

Senior Project Design: A Smart Pantry System

Dominik Sobota, DeVry University, Addison

Dominik Sobota is a student at DeVry University, Addison, IL, pursuing his BSEET degree. While attending DeVry University, Addison and Purdue University, West Lafayette, Dominik has completed a number of technical projects that involved power supply design; control systems design using HC9S12C32, TI TIVA, and Raspberry Pi 3. He is proficient in a number of software languages and application programs: Assembly Language, VHDL, C; C++, JAVA, PYTHON, HTML, LINUX and SQL, Multisim, OrCad/PSice, Energia, Quartus II, Eclipse, Visual Studio, Android Studio; CodeWarrior and PyChem. For his senior design project, Dominik worked in a team to design a SMART Pantry which utilizes a Raspberry Pi 3 and python application to keep a real-time track of the inventory. Dominik has gained valuable experience, in form of product development and testing, while doing his Internship at Contemporary Controls.

Mr. Spencer William Karlovits, DeVry University

Spencer Karlovits is a graduate of DeVry University's Electronic Engineering Technology program. He currently works as a technician wiring and manufacturing automated industrial control systems for roadside enclosures, including lighting controllers, traffic monitoring, and roadside surveillance. Spencer has taken a number of classes from Michigan Technological University where he studied Electrical Engineering Technology and Audio Production and Technology. Courses included programmable logic controllers, fundamental circuit's courses, digital circuits, special electronic devices, data communications, transducer theory, motor control and power systems, and an industrial robotics course. Some achievements made while at Michigan Tech include speaker design with passive crossover filter implementation, frequency and amplitude modulation transmitter and receiver design and implementation, programming field programmable gate arrays in VHDL language, motor control logic structures, and a certification through FANUC in handling tool and operation programming. Spencer transferred to DeVry University to complete his degree in Electronics Engineering Technology.

While at DeVry, Spencer expanded his programming skills by learning C++, Java, C (for specific TI microcontroller programming), assembly code (for TI MSP 432), and Python. He spent considerable time working on programming a variety of microcontrollers including Arduino, Raspberry Pi, TI MSP 432, and TI TM4C123G. With this knowledge, Spencer designed an automated microcontroller controlled car with proximity – object detection as well as auxiliary functions, such as automatic lighting control and temperature control. Additionally, this knowledge fueled the design behind his senior design project which utilizes a Raspberry Pi and high torque stepper motor to implement an automated pantry system with live-status inventory tracking. Other achievements at DeVry include designing and implementing a simple calculator with and LCD screen, momentary pushbuttons, and the MSP 432, designing and implementing motor control structures with ladder logic, and a significant amount of work with various transducers, the associated analog to digital conversion, and signal processing for a variety of automated functions. Spencer intends to continue pursuing a field within the scope of industrial automated control systems.

Dr. Ahmed S. Khan, DeVry University, Addison

Dr. Ahmed S. Khan is a Senior Professor in the College of Engineering and Information Sciences at DeVry University, Addison, Illinois. Dr. Khan has more than thirty-two years of experience in research, instruction, curricula design and development, program evaluation and accreditation, management and supervision.

Dr. Khan received an MSEE from Michigan Technological University, an MBA from Keller Graduate School of Management, and his Ph.D. from Colorado State University. His research interests are in the areas of Nanotechnology, Fiber Optic Communications, Faculty Development, and Social and Ethical Implications of Technology. He is the author of many educational papers and presentations. He has authored/coauthored the following books:

• Nanotechnology: Ethical and Social Implications (2012) • Technology and Society: Issues for the 21st Century and Beyond 3E, (2008) • The Telecommunications Fact Book and Illustrated Dictionary 2E (2006) • Fiber Optic Communication: An Applied Approach, Prentice Hall, N.J. (2002) • Technology and Society: A Bridge to the 21st Century (2002) • Technology and Society: Crossroads to the 21st Century (1996) • Technology and Society: A Spectrum of Issues for the 21st Century (1994) • The Telecommunications Fact Book and Illustrated Dictionary (1992)

Dr. Khan is a senior member of the Institute of Electrical and Electronics Engineering (IEEE), and a member of American Society of Engineering Education (ASEE), and has been listed in Who's Who among America's Teachers. Dr. Khan also serves as a program evaluator for the Accreditation Board for Engineering and Technology (ABET).

Senior Project Design: A Smart Pantry System

Abstract

This paper describes a senior design project in the scope of technical design and implementation of a concept incorporating an automated inventory selection system based on scanned items through the use of a user interface. The system, known as Smart Pantry, is designed to help average consumers organize purchased pantry items both by physical location and itemized inventory as well as purchase routine groceries more efficiently. Through incorporating elements of software integration in the user interface and a motor control structure, the development of this project encompasses a wide range of the team members' acquired skills. The unit allows a user to scan an item and add it to a specific physical location which is then stored in a database on the microcontroller. When retrieving said item, the user simply makes a selection on the generated list on the touch screen display causing the motor to drive the carousel to the location in which the item is stored. The data is stored in an SQL database which is also available on a developed smartphone application to facilitate the organization factor for the end user. The automated portion expands upon techniques for motor control systems. In this project, a driven stepper motor is used while providing live feedback to the microcontroller which ultimately optimizes the precision of the rotation resulting in accurate alignment for item retrieval. The project was undertaken by three students who are pursuing BSEET degree at DeVry University, Addison, IL. The concepts involved in software application development as well as database management were widely expanded upon within the development of this project. The specific focus of this paper is to describe the overall methodology in this product's development from the perspective of technical design and prototype development. The paper also describes in detail the structured approach in combining hardware with software processes, constructing relevant solutions to the technical problems encountered, and the development, testing and integration of a functional prototype.

I. Overview

Many aspects of day to day tasks have been greatly influenced by technological progression. This is especially true in the case of automation as well as data storage and access. With the increasing availability of microcontrollers at reasonable costs, it becomes intuitive to apply such a device to facilitating day to day tasks. The Adafruit Raspberry Pi in specific is one such microcontroller that allows for simpler display interfacing, camera interfacing, motor control, a large array of general-purpose inputs and outputs, and even a wireless internet module. Another microcontroller used in the project was Texas Instrument's TM4C123GXL. The

TM4C123GXL has few possibilities with its smaller processor, but provides the necessary computation power while minimizing power consumption. Applying these concepts to produce a product such as an automated pantry system with active inventory would provide another step towards completing daily tasks with much greater efficiency.

II. Problem Identification

In the scope of home automation, little progress has been made towards improvements in organizational methods for consumer purchased goods. With a focus on perishable food goods, a lack of efficient organization can generate a number of issues for typical consumers. The first issue is the amount of time a consumer spends grocery shopping. This includes both the actual time spent in the store as well as potential return trips to purchase forgotten items. The average time spent in a grocery store for a typical consumer is 41 minutes¹. Additionally, the average shopper will go to the grocery store between 1 and 1.5 times per week yielding on average between 35 and 53 hours shopping per year. While standard grocery lists assist with shopping efficiency and organization, items can still be forgotten or items that may have been previously purchased can be purchased again. Such duplicate purchases can cause unused goods to expire and be thrown out contributing to unnecessary food waste. Additionally, the inability to find where a purchased good has been stored within a household can lead to food expiration also leading to unnecessary food waste. A typical American household on average wastes \$640 worth of food each year, much of which is a result of food expiration². Figure 1 shows the results of a national survey of U.S. residents about the household food waste awareness and attitudes.³

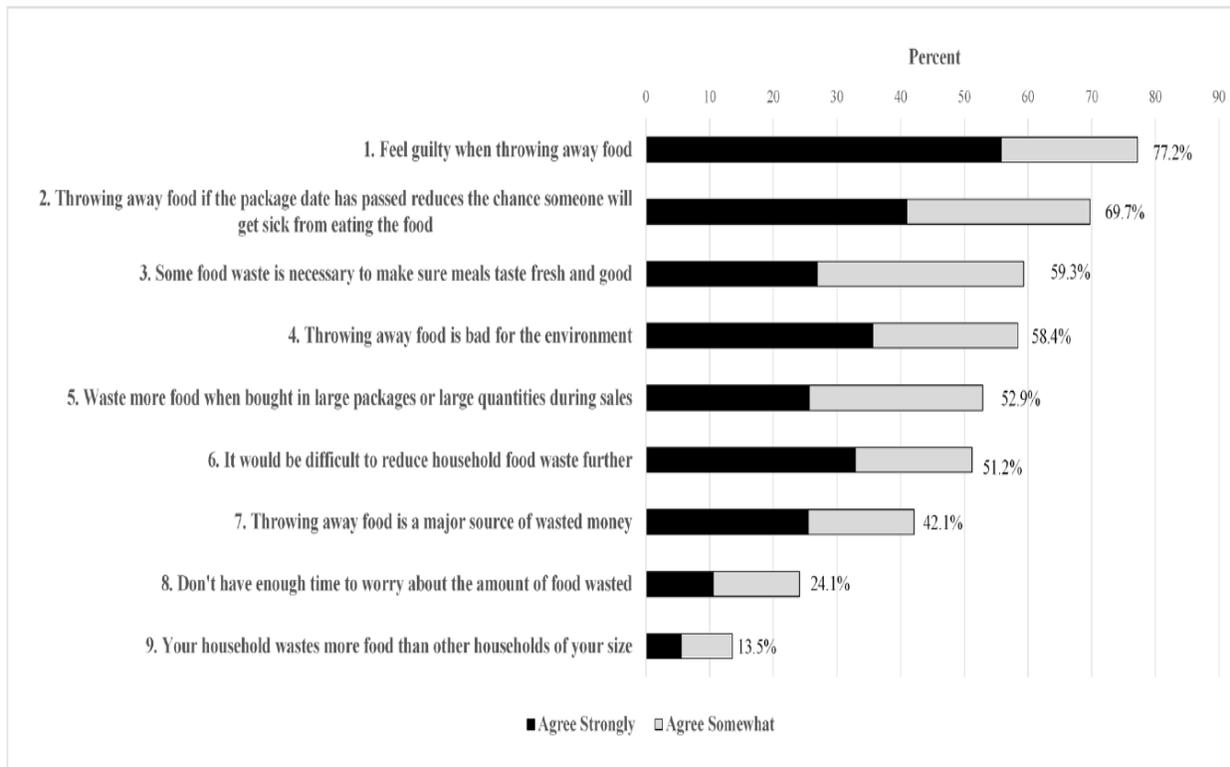


Figure 1. A food waste survey of 500 U.S. consumers provided by WRAP in 2014.³

III. Current Market Technologies

Currently, there are a number of products available in the market, which serve to keep track of inventory or serve to automate item retrieval, but not both. The *Samsung Family Hub* smart fridge is a refrigerator (priced \$1,600 to \$6,000) that keeps track of your fridge inventory through pictures⁴. Pictures are stored and made available through a smartphone application. Additionally, *Storage Motion*⁵ has developed a pantry system called *AutoPantry* that implements motor driven shelves for easy access to all stored food. This partially autonomous pantry does not keep any record of inventory nor is there any capability to store item location. Table 1 displays a comparison of the above mentioned products, and *Smart Pantry* the proposed concept that was developed in this project.

Table 1: Product comparison for food storage systems.

	Samsung	Stage Motion AutoPantry	Proposed Concept
Storage Type	Refrigerator	Dry Storage	Dry Storage
Customizable			
Itemized Inventory	(Takes picture of Fridge)		
Scanner for inventory			
Automated Item Retrieval			
Active Access to inventory (Inventory in Application)/Real Time update			
Price:	\$1,600 - \$6,000	\$1,992 - \$2,995	<\$500

IV. Project Objective

The main goal of this project was to develop an active inventory tracking pantry system that cuts down on food waste and allows for better organization of food. This concept implements a user interface, a scanner, and a controlled motor to account for: (a) inventory, and (b) to automate the food item retrieval process.

V. Project Solution

The concept behind the Smart Pantry system is to allow a user to scan-in purchased groceries through a data system. This allows a user to develop an active inventory of what food is available in the pantry. The data stored includes the product type, product name, the product's expiration date, and the location within the pantry that the item is stored. This information is made accessible through a touchscreen interface on the unit itself as well as through a smartphone application that is linked through a wireless connection made available through the Adafruit Raspberry Pi microcontroller. The item's storage location is saved and called upon for retrieving the desired item. Essentially, this system allows the user to always have an updated inventory of the food stored in the unit, where the food is, and upon item selection can automatically retrieve the product stored in the pantry. The ultimate objective in automating and storing perishable good information is to decrease shopping time for consumers, reduce food preparation time for consumers, and to make efforts towards reducing the amount of food waste produced within a household.

VI. System Block Diagram

Figure 2 displays a top-down view as well as a side view of the physical layout of the mechanical aspect of the system. The concept implements a motor driven “Lazy Susan” rotating pantry unit. Figure 3 displays the initial power distribution for the system. The power source is a wall plug (120 VAC). The incoming line feed will use a fuse to limit the current and surge protection to suppress transient spikes that could potentially damage equipment. The voltage is stepped down, rectified, and filtered through the implementation of a Delta AA15S0500 power supply. This generates 5 VDC to supply power for the Raspberry Pi microcontroller and motor control. Figure 4 shows signal flow for data processing and storage through the Raspberry Pi microcontroller. The inputs for the controller include a USB barcode scanner and a touch screen interface for user control and data access. This is where the data entry for the food information is initialized. The said data includes food name and type, physical location, and expiration date. This information is then stored in memory and referenced through the stepper motor control driver on the output of the Raspberry Pi. Figure 5 displays the technical details entail in the implementation of the various controllers with the integration of the stepper driving circuit. A 555 timer with a 1 kHz clock pulse output is used to generate a signal for the stepper motor driver which controls the stepper motor rotation. Additionally in the data block, the data is made available through a Wireless Internet connection that can be accessed through a developed smart phone application. Finally, Figure 6 represents the elements that require the Wireless Internet connection. The inventory data stored in memory will be made accessible through a smartphone application that links to this wireless connection. Most importantly, the Wireless Internet connection serves as a means for the information scanned through the barcode scanner to access a database to properly identify the scanned item.

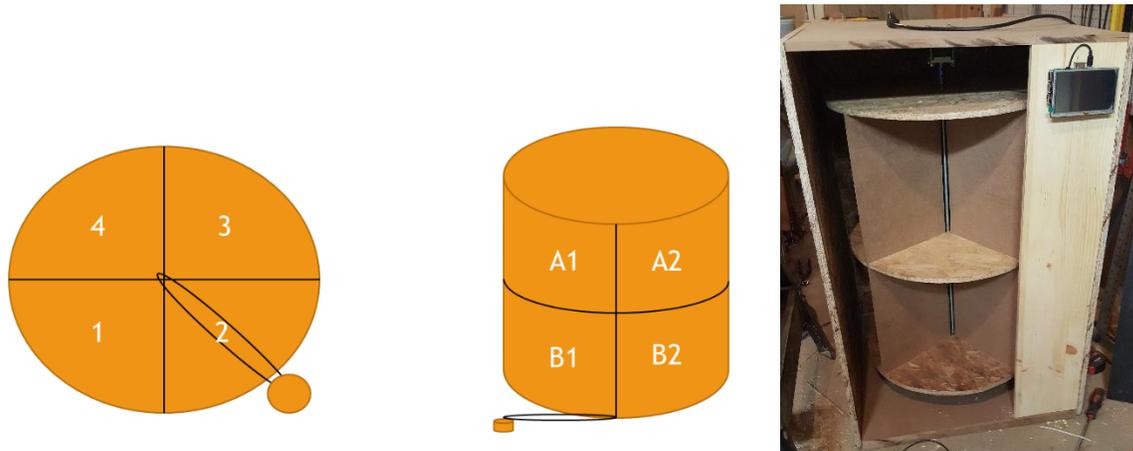


Figure 2: Top-Down and Side View of Physical Concept for Motor Driven Rotational Unit ⁶.

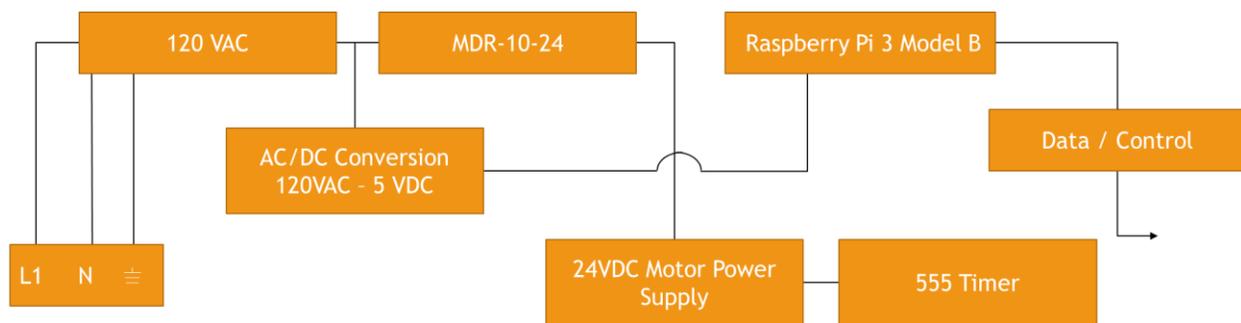


Figure 3: Power Distribution Block.

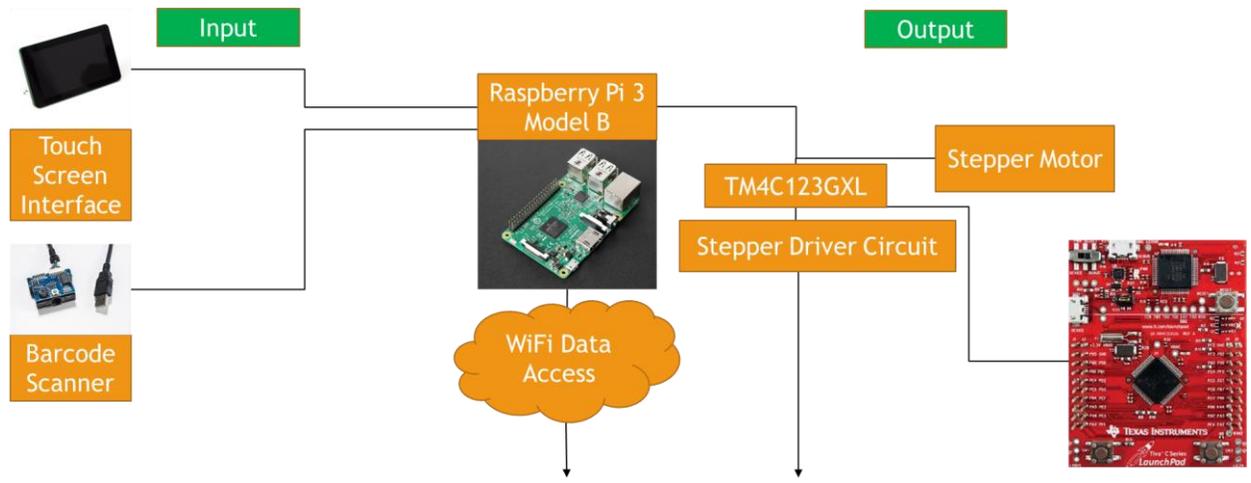


Figure 4: Data Control Block.

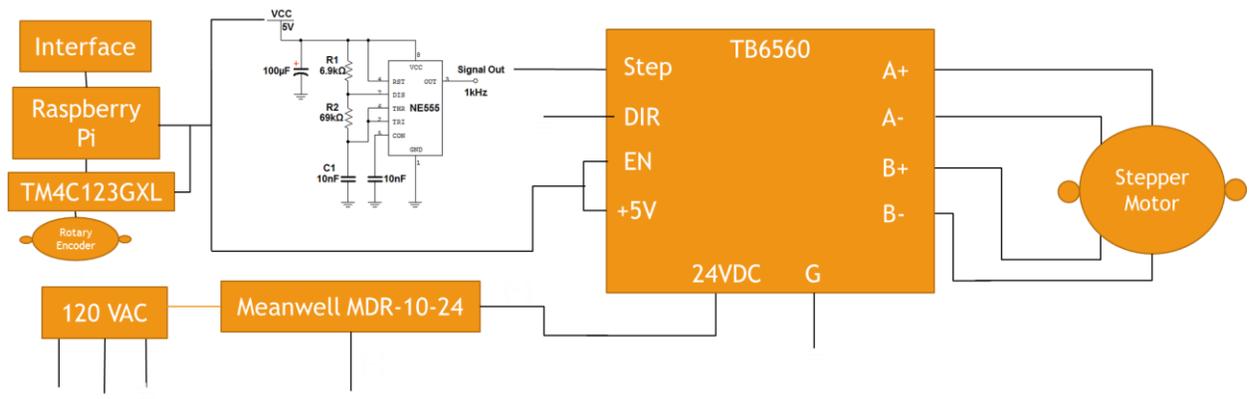


Figure 5: Stepper Motor Driver Circuit ⁷.

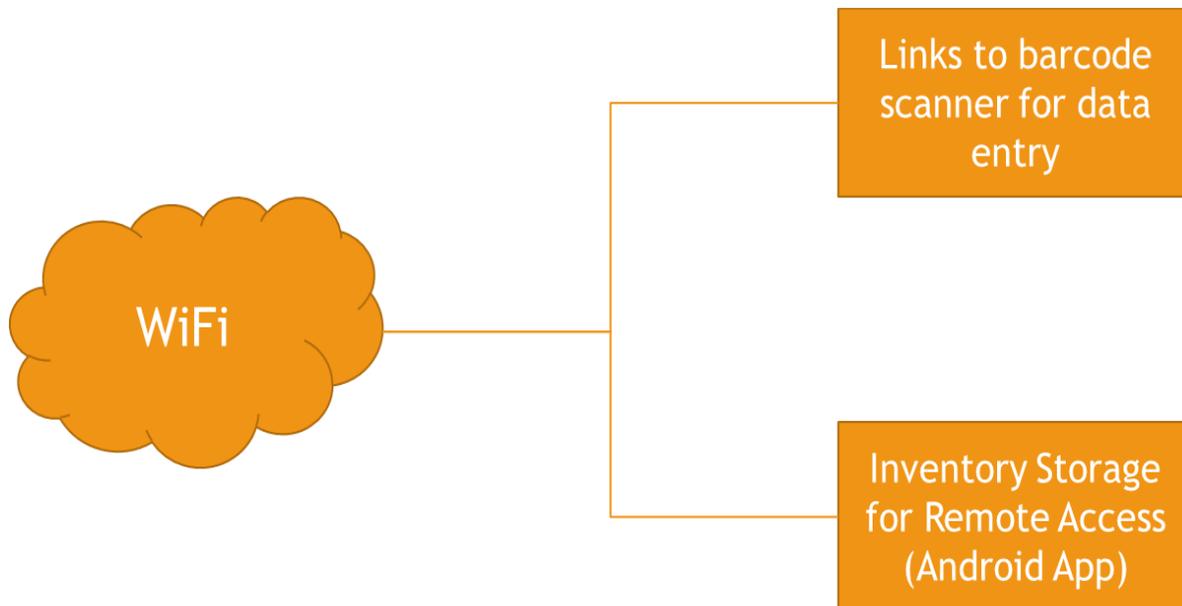


Figure 6: Simple Wireless Internet Connection Diagram.

VII. Prototype Development

The prototype development involved design of domains: mechanical design, hardware design, and software design. A 555 timer with an output of 1 kHz square wave was coupled to a stepper driver module. The module drives the stepper motor to a precise position based on the signal from the rotary encoder. This ensures that the proper position is referenced on every call. The rotary encoder is directly coupled to the 3/8 in threaded rod, meaning any rotation of the attached internal structure causes an increment in a count. This is used to create reference points for each bin location. The threaded rod is mounted to a bearing assembly above the rotary encoder and supports the entirety of the load to avoid damaging the encoder. Each shelf is also directly mounted to the rod and is coupled at the top to the stepper motor. The internal structure is fixed between the bearing assembly and the stepper motor. The motor control structure was developed with the use of a TIVA TM4C123GXL Microcontroller which reads conditions based on the output of the interface linked to the Raspberry Pi. The Raspberry Pi takes the user information and the information from

the bar code scanner for either storage or to relay information to the motor control microcontroller. This separation between the interfacing microcontroller and the motor control microcontroller allows for flexibility and stronger precision between the two elements.

The software design evolved from the use of tick boxes for the shelving units to buttons. Rather than having a tick box being checked and then clicking a button to switch shelving, buttons were created that automatically switch when pressed. The “Add,” “Remove,” and “Delete” buttons were incorporated to indicate the real time state of the food inventory. The “Add” button is to add the product to scan in. The “Remove” button allows you to remove the item from the shelving unit. And the “Delete” button allows deleting in case of an error. A “pop up” window was added to display the product and allows the user to tag it with various categories such as canned, cereal, sauce, spices, etc. The MySQL was integrated with the Raspberry Pi. Table 2 lists resources used for the project.

Table 2: Resources used for the Project

TB6560	Stepper Motor Driver	SainSmart	\$ 17.99
Raspberry Pi 3	Microcontroller	Amazon	\$ 37.58
Raspberry Pi Display	5" Touch Screen Display	Amazon	\$ 38.99
Stepper Motor	Nema 17 Bipolar 2A 4 lead	Amazon	\$ 13.99
USB Barcode Scanner	TaoTronics	Amazon	\$ 49.99
<i>Miscellaneous Hardware</i>	<i>(Construction Material)</i>	<i>Misc</i>	<i>\$ 124.00</i>
<i>Miscellaneous Components</i>	<i>Motor Driver Clock Circuit</i>	<i>Misc</i>	<i>\$ 10.00</i>
MeanWell EDR-120-24	24 VDC Power Supply	Allied Electronics	\$ 11.50
KY-040 Rotary Encoder	Rotary Encoder	Amazon	\$ 8.99

VIII. Testing Methods

As previously addressed, the prototype development required a specific procedure to ensure everything functions together as a system. Much of the integration of the hardware required properly functioning software and vice versa. Finally, the mechanical system operation required that both the hardware and the software interfaces work properly. The tests that have been implemented are shown in Table 4.

Table 4: Initial Test Procedure Outline

PHASE	PROCEDURE	Result Summary	Pass	Fail	Troubleshooting
1	Test motor control system for proper positioning through manual control to location address	Fixed with 555 Timer, external clock supply	X		Timing inconsistency
2	Test scanner system to link product UPC to database	Replaced camera with USB Scanner picks up UPC data	X		Camera does not recognize UPC
3	Test data storage to interface	Program is intergrated with server	X		Unable to use remote server to send Database; Local server implemented
4	Test product retrieval through interface	Application is able to pick up data from MySQL	X		JASON was encoded information; need to be decoded
5	Test Application inventory through WiFi link	Able to pick up data through WiFi	X		Server issues; unable to use port forwarding/assess through Raspberry Pi
6	Angle calibration for timing code	Divided sections and used delay function to hold output to clock pin	X		Ensuring output of pin turns off motor at correct location
7	Home location to reset motor location	Used a door switch to send input to know the location of the motor	X		Having a fixed location allows for reference when pantry first starting up; Auto adjusts.
8	GUI (too small of screen)	Adjusted the GUI to accomidate for easier selection with use of buttons	X		Font too small to use/select; chose buttons and check boxes for customer use

Table 5: Modifying of 555 timer circuit for 1kHz square wave output.

R1 Value	Output Frequency
6.5 k Ohm	950 Hz
6.9 k Ohm	990 Hz
7.1 k Ohm	1 kHz

Table 6 displays the results from using the 555 timer feedback to determine the required number of output clock pulses required for a quarter rotation, which would line up with physical bin locations.

Table 6: Implementation and results of 555 timer feedback to microcontroller.

Pulse Counts	Expected Change in Angle	Actual Change in Angle	Difference	Percent Error
850	90°	75°	15°	16.66%
900	90°	81°	9°	10%
1050	90°	100°	10°	11.11%
1025	90°	95°	5°	5.55%
1000	90°	90°	0°	0%

Table 7 displays the results of the precision testing of the previously implemented 555 timer feedback system. As shown by the results, the stepper motor encountered skipped steps. Because the motor doesn't actually have a depiction of its location and only depends on the counted input signal, the percent error in ideal location compared to actual location increased as the process continued. This would require a new solution for a feedback loop system.

Table 7: Results of consistency test for 555 timer feedback loop.

Cycle Count	Desired Angle	Actual Angle	Difference
1	90°	90°	0°
2	180°	180°	0°
3	270°	280°	10°

4	0°	10°	10°
5	90°	105°	15°
6	180°	200°	20°
7	270°	290°	20°
8	0°	20°	20°
9	90°	110°	20°
10	180°	205°	25°
11	270°	295°	25°
12	0°	25°	25°

Table 8 displays the results after the rotary encoder was implemented with the stepper motor. The rotary encoder was able to accurately determine, through an incrementing count caused by the system rotation, which position the motor was in.

Table 8: Results of accuracy of stepper motor after rotary encoder implementation.

Cycle Count	Desired Angle	Actual Angle	Difference
1	90°	90°	0°
2	180°	180°	0°
3	270°	270°	0°
4	0°	0°	0°
5	90°	90°	0°
6	180°	180°	0°
7	270°	270°	0°
8	0°	0°	0°
9	90°	90°	0°
10	180°	180°	0°
11	270°	270°	0°
12	0°	0°	0°

Table 9 lists the problems encountered and the solutions adopted during the development of the prototype. Figures 6-12 displayed in Appendix A present the details of the code developed for the prototype.

Table 9: Encountered Problems/Solutions.

Problems Encountered	Solutions
Raspberry Pi clock/motor issues	Created an external 555 timer
Python code implementation	Researched Python coding
MySQL Server Issues	Created local server
Raspberry Pi camera not picking up barcodes	Bought/Implemented a USB barcode scanner
Touch display formatted improperly	Modified display on Raspberry Pi root file
Touch screen not operating properly	Modified touch screen driver and calibrated screen
Ownership of files	Using "sudo su" in terminal to allow for transfer
API importing of string from array	UFC-8 code decoding in Python

IX. Evaluation and Synthesis of Learning

The basis for instructor evaluation relied on a variety of aspects. The ultimate goal was for the student team to develop a product over the course of four sessions following typical procedures for product development. The primary evaluation standard was based on how well the team was able to follow or adapt to the team's developed schedule. Milestones were set in the initial session and organized through a Gantt chart. As the development progressed, tasks and subtasks were analyzed for completion as well as documentation of progress.

Progress was documented through four key methods. The first method analyzed individual weekly performance through "engineering log books." Each member would submit a summary of his or her progress from the previous week. This summary included tasks completed, time spent on tasks, surfacing issues and troubleshooting, and intended plans for the succeeding week. The second method for progress evaluation incorporated a more neatly formatted compilation of team members' progress through "bi-weekly reports." Bi-weekly reports were used to examine overall

team progress in relation to the Gantt chart as opposed to just individual efforts. The third method followed progress overviewed in bi-weekly reports in the form of formal progress presentations. This allowed the student teams to present overall progress of their products, discuss any issues occurring, and brainstorm with classmates outside of teams to continue progress. The fourth method of documentation was presented in the form of formal project reports at the end of each session. Reports included summaries of progress, design summaries, technical documentation, technical schematics, and test procedures. The four reports were cumulative as the student team added information to previous reports to assist in developing the final report. The final aspect of evaluation was based on the final product's proper performance (demonstration of prototype) as part of the final formal presentation. The presentation covered the entirety of the development process as well as a demonstration. Additionally, a final report was submitted outlining the design process from start to finish.

Despite various hardware and software problems which were encountered, the project prototype was successfully implemented and demonstrated. The project enabled the team to incorporate concepts learned in the introductory classes of electronics and programming. The project has widened the scope of learning and knowledge in the electronics and computer fields for the team members. The project also helped the group members to develop higher levels of knowledge by learning totally new concepts and materials that were not covered during the course work.

The senior project course sequence also presented an excellent opportunity to directly measure the competencies (program objectives) of EET/CET/BMET graduating students. Two Rubrics, a national and a local, are used to evaluate each student on achieving program objectives (competencies) based on direct observation. The national assessment rubric is designed to gauge the student performance in achieving the program objectives, and the assessment data is used to take corrective action in terms of curriculum design and implementation. The local assessment tool is designed to identify student strengths and weaknesses at course sequence level; the assessment data obtained is used to take corrective action at local level (campus) by revising the course contents and teaching methodologies at the lecture and laboratory levels.

X. Conclusion

This paper described the concept of implementing a “Smart Pantry” system. The system scans food items and stores specific data in memory to be accessed through a touch screen user interface as well as a smartphone application in order to keep a personal itemized inventory. Additionally, the data stored includes a location for the food within the unit and is called upon in the automated item retrieval process. With the use of an Adafruit Raspberry Pi microcontroller and several peripherals, cost of prototype is much lower than similar competitive products in the market. In short the successful implementation of the project satisfied the requirements of the senior design capstone course at DeVry University, Addison, Illinois. The course requires a team effort to engineer and develop a concept as a solution to a problem. This allows students to gain technical knowledge, enhance their problem solving skills, and learn how to work in a group environment for successfully implementing a project.

References

- [1]. Grocery Shopping Statistics: 23 Fun Size Facts to Know. (n.d.). Retrieved June 13, 2016, from <https://www.creditdonkey.com/grocery-shopping-statistics.html>
- [2]. Malcolm, H. (2015, June 24). U.S. households trash \$640 in food a year. Retrieved June 13, 2016, from <http://www.usatoday.com/story/money/2015/06/24/americans-annual-food-waste/28988971/>
- [3]. Qi D, Roe BE (2016) Household Food Waste: Multivariate Regression and Principal Components Analyses of Awareness and Attitudes among U.S. Consumers. PLoS ONE 11(7): e0159250. <https://doi.org/10.1371/journal.pone.0159250>
- [4]. Samsung's \$6,000 smart refrigerator: A fridge too far? (n.d.). Retrieved June 13, 2016, from <http://www.cnet.com/products/samsung-family-hub-refrigerator/>
- [5]. Storage Motion. (n.d.). Retrieved June 19, 2016, from <http://storagemotion.com/autopantry/>
- [6]. 2-Tier Wood Pie-Cut Cabinet Lazy Susan. (n.d.). Retrieved June 19, 2016, from <http://www.lowes.com/pd/Rev-A-Shelf-2-Tier-Wood-Pie-Cut-Cabinet-Lazy-Susan/3032382>
- [7]. Stepper Motors. (n.d.). Retrieved June 13, 2016, from <http://www.tigoe.com/pcomp/code/circuits/motors/stepper-motors/>

Appendix A: Details of the Code

```
1 import mysql.connector
2
3 userN='pi'
4 userP='raspberrry'
5 host='127.0.0.1'
6 database='piproject'
7
8 upc="40"
9 name="banannas"
10 location="B1"
11
12 def addToDB(upc,name):
13     itemData = (upc, name, 'B1')
14     add_Item= ("INSERT INTO piproject nobjects "
15              "(UPC, NAME, LOCATION) "
16              "VALUES (%s, %s, %s)")
17
18     cnx = mysql.connector.connect(user=userN, password=userP, host=host, database=database)
19     cursor = cnx.cursor()
20     didwork = cursor.execute(add_Item, itemData)
21     print(didwork)
22
23     # Make sure data is committed to the database
24     cnx.commit()
25
26     cursor.close()
27     cnx.close()
28
29 def getFromDB():
30     cnx = mysql.connector.connect(user=userN, password=userP, host=host, database=database)
31     cursor = cnx.cursor()
32
33     query = ("SELECT UPC, NAME, LOCATION FROM nobjects")
34
35     cursor.execute(query)
36     items=[]
37     for (UPC, NAME, LOCATION) in cursor:
38         items+=[[UPC,NAME,LOCATION]]
39
40
41     cursor.close()
42     cnx.close()
43     return items
44
```

Figure 7: Code used to run MySQL server to retrieve, add and remove items.

```

1  #!/usr/bin/python3
2
3  from tkinter import *
4  from tkinter import ttk
5  import httplib2 as http
6  from urllib.parse import urlparse
7  import json
8  from connectMySQL import *
9  from p import *
10
11 def refresh():
12     popUP = Toplevel(root)
13     popUP.title("Select A Location!!!")
14     creatorL = Label(popUP,
15                     text="PLEASE SELECT A LOCATION\n\n\nCreated
16                     By:\n\bSpencer Karlovits\n    Project Manager\n\bKristoffer Francia\n
17                     Accounting & Report Development \n\bDominik Sobota \n    Research & Testing",
18                     font=("Arial", 12))
19     creatorL.pack(anchor=W)
20
21
22 def addThis(inputUPC):
23     global locationSelected
24     if locationSelected == 0:
25         refresh()
26         return
27     location = str(locationSelected)
28     text = inputUPC.get()
29     data = getUPCInfo(text)
30     print("text is: ")
31     rawdata = data[0].get('items')
32     title = rawdata[0].get('title')
33     addToDB(rawdata[0].get('ean'), title[:45], location)
34
35     print("THIS PRODUCT IS : " + rawdata[0].get('title') + " WITH A UPC OF: " + rawdata[0].get('ean'))
36     locationInfo()
37     return (data)
38
39
40 def getUPCInfo(UPCnum):
41     headers = {
42         'Content-Type': 'application/json',
43         'Accept': 'application/json'
44     }
45     ch = http.Http()

```

Figure 8: Part 1 of 4 of Main code. Majority of beginning are function definitions.

```

40 def getUPCInfo(UPCnum):
41     headers = {
42         'Content-Type': 'application/json',
43         'Accept': 'application/json'
44     }
45     ch = http.Http()
46     lookup = urlparse('https://api.upcitemdb.com/prod/trial/lookup?upc=' + UPCnum)
47     resp, content = ch.request(lookup.geturl(), 'GET', headers=headers)
48     content = content.decode('utf-8')
49     print(resp)
50     data = json.JSONDecoder().raw_decode(content)
51     print(data)
52     return data
53
54     locationSelected = 0
55
56 def locationInfo():
57     global locationSelected
58     itemText.delete('1.0', END)
59     items = getFromDB()
60     for index in range(len(items)):
61         if items[index][2] == str(locationSelected):
62             itemText.insert(END, items[index][1] + "\n")
63
64 def locationOne():
65     global locationSelected
66     locationSelected = 1
67     locationInfo()
68     moveToLocation(1)
69
70 def locationTwo():
71     global locationSelected
72     locationSelected = 2
73     locationInfo()
74     moveToLocation(2)
75
76 def locationThree():
77     global locationSelected
78     locationSelected = 3
79     locationInfo()
80     moveToLocation(3)
81
82 def locationFour():
83     global locationSelected
84     locationSelected = 4

```

Figure 9: Part 2 of 4 of Main Code. Top function offers accessibility to retrieve UPC information.

```

82 def locationFour():
83     global locationSelected
84     locationSelected= 4
85     locationInfo()
86     moveToLocation(4)
87
88     root = Tk()
89
90     root.wm_title("Smart Pantry")
91     content = ttk.Frame(root, padding=(3, 3, 12, 12))
92     frame = ttk.Frame(content, borderwidth=5, relief="sunken", width=350, height=350)
93     itemlbl = ttk.Label(content, text="Scan Item")
94     upcT = ttk.Entry(content)
95     itemContent = ttk.Entry(content)
96
97     shelflbl = ttk.Label(content, text="Shelf #")
98
99     onevar = BooleanVar()
100    twovar = BooleanVar()
101    threevar = BooleanVar()
102    fourvar = BooleanVar()
103
104    onevar.set(True)
105    twovar.set(False)
106    threevar.set(False)
107    fourvar.set(False)
108
109    one = ttk.Button(content, text="One", command=lambda: locationOne())
110    two = ttk.Button(content, text="Two", command=lambda: locationTwo())
111    three = ttk.Button(content, text="Three", command=lambda: locationThree())
112    four = ttk.Button(content, text="Four", command=lambda: locationFour())
113    addlbl = ttk.Button(content, text="Add", command=lambda: addThis(upcT))
114    dl1lbl = ttk.Button(content, text="Delete")
115    remvlbl = ttk.Button(content, text="Remove")
116
117    content.grid(column=0, row=0, sticky=(N, S, E, W))
118
119    frame.grid(column=0, row=0, columnspan=4, rowspan=2, sticky=(N, S, E, W))
120
121    scroll = Scrollbar(frame, orient=VERTICAL)
122    itemText = Text(frame, yscrollcommand=scroll.set, height=12, font=("Arial", 20))
123    scroll.config(command=itemText.yview)
124    scroll.pack(side=RIGHT, fill=Y)
125    itemText.pack(side=LEFT, fill=BOTH, expand=1)
126

```

Figure 10: Part 3 of 4 of Main Code. The use of Tkinter for Graphical User Interface.

```
126
127     itemlbl.grid(column=4, row=0, columnspan=2, sticky=(N, W), padx=5)
128     upcI.grid(column=4, row=1, columnspan=3, sticky=(N, E, W), pady=5, padx=5)
129     itemContent.grid(column=4, row=2, columnspan=3, sticky=(N, E, W), pady=5, padx=5)
130     addlbl.grid(column=4, row=1)
131     dltbl.grid(column=5, row=1)
132     remvlbl.grid(column=6, row=1)
133     shelflbl.grid(column=0, row=3, columnspan=2, sticky=(N, W), padx=5)
134     one.grid(column=0, row=4)
135     two.grid(column=1, row=4)
136     three.grid(column=2, row=4)
137     four.grid(column=3, row=4)
138
139     root.columnconfigure(0, weight=1)
140     root.rowconfigure(0, weight=1)
141     content.columnconfigure(0, weight=3)
142     content.columnconfigure(1, weight=3)
143     content.columnconfigure(2, weight=3)
144     content.columnconfigure(3, weight=1)
145     content.columnconfigure(4, weight=1)
146     content.rowconfigure(1, weight=1)
147
148     root.mainloop()
149
```

Figure 11: Part 4 of 4 of Main Code. Tkinter's orientation of graphic objects.

```
1
2 import RPi.GPIO as GPIO
3 import time
4
5 def moveToLocation(oldLocation, newLocation):
6     GPIO.setmode(GPIO.BCM)
7     GPIO.setup(16, GPIO.OUT)
8     GPIO.setup(20, GPIO.OUT)
9     for x in range(0, newLocation):
10        GPIO.output(16, 1)
11        GPIO.output(20, 1)
12        time.sleep(1)
13    GPIO.cleanup()
14    oldLocation = newLocation
15
```

Figure 12: Code used to interface with output to second microcontroller

```
1
2
3 import httplib2 as http
4 from urllib.parse import urlparse
5 import json
6
7 def getUPCInfo(UPCnum):
8
9     headers = {
10        'Content-Type': 'application/json',
11        'Accept': 'application/json'
12    }
13    ch = http.Http()
14    lookup = urlparse('https://api.upcitemdb.com/prod/trial/lookup?upc='+UPCnum)
15    resp, content = ch.request(lookup.geturl(), 'GET', '', headers)
16    content = content.decode('utf-8')
17    data = json.loads(content)
18    print(data)
19    return data
```

Figure 13: Code used to test UPC item retrieval.