University of Ottawa

GNG 2101 B03: Intro to Product Development and Management for Engineers

Deliverable C: Conceptual Design, Project Plan, and Feasibility Study

Submitted by

LIB CARTE, Team 33

Aymane Aziz, 300193353

Kevin Tu, 300188166

Julio Midence, 300170486

Aidan Billou, 300199731

Joon Lee, 300184520

Mana Azarm

David Londono

September 30, 2021

University of Ottawa

Abstract

This document outlines the ideation of the subcomponents required gathered from the functional decomposition for the library wayfinding app and the final plan which will be used to base future project plans.

List Of Figures

List Of Tables

Table Of Contents

1.0 INTRODUCTION

This document outlines the subsystems of the library wayfinding project, potential design concepts, analysis and evaluation of various solutions. The problem the application is working on solving is creating an accessible program that will guide users, with and without disabilities, in an accurate and simple to follow way. The product will be in the form of an application that requires no assistance to operate. After evaluating the proposed solutions, a finalized design will be presented. This design will be based on the customers needs gathered from the previous deliverable. These needs include accessibility, accuracy, usability, and affordability just to list a few.

Two main sections have been identified for this project, hardware and software. These sections will be broken down into smaller components for discussion in terms of possible solutions and analysis. For the smaller components that could be improved on, concepts were generated and based on the analysis, the final concept was selected and planned out for prototyping.

2.0 FUNCTIONAL DECOMPOSITION

Based on the problem statement and customers' needs gathered, the functional requirements of the wayfinding project were formed and broken into two major sections in hardware and software. The software focuses on the application part of the project while the hardware focuses on the beacon and microcontroller.

The table below shows the clients need from Deliverable B

Table 1. Client needs

#	Need	Design Criteria	Importance (1<5)	Functional , non functional or constraint
1	The app is accessible to users with vision impairments	Audio / sensory cues	4	Functional
2	The beacons can operate without frequent charging	Larger / more efficient batteries	3	Constraint
3	Accessible to non english speaking users	Bilingual	2	Functional
4	The application is design	Usability	5	Functional

	intuitive			
5	The product is compatible with both android and ios	Compatibility	3	Functional
6	The product is affordable	Minimum cost	4	Constraint
7	The app is accurate (Location-wise)	Accuracy / User safety	5	Functional
8	The beacons are portable	Portability / Size	3	Constraint

The following list is the decomposition of the project into two major sections: Hardware and software.

2.1 Hardware

- Beacon system
  - Placement
- Power and Power management system
  - Battery
  - Voltage regulation
- Microcontroller
  - Communication between arduino and app
  - Communication between arduino and beacon

2.2 Software

- UI/UX
  - Auditory and visual components
  - Program logic (Program interactions)
  - Language support
- Database
  - Admin access login information
- Algorithm
  - Pathfinding
  - Location tracking
- Communication
  - Arduino communication
  - Database communication

The hardware focuses on the accuracy, cost, size, and the battery life aspects of constraints and functional requirements which was outlined in Deliverable B. Having hardware components that meet our targets is necessary for the application to provide accurate path tracing as low quality beacons may not provide the resolution required and a beacon running out of power mid trip would significantly reduce accuracy.

The software component addresses the user friendliness, accessibility, accuracy, language support, and phone compatibility aspect of the constraint and functional requirements. First, the UI works to provide solutions for user friendliness, accessibility for individuals with disabilities, and language support for both french and english users. Next, the database connects with the UI to provide additional functionality for the user such as posting announcements which our client requested. Lastly, the algorithm and communication completes the accuracy requirement for the program. The pathfinding algorithm must convert current location data of the user received from the arduino to create a precise path for the user to follow. Without proper communication between the application and a good algorithm that can calculate the optimal path, the functional requirement of accuracy cannot be achieved.

A use case diagram was created to visualize and communicate the main functionalities that each user will be interacting with when using the application. The user of this app must be able to view the announcements, use the navigation, sign in if they have an account, enable the text to speech for the visually impaired, and toggled between english and french for non-english users and vice versa. The use case diagram below summarizes the key interactions of the program with the actors of the application.

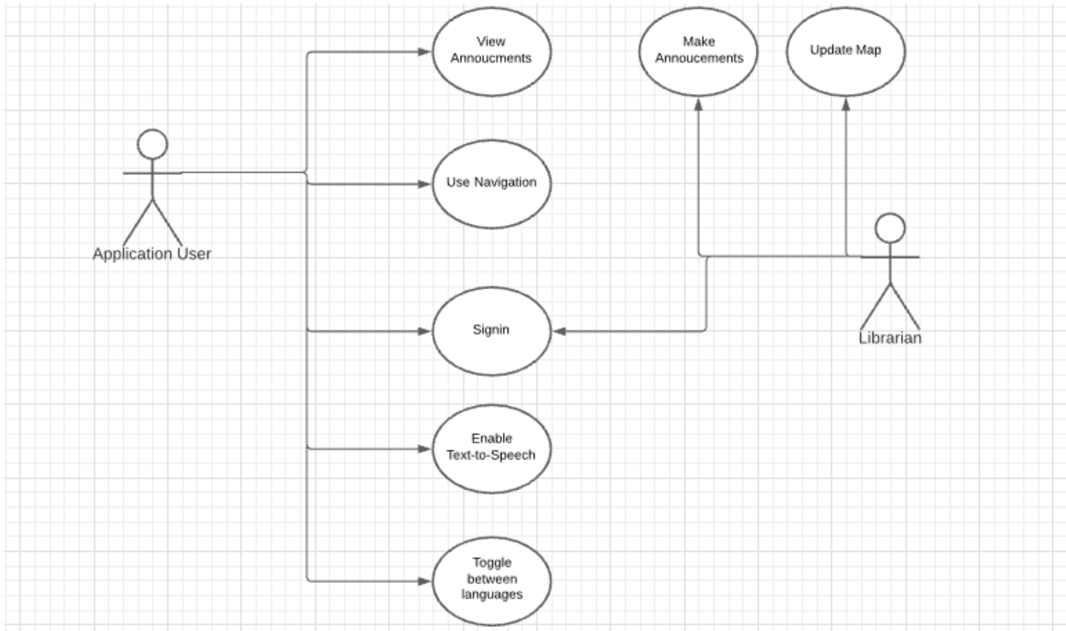

Figure 1. App functionality use case diagram

3.0 CONCEPTUAL DESIGNS

Each team member was assigned one subtask to ideate solutions. These conceptual designs were discussed as a team in order to come up with a final solution.

3.1 Aymane's Concepts (Beacons and arduino)

3.1.1 Microcontrollers (Arduino) per beacon

As mentioned in the target specifications in deliverable B, the final product should have a location tracking technology. This technology can be implemented by using bluetooth beacons controlled by microcontrollers[3]. There are two possible solutions, using one microcontroller for each beacon or one central microcontroller for all of them.

3.1.2 One microcontroller for each beacon

This figure shows every beacon connected to a microcontroller on the shelves.

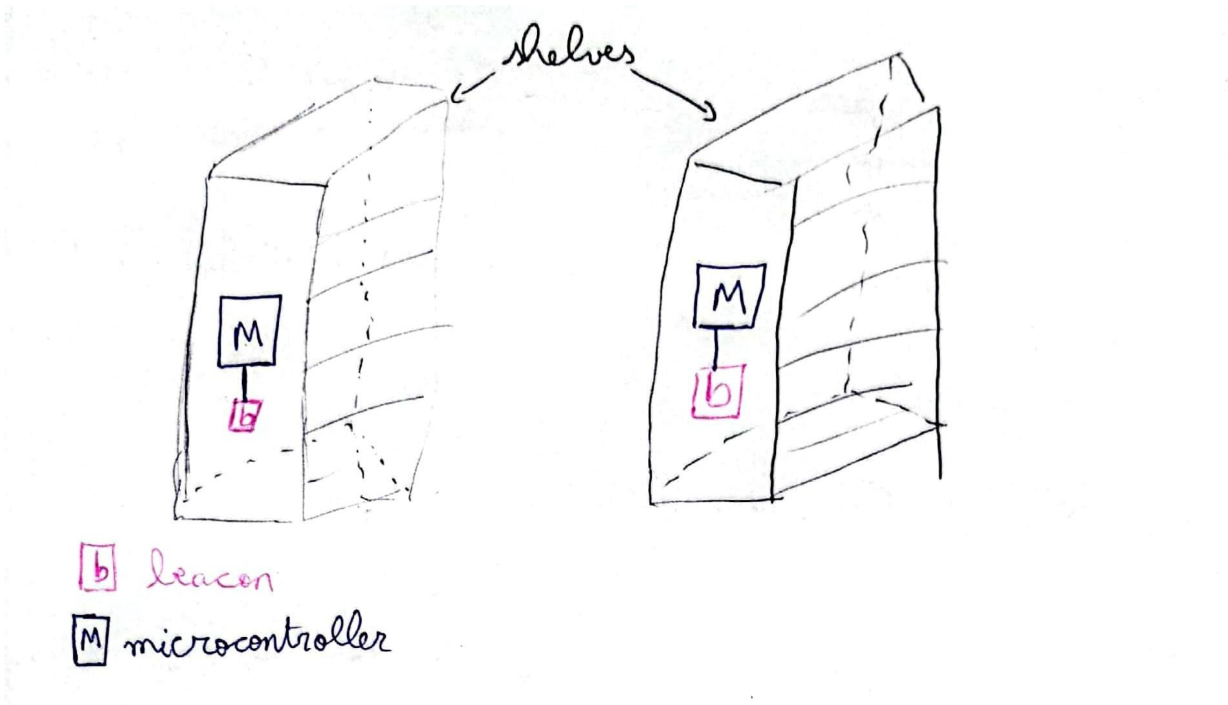

Figure 2. Beacon concept 1

Pros:

Using one microcontroller for each beacon is easy to implement, because the same code will be used for all of them.

Cons:

Using one microcontroller for each beacon takes a lot of space.

Using one microcontroller for each beacon is expensive because a lot of Arduinos are needed.

3.1.2. One microcontroller for all the beacons

This figure shows one microcontroller connected to different beacons on their respective shelves.

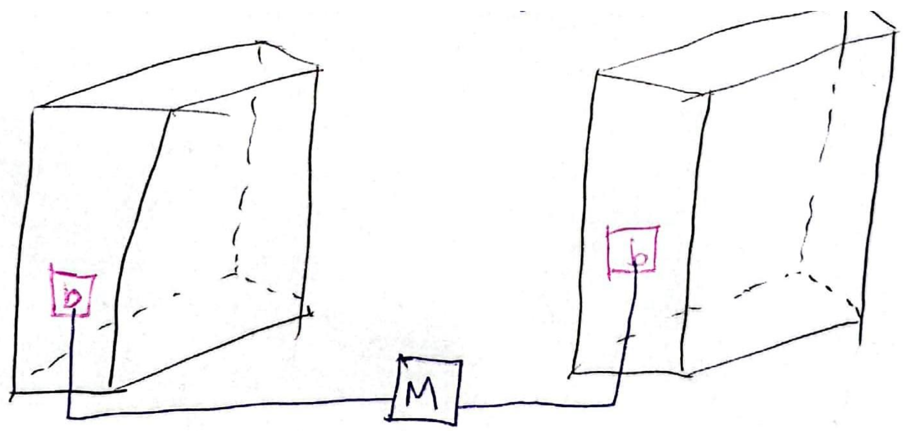

Figure 3. Beacon concept 2

Pros:

Using one microcontroller for all the beacons is cheaper because only one Arduino is needed.

Using one microcontroller for all the beacons takes a negligible space.

Cons:

Using one microcontroller for all the beacons is hard to implement because the Arduino has to collect data from many beacons.

3.1.3. Source of energy for the microcontroller

In order to function, the microcontroller needs to be powered either by a battery that needs to be changed regularly or recharged, or by being directly plugged into an electrical outlet.

3.1.4. The use of a battery

This figure shows the use of a battery to aliment a microcontroller that is connected to a beacon.

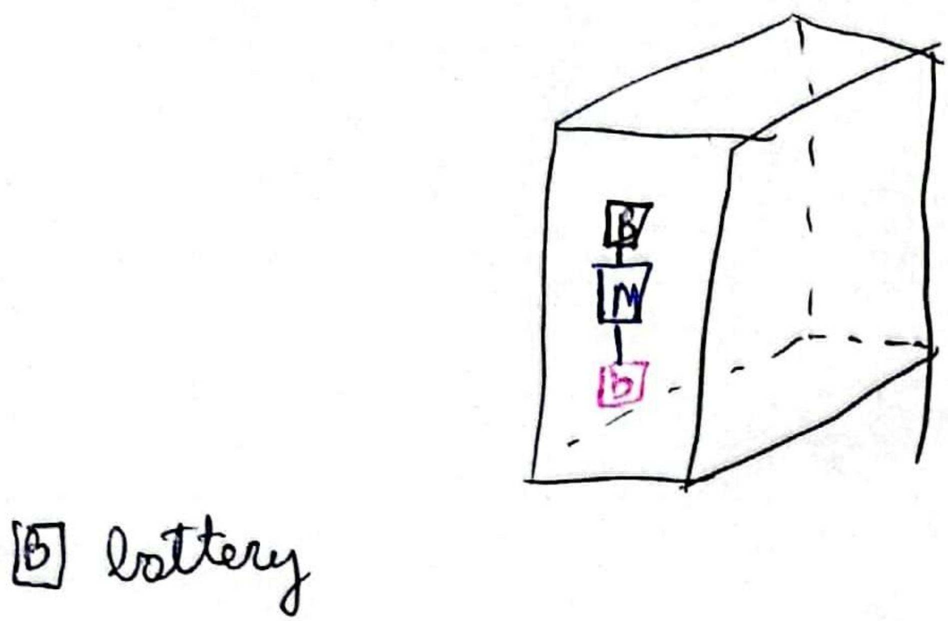

Figure 4. Battery concept 1

Pros:

By using a battery to power the microcontroller,it can be put anywhere because the battery is portable.

Cons:

A battery needs to be changed from time to time or recharged so it can add additional costs.

3.1.5. Plugging-in the microcontroller.

This figure shows a microcontroller plugged into an electrical outlet and connected to a beacon.

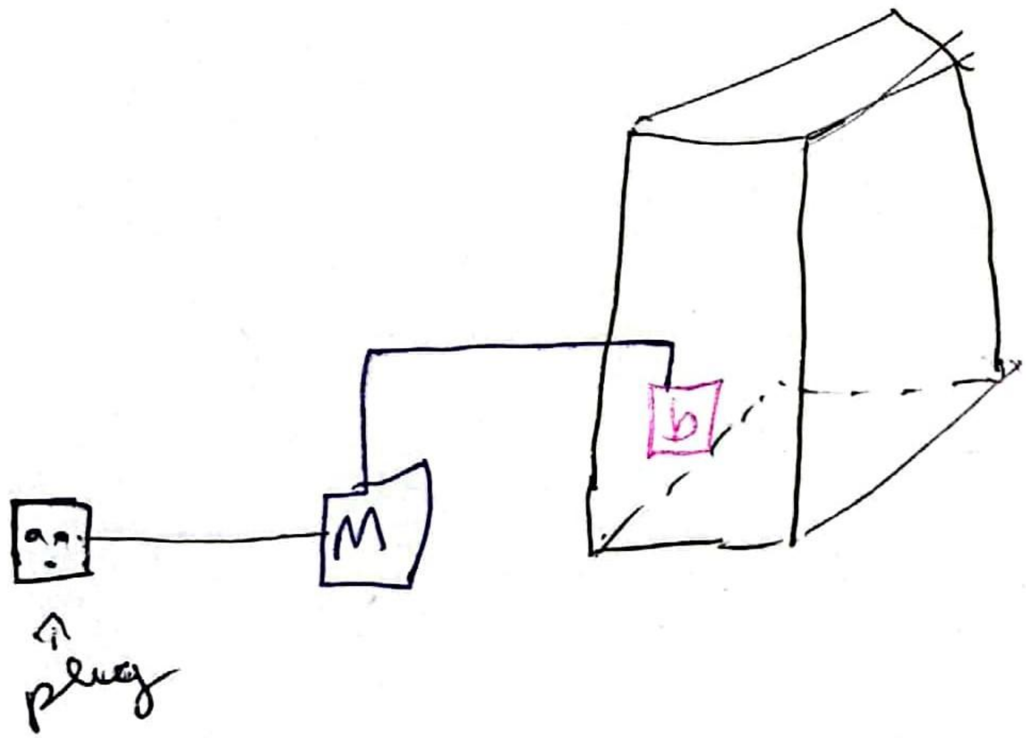

Figure 5. Battery concept 2

Pros:

Plugging-in the microcontroller reduces the expenses because no battery will be purchased, only local outlets are used.

Cons :

A downside to this concept is that the microcontroller needs to be placed next to a plug, which restrains the areas where it can be installed.

3.1.6. Conclusion

These concepts above meet the client needs and target specifications declared in deliverable B, by using one microcontroller for all the beacons expenses are reduced which makes the final product affordable, that's the sixth need in Table 2 of the deliverable. By using a battery to power the Arduino, the system becomes portable which meets the eighth need of the same table and the design specification about extended battery life in Table 7.

3.2 Kevin's Concepts (Navigation)

3.2.1. Visual compass navigator

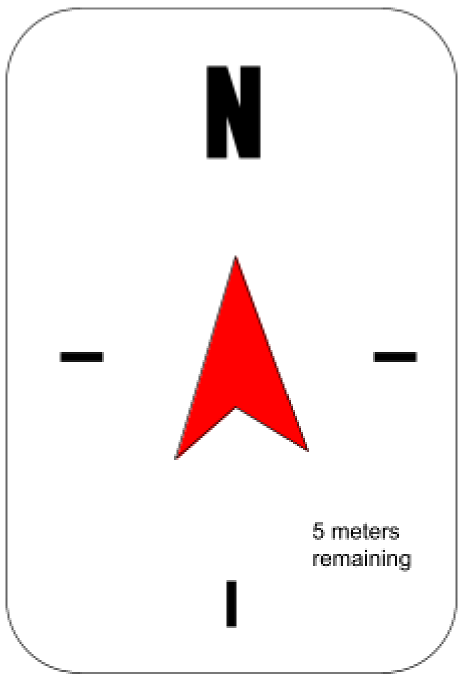

Figure 6. Navigation concept

Once the user's destination is determined, the compass will continue to point in the direction using the magnetometer to determine the direction you're facing and the beacons around the library to determine the user's location. The user can then follow the arrow until they have arrived at the book they want. The user will also be able to view the distance remaining.

3.2.2. Vibration Navigator

To assist those who are impared, the vibration navigator will use the magnetometer in the device to determine the direction the user is facing as well as the beacons that are placed around the library to determine the user’s location. Then, constantly vibrated the closer the user gets to their destination. As the user arrives closer and closer, the frequency of the vibrations will slowly increase similar to that of a metal detector where the feedback you receive becomes stronger as you approach the location. The frequency of the vibrations will also increase as the user faces the right orientation which notifies the user that they are on the right track. Once the user arrives at their destination, they will receive auditory feedback notifying them that they have arrived.

3.2.3. Auditory Navigator

The auditory navigator is similar to that of the vibration navigator which will use both the magnetometer in the device and the beacons around the library to locate the user and guide them by using the auditory feedback emitting from the device. The closer the user gets to the desired location, the stronger the audio feedback will be. A text to speech will also be available which will tell the user the direction to go.

3.3 Joon’s Concepts (Algorithm)

The algorithm in the program refers to the pathfinding algorithm for user navigation. This relates to the client's need for an accurate and intuitive application as described in Deliverable B. The pathfinding algorithm accounts for the other half of the accuracy functional requirement since even an accurate location of the user will not be useful if the algorithm guides the user to the wrong location. Accuracy can also be a safety hazard for the user. Additionally, a slow processing time and precision would give an impression of an unreliability in the program, removing the easy to use, intuitive experience that was created from the simplistic UI.

3.3.1. A* algorithm

The A* algorithm works by taking the distance from the starting node and the distance from the end node, adding them up to get a combined value which is put in a priorityqueue to order them by the lowest value to find the path with the least distance [1].

The figure below shows an example of A* algorithm in a grid map layout

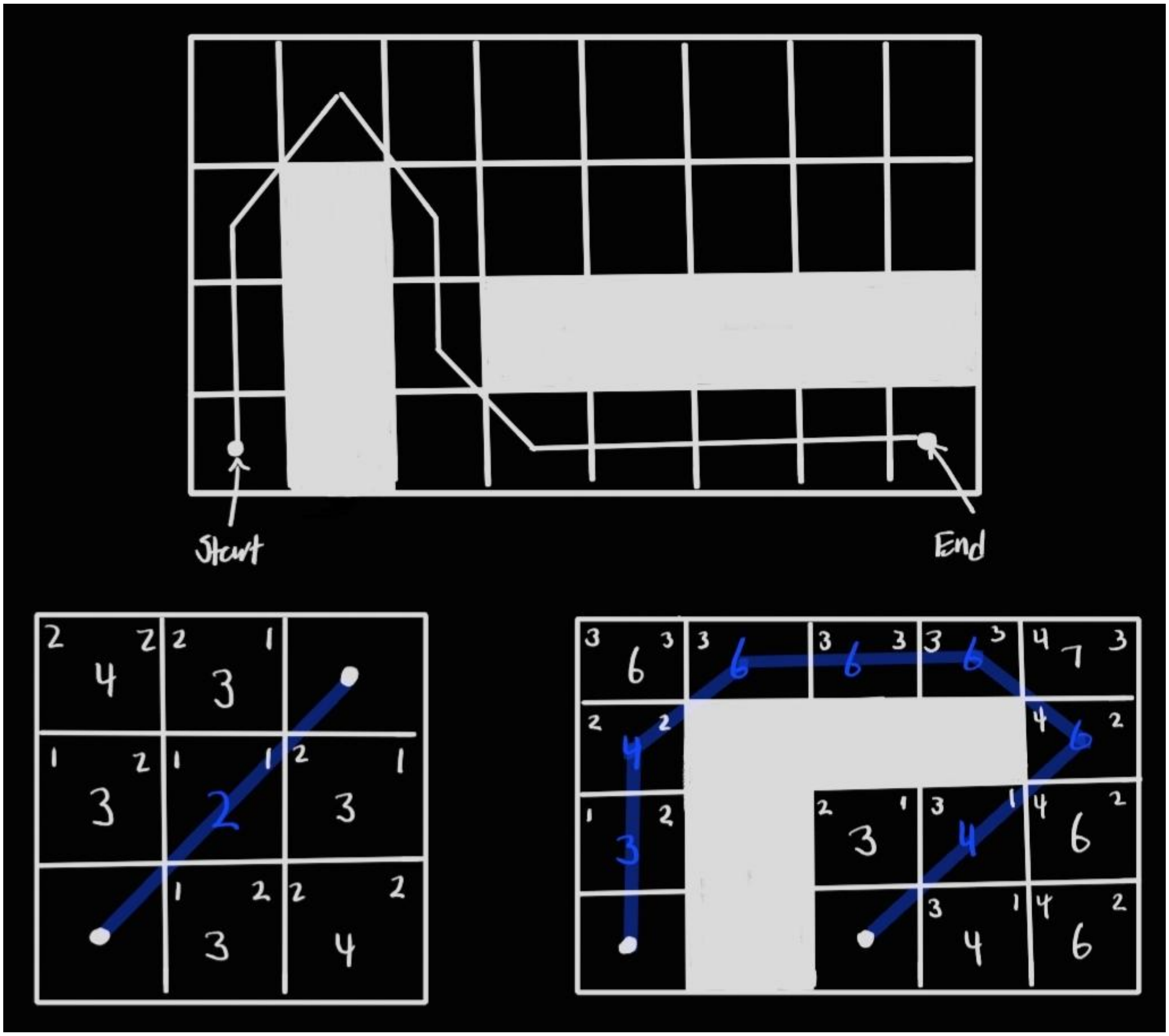

Figure 7. A algorithm map*

The numbers on the top left of the boxes represent the distance from the starting point while the numbers on the top right represent the distance from the end point. This is combined to get the sum which is the center value in each tile.

3.3.2. RRT*(Rapidly exploring Random Tree)

The RRT* works by picking a point on the map at random and having the nearest node attempt to reach for that point within a limited range [2]. When a node is added, it optimizes its path and removes zig-zag like patterns [2]. This is repeated many times and a branching effect is observed. This is able to quickly explore various paths with a lot of obstacles and with enough nodes, the end point can be reached with an mostly optimal path [2].

The figure below shows an example of what RRT* looks like. The white dots represent the nodes with the lines representing the paths.

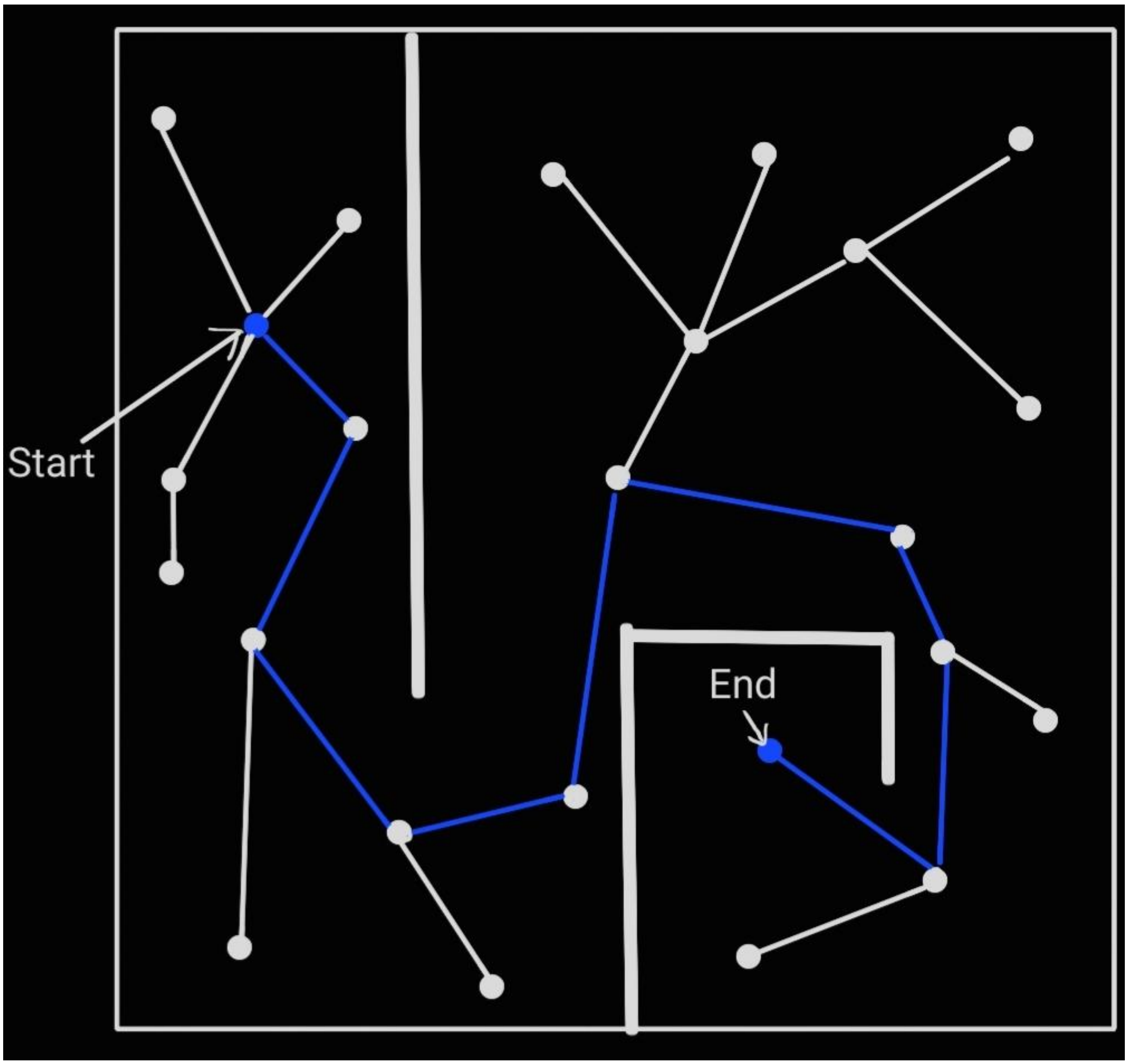

Figure 8. RRT algorithm map*

3.3.3. Manually Planning the Paths

The points of interest are connected manually by a person. This method is time consuming but very simple to implement. The figure below shows the manual connection between the nodes.

The figure below visualizes what manual plotting could look like. The white dots represent the points of interest and the lines, the preplanned paths to those points.

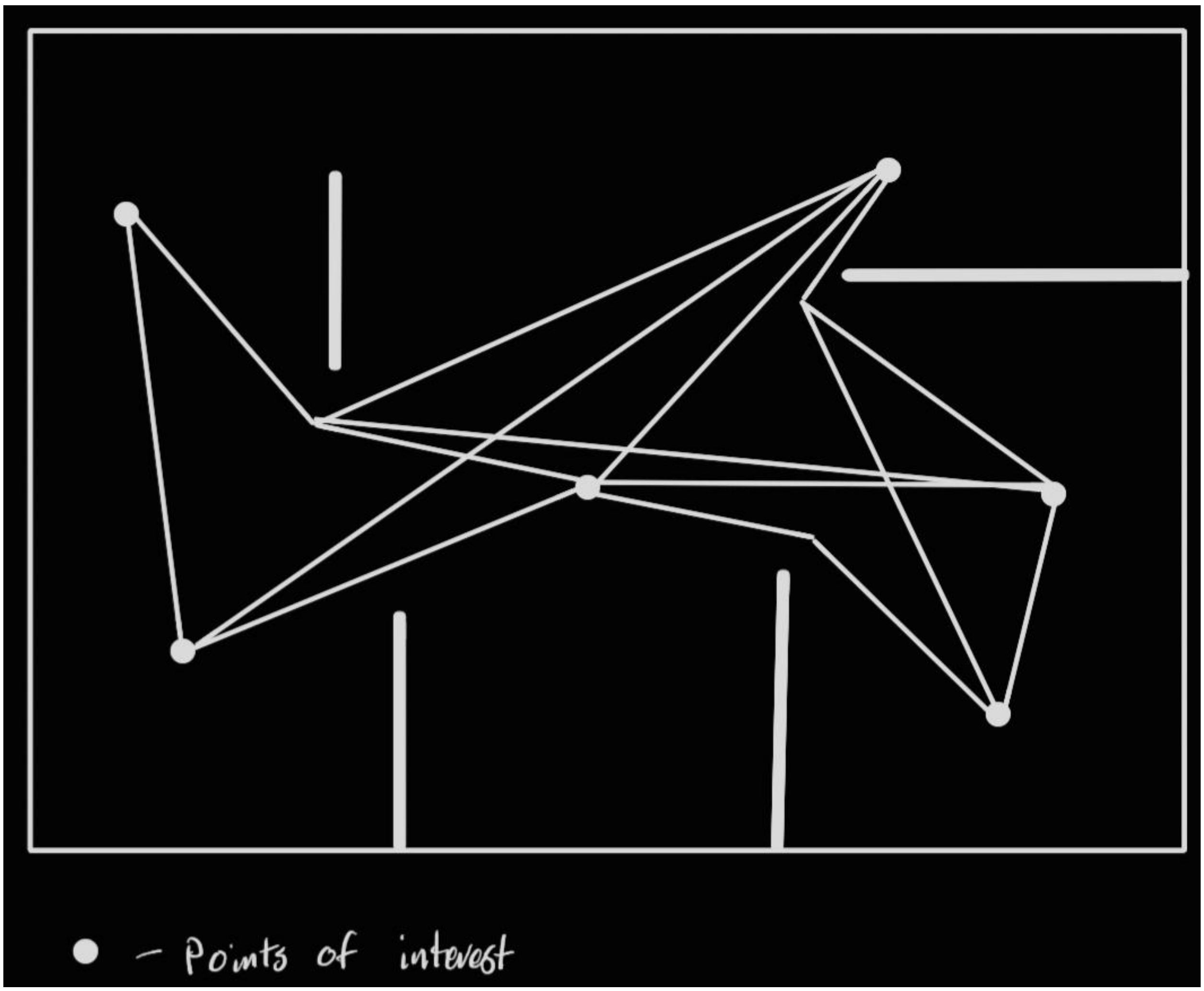

Figure 9. Manual plot map

Every point of interest in the library can be mapped manually to each other point so that a person could go anywhere from point A to point B. However, this becomes very difficult very fast as the possibilities grow by n(n-1) where n is the number of points of interest. This method is not able to dynamically change so if there are changes to the library layout, all the affected paths must be manually readjusted.

3.4 Julio's Concepts

The figure below shows the different concepts created in order to power the arduino(s) and the beacons. The first concept focuses on the generation of power. This would be a renewable way to implement a power management system. Achieving renewable energy would be a difficult task which is where the second concept comes into play. Replacing the voltage regulator of the Arduino with a DC-DC buck converter would reduce the energy consumption. The safest and most simple way to properly run the arduino is to plug it into a wall outlet. The beacons can either be powered by an internal battery or be connected to a rechargeable battery station.

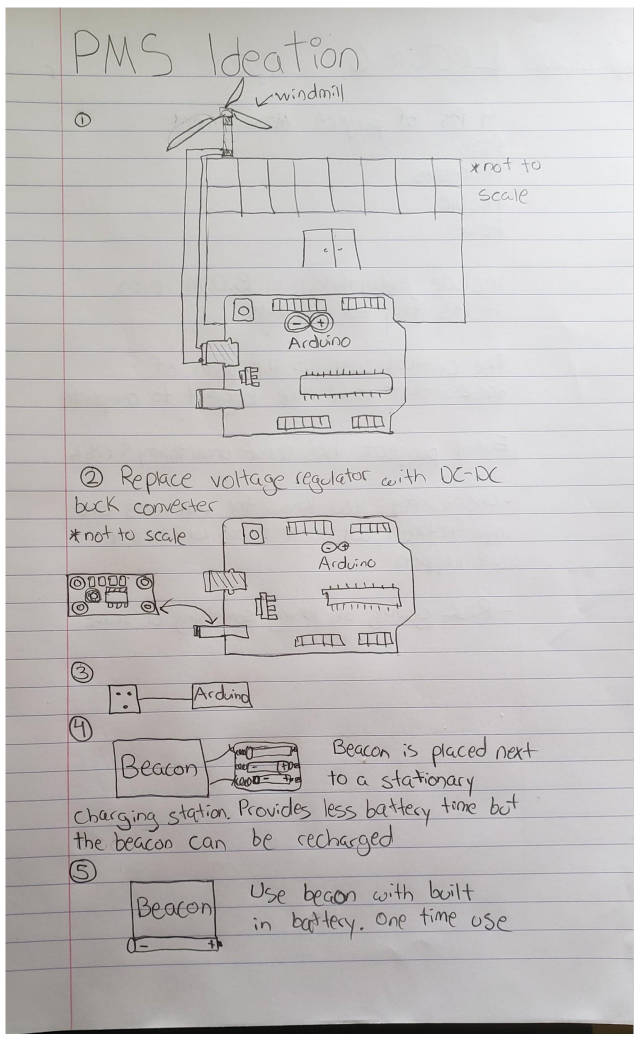

Figure 10. Power system concepts

3.5 Aidan's Concepts (UI / UX)

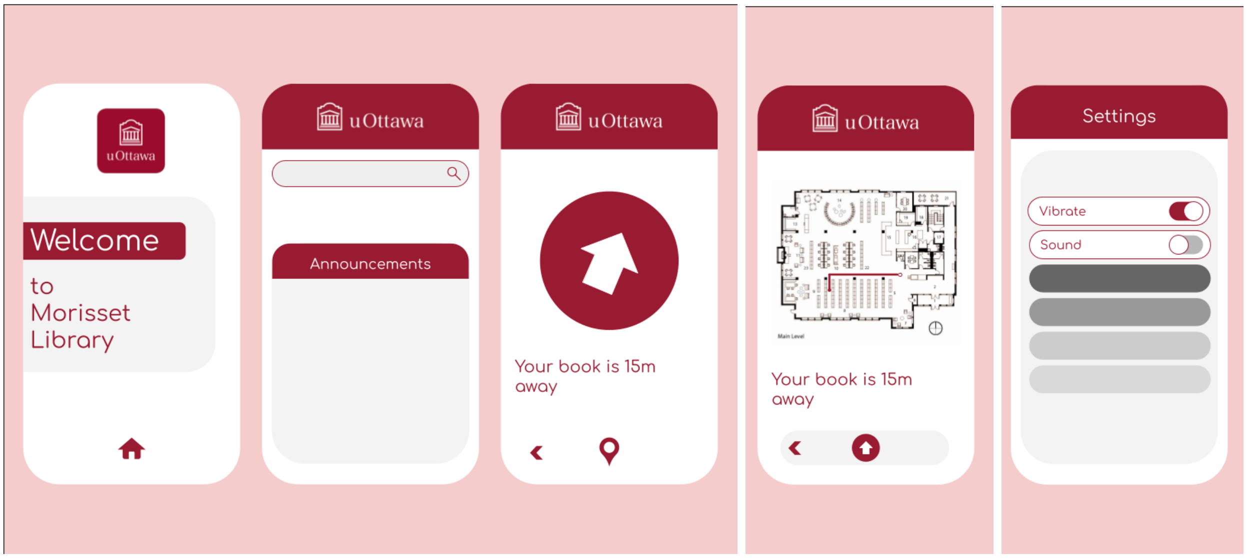

Figure 11. UI layout concepts

The UI that has been created with both ease of use and user friendliness in mind. The UI was made with the least amount of screens the user has to cycle through to reach what they are looking for in order to allow for quick wayfinding. The user is greeted with a welcome screen which allows them to go directly to the homescreen with the push of a single button. Once they have reached the home screen, they can choose to either view the posted library announcements or search for a book. Once the book has been located in the database, the user is shown how to get there either through a compass-like navigation tool or as an overview of the library. From the homescreen, the user can access settings where they can change the language, or turn on other features intended for the visually impaired as declared in the client's needs. The simplicity of the design helps the user focus on the app's features rather than being distracted by other factors. This ties directly into the client's need found in Deliverable B stating that the app is design intuitive.

4.0 ANALYSIS

For the pathfinding algorithm, the A*, RRT*, and manual insertions of paths were presented. The A* is good for finding the optimal path to a given location depending on the resolution of the map. By adding more nodes, or grids essentially to the map, the more accurate the algorithm when translated into real life. However, this solution will be slow to compute if the obstacle is placed in an unideal place and in the worst case, could need to compute upto (n-1)^2 number of points. If the person moves too far away from the path, a new path will have to be calculated, making this solution less desirable.

Manually plotting the path for the various points could be possible but would not be scalable. As more points are added, each node will have to connect to every other node, and trying to manually plot that would be very difficult.

The RRT* or the rapidly exploring random tree works by generating a random point on the map and having the nearest node to reach in that direction within a certain distance and this process is repeated many times, growing out branches while optimizing the paths, so with more nodes, the accuracy grows. This is good for rapidly searching unknown spaces with a lot of obstacles and finding an approximation of the optimal path. Since the library is large and contains many obstacles, this appears to be the ideal solution for finding the best path around the library. If the user also deviates from the path, the other paths generated from the semi random branching process can be used to reroute the user, saving computation time.

The pathfinding relates back to the target specification of accuracy. However, there are limitations to how accurate this software will be when translated into real life due to noise in the data collected from the hardware. Some precautions can be taken such as adding a bound to how far the user's location can deviate from the path before recalculating and removing any sudden jumps in location through using a running average of the user's position or an exponential filter.

Multiple designs were discussed for the navigation. Having the user track the book using a compass as well as using a process similar to that of a metal detector giving audio or vibrational feedback and a text to speech that gives the user directions to the book. For visual impairments, the best option would most likely be the auditory navigation due to the nature of the vibrational navigation which requires the user to go through trial and error. This is both inconvenient and dangerous as it requires the visually impared user to constantly walk around the library. Hence, the ideal navigation to allow everyone to navigate through the library would be through the use of auditory directions and a visual compass which are concepts one and three.

The different power managing systems were discussed with the team. Having a windmill or a renewable energy mechanism powering the Arduino did not seem feasible. The safest and simplest way to power the Arduino would be to connect it to an outlet. The beacons can either be powered by an internal battery already included or be connected to a charging station. The clients want a long lasting battery life so new beacons do not need to be ordered and replaced after a short period of time. The team's target specification is to deliver a system that will be able to operate for an extended period of time. For these reasons the beacons will be powered through a rechargeable battery station. This way the batteries for the beacons can be charged while the beacon is in use.

In the final iteration of the homescreen design, a login button and settings shortcut was implemented. The login button allows faculty members to edit the location of the beacons and troubleshoot any issues. The placement of the settings button on the home screen allows users with visual impairments to directly access features such as sound and vibrate. Having this feature pop up immediately instead of having the user navigate the app ensures the most user friendly experience.

Using a centralized microcontroller for all the beacons is a better idea, because it will save both space and money which meets the client needs about affordability and portability declared in table 2 of deliverable B. In addition, this solution will make the collection of information from the different beacons easier.

5.0 FINAL CONCEPT

The figure below shows the final concept for the power management system of the beacons and Arduino. The Arduino will be plugged into a power outlet because it is the safest and simplest way to make the microcontroller run. The beacon will be able to connect to a rechargeable battery station. This way the beacon is not a one time use and can be used over and over again. The batteries for the beacon will be able to be charged separately while the beacon is in use.

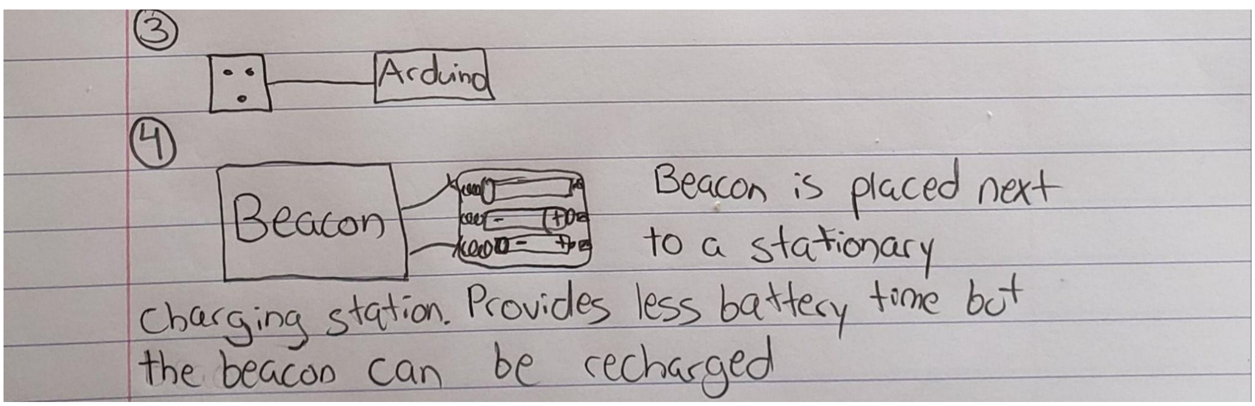

Figure 12. Final concept for power system

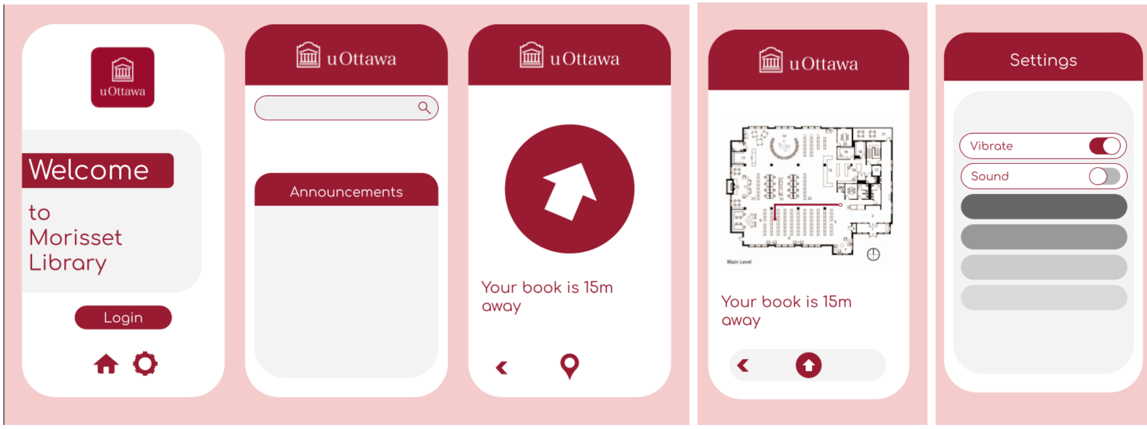

Figure 13. Final concept for UI

This final concept of the UI satisfies the target specifications in terms of simplicity and user friendliness. The implementation of the settings shortcut from the home screen as well as minimizing the number of pages the user has to go through to get to the wayfinding. The final product ensures the user knows how to navigate the app without struggling or needing further instruction.

The selected method for pathfinding is the RRT* and will be implemented in the navigation process shown in bottom of the flowchart below.

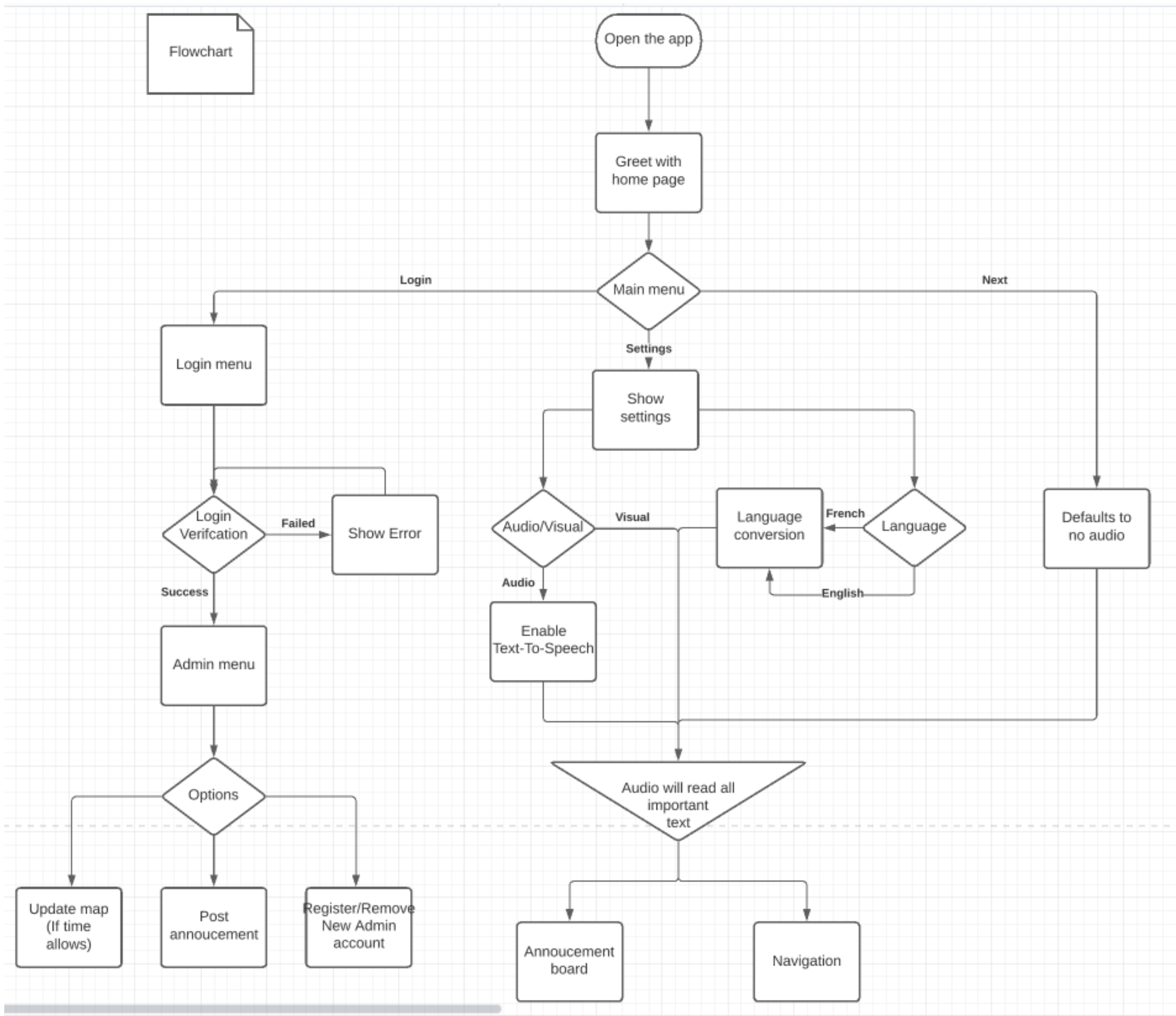

Figure 14. Final concept of program flow

Based on the analysis, this flowchart was created and outlines the program logic for the application. As mentioned, the user is greeted with the main page and is presented with the options from the start for quick access to the text-to-speech and language selection. Once the user selects their ideal settings, they are able to easily access core functionality of the program. This supports the target specifications and functional requirements such as aesthetics, ease of use, multi-language support.

Based on the analysis, the final concept for the beacons and arduino part is the use of one central microcontroller for all the beacons.This way, the expenses are reduced and the space needed for the system is less important.

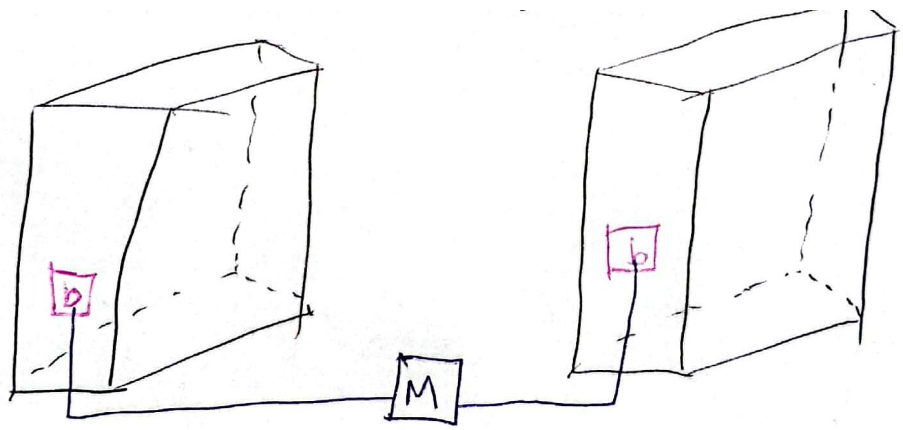

Figure 15. Final concept for arduino, beacon concept

A downside to our current implementation is that a wall plug is required near the arduino for it to be powered. A long wire could be used but could become a tripping hazard and may affect aesthetics of the library. There is also the issue of the algorithm having to calculate the path each time a user goes from point A to point B instead of saving that information for future reference.

6.0 CONCLUSION

In this deliverable, each of the design concepts that were created were compared. The concepts are the fruits of individual designs the team members came up with. From this, each of the concepts were reviewed by the team and it was seen how they fit into the target specifications. After deciding on the best ideas presented, a global design concept was made which captures all of these ideas and combines them into a functioning prototype. The functioning prototype encapsulates the best ideas from the UI, the power management and placement of beacons and microcontrollers, and the algorithm for finding the most efficient path.

The figure below shows the project planning done on Wrike and the current progress. Future steps have been assigned

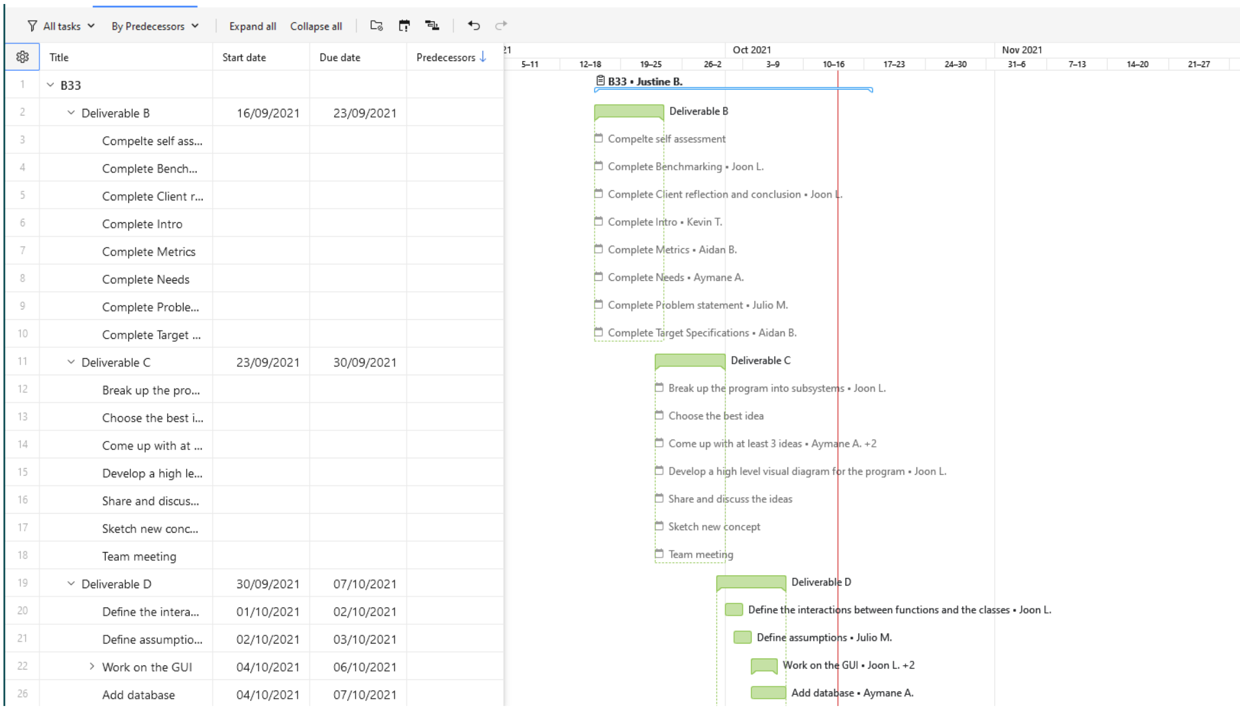

Figure 16. Gantt chart

REFERENCES

[1] N. Swift, “Easy A* (star) pathfinding,” *Medium*, 29-May-2020. [Online]. Available: https://medium.com/@nicholas.w.swift/easy-a-star-pathfinding-7e6689c7f7b2. [Accessed: 12-Oct-2021].

[2] T. Chin, “Robotic Path Planning: RRT and RRT*,” *Medium*, 13-Feb-2019. [Online]. Available: https://theclassytim.medium.com/robotic-path-planning-rrt-and-rrt-212319121378. [Accessed: 12-Oct-2021].

[3] “How to Parse Bluetooth Beacons (iBeacon and Eddystone) with Arduino 101” https://www.andreasjakl.com/parse-bluetooth-beacons-ibeacon-eddystone-arduino-101/