/article/5/5/99-0502_article. [Accessed: 05-May-2021].

[2] “Peak summer: More than one in six people likely to get food poisoning,” *American Osteopathic Association*, 19-Jul-2019. [Online]. Available: https://osteopathic.org/2019/07/02/peak-summer-more-than-one-in-six-people-likely-to-get-food-poisoning/. [Accessed: 05-May-2021].

[3] “TECHNICAL DATA MQ-137 GAS SENSOR,” *HANWEI ELECTRONICS CO., LTD*. [Online]. Available: https://eph.ccs.miami.edu/precise/GasSensorSpecs/NH3.pdf. [Accessed: 05-May-2021].

[4] “Ammonia Gas Sensor (Model: MQ137) Manual”, *Zhengzhou Winsen Electronics Technology Co., Ltd.* [Online]. Available: https://cdn.sparkfun.com/assets/7/0/2/f/8/MQ137__Ver1.4__-_Manual.pdf. [Accessed: 05-May-2021].

[5] K. Choi, “classify_picamera.py,” *Food Savers*, 05-May-2021. [Online]. Available: https://github.com/kchoiuiuc/Food_Saver/blob/master/classify_picamera.py. [Accessed: 05-May-2021].

[6] “How Long Does Food Last? Guide to Shelf Life & Expiration,” *Eat By Date*, 12-Aug-2015. [Online]. Available: http://www.eatbydate.com/. [Accessed: 05-May-2021].

[7] E. Huffstetler, “How Long Does Food Stay Good in the Fridge?,” *The Spruce Eats*, 05-Oct-2020. [Online]. Available: https://www.thespruceeats.com/the-shelf-life-of-refrigerated-foods-1389306. [Accessed: 05-May-2021].

Appendix A Requirement and Verification Table

A.1 Sensing Subsystem

Requirement	Verification	Status
The system detects when the door opens and closes.	1. Open the door for at least 10 seconds. 2. Close the door and wait a few moments (~ 1 min). 3. Open the door again. 4. Check if the Raspberry Pi had completely turned off and turned on again.	Y
The device will be able to sense the presence of gasses related to the expiration of food. (10 pts)	1. Verify that there are no gasses detected. 2. Add a non-expired item and seal the system (refrigerator). 3. Wait and see if gas is detected. 4. Remove the previous item and add an expired item. 5. Wait and see if gas is detected.	N

Table 3. Sensing Subsystem RV Table

A.2 Digital Subsystem

Requirement	Verification	Status
The device can classify food items and create corresponding timers.	1. Scan each food item. 2. Check if the device correctly classified and created the timer for the food item with the application.	Y
The device can set a custom expiration timer based on the user input.	1. Check if there is a recently scanned food item. 2. Modify the timer with the application. 3. Test if the entries are appropriately modified.	Y

Table 4. Digital Subsystem RV Table

A.3 Power Subsystem

Requirement	Verification	Status
The switch will be able to supply and decline power to Raspberry Pi.	1. Verify that the switch can supply the power for Raspberry Pi while the door is open. 2. Verify that the switch declines the power completely when the door is closed.	Y

Table 5. Power Subsystem RV Table