Pantry Application Notification System (P.A.N.S.)

Date: November 2, 2020

Team Members:

Jonathan Aldridge

Ben Beauchamp

Zach Leggett

Jeremy Plunkett

Project Description

Uncertainty over what to buy at the grocery store is a common frustration, felt by many on a weekly basis. Unless meticulously planned beforehand, it is often very difficult for shoppers to know what items to buy, since they only have their own memories to go on. This leads to buying unneeded things and forgetting to buy the items that are needed. The Pantry Application Notification System (P.A.N.S.) gives a solution to this problem by allowing shoppers to open an app and immediately know what items are in their pantries. This system will use a barcode scanner and weight sensors to keep track of what is in the pantry, then display that information on an app, which will be viewable to the user. This will allow a grocery shopper who is unsure of what to buy to simply pull out a mobile device and know exactly what items are needed.

Technical Approach

System Overview

In order to implement the system, many subsystems were utilized and are explained throughout the proposal. Essentially, the user will utilize their phone to interface with the web app, hosted on the microcontroller, in order to view the live weight status of many of the items in the pantry. The user must place the necessary pantry items on the weight sensors and utilize the barcode scanner to keep an inventory of items not placed on the sensors. The weight sensor data will be stored in a SQL database and accessed from the website for easy viewing by the user. Please refer to Figure 1 for a summary of the system interactions. The fact that the system contains various software components and will be near food and liquids heavily influenced the constraints and standards considered and described below.

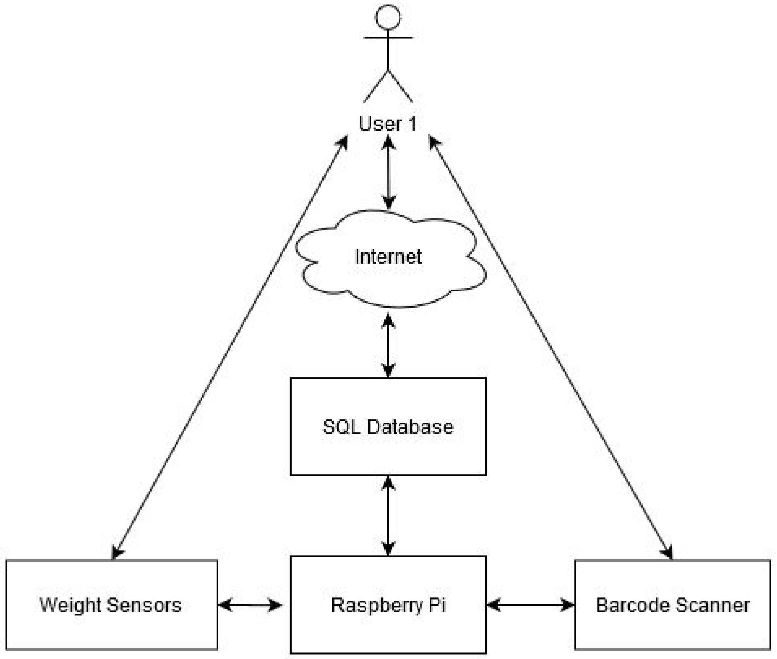

Figure 1: System Overview

Constraints

Economic

Our budget for this project is 125 dollars. The main expense will be the controller, which will cover about a third of the budget. Beyond that, all our expenses should be relatively small. A barcode scanner and four weight sensors will be used, along with a platform where the weight sensors will be attached. These are the main items that will need to be purchased, which should leave some room in our budget to account for price increases or slight plan changes.

Size

Size is a key constraint for this project because the entire system will need to fit neatly into a pantry. The weight sensors will be attached to the bottoms of platforms where food will be set and are small enough that they will hardly be noticed. The barcode scanner will need to be in a place where the user can easily access it, so it will sit on some unused shelf in the pantry, near the opening, but it will be small enough that it should be hardly an inconvenience. The Raspberry Pi controller is a somewhat sizable object that will need to be near the weight sensors for wiring purposes. It will be placed beside the weight sensors, as shown in Figure 3, and should stay mostly out of the way. In summary, this project will require several foreign items to be in the user's pantry, but they should all be small enough and far enough out of the way that they will not be an inconvenience.

Ease of use

The goal of this project is to create an easy-to-use system for tracking items in a pantry. An average consumer, with little or no technical knowledge, should be able to easily install and use this system. In order to accomplish this, many of the tasks will be automated, such as using the barcode scanner to track products. An easy-to-use web interface will also be implemented to provide easy access to the database of pantry items. Ease-of-use has been a major consideration throughout the entire design process.

Health and Safety

As this project involves being around food, health and safety has been a design consideration. All electronic components and wires will be properly enclosed to prevent any unwanted interaction with food items. The enclosures for all electrical components will be dust and light splash resistant to prevent any problems. Also, electronic components will not directly interact with food products.

Standards

IEEE 802.11ac - WiFi

The IEEE 802.11ac standard is a specification for LAN communications. The 802.11ac standard increases the maximum net data rates from previous iterations. It also specifies the use of multiple bands, both 2.4 GHz and 5 GHz. The 802.11ac standard also allows the use of many devices on one access point, which is more inline with modern WiFi systems. The use of 802.11ac will offer a wide range of compatibility with both new and legacy WiFi systems in the user's home. The Raspberry Pi used in this project will utilize 802.11ac WiFi in order to host a webserver, which the user can interact with.

NEMA Type 2 Enclosure

NEMA Type 2 enclosures are constructed for indoor use to provide a degree of protection to personnel against access to hazardous parts; to provide a degree of protection of the equipment inside the enclosure against ingress of solid foreign objects (falling dirt); and to provide a degree of protection with respect to harmful effects on the equipment due to the ingress of water (dripping and light splashing). Utilizing a NEMA Type 2 enclosure will ensure that all electronics are safe when placed around food or liquids.

RFC 2616 - HTTP/1.1

The RFC 2616 defines the Hypertext Transfer Protocol version 1.1 (HTTP/1.1). HTTP is an application-level protocol for distributed, collaborative, hypermedia information systems. HTTP/1.1 includes more stringent requirements than previous versions in order to ensure reliable operation. For this project, a web server will be used to host a simple website, which will utilize the HTTP/1.1 protocol. All standard definitions and procedures for the HTTP/1.1 protocol will be used.

HTML Living Standard

The World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG) have collaborated to create a common definition and standards for the Hypertext Markdown Language (HTML). HTML is the standard markup language for documents designed to be displayed in a web browser. HTML describes the structure and content of a web page. Our web server will utilize HTML when designing web pages. All standard HTML formats and protocols will be followed.

Python PEP8

The Python PEP8 standard provides coding conventions for the Python code comprising the standard library in the main Python distribution. PEP8 defines coding practices such as tab usage, source file encoding, imports, file and variable naming conventions, and much more. PEP8 standards will be utilized for all Python code used throughout this project. This standard will ensure that all code is written cleanly and is easy to read. PEP8 will also ensure that any code can be integrated with other standard libraries or projects.

Microcontroller

The microcontroller is a critical component of the design, as it will interface with every other hardware and software component. The microcontroller needs to be able to interface with the weight sensors and barcode scanner, and it needs to be powerful enough to run a basic SQL database and web server. This basis was used to form a list of criteria the microcontroller would need to meet to function well in the design. Eventually, 8 criteria were chosen to compare different options: clock speed, power draw, RAM capacity, GPIO, cost, community support, networking capability, and USB capability.

These criteria were chosen based on what the controller needs to do in the design, and certain factors were more important than others when comparing different options. For example, power draw was not as important because the system is going to remain plugged in to a wall AC outlet. Even the highest power draw microcontrollers will not pull nearly enough power to cause any noticeable negative effects. Criteria such as networking capability and RAM capacity, however, are very important. The controller needs to be able to run the SQL database and web server, so it must have both the ability to run these processes and have enough processing power and resources to keep them up.

In order to analytically compare the microcontrollers, a Pugh chart was used. This is because a Pugh chart allows the controllers to be compared using the weighted criteria. Each criteria is weighted from one to five, with five being the most important criteria and one being the least. Then, each microcontroller is assigned a value of -1, 0, or 1 for each criteria. A value of 1 means the microcontroller is very strong in that criteria, 0 is neutral, and -1 means it has the weakest support.

Design Criteria	Weight	Arduino UNO R3	Arduino Mega 2560 REV3	Raspberry Pi 3 Model B	Raspberry Pi 4 Model B	Teensy 4.1
Clock Speed	3	-1	-1	1	1	0
Power Draw	1	1	1	-1	-1	0
RAM	3	-1	-1	1	1	0
GPIO	4	-1	0	1	1	0
Cost	1	1	-1	0	0	1
Community Support	2	1	1	1	1	-1
Networking	5	0	0	1	1	-1
USB	2	0	0	1	1	-1
Total		-6	-4	18	18	-8

Table 1: MIcrocontroller Pugh Chart

As can be seen from the chart, both of the Raspberry Pi models scored the highest rank in the comparison, so these two became the primary choices. The Pi 4 was ultimately chosen, as the Pi 4 had superior specifications to the Pi 3 and was essentially the same price.

Weight Sensors

In order for the user to determine the current level of a particular pantry item, weight sensors will be used to estimate the percentage of the remaining item by calculating current weight divided by initial weight. Upon researching weight sensors that are compatible with the selected controller, the 50 kg load cell was picked due to its ease of use and low cost. In order to interface the load cell with the Raspberry Pi, the HX711 module is required to convert the analog reading to digital. Because this setup is very straightforward and works very well with our application, no other alternatives were considered.

As can be seen in Figure 2 below, the team decided to go with four sensors, corresponding to four live data readings, for the design. However, this design can be easily extended to support more weight sensors if desired. Based on the wiring diagram, each of the HX711 modules require 5V power supply which will be supplied from the Raspberry Pi. Additionally, the modules will communicate with the Raspberry Pi via the GPIO ports seen in the wiring diagram. The HX711 can be used with a pre-existing python library, discussed in more detail in the Software section, which will allow us to calibrate the sensors and continuously capture data readings from the IO ports.

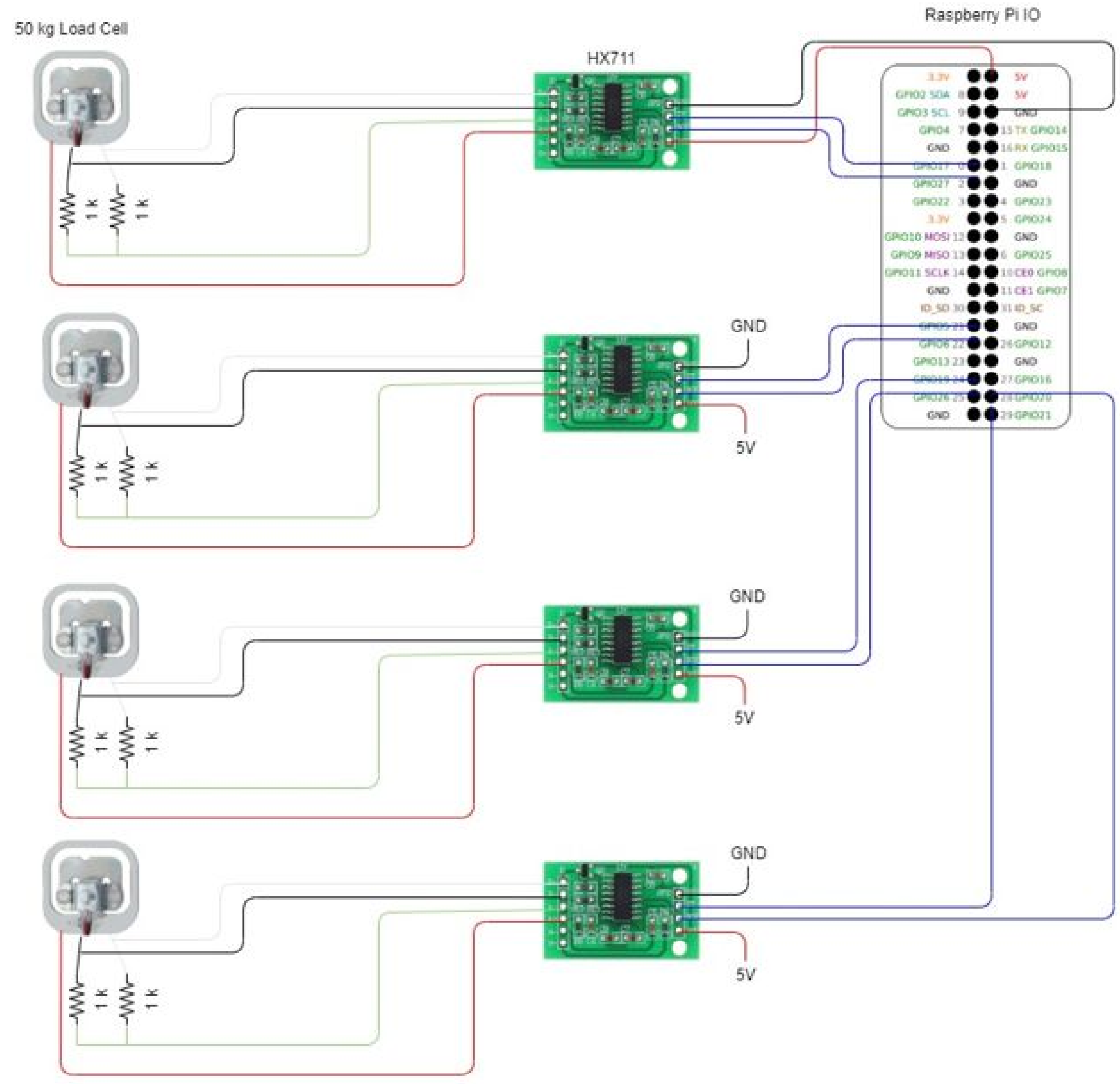

Figure 2: Weight sensor wiring diagram

In order to properly utilize the weight sensors, a bracket must be used which allows the inner part of the sensor to flex relative to the outer square. This can be accomplished by adding a spacer between the outer rim of the load cell and the platform on which the object to be weighed will sit. The team was able to locate an existing mounting frame from: https://www.thingiverse.com/thing:2624188 and plans to have multiple frames 3D printed. Additionally, the team designed a platform which houses the four weight sensors, and frames, along with the Raspberry Pi and potentially the barcode scanner (discussed in a future section).

A to-scale 3D model of the platform was created, and seen in Figure 3. As can be seen, the weight sensor frame will be mounted to a 6”x6” platform (in green) which should allow most pantry items to sit on top. The platform and sensor frame combination will then be stationed on a plywood surface which has been designed to easily be placed on a pantry shelf or counter top. While the design will only support live readings of up to four items, a barcode scanner will be utilized to keep an inventory list of other items in the pantry and will be discussed in the following section.

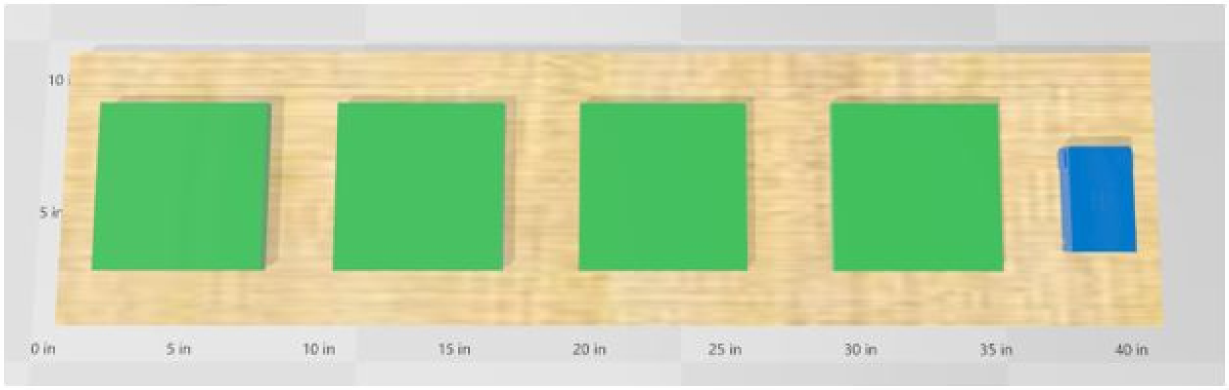

Figure 3: Weight sensor platform top view

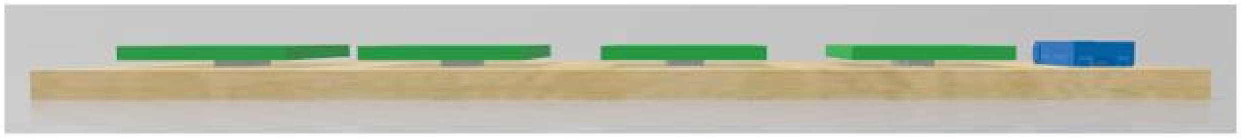

Figure 4: Weight sensor platform front view

Barcode Scanner & API

In order to keep an inventory of all items in the pantry, a barcode scanner will be used. The scanner will be connected via USB to the Raspberry Pi and will capture the Universal Product Code (UPC) from each product. The UPC consists of 12 numeric digits that can be used to gather information about a product, such as the name and price. Once the UPC has been captured by the barcode scanner, a variety of APIs will be used to query multiple UPC databases for information about the product. A table detailing the APIs and databases that will be used can be seen below.

Barcode APIs/Databases	
Name	**Link**
UPC Database	https://upcdatabase.org
UPC Lookup	https://www.upcitemdb.com
International Barcodes Database	https://barcodesdatabase.org

Table 2: Barcode APIs

The team decided to use several different databases to offer redundancy within the system. There is a chance that a given UPC may not be in the first database queried, but it may be in one of the others. The information from each database can also be verified by comparing it to the results from the other databases. If a given UPC can not be found in any of the databases, then the user will be prompted to manually enter the product information via the web interface. The information entered by the user will be saved and associated with that UPC, such that when that UPC is scanned again the information will be automatically applied. The barcode scanner and API will allow the system to easily identify and record the products stored in the user's pantry.

Software

By the nature of the project, there will need to be different software packages running on the Raspberry Pi. In particular, the software stack can be seen in Figure 5, and each element will be discussed in more detail. Essentially, the Pi will house Python 3.9, an Apache web server, a mySQL database, and PHP.

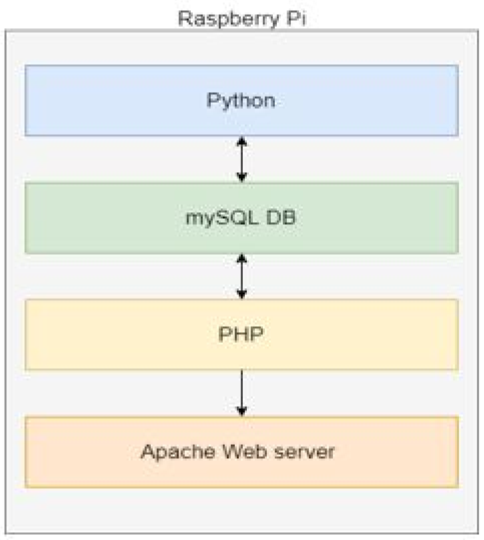

Figure 5: Software stack

After some discussion, the group decided to use python as the main programming language for the project. Many of the group members are experienced with python, and there exist numerous resources for python programming on the Raspberry Pi. Additionally, python is lightweight and free, and it can easily be installed on the Pi. Furthermore, there already exists a python library to allow data collection from the load cells and HX711 module mentioned above, which will greatly reduce the code needed to interface with the weight sensors.

The team plans to write a python script which will collect data from the weight sensors and barcode scanner to store into a SQL table. The finer details have not yet been worked out, but the script will log the current value of the weight sensors into the table when there is a significant change in the weight. Each table entry will consist of four fields: Item Name (string), Current Weight (float), Initial Weight (float), and Percent Remaining (float). The Item Name column will be populated by the return of the API call from the barcode scanner reading or by manual entry from the user. The other columns will be determined from the readings of the weight sensor. Additionally, the table can contain entries with no weight association to keep up with an inventory list of items that may be too light to weigh (spices, etc.).

In order to display the weight sensor readings and inventory list to the user, the Pi will host an Apache web server utilizing PHP to display dynamic web pages. A web app will be created which will initially display the contents of the SQL table, via a PHP script, that will ideally be accessed from the user's phone on any network. Once the group has successfully connected to the SQL DB and can display the table, the group will look into creating a nicer front-end to display the data and allow the user to manually input data into the table and potentially alert the Raspberry Pi that the user wishes to place a new item on one of the weight sensors.

Management

The design team conducts meetings every Monday afternoon over Discord video chat to discuss what was accomplished and what needs to be done in the upcoming week. The text chat in the Discord server is used to communicate throughout the week with the team. Since the team on this project is small, there are no rigid roles for each person, as the group is able to collaborate well amongst themselves. However, there are certain roles that some of the team members have taken on naturally. Jonathan Aldridge is the Project Lead, Zach Legget and Jeremy Plunkett are collaborating on the software aspect of the design, and Ben Beauchamp is working on the hardware design. Together, the entire team discussed all of the different aspects of the design to determine the most optimal final product.

All files needed for the project are stored in a shared Google Drive folder that the entire team has access to. The timeline, meeting notes, and other assorted documents are all stored in the Drive and can be accessed and edited by everyone on the team.

Meeting notes are stored on the shared Drive for the team members to access. They can be accessed at this link: https://drive.google.com/drive/folders/1QgGVGWG2op7hgiMuVGt7MtTtwno1n352?usp=sharing

Budget

The budget is in an Excel spreadsheet primarily managed by Jonathan Aldridge, but other team members are also able to contribute. This project is self-funded, and the cost is expected to be fairly low due to many of the components being inexpensive. The spreadsheet is hosted on the shared Drive and can be accessed here: https://docs.google.com/spreadsheets/d/1bYfph8h1tpyEgECxfuNoD6on-S6UK07Macsq4vcb_v0/edit?usp=sharing

Timeline

A Gantt chart is being used by the team to keep track of the progress made throughout the semester. The Gantt chart is updated during the team's weekly meeting and serves as a pacing guide and reminder to the team of upcoming deadlines. The timeline is kept in the team's Google Drive and is accessible from the project website and the following link: https://docs.google.com/spreadsheets/d/120OwW1HsScWIFaSJwOBCxG2Z6OXZ6kK9EZuiiHj4pU4/edit#gid=0.

Facilities

The facilities anticipated to be used throughout the project are: Auburn University engineering labs and the Aldridge family's woodworking tools.

Disposition Agreement

The design team for the Pantry Application Notification System has agreed to disassemble the project at the end of the spring semester of 2021 and give all purchased hardware to Zach Leggett, to use as he pleases. By signing below, each team member gives his assent to that agreement and gives up his rights to all designs involved in the implementation of this system.

Members:	**Signatures:**
Jonathan Aldridge	[illegible]
Ben Beauchamp	[illegible]
Zach Leggett	[illegible]
Jeremy Plunkett	Jeremy Plunkett