CptS 484: Software Requirements

WRS Evolution

Requirements Elicitation

Team Members

Christopher Young

Cong Trinh

Taryn Burns

Anne Lin

Sean Luchessa

Table of Contents

Revision History

Date	Version	Changes	Editor

[1] Introduction

1.1. Purpose of the System

1.1.1. Our goal is to build a smartphone app that will assist visually impaired people with safely navigating their way indoors. For example, a blind visitor may want to go from one office to another and thus will need help finding a safe route to their destination. The app will ask for the destination, give instructions for their desired route, and take note of any obstacles that may stand in the user's way. It will also take into account the familiarity of the route and the time it takes to reach the end of it. In addition, our team will incorporate several requirements into the app. They include but are limited to:

- A ubiquitous system
- Usable for blind people/visually impaired
- Customizable
- Easily extendable
- Choose the fastest route

The primary audience for this app are people who are severely visually impaired. Therefore it is important to not only make the app work accurately, but also have great usability.

1.2. Scope of the System

1.2.1. The scope of the system includes achieving both Phase I goals and Phase II goals, completing all the necessary requirements in the requirements doc, and passing all of the test cases for the project. The team is currently working on completing Phase One, which contains:
Designing the requirements model,

- Specification for the app,
- A complete WRT document,
- A revised Phase I plan, a collection of our team meetings,
- A Powerpoint presentation of our project,
- A prototype implementation of the app

It is due by October 13th, 2019. Everyone in the team is assigned to a certain section of each project document. In addition, there are also weekly group meetings in order to ensure the project's quality and to maintain good communication between each team member.

1.3. Objectives and Success Criteria

1.3.1. Our main objectives include:

- Being able to navigate indoors
    - The app will need to be able to instruct the user to go from one room in the building to another
- Safe, quick, and comfortable navigation
    - The app will need to notify users of any possible obstacles in the way while also providing the quickest and most convenient path possible.
- Easy to use
    - Due to the intended target, the app will need to be user friendly to the visually impaired.
    - The app will offer a variety of features that will make the user's journey much easier. They include short, concise instructions, the ability to pause and change destination, being able to utilize the user's routine schedule, and place emergency calls if needed.

Our success criteria will include:

- Whether or not the user is able to reach the destination within a certain amount of time
- If the app is able to detect typical obstacles that might be in the way
- If the user is able to utilize the app's features without any bugs/errors.
- If the user is able to use the app without encountering any huge problems, such as the chosen route being not optimal and encountering any unwanted obstacles.

1.4. Definitions, Acronyms and Abbreviations

Unified Modeling Language (UML) - A diagram with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

Internet Operating System (iOS) - An operating system used for mobile devices manufactured by Apple Inc.

Xamarin - An IDE that allows for the development of cross-platform mobile applications using C#.

Android - An open-source operating system used for smartphones and tablet computers.

Gyro sensors - A device that senses the change in rotational angle per unit of time.

GPS - Stands for Global Positioning System. It is a radio navigation system that allows users to determine their exact location in the world.

Waterfall model - A breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialisation of tasks.

C# - An object-oriented programming language made by Microsoft.

Model View Controller (MVC) - A software design pattern commonly used for developing user interfaces which divides the related program logic into three interconnected elements. They include the *model* (data), the *view* (user interface), and the *controller* (processes that handle input).

General risk - An activity or event that may compromise the success of a software development project.

Acceptance testing - A level of software *testing* where a system is tested for acceptability. The purpose of this *test* is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.

Voice Recognition - The ability of a machine or program to receive and interpret dictation or to understand and carry out spoken commands.

1.5. Overview

1.5.1. We are currently designing and creating a smartphone app called “Theia” that will help assist the visually impaired indoor navigation. As stated in the project requirements document, the features needed for this app consists of:

- Indoor Navigation
- Obstacle Detection
- Consideration of Route Familiarity

Therefore, we will include building maps and utilize the GPS in order to ensure indoor navigation, use the phone’s sensors in order to detect any obstacles that may be in the way, and have a Room Profile for users who mainly want indoor navigation for their daily routes.

In addition, we also need to complete the needed requirements. They are categorized into two sections - functional and non-functional. Since non-functional may be more important to accomplish than functional ones, our team will be focusing on the former first before moving onto the latter.

There are currently two phases for this project: Phase I, which includes coming up with requirement models, specifications, and a prototype implementation and Phase II. Our team will be completing Phase I and Phase II by their due dates.

[2] Issues

2.1. Domain, Stakeholders, Functions & Non-Functional Objectives

2.1.1. Domain

Preliminary Domain ID	Preliminary Domain Description
PD1	Individuals with severely impaired eyesight, ranging from legally blind to total blindness.
PD2	Caretakers of the disabeld.

2.1.2. Stakeholders

INFLUENCE	High	● Professor Bolong	● Blind community ● Team Members
	Low	● App Raters	● Washington State University Student ● American Council on Education ● Parents
		Low	High
		INTEREST	

2.1.3. Functions

Preliminary Functional Req. ID	Preliminary Domain Description
PFR1	Object detection in order to avoid any collisions when traveling.
PFR2	Emergency contact dialing from application
PFR3	Speech output

2.1.4. Non-Functional Objective

Preliminary Non-Functional Req. ID	Preliminary Domain Description
PNF1	Safe, fast, and comfortable navigation for indoor places.
PNF2	User-friendly interface for the visually impaired.

2.2. Functional Requirements

Requirement ID	Functional Requirement	Description	Priority (1-3)
FR1	GPS tracking	Track the user position with gps while navigating through building. The user should receive updates throughout their trip based on the progress they have made. GPS will be used to tell if the user has stayed on their path and give the direction system information on their position.	1
FR2	Object detection	Detect objects with the camera to alert user of upcoming path obstructions. This includes tables, open doors, other people, and various objects that would inhibit their ability to walk along the route. This will utilize the camera hardware and an algorithm to detect objects in the view.	2
FR3	Voice recognition	Ability for user to give commands through speech. The main users will not be able to use a graphical interface, so voice commands will need to be the main interaction method with the app.	2
FR4	Settings	Save user configuration, common locations, speed, buildings.	2
FR5	Navigation	Provide correct directions to navigate the user through the building to their destination. This will be delivered through speech, alerting the user to continue straight, to make any turns, or when they have gone the wrong way.	1
FR6	Map parsing	Ability to parse imported map data in a way that directions can be obtained from the output.	1
FR7	Emergency reporting	A fail-safe will be implemented to alert emergency contacts in the event of a failure.	3
FR8	Hardware Buttons	Remapping hardware buttons to allow for	3
FR9	Settings UI Page	Provide a UI for configuring settings	3
FR10	Vibration	Vibrate the device when the app alerts the user	3
FR11	Navigation UI		1

2.3. Non-Functional Requirements

Requirement ID	Non-Functional Requirement	Description	Priority (1-3)
NF1	Directions	Provide directions to the user through the use of various software and hardware. A system is needed to determine directions from current location to destination using data provided from imported map.	1
NF2	Blind User Interface	Interface (apart from settings) must be usable without the ability to see the screen. Utilizing voice commands and hardware buttons will be needed.	1
NF3	Efficient Directions	Will provide the shortest reasonable route to the destination. The system will determine the shortest route from location to destination.	3
NF4	Caretaker Options	Setup for the app that is targeted towards to caretaker. This can include gui.	2
NF5	Availability	The app will need to be running throughout the trip and any failures will need to be followed up by alerting proper entities.	2

2.4. Testing Requirements

Test Case ID	FR ID	Test Case	Test Steps	Test Data	Expected Results
TC1	FR1	Verify Gps	1. Detect Gps Signal 2. Sample Data 3. Confirm Data as current location	Location = Spark	Lat = 46 Long = -117
TC2	FR1	Detecting Movement	1. Detect Gps Signal 2. Continually sample data 3. Move with changing latitude 4. Move with changing longitude	Start Location = {Lat,Long} End Location = {Lat, Long}	Start Location = Start Location End Location = End Location
TC3	FR2	Detecting object	1. Object obstructs users path 2. Bring object into camera view	Object = chair	Camera system detects and flags the object.
TC4	FR2	Alerting user of object	1. Object gets detected	Object detected = True	Camera sends alert to navigation system
TC5	FR3	Storing voice input	1.User speaks test phrase 2.App stores correct phrase	Test phrase = “This is a test”	Stored phrase = “This is a test”
TC6	FR3	Parsing voice input	1.User speaks test command 2. App follows the spoken command	Test command = “Call emergency phone number”	The app will call the emergency phone number upon hearing the command.
TC7	FR4	Saving settings configuration	1. User puts in their test configurations into the setting. 2.App saves it until user decides to change it.	Test settings: Set volume to super high, change emergency contacts, map, and room profiles	The app will save the given settings correctly.

TC8	FR4	Settings changes take effect in app	1.User changes their setting configuration. 2. App takes in the change and then saves it.	First put in one set of setting configurations and test if the app saved them correctly. Then input another set of setting configurations.	The app will correctly save and show the new settings configurations.
TC9	FR5	Directions come from map	1.User inputs starting location and destination 2.User starts navigation 3.App selects the best route and gives the correct instructions on screen and verbally	1.Put in test locations, follow navigation, and see if instructions pop up and if they are correct (user arrives to the test destination).	User follows directions and arrives at the correct destination.
TC10	FR5	Direction appears when required	1.User inputs starting location and destination 2.User starts navigation 3. App shows instructions during route	1.Put in test locations, follow navigation, and see if instructions pop up	User sees instructions pop up as they are traveling and they arrive at the correct destination.
TC11	FR6	Store input map	1.User inputs desired map 2.App correctly stores map	Input test map	Stored map = test map.
TC12	FR6	Map parsed into data structure	1.User input test map 2. App uses the map to give instructions for route	Test map input = Sloan Building	The app will utilize the map's layout to find the best route for the desired location in the building.
TC13	FR7	App crash detecting	1. Crash code sent to application 2. Application crash 3. Emergency contacts alerted	Exit code = crash (-1)	App logs crash and performs emergency call
TC14	FR7	Emergency contact alerted	1.App senses that the user has stopped moving for a long time during navigation or is	Stop moving during navigation for a	App will immediately call the emergency

			commanded to call the contact 2.App then immediately calls the emergency contact	long time	contact
TC15	FR8	Hardware button press detected	1. Hardware button gets pressed 2. Signal gets intercepted by navigation system	Up or Down volume button pressed	Navigation system detects correct button
TC16	FR8	Default action remapping	1. Hardware press detected 2. Default action intercepted 3. Remapped action performed	Navigation system detects button	Remapped action performed instead of default action
TC17	FR9	Settings Menu Opens	1. Click on the gear symbol 2. The app then takes you to the Settings Menu	Click on the gear symbol	The user will be taken to the Settings Menu
TC18	FR9	Options appear for user	1. Open settings menu	Click on the gear symbol	Options display on screen
TC19	FR10	Vibration activates	1.Navigation sends signal to vibration hardware 2.Hardware receives signal and operates	Vibration signal sent	Phone vibrates
TC20	FR10	Vibration activated on event	1.User goes the wrong way during navigation 2.The app will send off a vibration to signal the user's error	Test instruction: go the opposite way for one of the instructions during the navigation	The phone will vibrate once it senses that the user's position is going the wrong way.
TC21	FR11	Navigation UI opens	1. Menu screen switches to Navigation 2. Voiced directions start once directions are loaded on screen	Navigation request and directions set	Navigation UI should load without fail and have all features ready to go for the user
TC22	FR11	Button functionality	1.User clicks pause 2.User clicks resume	Ongoing navigation	App pauses navigation then starts when buttons are pressed

[3] Issues with Preliminary Definition Given (ambiguities,' incompleteness,'inconsistency,'conflicts...)

3.1. Unclear definition for the “Visually impaired”

3.1.1. Description

- The word “visually impaired” is a bit ambiguous in that it doesn’t state who exactly qualifies as visually impaired. By clearly defining the term, our stakeholders can know exactly who the primary audience is for the app and know what sort of different features for different ranges of people.

3.1.2. Options for resolving the issue

- We could set a range of poor eyesight. For example, the range could be people with -1/20 vision to people who are legally blind to people who are blind. But how do we justify this estimate?
- We could use a definition based on how far away the person can read words. Since the app is for people who are unable to reliable read the instructions on the screen (hence the need for a user-friendly interface), we could set the limit to people who are simply unable to reliable read ~three feet away to people who are unable to read at all.

3.1.3. Decision and rationale

- Option 1 as by clearly defining the range, we are able to know what features and considerations to incorporate into our app.

3.2. What constitutes a ‘safe, reliable and comfortable travel for indoor places?’ (PNF1)

3.2.1. Description

- Safe, reliable, and comfortable are great descriptions of the app’s navigational functions. But what do they mean, exactly? How does it show that in the app? By giving clear definitions, our stakeholders and team members know exactly the quality of the navigation feature.

3.2.2. Options

- We could set a definition for each word relating to our navigational app. For example, safe navigation would mean having an object detection feature so that during navigation, the user will be able to avoid any obstacles in the path. A reliable navigation would mean using a good GPS. A comfortable navigation would mean using a GPS that calculates the fastest route with the least amount of obstacles. All of those together would mean the app has a safe, reliable, and comfortable navigational feature.

3.2.3. Decision and rationale

- Option 1 as it allows the least ambiguity in defining the descriptors.

[4] Issues with NFR Given (ambiguities,' incompleteness,'inconsistency,'conflicts...)

4.1. Definition of Caretaker and their roles is Unclear (NF4)

4.1.1. Description

- How does the caretaker communicate with and respond to the visually impaired user?

4.1.2. Options

- Our application will include a two-way communication tool instead of it just being one-way.
- Caretaker can use “natural means” such as speaking or braille or pictures.
- The user and caretaker may use other smartphone tools that are outside of the app.

4.1.3. Decision and rationale

- Option 1. If something dangerous happens or they need to immediately inform their caretaker of something, then it is best they can use the app to do so. Through this, the user won’t have to waste much time. There is also the added convenience bonus.

4.2. How will the directions be presented through the app?(NF1)

4.2.1. Description

- Due to the user’s eyesight, it may be difficult to read the instructions on-screen.

4.2.2. Options

- Symbols such as directional arrows can be given and due to their simplicity, can be easily understood.
- Verbal instructions can be given so that the user does not have to rely on their eyesight to follow instructions.
- A mix of option 1 and option 2 to cover all areas - verbal and sight. By having both options, one can utilize both to maximum understanding of the app’s instructions.

4.2.3. Decision and rational

- Option 3 as it’ll cover all of the basic form of communication for the user. Having both verbal and visual aids will help the user the most.

[5] Issues with FR Given (ambiguities,' incompleteness,'inconsistency,'conflicts...)

5.1. Reliability of Emergency Contact System (FR7)

5.1.1. Description

- How do we guarantee that emergency requests made by the user get to their intended contact? The scenarios involved could range from somewhat dangerous to life-or-death. It would be bad if denial-of-service happened.

5.1.2. Options

- 911 could be called right after the emergency contact is called or if the emergency contact could not be reached.
- Contact emergency contacts based on hierarchy: emergency contact, 911, then nearby hospitals.

5.1.3. Decision and rationale

- Option 2. Having a hierarchy makes understanding how the emergency contact system works much clearer. In addition, by following the hierarchy, it reduces room for confusion and covers the basic scenarios.

5.2. Caretaker Options (NF4)

5.2.1. Description

- How do we make the interface and app easy for someone who may not be familiar with such things?

5.2.2. Options

- Link the manual guide in Settings/Help section so that the caretaker can read up on how to use the system

5.2.3. Decision and rationale

- Option 1 since it's a low-effort and easy option for the caretaker to utilize and it allows the manual guide be used for its purpose.

[6] Improved Understanding

6.1. Problem

6.1.1. Travelling to various buildings is something most people do every day. Whether it's for work, school, business, or other reasons, it is something that is important to a lot of people. For the visually impaired, it is a bit more difficult to do as they have to rely on special tools and services to get to the places they need. Even then, the tools they use are either too narrow in their usage or are not reliable enough to get them where they need and in a safe, comfortable way. In addition, there is also the lack of emergency options when it comes to travelling on your own.

6.2. Goal

6.2.1. Our goal is a cross-platform application that will be used to aid the blind with navigating buildings. This app will assess the needs of a blind person to lead the implementation of a helpful tool. Hardware that will be utilized includes voice commands, GPS for directions to navigate hallways, and make use of various sensors to detect obstacles. It will be an easy-to-use and powerful solution for the visually impaired users who want to be able to travel safely and reliable indoors.

6.3. Improved Understanding of the Domain, Stakeholders, F and NF Objectives

6.3.1. The Theia App will be used by people regardless of age, with visual impairments as defined below:

- People who are severely near-sighted
- People who are legally blind
- People who are unable to see at all

6.3.2. The Theia app should aid travelling by the visually impaired in the following locations:

- People who live in cities
- People who work in cities.

- People who work on campus
- People who live on campus
- People who either mainly work or live in an area that has a decent amount of buildings.

6.3.3. The user shall be able to communicate their need for the following to a caretaker using the Theia App:
- Help in emergencies

6.3.4. The system shall provide the following sensory aids (visual and audible) to the user when using the app.
- The visual aids consisting of pictures, icons, and text.
- The audible aids are spoken phrases that follow every instructions and written question when using the app.

6.3.5. The system shall provide a way for the user to make any emergency calls to any emergency contacts, as well as send coordinates and the user's name to them.

6.3.6. The emergency contacts will be grouped and selected on a needs basis, with the ability for the user to add more.
- The system should be easily usable by the caretaker, by providing an easy tutorial in the user's manual that is linked in the settings.
- The caretaker should be able to perform any actions on the application that the user can

6.4. Improved Understanding of Functional Requirements

6.4.1. The system uses GPS tracking in order to track user's position during navigation and for aiding in guiding them to their desired location. In addition, it's also used to send coordinates to emergency contacts/911 in scenarios where the user is in danger.

6.4.2. The application shall make emergency calls when it senses that the user has fallen or when the user verbally commands it to do so.
- The application shall make calls in the following order: (1). Emergency contact, (2). 911, (3). Nearby hospitals

6.4.3. The system uses a map parser system so that the user can easily travel to any building they want as long as they input the right map building.

6.4.4. The system should give the user a Room Profile and set of settings to the user in order to let them customize their own needs and wants. Each user will have their own Room Profile so that the app will be able to readily adapt to each new user.

6.4.5. The system shall have an easy-to-understand navigation UI that allows the user to pause, resume, or cancel the navigation through the use of big, readable buttons/verbal commands.

6.4.6. The system shall have an object detection feature so that during navigation, the user can easily avoid obstacles that may be in their way.

6.5. Improved Understanding of Non-Functional Requirements

6.5.1. The system shall be user-friendly. It shall have voiced instructions and questions for the user and the set-up shall have an entirely voiced option. The instructions will be incredibly simple to follow and shall be a step-by-step procedure. All of the options and buttons will be displayed clearly and organized in a way that is easy to spot. There will also be a tutorial linked in the settings page in case the user needs help.

6.5.2. The directions for the navigation will be short and efficient, so that it is easy to follow and understand. The directions will be something like this: “Turn right/Turn left/Walk ahead for three minutes”.

6.5.3. The system shall have a good error-and-handling recovery. It shall run even if it encounters errors.

6.5.4. The system shall be responsive. Any command the user inputs or any page it visits shall only take up to ~1-2 seconds. Any instructions that is given shall also take up the same amount of time and the emergency calls the app gives also shall take the same amount of time.

6.5.5. The system shall be accurate. The settings shall be able to completely reflect the user’s inputs. The navigational system shall be able to accurately take the user to their desired destination using the most appropriate route. The instructions given shall be displayed accurately, whether through audible or visual means. All of the buttons should be able to give their proper functions if pressed and any calls given shall be done immediately and accurately. The calling system will also be tested before release as the user’s safety is of utmost importance.

6.5.6. The system shall have an easy-to-use options/interface for the caretaker to use. It shall include a language option, tracking option, map buildings options, and emergency contact option.

[7] WRS

7.1. W

7.1.1. Problem

Our problem is to create an app that will help those who are visually impaired, navigate through an area to get to their destination by using an app on their smartphone.

7.1.2. Goals

Design app page UI as a computer drawn image to provide reference for graphic interface code development by September 13th 2019

Mock design implemented in C# using Xamarin as an Android application by September 27th

Map designs of Sloan Hall created as hallways and room physical map by October 4th

7.1.3. Improved Understanding of Domain, Stakeholders, Functional & Non-Functional

7.1.3.●.1. Domain: Disabilities
- Visually Impaired

7.1.3.●.2. Domain: Activities

7.1.3.●.3. Stakeholders
- Visually Impaired
- Secondary: Caretakers

7.1.3.●.4. Functional Objectives

7.1.3.●.5. Navigating Indoors

7.1.3.●.6. Non-Functional Objectives

7.1.3.●.7. Safe Navigation

7.1.3.●.8. Fast Navigation

7.1.3.●.9. Comfortable Navigation

7.1.3.●.10. User Friendly Interface

7.2. RS

7.2.1. Functional RS

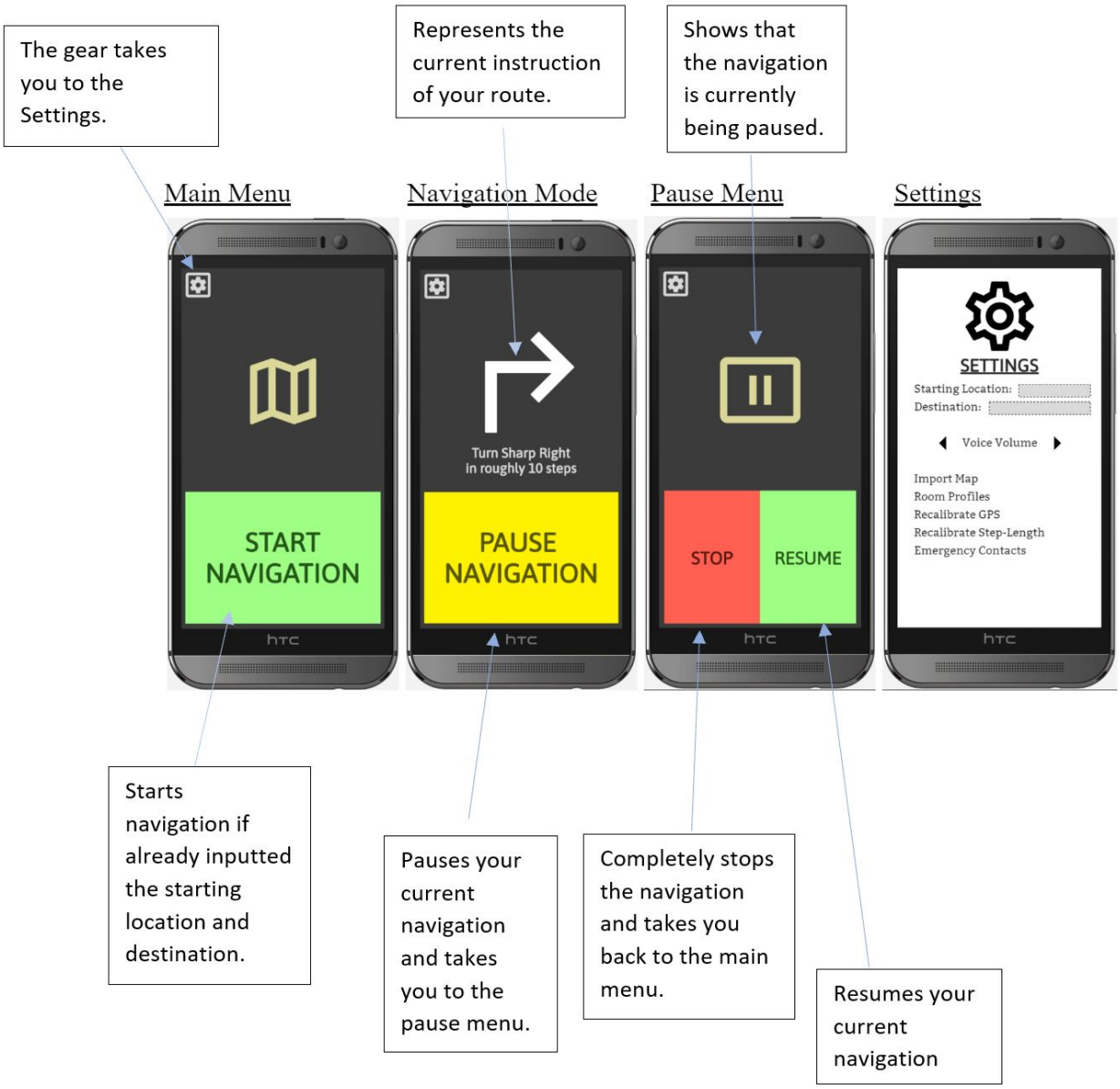
- Accepting from the user the location to go
- Figuring out routes to reach the destination
- Telling the user to stop at the right place to turn
- Detecting obstacles and telling the user what to do in order to avoid collision
- Placing emergency calls and messages, possibly after a fall or when the system has lost its current location
- Figuring out the next action(s) based on the user's habits/schedule

7.2.2. Non-Functional RS

- The system shall help the user safely navigate indoors
- The system shall lead the user through the fastest route
- The system shall be usable for blind people
- The system shall be ubiquitous
- The system shall be customizable for every user
- The system shall be easily extensible

[8] Preminimary Prototype

8.1. Prototype Overview



[9] Prototype Interface Mock-ups

9.1. Interface:

- UI:
  - Buttons
    - Start (Graphical / Hardware remapping)
    - Setting
  - Display
    - Direction Arrow
    - Small map
    - Steps taken / remaining
    - Warning about user being blind
  - Voice
    - Directions
    - Motivational messages (You're doing great! You went the wrong way! Congrats, you've reached your destination!!)
    - When to stop

- Start-Up (include Windows start-up music)
  - Destination (Building & Room Number)
  - Import Map
  - Length per Step
  - Name
  - Emergency Contacts

- Execution
  - Voice Inputs
    - Say "Start" on screen menu → Prompt for destination
      - Say Destination → Starts Navigation
    - Say "Stop" during navigation → Voice output possible commands.
      - Say "End" → Returns to menu
      - Say "New Destination" → Input New Destination
  - Turn off directions when finished with trip

        - Return to main screen
    -

- Settings
    - Voice (Sound, Language)
    - Naming Profiles (Input Room #, Room Name)

9.2. Features:

- Main Page
    - Start Navigation Button
    - Settings Button
- Navigation page
    - Direction arrow
    - Direction text
    - Pause button
    - Settings button
- Pause Menu
    - Pause visual
    - Settings button
    - Stop and Resume buttons
- Settings page
    - Data entry fields

[10] User Manual

10.1. Purpose of the Prototype

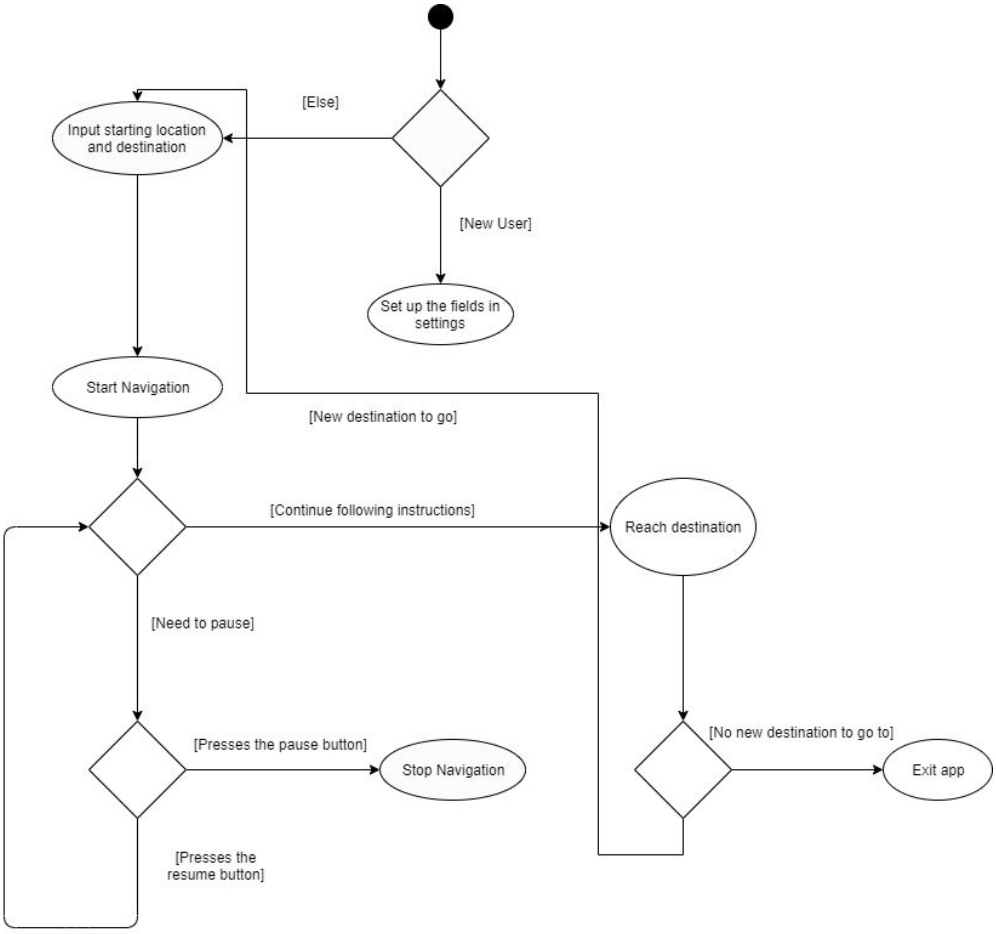
The purpose of the prototype is to be able to understand how the application is supposed to look like and what functionalities will be available. It will help allow the team members and the stakeholders get a clear picture on how the app functions IRL and what other features need to be added or edited.

10.2. Warnings

Because the app is for the visually impaired, it is important that we ensure that they are able to travel the route safely and accurately. Thus, we will be utilizing two different vibration/sounds as safeguards: One to tell the user that they have reached the right stop in their navigation and another to signal that they are too far-off from their route.

In addition, in cases of emergencies, the app will contact the emergency contact first, then 911. This is to ensure the user's safety no matter what happens. The app will send the user's name and coordinates when calling both the contact and 911 so that both parties have all the required information.

10.3. Diagram of the interface



10.4. Instructions

10.5. In order to get started, first go to settings.
10.6. Input the maps you want to use, room profiles, and emergency contacts.
10.7. Input the Starting Location and Destination.
10.8. Adjust Voice volume and recalibrate when needed.
10.9. Follow voice commands/commands on screen.
10.10. Press Start Navigation to start your journey
10.11. Press Pause Navigation to pause your journey
10.12. Press stop in order to fully stop your journey or resume to go back to it
10.13. Once the destination is reached, the app will ask if the user wants to travel to a different location
10.14. Press yes to insert a new destination or no to cancel.

10.5. Features

The features included in the Theia App include:

GPS Tracking: This feature is used for securing the user's coordinates for emergency situations. This will also help add in navigation as the user will receive updates based on their progress. The GPS Tracking will be activated when the user starts the navigation process or when the emergency contact system is activated.

Map Parser: This feature is used as part of the navigational process. In order to get the user to their desired building/room location, they first need to enter the map of the building. Through the map parser, the app is able to get the layout and therefore the location that the user wants. It then uses that as part of the navigation process.

Safe and Reliable Navigation: By calculating the fastest route and combined with short, simple instructions and the object detection feature, the user will able to reliably and safely get to their destination.

User-friendly Interface: There will be a voiced option for everything - instructions, reading certain things out loud, or commanding the app through verbal commands. For example, when the navigation starts, all of the instructions will be communicated through an audible and visual way.

Emergency Contact: This will be inputted during setup. The emergency contact system is for when the user is in a dangerous situation and needs help immediately. The call hierarchy is this: Emergency contact, 911, then nearby hospitals. The system will be activated whenever the user either presses the emergency call button, commands the app verbally, or when the app detects the user has fallen and is unable to give the command. In addition, the app will send a message containing a request for help and the user's coordinates to the contacts.

[11] Traceability

[12] Traceability Matrix

Req. ID	Requirement Description	Test Case ID	Status
FR1	Track user with GPS	TC1, TC2	TC1: Untested TC2: Untested
FR2	Object detected User alerted	TC3, TC4	TC3: Untested TC4: Untested

FR3	User speaks command Command is executed	TC5, TC6	TC5: Untested TC6: Untested
FR4	Settings configured Configuration saved	TC7, TC8	TC7: Untested TC8: Untested
FR5	Directions created Relay to users	TC9, TC10	TC9: Untested TC10: Untested
FR6	Map inserting Stored in data Direction creation	TC11, TC12	TC11: Untested TC12: Untested
FR7	Emergency Detecting Proper reporting	TC13, TC14	TC13: Untested TC14: Untested
FR8	Remapping hardware Button press detection	TC15, TC16	TC15: Untested TC16: Untested
FR9	Settings UI Page	TC17, TC18	TC17: Untested TC18: Untested
FR10	Vibrations	TC19, TC20	TC19: Untested TC20: Untested
FR11	Navigation UI	TC21, TC22	TC21: Untested TC22: Untested
NF1	Correct directions		
NF2	Interface accessible without sight		
NF3	Determine best route		
NF4	Setup options for caretaker		
NF5	No connections dropped during route		

[13] References

13.1. ...

[14] Appendix I: Process Details

14.1. Phase 1

14.1.1. Phase 1 Roles

14.1.2. Originally had sections of the document split up but ended up collaborating together on the entire document, including writing and editing.

14.1.3. Chris: Team Leader

14.1.4. Cong: Developer

14.1.5. Taryn: Developer

14.1.6. Anne: Developer

14.1.7. Sean: Developer

14.1.8. Meetings

14.1.9. Weekly meetings on Discord if group members are not able to meet in-person.

14.1.10. Weekly Tuesday meetings on campus as a group to collaborate and work on the project. This includes our own meeting notes that are put up on Google Drive for our reference.

14.1.11. Regularly communicate through Discord on a daily basis.

14.1.12. Activities

14.1.13. Collaborative work on both code and documentation work and brainstorming ideas (i.e. for prototype)

14.2. Phase 2

14.2.1. Roles

14.2.2. Originally had sections of the document split up but ended up collaborating together on the entire document, including writing and editing.

14.2.3. Chris: System Design and Documentation

14.2.4. Cong:

14.2.5. Taryn:

14.2.6. Anne: Documentation

14.2.7. Sean:

14.2.8. Meetings

14.2.9. Weekly meetings on Discord if group members are not able to meet in-person.

14.2.10. Weekly Tuesday meetings on campus as a group to collaborate and work on the project. This includes our own meeting notes that are put up on Google Drive for our reference.

14.2.11. Regularly communicate through Discord on a daily basis.

14.2.12. Activities

14.2.13. Documentation

14.2.14. Preliminary Prototype design and implementation

14.2.15. System Planning

[15] Index

A

Acceptance testing - A level of software *testing* where a system is tested for acceptability. The purpose of this *test* is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.

Android - An open-source operating system used for smartphones and tablet computers.

C

C# - An object-oriented programming language made by Microsoft.

D

Discord - Freeware VoIP application and digital distribution platform. This was originally used for the video game community. The platform allows text, image, video, and audio communication between users in chat servers.

F

Functional Requirements - Defines a function of a system or its component, where the function is described as a specification of behaviour between inputs and outputs.

G

General risk - An activity or event that may compromise the success of a software development project.

Goals - Refers to achieving a desired outcome at a specific end date

GPS - Stands for Global Positioning System. It is a radio navigation system that allows users to determine their exact location in the world.

Gyro sensors - A device that senses the change in rotational angle per unit of time.

H

Hardware - The collection of physical parts of a computer system.

I

Interface - Shared boundary across where two or more separate components of a computer system exchange information.

Internet Operating System - An operating system used for mobile devices manufactured by Apple Inc.

M

Model View Controller (MVC) - A software design pattern commonly used for developing user interfaces which divides the related program logic into three interconnected elements. They include the *model* (data), the *view* (user interface), and the *controller* (processes that handle input).

N

Non-Functional Requirements - A requirement that specifies criteria that can be used to judge the operation of a system rather than specific behaviors

O

Object Detection - A software feature that assists the computer to detect potential objects for different reasons.

P

Problem - A situation that needs to be solved

Prototype - An early sample, model, or an early release of a product built in order to test or lay out a concept or process for the later product. This prototype does not define the finished product (i.e. Interface wise, functionality, ect).

R

Requirements - A feature that is absolutely necessary in a software development project. There are different levels of requirements that may be more important in a project over another.

S

Stakeholders - People or a group of people affected by a software development project

T

Theia - Greek Titan Goddess of sight (Thea) and shining ether of the bright, blue sky.

Traceability - Ability to relate the various types of software artefacts created during the development of software systems.

U

Unified Modeling Language (UML) - Diagrams that visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

User Friendly - A machine or a system that is easy to use and understand.

User Interface - is the point of human-computer interaction and communication in a device. This includes displays on screens, appearance on a monitor, keyboard, mouse, ect.

User Manual - Lays out the steps and actions required to use the app.

V

Visually Impaired - Known as vision impairment or vision loss. It is the decreased ability to see to a degree such that it causes problems that cannot be fixed through usual means such as glasses and contacts.

Voice Recognition - The ability of a machine or program to receive and interpret dictation or to understand and carry out spoken commands.

W

Waterfall Model - A breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialisation of tasks.

WRS - Resource File including the problem, functional requirements, and non-functional requirements.

X

Xamarin - An extension of C# that allows for the development of cross-platform mobile applications