Inventory Locator Assistant for ETG Shop

DESIGN DOCUMENT

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

Revised: 4/26/2020 V5.0

Executive Summary

Engineering Standards and Design Practices

Engineering Standards

- HTTP
- IP Protocol
- UDP Protocol
- 802.11 Wifi Protocols

Design Practices

- Kanban Software Development
- Test-driven Development
- Peer Review
- Standardize Naming Convention for Variables
- Well Documented Coding
- Good Error Handling

Summary of Requirements

- User Interface
  - User can input a part to search for using both typed input and voice commands
  - User can input new parts and their coordinate location within the system
- Database
  - Stores a list of items, including Name, Keywords, Location (stored as X, Y coordinates), and Manufacturer.
  - Allows for easy and efficient querying
  - Should be cloud storage for other use by ETG.
- Hardware
  - A system of LEDs that cover the X and Y axes of all five cabinets
  - Some form of system specific hardware to use for user input

Applicable Courses from Iowa State University Curriculum

- COM S 319: Construction of User Interfaces
- SE 329: Software Project Management
- COM S 309: Software Development Practices
- SE 339: Software Architecture and Design
- COMS 363: Introduction to Database Systems
- EE 201: Electric Circuits
- EE 230: Electronic Circuits and Systems
- EE 186: Introduction to Electrical Engineering and Problem Solving II
- CprE 288: Embedded Systems I: Introduction
- CprE 488: Embedded Systems Design

New Skills/Knowledge acquired that was not taught in courses

- Android Development
- Cloud Database
- LED Circuit Development
- Speech to Text Android API Usage
- Web development using React
- Flask
- Python Development

Table of Contents

List of figures/tables/symbols/definitions

Figures:

Tables:

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 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.

1.3 Operational Environment

Our product will operate within the ETG Shop in Coover Hall. Hazards we expect include: people hitting the LED strips or tablet and server issues. We also expect our product to get dirty over its lifetime through spilled liquids or dust. The database for the project must always be running with the processing power for a medium load.

1.4 Requirements

1.4.1 Functional Requirements

1.4.1.1 The product shall accept input via voice or text queries

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

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

1.4.1.4 There shall be test and configuration routines for the system

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

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

1.4.2 Non-functional Requirements

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

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

1.4.2.3 The search functions must run in an efficient time

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

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

1.4.2.6 The product shall have easy to read and understand documentation

1.4.2.7 The product shall be easy to understand and use

1.4.2.8 The product shall be easy to modify if needed

1.5 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.

1.6 Assumptions and Limitations

1.6.1 Assumptions

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

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

1.6.1.3 The parts database shall be kept up to date.

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

1.6.2 Limitations

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

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

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

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

1.7 Expected End Product and Deliverables

ISU Server: A server is used as the central control unit for the system. The server will accept an audio or text input over Wi-Fi for a requested part. Voice-processing software will be installed to interpret audio inputs. The server then searches for the coordinate data of the requested part and sends it to the related microcontroller over Wi-Fi. Code written for the server will be provided to the client at project's completion

Android Tablet with Custom Mobile Application: A Nexus tablet running the Android OS is to be used as a peripheral for the system to accept verbal and typed inputs from the user. The input will be accepted through an application installed on the device and then sent to Flask API over Wi-Fi. Code for the application will be provided to the client at project's completion.

Wi-Fi Module Controlled LED Strips: LED strips will be attached to the cabinets in the ETG shop vertically and horizontally to align with an X-Y coordinate grid. A Wi-Fi enabled microcontroller will receive X-Y coordinates from the Flask API corresponding to the position of the part being searched for. The controller will then light up the LEDs within the strips corresponding to the X-Y coordinates. Code run on the microcontroller will be provided to the client at project's completion.

Project Documentation: Due to COVID-19 we will be providing in depth instructions for the client. These documents will include a parts list, setup instructions, usage instructions, testing instructions, and maintenance instructions. Along with these instructions, we will provide the design documentation to the client at project's completion.

2. Specifications and Analysis

2.1 Proposed Design

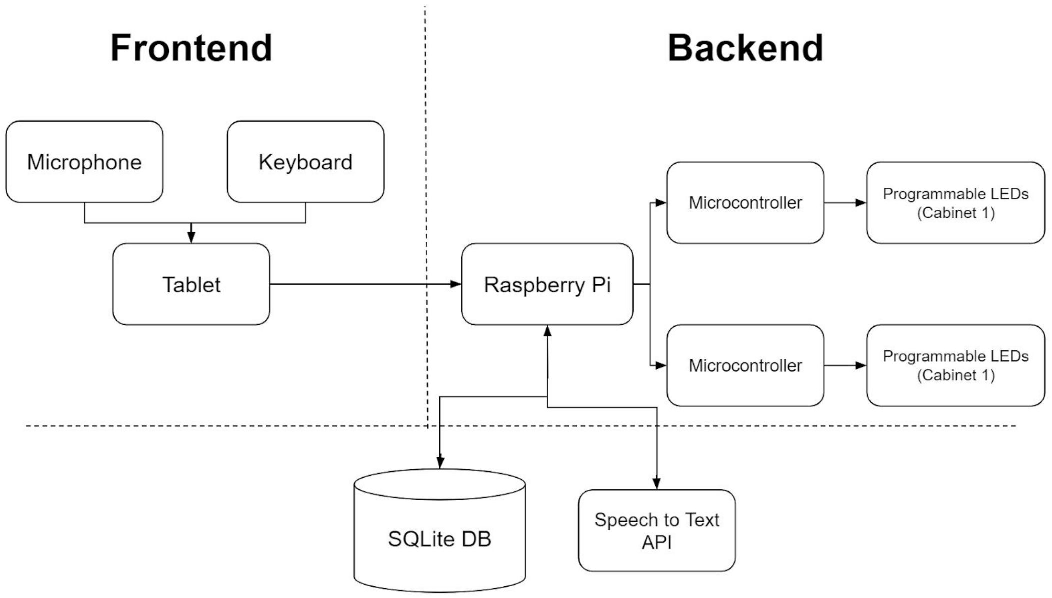


Figure 1: A Block Diagram of our proposed solution

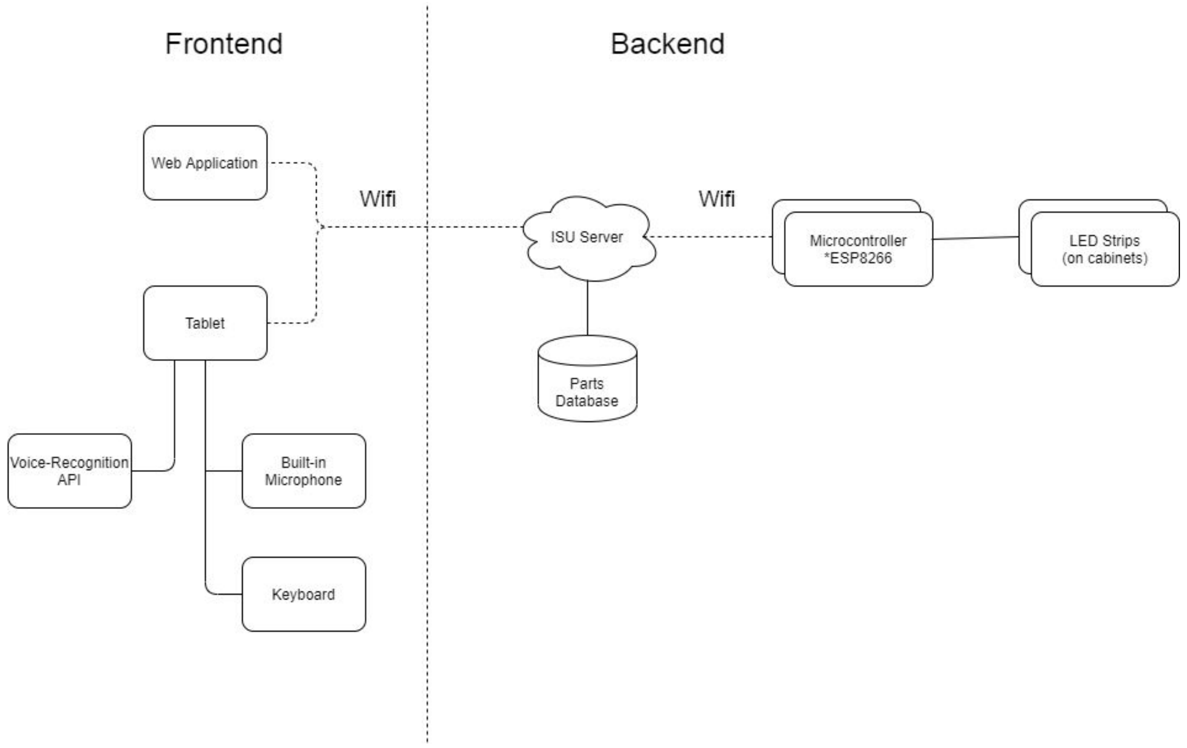


Figure 2: Block Diagram of final solution

2.1.1 User Interface

Our proposed design is to use a tablet running an android application as a user interface. For now, we intend to create a simple android application to fill this purpose, with more focus on refactoring and improving the look of the application later on in the project. We are also considering connecting a keyboard and mouse to the tablet to help make it easier to use, especially when entering considerable amounts of data.

2.1.2 Server, API, Database, and LEDs.

The core of our project revolves around a server that will host an API, host a database, communicate with an external speech to Text API, and communicate with microcontrollers to control the LEDs. We intend to build a simple API using flask, with endpoints that when hit by the frontend tablet will allow the user to find a part by text, find a part by speech, or add a part to the database. If finding a part by speech, we plan to experiment using existing python libraries and external APIs to see what gives us the most consistent results converting speech to text. From there, all three main functionalities will behave similarly by querying a database that will be hosted on an ISU server. Once our queries are complete, we can use a UDP connection to directly send the X, Y coordinates and desired colors to the LEDs via NodeMCU ESP8266 microcontrollers. These base API cases can then be easily expanded to support options like adjusting brightness, changing colors, more complex user commands, and searching for multiple parts.

2.2 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.

2.3 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.

2.4 Design Plan

Our design plan for the fall semester centered around establishing a base system. Our dependencies all centered around securing the required hardware for our project: the Raspberry Pi, the Nexus tablet, microcontrollers and LEDs. The spring semester we focused on removing the Raspberry Pi from the design and implementing the server. The final change to the design plan was to finalize a prototype for the presentation as well as create sufficient documentation to provide to the client so they may implement it without us.

3. Statement of Work

3.1 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.

3.2 Technology Considerations

3.2.1 User Interface

One key design choice we made was the inclusion of a Nexus tablet in the system compared to getting a touch screen for the Raspberry Pi. While the latter would have helped to eliminate some complexity in our design, based on our use case we decided that the ability to transport the user input system within the ETG was a valuable feature.

3.2.2 LEDs

Another design choice was the use of microcontrollers and strips of programmable LEDs or the use of individual, radio controlled LEDs. While the idea of individual leds is interesting, ultimately the issues of possible interference, reduced extendability, and large increase in cost of the project pointed us back toward our current design.

3.2.3 Database

When considering what database system to use, we considered several options including cloud-based databases like AWS, using a university-hosted server, or hosting it on the Raspberry Pi. We eventually decided to use a university-hosted server.

3.2.4 API

For the choice of what technology to use to develop our API, we decided to use Flask. While we considered using technologies like NodeJs, Java, or Django, we wanted something lightweight that would still fulfill our limited purposes. We had also used some preexisting python both for the initial testing of the Microcontrollers/LEDs and the initial testing of the database. Based on this, we decided that despite our teams limited python knowledge, it would be best to use Flask for our development.

3.3 Task Decomposition

Our project is divided into sections based on the hardware component along with other phases of the software development process. We also include a section for the setup of the hardware and software environment allowing us to have a foundation for the following sections and spend time determining the technologies we want to use.

- 3.3.1. Documentation
  - 3.3.1.1. Design Doc V1
  - 3.3.1.2. Design Doc V2
  - 3.3.1.3. Design Doc V3
  - 3.3.1.4. Design Doc V4
  - 3.3.1.5. Design Doc V5
  - 3.3.1.6. Design Review Presentation
  - 3.3.1.7. Parts List
  - 3.3.1.8. Setup Instructions
  - 3.3.1.9. User Instructions
  - 3.3.1.10. Testing Instructions
  - 3.3.1.11. Maintenance Instructions
  - 3.3.1.12. Final Presentation
- 3.3.2. Requirements
  - 3.3.2.1. Functional requirements
  - 3.3.2.2. Non-Functional requirements
- 3.3.3. Environment
  - 3.3.3.1. Hardware setup

3.3.3.2. Software setup

3.3.4. University Hosted Server

3.3.4.1. Communicate with LED strips via Wi-Fi

3.3.4.2. Receive input via Wi-Fi

3.3.4.3. Interpret voice input

3.3.4.4. Test database with dummy data

3.3.4.5. Fill database with real data

3.3.4.6. Database change and expansion functionality

3.3.5. LED Strips and ESP8266

3.3.5.1. Connect the ESP8266 to Wi-Fi

3.3.5.2. Receive commands on ESP8266 over Wi-Fi

3.3.5.3. Address individual LEDs using input

3.3.5.4. Implement special LED functions

3.3.6. Tablet Application

3.3.6.1. Basic functionality

3.3.6.2. Construct user-friendly UI

3.3.7. Testing

3.3.7.1. Test individual components

3.3.7.2. Test inter-system operations

3.3.7.3. System prototyping

3.3.8. Implementation

3.3.8.1. Prototype Implementations

3.3.8.2. Provide Users with documentation

3.4 Possible Risks And Risk Management

The largest risk in our project is that we will be unable to fully implement the database portion of the project. While there is some existing form of data for many of the parts, it is likely a majority of the parts in the ETG currently will have to be manually entered by team members. This will take a substantial amount of time and leaves lots of room for human error in the data entry.

The move to remote schooling due to COVID-19 has also forced us to change our deliverables. We dealt with this by discussing how we could provide the closest deliverables to our original project, and decided that the best approach was to additional instructions for the setup and testing of the system.

3.5 Project Proposed Milestones and Evaluation Criteria

Key milestones for the first semester were system connectivity and communication, basic typed and voice search, and tablet integration. Testing in this phase will be less intensive as we focus primarily on getting a functional product, we will focus on testing communication between the hardware, hardware functionality, and basic search functions.

Key milestones for the second semester were database completion, an improved user interface, scalability testing, and changing our deliverables. In this phase there was a much heavier focus on testing, especially with the user interface and search functionality.

3.6 Project Tracking Procedures

We will track progress on our project through weekly to bi-weekly reports that document work down and met and unmet goals. Current goals are posted on a Trello board where we can assign tasks. We have weekly meetings to stay on top of the project and to make sure that if an aspect of the project is behind schedule the whole team knows about it and can respond accordingly by shifting resources.

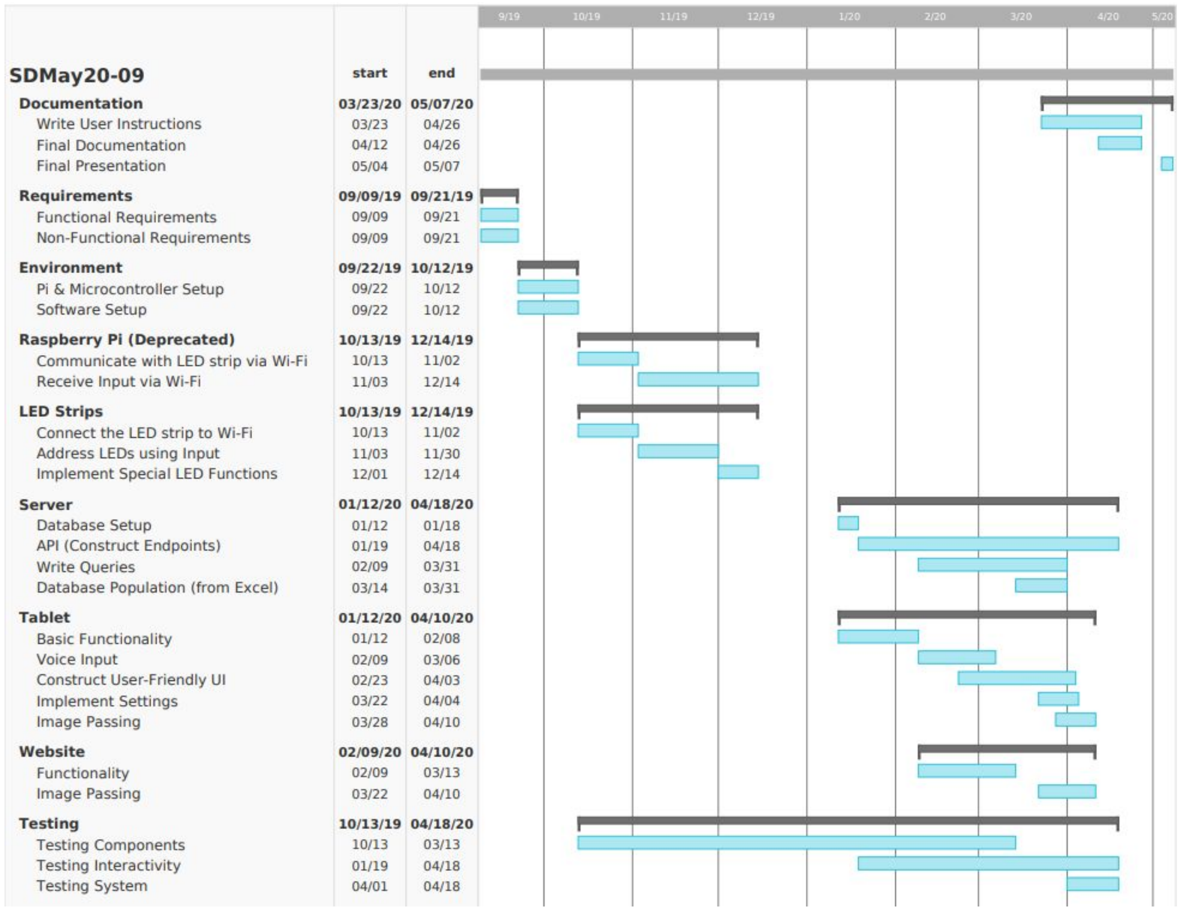
3.7 Expected Results and Validation

The desired outcome of this project is an inventory locator assistant for the ETG shop to help individuals find parts within the shop that may be unfamiliar with its organization. To achieve this goal we had a functional prototype of the system completed by the end of the semester (Dec 6th 2019), that exhibits basic functionality: receiving requests through voice or typed search, processing the search, and sending out signals to the appropriate controller. Development of the system continued in the second semester with a large focus on testing, user interface design, database construction, and scaling. We have set our schedule to allow ample time for testing so that any issues may be found before the products final release.

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

Our project has a time frame over the Fall 2019 and Spring 2020 semesters at ISU with our timeline starting on September 9th, 2019 and ending May 1st, 2020. Our timeline has been constructed as the following gantt chart. It separates the project into sections based on hardware components and phases of software development.



SDMay20-09	start	end
Documentation	**03/23/20**	**05/07/20**
Write User Instructions	03/23	04/26
Final Documentation	04/12	04/26
Final Presentation	05/04	05/07
Requirements	**09/09/19**	**09/21/19**
Functional Requirements	09/09	09/21
Non-Functional Requirements	09/09	09/21
Environment	**09/22/19**	**10/12/19**
Pi & Microcontroller Setup	09/22	10/12
Software Setup	09/22	10/12
Raspberry Pi (Deprecated)	**10/13/19**	**12/14/19**
Communicate with LED strip via Wi-Fi	10/13	11/02
Receive Input via Wi-Fi	11/03	12/14
LED Strips	**10/13/19**	**12/14/19**
Connect the LED strip to Wi-Fi	10/13	11/02
Address LEDs using Input	11/03	11/30
Implement Special LED Functions	12/01	12/14
Server	**01/12/20**	**04/18/20**
Database Setup	01/12	01/18
API (Construct Endpoints)	01/19	04/18
Write Queries	02/09	03/31
Database Population (from Excel)	03/14	03/31
Tablet	**01/12/20**	**04/10/20**
Basic Functionality	01/12	02/08
Voice Input	02/09	03/06
Construct User-Friendly UI	02/23	04/03
Implement Settings	03/22	04/04
Image Passing	03/28	04/10
Website	**02/09/20**	**04/10/20**
Functionality	02/09	03/13
Image Passing	03/22	04/10
Testing	**10/13/19**	**04/18/20**
Testing Components	10/13	03/13
Testing Interactivity	01/19	04/18
Testing System	04/01	04/18

Figure 3: Gantt Chart

A group of our size allowed for multiple components to be developed simultaneously meaning the tablet application, database, and microcontrollers can be worked on all at the same time without slowing each other down. This also gave us the ability to devote resources where they are needed the most at that stage in development, whether it be constructing new functionalities or testing.

We gave the development of the Android application and voice input feature a longer development time because members of our group are unfamiliar with the technology and expect many issues to come up. More time means that our voice input system will be effective and fully developed as our client would like verbal commands to be the primary method of user input.

With continued testing throughout the development of individual components, we expect for the testing of the combined system to be performed quickly and result in short implementation time. This allows for more time to be devoted to the development of the components.

We plan on sticking to this timeline in order to continuously make progress and manage our time effectively. This will help keep people on track and allow us to see if we are on time to finish our project before the end of the allocated time frame.

4.2 Feasibility Assessment

At the end of this course, this project should be able to input both voice and typed commands to find parts in the cabinets of the ETG. It will light up the designated area to alert the user of the position of the requested part. It will be able to add onto the database and look up the last search. As previously stated, the largest foreseen issue is that of data entry of existing parts. We are also quite likely to run into issues with several other parts of the stack based on our teams limited knowledge of the technologies that we use.

4.3 Personnel Effort Requirements

Task	How	Estimated Time
1. Design Document	Write an initial document and implement changes as they occur in order to have an updated record of what we expect and have done.	14 hours
2. Hardware Setup	Ensure all hardware is working and is able to connect to each other.	20 hours
3. Database Creation and Data Entry	Create a database based on lists of parts given to us by Mr. Harker that will be adjustable for the future.	40 hours
4. Raspberry Pi Implementation	Connect the Raspberry Pi to all components and make sure it can communicate with each other..	25 hours
5. LED connection	Ensure the LEDs are working and will receive commands from the Pi.	8 hours
6. Testing	Create a test environment before final deployment so we can ensure the project is working as planned. Debug and redesign as needed.	50 hours

Table 1: Estimated hours needed

4.4 Other Resource Requirements

This project will require a set of programmable LEDs, multiple microcontrollers, a power system, and a tablet. Additionally, we will require access to the ETG shop to take an inventory of parts for data entry and for a final installation of the system.

4.5 Financial Requirements

We have been given a budget of $500 for the entirety of the project. This will go to buying hardware such as LEDs and controllers as well as other physical parts such as adhesives. All of our purchases will be made through Mr. Leland Harker.

5. Testing and Implementation

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 the accuracy of the input

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 individual part.

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:

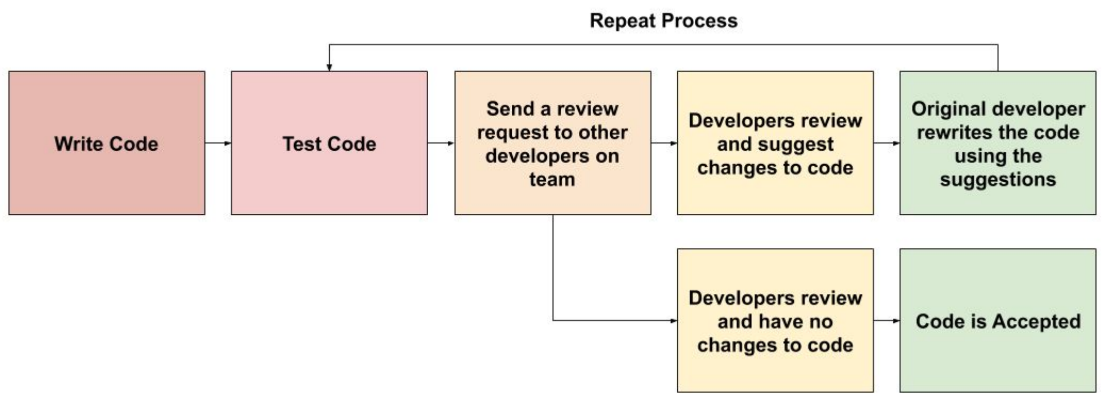
- 5.4.1 The product shall allow for the expansion of the system onto multiple cabinets
  - 5.4.1.1 Test how much we can expand the system by load testing the connections
  - 5.4.1.2 Verify that we can expand the system easily
- 5.4.2 The database used in the product must be able to be used outside of 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 4: 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

5.6 Results

During our initial research, we decided having more components in our system communicating via Wi-Fi instead of physical wires would let our project be expandable with the new devices over a larger area. We can do this by registering the devices with the Iowa State network, an option allowed to us since the project will be used on the Iowa State University grounds. Currently we are looking into multiple voice interpretation softwares and services to decide which one will be the most effective for our project through accuracy and speed of the interpretation process. By the end of the project we hope to have a stable product without any breaking defects. By testing the validity we can be sure that the client receives the product they expect.

6. Closing Material

6.1 Conclusion

For this project, we used a university-hosted server to host a database and use a voice-recognition API to locate parts. In the full product, a tablet would receive input either through voice or text. Using a voice-recognition API (as needed), the database on the server would be searched and information would be sent to the application on the tablet and to the microcontrollers to control the LEDs. The LEDs will help lead the user to where the part is.

6.2 References

- Voice controlled organizer

  Instructables, "FindyBot3000 - a Voice Controlled Organizer," *Instructables*, 20-Jun-2019. [Online]. Available: https://www.instructables.com/id/FindyBot3000-a-Voice-Controlled-Organizer/. [Accessed: 09-Dec-2019].

- AIY Voice Kit Instructions

  "Voice," *Google*. [Online]. Available: https://aiyprojects.withgoogle.com/voice. [Accessed: 09-Dec-2019].

- Initial scoping of microprocessors

  S. Santos, Shashi, Thomas, Miloš, A. M., and Pipi, "ESP32/ESP8266 RGB LED Strip with Color Picker Web Server," *Random Nerd Tutorials*, 16-Apr-2019. [Online]. Available: https://randomnerdtutorials.com/esp32-esp8266-rgb-led-strip-web-server/. [Accessed: 09-Dec-2019].

- Creating a LAN for our microprocessors to talk to the central pi

  S. Lovely, "How to use your Raspberry Pi as a wireless access point," *The Pi*, 19-Sep-2018. [Online]. Available:

https://thepi.io/how-to-use-your-raspberry-pi-as-a-wireless-access-point/. [Accessed: 09-Dec-2019].

- FastLED Arduino library

FastLED, “FastLED/FastLED,” *GitHub*, 26-Aug-2019. [Online]. Available: https://github.com/FastLED/FastLED. [Accessed: 09-Dec-2019].