1. Automated Inventory Tracker (AIT)

Figure A - Project Overview Display

Figure A – Project Overview Display

University of Central Florida
Department of Electrical and Computer Engineering

Dr. Richie and Dr. Lei Wei
Senior Design 1

Group 24

Gaspar Dantas - Computer Engineering

Sonu Thummar - Computer Engineering

Justin Rehg - Computer Engineering

Lody Morillo - Electrical Engineering

Table of Contents

Table of Figures

Table of Tables

Table of Diagrams

2. Executive Summary

The Automated Inventory Tracker (AIT) intends to automate a common task we have to deal with in our homes, which is tracking the products we have in our fridge, as well as when the product shall expire. This project will consist of a low cost design, consisting of a tracker that will be located outside the refrigerator at a convenient location for the users - attached to the PCB so there can be high accuracy on barcode detection. This product will be fused with a mobile application that we will develop to allow for easy compatibility with the user. We will allow the user to view all items within the fridge, to facilitate grocery shopping purchases, without the need of checking whether some items are needed. Our project will have a notification system to inform the user when an item is about to expire. This lightweight design will include a user friendly application that will scan entry and exit of products and update in real-time.

The purpose of smart inventory is to allow people to easily manage their inventories from their cell-phones with ease. A lot of food is wasted today because people forget to track their expiration date. This app will allow them to properly manage their items and also plan ahead. Also many people when visiting grocery shops forget what items they need or what items are finished from their inventory. With the help of smart inventory they can easily manage their grocery shopping list.

3. Project Description

3.1 Requirement Specifications

This system will be triggered every time an item is inserted or removed from the fridge and will keep adding up until the process is complete and the user can view all of the items through the mobile app. As displayed in table A, the inventory shall contain a scanner attached to the PCB, which will be connected to the application through bluetooth. The product details will then be inserted into our SQL DB, as discussed in table B, via API calls, however the user may also insert/delete products manually through the application, as well as have the functionality for the user to adjust expiration dates and percentage of storage left.

Table A. Hardware requirements

Table A - Hardware Requirements

1	The system should be able to scan the barcode/Qr Code
2	The system should be able to Capture all the information available from the barcode scanner like Name, Quantity, Expiration Date etc.

3	The system should automatically distinguish between name, expiration date , product specifications
4	Power supply to have our scanner function. 5 volt delivery
5	The system should distinguish between the add/remove functionality of the product(if a product is added or removed)
6	Bluetooth 5.0 adapter to connect and send data to the user's cell phone.
7	Button switch for turning ON/OFF the scanner system and display screen.
8	PandaBoard to be used as a platform for mobile software development.
9	Display to communicate information such as low battery to the user.
10	USB Hub 4-Port needed for extra USB connections.

Table B. Software Requirements

Table B - Software Requirements

1	The software should be able to store items from the inventory in a SQL DB (up to 16TB worth of data)
2	The software should have an API to retrieve product details.
3	The software should be able to insert records directly from the scan and allow users to add inventory items.
4	The software should be able to sort all the data by names, expiration date
5	The software should be able to delete the data
6	The software should be able to link to the scanner through bluetooth
7	The software should be able to send notifications to the user regarding item expiration date
8	The software should allow the user to adjust amount remaining of the product (0-100%)
9	The software should allow the user to add/remove expiration dates through datetime pickers
10	The application should run in any operating system, intended mainly for phone usage

3.2 Block Diagram

Diagram A has a clear demonstration of the AIT workflow. We intend to use the PCB switch to have a scan in or scan out notation to indicate addition or deletion of a product into our SQL DB. This will trigger the software to execute the CRUD operation based on the users selection as displayed in the diagram A. In terms of group workload distribution we have listed in diagram B how we intend to accomplish this objective in a teamwork fashion. In this category, we have divided the project into 3 sub-elements to facilitate the research and assignments. For the mobile application, Gaspar and Sonu will be the developers in charge of connected the front-end with the SQL DB and handling all of the API calls when a product gets scanned. Lody and Justin will be handling the scanner and sensors and how they will be integrated into the PCB. Lastly we will come together as a group and accomplish the microcontroller integration - implementing the bluetooth support and switches to the board.

Diagram A. General

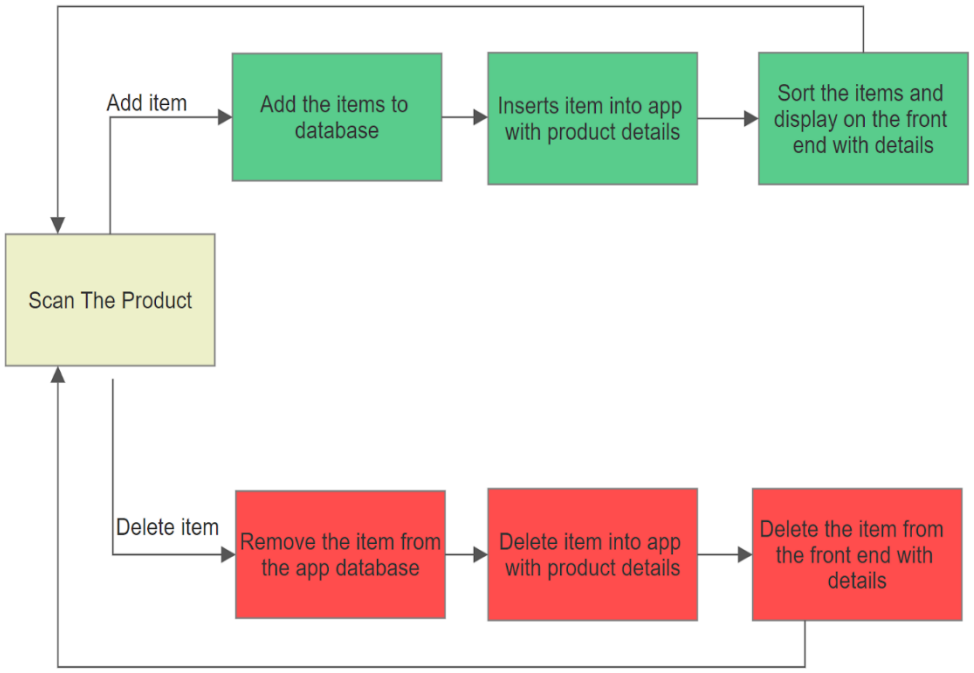


Diagram A - General

3.3 Group Breakdown

The following Diagram B is the group breakdown, here software part is assigned to Gaspar and Sonu. The hardware part is assigned to Lody and Justin. Once we have our hardware and software working, we will be all working together to combine them to make our project work. Further details and given below in Diagram B.

Diagram B. Group Breakdown

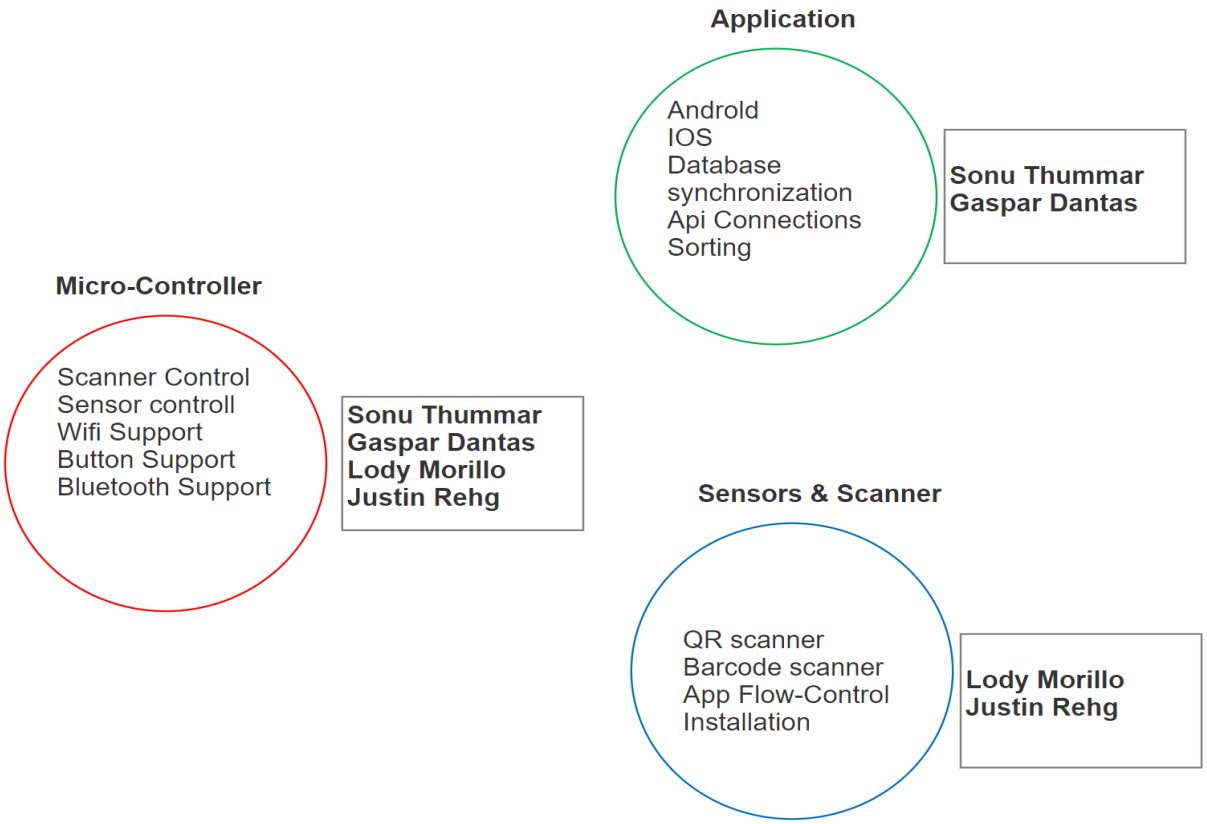


Diagram B - Group Breakdown

3.4 Decision Matrix

This project has been selected by our group, although we encountered some challenges that might cause us to redesign a few specifications, such as acknowledging not all barcodes will retrieve expiration date, so we can potentially make this adjustable by the user in case this information is not attained from the initial scan. Another challenge we have encountered is discovering an appropriate angle to facilitate the scan from our device from a static location in the refrigerator. We might have to modify and readjust by making this component an outer tool that the user can utilize instead, and this product will be a notification system specifying reminders of when something is about to expire within the app and from the device itself.

3.5 House of Quality Diagram

In this diagram C, we have blended customer expectations along with product requirements gathered from hardware and software specifications. We display the correlation across both and the goals we intend to accomplish.

Diagram C - House of Quality

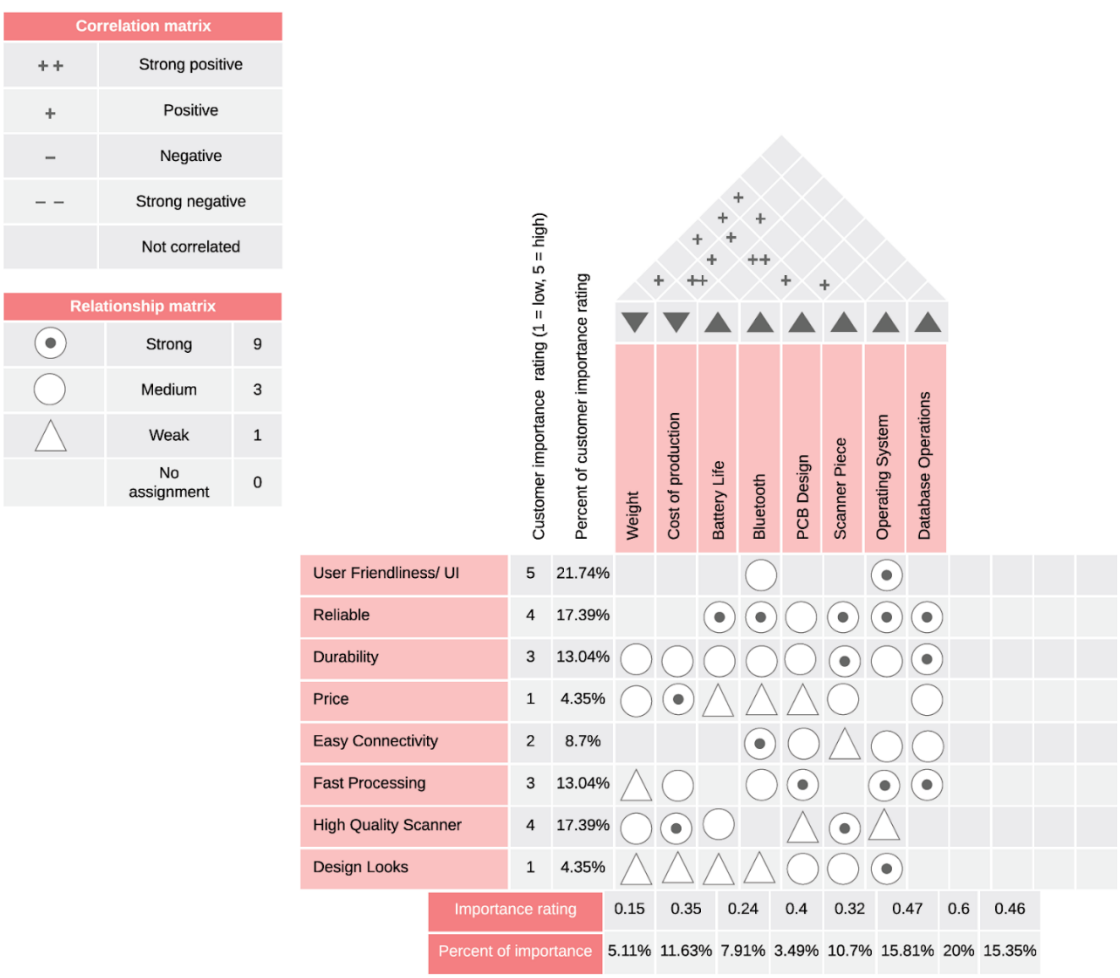


Diagram C - House of Quality

4. Research related to Project Definition and Part Selection

4.1 Previous Products

As seen below, the main companies investing in refrigerator technology advancements, with “Smart” capabilities are Samsung and LG. Typically these features included a screen attached to the fridge, with AI product detection and wifi integration. Below we shall individually characterize each product in detail according to their specifications and features

4.1.1 Samsung RF4289HARS (Intelligent Dispenser)

As seen below in Figure B, this is one of the earliest “smart-fridge” projects from Samsung. This product includes wi-fi integration, touch-screen capabilities, allowing users to utilize this screen as a smart-device to view emails, running Android as the OS. The smart-fridge had the sole capability of inserting items manually via an app. The user selects the product type to narrow down to what exactly the product is, then inserts expiration dates and the date inserted manually prior to being inserted into the DB via the app. Notifications can be enabled and a specified range of dates for this process as well.

Figure B – Samsung RF4289HARS

Figure B - Samsung RF4289HARS

4.1.2 Samsung RS27T5561SR

As shown below, Figure C is one of the latest “smart-fridge” projects from Samsung. This product includes a huge touchscreen display on the left side of the fridge. This device has image recognition to recommend items when the user is grocery shopping (connected to smartphone via app). This fridge also has cameras to allow for live views of what is inside. Also, the touchscreen display allows pictures to be added, notes/reminders as well, selected songs to listen to, etc. This modern product has a ton of updates and user-friendly capabilities compared to the last product described (Figure B), such as a connection to MasterCard to allow users to buy straight from the fridge, or via an app that connects to the fridge, recommending products that are currently low on.

Figure C - Samsung RS27T5561SR

Figure C - Samsung RS27T5561SR

4.1.3 LG Instaview

Figure D – LG InstaView

Figure D - LG instaview

As shown above in Figure D, LG has developed a product to compete with Samsung's innovative product RS27T5561SR (Figure B). This fridge has a display on the left side that can be transparent to view all of the items, but may also become a touchscreen device for the user.

This device uses ThinQ technology to differentiate each product when displaying to the user. There are also voice recognition and control capabilities similar to Alexa from Amazon. There are notifications sent in case the fridge door is open for a long time, and adjustments can be made in terms of the active temperature in the fridge. These features make this product extremely user-friendly similarly to the new product from Samsung shown in Figure B. These edge-breaking tech products have been released in 2020 (Figure B and Figure C), showing the huge impact of the touch-screen features and "smart-devices" in the world today, introduced to us about a decade ago. As observed, smart-fridges are starting to develop in the current market, showing strong potential for the next couple of years.

4.2 Relevant Technologies

While working on this project, we are using many tools and softwares which are relevant and are playing a vital role in towards the completion of what we are trying to achieve. Those software serves a critical purpose for organization and availability of documents that we need as a group to complete the project. The roles of these software and how we are utilizing those softwares are explained here in this section.

4.2.1 Google Docs:

We are using Google Docs to write our 120 Page research paper. Google Docs is a great tool to collaborate with the team members to write a Group Paper. Here, we can all use the same google doc at the same time and all the changes are updated in the real time. The Autosave feature of the document allows us to be free of the concern that we might lose our work in any unforeseen situation. We can access our document from anywhere without the need of carrying our computer to places. One of the best things is that our document is also viewable from our phones and hence, if we need to check on the document or the progress, we can do it remotely from anywhere. There is an option to create specific tasks for a member in the Google Docs. We are using that feature to assign different tasks to different team members. There is also an option to make comments to any section of the document. This allows members to notify other members what they think about that section and recommend any changes that are required. There are multiple tools that this software allows us. We can make Titles, Heading and paragraphs to this document and it automatically creates a Table of Content for us which includes Anchor Tags. This allows us to click on any section and it directly takes us to that section. We can also track the user activity on the document and it allows us to keep track of what tasks are completed and what tasks are remaining. Once we are done with all the formatting and ready to submit, we can download the document in various types like Microsoft Word, PDF etc.

4.2.2 Zoom:

During this Pandemic, this software has been extremely useful for all the students all over the world. In a normal scenario, we will be meeting people in person at campus for our weekly

meeting or when we get together to work for the Research Paper. However, it is not possible in the given scenario and hence, we are using Zoom Meetings to Collaborate together and work on our Projects. All of our weekly meetings are held in the Zoom. When working on the project, we are using Zoom's screen sharing option to show other members what we have done so far. We can also record our meetings so if a person is not able to attend the meeting can go back to the recording and get updates. Zoom allows you to make annotations while a person is presenting and hence, it becomes easy to point out different things. One of the features that has been really helpful is the Zoom's remote control feature where one can take over another person's screen and do work. While working on the software diagrams and tables, there are often times when we need to collaborate together and this feature is really helpful in those cases. Zoom also has a chat feature and we use that feature to send any link or URLs we need to share during our meetings. We have set recurring meetings for our weekly meeting and we all use the same link every week to join our meetings to work on our projects. Collaboration is very much needed when working on the projects and Zoom allows all the members of the group to come together to a place and collaborate on the project.

4.2.3 Microsoft Sharepoint

Microsoft SharePoint is a great tool to share all the documents, links, files and is accessible for all the members of the group at one place. At the beginning of our project, we have created a Microsoft SharePoint Team site. This site allows us to keep all our progress saved and accessible to everyone at any point of time. We have a document library in SharePoint where we have all our deliverables and design documents stored. This allows the group members to access all the documents or files at any given point if they find need to update any document. Microsoft Sharepoint also allows users to create custom links to softwares and files all over the internet. For example, we have our GitHub repository and Google Docs and both of them are custom made links available on our SharePoint site. Similarly, all the other resources and online tools that we are using for this project are linked to the SharePoint. It becomes very easy to organize the content and files by using this platform. SharePoint allows us to keep track of any document or article that was used to do our research needed for this project.

4.2.4 Discord

Discord is a VoIP, instant messaging and digital distribution platform designed for creating communities. This software has proven very helpful in communicating with the team members about the tasks and deadlines required for this project. We have created a team server for our group on the discord. On the Server, we have different channels that we use for different purposes. We use general for all the general chats, we use the notes and resources channels to save any important notes or resources that we find during our research so that all the team members can have a look at that information. We are also using the internal meeting channel to share our Zoom links there. It is very easy in discord to communicate any new information that comes to our attention. There are audio channels in the discord which we use to talk with the team members when we have any questions or updates. Overall, this tool has proven a great help to communicate instantly with the team members.

4.2.5 GitHub

For any Software development and especially in groups, it is highly recommended to use GitHub. It is a very helpful tool to manage the code and it does wonders when it comes to version control. When multiple people are working on a single project it is very important that each person completes a specific task assigned to them and keeps track of the version so that the code does not get mixed up. GitHub allows us to achieve this with ease. We have all our code safe on our GitHub repository. We have a master branch which will be working as a production branch . We will be creating a development branch and then from there, all the members working on the software have their own branch. This allows us to pull the code to our branch and commit our changes to that version. Once all the testing is completed, that code is pushed to the dev branch. Here all the members can pull the updated code and can test all the changes from their side. This way the code that we develop and works fine always stays in the develop branch and we do not have to worry about merging our codes and spending hours debugging any silly mistakes. Once everything works fine, in the development branch, we will then push our code to the master branch which will be our production code. Github also allows us to make comments when we are pushing our code so this helps us to know what was changed during that push. Github also takes care of the formatting and hence, if the code is not formatted properly, it will automatically do it for us which helps a lot in reading the code. The image below explains how these branch system works:

Figure E- Version Control Git Procedures

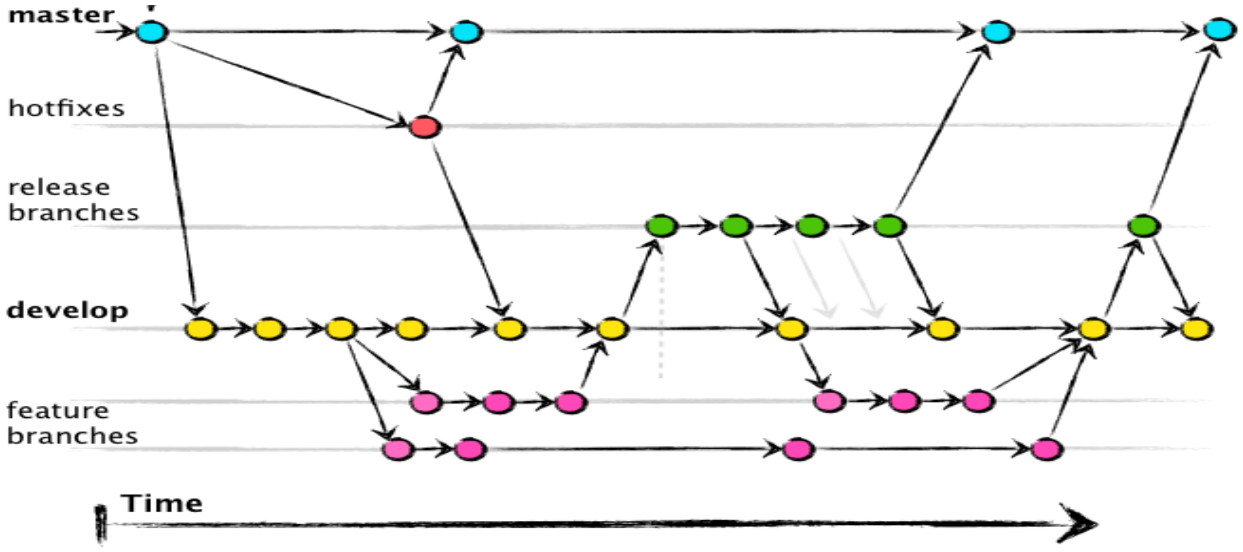


Figure E - Version Control Git Procedures

4.2.6 Draw.io

For this project, we needed to create a lot of diagrams, software flowcharts, schematics, block diagrams, Gantt Charts etc. Draw.Io is a tool where we were able to create all of this with ease. It is a great tool that allows one to create multiple forms of diagrams and flowcharts as required for the project. It is very easy to use this tool since it comes with a lot of templates of Block Diagrams, FlowCharts, Schematics etc. For this project, we have created most of our diagrams using this tool. It is also possible for two people to collaborate and work on this together. Once we are done with creating the diagrams, the tool allows us to download our diagrams in many formats like PDFs or JPEG, or PNG. This makes it very simple for us to attach that image in our research paper.

4.2.7 Eagle

This software is what we use to design our Hardware. We have been introduced to this software during our Junior Design class. This software is very helpful for the PCB design. We are using this software for designing the PCB for our project. This software is also a great tool to look for the parts required to build our project. Once we design our PCB, there is a tool in this software which will search for most vendors and allow us to order parts needed for Hardware. It gives a Bill of Material (BOM) which is great to keep track of the budget and easy to build tools. The PCB design feature of this software is a great tool and helps a lot to design the PCB.

5. Administrative Content

5.1 Project Milestones

Our initial milestone is to have a complete guide and specifications related to the project so we can have a facilitated build for SD2 in the Summer. We will establish an initial UI with plans on how this app will function (all user features) and start acquiring the proper skillset for development in Native. As shown in Table C, for Senior Design 1 we plan to complete all of the appropriate documentation necessary prior to the build, having an initial version of the application released, with the UI established and getting all of the designs complete for both the hardware and software of the AIT project. For Senior Design 2, as displayed in table D, we intend to order all of the necessary parts as documented and complete the build based on the designs of the PCB and mobile application. Our intent is to finish by the 10th week so we can have time for testing and properly debugging the system implemented.

Table C. Senior Design 1 Milestones

Table C - SD1 Milestones

No	Milestone(s)	Schedule
1	Research the project	6th week
2	Establish the UI	10th week

3	Integrate Documentation	12th week
4	Design the PCB	14th week
5	Design the physical model	15th Week
6	Design the App Screens	16th Week

5.1.1 MileStone Discussion - SD1

Here we will explain the Senior Design 1 Milestones in detail.

Research the Project:

One of the main things is to research the project and it is very important that we have enough information about the project that we are approaching and the components that we will be needing to bring the project to fruition. We planned to complete the research necessary for this project by 6th week. We have completed the research and we have now started working towards the development

Establish the UI:

When developing the software and Hardware, it is necessary to design the User Interface. UI is critical since it will be how the users will be using our project from the front end. We plan to complete the UI design by the end of 10th week. We were able to complete the UI design in time. The information is provided in this document in the UI section.

Integrate Documentation:

We need to document the findings and research for this project and then we can start building the project with the help of that documentation. This documentation will help us to go back and check our strategies on how to approach a specific problem. We plan integrate the documentation by 12th week.

Design the Physical Models:

We need to design how our hardware will look and how the things will fit into the model physically. Our Goal is to finish the Physical model design by 14th week.

Design the App Screens:

We need to design on how our screens will look like in our App. we need to discuss the colors, size, and views which we will integrate in our App. We plan to complete our App design by 16th week. So far, we have developed the Login Page screen and Main inventory page screen.

Table D. Senior Design 2 Milestones

Table D - SD2 Milestones

1	Order Parts	1st Week
2	Develop the App	5th Week
3	Build the model	7th Week
4	Integrate App with Model	9th Week
5	Testing & Debug	10th Week
6	Improvements/Fixings	11th Week
7	Showcase	12th Week

5.1.2 MileStone Discussion - SD2

Here we will explain the Senior Design 2 Milestones in detail.

Order Parts:

We will need to start ordering parts by the week 1 of the senior design 2 class. This is very important as there will be a case where a specific part could take time to arrive or it can also be possible that one of the parts might come defective. We want to allow us enough time so that under any of these circumstances, we have enough time to reorder parts. There can also be a case where we do not get our system to work in the first try and hence, if we have time, we can go back and order another part.

Develop the Application:

We plan to complete the development of the application by 5th week of Senior Design 2. We need to make sure that all the pages, functions, and operations are working as we desired. Our project is a high software based project and there will be cases where we will have to debug many things, if we develop the application by 5th week, we will have enough time to work on debugging the software and solve any issues that we face

Build the model:

We will simultaneously build our hardware model during the initial phase of the Senior Design 2 and we plan to complete our model by 7th Week. We need to create the PCB and use the different boards, connect the bluetooth and wifi modules with our hardware. If we complete this by 7th week, it will give us enough time to join both hardware and software pieces of the project.

Integrate App with Model:

Once we complete both hardware and software, it is time to integrate both the areas and complete our project for initial staging. Here we will need to make sure that the hardware is

connecting with the database and is able to make calls such as to add items and delete items. We also need to make sure that we are able to connect the hardware with the software for the first time because we will need to Authenticate the hardware so that it can make changes. We Plan to complete this integration by 9th Week of Senior Design 2

Testing & Debug:

Once we have developed our project, we will do a series of testing and debug on our project to make sure that everything is working as planned. We have created a test plan and we plan which we will use to make sure that our software passes all the tests. It is very important to test for all the scenarios possible and debug any issues that we face. We plan to complete this by 10th week.

Improvement & Fixings:

We plan to keep this week for any improvements and fixings that we might need towards the end. Some of the improvements might include having a recipe tab in the inventory where we can add some nice recipes which users can use to make dishes. We will be getting these recipes from open sources. Everything that we need should be done by 11th Week of SD2

Showcase

We will be showcasing the project to the Senior Design Committee during the final week of the SD2 class.

5.2 Financial/Budget

Our estimated budget for this project is just about $470, as shown in table E, which includes having a potent scanner, PCB, bluetooth support, a display, PandaBoard and switches (we are also taking into account additional quantities in case of errors). Our project is mostly software based as we are developing a user friendly app to associate to the components mentioned.

Table E. Budget Reporting

Table E - Budget Reporting

Material	Link	Quantity	Cost
Mimo Touch 2 - 7" Portable Resistive Touch Display, USB (UM-740R)	https://bit.ly/2Z5no3Z	1	$179.99
Barcode Scanner USB Laser	https://amzn.to/3rJKHwy	1	$29.99

PandaBoard (UEVM4460G-02-02-00)	https://bit.ly/3aZht66	1	$204.88
USB Hub 4-Port	https://amzn.to/2Z5D5bJ	1	$12.99
Bluetooth Adapter	https://amzn.to/3pfndOi	1	$8.97
PCB Custom	https://www.pcbway.com/	1	$6.50
5V Battery Pack Power Supply USB	https://www.adafruit.com/product/1959	1	$14.95
Button Switch Assortment Pack	https://www.adafruit.com/product/1010	1	$5.95
Breadboard-friendly SPDT Slide Switch	https://www.adafruit.com/product/805	3	$0.95(x3)= $2.85
Refrigerator	Team member	1	$0
TOTAL			**$467.07**

6. Unified Modelling Language

6.1 Gantt Chart

For the purposes of project management and ensuring we are on track to complete the software scope, we sub-tasked all components for senior design 1 and senior design 2 and created the Gantt chart shown on diagram D. These categories include preparation, specification, architecture, documentation, development and testing/debugging. We gave each other a month to spare for debugging/testing in order to ensure the project is complete properly, with no found issues.

For the first month of the project we found most effective to do more planning and learn about different methods we can use to properly strategize and trace goals. Our intent of utilizing a gantt chart is to be able to have realistic goals within the allocated time-period and meet smaller goals (sub-tasks) to ensure we are on the right track of the project completion.

In order to maximize the amount of research set for this project and still be weary about the timeline and deadlines established from our senior design class, a Gantt chart was the

ultimate resolution for defining time-spans for documentation, development, designing and still giving us some time to be ahead and have deadlines for complete test and debugging any encountered bugs in our system. Our intent is to be done with the entire project 1 month prior to the end of the Summer semester (2021) and give ourselves the time to ensure our system is fully functional.

Our team is currently implementing the database structure as we chose MySQL as the location for our inventory details. We have completed over 90% of all the designing elements from our software end, and will begin strategizing the development approach in the beginning of March - as displayed in the diagram D shown below.

6.1.1 Gantt Chart Detailed:

This is to explain the gantt chart in details. This gantt chart further divides our milestones in parts so it is easier for us to manage the tasks and assign the tasks to group members. This Gantt chart is for the software side. On the left we can see the list of tasks and on the right we can see the timeline. We have divided our Software development in 5 categories which are preparation, specification, architecture, specification, and document.

Preparation:

we need to prepare all the tools needed for the development. Here we need to decide the platform on which we will develop our software, we need to select the database and also research the APIs for the barcode lookup. So far we were able to complete the preparation part. We have selected the database and we were able to research multiple API services for the barcode selection. Hence, this milestone is now completed.

Diagram D

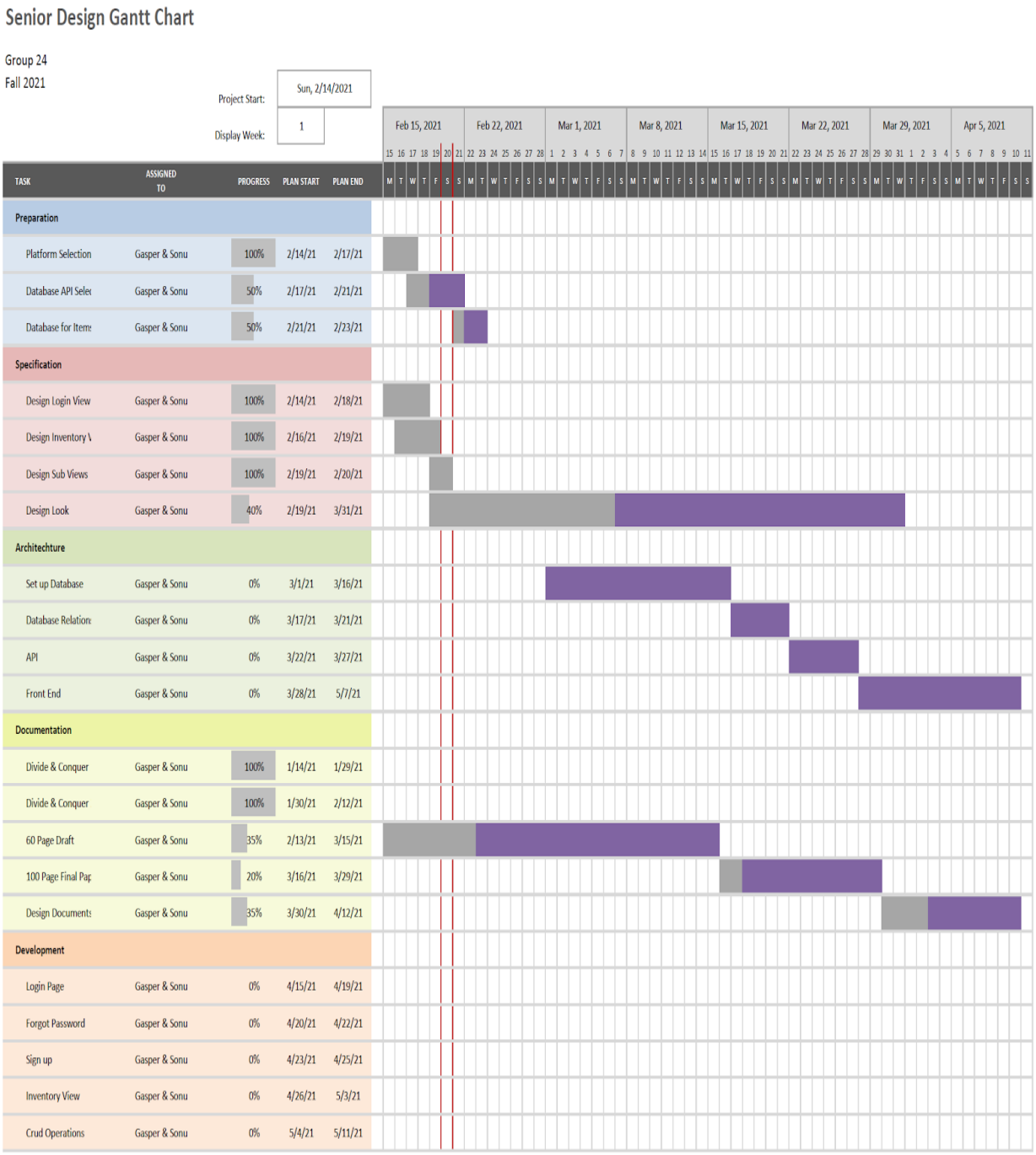
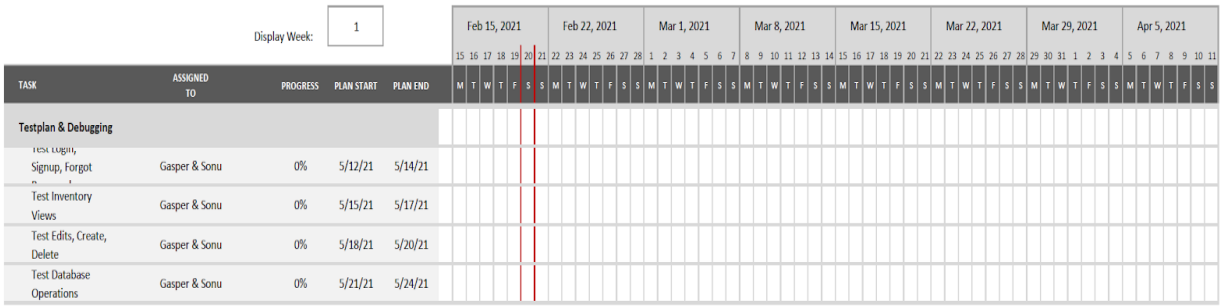


Diagram D - Gantt Chart



Display Week: 1

TASK	ASSIGNED TO	PROGRESS	PLAN START	PLAN END
Testplan & Debugging				
Test Login, Signup, Forgot	Gasper & Sonu	0%	5/12/21	5/14/21
Test Inventory Views	Gasper & Sonu	0%	5/15/21	5/17/21
Test Edits, Create, Delete	Gasper & Sonu	0%	5/18/21	5/20/21
Test Database Operations	Gasper & Sonu	0%	5/21/21	5/24/21

Specifications:

This Phase is to design the look of the software. Here we will come up with ideas on how our software will look. Like the Login Screen, pages, views and the inventory. We came up with the design of our software and how our software will look. We also have decided the colors which will be light blue and white colors. More details on the design can be seen in the UI section.

Architecture:

This phase really starts the development for us. Here we will be making our database, figure out the database relations and will also finalize the API. We will start making the skeleton for our screens and pages. We have started working on this piece as well. We completed creating the database in SQL. We have also created schema and tables with all our fields so far

Documentation:

Senior design 1 documentation is an important aspect as it gives an opportunity to research multiple things that are needed for the project. We have divided the phase in multiple tasks to meet with the deadlines and complete the documentation. We are making good progress for our documentation for this project. So far we are able to meet up all the deadlines and we are cruising towards the completion of the 120 pages final documentation.

Development:

We will start our development and we have divided our development by each page that we are making. We have divided our tasks in a similar manner. We have started development and it is in the initial stage. We have created pages like Login page and inventory page. Currently, we are working to configure Amazon's Cloud services so we can have our login and signup system ready.

TestPlan and Debugging:

After the development, we need to do testing and debugging to make sure that our project works as desired. We have divided our test plan in sub tasks so that we make sure all the test cases are taken into consideration.

As shown, we have completed most of the design elements, and we're now working on the database section from MySQL such as setting up the schema (as shown in diagram D)

6.2 Flow Chart

Given the design work and representation in the UI, we may now have a ‘big-picture’ analysis, now that all of the elements are sub-tasked. One of the best ways to have a software process defined is by a flow chart, so we decided to mitigate risks and have an overarching perspective of the users capabilities across the application. we can observe how everything is tied together, where DB transactions will be made, displaying from the very initial step of downloading the app all the way to performing the CRUD operations on the inventory view.

The flowchart may seem simplistic, but understanding the scope and defining limitations contributes heavily to the thought process when developing the app since now we have separation of concerns when working on the different aspects of its functionality.

Both the Gantt chart and the Flowchart are extremely valuable resources for any project, including for this one where it leans heavy on the software end, requiring a very thorough subtask listing to allow for a better projection of completion. Flowcharts are needed to prevent unnecessary logistics, therefore optimizing the process.

From a structure standpoint, the flowchart will ease the final steps from our Gantt charts, which will be testing/debugging. This is due to the thorough analysis of what is expected from each section of the tool (displayed in the flowchart). So if an issue is encountered, we will easily find the culprit of the found issue, making debugging/testing a much more simplified task as we expect the outcome from each process as shown below.

Diagram E – Flowchart

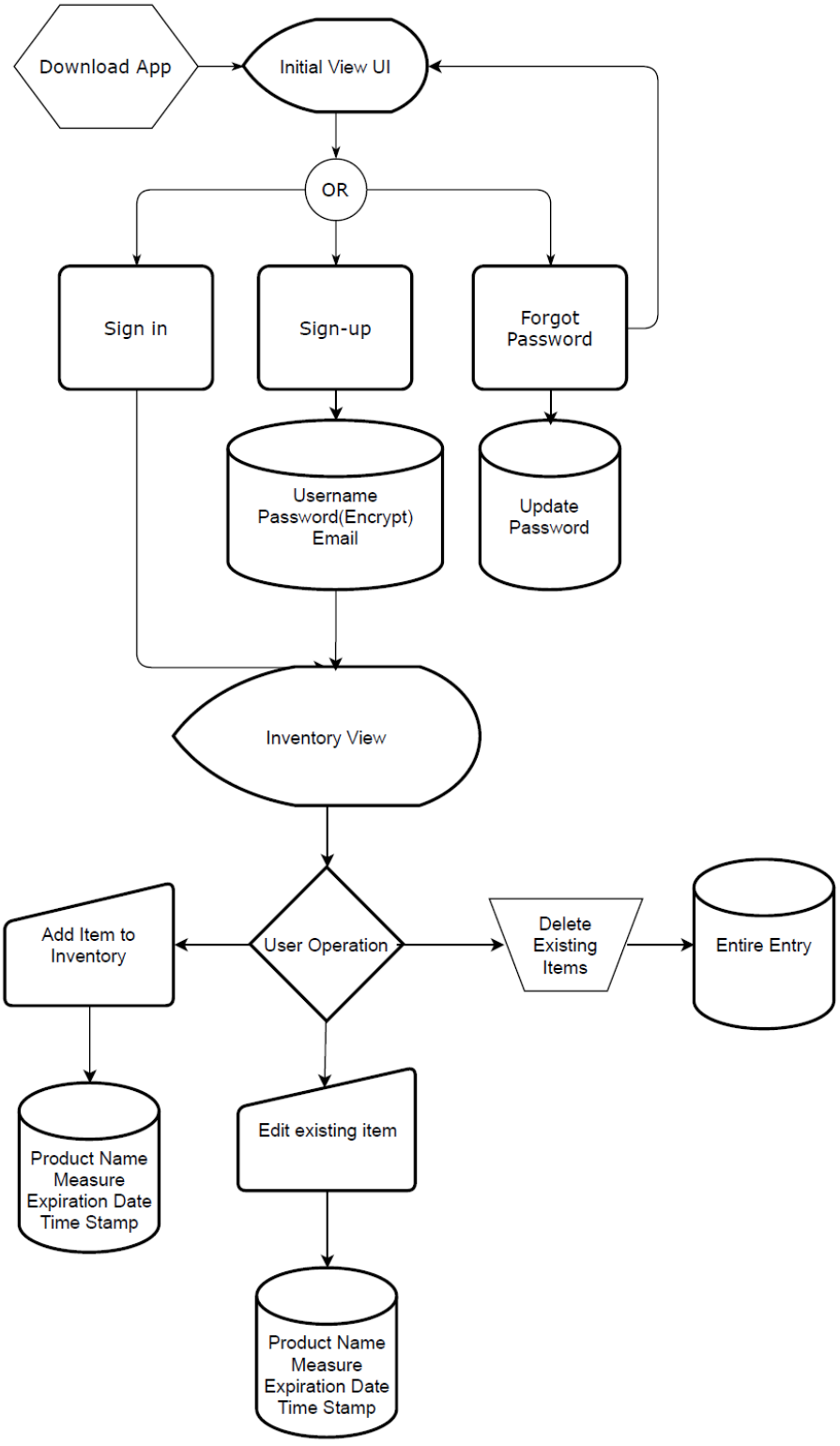


Diagram E - Flowchart

6.3 Use case Diagram

Diagram F – Use Case Diagram

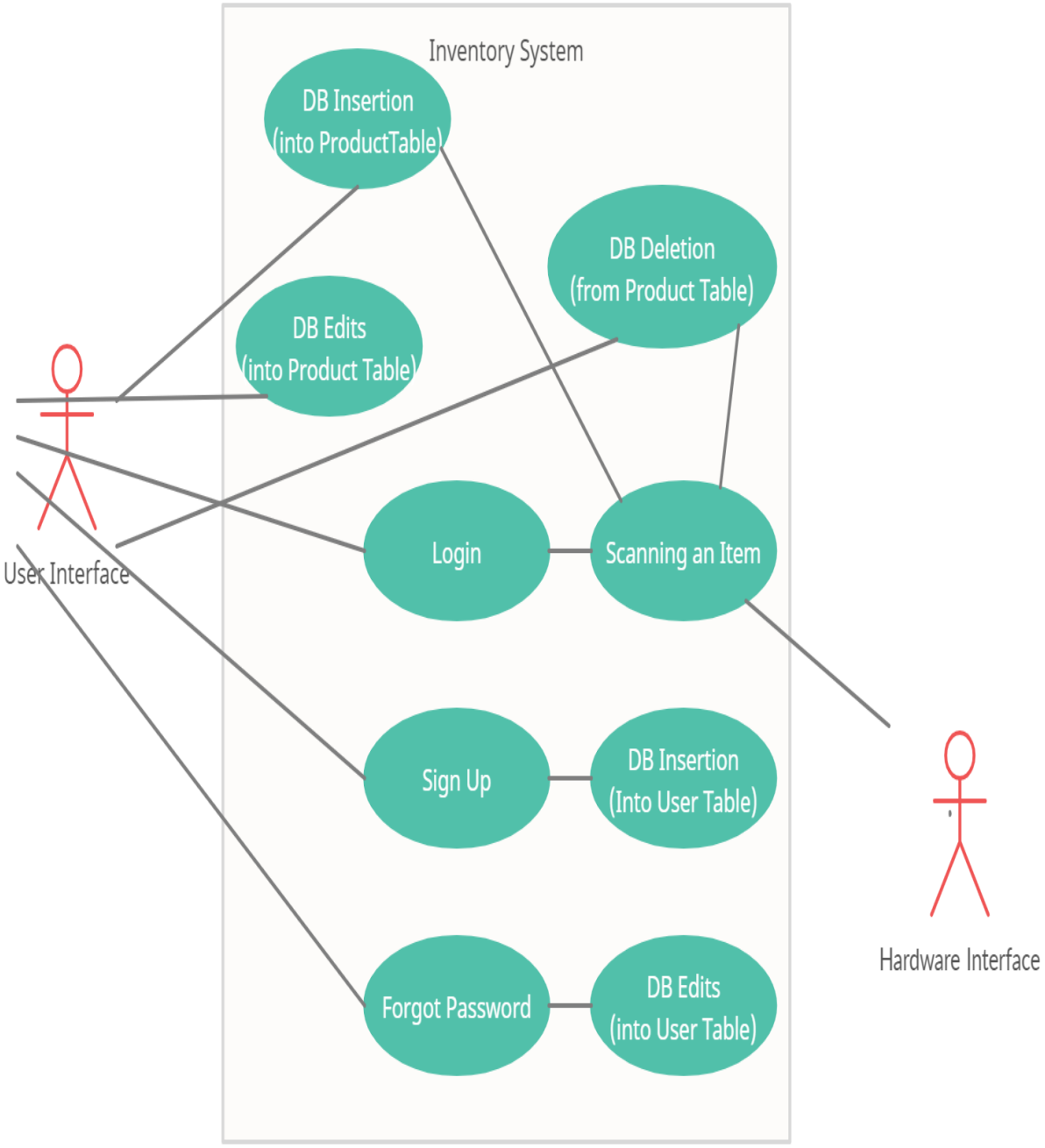


Diagram F- User Case Diagram

Another key element we opted to utilize for our project is a Use Case Diagram, which

displays how the system will work from different ends of the spectrum (from the application side and hardware side). As observed below in UI we have a representation of such workflow, where the user is able to engage with 6 areas from the UI - Login, Sign up and Forgot Password (all from the main screen) and the ability to do CRUD operations - adding, editing and deleting elements from the inventory (from the main screen). Meanwhile, the user can also make adjustments and interact with the hardware elements only when logged in, giving the user to scan items in and out of the inventory, depending on the positioning of the switch from the PCB. This is represented by the linkage between the Hardware Interface from the diagram and shows that items can only be added and deleted via the switch mechanism. We limit the user to make edits only via the application.

We also identified in the diagram which tables would be affected by the operations made to the DB. Since we have the user and product tables, these operations would affect different areas of work depending on which element the user is handling, through the hardware interface there is only access to the product insertions/deletions upon the signal that the user is actively logged in, otherwise the product shall not be registered in the app.

Having such diagram within our projects documentation is a huge benefit to the users and team members, since the Use Case diagram has the purpose of identifying the scope of the system, displaying the overall functionality from a technical point of view and informing the different capabilities the User has from the mobile application portion as well as the hardware section of the project. By containing this information, it makes the users and developers aware of how everything was constructed and how it all ties together.

6.4 Activity Diagram

Another software procedure we opted for this project was an Activity Diagram, as displayed in Diagram G. This process was meant to explain the different routes the user may take when utilizing the application. This process stems from the sign up on the software thread, whereas on the hardware thread users only utilize the scanner for the purpose of registering items and removing them from their inventory.

By deciding to create an activity diagram, this clarified the systematic procedures our software developers would withstand, by clearing the air on how the process would flow from start to end on a user’s perspective. When clarifying the process, the development becomes much easier as we have clear expectations on how the application should work for the users and the only room for improvement is the user friendliness design/architecture (rather than a systematic approach).

As shown in Diagram G, the green arrow represents the enabler for the hardware thread. Items will only be registered via login, which will identify the active scanner associated with the user (requested upon sign-up and can be changed at any time). This activity diagram helped us identify different portions of our project and how the SW and HW link together, allowing us to prepare ahead of time how these connections would be established, bringing us to the next topic, which explains how they will be integrated (HW and SW).

Diagram G – Activity Diagram

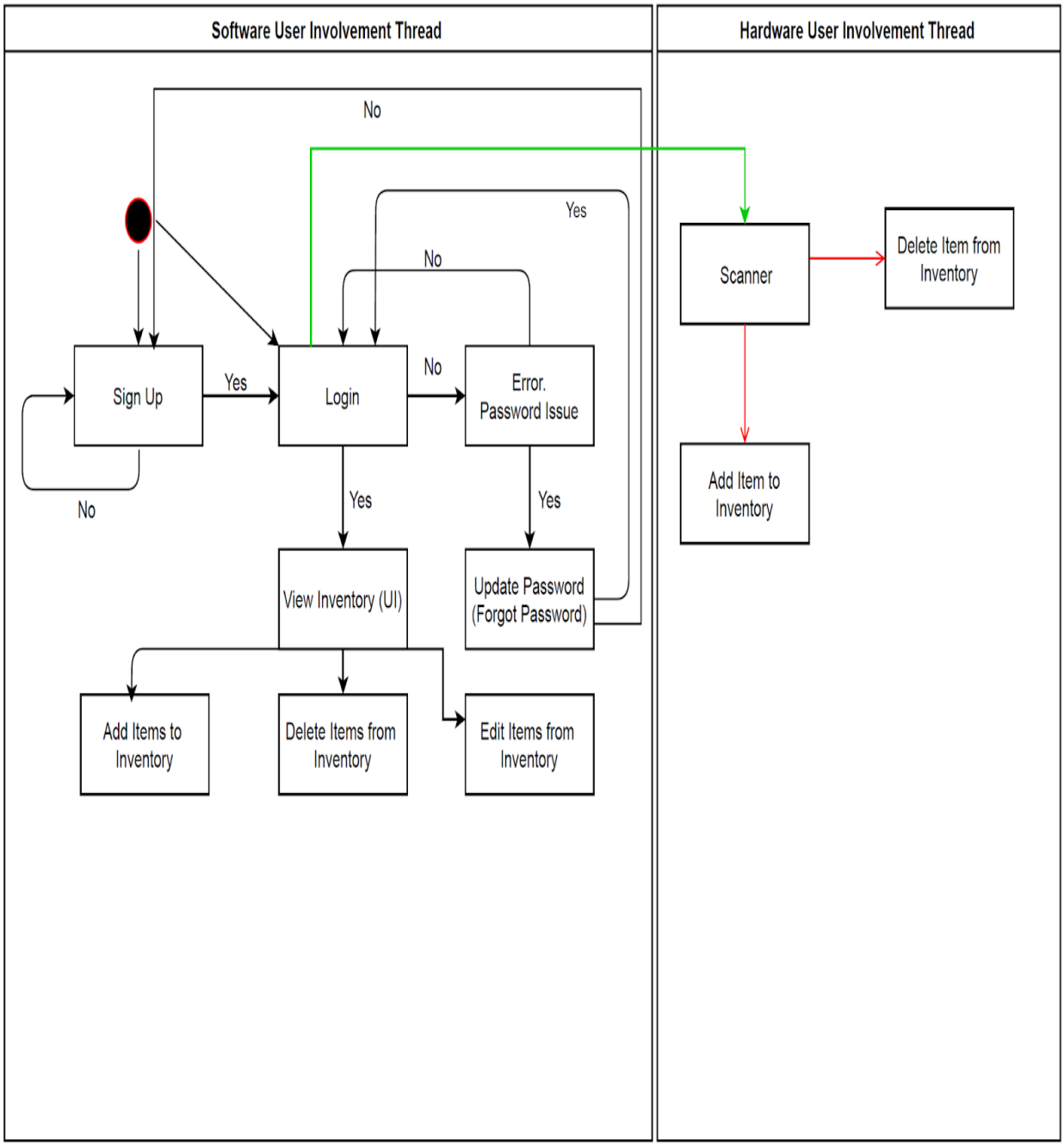


Diagram G - Activity Diagram

6.5 Entity Diagram

As observed below in Diagram H, we can notice the connections being made across the different tables. Within our User tables, we solely have user information needed to create an account within the application, such as the username, email, and password. Now, on the Inventory table, we have much more content, as it will hold many products of a single user. This data can be extracted from either the scanner or via the manual insertions in the mobile application. The mechanism we have resorted to was connecting the data via the users ID. This will allow for acknowledgment in the massive list of products within our database. We will know to whom the items belong to via the ID from the user's table. Though we did not have a solution to the reverse operation. What if the product is simply scanned, how would the database recognize the users ID? We solve this concern by ensuring the scanner is on a 1-1 ratio with the user. We will prompt scanner linkage on User Registration, meaning we will rely on the scanner's information linked to the user and will send this information for each product scan even though the user might not be signed in. Due to the linkage made on registration we will be able to filter through the users table and simply associate the products added via the scanner by checking who contains this given scanner. This will prevent any complications whether the product is added via the user interface or via the hardware bit of the project. Both methods will successfully be linked to the given user upon these circumstances, ensuring user-friendliness on both ends.

Diagram H – Entity Relationship Diagram

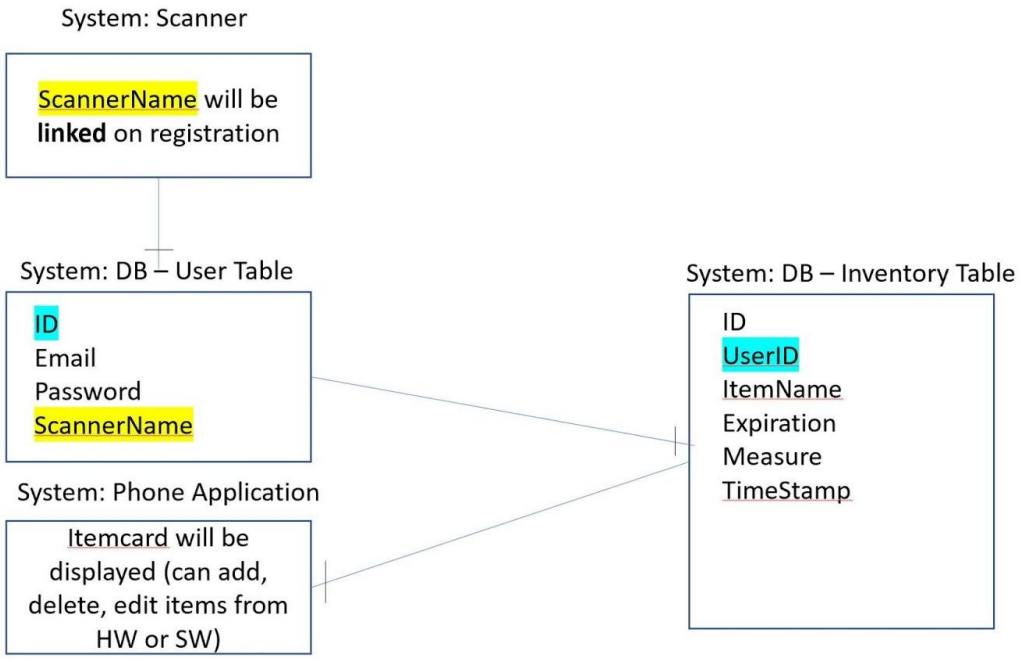


Diagram H - Entity Relationship Diagram

Diagram I – Foreign Key Relationships

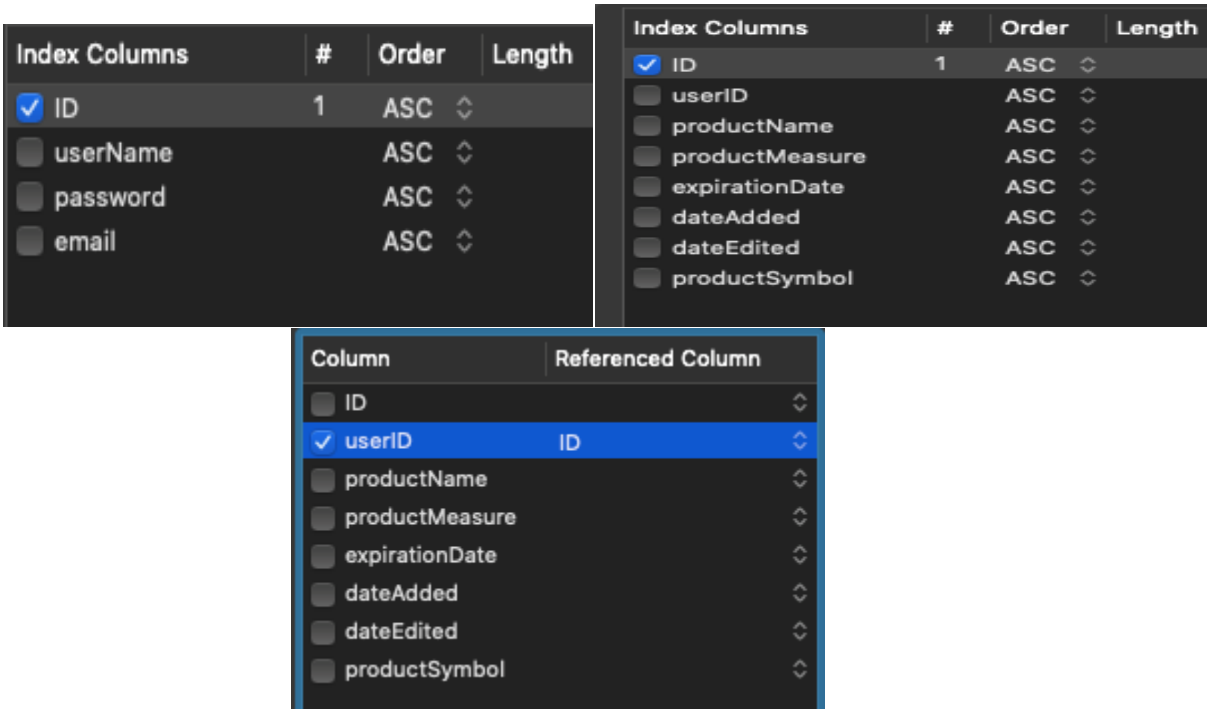


Diagram I - Foreign Key Relationships

6.6 Sequence Diagram

This is the sequence diagram for viewing an item. It has 4 stages which are user, application, database, and items. Here when a user requests to view an item in the inventory, the application will make a connection with the database. The database will then make a search query on the user's inventory and will be able to search for that item. This will then be returned to the application and the user will be able to view the item on their screen . Here they will be able to make any crud operations like edit, delete, update. Etc.

Diagram J - Sequence diagram for viewing an item.

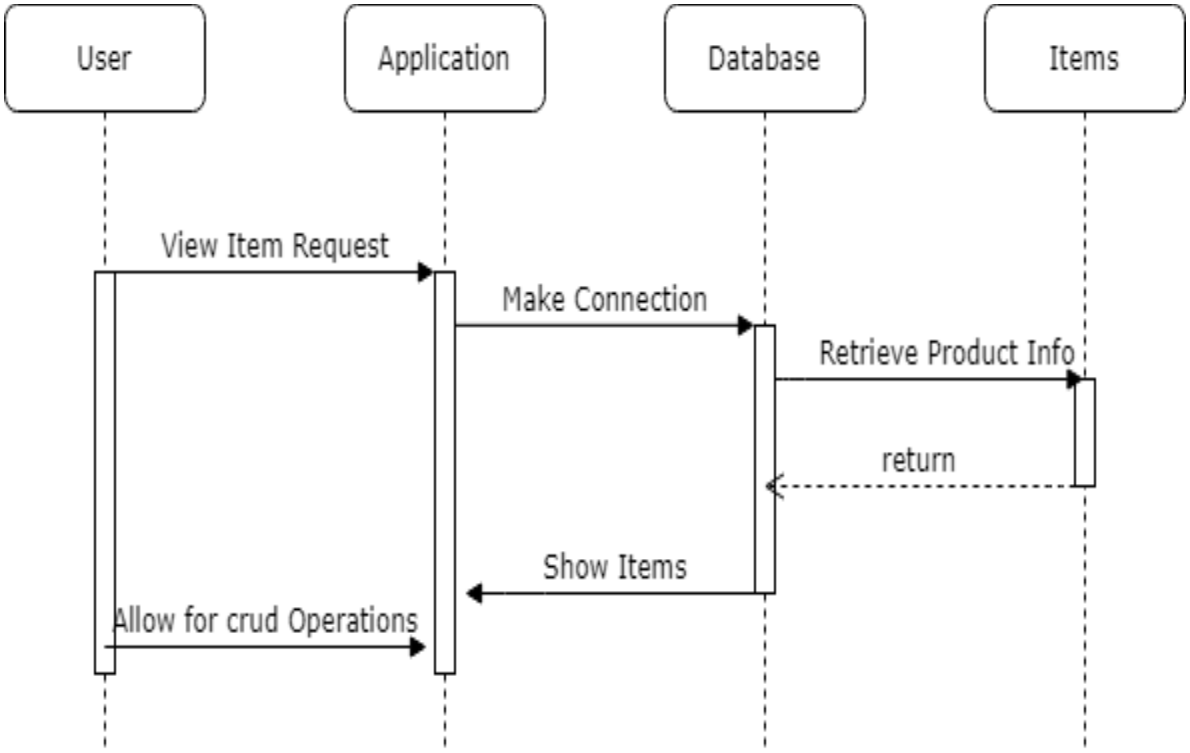


Diagram J - Sequence diagram for Viewing an Item

This is the sequence diagram that was created by us to explain the sequence of the application. There are 5 stages here which are user, application, database, apilookup, and items. Each of these are explained here.

User: This is the person that uses the application and in charge of making requests to the application. Users will make certain requests like creating items. View item. Delete item etc. These requests will need to be fulfilled by the app.

Application: This is the UI or the software that the user interacts with and gives requests to. The Application will have different views that can be used to create different requests which are made by the user. Application serves as the front end and will also interact with the backend to perform certain tasks to fulfill the requests made by the user.

Database: This is the backend of the application. We will store all the data here in form of tables. The users will also be authenticated here.

Apilookup: This is the service that we are using in order to provide us with the information that we need to store in our database. These service will scan in the barcode or search for the product information and return with results which will then further be stored in the database.

Items: These will be the actual items that are in a user's inventory. These items are stored in the database in the form of tables. There are multiple parameters that are associated with an item. When a user will make a request to edit, delete, or add items. This is where the actual change will occur.

Diagram K - Sequence Diagram for Application inventory:

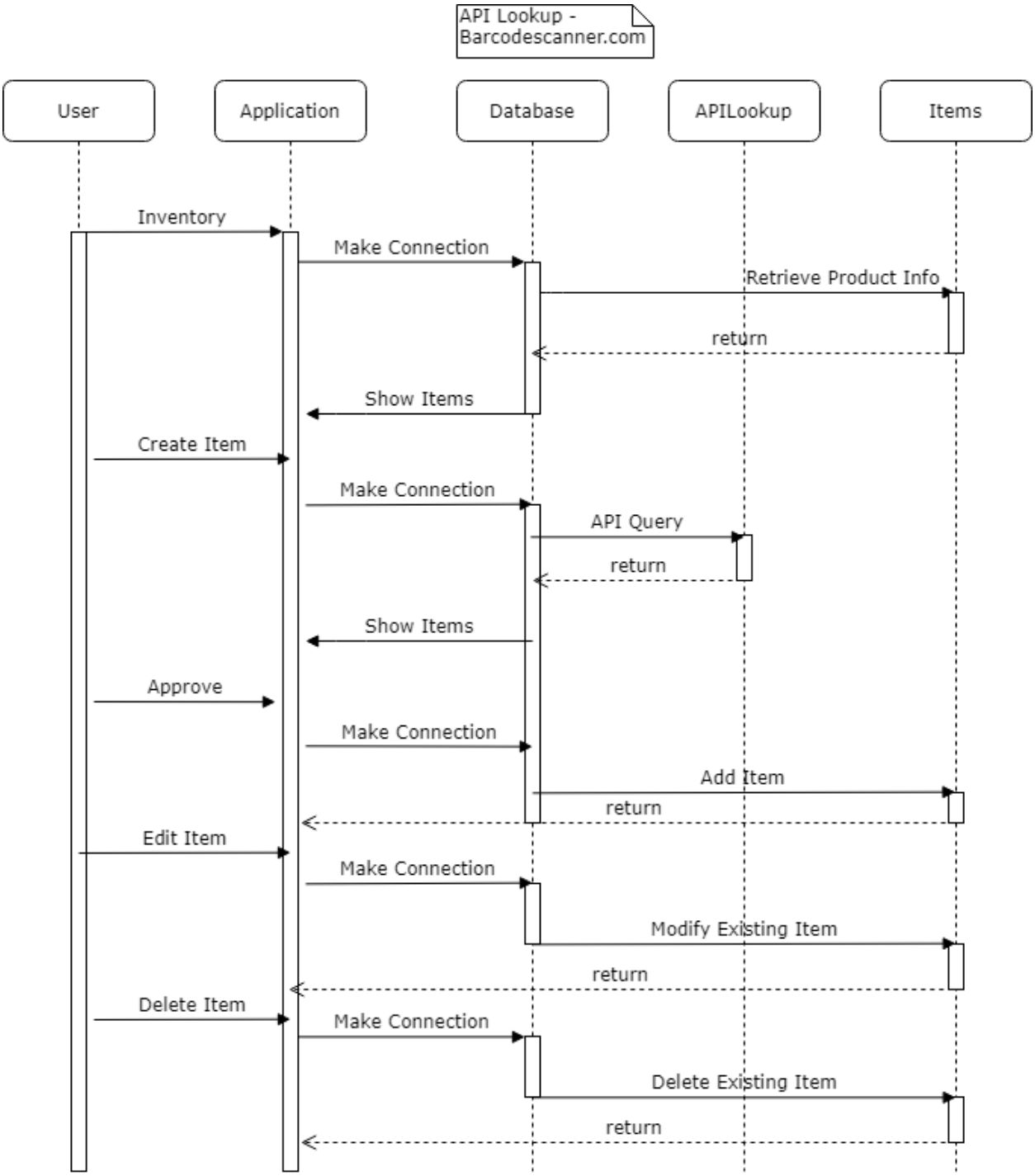


Diagram K = Sequence Diagram for Inventory

Now looking at the sequence diagram, we can see that when a user makes a request to view the inventory, the application will make a connection with the database. Thereafter, it will search for that user's account and retrieve the entire inventory with all the product information. This will be returned to the application and can be viewed on the inventory page of the application. When there is a request to create a new item by user they will enter all the product information and will hit the add items button, Now the application will again make connection with the database and database will make an API query to the barcode lookup service. This service will return with the result and the database will show that result in the Application. Now, users will have a choice to approve the item. Once the user approves the item, the application will again make connection with the database and will add the item to the database. This item will now be synced with the application and will be available in the inventory.

When a request is made to edit an item, what happens is that the application will make a connection with the database. The database will not search for that item in that user's inventory. It will then modify the parameters that are changed by the users in that item table. The item is then synced with the application and hence, users will be able to see that their item's values are now updated.

Similarly, users can make a request to delete an item. In this case, when a request is made to delete an item, the application will first ask the user to confirm the request that is made because there can be a case where the user marked the wrong item for deletion or accidentally clicked the delete button. Once the request is confirmed by the user, the application will connect with the database, find the item in the user's inventory and delete the entire item table. This way the item will not be in the inventory and after sync, it will not be visible in the application.

6.7 Class Diagram for Mobile Application

Diagram L – Class Diagram

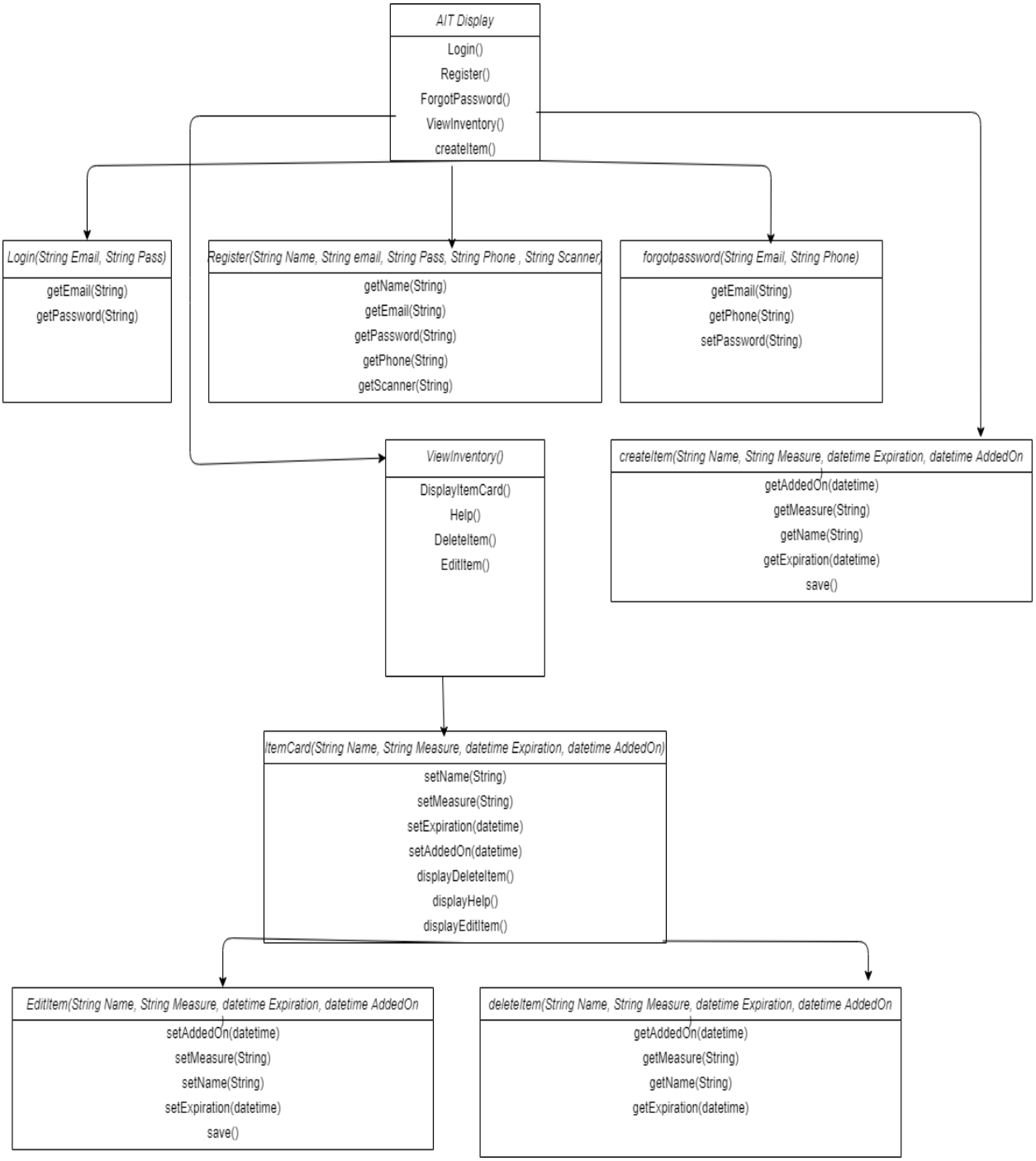


Diagram L - Class Diagram

6.7.1 Class Diagram Explained

The figure above is the class diagram that explains the class structure of our code. We have designed our code where the top most class will be the display of the app and different functionalities are given to different classes. Here is a detailed explanation of each class and how we are going to implement them.

AITDisplay(): This is the main class. AIT is the name of our project which stands for Automatic Inventory tracker. In this class, we have Login, Register, ForgotPassword, ViewInventory, and CreateItem sub-classes. We will be calling each of these classes based on user input. We will have different event listeners which will help us to route the flow as desired.

Login(): Our login will be taking two parameters which are email and password. We will get username and password from the user and then Authenticate the user to login to their account and view their inventory items. Here we will integrate Amazon's login services to authenticate the user.

Register(): Our register class will get five parameters as displayed in the class diagram. We need to take these inputs from the user so that we can create an account for the user on our server. The parameters that we will get are name, email, password, phone(optional), and scannerID. This is the place where we will register the scanner with the account. Once the scanner is registered with the account, we will use the ScanIN and ScanOut functionalities to add and remove the items from the inventory. For the password field, we will have a method that will ask the user to confirm the password by retyping it, once both the strings match, then we will send over all the data entered to the database and hence, create the account for the user. We will have a Register button at the bottom of the register page which will trigger these events.

ForgotPassword(): This class is specifically dedicated to the forgot password functionality. Here users will need to enter their email and phone(optional). The reason why we have phones here is because there will be possibilities where a user can lose their email. Once the user enters these values, and hits the reset button, we will need to verify whether that email is part of any accounts that we have on our database. If yes, then we will send the password reset email on their mail account.

ViewInventory(): This is also one of the major classes that we have. Once the authentication part is completed, the user will be redirected to this page where we will have the inventory displayed. This class has four different sub classes which are viewInventory, Help, DeleteItem, and EditItem.

CreateItem(): This is used to create items manually when the scanner is not able to scan the items as the barcode does not exist. It will get 4 parameters and will create an item table in the database upon hitting the save button.

ItemCard(): This is the item card class which will be used to create a card based layout to view each item in the inventory. This class will display the name, measure, expiration date, and Addedon parameters. Apart from this, the card will also have the functionality of deleting an item and editing an item.

EditItem(): This is a very similar class as create item. The difference here is only that we will not create a new database table for an item here but we will update the existing table with the new data as entered by the user.

deleteitem(): This will be a delete button on the card. User will only need to click the button and confirm the action. We will go to the corresponding table in the database and delete that table and hence, deleting the item. This will be similar to scan out functionality in the scanner but we will also provide this feature in the app as well.

7. Software

7.1 Planning Stage

7.1.1 Coding Plan

As we sought to develop our application, we initially took into consideration on existing projects that can benefit us by enlisting a wide range of utilities, such as ideas for the user interface, existing features other than product tracking and user-friendliness approaches. We searched any type of autonomous trackers, and encountered SMIDGE, which was also a senior design project from the University of Central Florida. They had a similar project scope to ours, but in terms of software they utilized both a web application portion as well as a mobile application one. After observing their strategies for testing, we incorporated some of their mechanisms, such as a variety of diagrams that breakdown the project plan from the very basic scope to a more detailistic approach, including the User Interface design. We planned out the entirety of the project's architecture, all of the database tables and columns we would use, and attempted to keep the project as clean and simplistic as possible. We optimized the scope of the project by having a separation of concerns when regarding the inventory elements and the user. These elements are grouped together, so the user has a much cleaner display on the initial load of the application.

Although we had a vast amount of features we sought to include, we were very limited due to time restrictions as we are planning to complete the project by the end of the Summer semester (July 2021). So our group of developers had to capture the absolute necessary features needed in the application, and the ones that are less important were ordered from topmost to least favorable feature to be added if time permits. Some features included: linking users, similarly to a social media page where one can add the other as friends and send notifications, another feature was to share recipes among users which could be enhanced by our inventory system by checking the active elements and triggering a reminder of which elements are missing for a selected recipe. We have a potential to expand this project into a new platform that can connect users and not only track elements, but also help one another by suggesting recipes and having tutorials on how the recipe was made. There isn’t an application like this in the market yet, but our goal was to keep it simplistic for the given time-frame and possibly expand on it after senior design is complete, by adding these mentioned features and give life to our original thoughts of making this a social platform as well.

In wake of the ideas generated by our team, we began by acknowledging the strong suit of every developer and the previous experiences such as the known tools and programming languages. As an outcome we opted for developing with the NativeScript framework as we would have the ability to have a test environment directly on our phone and the IDE would indicate whenever an error has occurred during the live testing. Another benefit we found was issuing a UI independent of the OS. So an android user would have the same view as an IOS or macOS user. Making the development much more focused on the user-friendliness, rather than having concerns elsewhere such as having to make either a working product for android or other operating systems, instead just having a generalized scope that will take into account any environment, capable of identifying the controls from one and replicate it for the other operating systems. We were also able to utilize a variety of different projects and view their forms and active templates. Being able to recycle that information was a key element for selecting this environment, since we would be able to focus primarily on the design and select exactly what we were envisioning, which is three layouts, one for the login page, another for signing up and the last one being the main user interface once logging in, displaying all of the items in the current users inventory. Being able to view the variety of options made us assess the cons and pros for each of the sample projects forms being reflected on our own, making us analyze closely each given situation and how it would benefit the user with each one.

We have strategized our next plan by initializing discussions on how the code would operate and tried to do some pseudocode on how the processes would be defined. We combined different ideas and perspectives and unified it all into a single rundown of how the application would work. This workflow was then elaborated into a wider schema, we studied different possible graphs and charts that could be beneficial to us when developing the application. Some of these include structure diagrams, schematics, use case diagrams, etc. These serve the purpose of obtaining a wide system and taking granular looks into the processes that contribute to the big picture. These software diagrams had a tremendous effect in our initial scope for development as we had already laid out all the requirements and workflow from start to finish, making the logistics much more concise.

Lastly, we took the decision of elaborating a detailed schedule for every bit of the software side of the project. These tiers included: the design of the project, database setup, initial form/GUI, software diagrams completion, etc. Among this schedule, we researched efficient mechanisms at a project level to minimize waste of time and maximize production, therefore finding a Gantt chart the most optimal route for our group.

Not only was the Gantt chart an enabler of effectiveness and productivity for our group on the software portion of the project, it was also decided to be used among the hardware as well. We took a very granular approach when adding topics to be completed and by when. Our scope of work seems to be very doable acknowledging the limited time-frame of the summer graduation. So we took that into account and added a section for our possible feature implementations after all of the debugging for the code and overall system is complete.

From the created Gantt chart, we sought to first complete the applications' overall design, the UML diagrams and database setup prior to any development. Although it took some extensive time of research, this route would clearly diminish any type of worry later on with

respect to flaws in our logic, preventing bugs from appearing in our system. As displayed in our house of quality diagram, we intended to facilitate the system and have an extremely user friendly application, so any user ranging from expertise and beginner level knowledge in the tech-field would be capable of utilizing our mobile application and have a beneficial time when utilizing it.

We have our environment setup, took a look at some sample projects utilizing NativeScript and have already begun the initial phase of software development, as all of the other phases prior to this category is already complete, including the UML diagrams that will help us understand the proper workflow whenever developing each portion of the application. We already worked out most of the logistics to prevent much issues that can potentially be found when debugging. We have our database elements already setup in SQL and began strategizing and separating what each individual developer will have in terms of the role for the project. Initially we sought the possibility of one generating the sign in and sign up, just having a UI with no functionality, while the other developer begins working on the cloud services portion of the sign up authentication.

This authentication would also take effect into the forgot password portion, so getting this setup would be a great accomplishment for the software team. Once completing these two assignments, we would move onto implementing the DB storage and testing the connection strings ensuring the proper functioning of both bits so this won't be an issue for our testing environment. As of now we have already set up all the local DB instances, although we do plan on moving this to a server once we begin pushing out from testing to the production phase of our application.

Another key element for our project that was decided for our coding phases from the initial plan was having a set repository in GitHub so that we can keep up with the version control that our software developers put out. We did not have any issues with finalizing this idea and have this in place, since we all had familiarity with GitHub and know that versioning is of the utmost importance when considering a software project.

In terms of branch management, we will be administering these issues by having two main branches, one called master (which will be the production version) and one called dev (with the purpose of adding features and fixing found bugs). We will then create any other branch to operate upon a given situation, such as implementing a feature that we consider to be necessary for the project or a found bug within our system, which will be handled among such branch. We will then do the process of merge requests to the dev environment so that we won't affect the branch currently operating in production. We have agreed that both of the developers must approve the merge request after a careful analysis of what has been changed, tested and after approval we would merge into dev. Depending on the urgency of the issue encountered or the feature needed, we would do a merge request from dev to master so the users can operate with the updated version upon the relaunching of the application.

If the user then experiences any issues upon production release, we will then handle this issue by restoring the previous version of master and take a closer look into the changes from dev brought into master. Version control was one of our top priorities in this project, since software is one of the heavy items within this design. We needed a guarantee that if anything had

an unexpected or introduced bug, this would not affect the working version at any costs. So we were very aware of the issues with only having a single version and dealing only with a single branch, causing us to think through and elaborating a proper system to mitigate the concerns we had discussed.

Upon the completion of the introductory displays such as sign up and sign in, we will begin polishing the GUI for the main display, we have laid out the logistics for all of the CRUD operations and the display layout, making the project much easier to handle due to the elaborate plan we had sketched out. We have already selected our data controls, similar to a listview where each inventory item will operate as a single row, containing the data associated to the item, such as the expiration date, product name, product details such as amount in g, L, mL, etc. There will also be a button attached to each row, for the purpose of deleting an item. The user can opt to handle the system either via the software or via the scanner, since the database will track which item is being referenced upon the scanning of the item. Therefore we have a workflow for the user to withstand either through the application or via the attached scanner. Although we intend to make this application self-explanatory, we will provide a tab with further instructions and documentation on the workflow.

These instructions will include a step by step procedure on how to insert products from a software standpoint, via manual creation through the application, as well as via the hardware through the scanner, which will put a queue on the database if and only if the user has associated the scanner with the user account through the sign up page or through the separate tab that links the account with a specified scanner. Even though the application will have few forms and be quite straightforward we decided to implement a section for the users with these instructions. These instructions will also be attached to this senior design paper, including from the very basics such as signing up, using the email authentication via the cloud services and how to reset the password. We have also decided to include labels inside the form itself, such as red asterisks “*” to indicate required fields. This user manual not only will have a software standpoint, but a hardware one as well, showing how to set up the scanner with the wifi modules, as well as getting the scanner linked up to the user account.

One positive aspect that drove us to sticking with the NativeScript framework was the wide usability it can withstand. Otherwise we would have to potentially make user manuals for an android user and for an IOS user, due to the distinction in buttons and positioning on the screen. We were aware of the upside that this framework is platform independent, meaning it will operate with the same control pertaining to the system the user is utilizing. The positioning of the screen will adjust accordingly, resizing depending on the platform used, making this developer-friendly as well. Instead of having to operate with Java and Swift, for android and IOS respectively, we can make a single consolidated application that can have its utility for both and work the same, restoring all of its properties.

For the purpose of code planning, one of the main drivers for what IDE and framework we were going to use, was the same ideology that drove us as top of priority in the house of quality sheet, which is user friendliness. Native framework excels in this field by doing a great job in terms of performance, even though it handles multiple platforms and is tested live by scanning a QR code. Testing is also very much facilitated, the developer can test different

designs and see how it looks on the screen of his/her phone. Then switch it and re-test in a matter of seconds straight to the phone, without losing the properties of the screen across platforms.

We were aiming to consolidate a user interface, with a good look and feel for the user within the first couple of weeks of development, being one of our main goals for the initial stage. We were successful in doing so and staying on track with our timeline to keep the momentum forward for the other tasks in the Gantt chart established. NativeScript since it is open source, was extremely powerful and helpful for us developers by giving a huge variety of resources to select from and begin observing which type of design would match what we were looking to replicate in our application.

Our developers began by trying out different projects and testing it on our mobile devices and were able to find some interesting forms and custom controls created by other NativeScript developers, this gave us a good start on implementing our very own user interface and provided a variety of resources to begin our forms. Due to our experience with web development, adapting to Native wasn't much of a struggle, since it has very similar attributes to when doing web applications.

According to our Gantt chart, we are in a good standpoint regarding our current status and we intend to keep meeting our deadlines ahead of schedule so we can produce potential features as mentioned previously. Based on the timeline, we are expected to complete full software development about 1 month ahead of schedule (not including the discussed features), because we are already expecting the need of troubleshooting and debugging any encountered flaw in logic within the system, as expected in any other software project.

By establishing our plans early, we were able to have a good rundown on all of the projects' intended capabilities, so by doing a pseudocode preparation, will help narrow down our logistics whenever implementing the actual code. Originally we had considered potentially including a tablet/screen to our project and having the app solely run via the tablet, although this would affect the scope of our project by affecting the cost of production, as well as user accessibility. We took these considerations and sought to minimize cost and maximize users interacting within our tool. So we had some brainstorming sessions and chose to make this project a mobile application as nowadays it is hard for a person not to have a smartphone readily available for using an app. We also took into consideration the smartphone type. We wanted to make this application as widespread as possible, so we thought of making separate applications, reusing the branches and keeping the same logistics for both platforms IOS and android. Though we encountered the NativeScript framework, capable of breaking the OS barrier and enabling the controls to be functional across different platforms. This was one of the many reasons why Native caught our attention. The debugging is also very developer-friendly by streamlining the process of testing and receiving error messages. The native playground was extremely pleasing to use, as the visuals can be tested immediately across any platform by using their native app and simply scanning a QR code to run the created project.

As far as planning, we have utilized previous software project experiences to bear in mind the potential of going wrong if the team does not collaborate early to meet the established schedule. Experiences from other classes at UCF, such as POOS, where we do a software based

project came in handy as we sought to utilize and trace similar strategies, having UML diagrams to help define all of our logistics before we even begin any sort of development.

After the software plan is complete, we intend to test the overall system and see whether there is any flaw from the hardware side of the project. We need the software and hardware to synchronize and enable the user to work from either end of the workflow, creating either items via scan in or via manual creation. Both of these processes should set a database element entry, so we will need to keep this in mind and test the functionality from both ends of the project. We will be able to utilize the UML diagrams which express the different threads - from software end to the hardware end, and view if the workflow is similarly expressed as to what we had defined in the earlier stages of the process.

In case of any complications, we have already established a time-frame for debugging both the hardware and software side of the project, so given the Gantt chart, we will be giving the proper time for testing and debugging, reason for which we have intended to complete the project one month prior to the summer ‘21 deadline.

7.1.2 Software Requirements Specifications

The software requirements are given in Table B. They highlight the requirements that are needed for the software to work. Here is a detailed explanation of each of those requirements. The first requirement states that software should be able to store the data to the database. This means that when a scan is done or an item is added, the software should be able to write that data into the database. The database allows upto 16TB worth of Data. The write should take approximately 5 seconds. The second requirement states that Software should have an API to retrieve product details. This is in regards to the API call made to the barcode lookup databases that are available online. It should roughly take 3-4 seconds to retrieve the data from the API.

The third requirement states that software should directly allow to write the item information from the scans. This means that if a scan occurs, it should automatically lookup from the online lookup database and write the entry in the database. The fourth requirements is about sorting. This means that we will need to sort the data by multiple parameters. It will be upto user to decide which sort they want. The fifth requirement states that software should be able to delete the data from the database. Here it is necessary because we want to keep the inventory updated. It should take 2 seconds to delete an entry from the database.

The sixth requirement states that the software should be able to link the hardware with bluetooth. This is necessary since we will need to register the hardware with the corresponding user account. Once the hardware is linked, it will have capabilities to scan and write or scan and delete. The seventh requirement is about sending the notifications. It is necessary to send the notifications to users once the items are about to expire. Users will need to allow the software to send notifications from their phone. The eight requirement states that the software should show the measure from 0-100%. This entry will be manually input by the user and software will show the remaining content once the entry is updated.

The ninth requirement is regarding the date/time picker. This means that the software should allow users to use the calendar module and add the expiration date if it is not available in

the barcode. The tenth requirement states that software should be able to run on multiple platforms. This software is a phone application and we want our application to run on the two major platforms which are IOS and Android. This way, we will be covering a large amount of the population who uses either of these platforms.

7.1.3 Software Design

7.1.3.1 Sign Up Page

Here, the detailed design of the software is explained. Starting with the Login Page. We have decided to keep this page simple and similar to login pages that users encounter in their daily lives. We are using two links. One to register and one to login. In the Register link, there are 4 fields. These four fields will be entered by users and we will grab them as a string. Now using Amazon's services, this data will be passed on to our database and where we will store the users account information. From the database, we will extract their email information and will use the services like MailGun or other email API services to send a confirmation email to users email stating that their account is created. Now, along with the confirmation email, we will send users a verification link which will be generated by Amazon services. Upon clicking that link, we will be able to verify that the user has access to that email. This will be the final step for users to create their account.

7.1.3.2 Login Page

Users who have already created their account will be using the Login button. Here they will be asked to enter their email and password they used when creating the account. The input in both of these fields will be taken as a string. We will be using Amazon's service once the user clicks the login button after entering the username and the password. Here we will be comparing the two strings email and password which the user enters with the same fields in the database. Once the strings match, the users will be able to log in to their accounts. Here the information which is sent from the app to the database will be encrypted with the help of Amazon's services. Since, this information is critical and sensitive, using an encryption will prevent eavesdropping. Talking about the strings, the email string will not be case sensitive. We will compare that string using an algorithm that accepts both Upper and lower case. For example, if a user enters their email as JoNDoe@eXample.Com , it will still be valid if the value in the database is jondoe@example.com. However, for the password, strings will be case sensitive so users will have to enter their passwords exactly as they entered when registering.

7.1.3.3 Forgot Password Link

There will be numerous times when users will forget their password due to various reasons. It is absolutely necessary to have a forgot password option available so that users have the ability to reset their password whenever necessary. If the user has forgotten their password, they can click on the password reset link. Upon clicking the link, users will be directed to a page where they will be prompted to enter the email that they used when registering for their account. When a user will enter their email and click on the Reset Password button provided below, they will be notified on the app to check their email for the reset password link. On the backend, our program will search for that email in the database and if the email exists, it will send a password reset email with the help of MailGun. We will be using Amazon's services to provide users with

the Reset link. Once the user receives the password reset email, they can click on the link or copy the url in their browser to open the link. The link will take them to a page where they can reset their password. Users will be asked to set a password according to the Password Requirements which were explained before. Upon successfully entering the password twice, users will be notified that their password is reset and they can now login with the updated password. On the backend, we will grab that string and update the password field associated with that email in the database with the new password entered by the user. Now, users can enter the new password and login successfully to their inventory.

7.1.3.4 Main Page

This page is where the users will be landed once they login. The design of this page is simple. We have a list view of cards on this page. Each card represents one item which is either scanned or manually entered by the users. The main reason to have a list here is because it is easier for the users to navigate through their inventory. At first we decided to have a page view for the main page. However, it is difficult for users to keep track of every item that way. On each card, there are 4 fields that are displayed. These fields here will be populated from the database if users are scanning a barcode which contains all this information. When a user scans the barcode, the API will grab the information and perform a lookup with one of the barcode lookup services, and return with the information such as Name, Measure, and expiration date. This information will then be parsed and stored in a Database table of products. Once, the products are added to the database, it will be synced with the application and displayed under the Main page. We will also take the timestamp of when the user scanned an item, this will be used for the fourth field Added on. This will help users to know when the items were added to their inventory.

We have a X button on each of the cards. This button is used to delete an item. There are several reasons where a user needs to delete an item. For instance, if an item is consumed or if an item is expired. In some cases, users might just throw away the item because it is no longer a need and occupying space in their inventory. Upon clicking the X button, the app will prompt the user for a final confirmation stating that “are you sure that you want to delete this item” with two options Yes and No. If clicked yes, the item will be deleted from the inventory. On the backend, we will make a delete call to the database for that corresponding item. The entry associated with that item in the database table will then be deleted. This way, it would no longer be available in the list view. Here is a flow chart (diagram L(which explains this

Diagram M - Deletion Process Flowchart

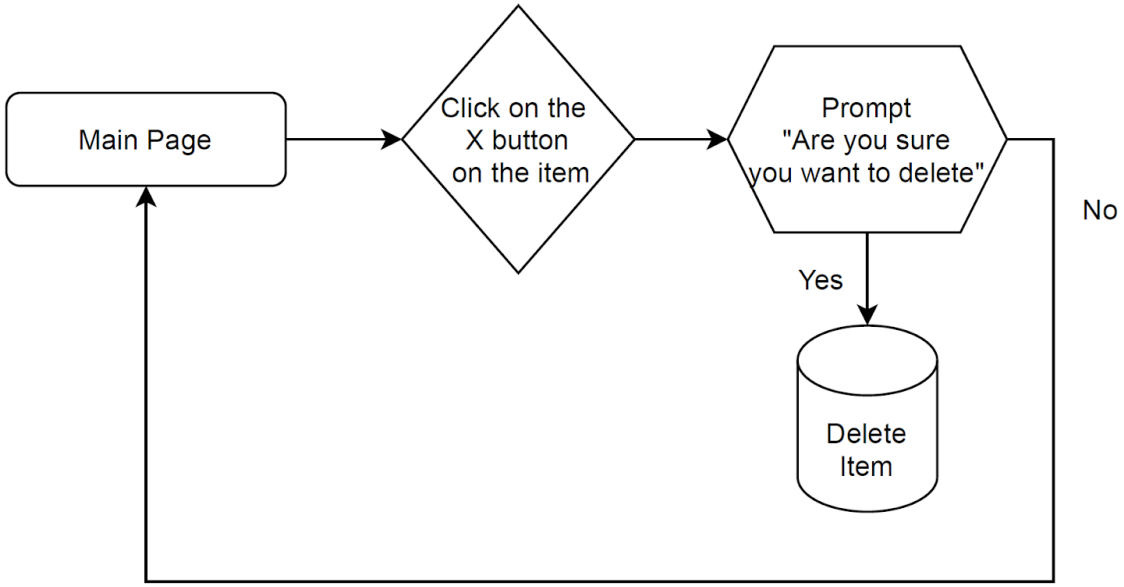


Diagram M - Deletion Process Flowchart

We will be using TypeScript, Html, and CSS to display each of the cards. There is a list view module in the NativeScript which will allow us to create the view. Then, using the card template, we can create each of these cards. We will create a CreateCard function which will be called every time an item is added to the inventory. Each time a card is created, we will add this card to the list view using the AddCard function. We will be designing each card to pull in the information from the database. This way we will have variables for each item and will use them to display the information on the card. Each of these cards will also be used as a button. Upon clicking this button, users will be directed to the edit item page of the inventory. We will be using an OnClick event each time a user clicks on the button. This event will trigger an action to open the Edit item page. We will use a sorting algorithm like quick sort or merge sort to sort the cards through different parameters. We can use Name of Item, Expiration Date, or Date Added fields as the parameters to sort our list. Users will have a choice to select how they want to sort their list. There will be a default sorting method which will be by the expiration date. This will enable users to have their items which are expiring the earliest to be displayed on their inventory at the top. We will need to parse the parameters from String to text for the sake of this operation. This way we will be able to sort the items.

7.1.3.5 Edit Page

This page will be needed for multiple reasons. Once an item is scanned using the barcode, it will Populate all the fields that are available from the barcode lookup. However, not all the barcodes will have all the information that we need. Some Barcodes like GS2 will return

us with the Name and Expiration Date and Measure which will be perfect since all the items are populated using this scenario. However, some barcodes will just return the Name of the items and the rest of the fields will need to be manually added to the inventory. In this case, the system will still create a card for the item but it will only have the Name and Date added fields populated.

In such scenarios, users can manually fill in the information by clicking on the card and going to the Edit Page. This page will have 4 labels and 3 input fields. Users can change the name of the item if necessary, add the fields like expiration date and measure of the item as well. The Added on field will be not editable since it is used from the timestamp when the item was added. Once all the fields are populated, then users can click on the Save button to update their items. On the backend, we will use the write method to update the fields in the database with the newly updated information for that table. Any information which is already there will be updated if there is a new item added to the field. This will enable us to update a field like name or expiration date if there is data there or add the data to the field if that field in the database is empty. When a user leaves a field empty when using the edit field page, that field will not be updated in the database. Hence, the system will display the data which was already present in the database or display blank if there was no data for that field in the database. Upon clicking the save button, users will be redirected to the previous page which is the main page automatically. In the OnClick Save button event, we will reroute the user to the main page. Users can also click on the Cancel button if they do not want to make any edits. This way users will be able to see their updated information in the inventory. Here is a flowchart explaining the same

Diagram N - Edit Page Flowchart

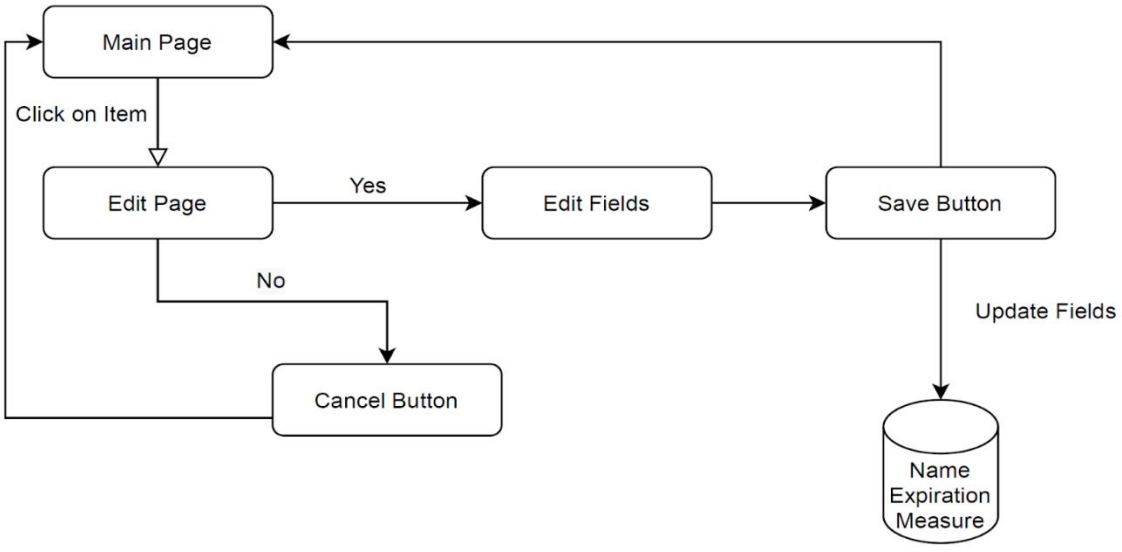


Diagram N - Edit Page Flowchart

We will use the Date calendar for the expiration field by using the calendar module of the native script. This way a calendar will be displayed and users can select their expiration date. There are two benefits of using this, one will be for better user experience since it will be easier for a user to select a date from the calendar instead of typing. The second reason is that when a field is populated using the calendar module, we will not need to worry about any typo's or discrepancies in date format which can be made by users. For example, some users will add a '/' between the date, month, and year whereas some users will add a '-' in between. This will cause problems for us to use that field for operations like sorting or sending notifications. However, if we use the Calendar module, we can be sure that the Expiration date parameter will be in a certain format and it will eliminate us from having to code for any extra cases like explained above.

7.1.3.6 Create Page

This page is used to add items in the inventory manually. There are multiple items in the inventory that will not have a barcode available. Some of the examples of this can be items like vegetables, fruits, etc. One more case that is possible are the leftovers from dishes that are cooked or brought from restaurants. Often, people will store these food items in their refrigerator. This type of items do not have a barcode and there needs to be a way to manually add these items. Hence, we will use a Create Item page for all these scenarios. There will be another tab or a '+' button on the main page from where users will be able to create items and add it to their inventory. This page will have a similar layout as the Edit Page. However, this time, we will use the Create button to create the time stamp and populate the Added on field. Users can enter the name of the time, its expiration date, and Measure using this page and click on the Create button to add the item to their inventory. Here, the name of the item will be a required field and users will not be able to click on the create button without populating the Name field. If they try to click on the Create button without populating the required field, they will be prompted with a message that the "field is required" in red text. The required field will be marked with a red asterisk as a standard so that users can know.

Diagram O - Create Page

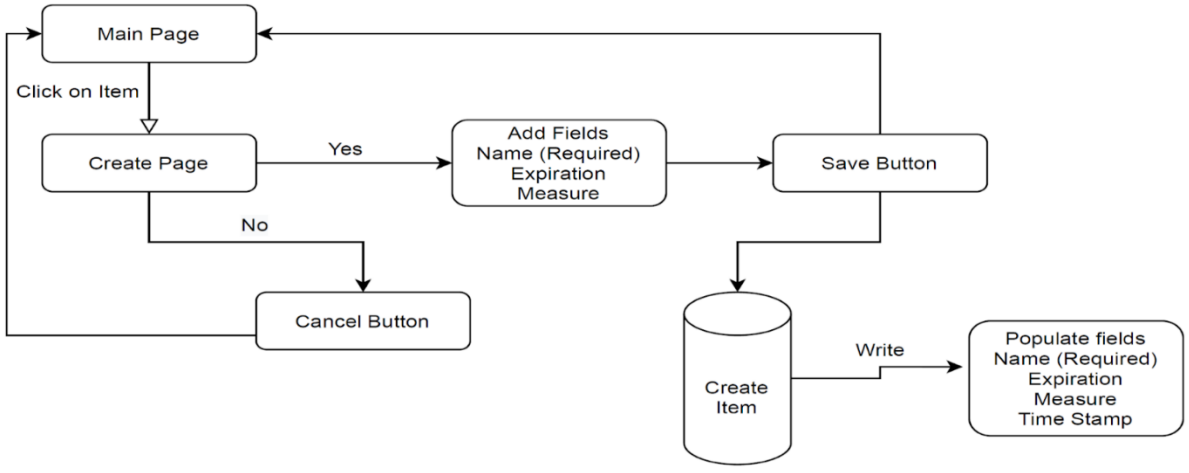


Diagram O - Create Page

Upon Clicking the Create button after populating all the fields, we will create a table for the item in the database and populate the fields from the inputs of the users. We will use a similar approach to write the fields as explained in the edit page section above. After Clicking the Create button, users will be redirected to the Main page using the similar action event as Edit Page. Once users are on the home page, they will be able to see their newly created item in the inventory.

7.1.3.7 Overall Design

This is the overall software design that we will use for our project. We will be consistent with the colors, size, and overall look of each component so that the look of the software is subtle and pleasing to the eye of the users. Our design's primary goal is simplicity. We want users to have a good experience when using our app and hence, we do not want to make things very complicated or complex. We have designed our software in a way that users will not have to learn things to use the software. They will automatically understand the flow of the software since it is designed in a way where a user can only do one thing at a given time. We have created separate pages for each task like Create Item, and Edit Items. This way users will not find the Main page too populated and confusing. Also, we are automatically redirecting the users to the main page once each task is completed. This way a user will not be confused on what to do or make an effort to go back to the main page each time they edit or create an item. The main page is a list view and users can scroll down to see the list of each item.

7.1.4 Product information retrieval via API

When we scan an item, we need a database where our API will lookup from the barcode and retrieve us the product details. We have researched a few barcode database lookup services that are explained here

7.1.4.1 Barcodelookup.com

Barcodelookup provides multiple parameters of a product such as information, price, photos just by scanning or typing in the product number. They have about 250 million products that can be looked up. They have their huge database that is sourced from retailers and commercial sites from all over the world. They currently support three formats of barcodes
UPC; Universal Product code that is used in the US, Canada, Australia, New-Zealand etc.
EAN: International Article Number which is used widely in Europe and other parts of the world.
ISBN: International Standard Book Number which is used for the lookup of books.

7.1.4.2 UPSitemdb

This database provides barcode lookup for 341 million products for UPC/EAN barcodes. They currently provide multiple plans which are free as well as paid. In their free plan, they allow a certain number of lookups per day i.e 100 lookups combined daily and upto 1 lookup every 30 seconds. The database provides their API and hence it is easier to integrate it into our environment.

7.1.4.3 Barcode Spider

Barcode Spider is another service that provides the similar functionality as the two mentioned before. This service also comes with an API which is an advantage. For the payments,

this service has three plans and the cheapest plan is $9.99 per month and it will allow for users to make about 1000 lookups per day and hence, it fits the budget.

7.1.5 Database architecture

To administrate the data from our inventory, we selected a relational database rather than non-relational since we have a structure in mind, therefore opting for a table-structured design. We will have two tables, one with the user information (ID, email, username and encrypted password) and the second with inventory details. This information will be a merge from API content (via scan) and user input for any updated information done manually (ID, User ID, product name, product measure, expiration date, date added and date updated). As observed, these tables will link to each other through the User ID, which is a foreign key, resulting from "ID" in the user info table. A simple representation of the data structure can be found below in diagram P.

For security purposes, the software developers shall encrypt the password as our team intends to utilize best software practices on this project. As seen in diagram P, we have identified our data structure as expected from a relational DB, and out of the variety of options such as PostgreSQL, Oracle DB, we opted to go with MySQL, as our software team has familiarity with MySQL (time management benefits) and being a practical, efficient DB.

Diagram P – Tables Relationship

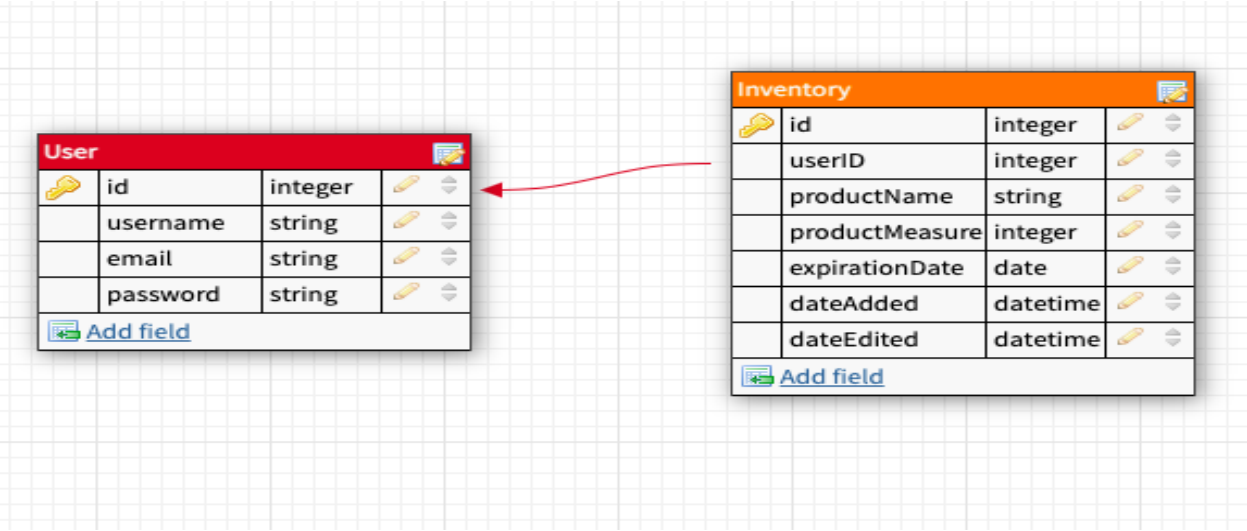


Diagram P - Tables Relationship

When selecting MySql over other DBs, we did much investigation to figure out what would be the most optimal for rapid back and forth transactions, as we expect the user to consistently be adding and deleting items. Some key advantages that made us opt for MySql is the flexibility and scalability, being useful from a small project up to massive loads of data, going up to an impressive amount of 2 TB, allowing for millions of rows of data to be stored

without an issue. MySql is also extremely popular and can be easily assessed when searching the web for debugging purposes, as it is very inexpensive and does the job in an efficient manner.

7.1.6 Programming Language/Framework Selection

For this application, we are planning to use a framework. There are many advantages of using a framework. Most Frameworks are open source and comes with licensing and hence it allows to build a commercial product. Frameworks often comes with good documentation and hence, it is easier to follow those when programming. Additionally, it prevents repetitive code and hence allows a programmer to build a project faster. Frameworks addresses many security risks and hence, giving programmers an ease. Finally, it is easier to integrate a framework for building any type of application.

We have some research done and decided that we will be using Angular for this project. Angular is an open source web application framework and it is TypeScript- based. It is developed by a team at Google and also by corporations and programmers in the community. Angular is built as a component-based allowing programmers to build scalable applications. It has well-integrated libraries, features like forms management etc. It also comes with a suite which helps programmers with building, testing, updating, and debugging.

Angular is written in typescript which is a higher version of JavaScript. It will fully compile javascript. For the front end, users can use HTML, and CSS. Our App will be compatible with both Android and IOS and for the cross platform compatibility, we will be using NativeScript. It supports cross-platform mobile applications with Angular and users can directly access all its APIs on Android or IOS. from Angular. This means that users can reuse CocoaPods and Android SDKs. Additionally, they can find free plugins , and samples from the marketplace. NativeScript is community driven and hence, a lot of help and support is available for the users from their forums. In addition. NativeScript is 100% free and is available as an open source tool.

7.1.7 Wifi Modules/Bluetooth Modules/Wired

For our application, we are planning to connect to our database server using the internet. For connectivity from the phone, users can connect either through wifi or through cellular data. They will automatically be connected to the services once they open the application. We will have a session timeout after 30 minutes of inactivity and users will be disconnected from the service. This will enable data saving and security purposes.

For establishing the connection between the scanning device and the application, we have 3 options in mind. The first is to have a wired connection from the device to the phone. This will enable fast transmission of data but is not a very user friendly way. The second option is to connect the device to the application using the WiFi. This is a good option, however, users will need to turn on the wireless hotspot from their devices and if they do not have cellular data service, the app will not connect to the cloud service. The third option is to have a bluetooth connection from the device to the app. By using this option, users will have the ability to be connected to the internet and at the same time, their device will be connected to the phone. Hence, as of now, we have planned to integrate a Bluetooth 5 adapter on our scanner. At times,

users will not have their applications open while they are scanning their items into their inventory.Bluetooth 5 is the latest major version of bluetooth which offers faster, reliable, and distant connection than its predecessor. It provides upto 2mbps speed which should be enough for the transmission necessary for our project.

We want users to have ease in terms of connectivity when they are using our app and hence, we have plans to use a Wifi adapter in the device as well. This will be connected directly to our servers and database enabling users to scan the products without opening the application. Once the items are scanned and entered into the database. Users can open their application at any time and the products will appear on their inventory after being synchronized with the database.

7.2 Software Implementation

7.2.1 DB Setup instructions (macOS)

1. Download mySql Workbench via the link https://downloads.mysql.com/archives/workbench/(DMG file) → (Note: use 8.0.21 since the most recent ones have bugs in the system)
2. Download mySql via the link https://dev.mysql.com/downloads/mysql/ (DMG file)
3. Go to system preferences, click on MySql, initialize DB (set up a password for DB)
4. Go to Sql Workbench and make a connection to the DB (root = name by default and apply the password you had set on step 3).
5. Utilize DB Schema from the 60 pg document (Includes User Table and Inventory Table)

Notes: For windows users utilize the same approach but apply “windows” instead of MacOS for the operating system option on the links for step 1 and 2.

7.2.2 User Interface (UI)

One of the main goals of our team is to create an app that is easy to navigate, user-friendliness is at the top of our priority when it comes to designing the front-end. We also care about the DB transactions and how quickly it will display to the user’s screen, but since the interface is the key to accessing inventory items, we want this to be accomplished by users from all ages, ranging from experienced tech-users to completely new members of this community. We intend to accomplish this with the layout found in diagram Q, R and S.

To facilitate our development, we decided to break down the design into separate subtasks, allowing us to brainstorm along the different sections to make this user interface as clean as possible for the users. Diagram Q specifically refers to the first form that will be seen by the users, the login screen. The users will have the ability to do three things on this form. Either login, which will bring them to the main form shown in diagram S, go to the sign up page (this field in the diagram contains the DB elements that shall be inserted) or forgot password page. For the forgot my password section, we shall implement an authentication link to ensure high end security for the users.

As shown in diagram S, we plan on having a tabular view for the main display, where the UI is split into the inventory and creation section. The inventory view includes all of the items with their respective descriptions, such as measures and expiration dates. The user will be allowed to make edits directly to the datagrid, and simply click on an X button for deletion

purposes. For items that were not able to be scanned, such as fruits and vegetables, the user may create/insert that item into the inventory by going to the creation tab as shown in Diagram S.

As observed, we tried to minimize the amount of elements within the form, to keep it simple for the users. The main page consists of a simplistic view of all current elements, as well as the capability of performing CRUD operations all in the same page after signing in.

By establishing a user-friendly UI, we will provide easy navigation and objectives will be straight to the point with no extraneous tabs of information. The main aspect of the application we are counting on is the processing of information, meaning the transactions back and forth from the DB to the UI. This will require proper integration of hardware and software so the process of scanning can directly trigger to the application via the bluetooth support. We are confident that MySql will be able to support our needs in this sense where we will have effective storage and transactions.

Going into the details, for the login page, when the user clicks on the app for the first time, they will be directed to the Login/Register page. For registration, we are planning to use Amazon's user authentication service for our apps to create user accounts. We will grab information such as First Name, Last Name, Email Address and Password from the users and store it in our database. We will then ask users to confirm their emails from the link that is sent to their email when they sign up. When they confirm the link, they will be able to login to the App. Our password will be encrypted and minimum requirements will be a minimum 8 characters which should include at least one Upper-case and one lower-case alphabet along with a number and a special character.

For the inventory page, we will be displaying a product in a card which will have its Name, Measure, Expiration, and Added on fields. Some or all of these fields will be retrieved from the barcode scans and it will be populated automatically. This depends on the type of barcode that we are scanning and the information it contains. The fields that are not populated automatically can be populated manually. Upon clicking on the card of the product, the user will be directed to another page where all the fields can be edited and the information can be added. We will also have a delete icon on each card which will allow users to delete their items when they complete their use. We will be integrating gestures in our app which means that if a user swipe left on their card, the product will be marked as deleted. We will be using pop up/prompts for users to confirm when an item is being deleted. Users can turn this feature on or off accordingly.

For the create page, users will have two options, either scan from barcode using the device or add a product manually. We will also be capturing the timestamp of each product when a product is scanned or entered which will allow us to populate the Added on field.

For the look and feel, we will be using light and subtle colors on the app so that the app looks professional. We will be using different font sizes for different information that is displayed on the app. This will allow for users to navigate the app easily and thus allowing a nice and smooth experience. We are currently in process to design an icon for the app and that icon will be used to open the app. Also, we will be using nice pictures and will also include a tutorial

on the app which will allow the users to learn and navigate to the app when using for the first time.

Diagram Q – Initial page

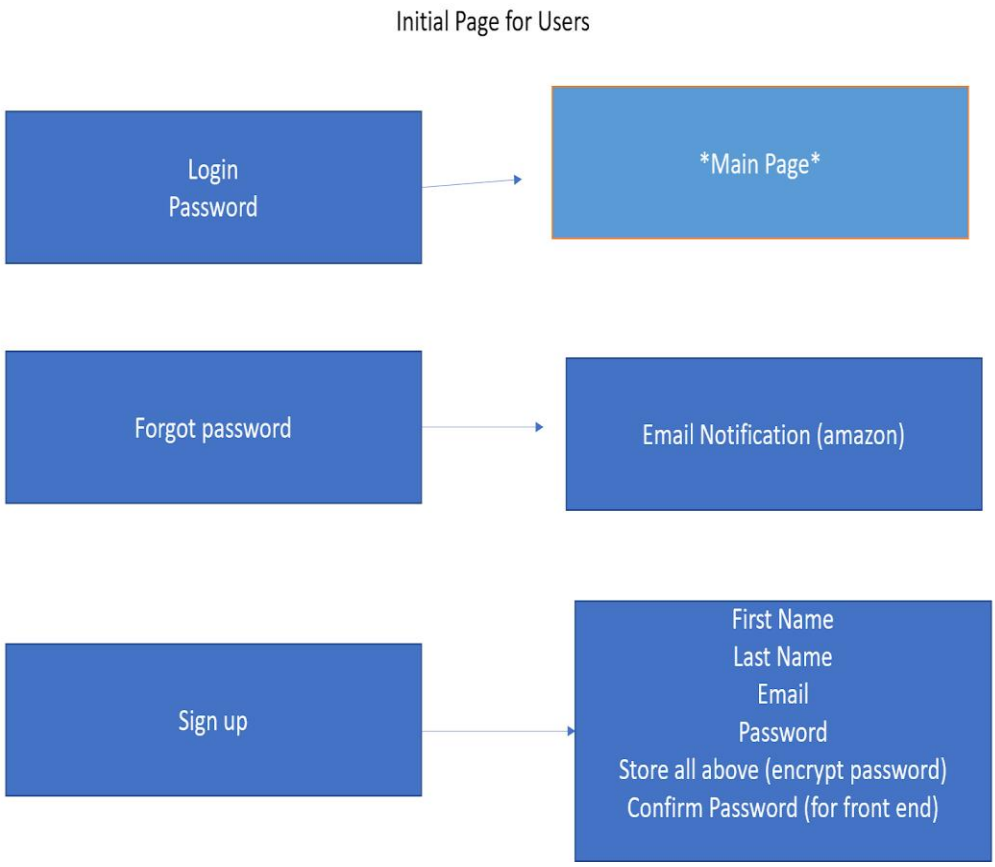


Diagram Q - Initial Page

Diagram R – Main Page

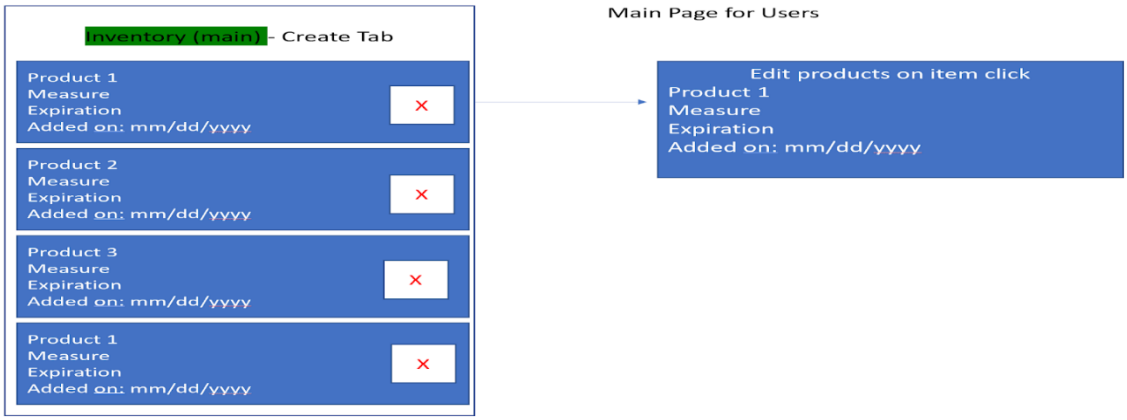


Diagram R - Main page

Diagram S – Create Item

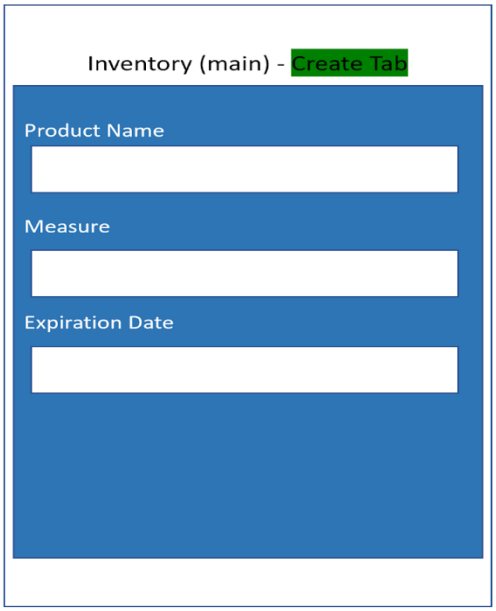


Diagram S - Create Item

7.2.3 Alert methods

Our App will be using the App notification system to provide alerts to the users. We will come up with a tune which will be specific to our app which will notify the users even when they are not looking at the screen. One of the main purposes of the notification system is to inform users when their products are about to expire. Currently, we have plans to notify the user when the item is about to expire in a month, week, and a day. There are multiple products in an inventory that have an expiry of more than a year and sometimes cannot be practically consumed within a day or a week and hence, we decided to send out notification when the product has 1 month remaining to expire. Additionally, we will be sending weekly notifications to the phones to all the users to remind them to scan their items if they are planning to buy groceries or have already bought them. Users will need to allow for the notifications services to be enabled from their phones in order for the app to send out notifications. If the user disables this option, they will not get any notifications.

Figure F - Barcodes

Figure F - Barcodes

7.2.4 Barcode Lookup API

The Application Programming Interface (API) is when a software intermediary allows communication between two different applications. In this case, we will need to do API calls in order to extract the product details from the inventory item scan. We will have the barcode from the product via the scanning process, therefore we are able to do a specific search on an external DB, that contains hundreds of thousands of products along with the product name, description and sometimes even expiration dates depending on which barcode is used.

Utilizing an API is essential for this project, since we need to retrieve all of the information associated with the item scanned and store such data into our very own DB. We need to capture this so we are capable of displaying the inventory associated with each individual user. Prior to selecting this project, we made sure this was possible by researching potential DB's out in the web that contain such information, and were fortunate enough to find a variety to select from.

If time permits, our group has also discussed the possibility of having a feature for our app which includes performing API calls to retrieve images associated with the items in the inventory, so we can display to the users a visual representation of all stored items within the fridge. By acknowledging the completed and potential tasks, we can observe the immeasurable amount of benefits API's can bring to the project, allowing for increased user-friendliness, plenty of new capabilities/features that can assist the application to get higher usage, etc. This all ties together with the original objectives we have with this application, making it very simple, user-friendly, with features/capabilities that are very straightforward so any type of user can understand how to utilize the app from the very start (from new tech-users to experienced ones).

In our project, this is a very important piece. When we scan our items through Barcodes, we will need to connect to an API service to get the results. Here is the explanation on how the APIs for barcode lookup works. The barcode detection API is the subset of Shape detection API. This means that it takes the share of the barcode and extracts the numerical value associated with that barcode which is then further used to search for products on a database. When a search

query is made through a barcode, the API will automatically search on the database and return the results. Generally, these results are returned in some programming language format such as JSON format. Hence, we will need to get that information and store it in our database. Every user account on the barcode lookup services are assigned with an API Key upon creating the account. This API key will be used for Authentication purposes.

There are multiple parameters that can be used to search for the product. These parameters can be Barcode, Name, mpn, asin, category, manufacturer, brand, etc. For our project we are going to use a barcode parameter. Hence, if we use UPC, EAN, or ISBN number that will automatically use the barcode parameter on the barcode lookup service. This will return us with one specific product associated with that barcode. If that product is not available on their database, it will return null. For other cases where the barcode is not used, more parameters mentioned above need to be used. When the number of parameters are more, it will return better refined results. There are multiple return codes that can notify the developer of what is going on. For example, there are codes for successful search, Invalid API key, No Data found, and search limits exceeded. These codes are useful to let the users know what happened when they performed a search. There are limits in the amount of searches that a person can do. This depends on the subscription being used by the account holder.

These kinds of barcode lookup services store millions of product information that can be used for people to get on a product. Users can go ahead and submit a barcode along with the product information on these services

7.2.5 Server

There are many types of servers like web-server, mail server, proxy server, print server, file server, database server etc. For our project, we are going to use a database server where we will store our database, and a cloud server for our application. We are looking at multiple options for cloud servers like Amazon AWS, Microsoft Azure, or Google APP Engine. Some of these are providing free server usage for students and we will take advantage of that. These cloud servers are virtual servers which are running in a cloud computing environment and hence, they are alternatively referred as virtual servers. These cloud servers can be accessed anywhere remotely with the internet since they are built and hosted on a cloud computing platform. These kinds of servers can host any software just like a physical server and they can also function as independent units. This type of server exists due to virtualization. Installing a hypervisor software on a physical server and extraction of their resources and then pooling them together will create virtual servers. These servers can be used by a single organization or can be shared between multiple organizations

The other server we will use is the database server, they are used to store and manage databases. These databases will provide the data access to the users who are authorized. One of the advantages of this kind of server is that the data stays in a central location and it can be backed up regularly. In an organization there can be many databases stored in one or group of servers which will be configured to handle client requests and protect the data. This data can be accessed by a UI which is created either by the organization or a third party. For example, we are using our application to create cards in the inventory to view this data on the database on our

app. Hence, the users will be interacting with the data through the UI which will be app but the actual data will be stored in the SQL server or Oracle which are in the database server.

We are using mySQL workbench to create the database tables and schemas. Once we have everything tested and ready, we will then look for our options to get a database server and store our database there. There are many services that can be used to access the database online like phpMyadmin, mySQL workbench etc. It is important that the server is properly configured and provides us the services that we need. For now, we are not looking for a very powerful server since this project will be used only by us for Senior Design. However, in the future, we will need to expand the resources on the server so more people can use our product concurrently. To set up the server, we will first need to get an Azure plan which provides upto $100 of resources free to students. Once, we have that we can go ahead and configure our server. This is where we will have our application and services running. The database server can also be created on this server. Once we create a database on the server, we will need to create a super user. This user will be given all the privileges in the database. We need to do this to properly administer the database and make our testing phase easy.

At first, we were planning to use a physical or a localhost server. But upon checking and thorough researching, we found out the it is much easier to get a cloud servers since there are many services available , and also getting a cloud server will enable us to work from anywhere remotely so we don't have to worry about coming together at one place inside the network or VPN to get access. Hence, we will be working on this plan going forward in SD2. It will only take us a few hours to configure these servers since most of the work on the database is already done locally.

7.3 Testing/Debug Approach

Testing is a very important phase of this project. If we look at the Gantt Chart, we have allocated almost 2 weeks of testing. We have divided our testing in a way that each component of the project can be tested. We will do Code testing, App testing, Database testing, and Server testing.

For Code testing, we will be using the Angular CLI with the Jasmine testing framework. We will also be using one for the test suites for testing purposes and make sure that our code passes all the tests. These tools make testing much more easier and organized and also takes care of many tests.

For App testing, we will be using the NativeScript’s Playground app. This app is available free on IOS App Store and Android Play Store. We will need to compile the latest version of our code in the NativeScript Environment. Thereafter, we will need to scan the QR code from the playground app from our phones and our application will be available to be tested on our phones.

To successfully test the application, we should pass the following tests:

- The application should be opened using the App Icon from the Phones

- should be able to Login Successfully
- Should be able to Register Successfully
- Should be able to Reset Password Successfully
- Should be able to view the inventory tab
- Should be able to see the products cards properly
- Should be able to click on the product and edit the information
- Should be able to delete a product
- Should be able to Create a new Product
- Should be able to successfully get the notifications
- Should be able to successfully exit
- Should be able to connect to the database

7.3.1 Database Testing:

Database testing is necessary because we need to be sure that we can rely on the search queries, validity of data, recovery of data, and be able to transmit the data securely. We will need to make sure that all the requirements that we have for the products are fulfilled correctly and make sure that data is inserted in the database correctly. We will need to make a test file and make sure that it passes all the scenarios given below:

- Should be able to add an item
- Should be able to remove an item
- Get item information if that item exist
- Change the measure of the item
- Create a list of all the items
- Create a list of all the items from and to a specific date
- Should be able to respond to the API calls
- Should be able to send error messages to the server.

7.3.2 Server Testing:

We need to test the web server to make sure that we have connectivity and without it, we will not be able to use our product. In order to successfully test the web server we should be able to pass the following tests.

- Connect to the internet via DHCP
- Should be able to ping the server
- Should be accessible by external networks
- Should be able to test the database from the server
- Should have able to restart and shut down
- Should be able to reach via IP

7.3.3 Debugging:

Debugging is an essential part of development and whist testing, we will come across many bugs in the project that we will need to debug. We will be using a debugger to allow us fast debugging. The environments that we are using come with their debugger. We will also use a log file where all the activity is being logged and this is where we will write our errors from the

console. By doing this, it will be easier for us to identify where in our project, we are getting any errors along with their description. This will help us with faster diagnostics and will allow us to focus more on the error rather than finding the error. For databases and servers, we will use the same procedures as well. If there are any logical errors, we will be able to identify them faster since we will be using a framework and our code will be organized.

7.3.4 Testing Environment

For our testing environment, as we had discussed, we will be utilizing the NativeScript playground, which allows us to test the application from either Android or IOS. We had initially discussed the IDE which made the most sense. Visual Studio Code was one of our main options, but we chose the Native environment because we would be able to have much more visuals of the project and since there are so many projects out there, we would be able to establish a much cleaner design with a nice feel to it for the users.

Although Visual Studio Code would get the job done and we all had familiarity/experience with it, we wanted to learn a new skill in the process of creating this application. From our experience so far, testing in the Native framework has been a very neat experience. We were capable of looking at a wide variety of sample projects that other users have created and have an idea of how they were capable of adjusting the controls to shape it into their very own. We saw neat features such as custom control creation and adaptation to the purpose of their project. So we saw some slight ramification of the list view control we had previous experiences on, with a merged button for CRUD operation purposes, which is actually what we had in mind for the product details view from the main page.

The testing environment was very clean, we observed how it would operate across different operating systems as this was a major concern for us, as we wanted to have an established application that could benefit from the variety of OS's NativeScript framework could offer. So the look and feel was a major success so we invested our time into learning this new system to initiate our very own project from it. Since we focused primarily on the user-friendliness and set that as one of our topmost priorities for this software project, after observing how hassle-free it is to adjust the look and feel of the project, we stuck to this framework and have begun implementing the UI given the variety of projects available.

Another benefit we were able to attain from Native was that the project layout was well established, giving us an idea on how to implement the proper designs, which was not very difficult to get used to. We were able to find projects that we could benefit from, such as barcode scanners, so by acknowledging the existence of a project similar to what we envisioned, helped us boost our confidence in getting to implement something along the same lines. We were able to test these projects on the spot, but simply scanning a QR code and linking our phone to the project in Native via the playground app. This made it possible to test out all of the functionalities of the given projects that were already out as sample projects from the wide range of users that are fully engaged with this environment.

Despite not being completely familiar with Native, these benefits mentioned helped enhance our confidence that this is the environment we wanted to engage with. There are a

variety of benefits we were able to acquire, although, the main one is the ability to run this application via Android or IOS, keeping the same look and feel from the actual operating system, with the same positioning of each control with respect to the positioning on the screen, which will help us achieve the objective of wide-spreading this mobile application independent of the platform of the user.

One major concern we had when opting for this environment was any type of performance issues, since it would be dealing with multiple platforms, maybe the runtime would be affected, or potentially even the button would not be accurately displayed on the set position since each platform type has its own size. So these were some of the main issues we considered. We have done extensive research and a variety of users have expressed frustration with some of the versions of Native, mentioning it was very slow and not worth it due to such constraints. Based on our findings, the current version, which is NativeScript 12.14 is very consistent and performant, it does a great job at maintaining the same look and feel across each platform based on our tests from IOS and android devices.

To us, we considered runtime, but as mentioned in the house of quality of our project, this is not a major concern as we are not intending to make an ultra-fast application. Our focus is primarily getting the job done in the most user-friendly way possible, meaning the functionalities are all accurate, such as the CRUD operations. Having a clean design, with not too much of a crowded screen. We want a very simplistic app, where anyone can benefit from using. Efficiency is definitely one of our goals, we do want the application to be rapid, but we are not dealing with an enterprise level of data, so optimization in terms of our production database and accessing memory shouldn’t be a major concern in this project as there won’t be extreme amounts of data, so runtime will be fairly rapid compared to a regular application that deals with DB connections.

The way the preview of your application works is by completing all of the development of the project then scanning a QR code to access the application from the mobile device. We are able to do so as well with other projects, as a way of testing and even learning about how the code is operating. Below is an image of how NativeScript operates (figure H) and the QR code display (figure G) on how we access the application and get it previewed from our phone.

Figure G - QR Code Native Playground

Figure G - QR code Native

Figure H - Native architecture

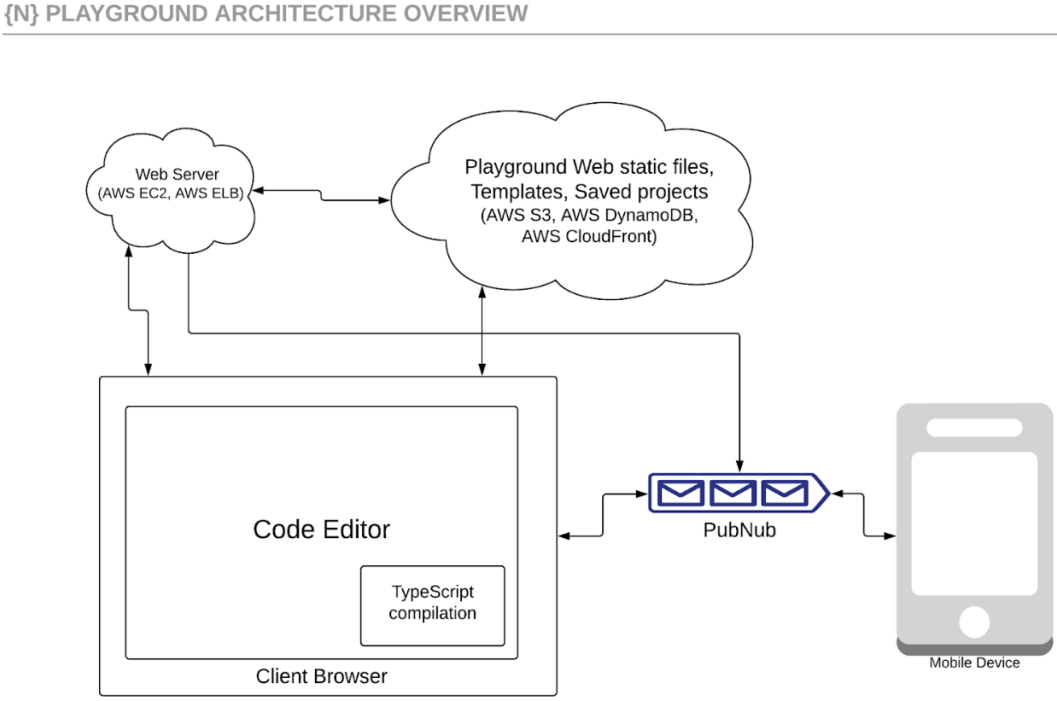


Figure H - Native Architechture

7.4 Performance

Regarding the plans for the performance of our app, given SQL servers' metrics on query time, it indicates a query should take 20-500 ms to get retrieved. As we don't have too much information to be pulling, our goal is to maintain a maximum amount of 750 ms per load. We will not be refreshing the app every X amount of seconds. Instead, we will be having a refresh button that will allow the user to update as soon as they want a refreshed list of their inventory items.

As far as the hardware, we are estimating a 900 ms timeframe for the scanner to reach our sql server via the wifi module. We are taking into account any performance issues with the users active wifi session, which can potentially affect this estimate by a margin of 1-2 seconds depending on the status of the connection.

Considering the load would take 750 ms, we estimate the refresh to take around 500 ms, since we will be optimizing the runtime by just comparing memory with the list of items from the database, instead of removing all the items and adding each one again to the listview control.

The cloud services authentication for forgetting the password or for signing up will take between 900 ms and 1800 ms to reach out to the provided email and confirm the access code provided. Making this type of request is a little bit lengthy, but is understandable as it needs to ping the DB for the users' email information prior to performing the email sent with the generated code.

Lastly, we need to take into account the sign up portion compared to sign in, so this process of insert into sql before having a compare whenever attempting to login will take just around 150-400 ms as found on the sql website via previous metrics. Therefore by the time the user types their email and password, the sign in should be taken into effect and have a successful attempt when trying to login to their account for the very first time

8. Software and Hardware Integration

8.1 Idea Explanation for HW and SW connection

Make changes to DB diagram and include "ScannerName" as a column

1. HW will have wifi capabilities making it possible to reach the server
2. On mobile app, force users when signing up to link a scanner to their account (ScannerName) via a scan on barcode or typing
3. DB will check device trying to ping server and we can allow device name to match "ScannerName"

8.2 Project Testing Overview (SW and HW)

For the Autonomous Inventory Tracker, we aim to begin testing the whole system about 1 month prior to the summer ‘21 deadline, so we will begin this process around June ‘21. Although, we will be focusing on testing and achieving positive results as we are developing and testing the bits that we have just implemented.

As discussed previously in our Coding Plan section, we will be separating our software and system testing into different stages. Initially, we will be working on the user sign up page and the sign in as well. Once completed all of the GUI elements, cloud services and DB connections, we will begin testing in our local DB to ensure we have proper results, such as encrypting the password properly, all of the data elements are properly stored, etc. Next, we will focus on the creation of the main form, attaining all of the elements from the DB on load and be able to capture all of the users inventory items and product details such as expiration dates, product quantity, etc. By creating items and testing all of the CRUD operations from the main form we will already be doing some sort of minor testing, but the main bulk of ensuring the system is properly functioning will be near the end, 1 month prior to the demonstration of the functional app to the UCF Engineering committee.

We have laid out a more in depth look at what the schedule lies for us in our Gantt chart. The hardware team will be dealing with the PCB and testing each of the components individually prior to setting up the system. We have given some time just to ensure the components all work properly as we do not expect them to all be functional from the first get-go.

From the software side of the project, we will have a backlog of issues within GitLab, allowing us to be aware of any debugging and code-fixes that need to be implemented within our project prior to launch that we have found via testing among the group. This backlog theoretically will be filled with items to do (features and bugs), we already have all of the necessary features inside our backlog, so for each issue the developers decide to get involved in, requires them to branch off of our master branch and after both developers accept the merge request, then this branch will get merged into dev. Depending on the urgency of the bug or feature, we will release a merge request from dev into master. Version control is extremely valuable in the software scene, especially in our project as we won’t have to be daunted by the fact that there is a major issue out on production. This will only occur if we do not do a good job at testing the branch we passed into dev. That is why testing is going to play a critical role in the success of our project. We will need to do thorough testing prior to each merge request into dev and focus even more when the time comes to merge dev into master as master will be our main branch that will be published out to the users linked to our production database from a server. For the purpose of utilizing dev, we will be running all of those operations within our local database, which we already have set up and functioning, including all of the primary keys and foreign key references from the user table to the product inventory table.

As we are primarily using version control via GitHub to track all of our changes, testing will be much smoother as we may see the reflected changes and provide comments on it in case there is a disagreement on how a process is being affected. GitHub is a very valuable tool, especially for the software side of the project, this makes us not have to stress on testing the main

branch (master) all the time, as we will barely be making changes to it, only to outside branches addressing particular bugs or features from our backlog and those will be merged into dev before even making its way over to master.

We opted to have a separation of concerns for our project regarding the hardware and software end, since we will individually ensure its accuracy and precision, prior to doing the final merge of both, via integration. Our strategy for integrating both hardware and software was challenging at first, we had to discuss many potential solutions as the scanner is readily available at all times, while the mobile application is not (by this I mean the user needs to be logged in, otherwise it will not register the scans from the hardware end). So our solution to this issue, which we plan on implementing whenever we have the opportunity to merge both bits of the project, is to register a scanner at the sign in phase. Therefore, a scanner is linked to a single user, which will allow for scans even though the user is not signed in. The scanned items will be placed in a queue, pending sign in to register all of the items to the DB so the user will view them as soon as logging in to the mobile app and the items will be displayed in the control. We have modified our GUI layout to accommodate this situation and have a tab where we will allow for users to link their account to a new scanner.

This method of linking an account with a scanner works by capturing the scanner information upon the sign in, therefore this information is registered into our DB, allowing for us to associate a scanner device (field in the users table) and link it to the product scanned. So this process will first register the product, including the details tied to the product upon the initial scan. Next, the DB will have a trigger via the “ScannedBy” field from product details and this should match a scanner device from the users table. If this is not true, then the product will get deleted since it is a scanner not associated to a user, otherwise it will sync up the data to the user who has such a scanner in place from the setup section. Otherwise if the user ever changes scanner or wants to share with another user, this may be done by the re-linking tab which we have implemented into our user interface.

Lastly, we will be verifying whether this whole workflow is successfully taking place. The system to be fully functional will require the separation of concerns, meaning it will need to first work individually, the workflow of solely software working primarily from the mobile app should be fully functional, the PCB, scanner connections should also be working properly prior to doing this whole merge. Testing each section will be a challenge, but with proper timing and adjustments as established in our Gantt chart, this process will be facilitated.

Upon successfully testing both ends, the whole system should now be ready to integrate and we will have to utilize our UML diagrams, which express the users workflow from concurrent threads of software and hardware (vice-versa), as the user may opt to perform CRUD operations via either the app or the hardware end via scans. So as it depends on the users’ discretion, we will need to properly test the whole system and ensure that the overarching project is working in the way we would expect it to based on the diagram workflows created.

9. Hardware

9.1 Hardware Planning

9.1.1 Development Boards

9.1.1.2 ASUS Tinker Board

The first development board we are considering is the ASUS Tinker Board. This development board has a Rockchip RK3288 SoC (System-on-a-chip) with a quad core ARM Cortex A17 CPU which runs at 1.8 GHz. As expected with a system-on-a-chip usually comes with a GPU (Graphics Processing Unit). The GPU on here comes with an ARM Mali T760 GPU running at 600 MHz. Also, this board has a full-sized HDMI 1.4 port, supporting up to 4K/30fps and next to the HDMI port there is a micro USB port for 5 Volt power sources. On the other side of the HDMI port there is a 3.5-millimeter Audio output TRRS jack, used for audio output or a microphone input, that goes up to 192 KHz/24 bit. Looking at the top end of the board there is a 1 GB ethernet port and next to that there are 4 USB 2.0 ports. The Tinker Board also provides 40 GPIO connector pins on the top edge of the board with color coding for less confusion when connecting the right connectors.

There is a DSI port on the other end of the top of the board, where LCD displays can be connected and a CSI port around the middle of the port for cameras. This board also comes with a metal shielded (It's a circuit board on a circuit board) WIFI/Bluetooth 4 (wireless BGN) chip that is found on top of the board with a built-in antenna near the DSI port includes hardware decoder of H.264 and H.265.

Under the board, the user can find the two RAM chips of 2GB RAM of DDR3; the micro SD card UHS-1 slot is, as expected, under the board as well. Since the Tinker Board does not have any onboard flash memory, the user must also purchase a micro SD card in order to use this development board. When a user wants to program this board, they will have to save their operating system on the SD card from the computer and then insert the SD card to the Tinker Board.

There is a WIFI/Bluetooth module on the top of the Tinker Board connected to an antenna that the user can replace for a different antenna of their choice if the user so chooses to. This Tinker Board comes with a display connector for 7-inch LCD screens and there is also a camera connector.

The Tinker Board comes with a manual and a heatsink, which is a bit unusual for a single board computer but the manual advises the user to use the heat sink on top of the system-on-the-chip on the Tinker Board because the chip will increase in temperature to 145 degrees Fahrenheit (62 degrees centigrade), and that could accidently burn the user if the heat sink isn't applied.

The Tinker Board comes with a full-size HDMI port, a 3.5-millimeter jack, a display/camera connector, and 40 pin GPIO connectors. The board also includes four mounting holes to place the development board in a protective case.

Figure I – TinkerBoard description

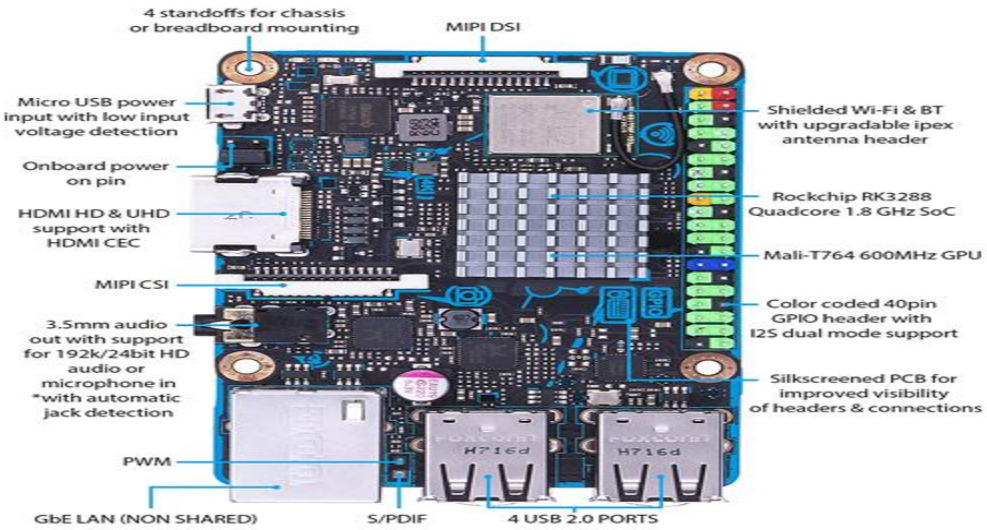


Figure I - TinkerBoard description

9.1.1.3 BeagleBone Board

Our second development board we are considering is the BeagleBone Black. The BeagleBone Black is a community supported development platform created for hobbyists and developers. The BeagleBone company is a part of an on-line community resource, where they are confident any question a user may have, will be helped in a timely manner. It is priced at $44, and it comes with a USB cable and an instruction booklet that should be read. In addition, this development board is compatible with Android software, with an allowable performance level and with more than plenty of room for development building at a reasonable price.

The BeagleBone Black is 3.4” x 2.1”, making it the average size of a credit card. This development board uses an AM335x processor which contains a 2,000 MIPS with a speed of 1 GHz superscale ARM cortex-A8 AM3358. The memory on this board supports 512 MB of RAM in the form of DDR3, which runs up to a speed of 800 MHZ. The BeagleBone Black has 2 GB of on-board memory and runs linix out of the box. If a user of this development board were to need additional memory, the user can expand the storage partition by using the micro SD card slot, located under the development board, allowing the user to insert their microSD card.

The BeagleBone Black has one USB 2.0 port, which means we will have to use a USB hub to allot additional USB 2.0 ports. It also has a 10/100 megabit ethernet connectivity for internet usage. The BeagleBone Black is capable of full HD output via a micro HDMI port.

The BeagleBone Black has a TPS65217 PMIC (power management integrated circuit) to provide power management to the BeagleBone Black; it also contains one additional LDO (Low-

dropout regulator), which is a type of DC linear regulator that regulates the voltage being outputted by the development board no matter if the input voltage being supplied to the development board is close to the outputted voltage value.

The BeagleBone Black can be powered via mini USB or DC jack. The DC jack has a 5.5 mm x 2.1mm barrel plug centre positive that allows the development board to receive 5 Volts to power. There are two ways to power the BeagleBone Black. One way is to power from a 5 Volts source, the development board won't be turned on if the user's intention is to use the device as a standalone device via HDMI to micro HDMI cable and an ethernet cord cable as well as a 5 Volt DC power source. The second way is to connect the BeagleBone Black to a computer via the USB to mini USB cable that comes with the box. With the USB going into the computer and SSH going into the board. One of BeagleBoards open community blog, The PiHut, mentions that the recommended minimum current supply for the BeagleBone Black is 1.2 Amps, but for board that are connected to capes or USB peripherals, there should be at least 2 Amps of current provided to the board.

Figure J – BeagleBone Black description

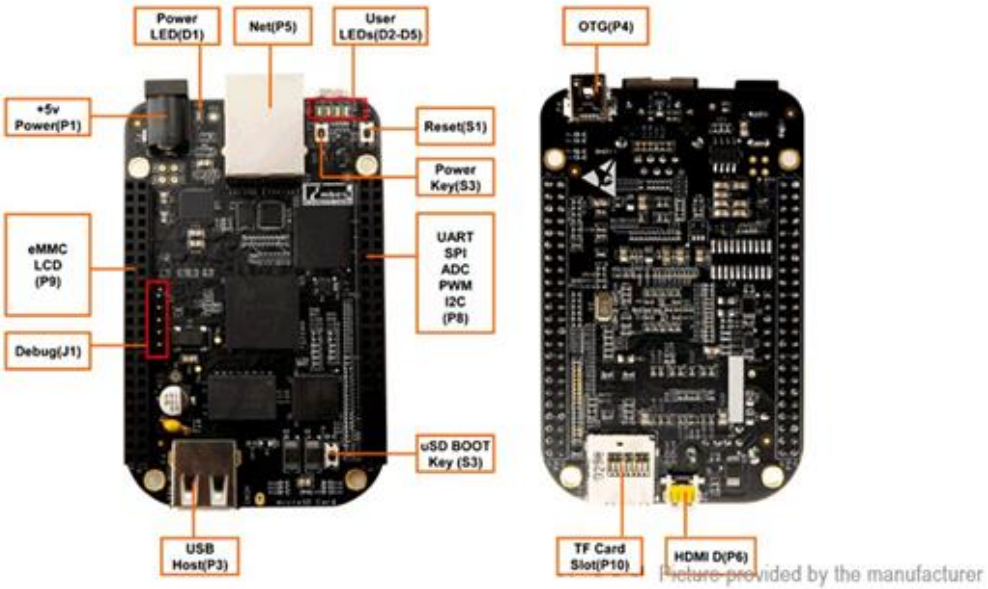


Figure J - BeagleBone Black

Reprinted from BeagleBoard.org, licensed under a Creative Commons Attribution-Share Alike 3.0 license.

There are 65 GPIO pins and various interfaces from ITC to analog inputs and outputs, which are the female pins on the side of the board. Also, there are three user input buttons, rest button, a boot button and a power button. The BeagleBone Black has four built-in LEDs.

JTAG is used to design and develop software applications, as well as control hardware to test printed circuit boards. Its focus is on functional and structural board tests as well as system programming using the JTAG interface.

9.1.1.4 Rock960 Board

The third and final development board we are considering is the Rock960 Board. This development board has a Rockchip RK3399 SoC (System-on-a-chip) with an ARM Cortex-A72 Dual-core CPU that goes up to 1.8 G, a Quad-core Cortex-A53 that goes up to 1.4 GHz and a ARM Mali T860MP4 GPU. It's processor supports multi-format video decoders such as H2.64/H.256/VP9 up to 4Kx2K @ 60 frames per second and supports H.264/MVC/VP8 encoders by 1080p @ 30 frames per second, JPEG encoder/decoder and an image preprocessor/postprocessor. The Rock960 RK3399 processor has a dual channel memory interface that can hold rigorous memory bandwidths. The memory in this board holds 4 GB LPDDR3-SDRAM at a rate of 1866 MHz. It also includes eMMC 5.1 storage built-in. There are also 4 lines PCIe 2.1, which allows the user to attach an SD card.

The Rock960 has a AP6356 WIFI/Bluetooth module from AMPAK Technology, which complies with IEEE MIMO standard, that can go up to 867 Mbps in speed, as well as being able to connect the wireless LAN (Local Access Network). The Rock960 RK3399 supports HDMI 1.4 and 2.0 at 120 Hz up to 1080p and 4Kx2K at 60Hz HDTV display resolutions. This board also includes one USB 3.0 Type A controller, one USB 3.0 Type c, one USB 2.0 Type A port controller, one USB 2.0 high speed expander header, that all have the ability to work alone. The USB Type c port can be used to download/flash firmware and debug or it can be used to connect a USB Type c hub to acquire more USB ports, Ethernet port, VGA port, HDMI port or DP port.

The Rock960 development board can accept power from an 8 Volt source through the DC jack or through the DC_IN pins on the Low Speed Expansion Connector. The Rock960 board also has two buttons, one reset to reboot the system and a Maskrom button used for firmware flash/upgrade. This board also comes with six LEDS, two are used to indicate WIFI activity (Yellow) and Bluetooth activity (Blue); the other four LEDs are user LEDs (Green), where the Rock960 Board controls them from the RK3399 GPIO. Finally, Rock960 Board holds three UART ports, which support up to 4 Mbps, two of the UART ports (UART3 and UART4) are located on the Low Speed Expander, one of the UART ports (UART2) has a 3 pin header.

The board is a $90 development board, that includes a 12 Volt power adapter, a transparent acrylic case, a pack of screws to attach the acrylic ase to the board, a 1 inch x 1 inch heat sink and a mini tube of heatsink plaster.

Figure K – Rock960 Board

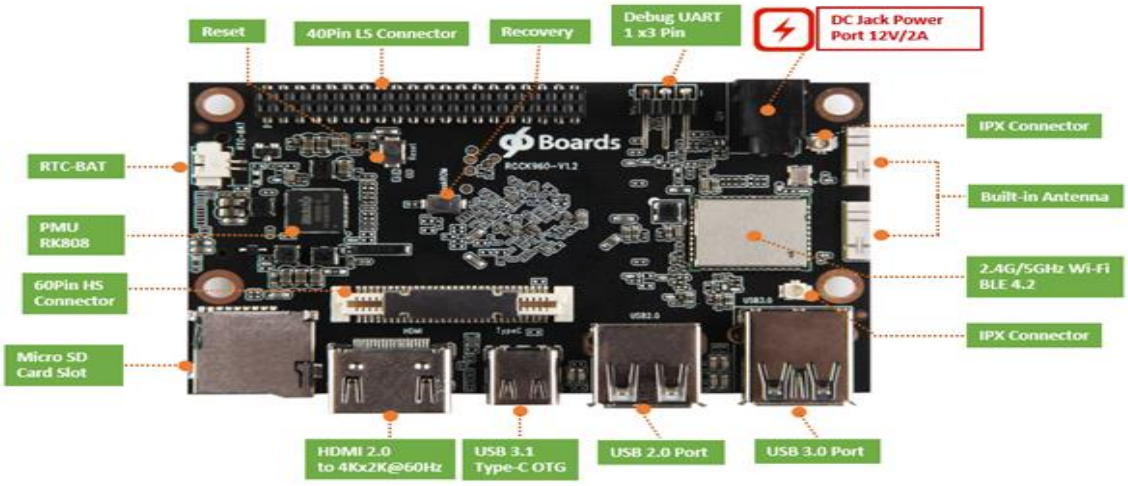


Figure K - Rock960 Board

Figure L – Rock960 Board Chip Layout

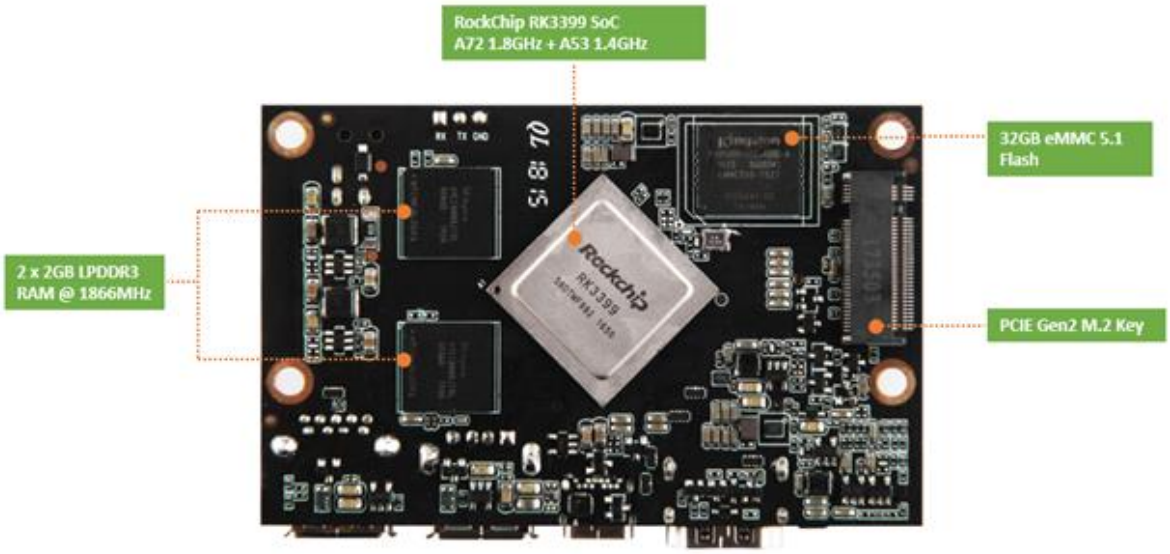


Figure L - Rock960 Chip

Reprinted from BeagleBoard.org, licensed under a Creative Commons Attribution-Share Alike 3.0 license

9.1.2 Sensors

A barcode is used on products for a scanner to identify a specific product. These codes are a low-cost way for businesses to access information about a product. There are many different types of barcodes, and are separated into two categories: 1D, and 2D. 1D barcodes are usually shown as a horizontal image of black bars of variable length, and with variable spacing. 2D barcodes are commonly shown as a box with black pixels. For our project, our goal is to scan data from barcodes of food, most of which will have a 1D barcode called a Universal Product Code (more commonly referred to as UPC).

The UPC is made up of twelve digits, and these digits can be broken into three parts: the manufacturer number, which is 6 digits, the item number, which is 5 digits, and the check number which is the last digit. These digits are displayed, usually beneath the barcode itself, for people to read in case the barcode is damaged. The manufacturer number corresponds to the specific company, and a company must request one to have it assigned. The item number is different for each individual item the company sells, and the check number is used as a way to double check the item scanned is actually the correct item. To test it manually, add the odd numbers together and multiply them by 3, then add all the even numbers to the result. When adding the check number, the total number should be divisible by 10. If this is not the case then there was an error in the scan.

To read any barcode, we need to have a scanner. For reading a 1D barcode, the scanner emits a laser and collects the emitted light using a photoelectric cell. For reading a 2D barcode, a camera is needed to capture the entire image as a program deciphers the data in the matrix. Since we're going to be scanning UPCs, we will focus research specifically on 1D scanners.

9.1.2.1 Tera 5100 Barcode Scanner (specifications and details)

The Tera Barcode scanner is a versatile handheld scanner for use specifically for one dimensional barcodes. This particular scanner, comes with a stand, which allows it to be angled without holding it giving it a hands-free option if desired, but also lets it rest while idle, instead of storing it in a drawer it could be resting securely on a counter, or mounted to the side of a fridge. It can be either plugged in via USB to be turned on and used, or used wireless through a USB fob. The range of distance to operate would be whatever the length of the usb cord is, up to a maximum of 328 ft, assuming no barriers. This is more than enough, but is beneficial that it does not have to be attached to the supporting device. It has a 32bit CPU to decode the barcodes at a speed of up to 300/sec, and can be set to a continuous mode to scan without pressing the trigger multiple times. The battery is a 1300mAh which has a 12,000 hour scan lifespan, and after a full charge which takes only 3 to 4 hours, will last for about two weeks or 40000 scans before needing a recharge. It also has a storage mode that allows it to retain scans in its built in memory.

Figure M - Tera 5100 Barcode Scanner

Figure M - Tera 5100 Barcode Scanner

Table F – Tera Specifications

Table F - Tera Specifications

Feature	Description
Dimensions	Length: 2.7 inches Height: 6.3 inches Width: 3.5 inches
Internal CPU	32bit
Scanning Modes	Manual Continuous
Connection Modes	Wired USB 2.0 Wireless USB 2.4 GHz
Battery	1300mAh
Upload Mode	Instant Storage based
Supported barcodes	UPC, EAN, Code 128, Code 39, Code 39 Full ASCII, Codabar, Industrial 2 of 5, Interleaved 2 of 5, Code 93, MSI, Code 11, ISBN, ISSN, China Post, GS1 Databar, Code 32
Price	$34.98

9.1.2.2 REALINN Mini Barcode Scanner (specifications and details)

The REALINN scanner is a pocket-sized one dimensional scanner, and while we aren't designing this to take on the go, the size could still be beneficial as it would take up less space on a counter. It has its own LED screen to show the current time, it's connection and battery level. This scanner is able to be connected to our processor with a wired USB 2.0 connection, a wireless connection through a USB hub, or with a Bluetooth connection. The connection distance is comparable to the Tera scanner, although the Bluetooth range is a maximum of 164ft. This device is also able to read a barcode from a screen, though would not likely be necessary for our purposes. This also has a timestamp functionality which can be added when reading a barcode. The battery is a 1500 mAh, which has an 18 hour continuous use battery life, or is able to last up to two months while in standby mode. This scanner also has a vibration function to alert whether the scan is successful or not. It also has the ability to store barcodes in it's built in 16MB memory for offline storage. This device has three separate scanning modes as well. The first is a manual button press to scan. The second is a continuous mode to scan multiple things at once, and the third is an auto sensor that will scan when a barcode is placed in front of it, probably with a proximity sensor before going back to low power mode.

Figure N - REALINN Mini Barcode Scanner

Figure N - Realinn Mini

Table G – REALINN Specifications

Table G - RealINN Specifications

Feature	Description
Dimensions	Length: 1.78 in Height: 4.68 in Width: 1 in
Scanning Modes	Manual Continuous Auto sensor

Connection Modes	Wired USB 2.0 Wireless USB 2.4 GHz Bluetooth
Battery	1500mAh
Upload Mode	Instant Storage based
Supported Barcodes	UPCE, UPCA, EAN8, EAN13, Code39, Code32, Code128, Code93, Code11, Codabar, Industrial 2 of 5, Interleaved 2 of 5, ISBN, ISSN, MSI-Plessey, Febran
Price	$49.99

9.1.2.3 Symcode Omnidirectional Barcode Reader (specifications and details)

This scanner is a small scanner that rests on a surface and is designed to be hands free specifically. It is able to scan and read both one dimensional and two dimensional codes, and uses a button to switch between three different modes: 1D only, 2D only, or both 1D and 2D. Using an infrared induction trigger it is able to automatically scan when a barcode is placed in front of it. It also uses CCD imaging to help it read barcodes when any imperfections may be present, like damage or distortion.

This scanner is however only a wired USB scanner, and will have to be physically connected to our device. It also houses an ARM 32-Bit processor but only has a scan speed of 200/s. To scan the images it uses a 640 * 480 CMOS image sensor. The CMOS image sensor is how it reads the barcodes which is different from the other scanners that use lasers and light sensors. The image sensor will still scan in light but read it as an electrical input and create and decipher the image based on that. This is how it’s able to read both 1D and 2D barcodes. It does not look like the Symcode scanner uses a battery, because it relies on being plugged in. It also does not seem to have any kind of internal storage and would rely on the device to be online when scanning.

Figure O - Symcode Omnidirectional Barcode Reader

Figure O - Symcode Omnidirectional

Table H – Symcode Specifications

Table H - Symcode Specifications

Feature	Description
Dimensions	Length: 3.19 in Height: 5.51 in Width: 3.19 in
Scanning Modes	Auto sensor
Sensor	640 * 480 CMOS
Connection Modes	Wired USB 2.0
Power	DC 5V
Upload Mode	Instant
Supported Barcodes	1D: EAN-8, EAN-13, Codabar, CODE 39, CODE 93, CODE 128, China Post, GS1-128, UPC-A, UPC-E, ISBN/ISSN, ISBT , Interleaved 2 of 5, Standard 2 of 5, Matrix 2 of 5, Industrial 2 of 5 , MSI, RSS, ITF14, Telepen 2D: QR Code, Micro QR Code, Data Matrix, PDF417, Micro PDF417, Maxicode, Aztec, Hanxin
Price	$49.99

9.1.2.4 Scanner Comparison

TABLE I - Side by side comparison of the specs.

Table I - Side by side comparison of the specs.

Features	**Tera**	**Realinn**	**Symcode**
Dimensions	Length: 2.7 inches Height: 6.3 inches Width: 3.5 inches	Length: 1.78 in Height: 4.68 in Width: 1 in	Length: 3.19 in Height: 5.51 in Width: 3.19 in
Scanning Modes	Manual Continuous	Manual Continuous Auto sensor	Auto sensor
Connection Modes	Wired USB 2.0 Wireless USB 2.4 GHz	Wired USB 2.0 Wireless USB 2.4 GHz Bluetooth	Wired USB 2.0
Battery	1300mAh	1500mAh	N/A
Upload Mode	Instant Storage based	Instant Storage based	Instant
Supported barcodes	UPC, EAN, Code 128, Code 39, Code 39 Full ASCII, Codabar, Industrial 2 of 5, Interleaved 2 of 5, Code 93, MSI, Code 11, ISBN, ISSN, China Post, GS1 Databar, Code 32	UPCE, UPCA, EAN8, EAN13, Code39, Code32, Code128, Code93, Code11, Codabar, Industrial 2 of 5, Interleaved 2 of 5, ISBN, ISSN, MSI-Plessey, Febran	1D: EAN-8, EAN-13, Codabar, CODE 39, CODE 93, CODE 128, China Post, GS1-128, UPC-A, UPC-E, ISBN/ISSN, ISBT , Interleaved 2 of 5, Standard 2 of 5, Matrix 2 of 5, Industrial 2 of 5 , MSI, RSS, ITF14, Telepen 2D: QR Code, Micro QR Code, Data Matrix, PDF417, Micro PDF417, Maxicode, Aztec, Hanxin
Price	$34.98	$49.99	$49.99
Unique Features	Stand included Possible to mount	Timestamping scans Display	2D barcode scan Image sensor CCD/CMOS

The biggest upside to the Symcode scanner is the 2D scanning capability, but most of the other specs are a downgrade in comparison to the other two. It has less connection options, no self storage option, and only one scanning mode. The Realinn scanner has some nice features, but we're confident in our ability to timestamp our scans ourselves without relying on the scanner to do it, and we feel the display isn't incredibly necessary. It does boast a bigger battery, auto sensing, and bluetooth connectivity. However, since our device needs to connect to the internet to communicate with our app, we believe a bluetooth connection would be unnecessary and a little more complicated than just relying on a wifi connection. We believe the versatility of the Tera scanner, along with the lower price will be our best option and is what we are choosing to use with this project.

9.1.3 Wi-Fi modules vs. Wired

An advantage of having our system to have a wireless network connection will be the flexibility of our product placement that this sort of network offers. Having Wi-Fi integrated in our system, we can connect it to an already established network, with no worry about wires accumulating in the location the system happens to be placed.

Belkin Wireless G WiFi F5D7050 USB Network Adapter - This network adaptor can be used with the Tinker Board, Rock360 board, and BeagleBone Black board. The Belkin Wireless USB Adaptor has a generic driver that may be difficult to configure. Also, as the name suggests, the Belkin Wireless G Wif-Fi USB Network Adaptor can be connected directly on the board via USB. The wireless Network Adaptor has 802.11g of wireless connectivity, which equates to about three times the range of the 802.11a wireless adaptor and has backward-compatibility with all 802.11b network devices. A 64-bit and 128-bit WEP encryption is also featured wirelessly.

Figure P – Belkin Wireless Wifi Adapter

Figure P - Belkin Wireless

Table J Belkin Wireless G

Table J - Belkins Wireless G

	Specification
Network Standards	IEEE 802.11g
Range	Up to 400ft
Security	WPA, WPA2, 64-bit/128-bit encryption
Specification interface	USB 1.0, 1.1, 2.0
Operating Voltage	5 V

9.1.3.1 TX2-RTL8723BS (specifications and details)

The TX2-RTL8723BS is a wifi chip produced by Realtek and is embedded into the ASUS Tinkerboard. It is shielded on the board, and comes with an IPEX header that allows for an optional antenna upgrade. The TX2-RTL8723BS has IEEE 802.11 b/g/n connectivity. The IEEE 802.11n means it's able to connect to multiple bands of frequency. The most common will be 2.4 GHz and 5 GHz. The wifi chip has a Gigabit connection. The security ranges from 64-bit, and 128-bit WEP, WPA, and WPA2. Allows for a 4.0 Bluetooth + EDR (Enhanced Data Rate) connection.

Figure Q - TX2-RTL8723BS

Figure Q - TX2-RTL8723BS

Table K Tiwi R1 Module

Table K - Kiwi R1 module

	Specification
Network Standards	IEEE 802.11b IEEE 802.11g IEEE 802.11n
Security	WPA, WPA2, 64-bit/128-bit encryption
Specification interface	integrated into the board
Operating Voltage	3.3V

9.1.4 Introduction to Printed Circuit Boards

Traces on printed circuit boards or PCBs, have basically made all of modern electronics possible. Before printed circuit boards became widespread in the later part of the 20th century. Electrical components inside of an appliance looked very heavy, messy and bulky. Inside televisions in 1948 were full of tangled wires. Fortunately, the invention of embedded wires in a flat piece of fiberglass has taken over and because so, we can fit a computer in a backpack, instead of a giant cabinet. You might be wondering how it's been possible to pack connections from bulky wires to a flat piece of fiberglass. The process of creating such a small flat piece of fiberglass is similar to how CPUs are manufactured.

A printed circuit board is a multilayered network that contains hundreds of copper wires. Printed circuit boards provide structure and organization for all components to be mounted. The wires in between the components allow each of the components to communicate to each other and work together.The more wires there are connecting one component to the other, allowing for more data to travel through the PCB. Traces are built in on different levels in the PCB, making them electrically separate. Each component can communicate on the PCB, with the SoC and other microchips, though thousands of different wires that are precisely organized.

The term "Motherboard" is a PCB with the components attached to the board. A PCB is simply a flat board with no components attached to it. Printed circuit boards have conductive wires or traces. The wires appear to be on top of one another but they are not actually touching. The other material on a PCB is a non-conductive insulator created from woven fiberglass with an epoxy resin binder called FR4.

In a smartphone, there is a microchip called "System on a Chip"(SoC). The System on a Chip is attached to the PCB by a connection point called a "Ball Grid Array". This array grid connects the mind of the smartphone to the wires or traces on the PCB. There are other grid arrays on the PCB for other wireless microchips. For example, the memory chip has a place on the PCB and the wireless chip, as well as capacitors,and resistors.

To make a PCB, slices of fiberglass are placed one on top of the other with a resin that helps them stick to one another, which then makes one flat piece of fiberglass. Layers of copper are then added on the top and bottom of the fiberglass. A chemical called Photoresist is then used to coat the copper on both sides. Then, a pattern that contains the traces, is placed at the top on the copper layer and exposed to UV light. The board then is cleaned, and the areas that didn't contain traces are cleaned away, leaving behind traces of pure copper. This is the beginning of the PCB creation. In the following steps, there will be etching additional copper and more cleaning away the untraced copper, just like in the first step. This is a process that is done layer by layer.

9.1.4.1 Basic Board Structure (2 Layer Board)

The board itself is made up of a couple of different layers. A basic 2 layer board contains a critical layer, that is the copper layer. The copper layer is where the traces are, that actually are the wires within the circuit that are connecting the different components in the circuit. In 2 layer boards there is another copper layer at the bottom. Between the top copper layer and the bottom copper layer is a dielectric material, which is essentially fiberglass. This fiberglass material is referred to as FR4. FR4 has a particular dielectric property. FR4 is also called the middle layer and also sometimes called the substrate. Looking at a basic board (2 layer board) we have copper on the top and copper at the bottom with FR4 substrate in between. Also what most boards have, is a layer that sits at the top of the copper and on top of the copper at the bottom, called a solder mask. Solder mask is a layer of insulation that sits on top of the copper so that things won;t touch each other. Then there is this layer of Silkscreen that sits on top of everything. The Silkscreen is text that allows us to identify what the components are.

9.1.4.2 Board Structure(Multi-Layer Board)

A multi-layer board is like a sandwich of the basic 2 layer board. There is a copper layer, some FR4 and a layer of copper. Then there is a material called prepreg just before another copper/FR4/copper structure. Prepreg is a cheaper material, which helps the individual elements to adhere. This is essentially a 4 layer board. A 4 layer board is very common. However, if we inspect a motherboard in a computer or something of that level of complexity, there may be up to 16 layers in those kinds of boards and it may be a very complicated design. Most designs are somewhere between 2 and 6 layers.

9.1.4.3 Soldermask and Silkscreen

The solder mask is the green layer of material found on generic PCBs. If there is exposed copper on the board, those are called pads. Pads have a silver look to them and are slightly protruding from the board. We can solder on these pads. There are types of pad, there are annular rings , which are around plated through-hole. A plated through hole, is a hole that has been drilled through the board and it is plated with copper all the way through the board. We will typically solder onto these. There is another type of through-hole that we do not solder to, which is called a via. A via is connecting one layer to another layer. It is connected by a wire on one layer to another layer. When looking at a PCB, there are dark green layers and light green layers, contrasting each other. The light green lines are basically copper, sitting on top of the FR4 substrate. There may even be big light green areas, these areas are called copper pour. Where copper is effectively in a large plane. The skinnier light green areas are called

copper traces. Essentially, what has been done is the silkscreen has been sat on top of the top layer and so wherever it crosses over copper we'll see this lighter color. The darker areas are basically open boards. It is the solder mask sitting on top of a blank FR4 substrate. The pads that can be used to solder on are for through-hold components. There are also pads for surface mount components. We would solder components directly on to these pads. The silkscreen is all the white text that can be found on top of the PCB. Silk screens are a critical thing that will allow us to know what components we are actually looking at when we are looking at the board.

9.1.4.4 Printed Circuit Board Layout

In this section we will talk about how to piece together a printed circuit board. The first thing to take note of are the mechanical prerequisites. For example, is there a shape that the board has to be in? If the board is supposed to fit into a specific enclosure? The shape needed will be a constraint factor. The mechanical components such as potentiometers, switch or a jack, are components we would like to know where they have to be.

When deciding where components will be placed, we must think about how the board as a whole will be used. Once thoughts about how the board will be handled, deciding on where to place the components become easier. Also, thinking about how the signal is going to flow? Where are my inputs? Where are my outputs? How is the power going to get in? So, a power jack port can be found at the corner, next to the jack there can be a USB port, next to the slide switch there might be a slide switch. Some input ports may be placed on the other side of the board and the output ports may be placed at an adjacent position. It all depends on how the circuit will be handled as a whole. These will be the beginning constraints of what is to be built. We have a decent idea of where the circuit sections have to go.

When a printed circuit board is made, we can get 1 ounce copper or 2 ounce copper. It is cheaper to get 1 ounce copper and it is said to get the job done pretty well. Then we'll consider how many layers the printed circuit board should have. Boards that are 4 layers tend to be more expensive, so the majority of the time 2 lay boards are sufficient enough because it works and it is inexpensive. Our design is easily enough to do as a 2 layer board. A 4 layer board will give us a dedicated VCC plane in the middle and a dedicated ground plane in the middle. The 4 layer is cool because there isn't much concern about stringing weird traces to get things done, like we would have to with a 2 layer board. The 4 layer board also gives us the additional capacitance between the two layers. So, it will act as a filter, decoupling everywhere.

A trace is a way to complete a circuit on a printed circuit board design. Trace width for low current path, and we don't expect the current to be running more than 20 milliamps, the majority of the time we will use a 10 millimeter trace. If the current is over 20 millimeters, the trace width is to be increased. Generally, a 10 millimeter trace is good for about 30 milliamps. Auto routing application should be avoided because it will produce a sloppy trace outcome. We want to keep our traces nice and elegant and auto routers don't generally produce nice traces. We also avoid right angles in our traces. A right angle on a trace will round the edges. Logical devices read edges, so the edge needs to be relatively intact. Avoiding right angle traces is a rule of thumb to avoid running into issues. The goal when designing a PCB is to keep the probability of issues down.

When we are designing the schematic, we have a list of parts we need to arrange. The sections of the circuits are going to be arranged one the board in literal sections. For example, there is a power section, an amplifier section, a microcontroller section, senors section or RF section. These sections are going to be placed on the board according to their position in the signal chain. They all have signal chains of some kind. For example, let's say our PCB has headers on both sides of the board, that are inputs and outputs respectively.
The signal chain dictates where these sections go. When designing a circuit board, the flow of the board is important. The circuit stages are built according to the schematic. We literally look to see that we have an op-amp in the schematic and we'll place close by components around the op-amp because they are close by and that is critical. That is how components are placed together on a circuit board.

It is important to keep components that are related to a certain circuit section close to each other, because it's best to keep those traces short and elegant. The first section that is good to start with is the power supply, which is to say a regulator,a switch and may be a fuse. The switch is generally always close to the power jack. A switch is a good power controller when you want to cut off power completely from a power supply on your circuit. Generally, like the schematic will probably show , it will have a cap on the input, a small current regulator, and a cap on the output. That circuit section could be all those components in one section. it is good to have the power supply out of the way in a corner and then feed all the other sections. This should be done according to what the schematic shows. Components that are close electrically in a schematic to be close physically on the board.

To feed these components, a modified star configuration is very helpful. a star configuration is that each section gets its own dedicated line back to the regulator. In the schematic there are going to be a zillion different places potentially that the VCC is going to need to go to. However, it is unrealistic to run a line from the power supply to every single component that has to be covered by VCC. If it were a 4 layer board, it would be easier because there is a layer of copper underneath and can be accessed with a push of a button. That is the VCC in a 4 layer board, you can get it anywhere. However in a 2 layer board, planning is required with how the VCC will get distributed.

The sections on the circuit board will be divided. For example there will be a microcontroller section, a sensors section and an RF module section. Now, to give these three sections power, we will section them in the schematic.

When building the circuit sections, look at the schematic for each component and distribute the VCC as efficiently as possible for that section. The parts close to each other electrically, you want them to be close to each other physically on the board. For example, an op-amp has two points that VCC needs to be fed and four places needed for ground to go. When putting this circuit together, the goal is to minimize the common points, so that there is only one feed point, instead of four feed points. On an op-amp there is an input voltage source physically and an output voltage source coming out physically to go to the next stage. Start by looking at the critical ones, anything that is attached to a high impedance line, on an op-amp, has to be a pretty tight path.

The whole bottom of the board will be copper. Having laid down all the signal traces first. Some of the traces will have to go on the bottom, from component to component. Ideally we try to keep all traces on the top of the board, so that the bottom of the board can be all copper, but we will always have a few traces on the bottom. Then the easiest thing to do is lay down a huge layer of copper all on the bottom. Any place on the circuit that has to go to ground will do so by pressing down. To test the ground and the traces out, to make sure all things are working as intended, we get the DRC (design rule check) all through this design. The Design Rule Check will let us know about any dimensional problems. It will tell us if there are any air wires, but it won't tell you if the current is coming back to the power supply (the return path). At all my circuit stages, I'm going to be looking at how much current the stage is drawing, and what the current path looks like. If it's a low current return, most likely there isn't much to worry about because there will be enough copper to find multiple paths and it will use the path of least resistance. If the current return were higher, our return path should be as straightforward as possible.

Next, we'll talk about aesthetic things like silk. Between the time we submit this board for prototype and by the time we get it back, we are going to forget everything about this board. Putting some silk on it will help us know what part is what. Silk will give us headers and labels on our pins. The USB ports and power jack port will be obvious but for the places that have different signals, it will be a big help for us to have them labeled. The resistors will be labeled, for example, a resistor will be labeled, "R3". When actively working on a board with probes and iron to rework, silk will be desired and will help a lot.

When we run our prototype and we get it back, we build it and we test it. When we test we'll limit the current on our power supply, down to 100 milliamps. When the PCB was being built we had a rough idea of how much current the board was going to draw with no input and no output, like 30 to 40 milliamps when it's running quiescently. So, when testing the PCB for the first time, it is best to set a limit on the power supply to 100 milliamps or 75 milliamps. Something a little bit above but not high enough that anything is going to burn in the board. Do this insures us that, when we finally hook up power, if the power supply goes into a rail situation or a limiting situation, I know that I have a short and nothing has been destroyed. Another way to avoid burning the PCB, is to us a meter. Place the probes from ground to VCC, to check it doesn't say 0 Ohms and place it at various other places in the circuit. If it doesn't do what was expected, this is the time to go back and find out why. We might have to redesign the whole board, and relayout.

9.1.4.5 Design rules for Printed Circuit Board design

Having a set of design rules, provides us with help with looking at the requirement interactions. Almost all designs for printed circuit boards have a logical side and a physical side that are to be properly managed in order to establish the perfect design. Design rules allow the designer to define the system and have a list of checking these requirements, giving us an interface between the logical and physical side so an actual board can be produced. With the checked off requirements we have instant validation of the design and proof that it will work once assembly is complete. With the requirements being validated and checked off, creating reports and certain information is made with clarity as well.

When design rules are brought upon a system, they come from the PCB aspect of the system, from the schematic side of the system and from the manufacturing side of the system. Finding a combination of necessities from all three, will help create rules that will help us cover all the areas of the PCB design. Incorporating the following six rules when making a design will help us stay on the right track throughout the development of the PCB.

A polygon connect style

How to connect from a via or pad, to a polygon itself. This insures the attainment of proper therma connection, proper current connection to the copper and theses polygons. Without this information it is possible that a pad is missed and proper connection to a polygon is achieved.

Solder Mask Expansion

If the soldermas expansion isn't properly defined across the pads, it is possible that it then would be over defined or under defined. If not defined uniformly across a component, it is possible to get tombstoning because one side will cool faster than the other and that effect will be produced. Creating a rule for the Solder Mask Expansion makes sure that everything is uniform and everything is done right the first time.

Routing Via Style

As we are running our routes, it is important to make sure we are using the right type of via. If the via has a diameter to hole ratio, it is possible that it can't work with the current that needs to be driven through it. It can get fused vias and have different problems. It is very important to associate different via styles for the different routes throughout our design.

Component Clearance

Let's say we have a component that is fully defined with a 3D model, and we have a 3D body for it. The component clearance will be able to tell me vertical clearance from an enclosure. A component clearance will tell me clearance from other components. If there isn't a 3D boday and it is just a 2D footprint or a courtyard, you'll be able to get distance based on the primitives on the copper layer and the silkscreen layer.

Electrical Clearance

Looking at the actual traces and pad clearance, it's the same type of information, it's just on a smaller level. If the appropriate clearance isn't there, different types of issues can arise that aren't ideal for any design.

Routing Width

The width of the routes is very important. To ensure that everything is as it needs to be, are your power routes, the sufficient size to carry the current that you need. If this information isn't obtained it can be very detrimental. We can have characteristic impedance controlled routing , we can have different impedance profiles to apply to these routing widths and make sure that everything is created in a manner that is beneficial to the design as whole.

9.1.4.6 Printed Circuit Board Design Tips

It is a good idea to breadboard the project before designing the PCB schematic in the design software. It's faster to find issues in the design outside of the design software, than it is to find issues while editing components on your editor. Also, when the finished PCB arrives from the manufacturer, it is also a good idea to test the newly manufactured PCB using a breadboard before soldering any components on the board.

Be patient when placing components. When time is spent on carefully placing the components on a PCB editor is never wasted. The placement influences greatly on the outcome of the PCB design. The key point is to avoid the overlapping of the rubber bands. The less overlap the lines do, the better the routing signals will be when using the PCB.

Once finished with component placement, routing is next. Considering the directions of the routing directions of a layer before laying the traces. For example, in a 2 layer PCB, route the top layer in a horizontal direction, while the bottom layer routing is going in a vertical direction. Having directions that are inconsistent, can damage the entire design, which are time consuming and costly.

Normally when laying down the traces on a PCB, we tend to use the same width for all the tracing in the design. This is not recommended. We will always consider the maximum current that can flow through the traces in our design. So if a trace carries high current in the design, using a trace that is larger in width is more suitable. Having a higher trace width, provides a higher heat dissipation, when compared to traces with low width. A higher trace width prevents failure.

A PCB is made, by making etching away the unused copper and leaving the traces intact. Using the unused copper for any signal rather than etching it out is called copper pour. Copper pour is done by making space for traces and pads for the components. Copper pour is commonly used for ground signal or power signal. The large surface of copper pour gives a low impedance path for current and it makes connection a lot easier.

Paying attention to spacing is an important aspect when designing a PCB. The software does automatically establish a good amount of spacing between traces and components. The design software also raises a flag when proper spacing isn't being given to a component or trace. As a designer, being aware of the space requirements is important.

9.1.4.7 All PCB Manufacturing

The manufacturing company that will create the PCB for this project is ALL PCB. They are located in Jiangxi, China. This company was recommended to us by previous senior design students that received very good service from the company. The company gives first time customers a free trail of up to 4 layer PCB, the customer also gets free shipping. All PCB has reliable shipping, as they promise to deliver within 2 weeks time. To place an order with ALL PCB, one simply logs in at ALL PCB website, fill in the required specifications on the quote page and add to cart. Lastly, upload the Gerber file, click the button labeled "Buy All", check the shipping address and other details and click submit.

9.1.4.8 Power source implementation

A power supply is a source with electrical energy and provides that electrical energy to your device. Another form to define a power supply is a device that takes electrical energy and converts it from one form to another and transfers the electrical energy to a load. There are cases where power supplies such as batteries, generators, and solar cells essentially provide power to a load.

Figure R - Solar Energy

Figure R – Solar Energy

A power supply must get its electrical energy from somewhere, in order to supply electrical energy to a load. For example, solar panels get energy from the sun, smart batteries have a built-in charger, so it's considered a self-contained power supply.

Figure S – Power Supply

Figure S - Power Supply

The electrical power that comes out of the wall isn't always compatible with what a certain device needs, so a power supply can transform that electrical power to a specific shape that the device can take. For example, the alternating current from the wall is taken by an AC

power supply device and transforms it to DC (direct current) that the device needs. The majority of electronic circuits work with direct current. The electrical energy that comes out of a battery is DC. So, when an electronic device has a battery as its power supply, the electrical energy doesn't go through a transformation because the electrical energy is already the DC coming from a battery.

The physical aspect of a power supply can come in different forms, such as open framed or in an enclosed frame.

The majority of household devices use low voltages of DC. When it comes to how the power is divided within the device, there are separate power supplies for each component, such as the camera, the display, and for the LED lights. Every device in an electronic device needs a certain amount of power, measured in voltage.

AC current is like a wave of electrical power, while DC is a line of electrical power. The wave of an AC current can flow up and down at a certain speed, called the frequency. Voltage and current are also very important to differentiate. Voltage is like the pressure of water in a tube and the current is the amount of water in the tube. The power supply changes the pressure and can regulate the current.

Figure T– AC vs DC

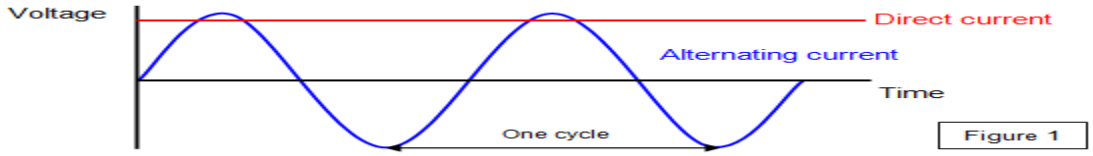


Figure T - AC vs DC

9.1.5 Rationale for Hardware Component Selection

9.1.5.1 Board with ARM-based processor

We decided to choose the ASUS Tinker Board as our development board, which has a ARM-based RK3288 SoC processor. The ASUS Tinker Board is meant to be used as a mobile software development platform. In our project we will use the ASUS Tinker Board as a model for our very own printed circuit board design.

The table below shows the specifications of the ASUS Tinker Board:

Table L - Tinker Board Technical Specifications

Table L - Tinker Board Technical Specifications

CPU Performance	-ARM-based RK3288 SoC at 1.8 GHz -Equipped with four cores to -enhance multi threaded application performance
Graphic Performance	-ARM Mali-T764 GPU
Memory and Stream Performance	-2GB of RAM Dual-channel DDR3 SDRAM
Network Specification	-Features a gigabit Ethernet port
Connectivity	-4 USB 2.0 -GPIO connectivity (Color Coded)
Extra Storage	-EMMc Memory 16GB -Micro SD card Slot
Dimensions	-2.13 x 1 x 3.37 inches
Display	
Áudio	-RTL ALC4040 CODEC
Wireless Connectivity	-Wifi 802.11 b/g/n with an upgradable IPEX antenna -Bluetooth V4.0 + EDR
Accessories	-Heatsink -User manual

9.1.5.2 Wifi module

We decided to use the TX2-RTL8723BS module which features support with Bluetooth 4.0 + EDR and for WLAN(802.11 b/g/n). The reason for choosing this model was because the cost is factored into the board and would reduce cost. The fact it features Bluetooth is a bonus for us if we decide to use it.

TX2-RTL8723BS Technical Specifications:

64-bit, 128-bit WEP, WPA, WPA2

Operating Voltage: 3.3 V

9.1.5.3 Hardware Comparison

The table below compares the three development boards we have considered. Our final decision will depend on which board has the best performance, so that we can run our program on the best possible platform. Also, our decision will depend on which board offers great online help, just in case we have any questions or concerns while we are using the board, there plenty of other users that have asked and received answers online. Lastly, our decision will depend on which of the development boards has the best hardware features. The BeagleBone Black is one of the most popular boards out of all three development boards. However, the board doesn't feature JTAG, therefore we would have to purchase a JTAG mount as well as installation of JTAG, which will lead to much confusion. The processor on the BeagleBone Black isn't as fast at the ASUS Tinker Board and Rock960 Board. The Rock960 Board isn't a popular development board, so there aren't any online forums that show other users inquiring for help, like the other boards have. The ASUS Tinker Board does have a strong community of previous users, making it easy to find help on the board if we were to ever come across any problems while implementing our program. Even though the ASUS Tinker Board is on the higher price spectrum, the board has great performance speed, enough peripherals to take care of all of our needs, and there are various places online to find technical support.

Table M Hardware Comparison and Price

Table M - Hardware Comparison and Price

	ASUS SBC Tinker Board	BeagleBone Black	Rock960 Board
Board Processor	1.8GHz	1 GHz	1.8 GHZ
Board Memory	2 GB RAM	512 MB RAM	2 GB RAM
CPU	Rockchip Quad-Core ARM SoC RK3288 Processor	AM335x ARM® Cortex-A8 Processor	Rockchip RK3399

Software Compatibility	Whatever your software preference	Android, Ubuntu, Gentoo, Whatever your software preference	Whatever your software preference
Wifi	Shielded Wi-Fi	Shielded Wi-Fi	Wireless Wifi/Bluetooth
Connectivity	◦4 x USB 2.0 Port ◦Micro SD (TF) Card Slot ◦Micro USB Power Input	◦USB 2.0 Port ◦Ethernet ◦Micro-SD ◦Micro HDMI Output	◦USB 2.0 Port ◦USB 3.0 Port ◦Micro SD Card Slot ◦HDMI 2.0 ◦DC Jack Power Port
Dimensions	3.37" x 2.13"	3.4" x 2.1"	3.34" x 2.16"
Website	www.amazon.com	www.mouser.com	seeedstudio.com
Price	$75.86	$43.75	$99.00
Tax	$4.93	No tax	No tax
Shipping	Free	$7.99	$13.07
Total	$80.79	$51.74	$112.07

9.2 HW IMPLEMENTATION

9.2.1 Communication Systems

One very important feature of our application is being able to alert our users of information they would need. This could range from confirming a product has been scanned to a reminder going off to replace or remove a product that has expired. For our project we will need to communicate information wirelessly to the user through the application. In this section we will research different communication types to familiarize ourselves with our potential options for our product. We aim to research multiple forms of wireless communication not only to determine which will be easiest to implement but to also improve our user experience.

SMS - SMS stands for Short Message Service. It originated in Europe and was created by a German engineer and his colleague with the purpose of sending a message from one phone to another. It is the most common form of text communication used today. It can send up to 140 Bytes (or 1120 bits) of information, and if using a 7 bit encoding it is possible to send 160 characters per message. If using 16 bit encoding the messages would be limited to 70 characters but it would allow for multiple languages to be used in this messaging service. It's possible to send more data through multiple messages in a process called concatenating SMS. The downside to this though would be that this is not supported by every platform and may cause issues. SMS are not able to send images without first using an extension called EMS or Enhanced Messaging Service. This runs into a similar issue as the concatenated SMS messages though in that it is not supported by every platform. To receive a SMS it must first be sent through a SMS Center or SMSC (and possibly SMS Gateways), and a phone's address, which is usually the phone number, is required to send a message. In order for us to send messages we have a few options to do it. The first would be to use a phone with a router to connect to a SMSC but this is not a good option for us for a couple of reasons. The first being that it would require a phone to be dedicated to doing this, and the second is the send rate would only be 6 to 10 messages per minute and would be too slow. The second option is to connect individually to a SMSC or gateway to send messages but this can be difficult to do. In order to do this we would need to buy the messages to send from a wireless carrier and they would want there to be many messages sent before even making an offer. It's also possible to order SMS messages in smaller amounts either through gateway providers or through resellers. This would require us to create an account and either pay for credits to use messages or pay for messages individually. This would likely be what we would have to do if taking this route and each message would be 0.06 - 0.07 dollars which could end up being incredibly costly for simple alerts.

Requirements:

-User's cellular phone number

-SMS Gateway or Reseller service

-6 or 7 cents per message sent

Bluetooth - Bluetooth technology is a short range wireless technology that's used to communicate between devices using UHF radio waves between 2.402 and 2.480 GHz. It was originally conceived to be used in place of RS-232 data cables as a wireless option. It's named after Harald Bluetooth, a Danish King from the late 900s, and the logo is derived from a combination of runes that make up Haralds initials. Because a bluetooth connection is a direct connection from one device to another it removes a need for a router as a middle man and because of this means its a more energy efficient option. However since it makes a direct connection it needs to be within range to be used. So if we were to alert a user that something has gone bad and needs to be replaced they must be nearby the device to be notified. This could cause some issues and could lead to a very unpleasant user experience if they missed a notification while at the store. As of 2020 there are two different types of bluetooth devices: low energy (LE) and Enhanced Data Rate (EDR). LE is a little more desirable, because as the name states it saves energy and would reduce costs, and while the EDR has a better bitrate, they still operate on the same frequency. The ASUS Tinkerboard includes a wifi chip that has a bluetooth 4.0 + EDR component. For the purpose of this project the research will focus on EDR over LE since it's a component we already have access to. Bluetooth devices have to be paired together to work. When coming in range with one another they will communicate to determine if they're trustworthy before connecting. This process creates a network. When using peripheral devices to connect to a central hub, like a smartphone, it creates a personal area network, or a piconet. A piconet allows each device connected together to hop frequencies together through a process called adaptive frequency hopping. It's also able to dynamically determine which frequencies are stable, and optimal for use. This can be really useful when in a building using multiple bluetooth devices, it would allow each one to find it's own frequency without interfering with other people's devices. If our product could be used in crowded areas this could be a really useful option for it, although I believe the common situation is in a household and this may not be a common issue for us to deal with.

Requirements:

-Bluetooth adaptor (included in Tinkerboard)

Push Notifications - Another option is to use push notifications which are similar to text messages sent by the web application to the phone but only to users who have the app installed. Apple was the first to implement push notifications in 2009 called APN. In order to send push notifications we would need a user's permission on their phone to send them. After this we would need to subscribe them to push notifications and to do this with Javascript we would use a Push API. Once the subscription is successful we'll receive a subscription object which will need to be stored long term. To send a push notification it doesn't actually come from our application but from the user's web application through a push service. In order to send a push to the user we need to send a request to their web service. The push notification would need to include information like the message to send, the user to send it to, and other information related to the push. Once the push service receives our request it will route it to our user. Each browser uses its

own push service, so we will not have control over that. Lastly we'll need a key to encrypt the message. This will allow the web service to help verify your request.

Requirements:

-Phone permissions

-Push API

-Database space to store subscription status

Email - Email is yet another option we can use for communicating with our users. Email stands for electronic mail, and since most people have an email address it could serve as an effective tool for communicating with our users. In order for us to send an email to a user we would need their email address. Through javascript we would be able to choose what our message would be, and through a mail sending service we could code a process to send email alerts when necessary.

Requirements:

-User's email address

-Proper packages for Javascript

-Web service to send emails.

9.2.2 Hardware Component Interaction

This project is divided in two elements: one being the software and the other being the hardware. The software will cover the website and the mobile application the user will have access to from their phones. The hardware component will only be one third of the project, however it is a very important piece of the user experience as the user will need to scan food items to be recorded into the system.

The largest hardware component is the development board. Selecting the best development board, ensures us of having all the needed ports and applications needed to make our system come to life. The ASUS Tinker Board doesn't come with it's required power supply cord. Nevertheless, a power supply cord can be purchased from the same website we purchased the development board from.

Now that we have the power supply for our ASUS Tinker Board, we can connect our other project components like the USB barcode scanner, and the wifi module to the board. Luckly, the ASUS Tinker Board has 4 USB 2.0 ports, so there are two USB ports to spare on our development board. Had we gone with the BeagleBone Black or the Rock960 development boards there would have been a need for us to buy an extra USB hub to connect our other components, because these boards come with one USB 2.0 port. As for the selection on our barcode scanner, we were purposely in search of a barcode scanner that is powered by a

USB port because we knew USB 2.0 is a popular type of port in the world today. The barcode scanner will be used to scan UPC codes found on most food products.

The other important connection the hardware has is a wireless one. The ASUS Tinker Board has a built-in wifi module, which allows us to connect to the internet wirelessly. Having access to the internet wirelessly allows us to store information and allocate information wirelessly. The wifi access will synchronize the system whenever the user scans a food item into the system. The new food items scanned will appear on the website and the mobile application because of the synchronization connection allowed by the wifi.

Figure U – Hardware Project Display

Figure U - Hardware Project Display

9.2.3 Power

In our Power Implementation section on the research phase, we specified two different approaches on how to deal with connecting the system to power. The first choice was to unite the power supply cable of the development board and the power cord of the refrigerator by cutting and stripping the power supply cable and taking off some of the coating of the refrigerator‘s power cord to wrap one inside the other, being careful to not switch or join different polarities together. This choice was somewhat unsafe and unreliable since one small mistake could end up causing damage to us or the equipment used; which in turn, would mean we would have to come up with more money and time for repairing or buying new parts for our project. The second option was the one we decided to implement on our final design. It is a much simpler alternative and deals less with electrical issues. However, there is no risk of causing any harm to anyone working on the project, or to the equipment used. This choice consists of getting a simple cord extension that enables the connection of at least two power cords to its end. With this purpose, both cables coming from the refrigerator and development board respectively are plugged in one end. 74 The other end of the cable would be plugged into the wall, making it simpler as

ultimately only one cable would need connection to power. We plan on using Velcro to adhere the other end close to the back panel of the refrigerator in a discrete placement. Figure 41 shows the final arrangement of how the system would be wired (the extension cable is not visually exact to the part we will ultimately be using, the figure only attempts to show a simple configuration of components).

9.2.4 PCB Design

For the Printed Circuit Board design, we based it on the ASUS Tinker board. The ASUS Tinker Board has great features that were very useful in the testing and debugging phase of our system. Now, creating our own version of the ASUS Tinker development board with only the features we need; we can produce a simpler, much more practical board.

Our PCB design will include the following components:

- Microprocessor
- 5 Volt input port
- SD card port
- JTAG
- USB port
- LEDs

Although the entirety of the ASUS Tinker Board was of great use for developers, such as ourselves, to configure a mobile application, there was a lineup of components that weren't going to be used in our design. Some of the components on the ASUS Tinker Board were the HDMI cable,

Figure V – PCB design prototype

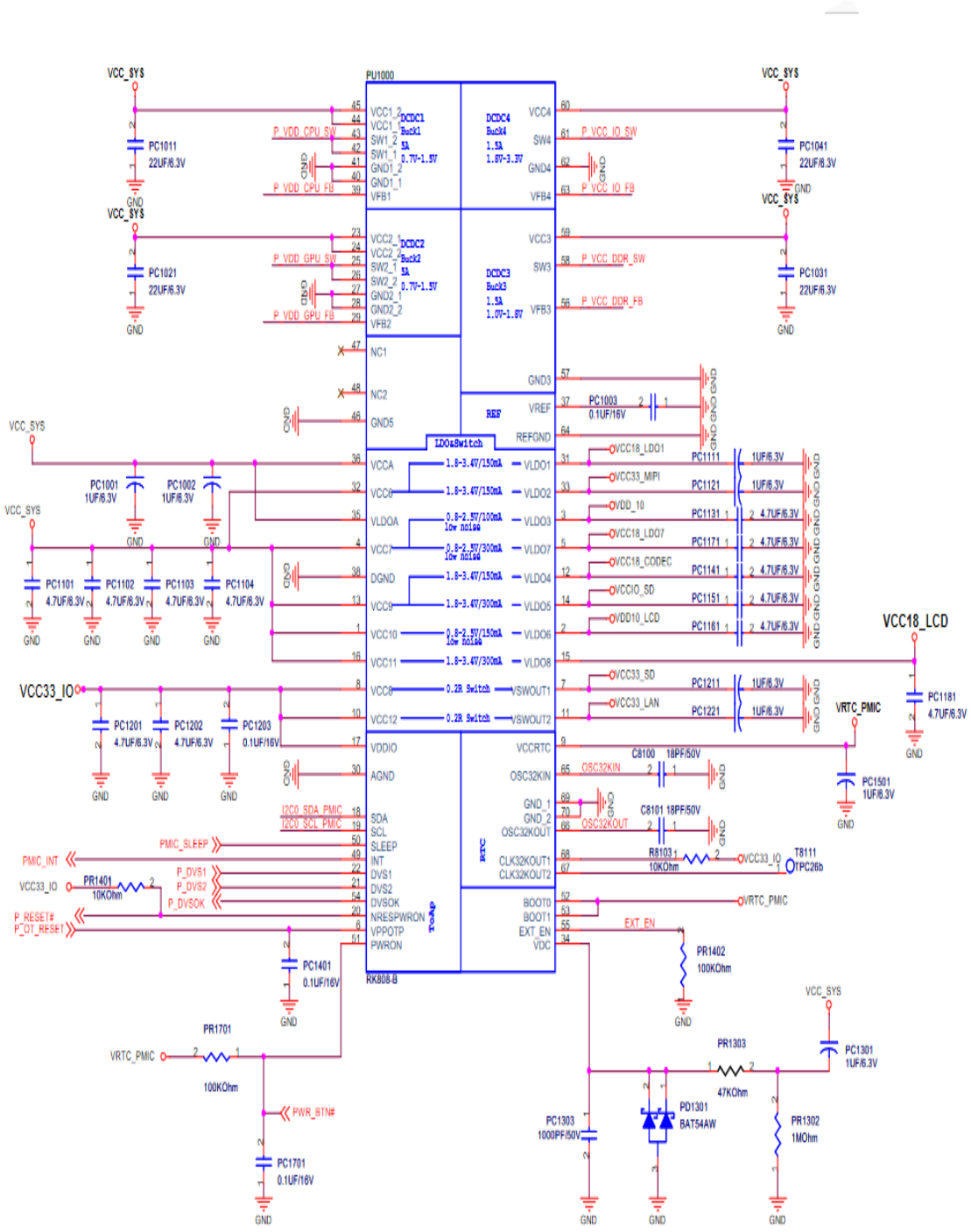


Figure V - PCB Design Prototype

Figure W – PCB Design Architecture

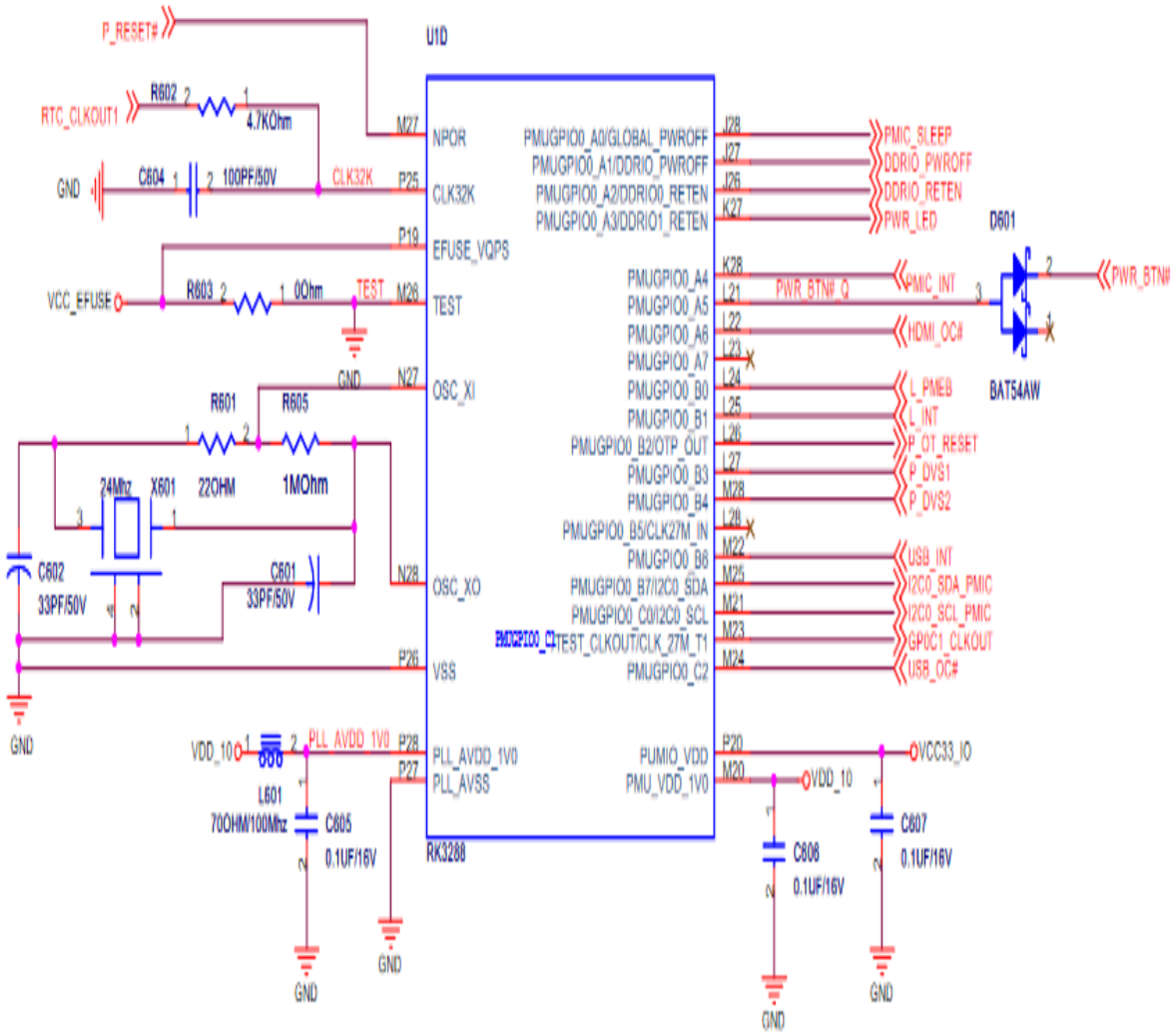


Figure W– PCB Design Architecture

Figure X – PCB Layout

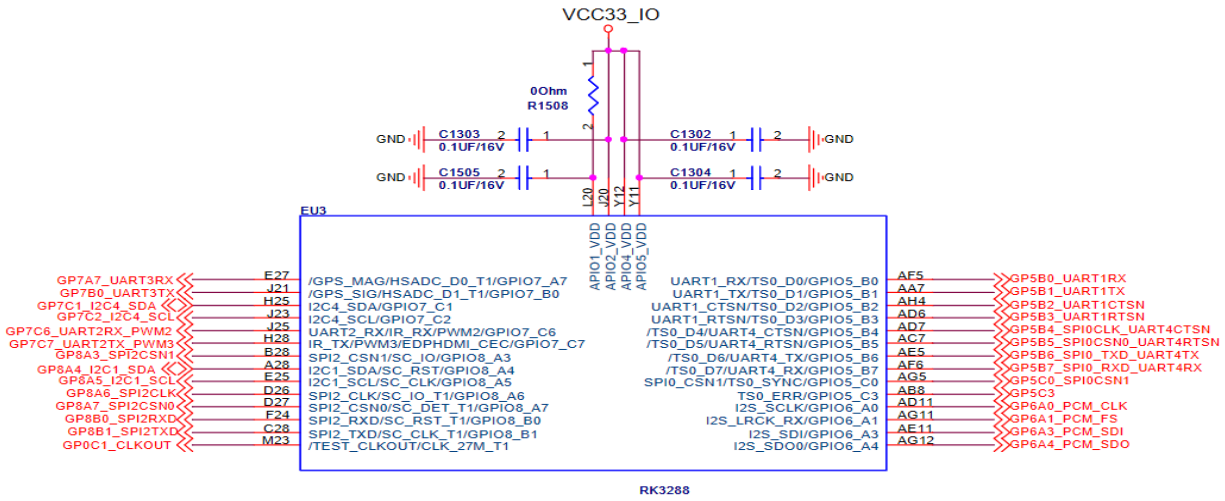


Figure X - PCB Layout

9.2.5 Printed Circuit Board Software Design Application

For our PCB, we will first need to design a schematic from scratch. We will download a schematic design software by searching on Google an application called “DipTrace”. This application is very user friendly, as it has multiple help sources on the internet. DipTace is also a free application, saving us more money on our design. Once on the DipTrace web page, we will click on the download button and boundload the “DipTrace 3.1 Freeware”, which is dedicated for a Windows PC. “DipTrace 3.1 Freeware” gives us two signal layers PCB, for non-profit usage only. This program allows us to create a PCB with 300 pins, allowing us to have up to 300 connections on our design.

Now that DipTrace has been installed to the PC, there are two important aspects that are important. The first being the schematic design and the second being the printed circuit board design. The first thing we will need to do is design the schematic, then we continue on the printed circuit board design. These two designs are linked together (the schematic design and the printed circuit board design) in the application DipTrace. If we ever need to edit the schematic, the printed circuit board will be automatically updated, as they are a union that are connected together.

To drag the screen around the screen on the editor, you can use the right mouse button. To select components, there’s a column on the left hand side that is called libraries. Each of the libraries hold different components. The discrete library holds capacitors and resistors. All components belonging to the library can be found on the lower window. At the top of the page there is a mouse pointer called “default selector mode” which allows you to

move components around the editing page by clicking on the item you want to move. The mouse wheel helps to zoom in and to zoom out giving more space options to move the item. Every item on the editor screen is context sensitive, which means that if we want to know anything about a component we have clicked on or edit some aspect of the component, we just hover the mouse clicker over the component and right click on it, go to properties and change the value, the name, or the manufacturer.

When connecting two components together, at the top of the page there is a button called "place wire". Clicking the end of one component and the clicking the end of another component, creates a connection between two components. Another thing that the components can do is be rotated by right clicking over the component and selecting the rotate button. Doing this process until the component is rotated at a desired position. The basics are zooming, dragging, connecting and rotating components on the editor screen.

We need to find a component in the standard libraries. Manufacturer names are listed alphabetically. There is a search filter called "Filter Off" on the left side of the screen, under the libraries column. Onced clicked on a search bar appears on the side of it. Here, all libraries can be searched by typing in the component that is needed. Once a component is found, the schematic diagram and the PCB footprint of the component is shown once the component is clicked on. Components contain two aspects, the schematic part and the PCB footprint.

If we want to change the color of the wires in the schematic design, we can right click on the wire and click on "Net Color". This permanently changes the color of the wire , making it easier to navigate and understand the schematic. Like this we can create our PCB design with the DipTrace application. It all depends on design constraints and preferences

Now moving on to creating the printed circuit board. We link our schematic to the PCB design page by going to file, clicking on "Renew Design from Schematic" , then click on "By components" and choose the schematic we created from the file window that opens up. It automatically is placed on the PCB editor screen. The schematic now looks like PCB patterns of our components from the schematic. All the pins are connected just like in the schematic design. If we would like we can see the 3D version of the model, by clicking on the 3D button at the top on the editor page. This allows us to the printed circuit board, however it is not encased because no outline has been drawn on the editor page around the printed circuit board components. Creating an outline for the printed circuit board, we click on the button at the top of the editor page called "Place Board Outline". Now, clicking on the 3D button at the top, the printed circuit board is shown with the traditional green base because an outline has been created. To edit the outline, when needed, we hover over the outline on the editor screen and adjust by holding the click on our mouse. To have a more precise outline edit, right clicking the mouse and selecting "Board Points" on the menu allows us to enter number values for each corner point.

The printed circuit board components on the editor are connected with elastic type on lines, making the process of arranging them easier. We arrange the components so that they are closer to the components they are connected to. For example, if one component is connected to

all other components, it's best to place that component in the center and adjust all other components around it.

On this editor we can also add some mounting holes. Mounting holes button can be found at the top of the editor screen. It's called "Place Mounting Holes". We apply them on to each corner of the printed circuit board editor with a click of the mouse. To adjust the size of the hole, right click the mouse and select "Hole Properties" to manually enter number values of the specific hole diameter in millimeters.

Once the components have been arranged to our desired location, we connect them together with the button at the top of the page labeled "Route Manual". This button draws tracks on the printed circuit board editor sceen, from one component to the next component. To edit the track to make it thicker or thinner, right click over it and select the "Trace Width" button and change the size in millimeters. The thicker the track is the better.

9.2.6 Final Outputs- Network File Name Generated by PCB Editor Designer

When the final process is done, what you ultimately create is a set of artwork. The artwork files are called Gerber files. Gerber files provide some indication as to how they should be laying out the copper and how they should be drilling the boards to the PCB manufacturer. Some of the files may be labeled as "TOP.art" and "BOTTOM.art", these would be the top and bottom copper layers. As well as "Silkscreen_Top.art", Soldermask_Top.art", Soldermask_Bottom.art",and "Solderpaste_Top.art". In addition there will also be a file called drill. Drill files tell the manufacturer where to drill holes in the XY plane and it also tells them how big those holes need to be. These files are some of what gets submitted to the manufacturer and are reviewed when it goes through Design Rule Check (DRC).

9.2.7 Standards

There are certain standards that are used in printed circuit board design. A common standard used is IPC-2221. It is a generic standard for printed circuit board designs. It comes from the institute for interconnecting and packaging electronic circuits, IPC. The critical elements that it outlines are some things that we need to know when we are designing boards. In this design standard, it gives approximate voltages between conductors, such as 0 and 15 volts. Depending on where the conductor is, whether it is internal or an external conductor. There are different widths that need to be maintained between the two conductors. As voltages go up, the distance between them goes up as well. Units in between conductors are measured in millimeters. In PCB design, very often the units are in millimeters. Another unit that is often seen is this unit called Mills, which is thousandths of an inch. The IPC-2221 standard gives us guidelines, like how to space conductors to prevent dielectric breakdown.

Another thing that we are looking into, is what size are the traces. How do we decide what those traces need to be? The width of traces is basically set by the current. If the current is

high, then we wonder how much voltage is being lost as the current goes through the resistance or this trace.Lastly, how much power is being lost , that the current traverses through the tace. We would like to minimize the resistance as much as we can. Resistance is a tricky subject. Often, temperature rise is thought about. Lots of current through the traces, makes power dissipation. Power dissipation, means the temperature will increase. So,how small should the trace be made? It depends on the temperature rise we want to have for a given current.

9.2.7.1 GS1 Barcode Standard

GS1 is the organization that manages barcode standards used in retail. It was originally called The European Article Numbering and originated in an office in Brussels. These barcodes are the ones we will be working with in our project. GS1 standards cover the EAN and UPC based barcodes along with databars, 1D, and 2D barcodes. It is thanks to these standards that we are even able to attempt this project. It allows for products to be scanned and identified in a common fashion and allows us to identify products we want to keep track of.

9.2.7.2 EEE 802.11 Standard

The IEEE stands for Institute of Electrical and Electronics Engineers and it is an organization that provides standards for local, personal, and metropolitan area networks. Since our Wi-Fi module is 802.11 b/g/n it will adhere to 802.11b, 802.11g, and 802.11n standards. The 802.11b standard can reach a maximum raw data rate of 11 Megabits per second. Devices using this standard can experience interference from other objects or products that are also on the 2.4GHz frequency band. The 802.11g standard also works in the same 2.4 GHz frequency band but uses an orthogonal frequency division multiplexing transmission scheme, similar to the 802.11a. It's maximum bit rate on the physical level is 54 Megabits per second, and around 22 Megabits per second average throughput. It is backwards compatible with 802.11b but this reduces it's throughput due to the legacy issues that come with it. The 802.11n standard improves on the previous 802.11 standards and came with added support for multiple input multiple output antennas. It is able to operate on both the 2.4 GHz and 5 GHz frequency band, however support for the 5 GHz band is optional. The overall data rate is within the range of 54 to 600 Megabits per second. This standard is a key feature for us as it allows us to communicate our device and our application to the internet and will enable us to communicate data. Thankfully our Wi-Fi module gives us multiple options to choose from, but the 802.11n will be the fastest option we could use.

9.2.7.3 Linux Standard Base

The Linux Standard Base is a project to standardize the system structure of the software of Linux and is a joint effort between many Linux distributions under the Linux Foundation. This includes the Filesystem hierarchy Standard that is used in the Linux kernel. The purpose of the Linux Standard Base is to develop and promote open standards for the sake of increasing compatibility among Linux distributions, and to enable software applications to run on any system that complies to these standards, even in binary form. The Linux Standard Base specifies the standard libraries, specific commands and utilities within Linux, and the layout. In our project our hardware will be running on a Linux based operating system, and it is thanks to these standards that we are able to use libraries and tools previously created and freely shared. This

saves us precious time, so we don't have to manually develop our own, and gives us an ease of use because it is easy to understand as well.

9.2.7.4 Power Supply Standards

There are many organizations that create standards for power supplies. Even on our Tinkerboard, one of these companies' logos, CE, appears on some of the smaller parts. The CE mark means that the device it is on conforms with the protection standards sold in the European Economic Area. This means these specific components have passed health safety and environmental standards by CE.

To sell power supplies at all, whether they are internal or external, requires that they meet safety standards based on the area they are being built. The main focus of safety standards for power supplies is to reduce and protect against fires, electric shocks and injuries in general. Any product that meets these requirements will usually be marked based on the region it was built and tested in.

There are three different classes of equipment that these standards apply to. Class I equipment requires that all conductive parts that can attain a dangerous voltage in the case of insulation impairment or failure, must be connected to a protective earth ground. Class II equipment must provide protection using stronger insulation, either double the normal amount, or reinforced. This means no ground is required for class II. Class III equipment operates using a Safety Extra Low Voltage, or SELV, supply circuit. This means it automatically protects against electric shocks, because it is not possible for any dangerous voltage to be generated in this equipment.

The IEC or International Electrotechnical Commission and ISO or International Organization for Standardization are the main organizations for electrical safety standards. Then there are regional agencies that operate in different countries and keep these standards in these countries. This means identification of certain standards will likely be labeled by the regional agencies, to identify the country where the piece was certified.

These standards apply to pieces in our board as well as our power supply for our board. It is thanks to these standards that we can operate with these components in a much more reliable and safe way. Without these, our boards may be inoperable far more frequently or could be prone to disastrous consequences like fires or electrical discharge.

9.2.7.5 IPC Standards

IPC stands for the Institute of Printed Circuits. It is an association whose goal is to standardize both the assembly and production requirements of electronic equipment and assemblies. IPC has been accredited by the American National Standards Institute (otherwise known as ANSI) and is recognized worldwide for its standards. These standards are used by the electronics manufacturing industry, and in regard to our project, applies mostly to our Tinkerboard, and our PCB. There are many standards that may apply to our board specifically covered by IPC. There are IPC standards for general documents, which include terms and definitions and data about printed boards. Design specifications are also included in IPC standards, and range from sectional requirements to printed board design, to land patterns and surface mount designs. Another aspect to IPC standards is material specifications. These include standards on the metal foil, laminate, flexible bases, adhesive and dielectrics. There are also

performance and inspection documents included in the IPC standards. These specifications relate to the wiring, acceptability of the board or device, performance of the board, and covers manuals for the boards. Some of the specific IPC standards that may apply to the boards we will be using include but are not limited to:

- Acceptability Standard for Manufacture, Inspection and Testing of Electronic Enclosures
- Requirements and Acceptance for Cable and Wire Harness Assemblies
- Acceptability of Electronic Assemblies
- Requirements for Soldered Electronic Assemblies
- Acceptability of Printed Boards
- Qualifications for Printed Boards
- Base Materials for Printed Boards
- Design and Land Patterns
- Data Transfer and Electronic Product Documentation

Each of these have their own IPC Standards codes that they adhere to, but are also connected to other IPC standards that are related to these and relevant, but don't need to be listed. It is thanks to these standards that we are able to rely on our boards. Due to these standards, we can be assured that our board is durable and reliable. Our project would not be possible without the use of boards made with these standards.

Figure Y - Listing of IPC standards. Image courtesy of Ipc association

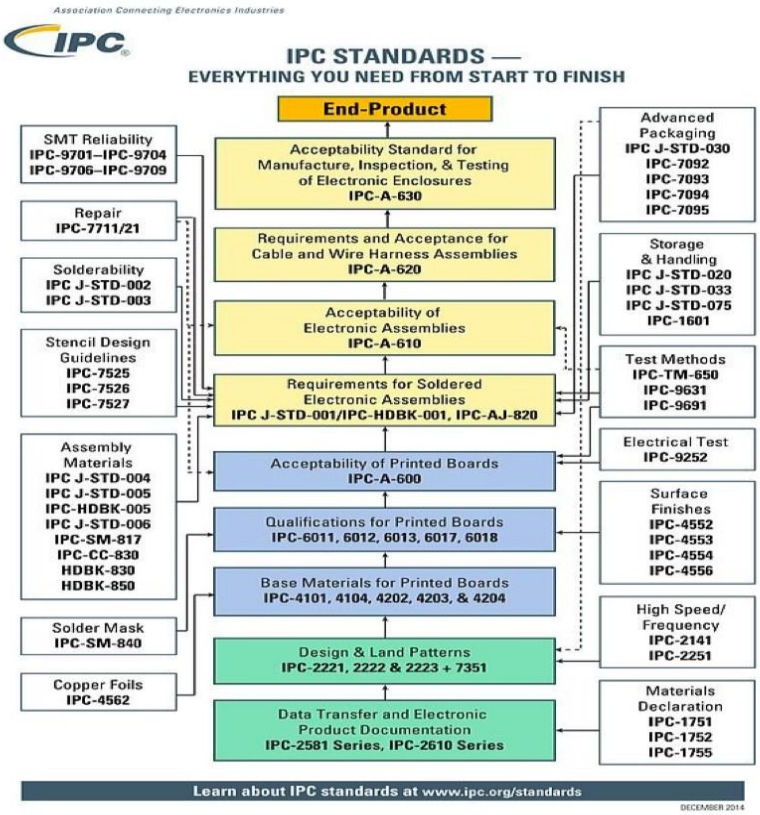


Figure Y - List of IPC Standard

9.2.8 Trace Width Calculator

A simple way of calculating the temperature rise we want to have for a given current is this calculator called Trace Width Calculator. It is taken from this IPCC-2221. We can input a particular current in the calculator. For example, let's say the current in 0.1 Amps going through a particular trace. Let us also say the ambient temperature is 25 degrees celsius. The trace is 1 inch long and we'll allow a 1 degree temperature rise. This means The temperature can use only one degree above the ambient temperature. If the trace was located in an internal layer, it says the trace width requirement is 2.60 Mills (1 Mil = thousandth of an inch). So, the trace needs to be 2.60 Mills wide. The calculator also tells us how many Ohms are associated with that, which is 0.0938 Ohms. If the trace is external, the required trace needs to be a width of 0.999 Mils. The internal layer gives us a wider trace than the external layer. Also the resistance of the internal layer trace is lower than the resistance in the external layer trace. We expected that a wider trace (Internal Layer) would give a low value resistance. For the skinner trace (External Layer) would give us a higher resistance. The critical specs we give as inputs is how much of a temperature rise do we want to have. So, on the external pieces of the board, air, which moves over the board. That power drop/resistance doesn't matter as much because the air is able to move over the top of the boards and carry the heat away. The internal layer of the board makes that heat hard to get out of there. So, I need a thicker trace, a wider trace to be able to the same amount of temperature rise. The critical things the calculator wants to know is how much current and how much temperature rise do you want to have.

9.2.9 Copper Weights

Another consistent is, thickness of copper. How thick is the copper? How thick is the copper in the z-direction? How thick is it off the board's surface? If there is an amount of 2 ounces of copper, it is basically saying that, in one square foot copper has been poured to make it weigh 2 ounces. Which equates to the copper being 2.8 Mils thick. Ounces is a unit of mass not for thickness. 1 Ounces - 28.3495 grams. 1 ounce and 2 ounce copper is sort of the standard values that we utilize. When doing a lot of power work, we often refer to the 2 ounce copper.where it is going to be thicker. In power circuits the current is going to be higher and typically the traces will be thicker. The thicker the copper gets, the lower the resistance is going to be.

9.2.10 Device Information

Ultimately we choose our packages. When we are capturing and editing our printed circuit board schematic, we may be interested in a specific component such as an op-amp LM324-N. The LM324-N has four different package types. There are two "DIP" chips (DIP = Dual Inline Packages and two basic surface mount components, one of which is a TSSOP package and the other is a SOIC package. The names of each other different packages are industry standard foot prints for four packages. Typically in the datasheet, there is mechanical

information that tells us information of the footprint of the device. This information will be very useful to us, when making sure that we align and have our holes. For Example, an op-amp will be considered a through-hole component, where the pins will go through the printed circuit board. We need to make sure the holes are placed in the right spots and that the pads that are placed around those holes are of the right width. This information about the footprint of the component is very useful to have. To make sure the board itself is set up properly.

9.2.11 Coronavirus 19 Pandemic

On March 11, 2020 the World Health Organization declared the Coronavirus Disease (COVID-19) as a pandemic. Some time after this, researchers began to understand the virus genome more and were able to relate it genetically to a previous coronavirus outbreak that was responsible for the SARS epidemic in 2003. The virus was renamed as the severe acute respiratory syndrome coronavirus-2 (SARS COV-2) but is still widely recognized as the coronavirus. The virus is spread from person to person through respiratory droplets which can travel through the air when we breathe, speak, cough or sneeze. Social distancing and wearing masks have been recommended and mandated in some cases.

On March 16, 2020 the University of Central Florida shifted all classes entirely online in the middle of the semester. In Spring 2021, we are still almost entirely remote. While UCF has been able to bring some classes in person, there are limitations made to the class sizes, the number of classes available in person, and the days a student is allowed to show up in person. Due to the shift to online classes many students had to move, because their sole reason for being near the University was to attend classes, and no one was certain when classes in person would resume. At least one of our teammates moved further away from where they were.

This has caused a couple of constraints for us. First, being online can make it harder to learn. Being able to go in person is preferred by our group, and not having that makes it harder to focus and harder to learn. We also now have an obstacle of meeting up. We're able to communicate through online platforms like Zoom or Discord, which has been helpful, but meeting up in person is harder now because of how distanced we are, and the fact we need to take precautions before meeting.

Another issue this will cause in the future for us is meeting to work on the physical parts of our project. Since the University is still limited access, the labs only allow limited numbers of people in, and in our group only two of us can use a lab together at a time. We believe the added resistance to meeting up makes it harder for us to connect as a group, work together, and time our meetings properly. Around the start of our project, one member lost a grandparent to the corona virus.

9.2.12 Economic Constraints:

We lack a sponsor for our project and are funding it entirely by ourselves. Because of this we decided early on to set our budget to four hundred dollars. This means each member will pay one hundred dollars to make sure this project gets created. This was crucial during our design phase of the project, as we had to ask ourselves constantly if a certain feature was necessary,

because if we weren't careful, we could end up with something to create which we would not be able to afford.

This also means that when choosing parts, we are trying our best to be frugal and effective. Because of this we may choose something that is cheaper and functional but may not be the best option overall for us. To add on to this, we will likely need to use our own products when possible, to save money. Instead of buying our own fridge we can use one of the groups fridges when testing. When using food or other products we can bring our own instead of buying new ones, ideally.

One of the consequences of these constraints is that we are cutting out some physical components of our original design or considerations to save money, but to put this in a positive perspective we believe if we wanted to make this a consumer product, we should design it to be as cheap and user friendly as possible and believe this philosophy will help us create a simpler design.

9.2.13 Time Constraints:

The Senior Design Class started on January 12, 2021. The team originally consisted of two members until January 21, 2021 when the third joined. However, at this point we felt a need to change our original project as we were all Computer Engineering majors and felt the original idea would require at least two Electrical Engineers. Luckily, we were able to find a fourth member on January 26, 2021. They had joined the class late and we worked with them to get caught up on where we were at and the project we had chosen to do. As a team we formed a little late, and this definitely cost us valuable time. Because we started our project in the Spring, we also had the option to finish our project in either Summer or Fall. The four of us wanted to finish it in Summer, but this means we now have about four or five months less to finish our project and may result in us not being able to implement every feature we originally wanted to.

Another component to consider is our lack of experience. Not every member has hands on experience working with or creating the components needed for our project and as a result we must spend time learning and researching to understand it to the best of our abilities. Our members are also taking other classes they need alongside working on our project and those reduce the amount of time we have available to spend working on the project.

9.2.14 Ethical Health and Safety Constraints:

One key ethical constraint is that we will be handling our user's email addresses and passwords. It will be our responsibility to make sure our system is safe and protected so users will not have to worry about getting this information stolen. While we do not need to save their names or other personal data, that does not mean we should be lax about our security. It is common for people to use the same passwords for multiple devices and accounts. This means if our saved passwords were to be cracked it could risk our users other accounts safety. If their email addresses are stolen from us, then we run the risk of someone attempting to send them phishing emails which could be dangerous as well.

Since our project is based around managing an inventory of food, this is a huge concern for us. We want to be able to manage expiration dates on our application to remind our users when is best to replace their food, or to alert them if it has gone bad and needs to be thrown out. A major concern with this is if our system does not alert properly, it may convince users that their food is safe to eat when it isn't. This is something we need to take into consideration when implementing our design and we will need to consider including a disclaimer to either remind them to check their foods integrity manually, and we will likely need to research into different foods expirations.

9.2.15 Previous projects HW designs:

A project with a similar premise to ours is called SMIDGE. This group worked during the Fall of 2010 through the Spring of 2011 and consisted entirely of computer engineering students. Because they lacked an electrical engineering student, they opted to buy more pre-built components to realize their goal as opposed to making their own. Their system is designed to be placed directly onto a refrigerator. This includes the LCD screen, the development board they use, and the scanner. The idea is to have the scanner placed on the side of the fridge with the screen on the front, so that the scanner can be used whether the refrigerator is open or not. Their system is powered through the same cable that powers the refrigerator but through a regulator will limit the voltage to 5V. Using a Wi-Fi adapter, it will connect to the user's network so it can communicate with their remote server. Their system will be able to handle the same data their mobile application and website can and will only need to sync occasionally instead of consistently. All of their hardware will be connected compactly around the refrigerator itself and be similar to an accessory of the refrigerator itself. Their budget was 600 dollars.

Their website interface is another important component of their project that should be explored. Using a website will require the user to create an account in order to sync with information on the system, and to act as an identifier so multiple users do not end up sharing information. The account requires an email and a password. Their interface is broken into three sections: inventory, shopping lists, and recipes.

Inventory is what is currently owned by the user and registered in the system. The shopping list will be based on user input and items already in the system. It will be able to retain previous data on other items that have been bought or searched for to help users find things they might have previously liked or things they habitually buy. The recipes will be entirely based on user input, as a digital log for whatever recipes they may want to keep.

The next aspect to look at in their project is their mobile application. The mobile application is the defining factor for the SMIDGE project. While it is capable of the same functions as the web application, it allows for easier use of those functions on a mobile handheld device. This is where the shopping list function really stands out. Having access to this on a mobile device means you don't have to carry around a paper of the list, and can use it while shopping, and can update things as needed. This includes but is not limited to product size, product weight, availability, recipes and ingredients. Another aspect of the mobile application is the notification system. The notification system will monitor different aspects, both physical and digital so it can alert the user of various issues of importance. It first needs permissions from the user to work, but once this has been satisfied it will be able to alert the user of expiration dates, alert the refrigerator temperature based on the sensor, and alert if the user is running low on some product. These alerts can be through either email or the application based on user preferences.

This project gave us some good material to discuss. When we first started we we're unsure and this gave us something to talk about, and let us consider certain components we may not have at first, and let us know what already existed so we could understand and try to make our project unique so it stands out.

9.2.16 TinkerBoard Operating Systems

There are a couple options for Operating Systems on our ASUS TinkerBoard. The first is a default OS that comes with our board called TinkerOS. It is based on Debian which itself is a linux based operating system. Another option is Lubuntu which is another Linux based operating system, but we can also use an Android operating system. For our project, we primarily need our board to be able to retrieve data from our scanner and send it to our application and database.

9.2.16.1 DebianOS

Debian itself is just a specific instance of Linux and the packages that come with it. It is free to download and use, with the exception of specific packages. For our project we should not need to rely on these packages, so in our case Debian is completely free to use. Debian acts as a seed for many other Linux based systems as well including Ubuntu, SteamOS, and Knoppix.

Currently Debian includes over 58000 software packages available for installation and provides its own tool for installing these packages. It also has over 1300 developers contributing to packages and as such, packages can update regularly, and Debian is able to expand at a rapid pace. There is a list of packages available online to view, and with this we can determine if it will fit our needs. Most importantly it needs to be able to use MySQL. It contains the packages but also contains a fork for MariaDB which is a MySQL variant. This may cause us to run into issues.

9.2.16.2 Lubuntu

Lubuntu's name is a combination of LXDE and Ubuntu. LXDE stands for Lightweight X11 Desktop Environment and is the default GUI for Lubuntu. It is designed to be a lighter, energy efficient version of Ubuntu. Lubuntu is primarily aimed at users looking for an operating system on low spec hardware that may not have as many resources as stronger computers. When Lubuntu reached version 11.10 it achieved official recognition as a part of the Ubuntu family. This means it is able to access and run any application Ubuntu can as long as the hardware is capable of it. The main difference between Lubuntu and Ubuntu is the LXDE. To understand more about Lubuntu we will look into Ubuntu as well.

Ubuntu is a free complete Linux operating system. It is built on the philosophy that the software should be available, free and accessible for anyone to use and customize as they would like to. It is updated regularly with a new release every six months, while every two years a long term support release will become available and supported for five years. It is available for bother desktop and server use and supports many processors including Intel, AMD, and ARM. It includes thousands of software including Linux kernel version 5.4 and GNOME 3.28 and covers standard desktop applications from email software, programming languages and word processors. It works alongside Debian and may allow for certain packages to run while Debian may not support it due to a build failure an any of it is 11 architectures. This may be important for us if we need a package that may fail to build on Debian.

9.2.16.3 Android

The Android operating system is an altered version of the Linux kernel and other open source software, and is used primarily in touchscreen mobile devices. It is a free, open source software and its source code is recognized as Android Open Source Project or AOSP. Android 11 is the most current version available of the operating system. Version 12 was announced on February 18, 2021 but has not been released. However, our Tinkerboard can only use up to Android version 10. One difference with this operating system is we may not be able to access the database directly like the other ones can. However, since it is Android, we should be able to run the application on it instead to access, store and retrieve data if needed.

9.2.16.4 Decision

Since we decided not to use a touchscreen interface for our hardware, running an Android operating system would be a little difficult and tedious. We would likely need to find a touchscreen to be able to access the interface on the board while working on it and would end up being a costly expense just to test. Also, since the Android OS would likely need to run the entire application being developed, this would not be feasible without a screen. Due to this we will not use an Android operating system for our board.

Instead, we will opt for the TinkerOS based on Debian to start. Since it is enabled initially on our device it will be easy to start with, and if we run into issues of packages not building, or some other unforeseen issue, we can rely on Lubuntu as a backup. Since they both run on Linux and use a lot of the same packages it should be an easy transition if needed.

9.2.17 Coding on our Board (ph)

The purpose of our board is to take the data from the scanner and transmit it wirelessly to the database for the application to access it. This means we need a program that automatically starts up when our device is turned on, waits for input from the scanner, and then sends that data to our database. This is why we are going to be relying on the mySQL packages on our OS. Ideally, we could also add a low power mode to rest while not in use, or an auto shut down to save power. One other issue we will need to solve, is being able to connect to the internet on our device without a screen.

MySQL works with a variety of other languages like C++, C, PHP, PERL and JAVA. PHP is generally the most popular because of its web application capabilities, however, our Tinkerboard does not need to do much outside of communicate with the database. We will opt to use C++ due to the existence of helpful libraries that we would not have to build ourselves in C. This should allow us to collect the data from the scanner input and configure it for MySQL to read and use.

9.2.18 Digilent Ultra Analog Discovery 2 Bundle

This bundle contains:

- USB A to micro B programming cable
- 2x15 flywire signal cable assembly
- 5 pack of 6 pin male headers
- Ferrite cable snap on
- 5 pack of Mini Grabbers with Leads
- 6 pack of Mini Grabbers with test clips

- 2 BNC to alligator clip cables
- 2 BNC to Mini Grabber cables
- Analog Discovery 2x15 Ribbon Cable
- Solderless Breadboard Kit: Large
- BNC Adapter
- Breadboard Adapter
- Breadboard Breakout
- Impedance Analyzer
- BNC Oscilloscope x1/x10 Probes (pair)
- Project Box

This equipment has been lent to us from the lab at the University of Central Florida to help give us the tools needed to test our electrical components. These pieces will allow us to check our resistors, capacitors, and inductors to make sure they are working, and allow us to build and test on a breadboard before finalizing the project.

9.3 Testing

9.3.1 Hardware Testing Environment

When buying our parts and equipment, our goal is to have one individual buy each part so the whole of our project will be together from start to finish. Depending on the piece for the project we may need to test some of it at a member's house to make sure it works prior to bringing it to the lab in case something needs to be replaced. When testing each component, we need to consider that it works individually, and works with the other components it will interact with. Since certain components like the scanner will already be encased, they will be safe to test inside of a house without risking any damage to its parts. With our ASUS Tinkerboard, we will probably test it individually inside of the lab, as to reduce the risk of a static charge damaging it accidentally or through our own error.

The lab we will use is the senior design lab at the University of Central Florida. The lab is in the Engineering I building on the fourth floor in room 456. The room has a limit on how many people can be in there and will be consistently clean and equipped with tools to help us test our components. The equipment available to use inside the lab consists of but is not limited to:

-Tektronix Oscilloscopes

-Tektronix Dual Arbitrary Function Generators

-Tektronix DMM 4050 Digital Multimeters

-Keithley 2230-30-1 Triple-Channel Power Supplies

-Dell Precision 3420 Computers

-SMD Rework Station

-Soldering and Desoldering Stations

-Digital Microscope Inspection Station

To access this laboratory, we will need to contact someone and schedule regularly to use it. We will likely need to create a schedule to work there twice a week and cycle our members out as needed.

9.3.2 Hardware Testing

First, we plan to look at the scanner. Since it is sturdy, we can test it at the home of one of the members. Initially, we need to make sure the scanner can turn on, and we need to test that the battery can both charge properly and hold a charge properly. We will need to test that the Wi-Fi connection and the wired USB connection work as well. The scanner works similarly to a keyboard and the inputs will be similar to keyboard inputs when plugged into a computer. We can use one member's computer can be used to test this functionality first. Once this is confirmed to be in order and does not need to be returned or replaced, we can bring it to the lab and test to make sure it works properly with the Tinkerboard before testing it with our software. We will also need to test the scanner's ability to read barcodes and make sure this works properly. Since barcodes come in various sizes and on various surfaces and curvatures, we'll probably need to grab some household items we each have to give us different testing materials. One approach we can take is to scan multiple times to and look at a success rate.

Next, we will need to test our Wi-Fi adapter. This will likely be done in a similar manner to our scanner where we test it first at a home to determine if it is working with a regular computer before bringing it to the lab to test with our board and make sure it can still connect to the internet. We will likely need to test it's distance to make sure its range matches closely with what was advertised.

Once this is done, we will need to test our ASUS Tinkerboard. This will require a monitor or television which can be acquired either in the computer lab or provided by one of the members. The first step will be to make sure it turns on and works properly. After this has been confirmed and it does not need to be replaced, we will have to install the proper operating system on it. To properly test the board, we are going to need to create a program on it to take in an input. The input should be from the scanner and it should act as a standard keyboard input, but this needs to be tested to make sure it works properly.

After this is done, we need to make sure the board can connect to the internet. We can connect our Wi-Fi module to our board and through the installed GUI on the Tinkerboard we should be able to connect to the university's Wi-Fi connection. It is not needed to connect directly with the phone being used to test the mobile application. Instead, we will have it connect to the internet and send the relevant data to the database for our user. Once it has been confirmed to connect to the internet and read scanner input, we will need to test an authentication process. We either need to link the physical board with a user identification or have some way to send the

device a specific user ID through our application, so we can use that information to access the database.

Once this data has been sent to the board and we can confirm it, we will be able to attempt to access the database. The user identification is important for this since we need to send the relevant data specifically attached to a certain user, whoever happens to be using the device at the time. This should be a part of the program that accepts the data from our scanner. Once this has been completed, we can begin to test our software side with this data.

Our work with the board is not finished here though. We will need to implement a few other key features after these. The first of which is auto starting our program once it is turned on. We will need to make sure this happens whenever the device is turned on after our program has been confirmed working because we will not be using a screen in the final build. The monitor we use to test will allow us to confirm that this process works without any input outside of our scanner. Once confirmed with the screen the process works, we need to test the same process with our mobile application to make sure there are no errors while no screen is being used, and to make sure our mobile application receives the proper data.

After this we need to make sure the on/off button works, and we may also spend some time, if we can spare it, to test a low power mode, in case the device is left on accidentally. This could help increase the lifespan of the device and reduce costs and energy consumption.

We will also need to test our power source. Prior to this the board should be able to take some power from the monitor, or from a provided power source, but we will need to test it with our own.

9.3.3 Preparing to Test a PCB Design

PCB design involves a lot more than getting electrons to flow through a circuit on a printed circuit board. PCB design includes how well a designer can layout a PCB to flow efficiently through the manufacturing process line for fabrication, assembly, and testing.

what is needed to be considered when tasked with implementing DFT into PCB layout. When looking at a finished PCB board, one might ask why would anyone want to test this PCB, when it already looks perfect? PCB designers don't often see the failure reports for some of the components on a PCB that has been assembled. Sometimes boards that may look perfect and fine may be rotten with manufacturability issues. It is a good idea to test PCB boards to check for manufacturability issues. If testing of a PCB board isn't looked into, it will show up again and will be flagged during final inspection. If a PCB design is caught in the final inspection phase, it will result in a need to redesign, resulting in more expenses.

The PCB layout engineer would benefit from implementing PCB assembly In-circuit testing. Also, benefit from using the SMTA/TMAG TP-101E testing guidelines and general guidelines when implementing DTF to the PCB layout.

9.3.4 Who Tests PCBs?

9.3.4.1 The Stakeholders for Testing

Let's talk about our testing stakeholders, something all PCB engineers should acknowledge is that in the manufacturing world every process and every material is tested for conformance every step of the way. In PCB manufacturing, material doesn't move without testing and verification. Do PCB designers take the same approach? Do we go on our next design steps only after checking, testing, and verifying the design constraints on our part? Do we, as designers, do as good of a job as our suppliers? With regard to asking the right questions and identifying and engaging and querying our project stakeholders to implement enough checking throughout the course of our design mission?

Let's identify a few PCB project stakeholders and also look into their requirements. The first stakeholder is the fabrication supplier. The fabrication supplier is required to test for continuity and shorts on the bare circuitry. The next stakeholder is electronic engineers. The electronic engineers have requirements for probing and debugging an assembled prototype circuit. The third stakeholder is EMS test engineers. The EMS testing engineers requirements for testing and measuring manufacturing process quality.

PCB layout can address each of these various stakeholder requirements, but PCB engineers need to understand that all these three different stakeholders have different test and verification requirements.

9.3.4.2 The Fabrication Supplier Stakeholder

Fabrication suppliers are given manufacturing data through Gerber files (or ODB++ files or IPC-2581 files or Excellon files). They use this data to process the bare PCB, using flying probe test equipment. The fabrication stakeholder supplier needs data in order to test their work. Their work is to get the bare board print and etch operation done, so that there are no shorts and continuity is consistent. This requirement does not require physical adaptability in the layout, with the proper data, fabrication suppliers can get to all the testable points that they need via component footprints and contacts. The fabrication supplies also use a flying probe test to tag net points on the surfaces of the PCB to verify continuity between traces and inner layers. The components cannot be checked at the supplier. This is simply just the bare board supplier, who is creating a component in the form of a bare PCB. When supplying good Gerber data to the fabrication supplier, it will be a top priority to include an IPC D356 netlist file. Also note that many offshore suppliers do not have the capability to deal with intelligent date formats. Some 90% of the world's PCB production is still being processed with basic Gerber (or Excellon) data format. In order for the offshore suppliers to test the board for continuity, they expect the IPC D356 netlist.

9.3.4.3 The Electronic Engineer Stakeholder

Another stakeholder is the electronic engineers, who are typically most familiar with the PCB design. They originate from the front-end design criteria and have a lot to say with regard to functionality. Electronic engineers sometimes foresee a need to add test points onto a PCB for

ease of measurement and verification of prototype circuits. Engineering test points or functional test points are added during schematic design. These test points are close to certain devices or circuits on the board that the engineer wants to test. Sometimes these test points are enough to allow for functional testing of the assembly, replacing the need for more expensive circuit test strategies. The schematic can call a test point into the layout, which may have its own reference designator, even a specific location for ease of manual testing access on the top side or the bottom side of the PCB.

9.4.4.4 The EMS Test Engineer Stakeholder

The third stakeholder is the EMS test engineers. The EMS test engineers work for the electronic suppliers to implement as much testing as possible on the boards they are going to be manufacturing to collect important data about the design as it runs through the process line to become a high quality PCB assembly. EMS test engineers must work with the design engineer to understand circuit functions and expectations. They also work with the EMS manufacturing engineers, to implement testing for manufacturing defects, which could occur on the PCBA. This is something that the electronics engineer has no visibility of. EMS test engineers are in charge of the quality of the entire PCB assembly. They need to have access to all nets on the printed circuit board, mainly from a single side. EMS test engineers develop in-circuit fixtures. They also develop routine software tests.

9.3.5 What will be Tested in the PCB Assembly?

9.3.5.1 The Most Common Mode Processes

Looking at what is tested on the PCB assembly, regarding visibility. Let's look at three of the most common processes modes for printed circuit board test inspection. First, there is an automated optical inspection testing that happens at the EMS provider, which is also known as AOI. Next, there is automated x-ray inspection, commonly called AXI. Last but not least is the process mode of in-circuit testing, which is commonly called ICT. The SNTA TAMG TP101E guideline includes a really nice picture of what potential failure modes the three processes inspectors' tests for and how their capabilities crossover. The diagram shows the three test modes that work together, and some have capabilities that others don't. The graph shows how many things can go wrong during the manufacturing and processing of a PCB assembly using this diagram. At the bottom of the graph, is the automated optical inspection. Here it can be seen that they will check for orientation, missing components, extra parts, inspection marks. For example, a ball grid array that is assembled onto the board, from a viewing standpoint, we can't inspect those solder joints, unless we get an x-ray. X-ray inspection is used widely for checking for insufficient solder or poor welding, marginal joints or voids. These are mostly mechanical checks. Automated optical inspection testing and automated x-ray inspection testing are mainly mechanical. In-circuit testing is an electrical test. It relies heavily on the incorporation of test points into the layout. ICT does, however, does check for some mechanical quality as well. In order to leverage this mode of testing, designers should include test points into the layout. Mainly concerned about in-circuit test as it relates to PCB design layout, we need to add test points.

9.3.5.2 Approaching the Conceptualization of the PCB

The conceptual stage of the PCB. This conceptual stage is the best time to implement DFT. There are many questions that must be answered before making the decision to provide testability in the layout, without addressing expectations before the design layout starts. During the conceptual stage there is little chance that all the PCB stakeholders' needs will be met. However, testing the board is all up to the PCB engineer. Usually, testing the board can be an afterthought. Posing just a few questions before starting the layout can be a good first step in saving the project lots of cost in the long run. When starting a project, asking or thinking about any possible test requirements and considerations. Creating a group of stakeholders to get after these questions before starting the layout.

Some conceptual questions to ask during the PCB conceptual stage, might be what category is this product? What is the chance of this product not working? Will this design be made in mass production? If this were a project that cost millions of dollars, PCB board designers aren't the people to make these decisions, one must consult with a stakeholder committee, or a company or an engineering team. Talking to a team about tests well before any copper gets laid out onto the PCB layout. The team must determine if they view testability or testing incorporated into the board as a cost or will the testing be looked at as an investment. This decision often depends a lot on volume that the project needs to produce.

9.3.5.3 Test Points

A test point is a small piece of metal end that is a connection point to test the circuit on a PCB assembly. The size of a test point depends on different aspects. If the designer wants to make sure the metal is big enough for a measuring tool that they want to use, they will probably make the test point metal bigger. It also depends on how much room is on the board. Is the board meant to be made as small as possible or does the size of the board not matter? Sometimes the board is called a "board under test" or "unit under test (uut)". The manufacturing processes that are involved. The size of the test point can also depend on the probe shape the tool that will be used to test has.

9.3.5.4 Test Probes

Let's look into the part that makes contact with the test points on a PCB, which is a test probe. Test probes are pins that are attached to an ICT unit. They make the interface connections between the PCB and the fixture. From a design standpoint, this is mainly referred to as probing test points and the metal pads are added to the surface of the PCB. These probes come in all different kinds of shapes and lengths, which is covered by the SMT STMEG SPEC. The test probes are shaped to make contact with many types of surfaces that are not only just metal pads. There are probes shaped to connect with connection pins, holes and many other features which can also be identified or tagged in the layouts as test points.

9.3.5.5 The Location to Best Place Test Pads

To implement DFT, one might wonder the best location to have a test point on a PCB. It is a very important part of this discussion. If it is not implemented with all the stakeholders in mind, failure at some point is going to occur. Once the determination has been made to implement DFT for in-circuit testing, the designer will do well to get familiar with one of the definitive guidelines for testability. It has been mentioned that it's a SMTA/TMAG TP-101E document available from the SMTA organization. This guideline does a good job of giving an overview of the world of PCB testing. It was created by PCB industry stakeholders to help all of the other industry stakeholders to understand their needs.

Test points are found almost anywhere on the board, such as the top or the bottom of the PCB layout. However, the points must be accessible after assembly and will not interfere with any of the other electromechanical parts or objects. Test points for ICT have key locational parameters, if they are not met, it will adversely affect the price of the ICT test pins in the fixture overall.

The SMTA/TMAG TP-101E spec guidelines give information about test point sizing and test point placement. These guidelines will keep the cost low during design. Test fixtures, schematically, are very expensive. The more complex they need to be, the more costly they become. It is best to start with the bottom side to access the test and also to avoid probing both sides. This is where we will want to start placing our test points, if it is determined that we are going to incorporate the DTF. To haphazardly, start placing test points access on the top and bottom because that would cost the test group to have to create, what is called a clamshell fixture, which is a lot more expensive and less accurate.

9.3.5.6 Size of Test Pads

Let’s talk about test point size. This being an expensive fixture, PCB designers must make appropriate measures to ensure that there is zero probability of test probe miss and there are ways to do that. One way to do this is to apply a large enough pad diameter to accommodate the probes without compromising the PCB topology. Enough to adversely affect electrical performance. Test points are a two-edged sword. Testability is often considered a two-edged sword. In order to get the test points in there, stubs need to be taken out and PCB designers must be cognizant of because that can affect and have an adverse effect to performance. So, communicating with the engineer about that would be a good idea. A test pad diameter that will give enough space for a probe is 0.035 on the top or 0.030 on the bottom. That is a pad size minimum, but a minimum specified in a capability spec doesn’t mean complete safety. So, applying 0.035 at the bottom is best, rather than the minimum, being 0.030, as specified in the document guidelines. Doing the minimum plus is best to ensure safety. Making the pad a square shape is also helpful to increase the area. Orthogonally, square test points don’t take up any more routing area or real estate. Many test engineers prefer a square test point, because it increases that from a diameter of 0.035 on each side, since it is a square. This increases the probable area.

There is a chart in the SMTA/TMAG TP-101E spec that shows test point sizing by order of preference. In the spec, it can be seen that what the testing stakeholders have advised, as far as target size for the top side and the bottom side test point pads diameters of the PCB. Again, size selection will also depend on what space the board offers as well as what size will result in less cost, while accommodating the considerations.

9.3.5.7 Positioning of Test Pads

There is a chart in the SMTA/TMAG TP-101E spec that shows the minimum pad to pad dimensions. This shows that it is very common to go down to 50,000 between test pad positions. Any less, while it can be done, it is going to cost a lot more. Any less of a diameter of test pad, will reduce the accuracy or the probability and a 15,000 separation between the test pads. The SMTA/TMAG TP-101E spec does a good job at outlining these parameters.

9.3.5.8 Designing for Testability

DFT is an automated process, synonymous with automation is design clearance. It is important to provide design clearance when implementing DFT. It's important to keep the distance between the prob and the board edge distance greater than or equal to 0.125. Again, it is best not to design the minimum requirement, just to make sure all will work. This distance between the probe to the board edge ensures that components and associated test points will not cause problems with a vacuum seal, that is involved with seating of the probe. Clearance gives the gasket a greater life. Clearance is considered good for any type of production line design. It is very important that designers connect with assembly stakeholders to find out how much clearance is required and also give a little extra buffer.

9.3.5.9 The point of Testing

Testing is important to find electronic failures, to track electronic failures and to bring together data to justify a justifiable fix. Testing in volume allows us to collect data in volume and to track this data to find out if it's happening in a certain area of the design consistently. If a particular component is consistently "tombstoning", that data can be used to come back to the design or some of the manufacturing processes in order to justify change. This could mean a re-design needs to happen to that PCB assembly. Maybe the footprint needs to be tweaked a little bit to match the manufacturing processes.

PCB assembly testing is often thought of as necessary. Some engineers feel they shouldn't have to pay an EMS provider to check their own work. Most engineers think their board is perfect and that there won't be any problems when it is being built. Some engineers also feel that if there are processes that are failing, why are they the ones that have to pay for a test fixture. However, in a world of product constantly being introduced, incorporating in-circuit tests goes hand and hand with volume production. The tests are done so that the product is accommodating of the customers needs. The tests are also done to gather data to improve the

processes. Tests are done to keep track of the health of the product and to over all identify problems, that is basically why we test.

It is recommended that PCB designers contact an EMS provider who is building and perhaps testing a PCB built by the PCB designer, and to ask for a tour of the facility the EMS provider to learn more about the challenges that go on when building a board someone has sent in to be designed. Taking an opportunity to observe their way their test processes will bring greater understanding as to what really goes on through the EMS providers perspective. It will open a greater understanding on why testing is so important in the world of volume PCB assembly production. As a PCB designer, asking a local manufacturing company stakeholder about the biggest manufacturing challenges.

9.3.6 How to implement DFT into the PCB Layout

9.3.6.1 Setting up Parameters

Once the PCB design is done being designed on the layout is when test points are added. The PCB designer application has the ability to automatically place test points. At the top of the screen select the “manufacturing” tab and scroll down, select “TestPrep” and select “Automatic”. It is much easier to apply test points automatically. The “Testprep Automatic” window will appear. We are going to be creating an in-circuit test fixture. This window will allow the set up of a few rules. Check the box that says “Allow test directly on pad”. Selecting the execute mode to be “incremental”. Incremental mode is best because there won't be a need to re-do anything that has already been done. So, if there is a change in the circuit and more test points need to be added, simply just re-run this incrementally. The displacement between test points will be 50 mils.

9.3.6.2 Advanced Parameters

To the right side of the window in a tab that says “Parameters”. Clicking on the “Parameters” tab to continue setting things up.Under the “Preference” section in the “TestingPrep Parameters” window, selecting a pin type to be “Any point”. “Any Point” means pins or vias. Under the “Text” section in the same “TestingPrep Parameters” window, it is helpful to label the test points. Here, selecting “stingNumeric” and typing in “TP” will be the text label that indicates a testpoint like TP1, TP2,TP3, etc.

Under the “Methodology” section in the “TestingPrep Parameters” window, allows us to select which layer will the testing point be placed. Our preference will be to test from the bottom of the PCB, so select “Bottom” under “Layer” option. For “Test Method” in that same section, select “Single” to add a single test point for each net. That is for in-circuit testing. “Node” or “Flood” could have been chosen but “Flood” is more for a bare board test. So, we are using “Single”. Under the “Restrictions” section in the “TestingPrep Parameters” window, in the “Minimum Pad Size” section will be set to 35 mils. Under the “Allow Under Components” section, under components will be allowed but only components that are on the top side of the board. There can’t be test points placed under componentes that are actually placed on the

bottom side of the board. Lastly, in this section "Assembly" will be selected from the "Component Representation" list options. The "Assembly" outline will decide what that area is.

Under the "Padstack Selections" tab in the "Testprep Parameters window", it's important to specify the actual desired pad stack to be used, we've selected "VIA_TEST". It is the correct size, and will replace current vias that are in the design with the new pad stack. Going back to the "Testpoint Automatic" window, by clicking "Ok" will allow the application to generate test points, which is actually a quick process.

9.3.6.3 Failed Nets

Once the test points have been applied, look in the log file by clicking "Viewlog..." on the "Testpoint Automatic" window. The "View of file testprep" window will appear. Here. one can check if the application was able to test every single net. At the bottom of the window there will be text that says "Total nets failed", the number next to it will show how many nets could not be done by the application. Next, scrolling up on that same window, the reason for each will be explained. Some reasons may be because they were not accessible from the bottom side of the board.

To fix these failed nets, one can go to the top of the application screen and click on the "Tools" tab, and click on "Reports". In this window, scroll and click on "Testprep Report". This will bring up a new window, that shows exactly where every test point was placed. Here, the exact net name will be shown, the number of test points placed. This will also tell you the type. For example, was it placed on a pin or a via, and if it was placed on a pin what was the exact pin? This testprep report tells the location, as in the x, y location of the test point. It also gives the reference designator of the test point. At the very bottom it gives a nice concise list of the nets that it was unable to test. It's best to simply extract those nets that failed and put them in a text file.

On the right side of the page there is a sidebar, with the three tabs, one of the tabs is labeled "Find". To highlight those nets, select the sun icon at the top of the application window. Getting this "Highlight" command, selecting "nets' and importing the list of the nets that were not able to test from the computer word file, under the file "Find by Name" under the "Find" tab. The nets that were unable to be tested, automatically. The nets that aren't tesible most likely don;t have vias on them, making them untestable from the bottom side of the board. These nets can be routed and test points can be added to a different location. This can be done one by one or one can re-run test prep and have the test points generated. Now going to the "View of the file testprep", it can now be seen that less nets have failed than before.

Another edit that can be done is, adding test points to ground and power. It's a good idea to sprinkle test points around the ground and power on the PCB. It's good to have more than one node on those nets tested. Highlighting the area of ground and power, one can then go back into the "Testprep" window but this time using the "Manuel", found under the "Manufacturer" tab at the top of the main application page. The "Manual" page will appear on the right side of the application page and select the tab that is labeled "Options". selecting the "Add" option under

the “Mode” section in the “Options” tab. Then moving the mouse directly on the editor page, to where one would like the test points placed. If a pad is underneath a component, the application won’t allow there to be an addition of a test point at the location. An error noise will be made.

9.3.6.4 Density Check

To test that these test points aren’t causing any problems, such as having too many test points in one area. By selecting “Testprep” under the Manucafer” tab and then selecting “Density check” in the “Testprep” option. In the “Testprep density Check” window, check the box that says “Unit area check”. For each area check 500 mil, type this value in the “Unit area square displacement”. For the “Max testpoint per unit area” type in 5. This is saying that we don’t want more than 5 test points in any 500 mil square area.

Now, running the “Density Check” on the “Testprep Density Check” window. A window that is called “View of file: testprep_density” will appear and here we can see more closely. This report shows where the problems are. Clicking on the coordinates for an area, will make the application editor screen zoom in on that area. Having this ability to see the area closely, we can see there are any connectors at the bottom of the board , which will be the cause of the area being flagged by the application. There may be test points on almost every single pin, in this area, making the area too dense.

9.3.6.5 Fixing the Points

Let’s say that the design is complete and that we are happy with the testing, next a test fixture is to be created. The first thing that should be done is fix the test points because we don't want them to move. The fixture is created by going to the “Manufacturer” tab at the top of the application page, selecting Testprep and then selecting “Create Fixture...”. Once the fixture is created, the test point won’t be moved, because it is very expensive to redo. The “Testprep Create Fixture” window will appear and click on “create fixture” on that window.

Next, we’re going to go to the top of the application screen and select the color wheel button. The “Color Dialog” window will appear, selecting “manufacturing” on the left hand side bar. This allows the selection of what feature of the PCB one would like to see on the editor screen. First clicking on the “off” button at the right side of the screen. This turns off the board outline. Next, selecting the fixture_Bottom box , for the fixture for the bottom will allow only those color/features to appear on the editor screen. The fixture can be seen on the screen, with each test point labeled. If we zoom in we can see all the test points are. Those are the test fixtures.

9.3.6.6 Export NC Drill Data

Finally, we are going to export the NC drill data. Selecting the “Manufacturer” tab at the top of the application page, selecting Testprep and then selecting “Create NC drill data”, to export our NC drill data. This file has now been exported. The file is saved to the computer under the name “bottom_probe.drl” in Notepad. This contains the drill data for the test

points. That is how easy it is. A Lot of time depending on the board being made, there will be more active work. If this needed to be cleaned up because the density check failed, then many test points would need to be moved around and alot of swapping would need to be done.

10 Overview

10.1 Summary SW

To summarize our goals and objectives for the software side of this project, we will be creating a mobile application that can operate on both Android and IOS, which will primarily have the ability to give the user a profile, which can manage all of their inventory items via CRUD operations in the app. These CRUD operations will be handled in both the software and hardware side of the project. Specifically in the software side, this will be mitigated via manual entries. The user can click on the create item tab, which will prompt them with textboxes and labels indicating which field represents the item name, item measure and expiration date. Then the user will have the ability to save the given item to their inventory. If any mistakes were found on any particular item, the edit button will specify the item and give the user to change the item description and save the made changes. For the delete, there will be an “X” next to each item card, which will simply delete all of the item information inside the database, whenever the user clicks on that button.

Regarding how the software development will occur, this project is being done in the Native framework, we have selected this framework to enable the ability of handling multiple operating systems and maintain the same look and feel across the different displays. Although we were not familiar with the environment, we opted to learn and grow our expertise in such areas. One of the main advantages we have encountered was that in Native, we are able to utilize sample projects and test others’ applications, being able to have a better grasp of the insight of these other projects.

Our main goal when developing is to implement a user-friendly interface, so we thought of a variety of ways on how to display the data. We will be displaying each inventory item through “item-cards” which will be similar to a list view, but instead of row by row, we will be customizing each row to contain a block of data chunks, which will be all of the description associated with the item.

The software will be constantly pinging the database to retrieve and update information. There are only two tables being utilized in this project, these are the User table and the Inventory table. The User table consists of every single user that has created an account to manage their data. We will encrypt the password for security purposes so there are no issues when the users decide to log in. The inventory table on the other hand consists of every single item that every user has, including all of its descriptions, including, but not limited to product measure and expiration dates. These two tables are connected via the “UserID” field in the Inventory table. This “UserID” is linked to the ID field in the User table. The ID in the user table is the primary key, being the foreign key for the inventory table, making it distinguishable among users.

We will also be implementing some embedded systems code to integrate the hardware to the database, making it possible for the scanner to connect directly to our database through the wifi-modules. Otherwise there would be no way of identifying how to insert records or even remove them. We shall implement some logistics to have some checks as well so the system can verify if there are any items with the selected barcode, and if there is none, the system will skip this operation and not perform anything into the database.

Initially our objective was to optimize inventory tracking, by simply keeping the user informed about expiration dates. This is no longer our main goal, although it is a feature that is capable within the application. Unfortunately, barcodes do not store expiration dates, though we are still able to capture product information, such as product name, measure, etc. Via API calls. We have selected a very renowned database system that has a huge architecture containing tens of thousands of records and product barcodes associated with all of its details, thus giving us the capability of automating this procedure upon a barcode scan. The way hardware elements will be utilized among this project, such as the scanners and how it will all be integrated (software with hardware) will be further explained in the Hardware Summary section of this project, below.

10.2 Summary HW

For the hardware implementation of this project, we have kept the design very simplistic, since our primary focus is on the users involvement with the application, which will be done via the interface. The components we have are the Printed Circuit Board, which will contain all of the following elements: the scanner, switches, green and red LEDs, wifi-modules, etc. The LEDs have the purpose of indicating to the user which phase it is currently set to, these phases mean if the item is ready to be inserted or if it is ready to be removed. The scanner should be connected at all times to the server as it is linked to the wifi-module, ready to update the items in the database. To modify the current state the Printer Circuit Board is in, all the user needs to do is modify the state of the switch, causing the green LED switch to the red one, indicating to the database which operation will be performed (insert or delete).

One very important bit of information is that the operations will not be completed if the scanner is not linked to the active user. This linkage is done via the software end of the project, the user when registering an account will be asked/required to link a scanner, if this scanner is no longer the active one being used, the user needs to go back to the mobile application and re-link this bit of information to the active one. The scanner is on a one to one ratio with the users since we need to identify the active scanner whenever a scanner is performing an insert or remove from the DB so we can acknowledge the active user through the scanner that was utilized.

The mechanism for scanning will be very simple so there is no room for errors from the users. The way the operations shall work is through the wifi modules which will be able to sync up with the server, register the identified product from the database and delete if the item exists, otherwise skip operation. If the item is being scanned in, it will simply insert the record with its according product details. There shall be some embedded systems code to perform some of these operations, but the linkage to the server shall be done via the wifi modules so the integration of software and hardware can occur.

The way the system will operate is via the printed circuit board, which will have an electric plug tied to it, allowing the user to simply insert if they are ready to put the tracking system into use, otherwise they can utilize the mobile application to perform any of the operations.

To make the tracking system very user-friendly, we opted in giving the user to select their preferred method of utilization. The system works both in a hardware environment as it does in a software on. All operations can effectively be done via each one individually, and as a whole the system comes together and will fluidly operate across both by having constant refreshes to the database. From the software end, to view the latest items, the user simply needs to click on the refresh button and the updated items shall appear.

10.3 Significance of our project

The main goal of this project is to help people manage their groceries by use of technology. One of the central ideas we had before deciding on any of the projects was to provide something to the community by using the concepts of engineering and computer science. This project will be very helpful for people who like to manage their food and groceries and be prepared ahead of time. People have a very busy life and it is very hard for them to keep track of things like groceries. If looking at a general idea, it is very easy to keep a list of groceries and food items but people generally don't find time to do this. Our goal is to help people who struggle with time and help them using the technology that we have today. This technology is very much active in big companies and stores like Walmart, Publix, Target etc. What we plan to do is to provide the same technology to people in their houses. There are many companies who are trying to provide the same technologies in their fridges and those fridges come at a high cost. What we are trying to do is to create a generic product that can be used in any fridge or any inventories in the house. Our project will also help people to reduce their waste of food as people unintentionally do not realize that the items that they bought a few months ago are about to expire.

As a learning curve, this project will give us an opportunity to work towards the topics that we have learned during the journey to become an engineer. There are many challenges that we faced during the initial phase of project planning because of the lack of knowledge. However after thorough research, we were able to come up with a solution on how to overcome those hurdles. For this app, we need to have an app and also hardware. We have spent a lot of time on deciding the parts and designing the board, a PCB, a scanner, and we will do the same building the entire model together. So far, we have learned about many things on our own and it gives a practical experience which is very important in the field of engineering.

We also want to focus on the future improvements that can be done towards this project. Due to the limited time frame, we will be able to integrate a set of functionalities but this project can be extended in many ways where it can be helpful to a lot of people. We want to integrate functionalities like recipes, methods to save power consumption, price feature etc. Some of these functionalities are easy where some of them can be hard. The idea behind saving the power consumption was to audit the items in the inventory and adjust the power in the fridge

automatically. These kinds of features will be very helpful in the coming future since we are moving towards a world that focuses on energy saving and use of renewable sources of energy. The price feature's idea was to add a field to the inventory that shows the cost of the item that is added and at the end, it will show a user the total cost of the current inventory. This would help users to manage their grocery bills. The recipe feature's idea was to show some of the recipes to the users based on the items that they have in their inventory. These ideas will be a great add on to the project in the future.

10.4 Future Improvements

As the automated inventory system was created with the intent of tracking the users items with a given timeframe of a few months, we will not be able to complete the designated ideas within the time allocated. We decided as a team to progress with these ideas and implement additional features that would carry over after the initial demonstrated version. Although we do not intend to make this commercialized product, we do want to expand software capabilities and add this to either the google-play store or the app store. As we will not commercialize the hardware element of this system, the mobile app functionality of scanning will become partially obsolete, instead we intend to converge our thoughts on the platform so it can become a digital interface like facebook, tic-tok and instagram. Instead of sharing pictures, news and ability to send messages, the platform would only be a network of recipe sharing. Where users can add friends, all friends will have a feed of recipes and other users can add comments, view/add videos of tutorials, etc. This idea will primarily be to mainstream a recipe sharing platform, where the inventory (manually inserted by the user) will be linked, so whenever looking at the recipe, there will be a "smart" feature, which will handle the missing elements and arrange these into a list for the user so they can add pending ingredients to a cart. Eventually this cart will no longer be simply a list the user can reference for a shopping run. We seek to engage with companies like Publix, Walmart, so that the product will be obtained online directly from the app, instead of having to be present in the store to buy the items or having a separate device and going over each item in the list to add to the cart and buy it online somewhere. This will automate processes, simplifying the lives of many users that have to keep track of their inventory manually.

The reason why we opted in removing the scanner for this system after current implementation is because if this were to be marketed, the user would need to pay a high amount just for the hardware components. The cost would potentially downsize the amount of people interested in the product. Also, by not having the hardware, we would be able to attain many users by simply focusing on developing the product on a software standpoint. There would be almost no cost, probably only a $1-$2 dollar fee for acquiring the app from the store, initially we are considering keeping the application at no cost and make this platform increase usage, so the users can understand how the application works, be familiar with the environment and get used to the system. This platform will only be effective if and only if we attain a wide range of users. Our target public ranges from young adults to retired people, that have an interest in keeping track of their inventory either from fridges or cabinets, want to automate procedures and mainly engage with other members to learn new recipes, as well as share their very own.

The key feature we would like to implement after this initial setup would be the user interaction across this platform. This includes adding friends, having a "view feed" tab where users can visualize everyone else's shared recipes. Although we seek to gain the intention of young adults, we are well aware some of this age group is not focused on cooking and learning new recipes. Despite this fact, users may decide to utilize our application due to other features such as the ability to keep track of all of the users product, including expiration dates, so that there is no waste in food/product within the household. This is a huge issue in the US today, we have exponentially grown the loads of disposed items, and this application can be one of the potential solutions to this problem.

Altering this current project into a singular platform, in a similar fashion to some sort of social media is one of the potential routes we take, though we did discuss some other options. As a team, we need to discuss the potential routes carefully and see who is still interested in being a part of this project. We have a few members that have high expectations on this project as a whole, while others might get carried away with other priorities, such as full-time jobs, etc. Other potential routes we had discussed includes augmenting the idea of barcodes. Our original intent as a group was to be able to automatically track barcodes and retrieve expiration dates from them. Although some barcodes have expiration dates embedded to them, some do not. We wanted to study the idea of standardizing a new type of barcode and start tracking expiration dates off of them, we have discovered most products have a separate code attached to them, this code includes the date created and how many weeks off until the product is not at its prime conditions. We could attempt to modify existing mechanisms and begin a new system for barcodes, which would make our mobile application much more effective, since the user will not have to manually insert expiration dates, these will all be automated whenever the user scans the product. This would be a unique system, no product or project has enabled automated expiration date queries for their products, this would be the first and with the idea we have for this project, we could capture the attention of many users that are trying to be cost-effective within their household.

As a team we were capable of exploring different options and combining ideas to come up with strategic goals for the long-term of this project. There are a variety of routes we can choose from, but the most intriguing ones that have a huge amount of potential in our opinions is utilizing our project and being able to transform it into a platform where users can interact with each other and share recipes or coming up with a centralized mechanism to be used in different products so that there's a universal barcode containing expiration dates, making our system have the automated functionality of retrieving product expirations. We have yet to decide our approach to these upcoming challenges, but for now our mindset is clear on achieving this first goal of having the hardware and software communicate to each other via wi-fi modules and updating information in real-time.

As we discussed removing the hardware elements of the project to make this product marketable and cost-efficient, we have also thought about minimizing the cost for the hardware by simply implementing some logic on our software end so there is a barcode reader on the application itself. This is another route we may explore as we come to a stopping point on this active project implementation. For the purpose of this project we were aware this could have been an optimal route, but we needed to have hardware elements to show our capability of not

only doing software development, but also merging the software with the hardware components that we have integrated.

We have many resources to continue this project and keep implementing features for this system, one confirmed feature that will be worked for this automated tracking system is the ability to share recipes to other users. We have come together as a team and believe this would be a major addition and we thought about having it done by the end of the summer. Although this would be extremely useful to have, we are putting it on hold for after summer since we want to test our current working version prior to committing to any further software development on our application.

10.5 Alternatives

For alternative technologies, we had discussed the possibility of simply having a software based project and containing a scanner implementation via the application, where the phone would do the job. Although we believe we could have been successful by doing such implementation, we opted with a different approach since we wanted to work on the electrical design on this project. For future implementations and features, we might provide the user the capability of scanning via their mobile devices, since the cost would be extremely high if needed to purchase all of the hardware components and had this product as a marketable item. Instead, we believe it could be beneficial to the user simply having the application, where the cost would be pretty much none.

For the purpose of this project, we had looked into previous products that are either in the market or had been sometime in the past. We had found some projects implemented by Samsung and found it very interesting how they had the touch feature of adding items into the fridge simply via selection. The issue is that a scanner would be much more time-efficient, although we did highly consider utilizing a touch-screen and embed this utility into a fridge so that the user can see it directly near the fridge. The main issue with taking this approach was the high expense of having another display other than the one from the user (their own mobile device).

Lastly, we had considered implementing a built-in sensor inside the fridge to automatically scan products as they are inserted/removed. The issue with this approach was having sensors resistant to low temperatures. Even though it is possible, it would be extremely costly and would be terrible to have to discover an appropriate angle that can reach all ends of the refrigerator and attain/scan the barcode within that quick time-frame of simply adding an item to the fridge. These possibilities made this approach much more complex than simply having a scanner linked to the printed circuit board and embedding the devices so that the user can simply utilize it prior to inserting into the fridge.

We knew this project would be possible since there is something very similar we use on a daily basis, which is the self-checkout in walmart or other big markets. Instead of simply adding and checking the item out for purchase, we will be doing database transactions to keep the inventory up to date depending on what the user chooses to do, either add or delete items from their own inventory. Below we can see the self-checkout system, which we really geared towards

having this user-friendliness, giving users from all ranges the quick sense of intuition on how to utilize it.

Figure Z – Hardware System Display

Figure Z - Hardware System Display

10.6 Research Methodology

For our project we were very thorough in researching every facet of our project. Most components to this project are unfamiliar for us, we wanted to be vigilant in understanding how to implement each component of the project. We split the team to research based on software or hardware. However, since the majority of the team has a background in software we wanted to make sure we put in the effort to understand the hardware we need to use. Since our project uses many physical components including a scanner, Wi-Fi module, and development board, we made sure to focus on each component specifically one at a time. On the software side we spent time researching the different stacks we could use, along with the different languages available to use. This includes research into mobile application development.

11. User Manuals

11.1 SW:

Here is the user manual that we will use on the help button in the inventory:

Welcome to the Automated Inventory Tracker software. This software is developed by UCF undergraduate students to provide users ability to manage their groceries and inventory items in a better way. This user manual contains all the instructions on how to access the application and how to perform different operations.

Please use the register page to create your account. You will be asked to enter your Name, Email, Phone, and required to set a password. Please click on the register button once you have entered all the information. Once the registration is complete, check your email for the verification link that was sent to you. Click on the verification link to complete the registration process. Please note the verification link expires in 24 hours and if a user fails to verify in 24 hours, the account will be automatically deleted. login to your account, please use the login button. Enter your email and password used at the time of registration to login successfully. If

the user is not able to login with the given email and password, please use the forgot password link. Here please enter your email and phone number and click on forgot password. We will send the password reset link to your email. Please click on that link to reset the password. Once the password is reset, you can successfully login into your account.

In the inventory page, you can use the create item button to create an item. Please enter name of item, expiration date, and measure if the item and click on create item button. You can now see that the item that you created is now visible in the inventory. The inventory page is where you will see the list of all the items. When you have a big list, you can scroll down to see all the items. In the list click on the sorting button to sort the items by your choice. You can sort the items by date added, date of expiration, or by name.

To edit an item, simply click on the item card. Now you will be redirected on the item page. Here you can see all the information about an item. Please click on any field to edit that field. Once all the changes are completed. Please click on the save button to confirm your changes. You will be redirected to the inventory view. You can now see that the information on that item is now updated To delete an item, simply click on the delete button on the item card in the inventory view. The application will ask you to confirm the deletion of the given item. If you confirm, the item will be deleted from the inventory. You can also choose to cancel to make no changes.

Please use the settings page to change the settings of the application. Here you can change multiple things. Like changing your scanner ID, login into a different account after pressing the logout button, or change the notification system. The notification system can be set up so that it will send alerts when an item is about to expire. This can be set up as 1 day, 1 week, 1 month, 3 months or all. Setting all will send you notification when an item is about to expire in all of the given time frames.

11.2 User Manual HW

1. Components to be aware of:
    a. Electric plug
    b. Printed Circuit Board
    c. Wifi Modules
    d. Switches
    e. Green LED
    f. Red LED
    g. Scanner

2. The User will carefully insert the Electric Plug into an available port in their household.
3. The WifiModules shall be activated, as well as the other 2 PCB elements (Scanner and Switch)
4. If the LED is green, means the user is available to INSERT an item into the inventory via scan

5. If the LED is red, means the user is available to REMOVE an item from the inventory via scan
6. To switch back and forth across Insert and Remove, simply change the switch state from the PCB.
7. The Wifi Modules should be already in progress so the scans can insert and remove elements in real-time.
8. To verify the elements are adding and removing correctly, open the mobile application, log in and view active items in inventory.
9. If the application is already open, simply refresh the page via the refresh button.
10. Once completing all of these steps, check the Software User Manual for more information on how to explore the mobile application. Enjoy!

***Important Note:
→ The Scan will only be registered and shown to your profile if and only if the scanner is LINKED to your account. Otherwise it will not perform the add/remove on your items within the current inventory. To change the linked scanner, view the Software User Manual.

Appendix A – Copyright Permissions

Your name*

Sonu

Your email*

sonuthummar45@gmail.com

Subject*

Need Permission for Using a Picture

Your Message

To whom it may concern, Good afternoon, we are a group of 4 students from the university of central florida, researching elements for our senior design capstone project. Can we be granted permission to access your content and add some references to our capstone documentation? Thank you, have a great rest of your day.

belkin

Name * Sonu

Thummar

PHONE NUMBER * 3214409190

EMAIL ADDRESS * sonuthummar45@gmail.com

Product Name or Model Number Belkin WLess G

How do I find my model number and serial number?

Brief Description of the problem* To whom it may concern, Good afternoon, we are a group of 4 students from the university of central florida, researching elements for our senior design capstone project. Can we be granted permission to access your content and add some references to our capstone documentation? Thank you, have a great rest of your day.

sonu thummar <sonuthummar45@gmail.com>
To: contact@tera-digital.com

Hello,
To whom it may concern,

Good afternoon, we are a group of 4 students from the university of central florida, researching elements for our senior design capstone project. permission to access your content and add some references to our capstone documentation? Thank you, have a great rest of your day.

Appendix B – References

1.2. What Is Debian?, help.ubuntu.com/lts/installation-guide/s390x/ch01s02.html.

About Harris AndreaHarris Andrea is an Engineer with more than two decades of professional experience in the fields of TCP/IP Networks, et al. “8 Different Types of Servers in Computer Networks.” Networks Training, 17 Oct. 2020, www.networkstraining.com/different-types-of-servers/#2_Database_Server.

Accelerated Shape Detection in Images, 18 Aug. 2020, wicg.github.io/shape-detection-api/.

Angular. (n.d.). https://angular.io/.

Automation, Tempo. “Design for Testability (DFT): Is It Really Necessary?” Tempo, Tempo, 5 Oct. 2018, www.tempoautomation.com/blog/design-for-testability-dft-is-it-really-necessary/.

Automation, Tempo. “The 3 Essentials of PCB Design Testing.” Tempo, Tempo, 11 July 2018, www.tempoautomation.com/blog/the-3-essentials-of-pcb-design-testing/.

Barcode Spider. UPC Code Lookup, EAN & ASIN | Barcode Spider. (n.d.). https://www.barcodespider.com/.

by: Sean Boyce, et al. “Track Everything, Everywhere With An IoT Barcode Scanner.” Hackaday, 5 Apr. 2021, hackaday.com/2018/08/15/track-everything-everywhere-with-an-iot-barcode-scanner/.

Contributor. (2020, March 2). 8 Advantages of Using MySQL. DevOps.com. https://devops.com/8-advantages-using-mysql/#:~:text=MySQL%20tops%20the%20list%20of,solution%20for%20full%20data%20integrity

Department, VSE | Test, et al. “PCB Design for Testability Guidelines Engineers Should Know.” VSE, 27 Aug. 2019, www.vse.com/blog/2019/08/27/pcb-design-for-testability-guidelines-engineers-should-know/.

Dos Santos, Wagner Gouvea. “Natural History of COVID-19 and Current Knowledge on Treatment Therapeutic Options.” Biomedicine & Pharmacotherapy = Biomedecine & Pharmacotherapie, The Author. Published by Elsevier Masson SAS., Sept. 2020, www.ncbi.nlm.nih.gov/pmc/articles/PMC7332915/.

Empower JavaScriptwith native APIs. NativeScript. (n.d.). https://nativescript.org/.

Hildenbrand, J. (2019, November 13). Bluetooth 5: Is it actually better, and do you need it? Android Central. https://www.androidcentral.com/bluetooth-5-it-actually-better-and-do-you-need-it.

Louw, Leanie. “6 Steps to Build a House of Quality – Six Sigma Approach.” Master of Project Academy Blog, 9 Dec. 2020, blog.masterofproject.com/house-of-quality/.

Pollette, Curt Franklin & Chris. “How Bluetooth Works.” HowStuffWorks, HowStuffWorks, 11 Nov. 2019, electronics.howstuffworks.com/bluetooth.htm.

PYMNTS.com. “Self-Checkout Hits A (Small) Speed Bump.” PYMNTS.com, 15 Feb. 2020, www.pymnts.com/unattended-retail/2020/self-checkout-hits-a-speed-bump/.

SMS Tutorial: Using SMS Service Providers (SMS Gateway Providers, SMS Resellers, SMS Brokers), www.developershome.com/sms/howToSendSMSFromPC3.asp.

SMTA/TMAG Task Forces. SMTA/TMAG Testability Guidelines TP-101E. SMTA, 2014.

Start Designing For Free: DbDesigner.net Guest Login. Start Designing For Free | DbDesigner.net Guest Login. (n.d.). https://app.dbdesigner.net/designer/schema/guest_template.

Stazzone, Shelly. “Complete Guide to Barcode Types & Standards.” Camcode, 13 Apr. 2021, www.camcode.com/asset-tags/guide-to-barcode-types-standards/.

Technology, Fuchuangke. “All IPC Standards in Electronic Industry.” PCB, 27 Dec. 2019, pcbboardassembly.com/ipc-standards-in-electronic-industry/.

upcitemdb. UPCitemdb. (n.d.). https://www.upcitemdb.com/.

“Barcode API Overview | Mobile Vision | Google Developers.” Google, Google, developers.google.com/vision/android/barcodes-overview.

“Bittele's Essential Design for Testing Tips for PCB Assembly.” Porto, www.7pcb.com/blog/essential-dft-tips-for-pcb-assembly.php.

“Chrome Platform Status.” Barcode Detection API - Chrome Platform Status, www.chromestatus.com/feature/4757990523535360.

“Circuit Board Design for In-Circuit Testing.” Altium, resources.altium.com/p/circuit-board-design-circuit-testability.

“Cloud Server.” VMware, www.vmware.com/topics/glossary/content/cloud-server.

“Coronavirus Updates: Page 16 of 19.” Coronavirus, 27 Mar. 2020, www.ucf.edu/coronavirus/updates/page/16/.

“Database Server.” Database Server - an Overview | ScienceDirect Topics, www.sciencedirect.com/topics/computer-science/database-server#:~:text=Database%20servers%20are%20used%20to,the%20data%20across%20the%20network.

“Designing For Testability (DFT).” Altium, resources.altium.com/p/designing-for-testability-dft#:~:text=SUMMARY,an%20acceptable%20rate%20of%20defects.

"DFT DFM DFA Design Considerations in PCB Design." TronicsZone, 17 May 2020, www.tronicszone.com/blog/dft-dfm-dfa-pcb-design/#:~:text=DFT%20is%20the%20process%20of,physical%20manufacturing%20process%20is%20over.

"How GS1 Standards Work - Standards." GS1, www.gs1.org/standards/how-gs1-standards-work.

"How to Create a Barcode Database on CodeREADr." CodeREADr, 22 Apr. 2021, www.codereadr.com/knowledgebase/creating-a-database/.

"Power Supply Safety Standards, Agencies, and Marks." CUI Inc, 20 July 2020, www.cui.com/catalog/resource/power-supply-safety-standards-agencies-and-marks.

"Push Notifications Overview." Web.dev, web.dev/push-notifications-overview/.

"Reasons to Choose Debian." Debian, www.debian.org/intro/why_debian.

"Samsung Smart Refrigerator: Family Hub Touchscreen Fridge: Samsung US." Samsung Electronics America, www.samsung.com/us/explore/family-hub-refrigerator/overview/.

"Self-Checkout Hardware." StrongPoint, 23 Apr. 2019, www.strongpoint.com/products/self-checkout-hardware/.

"Server." Paessler, www.paessler.com/it-explained/server.

"Software." Software - Tinker Board Wiki, tinkerboarding.co.uk/wiki/index.php/Software#TinkerOS.

"The History and Basics of IPC Standards: The Official Standards for PCBs - News." All About Circuits, www.allaboutcircuits.com/news/ipc-standards-the-official-standards-for-pcbs/.

"The Shape Detection API: a Picture Is Worth a Thousand Words, Faces, and Barcodes." Web.dev, web.dev/shape-detection/#barcodedetector.

"Tinker Board." ASUS USA, www.asus.com/us/Motherboards-Components/Single-Board-Computer/All-series/Tinker-Board/.

"Types of APIs and How to Determine Which to Build." MuleSoft, www.mulesoft.com/resources/api/types-of-apis.

"What Are APIs and How Do APIs Work?" MuleSoft Blog, 15 Jan. 2021, blogs.mulesoft.com/learn-apis/api-led-connectivity/what-are-apis-how-do-apis-work/#:~:text=API%20stands%20for%20Application%20Programming,the%20response%20back%20to%20you.