Phase 4 Report: Final Report

Ultra Smart Refrigerator

Team #25

A Report

Presented to

The Department of Electrical & Computer Engineering

Concordia University

In Partial Fulfillment

of the Requirements

of ELEC/COEN 490

by

|  |  |
| --- | --- |
| Georgiy Araslanov | ID: 40002321 |
| Gneykou Kengne Yvann Monny | ID: 40015234 |
| Nicholas Pollender | ID: 27398050 |

Project Supervisor

Sébastien Le Beux

Concordia University

April 16th, 2020

*“We certify that this is our original work.”*

# 1. Abstract

The phase 4 report is the final report detailing the overall completion of the project. The specifications, requirements and diagrams have been revised and finalized. This report presents the completed work of the project, the problems encountered and how they were overcome as well as several testing and validation techniques.

“Is your refrigerator not meeting your home appliance needs? The GYN Ultra Smart Refrigerator is the solution for you! Never worry about running low on food, forgetting your shopping list, or what you’re going to eat! With GYN, keep everything organized and enjoy unlimited access to our database of healthy recipes! So, what are you waiting for? Buy your GYN Ultra Smart Refrigerator today!”

Table of Contents

[1. Abstract 0](#_Toc37923329)

[2. Introduction 3](#_Toc37923330)

[2.1 Objectives 3](#_Toc37923331)

[2.2 Project Outline 3](#_Toc37923332)

[3. Description 4](#_Toc37923333)

[3.1 Design 5](#_Toc37923334)

[3.1.1 The Refrigerator 6](#_Toc37923335)

[3.1.2 The Server 6](#_Toc37923336)

[3.1.3 The Raspberry Pi 7](#_Toc37923337)

[3.1.4 The Application 8](#_Toc37923338)

[3.2 Development 9](#_Toc37923339)

[3.2.1 The Hardware 9](#_Toc37923340)

[3.2.2 The Server 10](#_Toc37923341)

[3.2.3 The Software 11](#_Toc37923342)

[3.3 Problems Encountered 13](#_Toc37923343)

[3.3.1 Major Problems 13](#_Toc37923344)

[3.3.2 Minor Problems 14](#_Toc37923345)

[3.4 Solutions Proposed 14](#_Toc37923346)

[4. Results 15](#_Toc37923347)

[4.1 Test Cases 15](#_Toc37923348)

[4.2 Minor Test Cases 19](#_Toc37923349)

[4.3 Additional Notes on Validation 21](#_Toc37923350)

[5. Conclusion 21](#_Toc37923351)

[5.1 Lessons Learned 21](#_Toc37923352)

[5.2 Closing Remarks 21](#_Toc37923353)

[6. References 22](#_Toc37923354)

[7. Appendix 23](#_Toc37923355)

[7.1 User Manual 23](#_Toc37923356)

[7.2 Technical Manual 36](#_Toc37923385)

[7.3 ELSEE Aspects 54](#_Toc37923398)

[Task 1 – Identifying Stakeholders 54](#_Toc37923399)

[Task 2 – Listen to the Stakeholders 54](#_Toc37923400)

[Task 3 – Learn from Experts 55](#_Toc37923401)

[Task 4 – Addressing ELSEE aspects 55](#_Toc37923402)

[Task 5 – Validation of the Projects Approach 56](#_Toc37923403)

[Task 6 – Reflection and Remarks 56](#_Toc37923404)

# 2. Introduction

In an ever-expanding world, the technologies used evolve as well. Refrigerators have been influenced by evolution over the years, eventually coming up with the ‘Smart Refrigerator’. Smart refrigerators come equipped with a plethora of new features, ranging from scheduling water to boil every morning, to a tablet installed in the door. Some features are more useful than others, and some have never been implemented. That is where the Ultra Smart Refrigerator comes in; a new take on smart home technology that aims to expand on existing features and introduce new features too.

## 2.1 Objectives

The objective of this project was to design and build the prototype of a refrigerator that expands on existing smart refrigerator technology and adds a few brand-new features as well. The design should be capable of visually informing the user about food types and quantities through a smartphone application, using the monitoring hardware within the refrigerator while also promoting organization. The application also supplies the user with a search engine for healthy recipes as well as suggest recipes based uniquely on the refrigerator’s inventory to promote a healthy lifestyle.

## 2.2 Project Outline

The project is split into three components.

Firstly, the hardware. This includes the refrigerator itself and the devices/sensors (such as scales, camera, temperature sensor, hall effect sensor and buzzer). Each device is ultimately connected to the micro-computer processing all the information.

Secondly, the server. It serves as a median for all the data gathered by the refrigerator as well as a database for all the recipes. The data is sent off to the smartphone application when it is required.

Thirdly is the software. The smartphone application is the primary means of interfacing with the data, and the program that runs within the refrigerator.

# 3. Description

Below are revised design specifications and system requirements based on what has been completed. The design specifications and requirements have been reviewed and re-assessed multiple times since Phase 1 to accommodate oversights and mistakes. Normal conditions assume the devices are operating in a suitable environment with no external source acting upon it.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Requirement/Parameter | Test Conditions | Values | | | Units |
| Min | Typ | Max |
| 1 | Refrigerator Dimensions (HxWxD) | - | - | 52x52x84.5 | - | cm |
| 2 | Maximum Weight on Load Cells | Normal | 0 | 2 | 10 | kg |
| 3 | Operating Temperature | Normal | - | 4 | - | ⁰C |
| 4 | Wi-Fi Connection Range | Normal | - | 5 | 90 | m |
| 5 | Refrigerator Operating Voltage | Normal | - | 120 | - | V |

Table 1: Revised Design Specifications

|  |  |
| --- | --- |
| ID | Description of the Functional Requirements (FR) |
| FR - 1 | The refrigerator will be able to measure the quantities of food. |
| FR - 2 | A software will be able to connect to the refrigerator, will receive and visually display the data collected by the refrigerator. |
| FR - 3 | The refrigerator will take photos within its interior and the user will be able to have a view of the inside of the refrigerator through the software. |
| FR – 4 | The user will be alerted in case the refrigerator has been opened for too long. |
| FR – 5 | The software will suggest healthy recipes, inform about nutritional values of foods and provide visual data about food usage/habits and potentially expired food. |
| FR - 6 | The software will store data on the device so that the information can still be accessed without an internet connection. |

Table 2: Revised Functional Requirement

|  |  |
| --- | --- |
| ID | Description of the Non-Functional Requirements (NFR) |
| NFR - 1 | The overall physical and visual design of the refrigerator will encourage users to keep everything organized and therefore save some time for the user. |
| NFR - 2 | The refrigerator will accommodate a broad target audience. Its features will be user friendly, it will be adapted for children, elderly and disabled people. |
| NFR - 3 | The features that the product will offer should inspire the users to achieve/keep up healthy habits. |

Table 3: Revised Non-Functional Requirements

## 3.1 Design

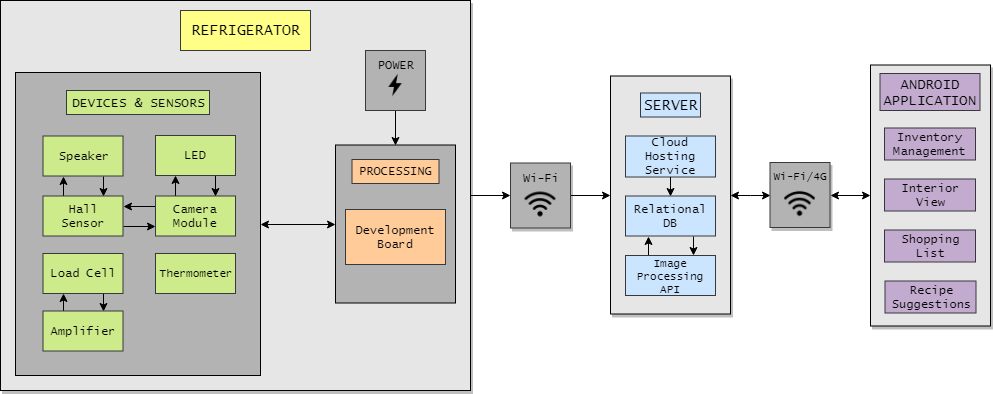
As mentioned previously, the project is split into three different components: hardware, server, and software. The information to follow describes all the work that has been completed and is working as intended. The block diagram below illustrates the three main components and their sub-components and how they are interconnected as of phase 3’s design approval. Overall, the project is partially completed. All the main features have been implemented; only some minor features were omitted (not depicted in the block diagram). This will be further elaborated in section 3.3.

Figure 1: System Block Diagram

### 3.1.1 The Refrigerator

It was determined that best way to design the refrigerator was to add parts to an existing refrigerator. In this case, a mini refrigerator was used, as a full-scale one would not be easy to transport or store. The appliance in question is the Danby DAR125SLDD which was re-purposed for the development of this project.

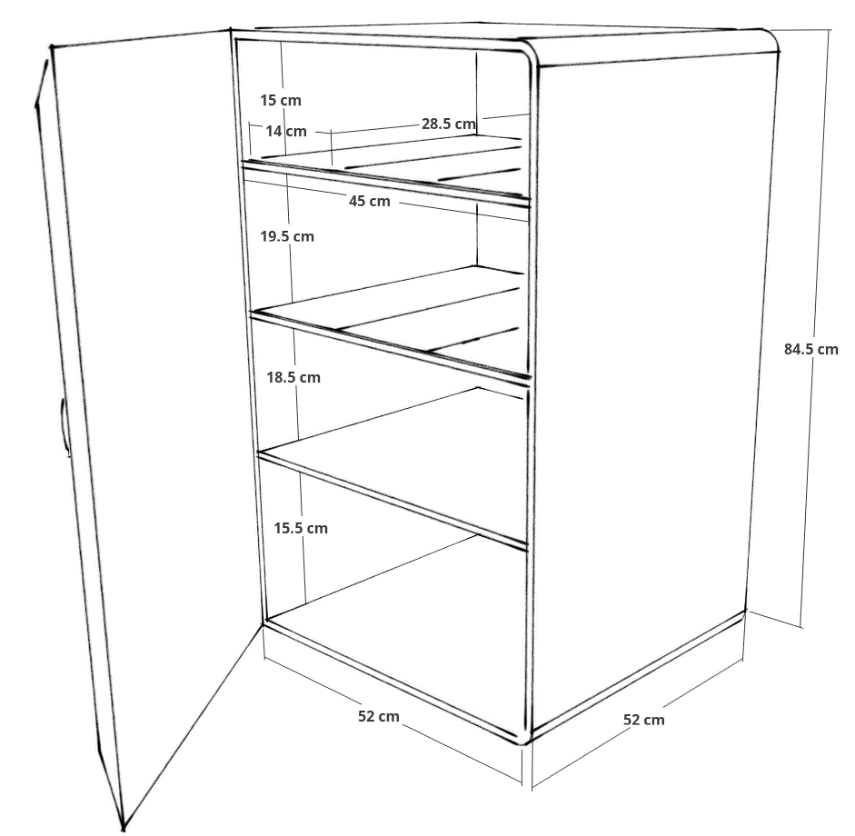


Figure 2: Danby DAR125SLDD Model and Dimensions

The top two shelves are used as the partitions for the scales. Each scale is composed of a tray and four SEN-10245 square shaped load cells on each corner of the tray, resembling a common household scale configuration. The load cells rest between the tray and the shelf and measure the change in weight applied to the tray. Each set of load cells are connected to the HX711 module, an amplifier specifically designed for load cells which convert the analog signal into a digital signal and amplify said signal so it can be properly read. The bottom shelf is monitored by the Raspberry Pi Camera V2, angled ideally to capture as much of the shelf as possible. These devices are connected to a Raspberry Pi 3 B+, a powerful micro-computer in a compact form factor with many capabilities. The Pi processes the data obtained by the load cells and the camera and sends it to the server wirelessly through a Wi-Fi connection. It should be noted that a PCB layout was designed for the overall connection of all the devices and sensors which would serve as a nexus for all the connections and ultimately reduce cable clutter.

### 3.1.2 The Server

The **refrigerator** sends a request to AWS to establish a connection. Once established, the fridge sends a request to store an image inside the S3 and another request to store the measurements in RDS. As soon as an image is uploaded to the S3 bucket, the Lambda function is triggered. Upon execution, it sends a request to Google Cloud through the Vision API to annotate the image with the desired information (which means items found in the image, as well as, any logo or text that might appear to describe the item). Google Cloud then responds with a JSON object containing metadata. Upon receiving the response, the Lambda function updates the database in RDS with the aforementioned data belonging to the image.

Next, the **user** on the other end, through the Android Application sends a request to AWS to establish a connection. Once established, the user requests the information regarding the refrigerator stored in RDS and S3. This information includes the image which describes the content of the fridge and the temperature of the interior of the fridge, to name a few. The information from the response also gives the application, the ability to recommend health recipes to the user.

Furthermore, significant time and effort were put into the **security** aspect of the system. While the refrigerator for example, only has the permissions to write into and not read from AWS resources (RDS & S3), the Android application only has read permissions into the database. However, without the proper credentials, the refrigerator and the Android application will not be able to read/write, from/to AWS. In addition to the aforementioned security measures, a whitelist of IP addresses is maintained in the Amazon Virtual Private Cloud (VPC) security to restrict access to AWS.

### 3.1.3 The Raspberry Pi

While it is not the main software component of the project, the Pi does run a simple program to process and send data to the server. Each scale requires calibration to ensure more accurate readings. When the program is first run, each scale is tared, setting the weight to zero. This assumes that there are no items on the scale at the time of taring. Every five minutes thereafter, the Pi will send the weights of each scale and a newly captured image to the server.

### 3.1.4 The Application

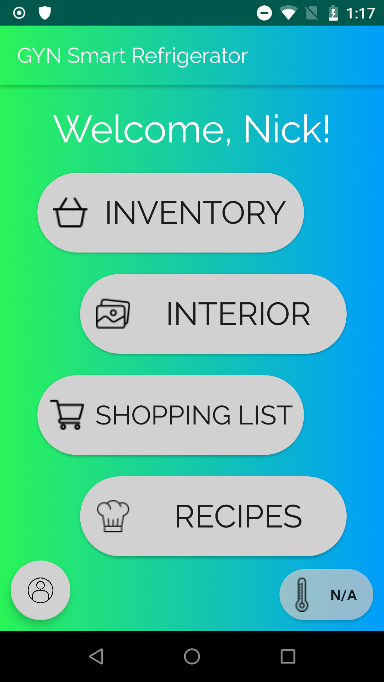
The application was designed for Android devices and is the only means for the user to interface with the refrigerator. It includes many useful features, such as a login, inventory management and interior viewing, a shopping list, and a searchable database of healthy recipes. Overall, the application sports a friendly user interface with large buttons and large easy-to-read text, soothing colors and simple navigation.

Figure 3: User Interface Design

Login – A simple security measure will prevent any unauthorized user from using the application and accessing data. The refrigerator has a unique identification and password that must be entered upon launching the application before gaining access to the features.

Inventory Management – From here, a user can view data of each partition, the weights recorded of each scale in the refrigerator and the partition name which can be changed at will. Furthermore, there is a refresh button that will fetch any updated weights from the server, and a button that opens a list of additional inventory/ingredients which is used to define extra items which are not typically stored in a refrigerator such as flour, salt, spices, etc.

Interior View – This feature is minimalistic; it fetches the latest image stored on the server (captured by the camera). The image is displayed to the user with a list of items found by the image processing API.

Shopping List – This feature is as stated, it provides the user with a shopping list. Items can be added or removed at will. Each entry includes an item name and a quantity. There is also a recommendation button which analyzes the weights of the scales, if the weights fall below a certain threshold, the user will be prompted to add the low inventory items to the shopping list.

Recipe Suggestions – The user may search for recipes based on name. Any close matches are displayed in the results, from which the user can view the recipe details, these details include but are not limited to serving size, prep time, descriptions, ingredients, instructions, etc. Additionally, the recipes can be bookmarked via the bookmark button, which can be viewed any time in the future without the need to search again. Also, there is a second button that analyzes the ingredients of the recipe and matches them to refrigerators inventory, and if there are any missing ingredients, they are automatically added to the shopping list.

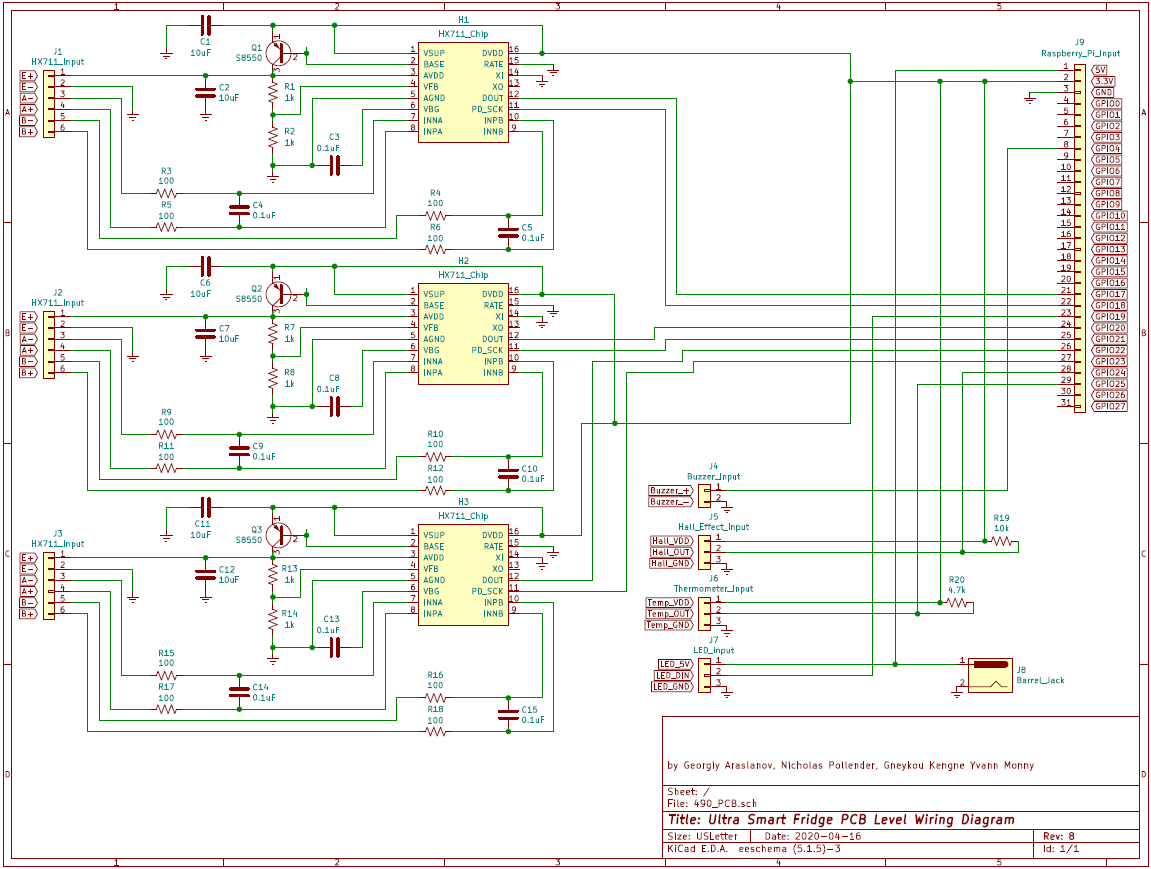
## 3.2 Development

Different procedures and methods for each component of the refrigerator were taken throughout the course of the project. This section will describe the process for developing various aspects and the software used to develop the project.

### 3.2.1 The Hardware

The hardware components were initially envisioned using the 3D modeling software SketchUp. This proved useful for determining dimensions and ensuring that all devices would fit within the refrigerator.

The PCB design was developed using the KiCAD software and includes every component necessary to connect all the devices. It even considers the bare-bones components of the HX711 modules. This configuration would enable the least amount of clutter in the refrigerator; only a bridge between the Raspberry Pi and PCB would be necessary along with the cables used to extend the other devices and sensors to their proper location (which would be easily concealable). As it will be explained in section 3.3, the PCB could not be printed.

****Figure 4: PCB Level Wiring Diagram

As far as hardware development goes, there were not any major developments required, as most of the project works lies with the software component. The scales were developed in the Capstone laboratory, testing and validating each component with the multimeters, and soldering various components.

### 3.2.2 The Server

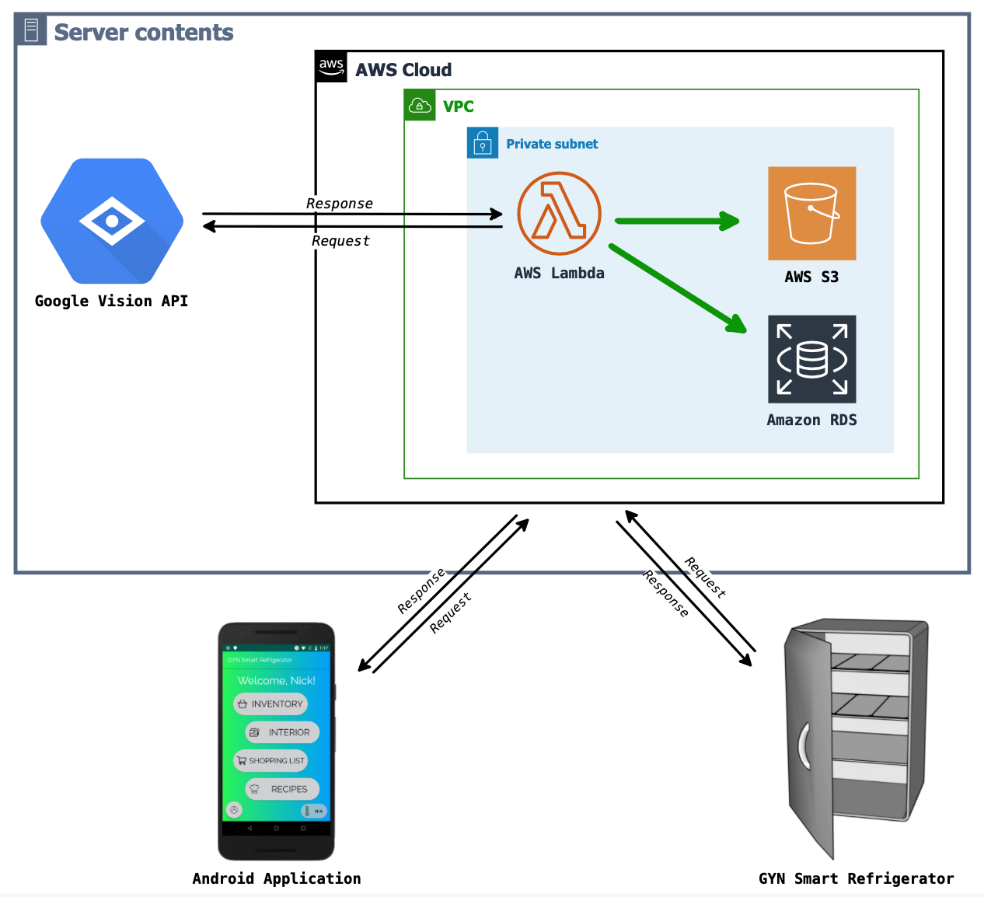
The server used for the creation of this product (Refrigerator + Android App) involved Amazon Web Services (AWS) and Google Cloud. The AWS data centre is comprised of AWS Lambda function, Amazon Simple Storage Service (S3) and Amazon Relational Database Service (RDS) Storage. The Lambda function - written in python - is used to trigger communication between the data centre and the Google Vision API (from Google Cloud). The S3 bucket is used as persistent storage that saves all the pictures taken from the interior of the refrigerator. The RDS (PostgreSQL-based) database, on the other hand, keeps all other information about the entire product.

Figure 5: Server Diagram

### 3.2.3 The Software

The project has two software components, the first being the program that is run on the Raspberry Pi. Firstly, the Pi runs on the Raspbian operating system, a 32-bit Linux distribution. This program was coded as a Python 3 script set to run automatically upon powering on the Pi. It uses various libraries associated to the services such as the HX711, and the Amazon server connection.

The application was developed for Android devices using the Android Studio IDE using the Java programming language. The application was designed with every type of device in mind, this way, it *should* be able to run on any Android release in the past 10 years and any device, although it should be noted that very old devices may struggle to run the application. Apart from the libraries included in Android Studio, the application utilizes the JDBC driver. JDBC stands for Java Database Connectivity, it is the main method for connecting the application to the server. As an added level of security, the application is not in constant communication with the sever, instead it is setup to only contact the server when specific actions are taken (such as refreshing weights, searching for recipes, etc.). The application also uses SQLite as a local database to store the information from the server, this includes the weights of the scale and other various inventory information, the image from the camera, various recipes, and shopping list items. The local databases are useful for allowing access to certain features of the application, even when not able to connect to the internet (although functionality is limited).

The user interface design of the application was initially realized with the use of the JustInMind software, a prototyping tool used to create mobile application interfaces and wireframes. This design was brought to light with the use of XML stylesheets in Android Studio. XML stylesheets uses the XSL language to transform and format the contents of an XML document and display them to the user. This enables for example the modification of the standard Android button to be given rounded edges, different colors, different style and sizes of fonts, and animations that enhance the user experience.

## 3.3 Problems Encountered

### 3.3.1 Major Problems

First and foremost, a major problem has affected the ability to fully complete the project school-wide. COVID-19 has caused the closure of the university, halting virtually all work and preventing any further progress. At the time of the closure, partial completion of the project had already been reported. As it stands, the refrigerator can be considered as a working prototype, the main components function properly and a full cycle of data transfer is possible, for example collecting weight from the scales, sending the weights to the server, and acquiring that data on the application.

|  |  |
| --- | --- |
| ID | Missing Features Due to University Closure |
| 1 | Hardware - Hall effect sensors to determine door status (open/closed). |
| 2 | Hardware - Temperature sensor to monitor interior temperature from the application. |
| 3 | Hardware - LED Strip to light up the refrigerator when an image is being captured. |
| 4 | Hardware - Buzzer to notify the user if the door has been opened for too long. |
| 5 | Hardware - 3D print mounting brackets for load cells. |
| 6 | Hardware - Production, testing, and validation of PCB. |
| 7 | Software - Different means of searching for recipes, ie Calories, Prep time… |
| 8 | Software - Open/closed door detection to capture an image |

On the other hand, some features are not included in the prototype. Due to social distancing and the inability to enter the university, some components of the project could not be acquired. Luckily these components are not an essential to the project and are mainly being used for features that already exist in smart refrigerators. Furthermore, as these components cannot be added to the refrigerator, the code for them has not been implemented either.

Table 4: Missing Features

### 3.3.2 Minor Problems

Of course, during development, several *smaller* problems will arise. In the case of this project, the problems that gave the most trouble have been noted.

Case 1: A couple of problems came from configuring the scale properly. After constructing each scale, they must be calibrated (through the code), through a few steps, a scale is given a reference unit which is used thereafter to ensure the more accurate weight measurements. The calibration process assumes the scale configuration is not altered, any modification might throw off the reference unit and cause inaccurate results. Furthermore, the load cell wires are extremely prone to electrical interference when testing outside of the refrigerator. The wires are relatively hidden while the scales are set in the refrigerator and generally do not show any signs of interference.

Case 2: On the software side, when using the Google Cloud Vision API for image processing, it has been observed that the results are not always accurate. For instance, if there is a variety of fruit in an image (about 4-5), the API will return “fruit” instead of each individual type of fruit. Unfortunately, due to the Vision API being a third-party service, this cannot be properly fixed. Additionally, assuming that a user puts a non-food item in the fridge, the API will recognize it as an item found in the refrigerator and count it towards the refrigerator inventory.

## 3.4 Solutions Proposed

As it has been made aware to every team, project completion may no longer be possible due to COVID-19. Instead of performing the final demos, additional submission requirements were assigned to each team. Also, the submission deadline was pushed back to accommodate these changes. Overall, the transition was handled well.

Case 1: A temporary solution for the scale calibration was to set the reference unit to the default value and taring the scale at the beginning of the code (whenever the refrigerator is powered on). A tare sets the scale weight to zero regardless of how much weight is present, in the case of the refrigerator, it would be necessary to remove all items from the scale before powering on the device. As an essential appliance, refrigerators are not turned off very often, it was determined that this solution is viable assuming the refrigerator was not powered off often.

Case 2: The proposed solution for the API identifying non-food items is to add a large filter of various foods, processing the list of items found and removing any non-food items before sending the data to the user. This method is not the most efficient as it can be slow but is effective in a pinch. Ideally, the Vision API would offer a food filter for the image processing, but this is out of the team’s control.

# 4. Results

## 4.1 Test Cases

The following test cases demonstrate some of the most vital process of the system. These cases ensure the proper overall functionality. If any of these cases fail, then it is guaranteed that the user experience will not be ideal, and problems may arise.

4.1.1: Raspberry Pi updates our data centre

**Conducted by:** All team members

**Scope:** The main purpose of this test case is to validate the fact that the measurements of the food quantities from the load cells, together with the picture taken with the camera are well received in our database and s3 bucket, respectively – collectively referred to as AWS resources.

**Test Procedure:**

* As part of the preliminary step, the Pi must be booted, connected to a hotspot (Wi-Fi or LAN), and configured with proper credentials.
* Next, a script (loaded on the Pi) is then run in a python environment. In this script, functions to connect to the server, carry out the measurements, capture the picture, and upload the data to AWS (in that order).

**Inputs:**

* Values recorded by the sensors (at normal operating conditions).

**Expected outputs:**

* The connection attempt to the AWS is successful.
* The measurements with the scale (system of load cells) are made and a picture is taken.
* Through the execution of some predefined instructions, the information from the previous step is uploaded or updated accordingly.
* Finally, the connection between the Pi and the server is terminated.

**Actual outputs:** As Expected.

**Conclusion:**

Pass 👍🏼

4.1.2: The smartphone receives updates from the data centre

**Conducted by:** Nicholas & Georgiy

**Scope:** The main purpose of this test case is to validate the fact that the information about the content of our smart refrigerator is obtained on request. The request, in this case, represents the act of opening a page (or an activity) in the app that requires data from the server.

**Test Procedure:**

* As part of the preliminary step, the package installed on the phone most be configured (with proper credentials to have access to AWS) and the smartphone must be provided with an internet connection. Finally, log in with the Device ID and password provided with the packaging.
* Next, navigate through the application.

**Inputs:** None

**Expected outputs:**

* When the inventory page is opened, the data from the scale is fetched from the database and displayed in the predefined text areas, accordingly. Default values are displayed in case the initial fetch is unsuccessful. Previously fetched data is displayed in case new updates were unsuccessful.
* When the interior page is opened, the picture (previously taken with the camera) from the s3 bucket (data storage for file objects in AWS) will be loaded together with correct information about items found on the picture. If nothing is to be found, a dummy picture is displayed, and no information about items found is displayed.
* When the recipe page is opened, the list of recipes (per search query) fetched from the database is obtained and displayed in a list view. Once clicked upon, all the information about each recipe is available within the application. The list view is empty in case the search is unsuccessful.

**\*** *For each of the points above, a connection is established before and terminated after, data is fetched, between the smartphone and the server.*

**Actual outputs:** As Expected.

**Conclusion:**

Pass 👍🏼

4.1.3: Only authorized devices can interact with the data centre

**Conducted by:** All team members

**Scope:** The main purpose of this test case is to validate the fact that the data centre is not accessible by any device even if they have the proper credentials. This security measure was added in case the credentials happened to have been exposed.

**Test Procedure:**

* As part of the preliminary step, the device on which the app is running, the Pi controlling the refrigerator, or any other devices should be connected to the internet. Any of the aforementioned devices should be configured with proper credentials associated with IAM roles attached to the security group in AWS.
* Add the IPs of some devices to the inbound rules in the VPC in our predefined security group.

**Inputs:**

* IP addresses of known devices.

**Expected outputs:**

* For all devices whose IP addresses are part of the security group, any attempt to access (connect) assigned resources in AWS will be granted since they will be considered as known devices.
* On the contrary, all the other devices whose IP addresses are not part of the security group are denied access upon request/attempt to connect.

*\* No access implies no data is read or written into our server.*

**Actual outputs:** As Expected.

**Conclusion:**

Pass 👍🏼

4.1.4: Vision API trigger/process

**Conducted by:** Yvann

**Scope:** The main purpose of this test case is to validate the fact that once an image (from the refrigerator) is received in AWS, a series of operations are executed to identify the items found on that picture.

**Test Procedure:**

* Lambda function on the server-side is properly implemented and configured with suitable credentials from Google Cloud to have access to the Vision API.

**Inputs:**

* Upload an image to s3 bucket or do an API call using a designated URL.

**Expected outputs:**

* Once, the input is made, the lambda function is triggered.
* Within the python environment, the Vision API and its function are used to perform some annotation on the image thereby producing a JSON output of containing its findings.
* This output is then filtered and reduced to useful information.
* The final result is used to update the database.

**Actual outputs:** As Expected.

**Conclusion:**

Pass 👍🏼

## 4.2 Minor Test Cases

The following test cases demonstrate some less detailed scenarios and easy to perform operations, notably black box testing, meaning that for a given input, there is an expected output, neglected the methods for obtaining said output. Each test was conducted several times to ensure proper functionality.

4.2.1: Changing the name of a partition (scale)

While in the inventory management page, hold down on a partition to display the name change prompt.

Input: New name

Expected Output: New Name is updated

Result: As expected

4.2.2 Attempting to exceed shopping list limits does not work (0 to 99)

While adding a new item to the shopping list, the user can define the amount of that item (from 0 to 99), it should not be possible to go below 0 or above 99.

Input: Shopping list item amount

Expected Output: Any number between 0 and 99

Result: As expected

4.2.3 Bookmarking a recipe saves the recipe locally

When viewing a recipe, it is possible to bookmark it. Bookmarking a recipe stores the data in a local database which should grant the user access even without a connection.

Input: Bookmark any recipe

Expected Output: Recipe accessible without internet access

Result: As expected

4.2.4 Adding missing items from a recipe to shopping list

When viewing a recipe, it is possible to add any missing items from the inventory to the shopping list.

Input: Use button for adding missing items

Expected Output: Any item missing from inventory, shopping list, extra ingredients are added to the shopping list.

Result: As expected 90% of the time. For some reason, from time to time an item will still be added to the shopping list even if it exists in the refrigerators inventory.

4.2.5 Adding or removing an item from a scale reports a weight ±5% to the actual weight

Live measurements can be viewed directly from interfacing with the Raspberry Pi. Adding and removing items changes the weight recorded by the scale.

Input: Add or remove an item from the scale

Expected output: The updated weight is within a 5% error of the actual weight

Results: As expected 80% of the time. The failed tests all occurred when placing items in non ideal locations (such as the edge of the scale). A successful test example is given below. The scale usually records a weight a little higher than the actual weight, likely due to the reference unit being default.

Figure 5: Weight Test

## 4.3 Additional Notes on Validation

While testing the scales, they were put under some validation techniques as well. The load cells were hooked up to a digital multimeter and had various measurements taken such as resistance and voltage. Each load cell is expected to have the same resistance. Also, the voltage shifts when applying pressure to the load cells, each individual load cell shifts the voltage slightly, and when all the load cells are affected at the same time, the voltage shifts much more. Unfortunately, the data sheet for these results were misplaced. The load cells were the only component that were tested this way, so it is not a total loss.

# 5. Conclusion

To conclude, the project was partially completed, while some features could not be included, the end result was still a functioning prototype. The main features were implemented successfully, including the weight monitoring and the Android application, being able to observe a change in weight from the application itself as well as accessing other data such as the images from the camera and the recipe database.

## 5.1 Lessons Learned

Overall, this project has taught some very important lessons in the proposal, design and development processes of a product. It has become clear why new products sometimes take a very long time to reach the market, taking things into consideration such as design modifications and testing and validation that may impact the development. Additionally, it was a very valuable experience working as a team while developing this project, with the added benefit of the project supervisor overlooking the process and providing constructive feedback. This passed year has been a huge help in taking the step into the real world to begin our careers as engineers.

## 5.2 Closing Remarks

It was an honor to be able to work with such an amazing faculty. A big thank you to Dr. Le Beux, Dr. Goodarzi, Dr. Soleymani. Dr. Fayyaz, and Mr. Rozhdestvenskiy for this opportunity. We will be treasuring this experience for the rest of our lives and greatly appreciate your time and effort throughout the last two semesters.

# 6. References

1 - Lindsey, Nicole. “Smart Devices Leaking Data To Tech Giants Raises New IoT Privacy Issues.” *CPO Magazine*, 13 Apr. 2020, www.cpomagazine.com/data-privacy/smart-devices-leaking-data-to-tech-giants-raises-new-iot-privacy-issues/.

2 - Gitlin, Jonathan M. “Sonic the Hedgehog's Movie Look Is Fixed Following Fan Outcry.” *Ars Technica*, 12 Nov. 2019, arstechnica.com/gaming/2019/11/sonic-the-hedgehogs-movie-look-is-fixed-following-fan-outcry/.

3 - Wroclawski, Daniel. “Haier Recalls Top-Freezer Refrigerators.” *Consumer Reports*, [www.consumerreports.org/recalls/haier-recalls-top-freezer-refrigerators-fire-hazard/](http://www.consumerreports.org/recalls/haier-recalls-top-freezer-refrigerators-fire-hazard/).

4 - Hodgson, Philip. *Tips for Writing User Manuals*, [www.userfocus.co.uk/articles/usermanuals.html](http://www.userfocus.co.uk/articles/usermanuals.html).

5 - Office of the Privacy Commissioner of Canada. “Summary of Privacy Laws in Canada.” *Office of the Privacy Commissioner of Canada*, 31 Jan. 2018, [www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/02\_05\_d\_15/](http://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/02_05_d_15/).

6 - Instructables. “Arduino Bathroom Scale With 50 Kg Load Cells and HX711 Amplifier.” *Instructables*, Instructables, 7 Apr. 2019, [www.instructables.com/id/Arduino-Bathroom-Scale-With-50-Kg-Load-Cells-and-H/](http://www.instructables.com/id/Arduino-Bathroom-Scale-With-50-Kg-Load-Cells-and-H/).

7 - “IoT Risks: How Smart Is Having a Smart Fridge?” *Le VPN*, 26 July 2019, [www.le-vpn.com/iot-risks/](http://www.le-vpn.com/iot-risks/).

8 - *Bitbucket*, bitbucket.org/MattHawkinsUK/rpispy-misc/raw/master/python/hall.py.

9 - gandalf15. “gandalf15/HX711.” *GitHub*, github.com/gandalf15/HX711/blob/master/HX711\_Python3/hx711.py.

10 - Tatobari. “Tatobari/hx711py.” *GitHub*, 13 Aug. 2019, github.com/tatobari/hx711py.

# 7. Appendix

## 7.1 User Manual

Included on the next page.

Version 1.1 • 05 April 2020

Nicholas Pollender

Georgiy Araslanov

Gneykou Kengne Yvann Monny

GYN Ultra Smart Refrigerator

User Manual

Table of Contents

[1. Introduction 26](#_Toc37920710)

[2. Key Features 28](#_Toc37920711)

[2.1 Food Scales 28](#_Toc37920712)

[2.2 Internal Camera for Image Processing 28](#_Toc37920713)

[2.3 Raspberry Pi Micro Computer 28](#_Toc37920714)

[2.4 GYN Smartphone Application 28](#_Toc37920715)

[3. Getting Started 29](#_Toc37920716)

[3.1 Refrigerator Setup 29](#_Toc37920717)

[3.1.1 Device Identification 29](#_Toc37920718)

[3.1.2 Powering On the System 29](#_Toc37920719)

[3.1.3 Scales 29](#_Toc37920720)

[3.1.4 Powering Off the System 29](#_Toc37920721)

[3.2 Smart Device Application Setup 29](#_Toc37920722)

[3.2.1 Installing the Application 29](#_Toc37920723)

[3.2.2 Logging In 30](#_Toc37920724)

[3.3 Application Navigation 30](#_Toc37920725)

[3.3.1 Login Page 30](#_Toc37920726)

[3.3.2 Home Page 31](#_Toc37920727)

[3.3.3 Inventory Management 31](#_Toc37920728)

[3.3.4 Extra Inventory 32](#_Toc37920729)

[3.3.5 Interior View 32](#_Toc37920730)

[3.3.6 Shopping List 33](#_Toc37920731)

[3.3.7 Recipe Searching 33](#_Toc37920732)

[3.3.8 Search Results and Recipe Viewer 34](#_Toc37920733)

[3.3.9 User Settings 34](#_Toc37920734)

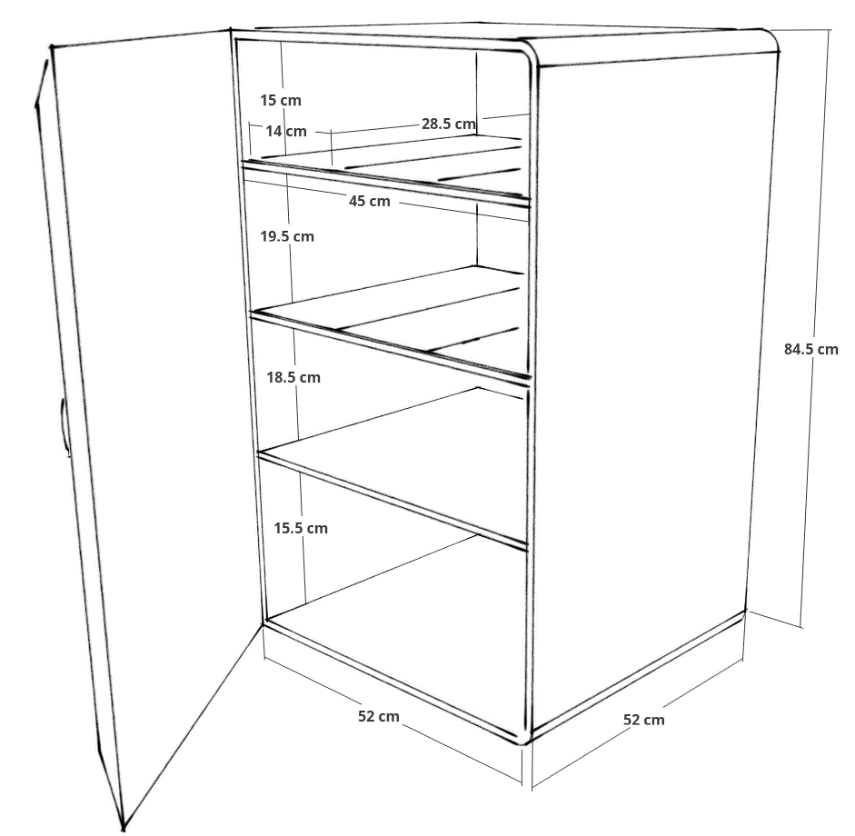
[4. Health and Safety Awareness 35](#_Toc37920735)

[4.1 Health Considerations 35](#_Toc37920736)

[4.2 Cleaning and Maintenance 35](#_Toc37920737)

This user manual is intended for users who have purchased the GYN smart refrigerator. The goal is to introduce the system to the user and teach them how to use the hardware and software. This guide assumes you have a basic understanding of home appliances and smart devices.

# 1. Introduction

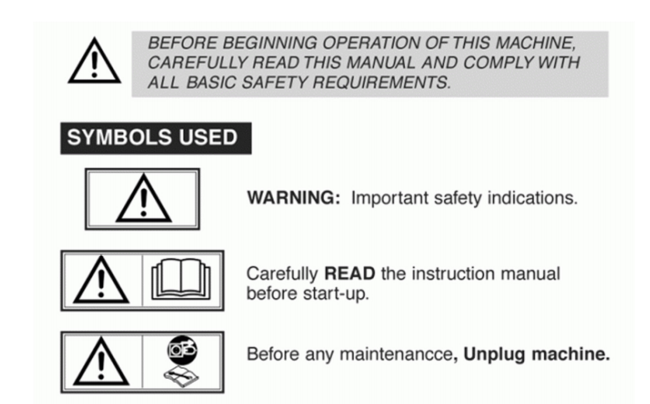


Thank you for selecting the GYN Smart Refrigerator. This appliance is a different approach to smart refrigerator technology designed to promote organization and keep up healthy eating habits.

The GYN Smart Refrigerator is powered by a standard three prong outlet via the power cord connected to the back of the refrigerator.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Requirement/Parameter | Test Conditions | Values | | | Units |
| Min | Typ | Max |
| 1 | Refrigerator Dimensions (HxWxD) | - | - | 52x52x84.5 | - | cm |
| 2 | Maximum Weight on Load Cells | Normal | 0 | 2 | 10 | kg |
| 3 | Operating Temperature | Normal | - | 4 | - | ⁰C |
| 4 | Wi-Fi Connection Range | Normal | - | 5 | 90 | m |
| 5 | Refrigerator Operating Voltage | Normal | - | 120 | - | V |

Table 1: Design Specifications



Please refrain from moving or modifying any device or sensor in the refrigerator. It may cause irreparable damage and void the appliances warranty.

# 2. Key Features

The GYN Smart Refrigerator is equipped with many features to make your life easier. Among these features are those that define this brand and product, features which are not typically seen in the common refrigerator.

## 2.1 Food Scales

Each GYN Smart Refrigerator has six pre-installed food scales. The scales are placed on the two top shelves, three scales on each. Do not attempt to displace the shelves as any significant movement may disconnect or break their wiring.

The scales are used to measure the weight of any item you wish to keep track of. The items on the scale are determined by the user, it can be a specific food, a beverage, a category of food, it can even be office supplies if you so chose.

## 2.2 Internal Camera for Image Processing

The bottom shelf is not left out of the equation, instead of scales this shelf is monitored by a powerful camera that periodically captures images of its contents. The images are processed by Google Cloud Vision, this application can identify objects in an image, such as the food stored in your refrigerator. Do not attempt to displace the camera, it positioned ideally to capture the whole shelf.

It should be noted that inaccuracies may occur, and objects found in an image may not always match the exact contents of the refrigerator. Furthermore, if there are any privacy concerns, this feature may be disabled at any time.

## 2.3 Raspberry Pi Micro Computer

Attached to the back of the refrigerator is the computer that controls all the devices within the appliance, it is the brains of the operation. The computer processes the refrigerators data and sends it to GYN’s secure server which is then transmitted to your smart phone so you can see the latest information about your smart refrigerator.

## 2.4 GYN Smartphone Application

Of course, the key features cannot be fully experienced without the main component for interacting with the system. The GYN smart refrigerator has a companion Android application (iOS support in the future). The app is very user friendly; it features large buttons, easy to read text, simple navigation and appealing visuals.

# 3. Getting Started

## 3.1 Refrigerator Setup

Before having full access to your new refrigerator experience, please follow this quick guide. This will ensure the correct functionality of the appliance. If any step is skipped or only partially completed, you may experience unintended problems.

### 3.1.1 Device Identification

Before powering the device, it is strongly recommended to take note of the device ID and PIN which can be located on the back of the refrigerator where the Raspberry Pi micro computer is installed. These credentials will be used to login to the application and gain access to the refrigerators data. Please do not share these credentials with any unauthorized user. The information should look like this:

|  |  |
| --- | --- |
| **DEVICE ID** | GYN\_A1XXXXXXXX |
| **PIN** | XXXXXXXXXXXXXXXX |

### 3.1.2 Powering On the System

The refrigerator is powered by the three-prong plug on the back of the appliance. Simply connect the plug to a corresponding three-prong outlet wherever applicable. Please allow at least 2 minutes for the computer to boot up, this will ensure that the software is functioning properly.

### 3.1.3 Scales

If the refrigerator was off and is being powered back on, either for the first time or after a power outage, it is strongly recommended that any items placed on the scale are removed before powering the appliance to ensure that the scales are reporting accurate data.

### 3.1.4 Powering Off the System

To power off the refrigerator, you need only to unplug it. Refer to (3.1.3) when powering the system back on.

## 3.2 Smart Device Application Setup

Please make sure you have access to an Android device before continuing this guide, iOS devices are not currently supported. Apologies for any inconveniences this may cause.

### 3.2.1 Installing the Application

Navigate to your device’s application store, for example, most Android devices use Play Store. Search for ‘GYN Smart Fridge’ and select the first result (it should be a green icon with GYN written in the center) and press install on the page that follows. Alternatively, you can navigate to the GYN website and install the application from the homepage at <https://www.smartGYN.com/>. The website is still under construction, however. Please read the permission requirements carefully before installing, this prompt explains what sort of information is being gathered from your smart device.

### 3.2.2 Logging In

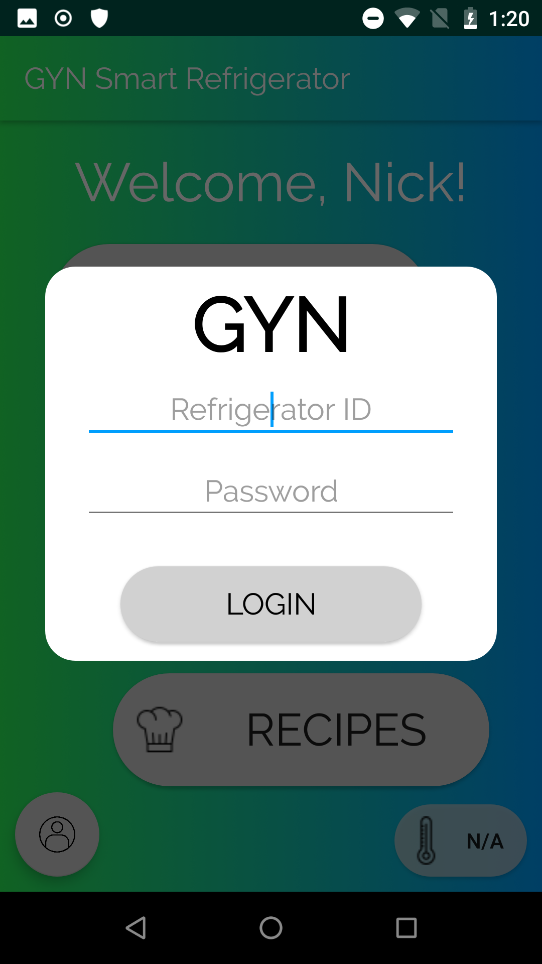
After installing the application and launching it for first time, a login prompt will appear preventing access to the application. Before continuing, make sure your device has a reliable connection to the internet (Wi-Fi or 4G), otherwise you will not be able to login. The login credentials are the device ID and PIN that appear on the back of the refrigerator (see 3.1.1). If you are unable to login, please ensure that you are using the correct ID and PIN, if you are still unable to login, it is possible that the servers a down for maintenance. In that case, please wait 5 minutes and then try again.

## 3.3 Application Navigation

The following subsections will provide instruction for using the application, each subsection is assigned to a different feature of the application. Read each of them carefully and refer to the figures.

### 3.3.1 Login Page

Please refer to (3.2.2) for details on login credentials.



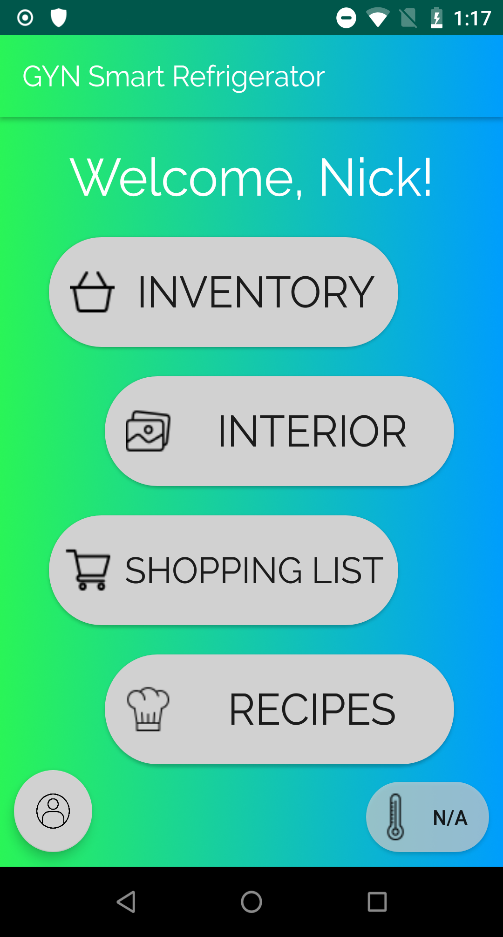
1. The text prompt marked ‘Refrigerator ID’ is where you type in the DEVICE ID.

2. The text prompt marked ‘Password’ is where you type in the PIN.

3. After typing in both credentials, press the ‘LOGIN’ button.

### 3.3.2 Home Page

Upon logging in, you will be greeted with this page.



1. Navigates to the inventory management page, for viewing weight of each scale.

2. Navigates to the interior view page, for viewing the image captured by the camera.

3. Navigates to the shopping list.

4. Navigates to the recipe searching page, you can search for new recipes here or browse saved recipes.

5. Navigates to the user settings, logout, change your name, or delete data here.

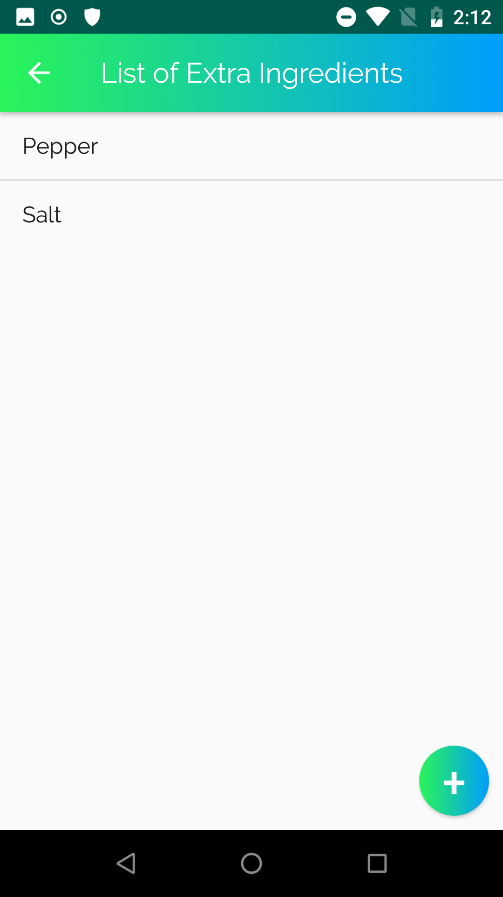
6. Temperature monitor for latest temperature in the refrigerator recorded.

### 3.3.3 Inventory Management

This page grants access to the weights of the scales in the refrigerator. Additionally, you can modify the names of each scale.

1. The letter is associated to a scale. Scale A is found in the top shelf to the left, B in the middle, C on the right. D, E, F are on the middle shelf.
2. The name of the scale and weight is displayed in this box. Holding down on the box will allow you to change the name of the scale.
3. This button refreshes the weight of the scales.
4. Navigates to the extra inventory page, for keeping track of items that cannot necessarily be monitored by weight.

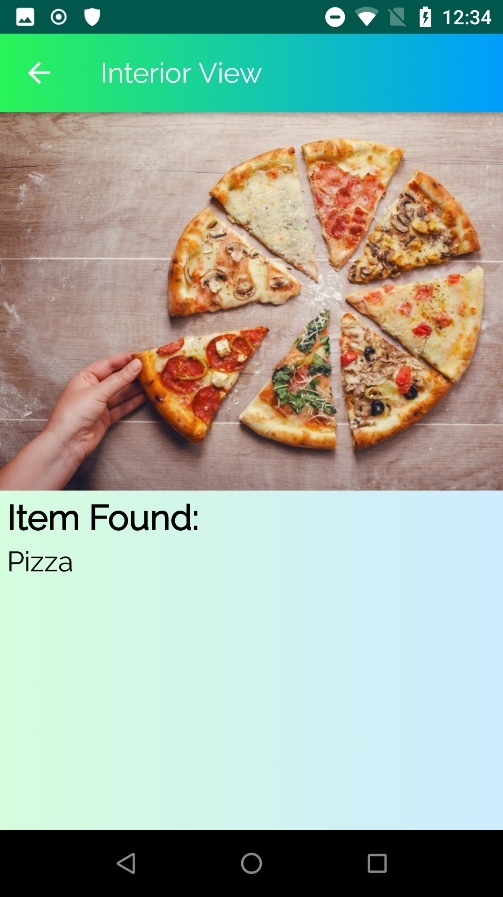
### 3.3.4 Extra Inventory

The extra inventory items are food items that you do not wish to weight or that are not typically stored in a refrigerator. This can be common ingredients such as flour or sugar, salt and pepper, etc.

1. An extra item, if you wish to delete it, simply hold down on it.

2. Press this to add a new item to the list.

### 3.3.5 Interior View

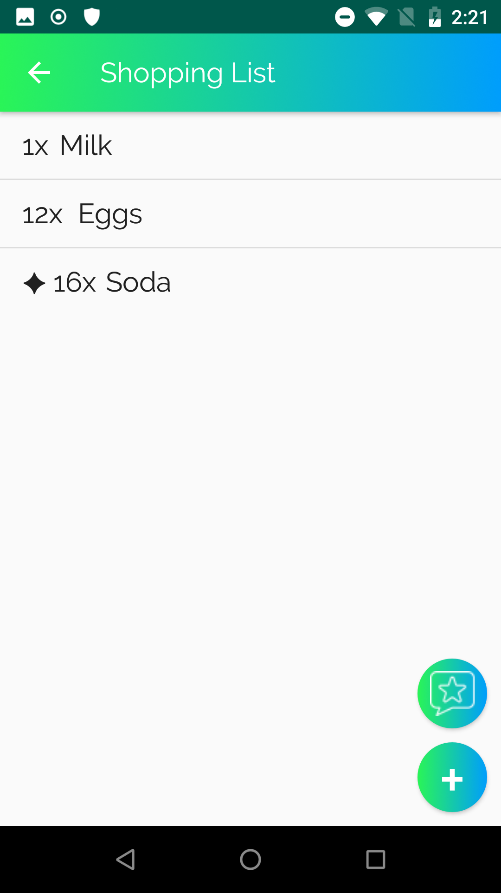


This page displays the most recent image captured by the refrigerator’s camera, and the items found in the image.

1. The image captured by the camera.

2. The list of items found in the image; this may not be entirely accurate sometimes due to the limitations of the third-party service used by this product.

### 3.3.6 Shopping List

The shopping list is your personal shopping assistant.

1. An item on the list has a name and a quantity. To delete an item off the list, simply press the item. To edit the quantity of the item, hold down on the item.

2. This is the recommendation button. If you press it, a prompt will appear if any items on the scales are low on inventory. If you wish, you can add those items to your shopping list.

3. Press this to add a new item to the list.

### 3.3.7 Recipe Searching

You can view and search for recipes from here.

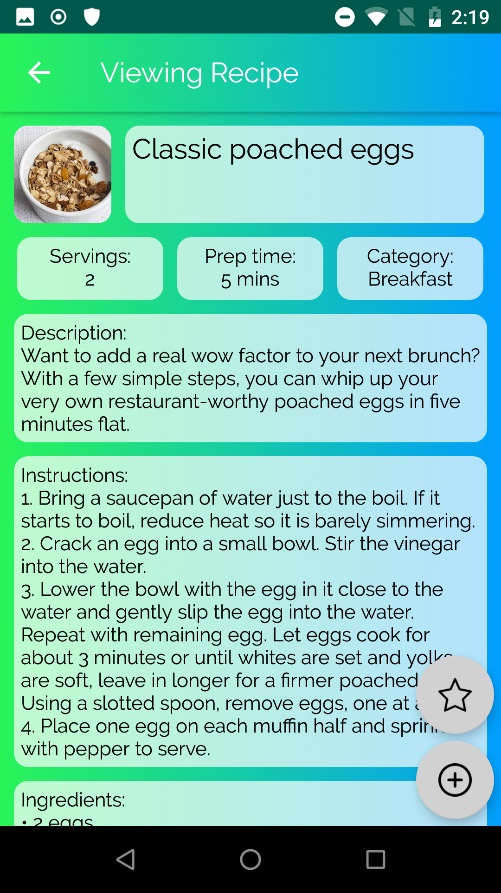
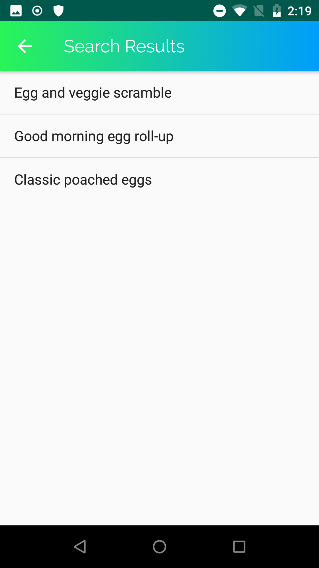
1. Navigates to the bookmarked recipes, you can view any recipe you’ve saved from your searches here.

2. Opens the search window for searching recipes.

3. Enter a name of a dish you wish to search, currently this is the only working search criteria. The other options have not been implemented.

4. Once you’ve entered at least one criterion, you can press the search button.

### 3.3.8 Search Results and Recipe Viewer

From the search results (or the bookmarked recipes) you can select a recipe and it will be displayed.

1. A recipe from the search results, pressing this will open the recipe viewer.

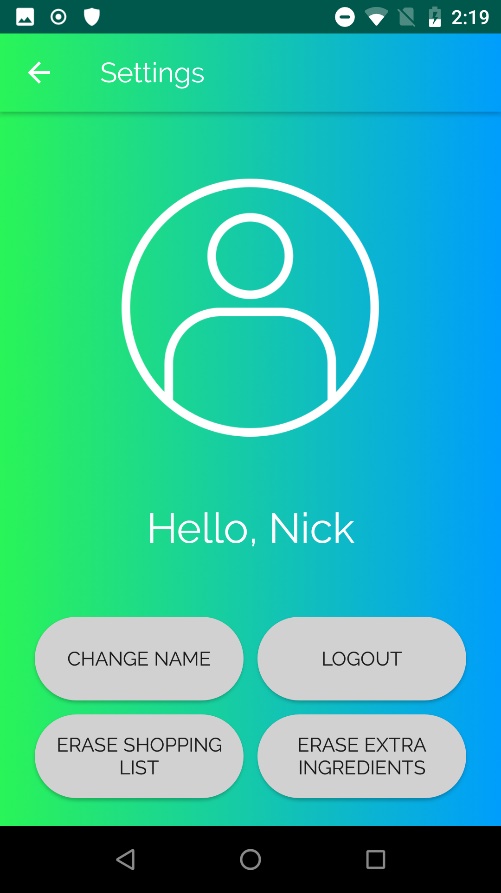
2. An image of the dish, pressing the image will download a pdf of the original recipe reference (if applicable).

3. The various information of the recipe, such as servings, prep time, steps, ingredients, etc.

4. This button bookmarks the recipe and saves it to your device. (only appears from the search results)

5. This button will search your inventory and shopping list and add any missing ingredients to your shopping list.

### 3.3.9 User Settings

You can change some user settings as well as delete data from here and logging out.

1. Pressing this button will allow you to change the name associated to the device.

2. Pressing this button will log out of the device and bring back the login prompt (see 3.3.1).

3. Pressing this button will delete your entire shopping list, be careful as this cannot be undone.

4. Pressing this button will delete your entire extra ingredients list, be careful as this cannot be undone.

# 4. Health and Safety Awareness

## 4.1 Health Considerations

When installing the fridge, do not place your fingers on the prongs of the plug while plugging it into an outlet to avoid electrocution.

Do not drill any holes in the fridge (or any similar action), this may puncture the refrigerant and release dangerous gasses into the air. This gas should not be inhaled. If you believe you have inhaled the gas, you should contact a health care professional immediately.

## 4.2 Cleaning and Maintenance

Do not attempt to perform maintenance on the appliance to avoid any unnecessary damage. If the system requires any kind of repair, please contact our support and a GYN technician will be made available as soon as possible.

When cleaning the interior, avoid pouring any cleaning product or liquid on the scales or other interior electrical devices. Additionally, if there is a spill of any sort on the devices, dry the area immediately.

End of User Manual

## 7.2 Technical Manual

Included on the next page.

## 

Version 1.0 • 15 April 2020

## 

GYN Ultra Smart Refrigerator

User Manual

## 

## 

Nicholas Pollender

Georgiy Araslanov

Gneykou Kengne Yvann Monny

Table of Contents

[1. Introduction 39](#_Toc37920092)

2. Features..……………………………………………………………………………………………………………………………….40

[3. Specifications 40](#_Toc37920093)

[4. Part Names and Operation Description 41](#_Toc37920094)

[4.1 Hardware 41](#_Toc37920095)

[4.2 Server 43](#_Toc37920096)

[4.3 Android Application 43](#_Toc37920097)

[5. Schematics, Components’ operation & Troubleshooting 44](#_Toc37920098)

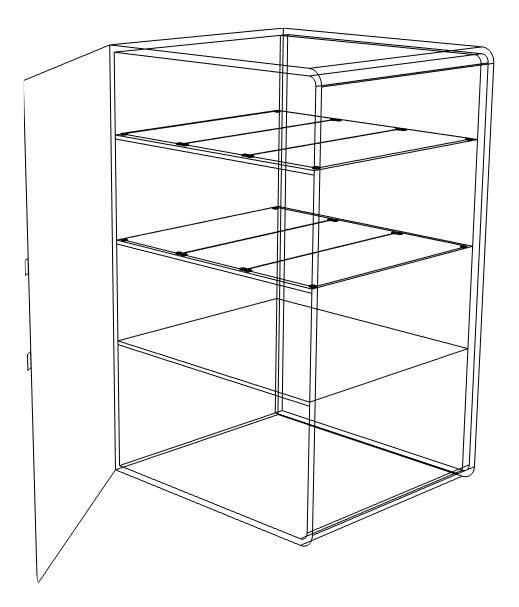
[6. Functions & Routines 45](#_Toc37920099)

[7. Appendix 45](#_Toc37920100)

# 1. Introduction

The aim of the present manual is to illustrate the technical design of the GYN Ultra Smart Refrigerator.

The focal point of the following pages will be the blueprint of the hardware and software of the machine as well as miscellaneous information that is crucial to the comprehension of functionalities of the product in general. This document contains details on part descriptions, schematics, troubleshooting, wiring diagrams, flow diagrams and much more.

  
2. Features   
  
The GYN Ultra Smart Refrigerator is a kitchen appliance designed to store and artificially store cool food and drinks. The smart features of this device rely on wireless connectivity and electronic devices like a computer, a camera and sensors to visually inform its users through a smartphone application about the current quantities of foods placed inside while promoting organized storage and suggesting recipes based on the food quantities monitored inside.

The refrigerator’s top two shelves are partitioned into multiple sections and each section is clearly delimited with its own glass area with four load cells between the shelf and the glass. The load cells’ purpose is to read the change in weight when adding or removing items from the shelf. The third shelf is equipped with a high performance camera that periodically captures images of its contents. The images are processed by Google Cloud Vision and that is how the food stored on the shelf is identified. Ultimately, all the collected information is sent to a server through which the user’s mobile application will collect its data.

# 3. Specifications

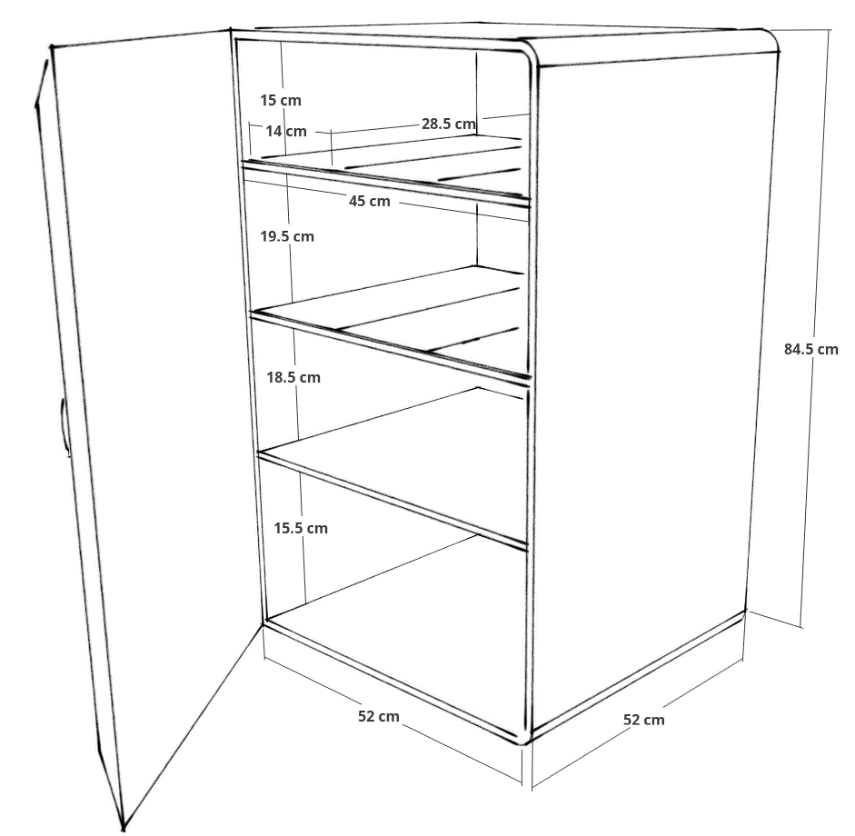
The figure below contains information on measurements of various major components inside and outside of the refrigerator. The table below contains some specifications on the refrigerator’s major requirement/ parameters.

Figure 1: Refrigerator Dimensions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Requirement/Parameter | Test Conditions | Values | | | Units |
| Min | Typ | Max |
| 1 | Refrigerator Dimensions (HxWxD) | - | - | 84.5x52x52 | - | cm |
| 2 | Maximum Weight on Load Cells | Normal | 0 | 2 | 10 | kg |
| 3 | Operating Temperature | Normal | - | 4 | - | ⁰C |
| 4 | Wi-Fi Connection Range | Normal | - | 5 | 90 | m |
| 5 | Refrigerator Operating Voltage | Normal | - | 120 | - | V |

Table 1: Design Specifications

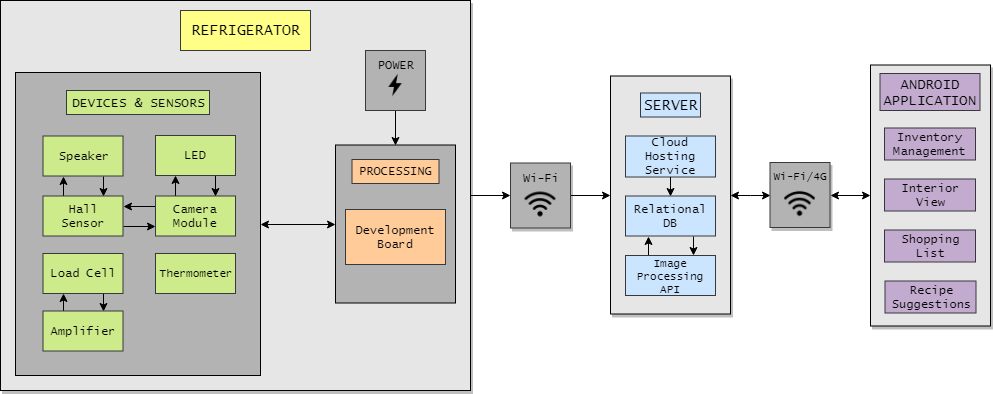
4. Part Names and Operation Description  
  
The GYN Ultra Smart Refrigerator’s system is composed out of three main elements: the hardware, a server and an android application. The figure below shows a block diagram that illustrates the whole system, its main components their sub-components and how they all communicate between each other.

Figure 2: System Block Diagram

## 4.1 Hardware

The hardware part of the product consists of a refrigerator with several devices, sensors and a processing unit inside.

To monitor the food quantities, six food scales made with load cells and an amplifier are pre-installed internally. The scales are placed on the two top shelves, three scales on each. The bottom shelf is monitored by a camera that periodically captures images of its contents. The camera is also connected with an LED light strip that illuminates the inside of the fridge in order to capture a clear picture. The thermometer has a single purpose: to measure the interior temperature of the refrigerator to ensure that the refrigerator is operating at the right temperature. The Hall Effect sensor is placed near the door to detect whenever the door is left open and when it happens; a loud sound is emitted by a speaker to warn the user. All the elements are connected to a processing unit, a Raspberry Pi that is connected to a power source.

The precise location of every part is illustrated in the following two figures located on the next page.

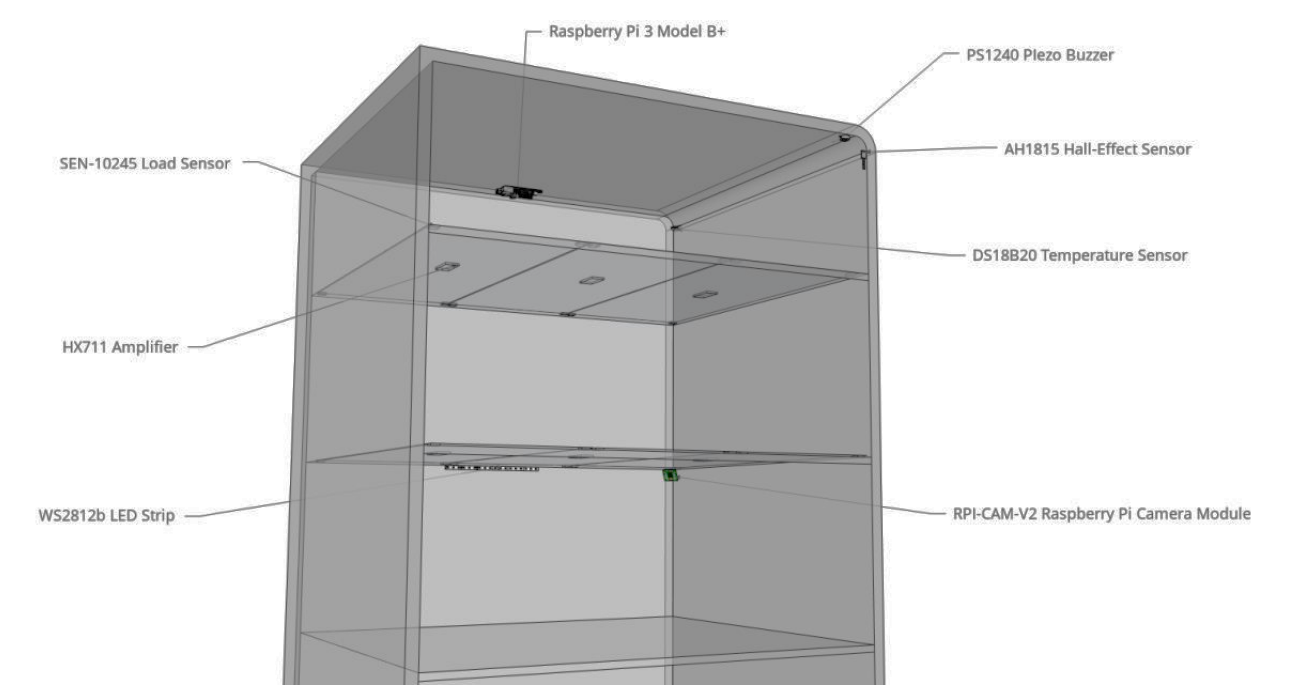


Figure 3: Hardware Configuration of the GYN Ultra Smart Refrigerator

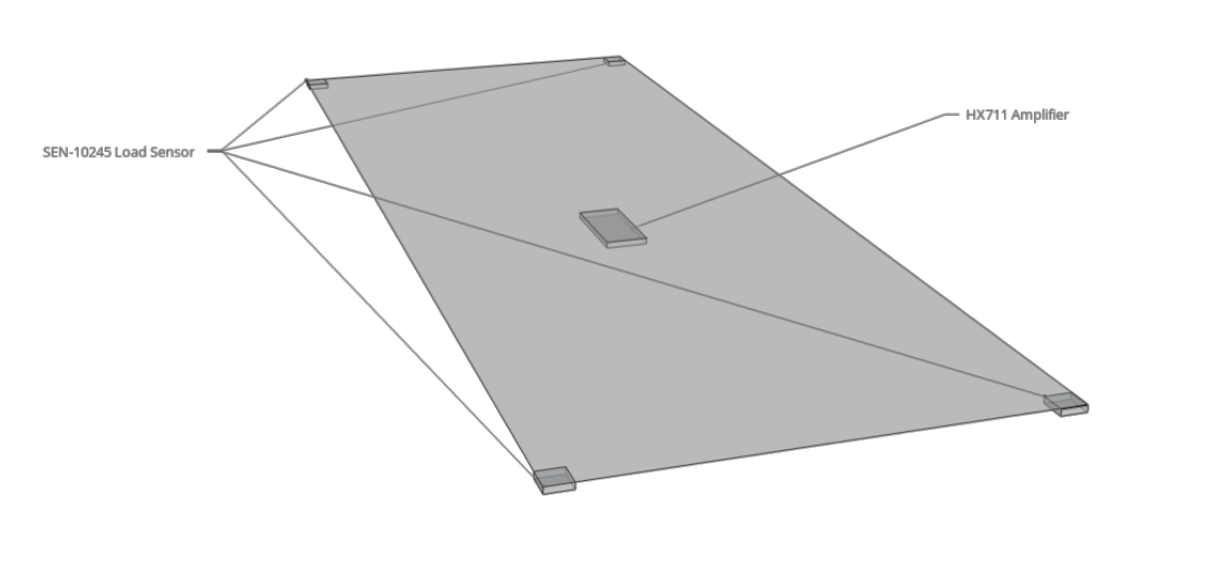


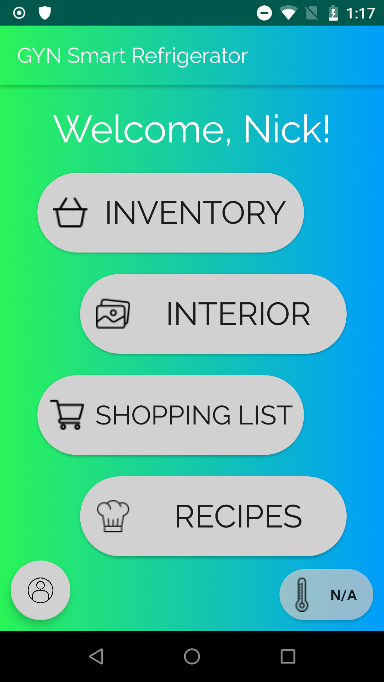
Figure 4: Hardware Configuration

## 4.2 Server

The refrigerator will use its wireless capabilities to get access to internet in order to communicate with a server. The precise services used by this product are Amazon Web Services (AWS) and Google Cloud. The server will be used to transmit data from the development board to the application through an online database.

The AWS data center consists of AWS Lambda function, Amazon Simple Storage Service (S3) and Amazon Relational Database Service (RDS) Storage. The Lambda function - written in python - is used to trigger communication between the data center and the Google Vision API that is used for image processing. The S3 bucket is used as persistent storage that we used to save all the pictures taken from the interior of the refrigerator. The RDS (PostgreSQL-based) database, on the other hand, keeps all other information about our entire product. When the refrigerator sends a request to AWS to establish a connection, the fridge sends a request to store an image inside the S3 and another request to store the measurements in RDS. As soon as an image is uploaded to the S3 bucket, the Lambda function is triggered. Upon execution, it sends a request to Google Cloud through the Vision API to annotate the image with the desired information.

## 4.3 Android Application

  
The software part of the product is a smartphone application intended for the user of the GYN Ultra Smart Refrigerator. The application was designed for Android devices and is the main way for the user to communicate with the fridge.

The main features of the application are the following: Authentication, which is a basic security measure that secures the user’s data from any unauthorized users. Inventory management, a window where a user can view the data of each partition, the weights recorded of each scale in the refrigerator and the partition name which can be modified. Interior view, which shows to the user the latest image that was captures inside the refrigerator. Shopping list, that provides the user with a modifiable list of groceries. And Recipe Suggestions, a feature that allows the user to browse for healthy recipes that are located in a database.

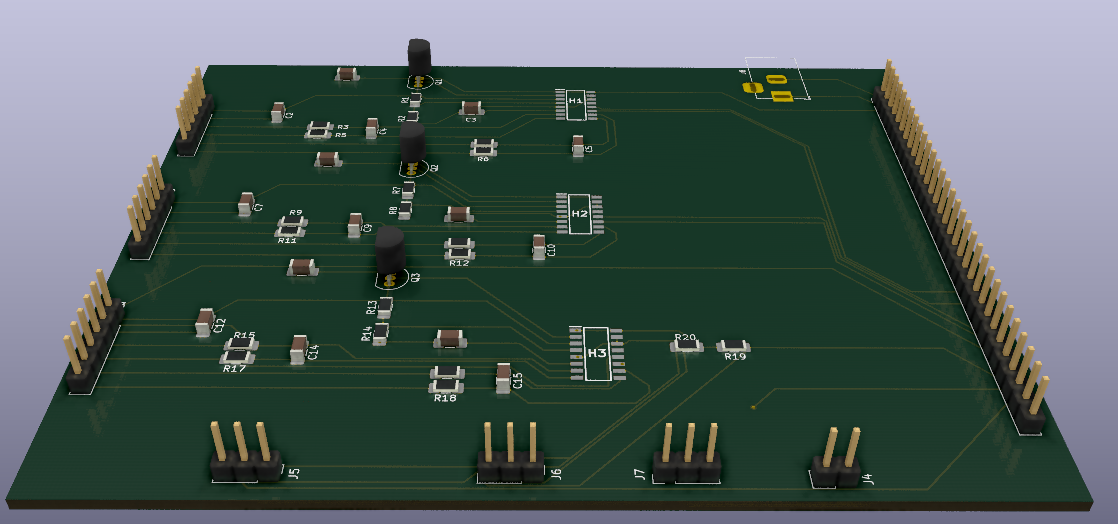
Figure 5: GYN Ultra Smart Refrigerator Companion App

# 5. Schematics, Components’ operation & Troubleshooting

The GYN Ultra Smart Fridge Hardware (see Figure 7 Ultra Smart Fridge Overall System Level Wiring Diagram in the Appendix) is composed of three main parts: the devices/sensors, the PCB and the Raspberry Pi 3B+.

As previously mentioned and as it is represented in the system diagram, the main devices and sensors are the Load Cells (represented as Wheatstone Bridge in the Overall Wiring Diagram, see Figure 9 Wheatstone Bridge Detailed Wiring Diagram in the Appendix for a detailed schematic of how the 4 load cells form a Wheatstone Bridge), Piezo buzzer, Hall Effect sensor, a thermometer and 4 LED lights. Those devices are connected to the PCB with female-to-female terminated wires. For troubleshooting purposes, if any of the devices malfunction, a replacement with the same Manufacturer P/N (see Table 2 Ultra Smart Fridge Overall System Level Bill of Material in the Appendix) may be installed following the guidelines as indicated in the schematics. The exact locations of the devices and sensors inside the fridge are all indicated in Figure 3 on Section 4.1 of this manual.

The PCB (see Figure 8 Ultra Smart Fridge PCB Level Wiring Diagram, Figure 10 PCB Layout and Table 3 PCB Bill of Material in Appendix), as illustrated in Figure 6 below, is made of nine connectors labeled from J1 to J9. The first eight connectors (J1-J8) are male header pins connectors installed to connect the devices and sensors mentioned in the previous paragraph (J1-J7) and the Raspberry Pi 3B+ (J8). The ninth connector (J9) is a barrel jack connector that powers the whole system as a 5V DC power source. The main purpose of the PCB is to provide power to the devices connected to it, link their inputs/outputs with the Raspberry Pi 3B+ and to convert and amplify the signal incoming from the load cells Wheatstone bridges.

  
Figure 6: 3D View of the PCB

As previously explained, the Raspberry Pi 3B+ serves as a processing unit and it is connected to the PCB with a 31 pin connector. The reason why a 31 pin connector is used when only 13 pins are used in the current design is to facilitate future development in case if additional GPIOs are to be used to connect more sensor/devices. This way, only the PCB board has to be changed/replaced and not the whole wiring/connectors.

Of the 13 pins used in the current version of the design, 10 pins are for used as GPIOs and 3 pins are for the 5V, 3,3V and Ground. Pin 1 is used to feed the Raspberry Pi with the 5V incoming from the Barrel connector (which also feeds the LED lights since they cannot be fed through the Raspberry Pi itself because of the amperage limit). Pin 2 is used as a 3.3V source for all the remaining devices and Pin 3 as a ground. All the remaining pins are used as GPIOs to either input or output data.

# 6. Functions & Routines

So far, it was seen how the hardware is organized and interconnected, but how exactly are all the devices working together to make the whole system work? Reviewing the whole system would be too complex and outside of the scope of this document, but let’s see a couple of examples.

To understand how one of the inventory functions of the Smart Fridge Android App works please refer to Figure 12 in the Appendix where flow diagram can be found that shows how the low inventory and shopping list functions are working together to make it all work. For a more technical point of view, Figure 11 shows some code fragments for some of the functions that are used in the Android App.

To get a deeper understanding of what exactly happens in the system when a user opens the refrigerator door and to understand what sequence of events occurs in which case, please refer to Figure 13 of the Appendix where flow diagram can be found that illustrates the exact chain of events from a hardware perspective.

# 7. Appendix

This section contains relevant Figures and Tables that contain schematics, bill of materials, code fragments and flow diagrams.

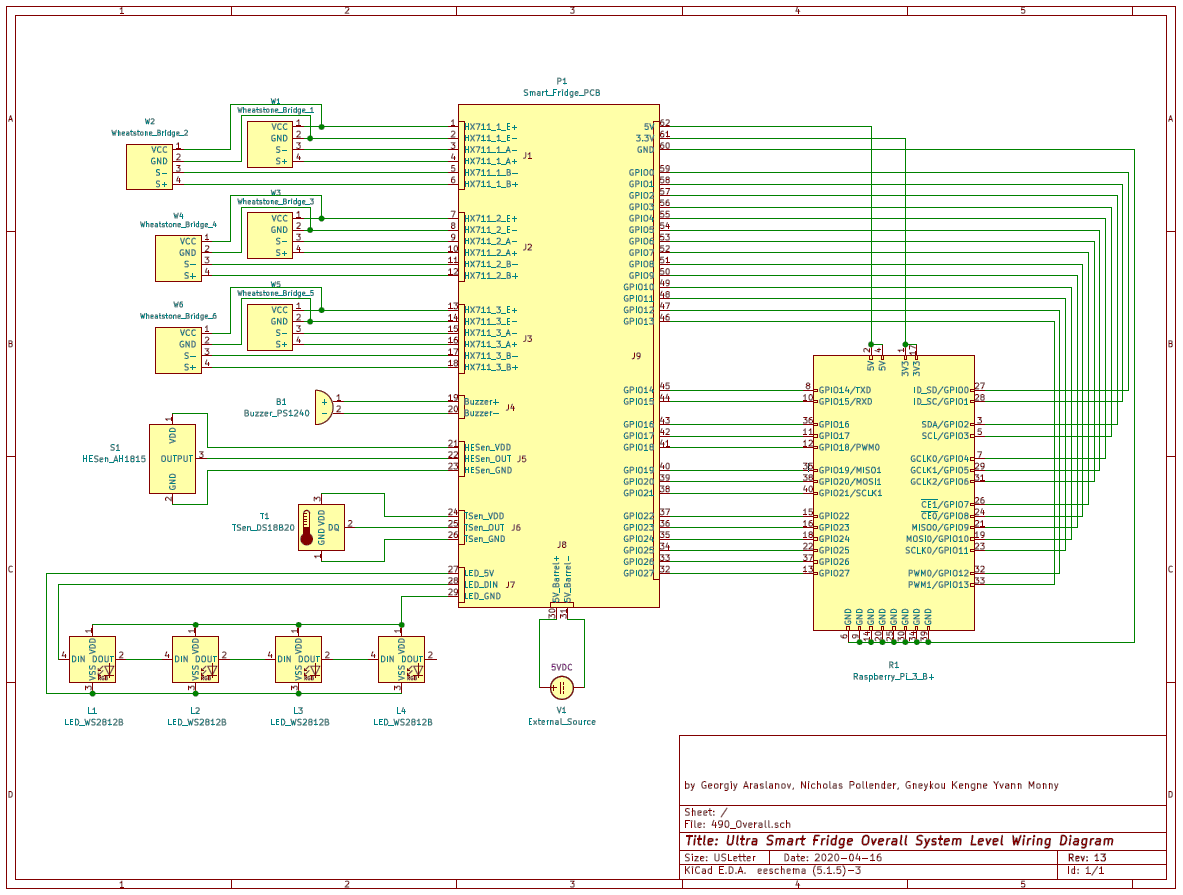


Figure 7: Ultra Smart Fridge Overall System Level Wiring Diagram

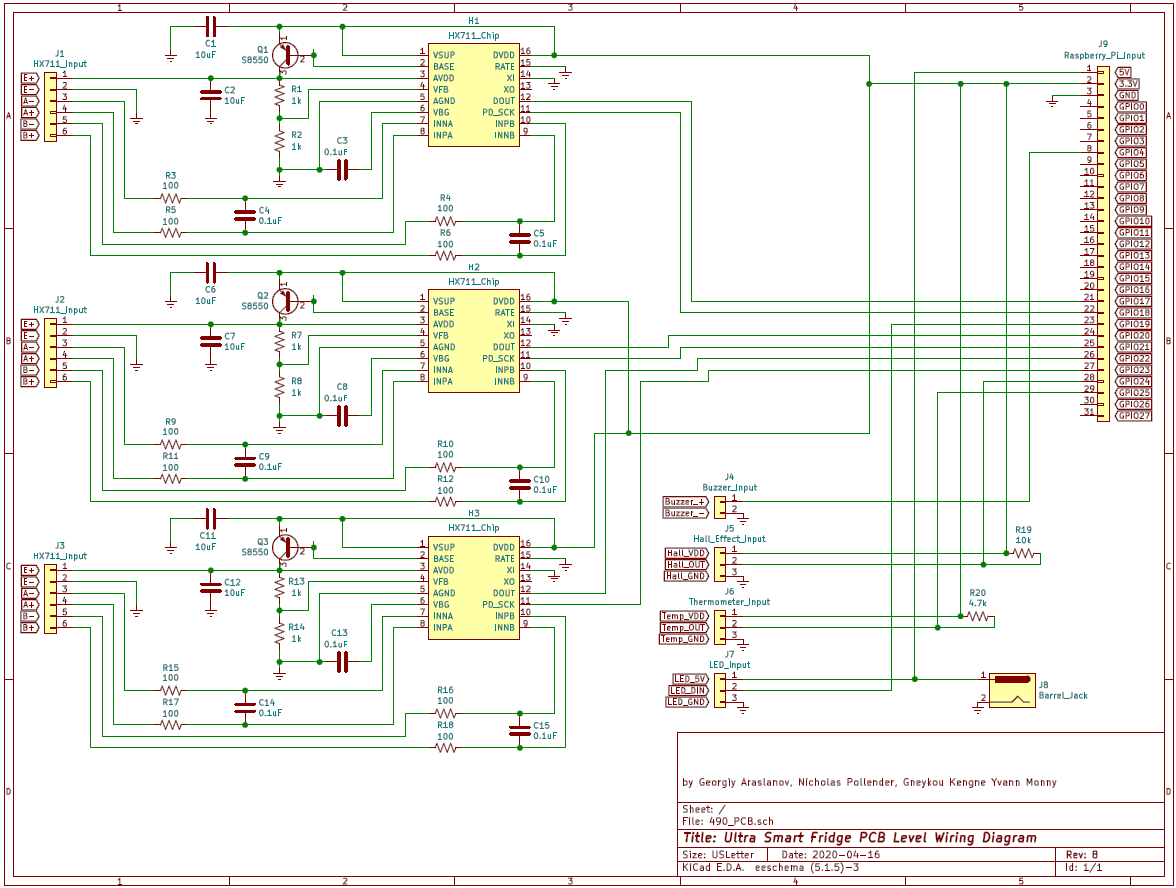


Figure 8: Ultra Smart Fridge PCB Level Wiring Diagram

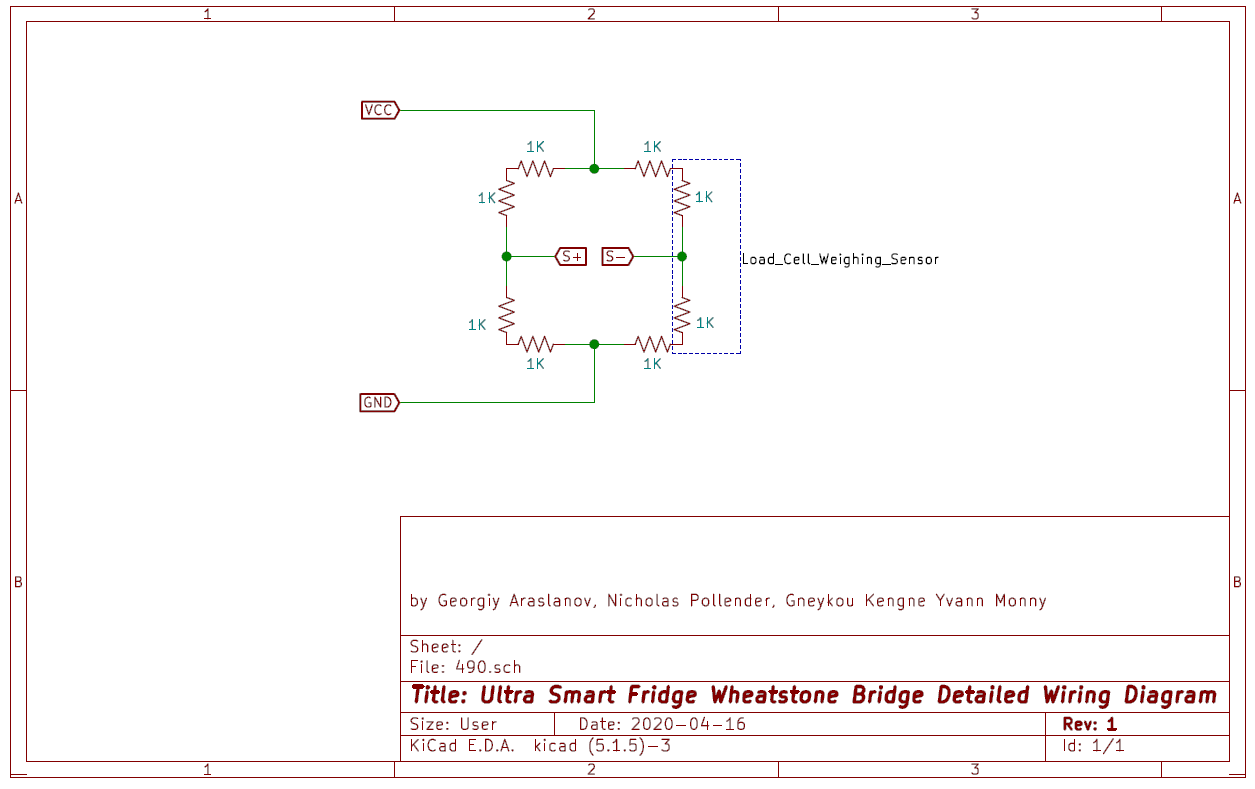


Figure 9: Ultra Smart Fridge Wheatstone Bridge Detailed Wiring Diagram

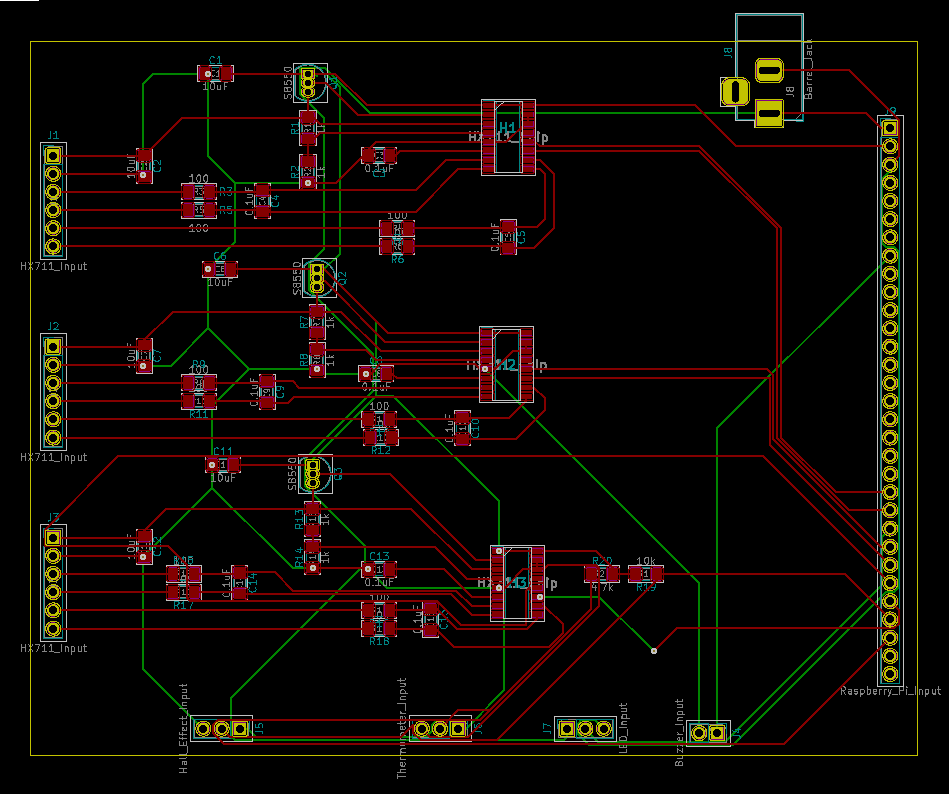


Figure 10: Ultra Smart Fridge PCB Layout

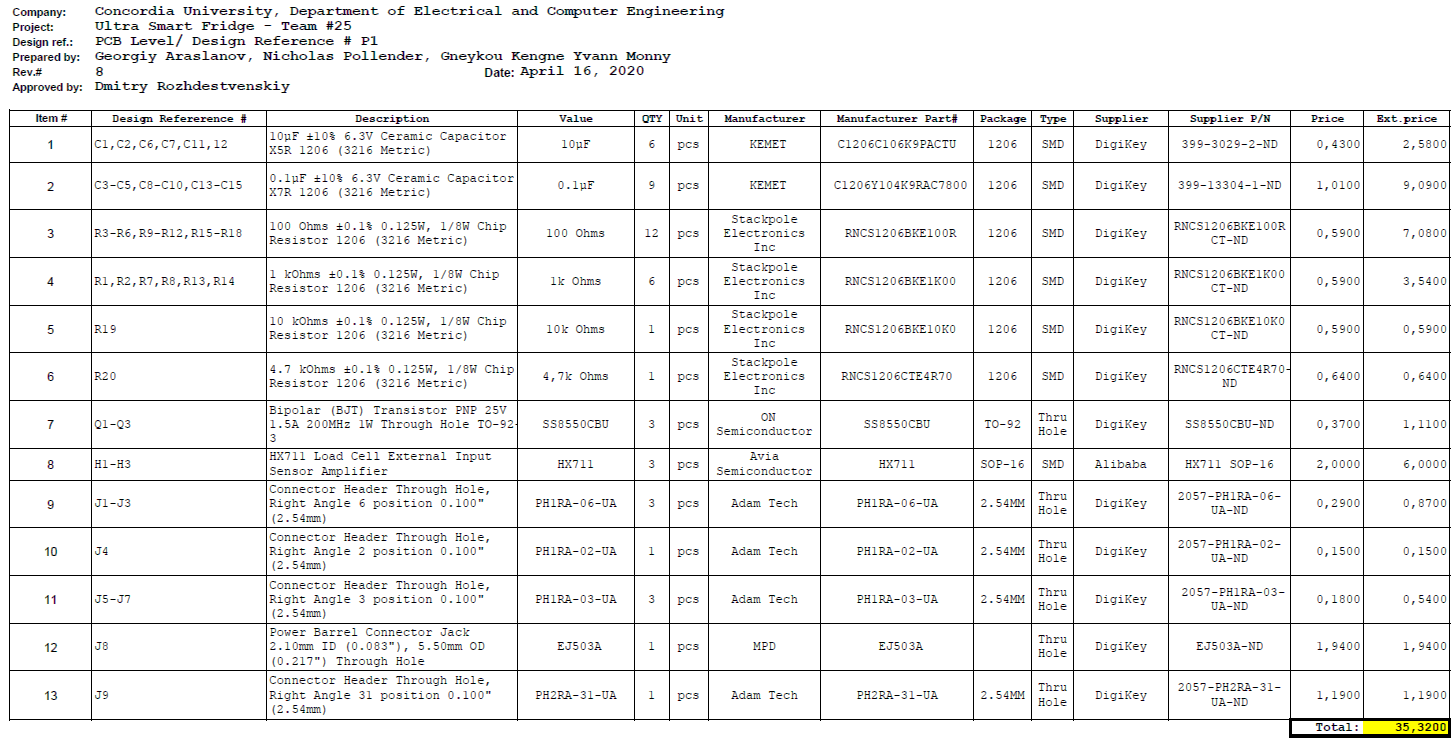
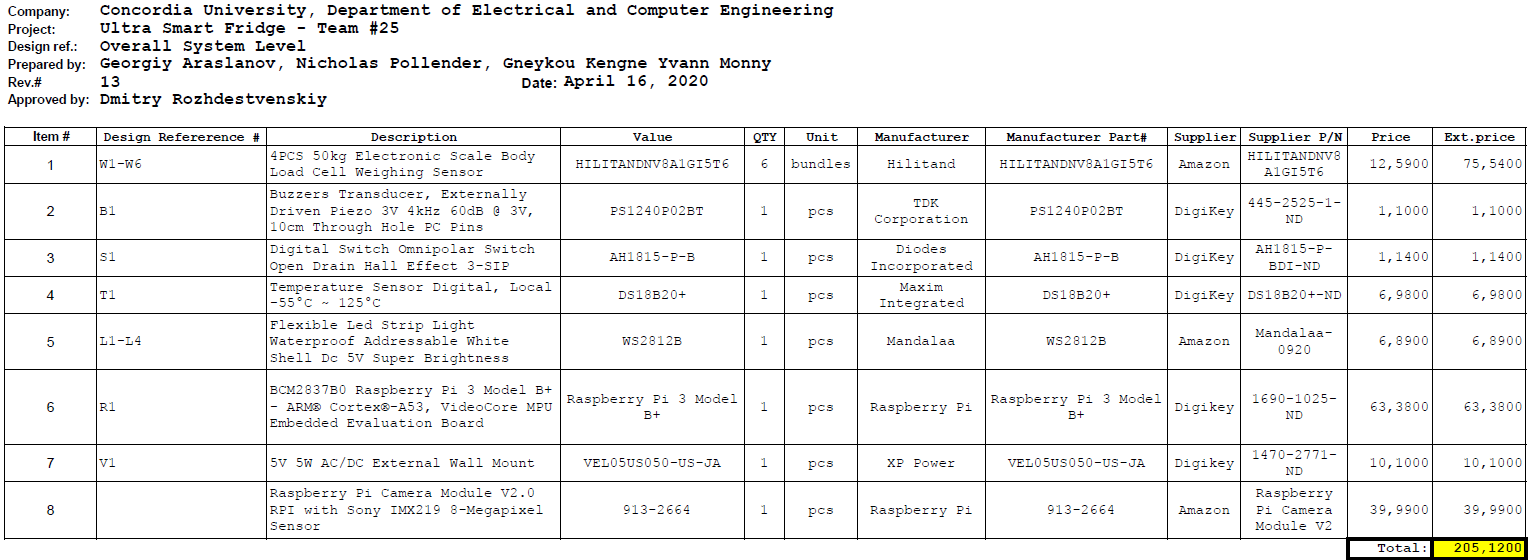


Table 2: Ultra Smart Fridge Overall System Level Bill of Material

Table 3: Ultra Smart Fridge PCB Level Bill of Material

//This method checks the weights of each partition and returns a result.

//If any weight is below the threshold, then the result is set to true, otherwise false.

public Boolean checkWeights() {

Boolean result = false;

int i = 0;

editor = sp.edit();

if (IDB.checkIfLow(1)) {

result = true;

} else {

i++;

} if (IDB.checkIfLow(2)) {

result = true;

} else {

i++;

} if (IDB.checkIfLow(3)) {

result = true;

} else {

i++;

} if (IDB.checkIfLow(4)) {

result = true;

} else {

i++;

} if (IDB.checkIfLow(5)) {

result = true;

} else {

i++;

} if (IDB.checkIfLow(6)) {

result = true;

}

else {

i++;

}

if (i == 6) {

editor.putBoolean(CHECKED, false);

editor.apply();

}

return result;

}

//this code fragments checks if there are any low weights, if there are, then it checks a SharedPreference to see

//if the low inventory prompt has already been used recently. If it hasn't, then the dialog appears with toSL()

if (checkWeights()) {

if (!sp.getBoolean(CHECKED, false)) {

editor = sp.edit();

editor.putBoolean(CHECKED, true);

editor.apply();

toSL();

}

}

//This method shows how a dialog box is made, and what happens if the user presses yes or no.

//If yes, then the app moves to the shopping list activity, and the user decides what to add to the shopping list.

//If no, do nothing.  
public void toSL() {

final Dialog dialog = new Dialog(MainActivity.this);  
dialog.setContentView(R.layout.delete\_shopping\_list\_dialog);

TextView prompt = (TextView)dialog.findViewById(R.id.areyousure);

Button yes = (Button)dialog.findViewById(R.id.yesbtn);

Button no = (Button)dialog.findViewById(R.id.nobtn);

prompt.setEnabled(true);

yes.setEnabled(true);

no.setEnabled(true);

prompt.setText("Low inventory on item(s), go to shopping list?");

yes.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainActivity.this, ShoppingList.class);

intent.putExtra("open\_r", true);

dialog.dismiss();

startActivity(intent);

} });

no.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dialog.dismiss(); }

});

dialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.TRANSPARENT));

dialog.show();

}

Figure 11: Code Fragments

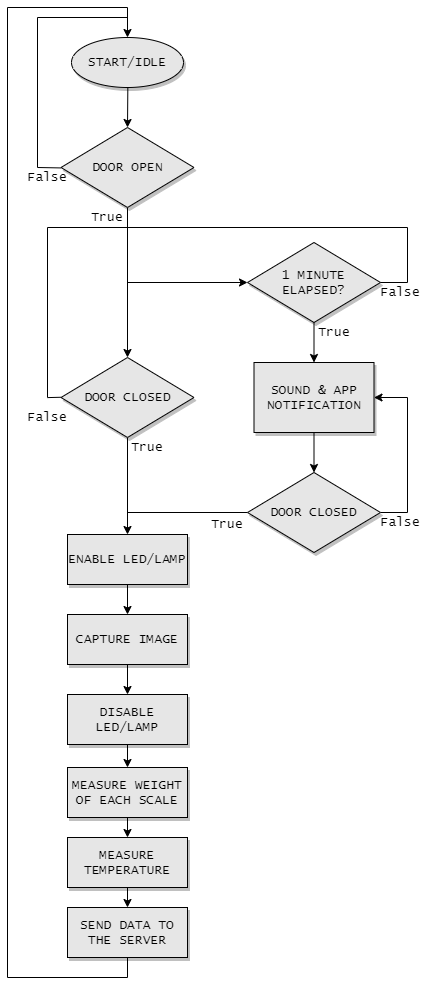
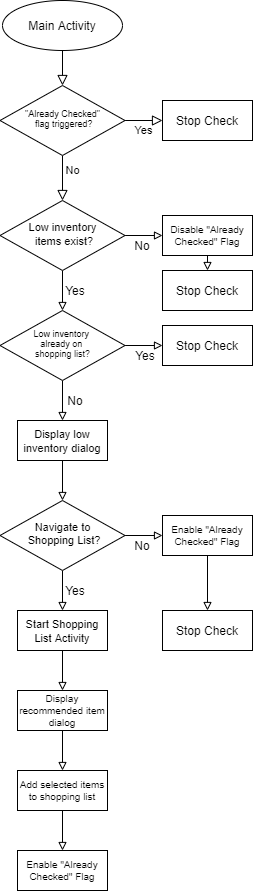


Figure 13: Refrigerator – Door opening Flow Diagram

Figure 12: Android App – Inventory Check Flow Diagram

## 7.3 ELSEE Aspects

### Task 1 – Identifying Stakeholders

In decreasing level of importance, the stakeholders identified for this product are:

* Consumers – The end user who purchases the refrigerator. They interact with the refrigerator and the smart phone application. Tasks include monitoring food quantity, creating shopping lists, searching for recipes, while also being encouraged to keep contents organized.
* Server Administrators – Responsible for managing and maintaining the server where the data is stored. Without the server, updated data cannot reach the end user.
* Programmers – Responsible for developing the application to control the refrigerator. Without the application, the functionality of the refrigerator is rendered virtually useless. Programmers push new updates to improve features and fix bugs of previous versions to ensure end user satisfaction.
* Manufacturer – Responsible for manufacturing the refrigerator, this includes sensor/MCU configurations, as well as the physical frame of the appliance.
* Retailer – Responsible for selling the refrigerator to the end user. The appliance is marketable as it stands with the new features and is versatile as it also includes common features present in existing solutions.

### Task 2 – Listen to the Stakeholders

In the real world, there exists real-life stories related to ELSEE failures, among them are the following stories. The task at hand is to make a product that can overcome these ELSEE failures.

* Firstly, a company named Haier that has recalled about 137,000 refrigerators because of a potential fire hazard. The Consumer Product Safety Commission did not identify any injuries, however there were reports of smoke, fire, and property damage related to the problem. The cause of this was an electrical component within the refrigerator, which was found to be short circuiting, ultimately leading to a fire. Customers were compensated with the offer of a free repair or a $150 rebate toward a new Haier refrigerator.
* Secondly, an article called “Smart Devices Leaking Data to Teach Giants Raises New IoT Privacy Issues” brings up the concerns that researchers and scientists are raising about smart devices (TV’s, smart home appliances) tracking and leaking end user data. A study showed that 72 out of 81 IoT devices shared data with third parties. The shared information went far beyond basic information about the device, the data included IP addresses, device specifications, configurations, usage habits, and location data.

### Task 3 – Learn from Experts

Organizations that oversee the field of the project at hand must be identified.

* Natural Resources Canada – The energy efficient division of the organization oversees certifications like EnerGuide and Energy Star which promote energy efficient products, lower utility bills, and reduce impacts on the environment.
* The Office of the Privacy Commissioner of Canada – Provides advice and information for individuals about protecting personal information for individuals as well as enforcing federal privacy laws, the rules for personal information must be handled. One aspect being the PIPEDA, which applies to the collection, usage, or disclosure of personal information.

### Task 4 – Addressing ELSEE aspects

Below is a summary of the ELSEE aspects discussed in Phase 2.

* Ethical – It is important to be transparent with the end user. Moreover, a layer of security should be added to any interaction with the data on the server, as well as restrict access to any unauthorized personnel.
* Legal – In the modern world, privacy has become crucial. And whenever privacy is mentioned, consent closely follows. It is important to make sure that personal information of the user is only collected, used or disclosed with the user’s approval, according to rules and regulations.
* Social – Being healthy should be apart of a user’s overall lifestyle. A healthy lifestyle helps keep away illnesses and diseases. One of the ways through which that could be achieved is through good nutrition. The features implemented in this product will enable users to achieve this, by suggesting healthy recipes based on the refrigerator’s contents, for example.
* Environmental – Given the fact that climate change is an issue these days, it is important to ensure that the product is made sustainable for the entire duration of its life cycle, from the manufacturing to the disposal of its parts.
* Economic – Compared to other smart refrigerators, this product is priced comparatively. This was done with the hopes of bridging the gap between those who can and cannot afford smart home appliances.

### Task 5 – Validation of the Projects Approach

This product features little involvement of third-party applications/technologies. The server is hosted on AWS, with the data center being in Canada. No unauthorized user can access the AWS root account. Furthermore, credentials for accessing the data are not committed to the GitHub repository even though they do not provide many privileges, only devices with authorized access to the data center will use these credentials. Security groups are also used to minimize authorized IP addresses.

The code of the software contains no sensitive information as very little is required from the user, with the only data being stored on the device is the user’s own refrigerator data, and if they wish, their name. All interactions within the system are secured with the use of ‘secret’ keys when any data transmission is made. Finally, the server uses PostgreSQL which adheres more closely to SQL standards, handles concurrency better than other platforms, and is known for protecting data integrity, making it less vulnerable to data corruption.

### Task 6 – Reflection and Remarks

Reflecting on the ELSEE aspects has helped improve every facet of the product. If the socio-economic circumstances were to change, the product would not be affected simply because nowadays, refrigerators are affordable, and they are used to fulfill the basic needs of a human being.

Some of the assumptions and risks considered are the trust and belief in the good faith, professionalism and integrity of the programmers and server administrators as they will be working with sensitive information, assuming no misdeed such as tracking unconsented information.

In the case of a design failure, for example the stories discussed in Task 2, or just missing the mark and not developing a good enough product for the end user, the result is pushing the engineer to adopt different methods for developing a solution. For example, completely re-imagining a design to make society better, this can be accurately depicted with the recent release of the *Sonic the Hedgehog* movie where fans were outraged by the initial design of the protagonist and the studio gathered that feedback and completely re-designed it. Feedback is a key measurement of success, it cannot always be achieved immediately, and sometimes failure is a necessity to reach success.