

March 31, 2018

Andrew H. Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia V5A 1S6

Re: Capstone Project Design Specifications for *SafeLift* and *AppAid*

Dear Dr. Rawicz,

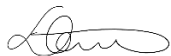
Enclosed is a document that provides the design specifications for our Capstone Project, *SafeLift* and *AppAid*. *SafeLift* is a proximity alert system that aims to improve the overall safety of workplaces that utilize powered industrial trucks such as forklifts. *AppAid* is a risk management app that helps manage and mitigate risks before they become fatal.

The objective of this document is to provide the technical specifications and functionality of the design proposed for our product. It includes the system and component design details as well the justification for the chosen design approach. This document will be used as a guide and reference during the development of the product.

Our team consists of four senior engineering students with backgrounds in electronics, software, embedded systems, hardware and mechanical design. Please do not hesitate to contact the team in case you require any further information at fmose@sfu.ca.

Sincerely,

Fred Mose,
Chief Executive Officer

A handwritten signature in black ink, appearing to read 'Fred Mose'.

PowerLift Safe Solutions

Enclosure: Design Specifications for *SafeLift* and *AppAid*.



Design Specification
SafeLift and AppAid
You don't have to be a statistic

Project Team Daniya Zafar
Fred Mose
Layomi Dele-dare
Wenhao Zhang

Contact Person Fred Mose
fmose@sfu.ca

Submitted to Dr. Andrew Rawicz
Prof. Steve Whitmore
Faculty of Engineering Science
Simon Fraser University

Issue date March 31, 2018

Abstract

This document outlines the design specification and implementation of *SafeLift* and *AppAid* system. It covers in detail, the component choices for each sub-unit, the specifications, justification of the selected design approach, individual component and system test plans, as well as the user interface design.

SafeLift is designed as a kit that can be easily installed on existing forklifts. It consists of a detection module and a warning/feedback module. The detection module includes a RFID system, ultrasonic sensors and a microcontroller unit. The feedback module consists of a speaker and a LCD screen which provide the driver with real time visual and audio warnings whenever an obstacle is within the danger-zone. *AppAid* is a mobile application designed as a tool to track workplace incidents and provide statistical analysis which will identify and mitigate foreseeable risks before they are fatal. Integrated together, the system will help improve workplace health and safety, resulting in increased productivity.

The design specifications in this document will provide the basis for the design and building of a working model. The proof of concept will specifically demonstrate the critical functionality of detection and warning modules. The prototype is aimed to be a more compact and efficient design with lower power consumption and aesthetic design.

Table of Contents

Abstract.....	iii
Table of Contents	iv
List of Figures.....	vii
List of Tables	vii
Glossary	viii
1. Introduction.....	9
1.1. Background	9
1.2. Scope	9
1.3. Intended Audience.....	9
2. System Overview	10
2.1. SafeLift – Forklift proximity alert system	10
2.1.1. AppAid – Risk management app.....	11
3. SafeLift Design Specifications.....	12
3.1. Physical and Operation Requirements	12
3.2. Electrical Requirements	12
3.3. Component Requirements	13
3.3.1. Microcontroller	13
3.3.2. Arduino Requirements.....	13
3.3.3. Raspberry Pi	13
3.3.4. Ultrasonic Sensors	14
3.3.5. RFID System	16
3.3.6. LCD Screen	18
3.3.7. Speakers.....	19
3.4. Software/ Firmware.....	19
4. Engineering Standards	19
5. AppAid Design	20
5.1. General Requirements	20
5.2. Security Requirements	23
5.3. Usability and Reliability Requirements	24
5.4. Operational and Reporting Requirements	24
5.4.1. Incident Reporting	24
5.4.2. Statistics Reporting.....	24
5.4.3. Training Module	25
5.4.4. Qualified Operator/First Aid Attendant Tracker	25

6. Safety and Sustainability	25
7. Conclusion	27
Appendix A: Test Plan.....	28
A.1. SafeLift.....	28
A.1.1. Unit testing.....	28
A.2. AppAid	30
Appendix B: User Interface (UI) Design.....	32
B.1. Introduction	32
B.2. User Analysis.....	32
B.2.1. SafeLift	32
B.2.2. AppAid.....	32
B.3. Task Analysis	33
B.3.1. Discoverability	33
B.3.2. Feedback	34
B.3.3. Conceptual model	34
B.3.4. Affordance	34
B.3.5. Signifiers	35
B.3.6. Mappings.....	35
B.3.7. Constraints	37
B.4. Engineering Standards.....	37
B.5. Usability Testing.....	38
B.5.1. Analytical Testing	38
B.5.2. Empirical Usability Testing	39
B.6. Graphical Representation	39
B.6.1. SafeLift	39
B.6.2. AppAid.....	41
B.7. Conclusion	42
Appendix C – Planning Appendix	43
C.1. Introduction	43
C.2. Scope	43
C.3. Risk and Benefits.....	44
C.3.1. Risks.....	44
C.3.2. Benefits	44
C.4. Market Competition.....	45
C.4.1. Competition	45
C.5. Personnel Management.....	46

C.6. Time Management.....	47
C.6.1. Gantt Chart for May – August 2018	47
C.6.2. Major Milestones	48
C.7. Budgetary Management.....	49
C.8. Conclusion.....	50
References	51

List of Figures

Figure 1: Blocks Diagram of Modules.....	10
Figure 2: <i>SafeLift</i> stages of operation	11
Figure 3: Position of components	11
Figure 4: Proof of Concept Electrical connections of sensors with Arduino	12
Figure 5: Arduino connected to Raspberry Pi, with a 10-inch LCD HDMI display	12
Figure 6: Model of how objects reflect sound waves[3].....	14
Figure 7: HC-SRO4 pins.....	15
Figure 8: Solidworks model of sensor casing	16
Figure 9: RFID system operation[5]	16
Figure 10: Linear polarized antenna waves [cite].....	17
Figure 11: Circular polarized antenna waves [7].....	17
Figure 12: UHF RFID module rs-232/ttl 902-928mhz	18
Figure 13: 10.1" LCD screen	19
Figure 14: Example of speaker for SafeLift	19
Figure 15: Device Fragmentation of iOS and Android [12]	21
Figure 16: Connection between database and webservice and webservice and app	22
Figure 17: Diagram of how components interact in MVC model	22
Figure 18: diagram of how components interact in MVVM model	23
Figure 19: Sample of LCD screen user interface display	39
Figure 20: (a) The login screen (b) The home screen (c) Home screen with a notification	41
Figure 21: (a) Incident stat UI (b) Incident reporting form (c) Forklift training UI	41
Figure 22: (a) UI for qualified operator list (b) UI for first aid attendant list (c) Floor plan UI.....	42
Figure 23: Gantt char for May – August 2018.....	47
Figure 24: Timeline that includes major milestones for the next four months.....	48
Figure 25: Example of active RFID that may be used.....	49

List of Tables

Table 1: Arduino Specifications	13
Table 2: Raspbery pi model B+ specifications	14
Table 3: HC-SR04 Specifications with satisfied requirements.....	15
Table 4: Differences between active and passive tags.....	17
Table 5: UHF RFID specifications with satisfied requirements.....	18
Table 6: Specifications of the speaker	19
Table 7: Breakdown of the duties for the SafeLift	47
Table 8: Breakdown of duties for AppAid	47
Table 9: Specifications of the active RFID tag.....	49
Table 10: Cost break down of parts from first four month	49
Table 11: Estimation of the cost break down for next four month	50

Glossary

RFID	Radio Frequency Identification
OS	Operating System
App	Application
API	Application Programming Interface
SOAP	Simple Object Access Protocol
REST	REpresentational State Transfer
MVC	Model View Controller
MVVM	Model View ViewModel
UI	User Interface
RH	Relative Humidity

1. Introduction

1.1. Background

Human loss and pain is immeasurable. A fatal workplace accident can change the lives of families, friends and even the employers of the victim forever. Additionally, occupational injuries can also be a major financial burden on the employer. A proper safety system that is put in place will ensure the reduction of such disastrous outcomes. With the help of *PowerLift Safe Solutions*, organizations can provide its employees with a safe work environment, which every worker deserves. Furthermore, the health and safety of workers will result in better productivity and efficiency at all levels.

Forklifts are one of the most used heavy machinery in workplaces that handle and transport heavy materials. Their power has made them a vital tool in ensuring efficiency and faster productivity in these workplaces. Owing to their size and weight, forklifts are among the most hazardous vehicle types in the workplace. A standard Clark CMP25 forklift weighs approximately 8,254 pounds and on average can carry a maximum load of 5000 pounds stacked up to 9.3 feet high on its forks [1]. Forklift operation environments include: pedestrians, blind spots and narrow aisles. Unfortunately, the dangers associated with them are greatly underestimated due to a lack of proper training and safety policies. Even at low speeds incidents involving forklifts are usually very serious and fatal.

PowerLift provides a practical solution that is reliable and economical in improving workplace safety of businesses that own and operate a forklift. The product has two components; *SafeLift* and *AppAid*. *SafeLift* is a hardware kit installed on the forklift, designed to give audible and visual warnings to the operator whenever pedestrians or obstacles in the vicinity of collision with the forklift. *AppAid* is a mobile application used by employers to provide training, keep forklift maintenance records, track workplace incidents and provide statistical analysis to help in preventing future accidents.

1.2. Scope

This document is a continuation of the previous document, Functional Requirements for *SafeLift* and *AppAid*. It provides a detailed description of design specification for *SafeLift*, the software solution *AppAid* and the housing designs. This document provides design decisions for the proof of concept as well as the prototype. Functional Test plans and User Interface Design for products of *PowerLift* are outlined in Appendix A and B of this document, respectively.

1.3. Intended Audience

This document is to be used by the engineers at *PowerLift Safe Solutions* for the development of *SafeLift* and *AppAid* as well as by the TAs marking this document and any potential buyers interested in the products. The engineers can refer to this document at any time during the research and development stages of the prototype and the final product. Engineers responsible for performing quality assurance can refer to Appendix A of this document directly as well as safety and sustainability sections of the report to ensure all safety concerns have been addressed and that the product fulfills all requirements and meets all expectations for proper usage.

2. System Overview

2.1. SafeLift – Forklift proximity alert system

The system consists of:

- two microcontrollers,
- five ultrasonic sensors,
- an RFID reader integrated with an antenna,
- an LCD screen and
- one set of audio speakers

The system is divided into a detection module and feedback module which are connected through the microcontrollers as shown in the block diagram below:

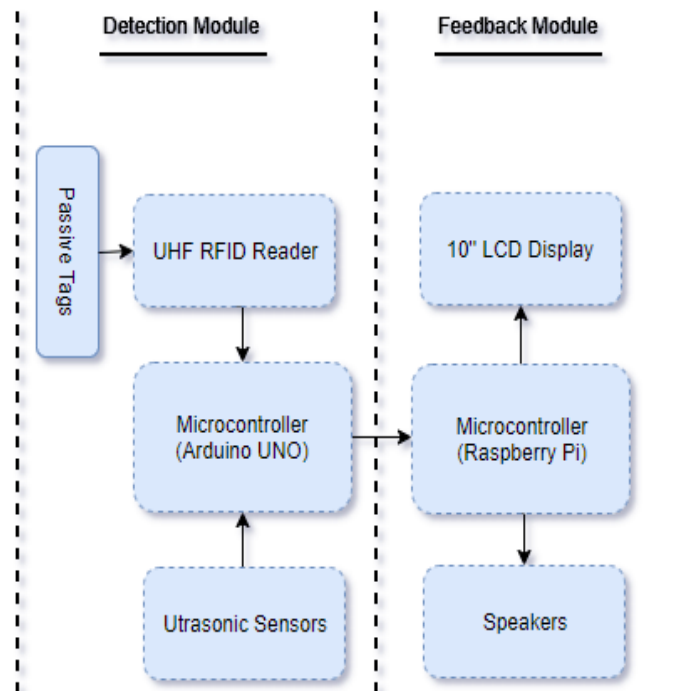


Figure 1: Blocks Diagram of Modules

The RFID reader will be programmed to do continuous reading every 100 ms whenever a tag is within range. The Arduino processes the data and sends it to the Raspberry pi which outputs the results on the LCD screen to display the number of tags detected for every read. The ultrasonic sensors continuously send waves which will be reflected by obstacles in the detection range, the data is then processed by the microcontrollers and the distance is displayed on the LCD screen.

The LCD screen which refreshes every 100 milliseconds will display the distance measured by each sensor separately to alert the driver of the general obstacle direction. Distance will be displayed in meters/centimeters and will be color coded to differentiate distance of obstacle. Green indicating obstacles at least 2.5 meters away

and red indicating less than 2.5 meters out. The speakers will have a unique sound with adjustable volume, that beeps when the obstacles are 2.5 meters away then turns into a continuous sound less than 2.5 meters out. The diagram below shows the stages of operation:

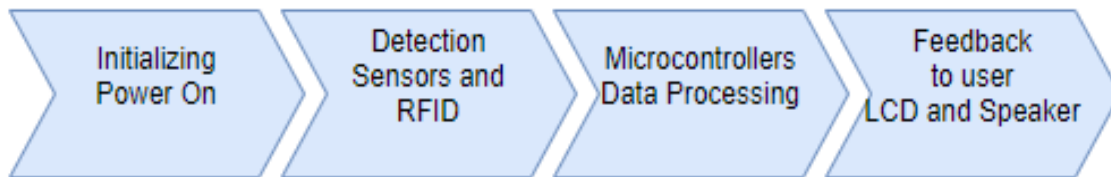


Figure 2: *SafeLift* stages of operation

The diagram below shows the positioning the components on the forklift:

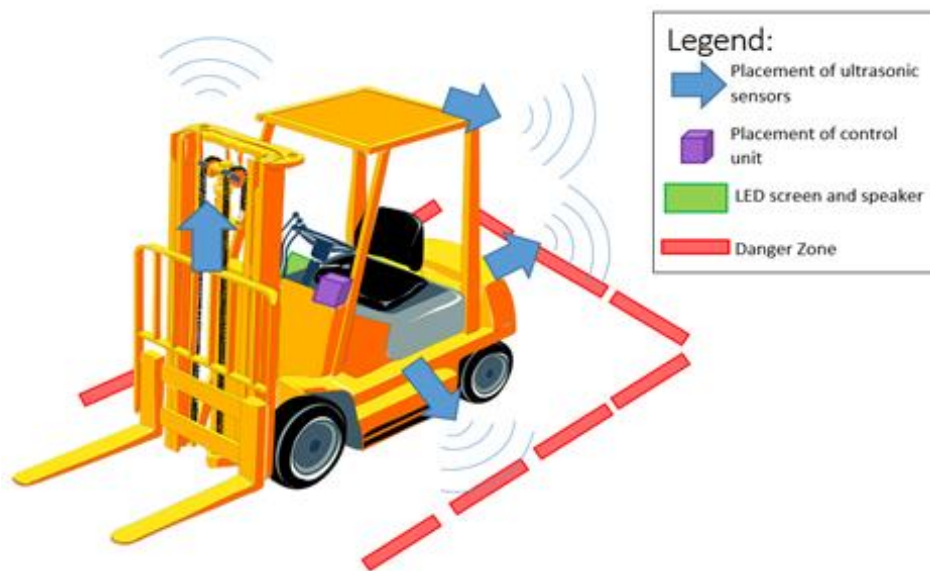


Figure 3: Position of components

2.1.1. AppAid – Risk management app

AppAid will be designed with extendibility and scalability in mind. The framework will be developed in a way that can be easily updated in future phases. There are 4 basic, logical components of the system: The Database Engine, the Class Library, the Server Procedure Proxy, the Server Application, and the Client Applications.

- Database Engine - Hosts the backend database which is used for central data storage.
- Class Library - Java package which holds all Java classes that both the server and client applications use to represent the data in memory. Most updates and additions will require only updating this library.
- Server Application - Implemented in Java. It provides methods and procedures that can be invoked remotely by a client application via to retrieve, update or generate data.
- Client Applications - Implemented in Java. Contains all presentation logic and interacts exclusively with the user. It uses the class library package defined above.

3. SafeLift Design Specifications

This section gives the detailed specification and justification of the components selected according to the functional requirements of the system.

3.1. Physical and Operation Requirements

The physical design consists of the component casing and mounting which are discussed and illustrated in the respective component sections below. The casings are designed to be as compact as possible and the mountings designed not to interfere with the driver's normal operation.

3.2. Electrical Requirements

The figures below show the circuit diagram for sensor unit the proof of concept. The input from the sensors are processed by the Arduino and sent to the Raspberry pi via a USB connection for feedback on the LDC display.

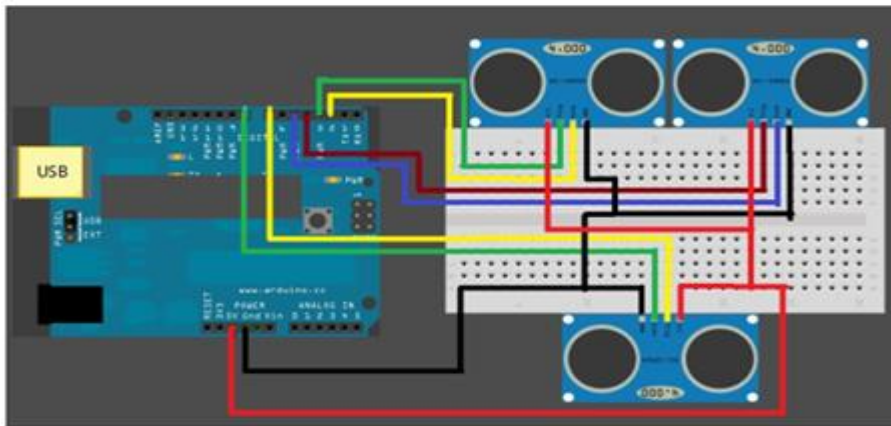


Figure 4: Proof of Concept Electrical connections of sensors with Arduino

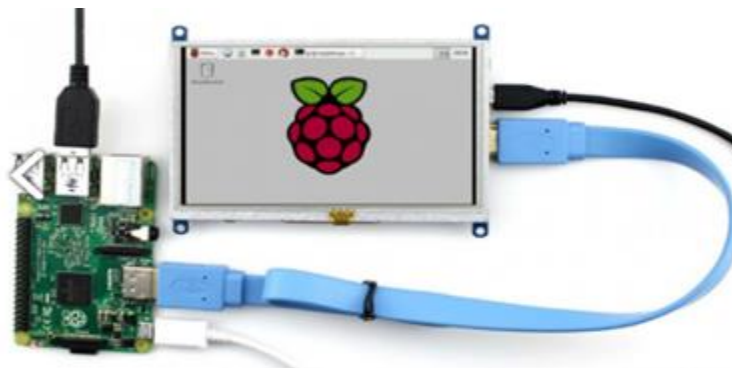


Figure 5: Arduino connected to Raspberry Pi, with a 10-inch LCD HDMI display

All the components selected have an operating voltage of 3.3V- 5 V (Req 3.3.3).

3.3. Component Requirements

3.3.1. Microcontroller

During the development phase, we will use two different microprocessors; Raspberry Pi 2 Model B+ and Arduino Uno. We are using two microprocessors because we predict that the output will take up a considerable amount of processing power of the Raspberry Pi. The Arduino will be used for taking sensor data and will transmit this data to the Raspberry Pi via USB connection. Once we have a working prototype we can use it to perform benchmark test and obtain actual data on resource use. This will then be used to determine the microcontroller that will be used in the final product.

3.3.2. Arduino Requirements

The Arduino will be used to control the sensor module in the initial development phases instead of the Raspberry Pi because the sensor, we predict, will not take a significant amount of resources compared to the screen output. Arduino provides an IDE, which will make it easy for rapid development. We would not have to learn how to program a microcontroller, instead we can work on the functionality. The Arduino will pull the input data from the RFID reader and ultrasonic sensors. Once it receives some data, it will send the data to the output module, the Raspberry Pi via Bluetooth. The Arduino will send information to the Raspberry Pi every second.

Table 1: Arduino Specifications

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

3.3.3. Raspberry Pi

Raspberry Pi Model B+ will be used as the microprocessor for the output module of *SafeLift* during the proof-of-concept stage. We believe that the graphical output will require more resources than the Arduino will be able to offer. Moreover, we separated the output and input into different modules because we want the process to be parallel. We chose Raspberry Pi because it is easy to use and does not require microprocessor programming. The Raspberry Pi will receive information from the Arduino and display the output onto a screen and play a sound through a speaker to alert the operator.

Table 2: Raspberry pi model B+ specifications

SoC	Broadcom BCM2836 (CPU, GPU, DSP, SDRAM)
CPU	900 MHz quad-core ARM Cortex A7 (ARMv7 instruction set)
GPU	Broadcom VideoCore IV @ 250 MHz
More GPU info	OpenGL ES 2.0 (24 GFLOPS); 1080p30 MPEG-2 and VC-1 decoder (with license); 1080p30 h.264/MPEG-4 AVC high-profile decoder and encoder
Memory	1 GB (shared with GPU)
USB ports	4
Video input	15-pin MIPI camera interface (CSI) connector
Video outputs	HDMI, composite video (PAL and NTSC) via 3.5 mm jack
Audio input	I ² S
Audio outputs	Analog via 3.5 mm jack; digital via HDMI and I ² S
Storage	MicroSD
Network	10/100Mbps Ethernet
Peripherals	17 GPIO plus specific functions, and HAT ID bus
Power rating	800 mA (4.0 W)
Power source	5 V via MicroUSB or GPIO header
Size	85.60mm × 56.5mm
Weight	45g (1.6 oz)

3.3.4. Ultrasonic Sensors

Ultrasonic sensors are commonly used in wide variety of proximity and distance measuring applications. They transmit high frequency sound waves towards a target which reflects the waves back to the sensor. The time it takes for the echo to return to the sensor, T , is measured and using the speed of sound (343m/s at room temperature) the distance of the object is determined[2].

$$Distance = Speed\ of\ Sound * T$$

where T is half the time to trasmit and receive

The object may either be at an angle or maybe directly in front of the sensor. The figure below shows how the sensor waves are transmitted to and reflected by objects.

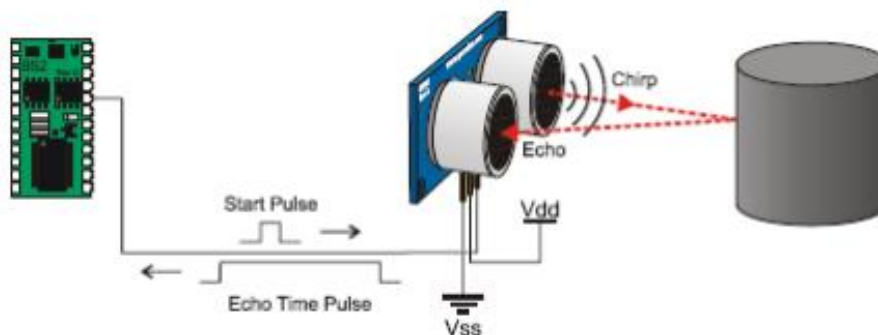


Figure 6: Model of how objects reflect sound waves[3]

For our design we chose the ultrasonic sensor HC-SR04 which is a high precision and low-cost sensor. The reason being that it is easy to use with Arduino the microcontroller in our design.



Figure 7: HC-SR04 pins

1. VCC- Connects to 5V power
2. Trig – sends pulse for sensor to go into ranging mode for object detection
3. Echo – sends a signal back, if an object has been detected
4. GND- completes electrical circuit

The table below shows the specifications of HC-SR04 Ultrasonic Sensors.

Table 3: HC-SR04 Specifications with satisfied requirements

Power Supply:	+5V DC	Req 3.4.2.3
Quiescent Current	< 2mA	Req 3.4.2.3
Working Current:	15 mA	Req 3.4.2.3
Effective Angle:	<15°	Req 3.4.2.6
Ranging Distance:	2 cm - 400 cm -	Req 3.4.2.1
Resolution:	0.3cm	
Measuring Angle:	30°	Req 3.4.2.6
Trigger Input Pulse width:	10uS	
Dimensions:	45mm X 20mm X 15m	Req 3.4.2.5

The casing for the will be as shown in the Solidworks model below.

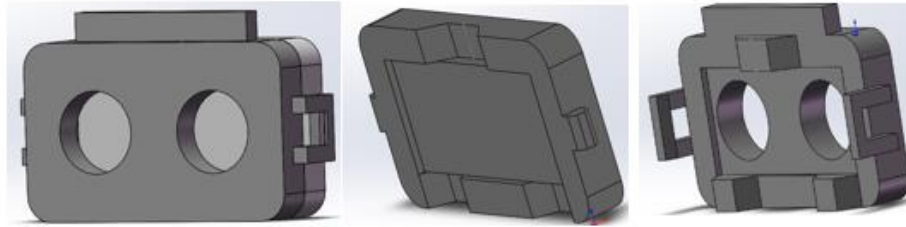


Figure 8: Solidworks model of sensor casing

3.3.5. RFID System

RFIDs operate within distinct frequency bands set by international and national standards. Generally, the most common frequencies are [4]:

- Low Frequency (LF): 125 – 134 kHz, offering data communication that is limited in data rate less than 1 kbits/s and read range of less than 1 m
- High frequency (HF): 13.56 MHz with data rates approaching 25 kbits/s and a read range of up to about 1.5 m
- Ultra-high frequency (UHF): 433, and 860-960 MHz with read ranges of up to 10 m and data rates up to about 100 kbits/s

The *SafeLift* system will use the UHF RFID (Req 3.4.3.1) system which typically consists of a reader, antenna and tags. The reader, through the antenna generates and sends continuous electromagnetic waves which hits the tags. The tags become energized from these waves and send back a signal transmitting the encoded data to the reader[5].

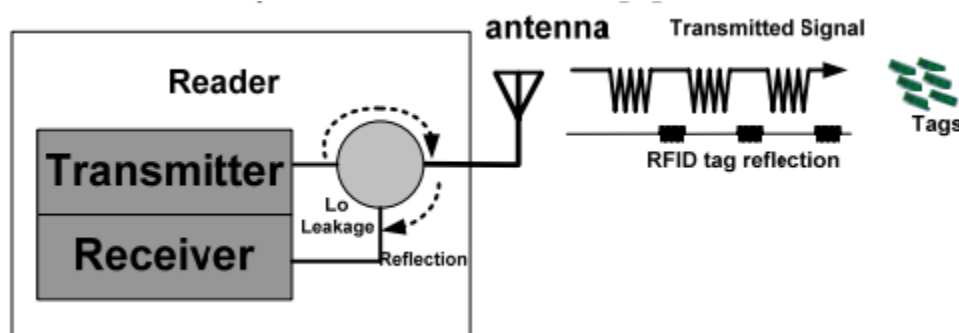


Figure 9: RFID system operation[5]

The two main types of RFID tags are active and passive tags. Active tags have a battery in them and emit a signal without being activated by the reader, so they emit a stronger signal than passive tags thereby increasing the read range [6]. Passive tags are generally smaller and significantly lower in cost with a much longer lifespan and therefore chose for our system (Req 3.4.3.3). Table 4 below highlights some of the major differences between active and passive tags.

Table 4: Differences between active and passive tags

Attribute	Active	Passive
Signal strength	Stronger	Weaker
Signal availability	Always on	Activated by reader
Size	Larger	Smaller
Initial Cost	High	Low
Maintenance	Replaced every 2-3 years	Indefinite lifetime

Antennas are categorized into two main groups; circular and linear, which refers to the polarization of the signal emitted [6]. The choice between linear polarization antennas and circular polarization antennas has a major impact on the detection capability of the RFID system. Linear polarization occurs when electromagnetic waves broadcast over a planar symmetry, horizontal or vertical. The figure below shows a horizontal linear polarization of the antenna.

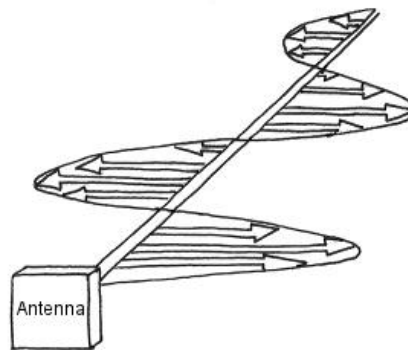


Figure 10: Linear polarized antenna waves [cite]

Linear polarized antennas have a wide range of detection due to their concentrated emission. They however require the tag to be at the same level of the emission in order for it to be detected. Circular polarized antennas, on the other hand have two-planar electromagnetic wave emission completing an entire rotation per wavelength. Shown below is the corkscrew-like emission pattern of a circular polarized antenna (Req 3.4.3.10) [7].

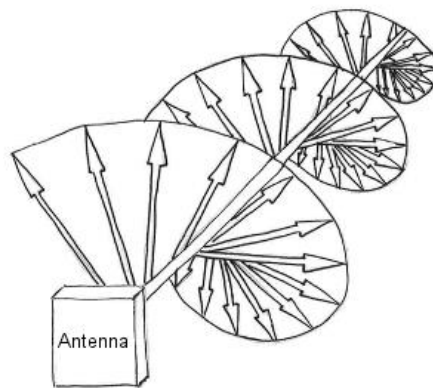


Figure 11: Circular polarized antenna waves [7]

Due to their circular pattern, they do not restrict the tags to be planar with respect to the antenna. However, circular polarized antennas have a shorter range as compared to linear polarized antennas since they lose about 3dB in splitting their power into two planes. Circular polarized antenna is chosen for our RFID systems since the tag orientation will not be reliable or consistent.

The UHF RFID shown below was the chosen, its specifications are shown in table below. It has multiple tag identification and a read distance of up to 6m (Req 3.4.3.2) with an 8dBi circularly polarized antenna.



Figure 12: UHF RFID module rs-232/ttl 902-928mhz

Table 5: UHF RFID specifications with satisfied requirements

Input Voltage	DC 3.5V - 5V	Req 3.4.3.6
Air Interface Protocol	ISO 186000-6C	Req 3.4.3.1
Spectrum range	902MHz – 928MHz	Req 3.4.3.1
Output power	18-26 dBm	
Operating current	180mA @3.5V (26 dBm Output), 110mA @3.5V (18 dBm Output)	Req 3.4.3.9
GPIO	2 inputs, 2 outputs (3.3V TTL level)	
Baud rate	115200 bps	
Host communication	TTL UART port, Wiegand 26, Wiegand 34	Req 3.4.3.11

3.3.6. LCD Screen

The LCD screen chosen is a 10.1" Capacitive 1024x600 LCD Screen (Req 3.4.4.1) shown in the figure below. It is a multicolor graphic LCD (Req 3.4.4.4) which is compatible with our Raspberry pi microcontroller without needing to write drivers for it.



Figure 13: 10.1" LCD screen

3.3.7. Speakers

SafeLift will use speakers for audio warnings for the forklift operator in case he is approaching an obstacle or pedestrian. Our choice of speaker was JETech Portable Mini Wireless Bluetooth 4.0 Speaker. The main reasons for this choice are:

- the small size of the speaker – so the system is compact
- high definition sound - so that the sound is heard above all noises
- Wireless - so that the system is not cluttered



JETech Bluetooth Speaker S2020 *1

Figure 14: Example of speaker for *SafeLift*

Table 6: Specifications of the speaker

Item Weight	136 g
Product Dimensions	4.3 x 4.3 x 4.7 cm
Item Model Number	2020-SPEAKER-BT-BK
Connectivity	Wireless, Bluetooth

3.4. Software/ Firmware

In the proof of concept, we will be using Arduino for the distance and detection data. The programming language used will be C/C++, whose standards and libraries are supported by Arduino. For communication between the Arduino and Raspberry pi, we will use simple serial communication over a USB cable with the programming done in Python. In displaying and updating the LCD screen a python script will be used since Python offers very easy to use libraries for graphical user interface design

4. Engineering Standards

All the electrical components of *SafeLift* will adhere to C22.1-15, the Canadian electrical code for safety standard for electrical installations [8]. This standard will ensure all the components operate at their intended voltage levels and wiring are safe to the users.

The UHF RFID system will adhere to the following standards [9]:

- ISO/IEC 18000-6 - Parameters for Air Interface Communications at 860- 960 MHz UHF Class-1 Generation-2
- ISO/IEC 15962 - Information technology -- Radio frequency identification (RFID) for item management
- ISO/IEC 15962 - Deals with the processing of data and its presentation to the RF tag, and the initial processing of data captured from the RF tag
- ISO/IEC 15963 - Describes numbering systems that are available for the identification of RF tags
- ISO/IEC 18046 - RFID tag and interrogator performance test methods
- ISO/IEC 18047 - RFID device conformance test methods

AppAid will adhere to the following security standards:

- ISO/IEC 18033-3[10] – Details encryption systems for data confidentiality
- ISO/IEC 27001[11] – provides requirements for an information security management system

5. AppAid Design

5.1. General Requirements

Originally Req 6.1.1 required us to deliver the proof-of-concept version of AppAid on Android Platform but after further research we have decided to switch to develop the iOS version first and then release an Android version. This means Req 6.1.1 will be changed to be available on iOS first. Req 6.1.7 will change to be available on Android. Both requirements will be satisfied as we will eventually release on both OS. Although Android has the largest market share in the world, 86.2% in Q2 2016[12], iOS is the shares market leader in the US and Canada. Android devices have a huge amount of device fragmentation, whereas iOS has a small amount. This fragmentation on Android means we must spend more time ensuring that our product is compatible with largest amount of Android OSs which is not the case for iOS. As seen in the chart below, developing the app on the newest version of iOS, would automatically make it compatible with 90% of the iOS devices satisfying Req 6.1.2. To do the same on Android we will target development on Kit Kat, which is version Android 4.4.

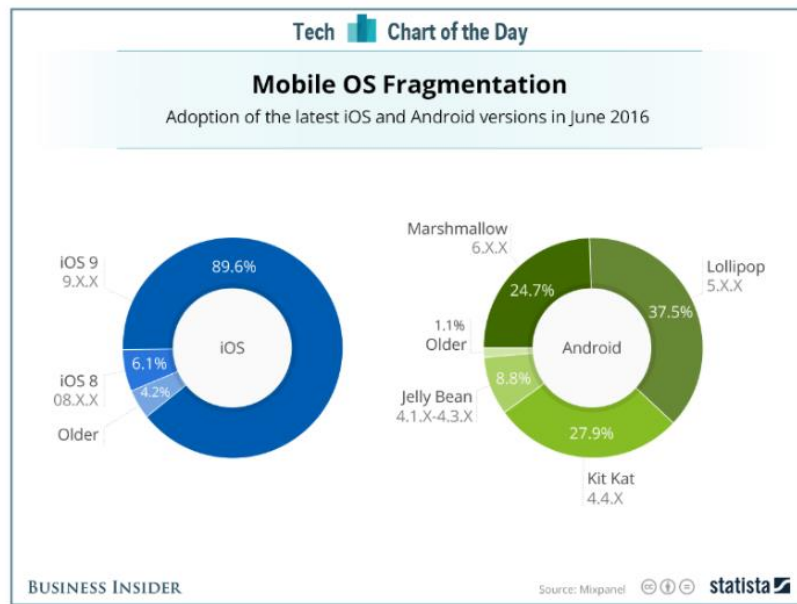


Figure 15: Device Fragmentation of iOS and Android [12]

Our users will be required to sign in to the app, which fulfils Req 6.1.3. The user interface of the login page can be seen in the UI appendix.

Once the user has been successfully authenticated, they will be allowed access to feature they have privileges for. We will accomplish this by storing the user privilege level in the database alongside their login info. Some of the possible features will not show up unless they have the required authorization. In the database we will have a table for each feature that requires authorization. Inside these tables we will store the roles that are authorized to use the feature. There will have a service called `AuthenticateUser`, which will determine if a user has permission to use the feature. The service will contain a function called `AuthenticateActivity` which will take in the name of the activity that needs to be authenticated. This function will call another one called `GetUserActivities` which will take in the current user of the app as a parameter. It will retrieve the role of the user and then retrieve a list of activities the user is authorized for from the database. This will allow us to satisfy Req 6.1.5 and Req 6.1.6. To make this feature as flexible and user friendly as possible, the employer will be able to create roles, as well as set the authorizations they have.

This login data will be stored in a database that we will be hosting and providing in the final product. Once the user has logged in once, they can setup an optional finger print login, as per Req 6.1.4. We will be using Android's `android.hardware.fingerprint` and iOS's Local Authentication API to leverage the finger print reading functionality of the respective OSs. This finger print data is only accessible to the finger print reading device. No external source will be able to access this raw data. This means that we cannot store the finger print data it in a database and check against it. The finger print device is essential a black box, but we can create a keychain that will allow the user to log in to our app once, the device ownership has been authenticated. This is done on iOS with the `KeyChain` API. The database will be connected to a web service that our app will connect to obtain required data from. Below is a diagram showing how the app will be connected to webservice.

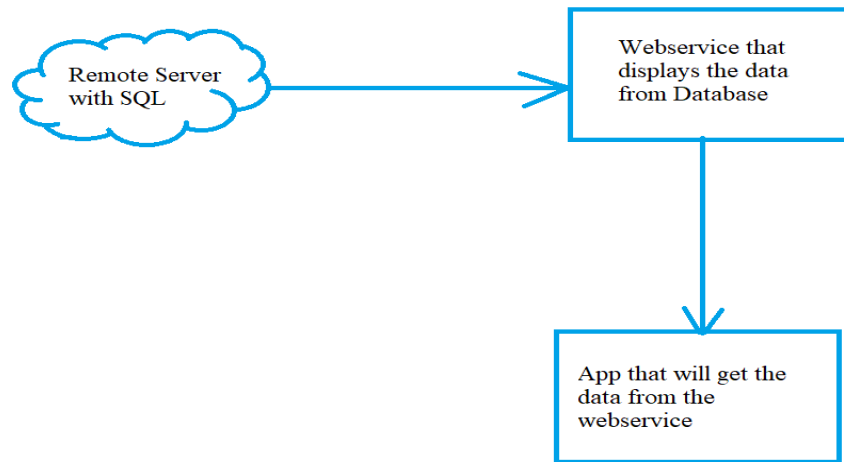


Figure 16: Connection between database and webservice and webservice and app

The webservice will be programmed in Java and we will be using the Jersey RESTful API in the webservice. The RESTful API, which is based on REST, representational state transfer, is preferred to other API such as, SOAP because REST requires less bandwidth [13]. This is how our app will handle HTTP request sent from the app. This is much more suitable for internet usage.

For all development phases of the app we will be using MySQL because the community edition of the database is free and has a great supporting community behind it. It is also easy to set up with a low learning curve, allowing us to rapidly develop the app.

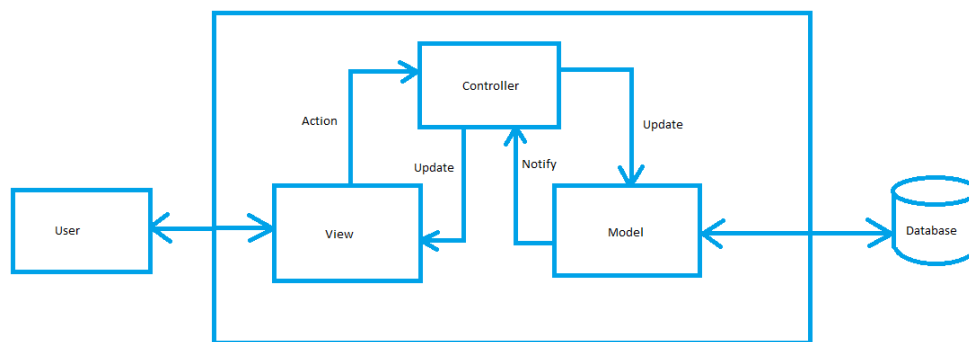


Figure 17: Diagram of how components interact in MVC model

The diagram shown above, is the architecture of the web service. The app will follow a MVC architecture design. User, in this case, the app will interact with the view component of the web service. The web service notifies the controller of the action the user wants to do. The controller will then update the model, which can be request for data or updating information in the database. The view will that be updated by controller, after it finishes updating the model.

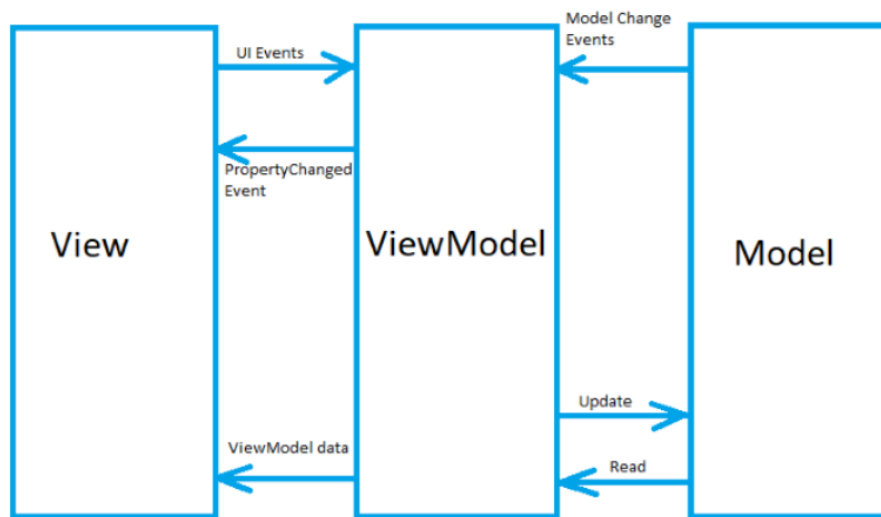


Figure 18: diagram of how components interact in MVVM model

The app will have the architecture that is shown above, which is a MVVM architecture. The model in this case is any data the app receives from the back end, or any other data that might be needed. The user of the app will interact with the view. The view will notify the viewmodel of the user interaction. The viewmodel will then update the model. This includes updating the data or sending request to the backend for information or updating the database. After the model has been updated, viewmodel will notify the view of any changes, and the view will be updated.

5.2. Security Requirements

Our app will provide various security features to make sure that the user's data is secure. During the proof-of-concept stage, our app will provide automatic log out feature. This feature will log the user out of the app after half an hour of inactivity. Inactivity means the user is not using a forklift and has not touched the screen for the past thirty minutes. The app will start a timer if the user has been inactive for ten minutes straight. Once this timer has reached zero, the app will log the user out. This is to prevent unauthorized access incase the user forgets to log out or loses their phone. This feature allows us to satisfy Req 6.2.2.

To satisfy Req 6.2.3, we will make sure our users' passwords are secure, we will be encrypting the password by hashing it with a salt that is randomly generated for each individual user. The library we will be using for this is jBCrypt, which is a Java implementation of BCrypt. BCrypt is a one way salted hash function based on the Blowfish Cipher [16]. We are using BCrypt because it has updatable iterations. This means we can easily change the number of times the cipher runs to compensate for devices getting faster as technology develops. We want the cipher to run slowly because it allows us to prevent brute force attacks.

Finally, to make sure the data of our clients are secure, and satisfy Req 6.2.1, the database will be encrypted. But instead of encrypting the at the database level, we will be encrypting with AES before we inject the data into the database. By encrypting at a higher level, the app can achieve much higher level of security then if we were to encrypt at the database level [17].

5.3. Usability and Reliability Requirements

As per Req 6.3.4, AppAid should be able to handle unexpected inputs. To do this we will use try catch statements in Java to handle unexpected results and in iOS we will use input sanitizing. Sanitizing the input means checking if the input is valid before using it. If the input is valid but not in a standard format, we will put it into a correct format and then use it. If it is an invalid input, we will report the error to the user instead of using it. The try catch statement will serve a similar purpose. Instead of checking if the input is valid first, we directly use it. If an error occurs we will catch it and then report the error to the user in easy to understand language, instead of the error output that is hard to understand.

AppAid will also prevent the user from losing data when crashing, as per 6.3.3. To do this we will have code that store what is currently already filled in on the form into a file that will be stored on the phone locally. After the app is restarted the user will be able to load the work back into the app. AppAid will also store the data when the app fails to submit information to the database. AppAid will then try to send the submission again when the connection to the database is restored. This will prevent the user from having to fill out a form again in case they lose internet connection.

5.4. Operational and Reporting Requirements

5.4.1. Incident Reporting

The incident reporting module is key to the success of app aid. As per Req 6.4.1, we will make the form easy to fill out. The fields of the form will be standardized to a form that will allow us to easily map it to a database and store it. This will also allow us to do statistical analysis without difficulty. To do this, we will be using UI elements like, drop down list and check boxes. A mockup of the reporting form is shown in figure 21(b).

To ensure that the form is filled out correctly, AppAid will check over the form to make sure all relevant data is filled in and provide basic error checking. For example, if a field requires a number, the app will let the user know he cannot input any character that is not a number. If a field is not filled in, the app will highlight the field. After the user clicks submit, the app will prompt the user, if they would like to check over the form to make sure there are no errors. At this point the user can chose to go back to correct any errors or continue. If they chose to continue, the form will be submitted to the remote server. After the remote server receives the data, it will be injected into the database. If at any time the app loses internet connection, or something goes wrong, our app will notify the user.

To meet Req 6.4.2, the app will fill out a pdf form from WorkSafeBC whenever a user requests an incident investigation form. This will be done using Apache PDFBox. The app will also allow workers and supervisors to view previous reports.

5.4.2. Statistics Reporting

Statistics report will allow the user to select the type of chart, and the statistics they want to see in charts. The charts will be made using Apple Charts API on iOS and on Android, we will be using the open source library, MPAndroidChart. Once the user has selected the statistics, they want to see, the app will retrieve the required information from the database, through the webservice. The app will then use the respective chart drawing API of the OS, to draw the charts. Figure _ show a mockup of the statistics reporting UI. This satisfies both Req 6.4.3 and Req 6.4.4.

5.4.3. Training Module

The third module of AppAid is a training module, as described by Req 6.4.5, contains relevant training infographics, and other documents. This information is maintained by the employer. The employer will be able to provide the employee with the newest information by adding a link to the resource, or by downloading it and then uploading it to the add. Any text documents, pictures, or PowerPoints can also be uploaded. By allowing the employer to upload the training material, we allow them to customize the training for their employees. The uploading feature will only be available to employers, or any role the employer grants authorization to. The employer will also be able to delete any irrelevant or outdated materials.

As per Req 6.4.6, Req 6.4.7, and Req 6.4.8, the module will track, monitor, and schedule bimonthly training. The database will have a table, called Resource_Progress, that will keep track of the progress of each resource provided by the employer. When the user closes the app, the app goes into the background, or finish with a resource, the app will insert into the table, the resource name, percent completion of the resource, and the current date. In the case of a document, the app will save data every time the user goes to the next page. Once the resource has reached one hundred percent, it will be marked complete. Another table, Overall_Progress, will keep track of the overall progress for each operator. To keep this table updated, we will have a database trigger that calculates a moving average for the total number of resources. The trigger will be placed on the Resource_Progress table, and it will be triggered every time the table is updated. Once the overall progress has reached 100%, the employee will have completed the training. To notify the employee we will be using a pop-up message.

To keep track of the last training session, the app will look in the Overall_Progress table. It will retrieve the latest date entry for an employee, that has 100% completion, and then compare it to the current date. If the period is greater than two months, a training session will be scheduled.

5.4.4. Qualified Operator/First Aid Attendant Tracker

AppAid will track the qualified operators and first aid attendants as per Req 6.4.9. To accomplish this, we will have two tables in the database, one called Qualified_Operators and the other First_Aid_Attendant. The table will also store information about number of incidents for each operator to satisfy Req 6.4.10. The app will update the local database each day after twelve am local time, or if an employee is removed from the tables. The app will also update if the employee is starts up the app after closing it. To track which employee is using which vehicle, the app will require an employee to sign out a vehicle before they use it. This will satisfy Req 6.4.11.

6. Safety and Sustainability

At Powerlift Safe Solutions emphasis is placed on ensuring that the product as sustainable and safe for the use. Powerlift Safe Solution is designed to improve the safety of workplace environments that incorporate the use of forklifts. Powerlift Safe Solutions will emphasize “cradle-to-cradle” design. Every product will be made from sustainable materials that can easily be recycled after the product’s life cycle. Since this product is designed to interact with clientele, extensive measures must be taken to ensure the safety of the user. During sustainability and safety analysis, emphasis on reliability will be placed on the various products components to ensure that the product works according to design. Failure to do so would result in injury since the product is heavily based on safety. A list of safety requirements includes:

- Secure fastening of devices to avoid injuries
- The device should not collect excessive electrostatic charge that is harmful to humans and electronic circuits inside the device.
- All electrical devices/components shall be encased properly.
- Device should be non-flammable, non-explosive, and easily transported.
- Wires and board shall be properly insulated and grounded to prevent shocks and electrical fires
- The unit must be resistant to different physical forces such as being dropped or getting hit
- All components must withstand operation temperature, humidity, and currents
- The unit should not overheat under continuous usage
- The electronic components shall not cause interference with other devices

Due to safety concerns for the user base, non-toxic materials have been chosen as well as materials that are free of any lead contaminations. This will minimize any harmful exposures and make the product as safe as possible. Due to the passive nature of the RFID tag as it does not have an in-built energy source, it emits no waves and poses no danger. The ultrasonic sensor will emit a low working frequency of 40Hz [5] which will pose no danger to the user. AppAid will include a secure login, a login time out and the encryption of data using a hash code.

Regarding the ultrasonic sensors and RFID, they will not be used or stored in vacuum or explosion prone areas, vaporous areas, areas at temperatures exceeding the rated range as well as places with radical convection. This is so that the reliability and the life the sensors and RFID do not decrease

Another very important aspect of sustainability is the environmental footprint of our product. Environmental standards CSA-C22.2 No. 107.2.01- Battery Chargers and CSA-C22.2 No. 0.23-15 – General Requirements for Battery-Powered Appliances standards. These standards are applied to stationary battery chargers and battery powered devices for indoor and outdoor use as well as portable and mobile charges. Since batteries are the most common form of hazardous waste, an implementation is needed to mitigate the disposal of batteries after the end of their life cycle. This will be done by including workforce and competency training in AppAid.

7. Conclusion

SafeLift, in combination with AppAid is a robust, durable, compact and powerful system for reducing forklift injuries. By using both RFID and ultrasonic sensors mounted on strategic positions on the forklift, detection of bystanders and obstacles will be possible. AppAid will provide forklift drivers with infographics and other training sessions as well as remind operators when they need re-training. The system design consists of three major sections:

- Software development aimed to develop an application to keep track of workplace injury and help retrain forklift operators
- Hardware development aimed to provide a robust, durable, and compact solution for reducing forklift injuries and workplace accidents.
- Firmware development aimed to enable communication between the microcontroller and the RFID components.

To minimize scheduling slippage, the prioritization system will be implemented to allocate resources efficiently. The focus will be on basic functionality of the product, this includes detection of obstacles using the RFID and ultrasonic sensor. A basic functionality of the application is also necessary.

In this document, there is clear justification of the requirement specifications for both the hardware and software aspects of the product. Including a detailed outline of the requirements of the electrical, and computational aspects of the product.

Although there are a few products out there that help improve the safety of forklift usage, Powerlift Safe Solutions has created a product that rivals the competition. It is the first product of its kind that combines both a hardware and software component. Powerlift Safe Solutions product will also be one that is of low cost and easy to use. This document has also described and addressed sustainability, safety and environmental concerns. Powerlift Safe Solutions' goal is to ensure a hazard free working environment as well as to ensure the reliability and sustainability of the product and that starts with following the standards that have been outlined above.

Appendix A: Test Plan

This section outlines the testing of the functionality to ensure correct system performance and satisfy the functional requirements. It is divided into three sections

- Unit testing
- System integration testing
- Safety design testing

The unit testing will be done first then the system testing which verify each unit does not affect the other's functionality when integrated together. The safety testing will be done both concurrently with the other tests and as the final testing stage. The comment column below can have one of the following descriptions:

- **PASS** – functionality is as expected.
- **FAIL** – wrong results. Further description of the problem is given of possible reasons for failure.

A.1. SafeLift

A.1.1. Unit testing

A.1.1.1. Ultrasonic Sensors Module

This unit comprises of the ultrasonic sensors and the Arduino.

Test Case	Verification	Comments
Accuracy of the sensing range and width	i) The distance of detection should be consistent for every test trial for each sensor	
Position to avoid interference between sensors - space out sensors at different distances	i) The positioning will be determined at point where the sensors waves don't interfere with each other	
False detection – place objects out of range in front of each sensor	i) The sensor should show no detection unless the object comes into range	
Maximum accurate sensing distance on different obstacle size – place obstacle in the threshold distance for detection for each sensor	i) The distance should match the specification for max distance +/- given accuracy error	

A.1.1.2. RFID System

The RFID system consists of the reader, tags, antenna and the Arduino.

Test Case	Verification	Comments
Tag reading - Place tag within reading range	i. Tag should be read and identified ii. Continuous reading- tag should be read every 3 seconds	

Max read range - Place tag a distance away set to be the maximum detection distance	i. Tag should be read and identified at the maximum set distance	
Tag read in multiple orientations - Place tag in read range and orient it in different angles	i. Tag should be read and identified for each orientation	
Multiple tag reading - Place several tags in read range at once and time how long it takes for all tags to be read	i. All tags should be read and identified. ii. Reading speed –tags read per second, should match specification of reader	

A1.1.3. Feedback Module

The feedback module comprises of the Raspberry Pi microcontroller speakers and LCD display and which notify the driver of any obstacle within the danger zone. The format and accuracy of feedback will be the focus of this testing.

Test Case	Verification	Comments
LCD format display	The LCD should display the appropriate format.	
LCD accuracy at different power levels	The LCD should function correctly and accurately and different power levels.	
LCD redisplay speed and accuracy	The LCD should display the correct general direction and distance of the object it detects.	
Volume level of the speakers- by find the most comfortable level	The volume level should be easily heard by the operator and at a level not to harm his hearing	

A.1.2. System Integration Testing

Once the units have tested individually they will combine to make the whole system with all functional features. Testing will include power consumption, accuracy of all units together and feedback response time.

Test Case	Verification	Comments
Individual Unit Interference - the sensor unit and the RFID system integrated together	Both units should output the accurate detection without corrupting the data of the other	
Sleep Mode	When the system enters sleep mode, the system will enter a low power mode and significantly save electrical consumption. When the user resumes the system, the user will not have to re-issue instructions or wait for the system to re-boot	
Duration of usage after battery is fully charged	Battery charge should be sufficient to allow usage for at least 4 hours on single charge	
System speed - the response time of reporting presence of obstacle	Once there is an obstacle in the detection range or tag in read range the system should warn the driver immediately.	
Power Consumption	The whole system will have low power consumption to allow for at least 4 hours usage of the battery	

Time and ease of installation	The installation of the system will be intuitive and will allow the user to complete full installation under 30 minutes	
System accuracy under different power levels left in the battery	Under different power level the system accuracy will not falter.	

A.1.3. Safety Design Testing

Test Case	Verification	Comments
Wiring enclosure- touch wiring and enclosure at connection points	i. All wiring should not cause any electrical shocks or short circuits when touched	
Overheating- turn on system and measure its temperature after at least 4 hours of operation	i) The units should not overheat, and temperature levels should remain below 40 C	
Edges on enclosures – hold the casings on the edges or corners	i. It should not cause scratching or cuts ii. All components fit well in the enclosure	

A.2. AppAid

Test Case	Verification	Comments
Test the log in	i. Login with correct user name and password. This should log the user name. ii. Login with correct user, but wrong password. This should tell the user the password does not match the user name. iii. Login with user name that does not exist. This should tell the user that the user name doesn't exist.	
Reporting an incident	i. Fill out all required fields and submit the form. The app should ask the user to check over the form and confirm it is all correct. After confirming, the data should be submitted to the database. A message notifying the user the submission was successful should appear. ii. Start same as case one, but this time disconnect the app from internet, by turning off Wi-Fi. When the app tries to submit to the database, it should now fail. The app should display a reason for why the report failed to submit. The app should also let the user know that the app will automatically try to submit the form again later when the connection to the database is available again. iii. Enter an invalid value or select and deselect a form field. The app should highlight the field in red and state the error the user has made. iv. Fill out the field but leave a few fields blank or with an invalid value. Submit the form. The app should notify the user that there are invalid value or fields that have been left blank and highlight those fields in red. The fields should not be reset. All values that have been filled in should still be there.	
Retrieve a previous report	i. Create an incident form and submit it. Open the incident report that was just created. The report should contain the same information as what you reported.	

	ii. Disconnect phone from Wi-Fi. Open the report from case i. The app should not be able to open this report	
Incident Statistic	i. Request an incident report for each possible type of report. Make a few incident reports if there are none. The reports should properly reflect what has been submitted.	
Training Module	i. Log in as an employer or other qualified role. Submit a training resource. It should now appear under training materials page. ii. Open any training material. The training material should be what it says. To check this open the original file of the same name and see if contents are the same.	
Qualified Operator/ First Aid Attendant Tracker	i. Open the qualified operator UI, it should report the correct list operators. Compare this to a database or add a new qualified operator and check if it shows up in the page. ii. Repeat a similar process as above for first aid attendant tracker.	

Appendix B: User Interface (UI) Design

B.1. Introduction

SafeLift is a collision and proximity detection system that detects obstacles and pedestrians wearing safety tags within a configurable 360° zone around any forklift truck. The user interface design focuses on anticipating what users might need to do and ensuring that the interface has elements that are easy to access, understand, and use. The goal is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals.

AppAid is an Android and iOS risk managements app. It allows the users of the app to report any incidents that occur at the workplace. It tracks all these injury reports and generates analytical reports for the employers. These reports are used by the employer to determine injury trends and allows them to initiate any preventive measures to prevent further incidents. The app also allows tracking of qualified forklift operators and first aid attendants. There is also an optional map module that allows the employer to select an abstract map that is the size of the warehouse. They can then mark any first aid, and other important emergency equipment, as well as divide the map into sections to allow employees to report locations easier.

The purpose of this document is to analyze the user interface of SafeLift and AppAid and provide an overview of the high-level user interface for proof of concept and prototype. Analysis encompasses both the hardware and software designs. It provides user & technical analysis and the usability testing methods to be used by the designing in developing the UI for the final product.

B.2. User Analysis

B.2.1. SafeLift

SafeLift will be designed in a simple way to easily delineate human and computer orientated task that are required to achieve system function. The user shall not be required to directly use the operating system, data management etc., instead these operations shall be hidden from the user and performed behind the scenes. A typical user would need basic knowledge of how to safely power up a use electrical device and the ability of reading and understanding the User documentation to operate the system normally.

The users should be well trained in using a forklift and have prior knowledge of forklift safety information because *SafeLift* only increases the forklift safety and is a standalone safety system.

B.2.2. AppAid

AppAid will be designed with ease of use in mind. The user will only be required to have knowledge basic knowledge of how to use an Android or iOS devices. The interface of each page of the app will be simple and uncluttered. The main page of the app will contain buttons that will lead to the pages for each of the main modules. When the user submits a for or request a report there will be a loading symbol to let the user know that their request is being processed. This will take the guess work out of knowing when a request has been completed. The reporting page will be easy to use. All the information required for the report will be laid out on an easy to understand form.

B.3. Task Analysis

This section strives to analyze and understand the work the user performs in specific circumstances and the subtasks that will be performed as the user does the work. The “*Seven Elements of UI Interaction*” outlined in Don Norman’s text (*The Design of Everyday Things*) are considered.

B.3.1. Discoverability

B.3.1.1. SafeLift

Discoverability entails designing the elements of the system to be visible and obvious to what they accomplish. The interaction modes should not force user into unnecessary or undesired actions, they should enter and exit a mode with little or no effort. The following are some key elements that should be easily discoverable:

- Power and reset buttons – these buttons will be added on the casing of the whole unit to allow user easy access when turning on or rebooting system.
- Connections ports and cables will be color coded to allows user intuitively to know the correct placements

B.3.1.2. AppAid

Discoverability is an important part of good UI design. By making AppAid as easy to learn as possible we can provide the user with a better experience. To make AppAid easy to learn and use, features must be easily discoverable but not every feature can be designed in this way. On a desktop application there is a considerable amount of space to place buttons for various features to make them easily discoverable but on a mobile phone, the screen space is small, making it important to prioritize features that are core to the use experiences. These core features are the one that we will design our and make easily discoverable. The following is a list of important features that the user needs to discover first and easily.

- **Users to need to log in to the app easily:** This will be the first page the user will see when they open the page and every time they are logged out it will take them back to this page. It will be an easy to use page with clear labels for the user name and password. The Login button will be made to look like a button that is commonly seen in many other applications. The design is shown in Figure _
- **The user needs to be able to access main modules easily:** Each of these modules will have a button placed on the first page the user will see after login in. This will allow the user to easily access each module.
- **The user needs to reach home page easily:** There will be a home button on the top of the screen in an option bar. It is located there because it is a common place to find such a button.
- **The user needs to sign out easily:** The sign out button will be placed at the top of the options bar inside an options menu. The option menu will have a menu icon that is commonly seen in many apps.
- **The forms will need to be easy to fill out:** The report form will have clearly labeled fields. These labels will be like those used in a work place investigation form used by WorkSafeBC and OSHA.
- **Easy to understand statistical reports:** These reports will be represented using various bar graphs and pie charts, and other graphs that the user are familiar with.
- **Easy to switch between modules:** Inside the options menu, there will be links to the other modules to allow the user to switch between module easily, without needing to go to the home page.

B.3.2. Feedback

B.3.2.1. SafeLift

Feedback is the most vital part of the *SafeLift* system and will greatly determine the driver's awareness and actions according to the provided feedback. To ensure fast reaction by the driver the interface provides visual and audio alerts through an LCD screen and a speaker. Displayed status messages will be intuitive and indicate risks level risk levels. The following are some element of the feedback that will improve user interaction:

- The LCD display will display the number of tags detected in the danger zone and distance of obstacle to ensure driver has enough time to avoid collision.
- The object distance display will be color coded to easily notify driver of the risk of collision. Far distance will be shown in green and closer distances in red.
- The interface will have flexibility of how the messages are displayed for example for distance display the driver can choose between meters or centimeters in accordance to what they are used to.
- The audio warning will also change volume according to how far the obstacle
- The system will also notify the user in case of low battery

B.3.2.2. AppAid

Feedback is helps to engage and explain to the user what is happening with the app. This can help improve user satisfaction and avoid aggravating experiences. To this end, AppAid will provide feedback at every possible step.

- When the user presses login or any other request there will be a loading icon that lets the user know that his login request is being processed.
- When a user request fails, a reason will be given in a pop-up overlay. For example, letting the user know that a request failed because he does not have a Wi-Fi connection.
- If a user forgets to fill in a field in the form, the field that is not filled in will be highlighted. A pop overlay will appear to let the user know which fields have not been filled in.

B.3.3. Conceptual model

A conceptual model is the mental model that people carry of how something should be done based on user knowledge and experience. SafeLift has elements that users already have prior use knowledge. The LCD screen, the speakers are items a typical user already knows it basic functionality. AppAid will operate similarly to common iOS app on various mobile devices with different physical characteristics.

B.3.4. Affordance

Affordance describes all actions that are made physically possible by the properties of an object or an environment.

B.3.4.1. SafeLift

In SafeLift the, the LCD display and speakers affords user to a have mental picture of how far an object is and thereby take appropriate action to avoid collision. Buttons, port connections allow user to easily connect and operate the system.

B.3.4.2. AppAid

AppAid will allow users to report any work place incidents. This is done through the reporting form. For employers, AppAid will allow the users to request statistical reports. This data will allow the employer to learn about any injury trends and allow them to implement corrective measures. The warehouse layout allows the employers to design an abstract representation of the warehouse or the actual floor plan if one is available. On this representation, the employer can place the location of emergency equipment, such as first-aid kits, and fire extinguishers, and to divide the map into sectors. This allows users to report location of injuries easier. The map will allow the user to the location of emergency equipment easier. Finally, the first aid attendant and qualified operator tracking will allow the employer to upload list of employees that are qualified to operate forklifts and employees that are first aid certified. This allows employees to identify first aid attendants easily. The list of operate allows employees to identify operators, so they can get help easily.

B.3.5. Signifiers

B.3.5.1. SafeLift

Signifiers hint at the affordance. The screen layout shall contain well understood visual that user can relate to real world actions which also are descriptive of SafeLift's operation. Any controls on the casing will also be clearly labelled to indicate their actions.

B.3.5.2. AppAid

The signifiers that AppAid will use are popup overlays that the user is familiar with from other apps that they may have used, to report any errors that occur during a user request. An example of this is shown below. In the incident reporting form, there will be question marks next to each field. This question mark will open a popup overlay that explain what the field is for, and what information is required in the field.

B.3.6. Mappings

B.3.6.1. SafeLift

Mapping describes the positioning of controls and the items they control. The power and reset button will be placed on top of the prototype casing and the connection ports on the sides in a sequential order in which the cable connection do not intercross.

B.3.6.2. AppAid

AppAid uses layouts that are commonly seen in other apps, so there is minimal learning required. For example, the home button is placed in a toolbar at the top of the screen. This is commonly found in many other apps. The log out button will be placed within an options menu in a tool bar at the top of the screen.

B.3.7. Constraints

Constraints limit the actions that can be performed by the user, thus increasing the usability of the design and reducing the likelihood of operator error. For SafeLift there are physical and logical constraints such as proper port and cable connections to allow the system to function properly. In AppAid when signing up or logging in there will be constraints on passwords length and composition. The incident forms will also be standardized to avoid wrong report filling.

B.4. Engineering Standards

The device meets all industry safety standards the documentation and development of both software and hardware components will adhere to standards set by regulatory bodies and the state. The following are that will be followed in the UI design:

Standard	Application
ISO/IEC 13251:2004 Typical graphical symbols used in the office	The symbol designs of the power button, reset button and on the LCD, display should be graphical symbols that are typically used on office equipment such as computers.
ISO 9241-112:2017 [20] Establishes design principles for interactive systems related to the software-controlled presentation of information by user interfaces.	Applies to LCD display and speaker to the perception and understanding of presented information
IEEE 1101.1-1998 [19] The dimensions and tolerances necessary to ensure mechanical function compatibility are provided	All physical user interface unit follow this to ensure reduction in design time and improve usability
IEC TR 61997:2001 [21] Guidelines for the user interface in multimedia equipment for general purpose use	The design of buttons, LCD screen, power switch, power plug would not cause any problem for the first time use without any special previous training.
RoHS Restricted Substances [22]	All the materials that we used in our device shall be RoHS compliant to ensure user is harmed during usage
CAN/CSA-Z107.56-M86 Procedures for the Measurement of Occupational Noise Exposure	The speakers noise level exposure on the workers even without ear protection should be within the set limit and not cause harm

B.5. Usability Testing

B.5.1. Analytical Testing

This section outlines the testing that will be done by the engineering design team using heuristic evaluations. Each evaluator will independently examine the UI, check for compliant and usability, find different problems after which they meet and discuss afterwards. The following the phases of the heuristic evaluation:

1. Pre-evaluation training – evaluators learn domain knowledge and information on the scenario
2. Evaluation – individuals evaluates UI and makes list of problems
3. Severity rating – determine how severe each problem is
4. Aggregation – group meets and aggregates problems
5. Debriefing– discuss the outcome and suggest possible improvements to address the issues

The evaluator gives each test a severity rating according the scale below:

- **Minor** – not fixing the problem will not negatively affect the user interaction and system
- **Major** – problem will lead to confusion and delays in operation, important to fix
- **Catastrophe** – problem results in wrong functionality of system, imperative to fix

B.5.1.1. SafeLift Tests

- i. Visibility of system status – the messages displayed by the LCD should be readable
- ii. The user control and freedom – user should be able to interrupt a sequence of actions to do something else without losing previously done work
- iii. Consistency and standards – the results displayed should be consistent in each test trial and should conform of accepted outcome.
- iv. Error prevention – the interface should alert the user in case of improper or unknown commands.
- v. Recognition rather than recall – the interface should reduce the user’s requirement to remember past actions by providing visual cues of such actions
- vi. Flexibility and efficiency of use – user can change modes to suit their comfort level
- vii. Help users recognize, diagnose, and recover from errors – user should be able to undo the last action or get verification prompts before that accomplish certain tasks
- viii. Help and documentation- the descriptions user documentation should match the outcomes seen by the user

B.5.1.2. AppAid Tests

- i. Easy to login – The login should be easy, and any errors that occur during the login should be displayed clearly
- ii. Error reporting – The app should clearly report any errors that occur during a user request. The errors should be easy to understand
- iii. Easy navigation – It should be easy to go between modules. The user should be able to go to another module through links in the options menu.
- iv. Correct information – The information contained in the infographics provided by the analytical module should be the correct information.

B.5.2. Empirical Usability Testing

This testing will involve the actual users of the device and will be done during the prototyping stage of development. The specific target user will be warehouse employers and employees whose feedback will enable us to develop the final product to be more user friendly. In this testing the user will be provided a complete prototype that performs all the functionality of the system. Once the device is installed and ready, the user is given a short training on how to use the device, the user documentation, safety and necessary troubleshooting information without too much technical descriptions.

Once the user is familiar with all the parts and their functionality, the evaluation begins. The designers will refrain from answering too many questions from the user instead just observe and take note of how ease of use, confusions and time taken to accomplish the desired actions. The users will be prompted to verbalize their thinking process as they interact with the system.

The designers will record their observation about where, when and why problems occurred. After the testing is done users are to be asked to fill out a questionnaire to rate the usability of all the features, give suggestions for improvements, their likeliness to purchase it and comments of whether the system improved the overall forklift safety. Once the feedback is collected and reviewed the design will be updated to reflect the changes in the final product.

B.6. Graphical Representation

B.6.1. SafeLift

The user interface has both visual and audio feedback. The LCD display will display the general direction of the obstacle depending on which sensor has done the detection. The number of tags read will also be shown to indicate the number of pedestrians in the danger zone. The figure below shows a sample of graphical user interface.



Figure 19: Sample of LCD screen user interface display

B.6.2. AppAid

The following diagrams show the mockup of the UI for AppAid. The final product will have an UI that resembles the ones shown below. As can be seen from below, AppAid follows an intuitive UI design, that users should be familiar with if they have use other phone apps.

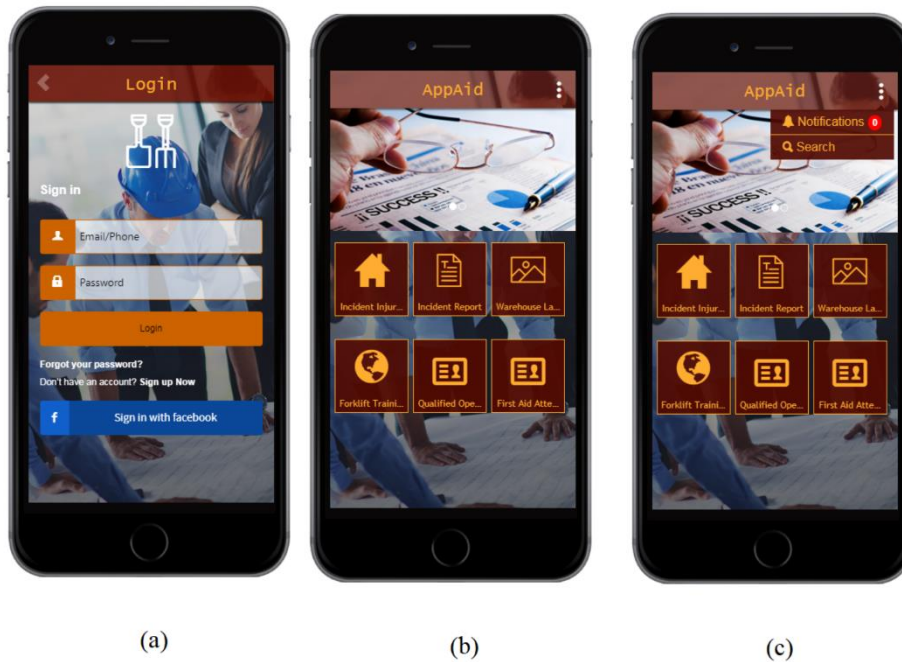


Figure 20: (a) The login screen (b) The home screen (c) Home screen with a notification



Figure 21: (a) Incident stat UI (b) Incident reporting form (c) Forklift training UI

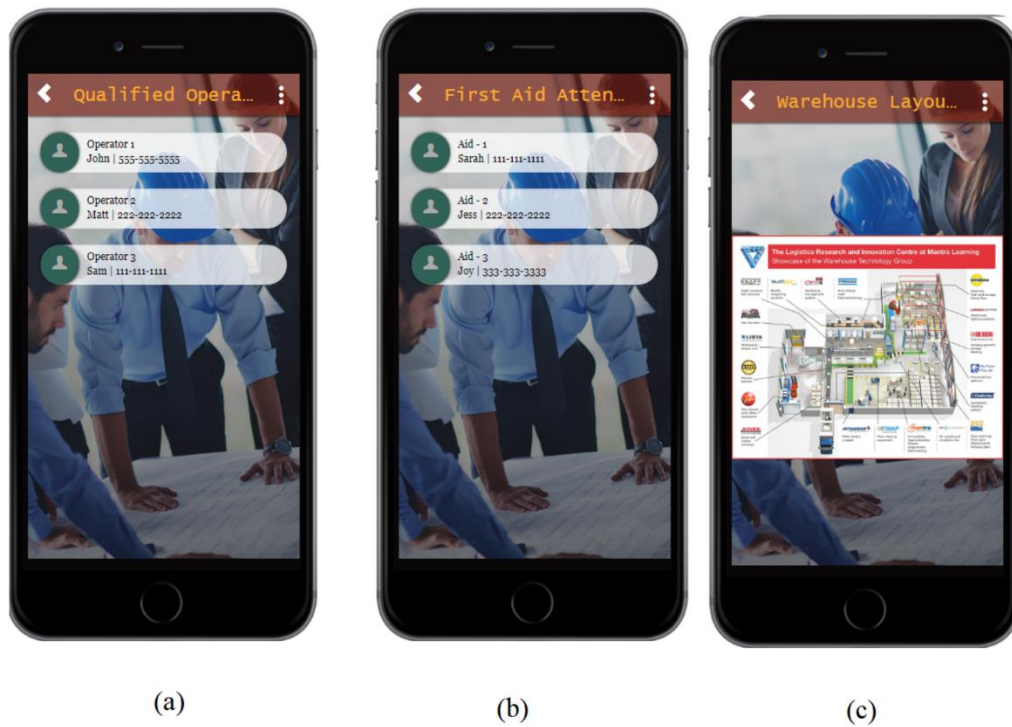


Figure 22: (a) UI for qualified operator list (b) UI for first aid attendant list (c) Floor plan UI

B.7. Conclusion

The UI appendix outlines the aspects of the user interface design by analyzing the knowledge a typical user would need to efficiently operate the system. It also covers the testing methods used by the designers in developing the best user relatable interface. The current state of the UI design focusses on the representation of the feedback of object detection, which will be illustrated during the proof of concept. In the next four months the rest of the user interface will developed for the prototype to be aesthetically pleasant and easy to use by end consumers while meeting all the functional requirements.

Appendix C – Planning Appendix

C.1. Introduction

In this fast-paced world where everyone is racing with life, often we subconsciously forget the basic rules of safety which, in 2016, resulted in 149,554 work-related injuries costing over a billion-dollar worth of claims in British Columbia alone[23].

Forklifts are used to stack, lift and transfer loads in factories, shipping yards, freight terminals, warehouses and other workplaces. While they offer the efficiency in material handling they are amongst the most hazardous vehicle types and continue to be associated in workplace injuries and death. The following are the most common incidents involving forklifts[24]:

- Object and pedestrian collision
- Loose loads falls
- Tipping from overloading or high-speed swerves

According to the 2016 statistics provided by WorkSafeBC, 26.2% of reported workplace fatalities occurred due to incidents involving heavy industrial vehicles including loaders, skidders and forklifts[23].

Clearly, there is a serious need to put in place strict protection measures and risk management systems to provide a safer workplace. Our project seeks to provide a solution to reduce the percentage of injuries and deaths caused by these powered industrial trucks.

Our solution, *SafeLift*, is a sensory system that can be easily integrated into existing industrial vehicles. *SafeLift* alerts the driver when a worker or an obstacle, that might be out of sight, is in the proximity of collision. Added on to this system is a risk management app, *AppAid*, that keeps record of all injuries and helps track accident prone locations in workplace so that concerned authorities can be notified to take appropriate measures to reduce calamities, fatal or otherwise

This Appendix outlines the following:

- The scope, risks and benefits associated with the project
- The target market and competition analysis
- Personnel, time and budgetary management

C.2. Scope

PowerLift Safe Solutions offers two products; *SafeLift* and *AppAid*. *SafeLift* is being designed as an aid for creating environment awareness for forklift drivers. The goal is simple – to eradicate workplace casualties due to forklift, fatal or otherwise. This is achieved by alerting the forklift operator of workers in close vicinity of the vehicle as well as notifying the worker of the oncoming vehicle via vibrating sensation in the RFID tags. *AppAid* on the other hand compliments *SafeLift* by providing training modules and highlighting accident-prone locations around the warehouse for the forklift operator to be extra cautious in those areas.

The purpose of this document is to specify the details of development for *SafeLift* and *AppAid* over the course of next four months. Subsequent sections of this document are intended to cover the underlying risks and benefits associated with the project, the target market for the product as well as its existing competitors. The document also highlights major milestones of the project, assignment of development tasks distributed amongst the engineers at *PowerLift Safe Solutions* and a detailed monetary requirement for the project.

C.3. Risk and Benefits

The goal of *PowerLift Safe Solutions* is to provide a robust as well as a cost-efficient solution regarding the issue of workplace safety. Forklifts are the main source of transporting heavy loads across a warehouse. However, due to their eccentric structure, forklifts inherit a great deal of risk. This section of the document highlights some risks and benefits associated with the project that are worth noting.

C.3.1. Risks

SafeLift is an aid for creating environment awareness for the forklift driver. Forklifts are dangerous pieces of equipment travelling with relatively high speeds in comparatively narrow aisles. This inherently introduces risks. One significant risk would be the operator's over-dependency on the product. For example, drivers not looking back while reversing, assuming they will be alerted if there is obstruction. In such cases if the sensors fail to detect the obstacle or are slow, the result could be disastrous. To avoid such incidents, the lift truck operator will have to always obey all forklift safety rules.

Since we will be using RFID tags, there is a risk of workers forgetting or losing their tags without informing management. In addition, electromagnetic interference for example from other machines or fluorescent lights could block proper tag reading. Our system therefore connects the ultrasonic sensors and RFIDs in parallel combinations rather than series. This ensures that the failure of one component does not affect the functionality of the others.

Another risk would be complacency of forklift drivers. Even highly skilled forklift operators can become careless in their daily routines due to the repetitive nature of their jobs. Stopping distances are often significantly underestimated and with only a maximum detection of 2.5 meters, our product may be slow in alerting the driver in case they are driving too fast.

C.3.2. Benefits

Workplace safety should always be the number priority for every employer. Providing a safe working environment gives workers a piece of mind and motivation to be efficient and more productive. With *SafeLift* drivers and pedestrians can confidently work in the same space without fear, while *AppAid* will help improve quality of drivers through the easily accessible training modules.

AppAid monitors maintenance records of forklifts allowing employers know the working condition of their forklifts, this avoids any unexpected break downs of forklifts assumed to be in good condition. It's estimated that a forklift out of commission for repairs costs thousands of dollars every day in lost profit. Forklift damage contributes greatly to the amount of unnecessary costs to a company. It's estimated that forklift damage can add

as much as 5% to the cost of a standard truck lease. When considering an entire fleet of powered trucks, the costs add up to thousands of dollars of unnecessary, unplanned costs each year[25].

It is the employer's responsibility to cover any medical expenses for a worker injured in a work-related accident. In addition, the loss of productivity due to absent of worker huge financial cost to the employer. The cost of forklift accidents involving injured workers can cost as high as \$38,000 in direct costs per injured worker and \$150,000 in indirect costs[25]. SafeLift and AppAid will help to reduce this cost allowing the employers to invest this money back into the business instead.

Using AppAid statistical analysis of workplace incidents employers can assess their work sites to identify existing and potential hazards and eliminate or control the identified hazards.

C.4. Market Competition

Today, almost every heavy material handling facility such as warehouses owns a forklift or some other kind of lift truck. These lift trucks pose a great hazard to those around them when operated unsafely. Often when people get negligent, they tend to forget about basic safety practices ultimately leading to avoidable accidents. In 2016 in British Columbia, 49,554 work-related accidents injuries cost over a billion-dollar worth of claims[23].

With the ever-increasing number of forklifts in the material handling industry, the need for a solution in reducing the accidents due to the operation of lift trucks has become very crucial.

PowerLift has created a low cost and efficient two-part system, *SafeLift* and *AppAid* that rivals the market competition. It is the first product of its kind that truly combines a hardware, firmware and software component

C.4.1. Competition

There are many existing solutions that try to solve the problems of safety associated with forklifts. One type of prevention system that is marketed by competitors is one that uses an LED spotlight or projector to alert the pedestrian of an oncoming vehicle.

We believe such types of safety, while low cost, is not adequate to prevent injuries. It requires the pedestrians to have awareness of the situation. If someone is using their phone or carrying a box that is blocking their vision, they will not see a light or projection on the ground. These systems do not alert the driver of any pedestrian or other obstructions. This can lead the driver to assume that there is no obstruction, which can lead to unnecessary injuries.

Other companies like Claitec and Cisco-Eagle provides a system that uses RFID. These systems have an antenna that can detect in a circle around the forklift for any RFID tags that can be worn by a pedestrian or attached to another vehicle. When a pedestrian or another vehicle with a tag enters the detection zone, it sets off a visual alarm as well an audio one to alert the driver.

While RFID can tell you that something with a tag has entered the zone, there is no way to know which direction or how far the object is from the vehicle. We believe that this can be vital information for the driver. If they know where the object is, they will have better awareness of their surroundings. This can prevent the driver from missing things, as even if they take extra 11 precautions, being human, they can miss some details. These systems also do not tell the driver if there are any obstacles in their way if it does not have a tag. If a person forgets to wear a tag or their tag gets lost, the driver will not be notified. A box, or other obstruction can fall

without notice, and if it is substantial in size, it could cause injury. A worker can forget and leave a box in aisle. If a forklift is carrying a large enough load that, their vision is blocked, they will not notice the obstruction in front of them. A forklift could hit obstacle that is hanging from the roof and these systems from the competitor cannot detect them.

With SafeLift we will not only have an RFID sensor but also an ultrasonic sensor system. The ultrasonic sensors can not only detect object that do not have a tag, but also provide range and location of the object. It will also provide overhead detection to prevent the forklift from hitting any overhead obstructions.

There are many competitors to AppAid, such as EHSInsight and INTELEX, and the traditional filing of a workplace incident form. These companies provide a typical application to manage workplace safety, risk and compliance. The solutions are provided as different modules. They include incident management, audit management, training management, and near misses, and many other modules. While these solutions are general and can be applied to many different workplaces, they may not fit the needs of a warehouse. They also tend to be expensive. Our app will be more affordable and include features that are more suited to the warehouse setting.

AppAid will include incident management, training management, and near misses, but also other features such as floor plan for the warehouse to show important safety details, forklift maintenance and condition tracking, and a tool for tracking all licensed power tool operators, first aid attendants. Our incident management will provide an easy reporting format that will include location, type, and cause of injury and as well as any other information required by WorkSafeBC. Once, we correctly meet the WorkSafeBC standards of incident reporting, we can easily upscale it to other provinces of Canada. For accidents involving forklifts information on the operator can be reported as well. This information will be used in our analytics module to provide graphs and other reports.

The additional features provided, we believe can be of immense value to a workplace. Our app can send reminders and help with maintenance of vehicles. This can help prevent any injuries caused by poor maintenance. Located in the app is also a list of all first aid attendants. The floor plan will show the location of all safety details. This is crucial information, and having it on hand, can help workers be more efficient during a crisis. Our app can also provide information on all certified operators. This can help workers locate someone when they need help instead of operating equipment they do not know how to use.

While AppAid and SafeLift both individually have competitors, as an integrated system, there is not any competition that can be found on the market. We believe that this will be the edge for our product. It can cost enormous amount of money and headache to integrate systems from different 12 vendors. By providing an integrated solution, we hope to provide a solution that is not only lower cost, but also easy to use.

C.5. Personnel Management

Fred Mose, CEO oversees the overall project and making final decisions. Fred has directed the project, keeping the vision that has been outlined throughout the entire project. Partnering up with the other officers, to strengthen and ensure the stability of the company. The CEO works closely with the CFO (Daniya Zafar) to prepare the budget and complete risk analysis throughout the lifecycle of the project.

Layomi Dele-Dare, COO oversees day to day operations and reports directly to the CEO. Layomi is responsible for the effectiveness of all processes, both internal and external, as well as providing timely, accurate and

complete reports such as the meeting minutes. Layomi is also responsible for ensuring that the scheduling slippage and shipping delays are minimized.

Daniya Zafar, the CFO, is responsible for product marketing. Daniya also works closely with the CEO (Fred Mose) to prepare the financial budget and reports.

Wenhao Zhang, the CTO, oversees technical issues and supports and is also the lead in the software development of the project. Wenhao consolidates and creates a plan for each technological platform as well as oversees all system design and changes in the overall architecture.

Table 7: Breakdown of the duties for the SafeLift

Engineer/Duties	Input Module	Output Module	Integration	Testing
Daniya Zafar			✓	✓
Fred Mose	✓	✓		✓
Layomi Dele-Dar		✓	✓	✓
Wenhao Zhang	✓			✓

Table 8: Breakdown of duties for AppAid

Engineer/Duties	Report Form	Statistical Analysis	Training Module	Warehouse Mapping
Daniya Zafar	✓		✓	
Fred Mose			✓	
Layomi Dele-Dar		✓		✓
Wenhao Zhang		✓		✓

C.6. Time Management

C.6.1. Gantt Chart for May – August 2018

The chart below shows the development and test plan duration for the four-month period of May – August 2018.

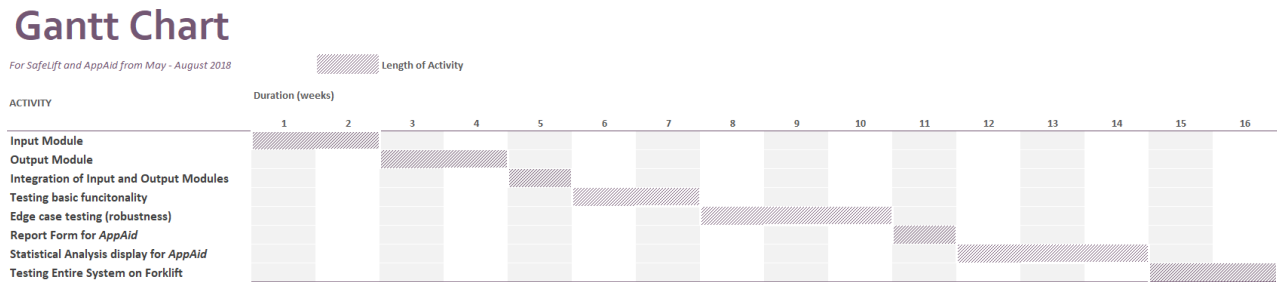


Figure 23: Gantt char for May – August 2018

C.6.2. Major Milestones

Some major accomplishments in development of *SafeLift* and *AppAid*, are summarized in the milestone chart below. It highlights all major processes and milestones to be attained in the future.

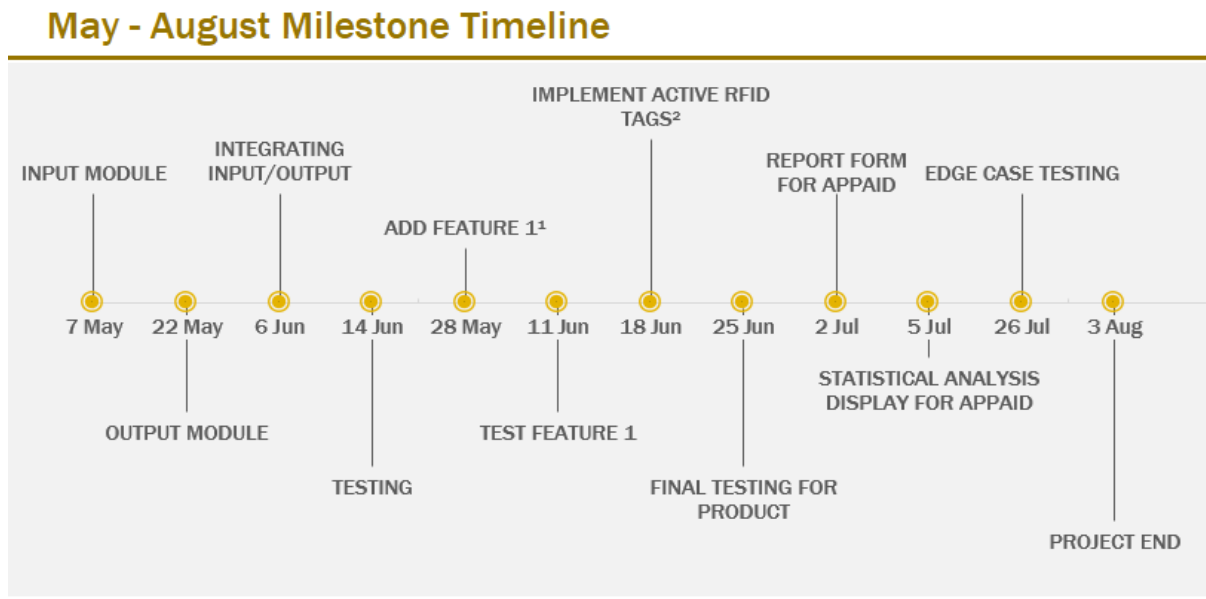


Figure 24: Timeline that includes major milestones for the next four months

¹**Feature 1.** Auto generating a hit report. This new function will submit an incident report *automatically*. There are two main reasons which make this a very vital feature for *SafeLift*. Firstly, in case of an accident, the forklift operator might be dishonest and not submit the accident report. Secondly, though this feature we can avoid accidents due to poor driving skills of the forklift operator. For instance, the forklift operator might be lucky enough to stop within a cm or two of hitting a pedestrian. Even though, in this case the accident did not happen, it very easily could have. Hence, to avoid such situations, an auto generated incident report will be sent through the Raspberry Pi to concern authorities indicating that the employee requires training or might not be best suited to drive forklifts.

²**Active RFID tags.** Currently, our system uses passive RFID tags to detect pedestrians. The main reason for using passive tags is to because they are cheap and have an indefinite lifetime. However, from the usability point of view, active tags are more suited for our purpose. The main purpose of our system is to alert the forklift operator **as well as** the pedestrian of the oncoming vehicle. In order to notify the pedestrian, we need some kind of warning via the tag. This is only possible if the tag is active and we use vibrating sensation to alert the pedestrian. The active tag we will be using is shown below followed by its technical details



Figure 25: Example of active RFID that may be used.

Table 9: Specifications of the active RFID tag

Range	Up to 90 m
Call Button	On/Off Tag
Operating & Storage Temperature	-20° ~ +60°C
Humidity	5 ~ 95% RH
Dimensions	(42 x 30 x 10) mm
Options	Built in Vibration Switch
Weight	9.1 g

C.7. Budgetary Management

With regards to budgetary management, the table below breaks down the cost estimation required in completing the prototype of our products

Table 10: Cost break down of parts from first four month

Component	Purpose	Cost
Arduino Uno	• Microcontroller for input module	\$13.89
Raspberry Pi	• Microcontroller for output module	\$109.95
Waveshare Raspberry Pi 10.1 Capacitive Touch Screen	• Visual display of output module • Displays the location of workers and distance of obstacles	\$146.92
Ultrasonic sensors	• Detecting obstacles as well as workers	\$12.70
RFID tags and reader	• Detecting workers	\$250.00
Wires	• Connecting components	\$11.39
	Total	\$544.85

Over the course of next four months, May – August 2018, we do not plan on buying any major components for *SafeLift*. However, we have a budget of \$ 300. Table 11 below shows a breakdown of the amount:

Table 11: Estimation of the cost break down for next four month

Component	Cost
Casing for microcontroller	\$ 50
PCB Design	\$ 150
Active RFID tags	\$ 100
Total	\$300

Potential Funding source for *SafeLift* and *AppAid* will be the *Engineering Science Student Endowment Fund* offered by the ESSS, whose application deadline is May 2018. Our project qualifies under Category B of *ESSEF*. We will apply to avail this opportunity in Summer 2018 semester. In the unfortunate event where the aforementioned funding opportunities become unavailable, each member of our team has agreed to contribute equally for the purchase of the material required to construct our final working product.

C.8. Conclusion

The Planning appendix outlines the planning aspects of the project. It covers the scope, risks and benefits, the market competition, research rationale, personnel, time and budgetary management. Throughout the 4 months, the timeline outlined above has been followed almost verbatim, and it is believed that the next four months, the timeline we have outlined above will be followed closely as well.

References

- [1] "Clark Michigan CMP25 LPG Forklift", *Ritchiespecs.com*, 2018. [Online]. Available: <http://www.ritchiespecs.com/specification?type=Lift&category=Forklift&make=Clark+Michigan&model=CMP25+LPG&modelid=103367>. [Accessed: 31- Mar- 2018].
- [2] "Choosing an Ultrasonic Sensor for Proximity or Distance Measurement Part 1: Acoustic Considerations | Sensors Magazine", *Sensorsmag.com*, 2018. [Online]. Available: <https://www.sensorsmag.com/components/choosing-ultrasonic-sensor-for-proximity-or-distance-measurement-part-1-acoustic>. [Accessed: 31- Mar- 2018].
- [3] W. Hamlet and W. Kusewich, "Using The Ultrasonic Sensor to Determine Location", *Engineering.nyu.edu*, 2018. [Online]. Available: http://engineering.nyu.edu/mechatronics/summit/SUMMIT2007/group6-Will_Bill/Summit2007Project.pdf. [Accessed: 31- Mar- 2018].
- [4] S. EVANCZUK, "Fundamentals of RFID communications", *Electronic Products*, 2018. [Online]. Available: https://www.electronicproducts.com/Passive_Components/Circuit_Protection/Fundamentals_of_RFID_communications.aspx. [Accessed: 31- Mar- 2018].
- [5] W. Jingchao, Z. Chun, C. Baoyong, W. Ziqiang, L. Fule and W. Zhihua, "A low cost integrated transceiver for mobile UHF passive RFID reader applications", *Journal of Semiconductors*, vol. 30, no. 9, p. 095007, 2009.
- [6] "High Performance Passive RFID Tags", *Omni-id.com*, 2018. [Online]. Available: https://www.omni-id.com/pdfs/RFID_Tag_Implementation_Testing_Deployment_Guide.pdf. [Accessed: 31- Mar- 2018].
- [7] S. Armstrong, "Circular Polarization vs. Linear Polarization: Which is the right RFID Antenna?", *RFID Insider*, 2018. [Online]. Available: <https://blog.atlasrfidstore.com/circular-polarization-vs-linear-polarization>. [Accessed: 31- Mar- 2018].
- [8] "C22.1-15", *ShopCSA*, 2018. [Online]. Available: <http://shop.csa.ca/en/canada/c221-canadian-electrical-code/c221-15/inv/27013892015>. [Accessed: 31- Mar- 2018].
- [9] "35.040.50 - Automatic identification and data capture techniques", *Iso.org*, 2018. [Online]. Available: <https://www.iso.org/ics/35.040.50/x/>. [Accessed: 21- Feb- 2018].
- [10] I. 18033-3:2010, "ISO/IEC 18033-3:2010 - Information technology -- Security techniques -- Encryption algorithms -- Part 3: Block ciphers", *Iso.org*, 2018. [Online]. Available: <https://www.iso.org/standard/54531.html>. [Accessed: 31- Mar- 2018].
- [11] "ISO/IEC 27001 Information security management", *Iso.org*, 2018. [Online]. Available: <https://www.iso.org/isoiec-27001-information-security.html>. [Accessed: 31- Mar- 2018].
- [12] "Apple Vs Android—A comparative study 2017 – AndroidPub", *AndroidPub*, 2018. [Online]. Available: <https://android.jlelse.eu/apple-vs-android-a-comparative-study-2017-c5799a0a1683>. [Accessed: 31- Mar- 2018].
- [13] "What is RESTful API? - Definition from WhatIs.com", *SearchMicroservices*, 2018. [Online]. Available: <http://searchmicroservices.techtarget.com/definition/RESTful-API>. [Accessed: 31- Mar- 2018].
- [14] "The MVVM Pattern", *Msdn.microsoft.com*, 2018. [Online]. Available: <https://msdn.microsoft.com/en-us/library/hh848246.aspx>. [Accessed: 31- Mar- 2018].
- [15] "ASP.NET MVC Overview", *Msdn.microsoft.com*, 2018. [Online]. Available: [https://msdn.microsoft.com/en-us/library/dd381412\(v=vs.108\).aspx](https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx). [Accessed: 31- Mar- 2018].
- [16] "Hashing Passwords in Java with BCrypt", *Stubbornjava.com*, 2018. [Online]. Available: <https://www.stubbornjava.com/posts/hashing-passwords-in-java-with-bcrypt>. [Accessed: 31- Mar- 2018].

- [17] "Selecting the Right Encryption Approach | Data Encryption Types | Thales e-Security", *Thalessecurity.com*, 2018. [Online]. Available: <https://www.thalesecurity.com/products/data-encryption/selecting-the-right-encryption-approach>. [Accessed: 31- Mar- 2018].
- [18] "Ultrasonic Sensor: Precautions for Surrounding Environment | FAQ | Australia | Omron IA", *Omron.com.au*, 2018. [Online]. Available: http://www.omron.com.au/service_support/FAQ/FAQ01346/index.asp. [Accessed: 31- Mar- 2018].
- [19] "IEEE 1101.1-1998 - IEEE Standard for Mechanical Core Specifications for Microcomputers Using IEC 60603-2 Connectors", *Standards.ieee.org*, 2018. [Online]. Available: <https://standards.ieee.org/findstds/standard/1101.1-1998.html>. [Accessed: 31- Mar- 2018].
- [20] I. 9241-11, "ISO 9241-11 - Ergonomics of human-system interaction -- Part 11: Usability: Definitions and concepts", *Iso.org*, 2018. [Online]. Available: <https://www.iso.org/standard/63500.html>. [Accessed: 31- Mar- 2018].
- [21] "IEC TR 61997:2001 | IEC Webstore", *Webstore.iec.ch*, 2018. [Online]. Available: <https://webstore.iec.ch/publication/6269>. [Accessed: 31- Mar- 2018].
- [22] "RoHS Compliance Guide: RoHS 10 Restricted Substances", *Rohsguide.com*, 2018. [Online]. Available: <http://www.rohsguide.com/rohs-substances.htm>. [Accessed: 31- Mar- 2018].
- [23] "Annual statistical reports - WorkSafeBC", *Worksafebc.com*, 2018. [Online]. Available: <https://www.worksafebc.com/en/about-us/shared-data/facts-and-figures/statistical-reports>. [Accessed: 31- Mar- 2018].
- [24] R. Pfeiffer, "3 Most Common Forklift Accidents and How to Avoid Them", *Tmhnc.com*, 2018. [Online]. Available: <https://www.tmhnc.com/blog/forklift-accidents-and-warehouse-safety>. [Accessed: 31- Mar- 2018].
- [25] "The Hidden Costs of Forklift Accidents | CertifyMe.net", *CertifyMe.net*, 2018. [Online]. Available: <https://www.certifyme.net/osha-blog/the-hidden-costs-of-forklift-accidents/>. [Accessed: 31- Mar- 2018].