Inventory Locator Assistant for ETG Shop

Final Report

Senior Design Team: SDMay20-09

Client: Mr. Leland Harker

Adviser: Mr. Leland Harker

Alejandro Buentello: Lead Hardware Manager

Caleb Gehris: Communications Manager

Jacob Linch: Lead Software Manager

Kurt Markham: Meeting Supervisor

Erin Power: Planning and Time Management Supervisor

Chris Rice: Lead Database Manager

Team Email: sdmay20-09@iastate.edu

Team Website: http://sdmay20-09.sd.ece.iastate.edu

Table of Contents

1 Introduction

1.1 Acknowledgement

We would like to thank and acknowledge that Mr. Leland Harker has helped guide us in the implementation of this project. We would also like to thank and acknowledge the ETG for their help ordering and selecting parts for our project.

1.2 Previous Work And Literature

This system is based off of an Instructable that was created by Dustin Dobransky. It offers an easy to use locator assistant for a small single bin storage system that helps the user find what they are looking for as well as similar products. However, the system that Dobransky created is not very suitable for larger scale applications or applications that utilize different storage systems like those needed by the ETG shop. We have modified Dobransky's creation to better suit our client by reducing the number of LEDs per bin to save cost, creating a variable set up system to allow use with different storage systems, adding a typed search option, adding a system for easy scalability, reducing the types of queries that are not needed, and adding a system to test the functionality of the system so that problems can be troubleshooted.

1.3 Problem and Project Statement

There are thousands of components in the ETG shop in a multitude of cabinets, such that it can take a considerable time for an employee to find a single component. In order to reduce the amount of time and confusion involved in finding a component, the ETG shop had requested a locator assistant system that would guide the user to find the correct component with minimal effort.

Our product will implement a solution involving a tablet that the client can use to request their specific component and a grid system constructed of LED strips used to identify its location. It will work through accepting voice or text inputs on the tablet. It will also have a database that contains all the parts and their locations that can be added, removed, or changed as needed.

1.4 Intended Users and Uses

Our intended users for this product are Leland Harker and the current and future employees of the ETG. It is to be used to assist ETG employees in the locating of parts in the shop and for Leland or other individuals to update the location of parts or add and remove them from the project's scope as necessary.

1.5 Assumptions

1.5.1 The product will only be used in the ETG shop and by ETG employees.

1.5.2 There will only be one user using the product at a single instance.

1.2.3 The parts database shall be kept up to date.

1.5.4 The product will have continuous WiFi connection during its operation.

1.6 Limitations

1.6.1 The total budget for the project will not exceed five hundred dollars.

1.6.2 The product must operate continuously during ETG shop business hours.

1.6.3 The LED strips must fit onto the front of each of the cabinets without excess.

1.6.4 Unable to test the system in its working environment due to the closure of the university over worries of COVID-19.

2 Requirements

2.1 Functional Requirements

2.1.1 The product shall accept input via voice or text queries

2.1.2 The product shall allow for adjustments of timing, brightness, and colors of the LEDs

2.1.3 The commands for “last search”, “all off”, and “all on” shall be implemented

2.1.4 There shall be test and configuration routines for the system

2.1.5 The product shall use visual motion to direct the user to the correct LED positions

2.1.6 The product shall allow for searches for multiple parts and use different indicators for each individual part.

2.2 Non-functional Requirements

2.2.1 The product shall allow for the expansion of the system onto multiple cabinets

2.2.2 The database used in the product must be able to be used outside of the product

2.2.3 The search functions must run in an efficient time

2.2.4 The product shall use authentication when communicating between the LEDs and the application

2.2.5 The product shall use authentication when communication between the database and the application

2.2.6 The product shall have easy to read and understand documentation

2.2.7 The product shall be easy to understand and use

2.2.8 The product shall be easy to modify if needed

3 Design and Development

3.1 System Design

2.3.1 User Interface

An Android application running on a tablet, giving the user the freedom to take the tablet and use the system from anywhere within the ETG.

2.3.2 LEDs

Strips of WS2812 individually programmable LEDs that allow us to control the color and brightness of each LED to better direct the user to the required part.

2.3.3 ESP8266 Microcontrollers

Wi-Fi enabled microcontrollers that allow the LED strips to be placed on any cabinet in the ETG with extra wires running all over the place.

2.3.4 Database

MySQL database hosted on a provided ISU server that will store information about the parts within the ETG.

2.3.5 API

Python based Flask Api which allows us to access the database for required information and send it to the microcontrollers and Android app.

2.3.6 Website

Allows the user to more easily manage the database; adding, editing, and removing parts as they please.

3.2 Design Analysis

While we originally chose to use a Raspberry Pi instead of a server, we determined that the fewer the components the less chance of a breakdown. The server allows us to accept inputs and change the database.

The LED strips gain Wi-Fi capabilities through connecting them to a Wi-Fi enabled microcontroller. This allows for LED strips to be put wherever necessary. This also allows easier addition of cabinets to the system as each cabinet will have its own respective microcontroller devoted to it. The microcontrollers we selected are also relatively cheap and easy to set up, unlike the LED strips which are more expensive due to the LED strips being multi-colored and individually addressable. These functions are vital to the operation of the system, so the cost will have to be accepted when future LED strips are needed.

3.3 Development Process

The development process that we used is kanban. We wanted to avoid a waterfall design because we have direct communication with our client, so the ability to adjust and change our designs as needed is going to be invaluable as the project progresses. Additionally, we decided against agile because a system of two week sprints would be difficult when our team members can only work on this project ~10 hours per week due to other commitments. By using kanban, we get the ability to work toward our own pace, while also having a good visual representation of the progress we have made compared to what is left to complete.

3.4 Alternative Designs

One of our initial designs we worked at included a Raspberry Pi that fulfilled the role of the server and database. The design had the Pi interact between the tablet and microcontrollers over the ISU internet with the Pi being placed in the ETG shop. The Pi also hosted a SQLite database that would store all the part information. As the project developed, concerns regarding the reliability of the Pi and the storage size of the database arose. It was from there that we decided to move the backend operations to a ISU-hosted server. This allows for expandability by increasing the size of the server and the reliability by not depending on the Pi's physical condition, whether it wears down over time or accidents occur to it.

4 Implementation

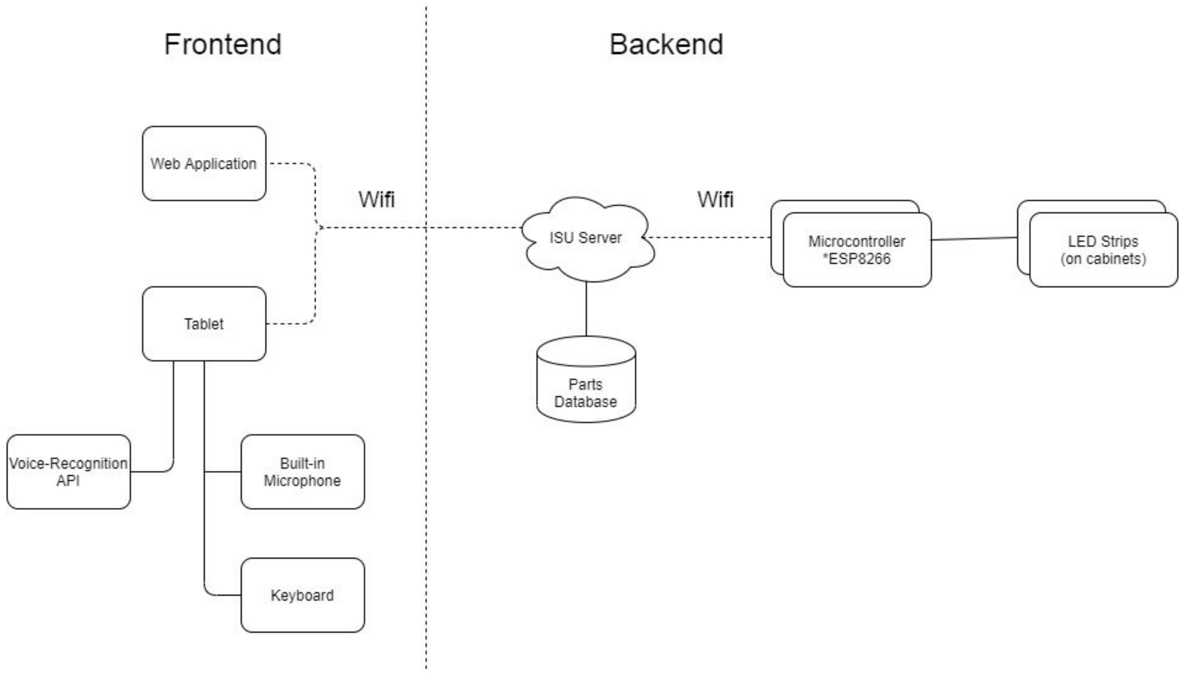

Figure 1: Block Diagram of Implementation

4.1 Implementation

The figure above shows the modules that were used and their connections to each other. On the frontend are two user interfaces: a web application that allows users to easily manage the database and an Android application running on a tablet that allows the user to query a part using text or speech input. For speech input, a voice recognition API provided by Google takes an audio stream and converts said stream to a text string that is then passed back to the application. Once the user has entered their search, the Flask API on an ISU server will process the search and return the part that best matches the user's query from the database. The server will also send the location of the part to the microcontroller on the corresponding cabinet. The microcontroller will then send a signal to the attached LED strips to light up the LEDs that correspond to the horizontal and vertical location of the part in the cabinet.

4.2 Setup

Due to the closure of the university due to COVID-19 we were unable to set up the project in its final location. Therefore, a set of instructions are available in Appendix 2 at the end of the document detailing how to do so. We will also leave a list of parts that will be needed to expand the project to additional cabinets in Appendix 3. The following will need to be set up:

4.2.1 Install the Android application on tablet of choice

4.2.2 Size the LED strips for each cabinet

4.2.3 Install the microcontroller(s) and LED strips

If additional microcontrollers are desired

4.2.3.1 Add the microcontroller to the ISU network

4.2.3.2 Install software on the microcontroller

4.3 Engineering Standards and Design Practices

4.3.1 Engineering Standards

4.3.1.1 HTTP

4.3.1.2 IP Protocol

4.3.1.3 UDP Protocol

4.3.1.4 802.11 Wifi Protocols

4.3.2 Design Practices

4.3.2.1 Kanban Software Development

4.3.2.2 Test-driven Development

4.3.2.3 Peer Review

4.3.2.4 Standardize Naming Convention for Variables

4.3.2.5 Well Documented Coding

4.3.2.6 Good Error Handling

5 Testing

5.1 Interface Specifications

While testing we want to verify that the communication between all of the parts are working correctly. To achieve that we will test the following:

5.1.1 Verify the communication between the application and the server

5.1.2 Verify the communication between the server and the LEDs

5.1.3 Test the persistence of data on any database add or update

5.1.4 Test the communication security between client and server

5.1.5 Check handling of network failures

5.2 Hardware and Software

5.2.1 Hardware

The following will be done to test the hardware:

5.2.1.1 Extensive manual testing following a test matrix

5.2.1.2 Creating a testing circuit to test the LEDs

5.2.2 Software

The following will be done to test the software:

5.2.2.1 Unit testing using JUnit testing library making sure all the internal functions work as expected

5.2.2.2 UI testing using the Espresso UI library testing that nothing breaks with the UI when changes are made to it

5.2.2.3 Manual testing to simulate a user using the application

5.3 Functional Testing

For our functional testing we will be making sure that all of our functional requirements are verified. The following is what we will test for each requirement:

5.3.1 The product shall accept input via voice or text queries

5.3.1.1 Test different words using different levels of volume to check input accuracy

5.3.1.2 Test different text inputs checking different outlier cases for accuracy

5.3.2 The product shall allow for adjustments of timing, brightness, and colors of the LEDs

5.3.2.1 Test different functions manually verifying the accuracy of the changes

5.3.3 The commands for “last search”, “all off”, and “all on” shall be implemented

5.3.3.1 Test the functions manually verifying the accuracy of the functions

5.3.4 There shall be test and configuration routines for the system

5.3.4.1 Run the testing and configuration routines and manually verify the accuracy of the tests and configuration

5.3.5 The product shall use visual motion to direct the user to the correct LED positions

5.3.5.1 Run the function and manually test the accuracy of the LED positions

5.3.6 The product shall allow for searches for multiple parts and use different indicators for each

5.3.6.1 Test queries with multiple results and verify it is allowed

5.3.6.2 Test running the function and manually verify the lights use different indicators for each part

5.4 Non-Functional Testing

For our non-functional testing we will make sure the non-functional requirements are verified. To verify the requirements we will do the following:

5.4.1 The product shall allow for the expansion of the system onto multiple cabinets

5.4.1.1 Test how much we can expand the system by load testing the connections

5.4.1.2 Verify that we can expand the system easily

5.4.2 The database used in the product must be able to be used outside of the product

5.4.2.1 Manually verify the ability to use the database outside of the product

5.4.3 The search functions must run in an efficient time

5.4.3.1 Analyze the search functions for efficiency

5.4.3.2 Measure the amount of time it takes to run the function

5.4.4 The product shall use authentication when communicating between the LEDs and the application

5.4.4.1 Use tools to check the security of the connection

5.4.4.2 Validate the accuracy of the authentication

5.4.5 The product shall use authentication when communication between the database and the application

5.4.5.1 Use tools to check the security of the communication

5.4.5.2 Validate the accuracy of the authentication

5.4.6 The product shall have easy to read and understand documentation

5.4.6.1 Have different users read the documentation to see if there is anything that needs to be clearer

5.4.7 The product shall be easy to understand and use

5.4.7.1 Have the client use the product and verify they understand how to use it and that it is intuitive

5.4.8 The product shall be easy to modify if needed

5.4.8.1 Verify the functions for the product are documented

5.4.8.2 Verify all code matches the same standard

5.5 Process

While developing the product we will use the following process to test and verify the product. First, we will use a test driven development approach when implementing a feature. The main process used when developing is as follows:

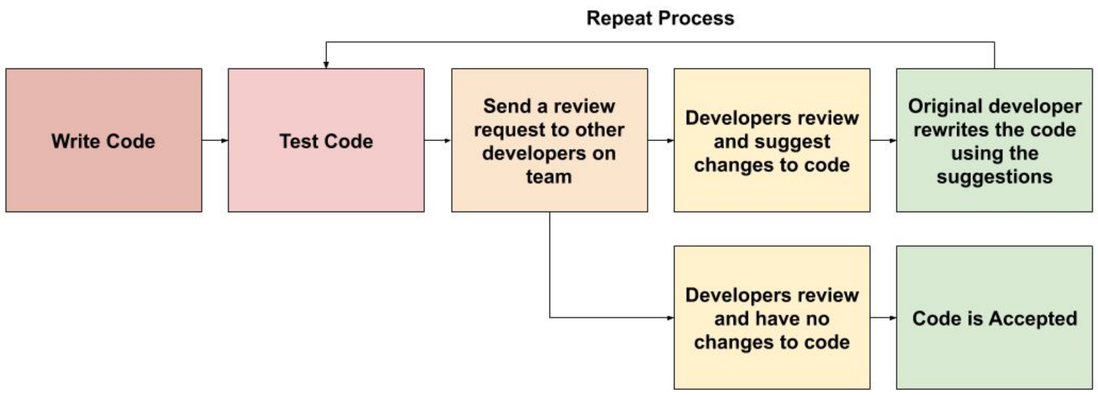
Repeat Process
Write Code
Test Code
Send a review request to other developers on team
Developers review and suggest changes to code
Original developer rewrites the code using the suggestions
Developers review and have no changes to code
Code is Accepted

Figure 2: Testing Procedure

The process we will use after a new feature has been implemented will use the following steps:

5.5.1 Manual test the feature

5.5.2 Run unit tests to validate the internal functions added

5.5.3 Run UI tests to verify nothing changed with the UI

5.5.4 Manual Regression test using a testing matrix to verify changes

Appendix 1 - Use Instructions

1. Open the app on an Android tablet
2. Type in your query into the search box or hold down the microphone button to speak a query
3. Tap the magnifying glass button to start the search
4. Tap the menu button at the top left of the screen to show access to other app features
5. Tap the “Add Part” tab to bring up a screen to add a part to the database
    a. Fill in the appropriate data and tap “Add”
6. Tap “Parts” to see all the parts in the database
    a. Tap the refresh button to get update the screen
7. Tap the “Settings” tab to change the server address if needed

Appendix 2 - Setup Instructions

Installing the Android Application

1. On the tablet change settings to allow apps from unknown sources
    a. Menu -> Settings -> Security -> check "Unknown Sources"
    b. May need to go select "Files" or "My Files" and allow from there
2. Connect the table to a computer
3. Select "Media Device" when prompted
4. Copy the provided APK file 'InventoryLocatorAssistant.apk' onto the tablet
5. Tap the APK file to initiate installation and wait for it to finish

Adding a New Microcontroller to the Network

1. Ask the IT desk in the ETG to add a new device to the network
2. The name of the device will be 'ilamicro**X**.ece.iastate.edu', where **X** is the cabinet number the device will be installed on

Installing the Software on New Microcontrollers

1. Connect the microcontroller to a computer with a USB A to Micro-B USB cable
2. Download the Arduino IDE (if not already installed) from **https://www.arduino.cc/en/Main/Donate** (donation is not required just click on the "Just Download" button)
    a. If popups occur asking for permission to install drivers from Adafruit or Arduino click "Allow" or "Install"
3. With the Arduino IDE open click on File -> Preferences
4. In the "Additional Boards Manager URLs" section copy and paste the following link: http://arduino.esp8266.com/stable/package_esp8266com_index.json
5. Click "OK"
6. Go to Tools -> Board: -> Boards Manager
7. In the Search bar type in: esp8266
8. Click "Install" on esp8266 by ESP8266 Community
9. Go to Tools -> Boards -> Generic ESP8266 Module

10. Go to Tools -> Port -> select the port (if multiple ports show up it will take trial and error, first one listed is usually the best option)
11. Copy the provided ‘FastLED.zip’ file to desired location
12. Go to Sketch -> Include Library -> Add .ZIP Library -> navigate to copied file location
13. Open the provided file ‘MicroprocessorCode.ino’ in the Arduino IDE
14. Click on the “Upload” button (rightward pointing arrow)
15. Wait for the terminal header to display “Done Uploading”
16. Unplug microcontroller from computer

Installing the Microcontrollers and LEDs

1. Place microcontroller in mini breadboard
2. Affix the breadboard to desired location on cabinet
3. Connect leads to LED (if required) so that the male end of the lead is able to reach the breadboard
4. Plug the ground (GND) lead from the LED strip into one of the ground pins on the microcontroller
5. Plug the 5 volt lead from the LED strip into one of the 3V3 pins on the microcontroller
6. Plug the data lead (DI) from the LED strip into the D4 pin (for the Y or vertical LED strip) or the D5 pin (for the X or Horizontal LED strip)
7. Peal of backing or LED strip and place on cabinet (additional adhesive or tape may be necessary to keep strip in place)
8. Repeat 4-7 for other strip
9. Plug in the microcontroller into a 5v USB power source

Sizing the LED strips

1. Determine length of strip needed
2. Cut strip close to desired length through the center of the gold contact pads between LEDs
3. For the following steps make sure you do so on the end of the strip with the data line being DI or Din not DO or Dout
4. Shave down waterproof coating (if applicable) to expose the contacts
    a. Easiest way it to peel coating away from strip with fingernail and use scissors or wire cutters to cut of the small portion covering the contacts
5. Solder leads to contacts on the strip that will be long enough to reach the microcontroller

Appendix 3 - Suggested Parts

Mini Breadboards:
https://www.amazon.com/ZYAMY-Solderless-Breadboard-Protoboard-Self-Adhesive/dp/B075DZ2Y3N/ref=sr_1_16?dchild=1&keywords=mini+breadboard&qid=1587925226&sr=8-16

ESP8266 Microcontrollers:
https://www.amazon.com/HiLetgo-Internet-Development-Wireless-Micropython/dp/B081CSJV2V/ref=sr_1_1_sspa?dchild=1&keywords=nodemcu+esp8266&qid=1587925311&sr=8-1-spons&psc=1&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyWDZROUJIWDNHSzJZJmVuY3J5cHRlZElkPUEwNjU4MTAwMklZN0dNTjEyWUJKVSZlbmNyeXB0ZWRBZElkPUEwNTYyNjkzMU5WTEI1SjdJUTlDJndpZGdldE5hbWU9c3BfYXRmJmFjdGlvbj1jbGlja1JlZGlyZWN0JmRvTm90TG9nQ2xpY2s9dHJ1ZQ==

WS2812 LED Strips (60 LEDs/meter):
https://www.amazon.com/BTF-LIGHTING-Flexible-Individually-Addressable-Non-waterproof/dp/B01CDTEJBG/ref=sr_1_1_sspa?dchild=1&keywords=WS2812+LED+Strips&qid=1587925386&sr=8-1-spons&psc=1&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyQ0dYWVFPQzNJTjZXJmVuY3J5cHRlZElkPUEwMDEyMjQyMjU4Mk5VUEVGRFlEViZlbmNyeXB0ZWRBZElkPUEwMTMwMjE2MVZQM0IwWEdRWU1SQyZ3aWRnZXROYW1lPXNwX2F0ZiZhY3Rpb249Y2xpY2tSZWRpcmVjdCZkb05vdExvZ0NsaWNrPXRydWU=

Jumper Wire Ribbon Cables:
https://www.amazon.com/EDGELEC-Breadboard-Optional-Assorted-Multicolored/dp/B07GD1TFBR/ref=sr_1_1?crid=1XRWOHNNEMKS7&dchild=1&keywords=jumper%2Bwires%2Bribbon%2Bcables&qid=1587925537&sprefix=jumper%2Bwire%2Bribb%2Caps%2C173&sr=8-1&th=1

USB A to Micro B Cables:
https://www.amazon.com/Android-Charging-Compatible-Smartphones-More-Blue/dp/B07CZHV9VF/ref=sr_1_5?crid=P6LZY2UX0WDN&dchild=1&keywords=micro+usb+cable+6ft&qid=1587925665&sprefix=micro+usb+%2Caps%2C190&sr=8-5

Android Tablet (Non Samsung or Amazon):
https://www.amazon.com/MatrixPad-Android-Quad-Core-Processor-Bluetooth/dp/B07PXB4F7N/ref=sr_1_3?dchild=1&keywords=android+tablet&qid=1587925934&refinements=p_n_feature_four_browse-bin%3A5928098011&rnid=2528832011&sr=8-3