Lewis University logo

**Final Project Report**

**Message Wave**

**Summer 2024**

By

**Daniel Schmidt and** [Spencer Blauw](mailto:spencerblauw@gmail.com)

Department of Computer and Mathematical Sciences

Submitted in partial fulfillment of the requirements

for the degree of

Master of Science in Computer Science,

Concentration in Software Engineering

**Advisor**: Dr. Khaled Alzoubi

**Wednesday 05 June 2024**

# ABSTRACT

This will be written near the end of the project.

# TABLE OF CONTENTS

Chapter Page

[ABSTRACT 2](#_Toc172395710)

[TABLE OF CONTENTS 3](#_Toc172395711)

[LIST OF TABLES 5](#_Toc172395712)

[LIST OF FIGURES 6](#_Toc172395713)

[Basic Project Information 1](#_Toc172395714)

[Project Title 1](#_Toc172395715)

[Client Name 1](#_Toc172395716)

[Problem Definition 1](#_Toc172395717)

[Project Objectives 1](#_Toc172395718)

[Project Team 1](#_Toc172395719)

[CHAPTER I: Introduction and Project Overview 2](#_Toc172395720)

[Problem: 2](#_Toc172395721)

[Stakeholders: 2](#_Toc172395722)

[Goals: 2](#_Toc172395723)

[Implementation Process: 3](#_Toc172395724)

[Specific objectives 4](#_Toc172395725)

[CHAPTER II: Background and Literature Review 5](#_Toc172395726)

[Introduction 5](#_Toc172395727)

[Goals of Emergency Notification Systems 5](#_Toc172395728)

[Reaching the Target Audience 6](#_Toc172395729)

[Capture the Attention 8](#_Toc172395730)

[Ensure Understanding 9](#_Toc172395731)

[Ensure Understanding – The Problems 10](#_Toc172395732)

[Respond Appropriately 11](#_Toc172395733)

[What has been done 11](#_Toc172395734)

[Why We did what we did 14](#_Toc172395735)

[Privacy Concerns 15](#_Toc172395736)

[Intellectual Property 16](#_Toc172395737)

[Building Trust 17](#_Toc172395738)

[What Has Not Been Investigated 17](#_Toc172395739)

[Case Studies 17](#_Toc172395740)

[Fundamental Issues 17](#_Toc172395741)

[Development Environments and Deployment 17](#_Toc172395742)

[Testing Methods 17](#_Toc172395743)

[CHAPTER III: System Design 19](#_Toc172395744)

[Introduction 19](#_Toc172395745)

[Overall Approach 19](#_Toc172395746)

[Goals 19](#_Toc172395747)

[Functional Requirements 19](#_Toc172395748)

[Non-Functional Requirements 20](#_Toc172395749)

[Specifications 20](#_Toc172395750)

[Tasks and Timelines 21](#_Toc172395751)

[Development 21](#_Toc172395752)

[Modeling 22](#_Toc172395753)

[Chapter IV: Implementation 26](#_Toc172395754)

[Chapter V: Testing 35](#_Toc172395755)

[Support 40](#_Toc172395756)

[CHAPTER VI: RESULTS 41](#_Toc172395757)

[CHAPTER VII: CONCLUSION 42](#_Toc172395758)

[REFERENCES 43](#_Toc172395759)

[Appendix A: User Manual 45](#_Toc172395760)

# LIST OF TABLES

[Table 1: Tasks and Timelines 21](#_Toc172395761)

[Table 2: Test Case 1 37](#_Toc172395762)

[Table 3 37](#_Toc172395763)

[Table 4 39](#_Toc172395764)

[Table 5: Dart Analysis 39](#_Toc172395765)

# LIST OF FIGURES

[Figure 1: Multi-Criteria Analysis to establish the relative weights of evaluation criteria 8](#_Toc172395766)

[Figure 2: A Sample WEA Message 12](#_Toc172395767)

[Figure 3: High Level OpenAlerts Architecture 13](#_Toc172395768)

[Figure 4: System Architecture of OpenAlerts 14](#_Toc172395769)

[Figure 5 22](#_Toc172395770)

[Figure 6 22](#_Toc172395771)

[Figure 7 23](#_Toc172395772)

[Figure 8 23](#_Toc172395773)

[Figure 9 23](#_Toc172395774)

[Figure 10 24](#_Toc172395775)

[Figure 11 24](#_Toc172395776)

[Figure 12: Creation of SQL Database 26](#_Toc172395777)

[Figure 13: Entering Contact into DB 26](#_Toc172395778)

[Figure 14: Updating Contact 27](#_Toc172395779)

[Figure 15: Deleting Contact 27](#_Toc172395780)

[Figure 16 28](#_Toc172395781)

[Figure 17: Home Screen with Group Creation 29](#_Toc172395782)

[Figure 18: Creating New Group 30](#_Toc172395783)

[Figure 19: Contact 31](#_Toc172395784)

[Figure 20: Local Storage of Contact 31](#_Toc172395785)

[Figure 21: Message Payload from User in Java Iteration 1 31](#_Toc172395786)

[Figure 22: Message Payload from User in Flutter iteration 2 32](#_Toc172395787)

[Figure 23: Message Payload 3rd Iteration 32](#_Toc172395788)

[Figure 24: Fourth Iteration 33](#_Toc172395789)

[Figure 25: User Interface Screens 34](#_Toc172395790)

# Basic Project Information

## Project Title

Message Wave

## Client Name

Denver Modern Jui Jitsu

## Problem Definition

Numerous organizations lack a way to communicate with large groups of people associated with their organization. Notifications, announcements and important information is therefore piecemeal and often does not reach the intended audience in a timely or convenient manner. Current commercial options are also complicated and expensive.

## Project Objectives

The objective of the project is to provide a free, easy to use application that can store contact information and then send messages to selected individuals in order to allow large groups of people within an organization to receive important information in a timely and efficient manner.

## Project Team

Team Leader: Daniel Schmidt

Team Members: Spencer Blauw

# CHAPTER I: Introduction and Project Overview

## Problem:

Many Nonprofit organizations, local government agencies and businesses lack a way to quickly, effectively and cheaply communicate with large groups of people. Notifications, announcements and important information is often never received, viewed or actioned. Current commercial options are either ineffective or expensive and require a large up-front cost or more likely a monthly subscription fee. These organizations often have no budget or a very small budget and paying for any service is difficult or not realistic. There are many ineffective options that have low costs such as Email, but the intended audience is often slow to view the message or often does not view the message at all.

## Stakeholders:

Specific to this project Denver Modern Jui Juitsu is one of those organizations that fit the above description. They need to communicate with a large group of people and the communication needs to be quick, efficient and cost effective, preferably free. Schedules and announcements need to be sent out, read and understood for the organization to function properly and for people to enjoy their services. Stakeholders would have download rights to the software produced by this project. The stakeholder will be able to download the application for free. The software will be owned by the developers.

## Goals:

Message Wave has numerous goals. The first is to build a free to use application capable of storing a database of contacts and sending messages to a select group of those contacts through SMS services in the form of a text message. Another goal is to make the application easy to use with an intuitive interface. The final goal is to ensure any messages sent through the application are received, capture the attention of the user, the message is understood, and appropriate action is taken. This will ensure the application is beneficial to both the stakeholder and the recipient of any messages.

## Implementation Process:

The project will use a plan based, also known as the waterfall methodology for this project as requirements are largely clear and fixed. Requirements will not change in any drastic way and when requirements are unlikely to change a plan-based methodology has proven to be the best approach (Butler et al., 2020). Agile development will not be implemented as the user will not be heavily involved; requirements are not likely to change, nor will multiple deliverables be produced which are hallmarks of agile development (Dawson & Dawson, 2014).

The first step will be to identify the requirements of the project. This will be conducted by an initial stakeholder meeting and then follow on meetings on a regular basis. The stakeholder will be able to identify needed features and narrow the scope of the project. In addition, research will be conducted to identify requirements of previously built applications, concentrating on emergency notification systems and health care alerts. These two systems push important information to diverse groups and aim for the intended audiences to receive, read, understand and action the information they are given. The goals of these systems are similar to the goals of this project and much research and development has been conducted in these two areas.

Following requirements the design will be built using component diagrams to visually represent the high-level functions of the application. The features that will have to be diagramed will be account management, the database of contacts, message payload from the user, processing members list and selection, SMS service and connection, message payload distribution and the user interface. The next step will be to test the application. Testing will involve multiple phases: unit testing, integration testing, system testing, and user acceptance testing (UAT). The aim is to ensure that each component works correctly both individually and as part of the whole system. For unit testing each module will be tested independently to ensure functionality. Integration testing will use combined modules and be tested to ensure they work together seamlessly. System testing the entire application will be used in a production-like environment to ensure it meets all requirements. User Acceptance Testing will involve selected users testing the application to ensure it meets their needs and is user-friendly.

The training plan will include building and providing a user manual. This will be a comprehensive user manual detailing how to use the application. Online Training Sessions with training videos will be prepared to demonstrate the application's features and usage.

The next step will be to deploy the application. In this step the stakeholder will download the application and start using it. Follow up with the stakeholder will occur and any issues will be identified and corrected. Updates will be pushed as necessary.

## Specific objectives

To be determined as the project progresses.

# CHAPTER II: Background and Literature Review

## Introduction

The central problems that Message Wave is attempting to address has been researched and had numerous applications built to address these problems. The two areas that have contributed much are emergency notification systems, mainly government organizations and health care alert systems within hospitals and emergency rooms. Both fields need to push information to diverse groups of people and ensure they received it, read it, understand and then take action based on the information. Emergency notification systems were developed early by government agencies to notify organizations and communities of pending or ongoing emergencies. Health care alert systems were developed by hospitals to push information quickly, coordinate efforts and streamline the information process.

## Goals of Emergency Notification Systems

The goals of the early emergency notification systems were four-fold. They were to reach the target audience, capture the attention of the audience, ensure understanding of the message and have the audience respond appropriately (Harris et al., 2022). Often this was difficult because with few exceptions the groups they were trying to reach typically did not go searching for warnings or updates, therefore notifications needed to be suitably conspicuous and have certain characteristics that encouraged individuals to read, understand, personalize, and store in memory the content that was being sent out. (Harris et al., 2022). The sender also had to ensure they sent accurate information that had been verified and was not contradictory. The message also had to be viewed as credible for the recipient to internalize the message and take action. Any sender who failed at either sending accurate information or being perceived as credible suffered a reduction in desired actions. (Harris et al., 2022). Building credibility and trust took time and effort. The messaging system had to establish itself as a credible source of information by consistently providing accurate and up to date information to a wide audience. The source also had to limit the notifications and messages it sent to only those necessary and relevant to the audience.

## Reaching the Target Audience

Reaching the target audience is the first goal of any notification system. There are numerous ways an emergency notification could be sent. The easiest and most cost-effective way to reach a target audience was and still is email. Although it is cheap, easy to use and has the ability to reach most individuals it is not an effective way to reach any audience. It is estimated that at least a quarter of emails never even reach a recipient in box due to spam filters. (Jenkins, 2009). In addition, it is estimated that email is opened and read at a rate as low as 10 to 25 percent. (Ray, 2012) Therefore, due to the fact that email often fails to reach a recipients in box and those that do are read at a very low percentage email must be ruled out as a viable messaging medium. This indicates why emergency notification systems have historically not used email as the primary and or only way to send messages.

According to European Emergency Number Association (EENA), there are three major ways of pushing effective emergency messages. The first is voice calls which are available worldwide, emergency SMS (eSMS) which are available in some regions only and Mobile Emergency Applications which are available in some regions only and require the user to download the application. (Repanovici, R., & Nedelcu, A., 2021). The greatest benefit of voice calls are they are available everywhere and do not require the user to pre-register or download an application. This removes hurdles from the process and ensures all users will be able to receive the message. SMS does not require the user to download an application but does require them to pre-register in order to receive messages. SMS messaging is used extensively by individuals and according to the latest marketing statistics, 23 billion text messages are sent each day worldwide. (Repanovici, R., & Nedelcu, A., 2021). This is a staggering number and illustrates the wide use of and familiarity of text messages. The mobile emergency application requires the user to both download an application and pre-register. Although the mobile application requires the most steps to use it is deemed to be the best medium to reach the target audience regarding emergency notifications after research was conducted by Repanovici and Nedelcu. The main objective of their research was to illustrate that mobile emergency alert apps can support the Emergency Management system throughout the world. They conducted the decision-making process by evaluating three methods for communicating: Voice Call, SMS and Mobile Applications. They used the Multi-Criteria Analysis to establish the relative weights of evaluation criteria. (Repanovici, R., & Nedelcu, A., 2021) After considering eight test areas Repanovici and Nedelcu concluded a mobile application was the best medium to use to reach the target audience. Through their research they found a mobile application was the cheapest, most accurate, easiest to use and prompted the timeliest response from users.

Figure 1: Multi-Criteria Analysis to establish the relative weights of evaluation criteria

A table with numbers and a number of text

Description automatically generated with medium confidence

Note. Table produced by Repanovici, R., & Nedelcu, A. representing multi-criteria analysis to establish the relative weights of evaluation criteria. Repanovici, R., & Nedelcu, A. (2021). Mobile emergency notification apps: current state, barriers and future potential. IOP Conference Series (p. 12049). Materials Science and Engineering.

The mobile application was ranked the best when compared to voice calls and SMS messaging, even considering it involves more steps for installation and use. This is despite the common problems associated with mobile apps. These problems include the user has forgetting that they installed an emergency app; the emergency app not being intuitive enough; it being too complicated to find the emergency app in a stressful situation and an emergency application simply not being installed. (Repanovici, R., & Nedelcu, A., 2021).

## Capture the Attention

It is estimated that cell phone users get over 60 message alerts on their phones a day. (Pielot, Church, K., & de Oliveria, R., 2014). Many of these messages are unimportant and often distract a user from their work. Therefore, it is essential that any message sent is not associated with the other irrelevant or extraneous messages users receive. In addition, emails and messages through social media applications were correlated with negative emotional responses whereas messaging applications were showed a positive response. Researchers attributed the positive responses due to the personal nature of most messages coming through messaging applications. (Pielot, Church, K., & de Oliveria, R., 2014). It is necessary the messaging application is used only for its intended purpose and is not used for unimportant or irrelevant announcements and only sends a message when the receiver needs the information or requires action from them. If possible, delivering messages at a consistent and re occurring time helps capture the attention of users. The conclusions are, only message when necessary and with relevant information. When doing this not only will the attention be captured but action is more likely to be taken.

## Ensure Understanding

The health care field has contributed much to the research of ensuring a message is understood by those that receive it. This aspect is critical within the health care field due to the health and safety risks associated with mis interpreted information. In addition, the increased complexity of chronic illness increases the need for communication to be clearly understood. Patients can suffer pain, injury and death if a message is not understood properly. (Tazegul, et al., 2017)

Instant messaging applications have been used extensively in the healthcare field to push information to individual doctors, nurses and other health care professionals with success. These applications allow information to be shared across the organization and eliminate the vertical reporting system that often leads to information siloing and the morphing. (Tazegul, et al., 2017) Messages, photos, scans and even training material is pushed through the messaging application and the results have been favorable. Health care professionals report positive experiences and hundreds of hours saved due to the messaging applications and the information disseminated through them. Information is passed quickly to everyone who needs it. Clinics around the world have reported the adoption of some type of instant messenger as their primary mode of communication. Clinics state they share patient details and images that reduce the time of shift changes. They also state that it reduced consultation time, ameliorated surgery performances and reduced clinical incidents. The use of an IM also kept staff up to date on cases. (Tazegul, Bozoglan, Ogut, & Balci, 2017) Through implementation of IM applications, the Health Care professionals determined for messages to be understood the message should conform to certain parameters. These include familiarity with the message format in order for users to quickly and efficiently identify critical information and understand how information is presented. The message also needs to use simple and concise language so as not to lose the reader’s attention or cause confusion.

### Ensure Understanding – The Problems

Arguably more important than what is in a message is what is not included in terms of ensuring understanding. The health care field identified numerous issues. Acronyms, jargon and technical terms should be avoided unless all the users are familiar with them. Other problems have included spelling and grammatical errors within messages due to the ease and speed of use of the system. Patient privacy was also a concern with the mass amount of information regarding patients being circulated to so many people. Relying on messaging applications for all communications has led to issues and health care professionals still needed face to face conversations to discuss more specific or technical issues. Complaints and problems identified have included reduced face to face communication, circulation of unnecessary information and feeling the need to be involved around the clock. (Tazegul, et al., 2017) Despite these issues messaging applications are widespread and effective in the health care profession.

## Respond Appropriately

Capturing the attention of a user shares many of the same characteristics with getting users to respond appropriately. If a message captures the attention of a user because the information is personalized, relevant and timely, these same characteristics of the message will likely prompt the user to respond appropriately if a clear and concise call to action is made. This call to action must be clear, reasonable, and appropriate to the situation. Failing to include a call to action is failing to illicit an appropriate response.

## What has been done

Numerous notification applications have been built by governments and the private sector. The US Federal Emergency Management Agency (FEMA) already supports the Wireless Emergency Alert (WEA) service. This service sends alert messages to the public on WEA-capable mobile devices via the SMS Short Message Service Cell Broadcast (SMSCB) (Falcão, et al., 2018) This emergency messaging system is familiar to most people.

Figure 2: A Sample WEA Message

A screenshot of a phone

Description automatically generated

Note. From Falcao, J., Krebs, J., Kumar, S., & Erdogmus, H. (2018). OpenAlerts: A Software System to Evaluate Smark Emergency Alerts and Notifications. Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers, (pp. 1250-1255).

CityWatch is used by hundreds of government agencies and corporations to support disaster recovery as well as push emergency notifications. The service is also used for automated scheduling and business continuity communications. (NewsRx, 2011) ActiveMQ is an open source MOM application for community and academic projects. ActiveMQ provides features that can be accessed in programming languages such as Java, C, C++, .NET, and Python. (Bish, 2013)  OpenAlerts is an application that aims to improve the WEA alerts that are utilized by FEMA. OpenAlerts is composed of three main sub-systems: a mobile application, a server that includes an Alert Originator dashboard and a push notification service. The dashboard is designed to allow the creation, scheduling, modification or deleting of messages. (Falcao, Krebs, Kumar, & Erdogmus, 2018) The push notification service sends metadata in JavaScript Object Notation (JSON) and it uses Amazon messaging service to push alerts to phones. (Falcao, Krebs, Kumar, & Erdogmus, 2018) OpenAlerts also uses MySQL for persistent storage of data.

Figure 3: High Level OpenAlerts Architecture

A diagram of a cloud computing system

Description automatically generated

Note. From Falcao, J., Krebs, J., Kumar, S., & Erdogmus, H. (2018). OpenAlerts: A Software System to Evaluate Smark Emergency Alerts and Notifications. Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers, (pp. 1250-1255).

Figure 4: System Architecture of OpenAlerts

A diagram of a software system

Description automatically generated

Note. From Falcao, J., Krebs, J., Kumar, S., & Erdogmus, H. (2018). OpenAlerts: A Software System to Evaluate Smark Emergency Alerts and Notifications. Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers, (pp. 1250-1255).

## Why We did what we did

Many of the emergency notification and health care applications discussed previously have become widely used and trusted. They serve as a basic framework for building an application with similar goals, albeit for a different objective. After considering what has been done with these applications, Message Wave aims to be an application that sends messages through the application itself, something that has proven to be the most effective. In addition, the message will be sent in a pre-arraigned format to ensure familiarity and assist with understanding. As much as possible the application will ensure messages are personalized, relevant and timely with a clear and concise call to action.

## Privacy Concerns

Message Wave will have to address privacy concerns and will look to previously built applications in the health care field for both solutions and to identify issues. Health Care professionals have access to very personal health data and therefore must address privacy concerns every day and in nearly everything they do. Their work is governed by many different entities including the Health Insurance Portability and Accountability Act (HIPAA). HIPPA requires organizations ensure the confidentiality and integrity of patient information, protect against anticipated threats against disclosures and unauthorized uses. (Liu, et al., 2019). As health care professionals have used messaging applications there have been violations of HIPPA and major concerns regarding the use of these applications. The applications were not necessarily built to comply with HIPPA and the simple act of sending an unencrypted message with health care data violates HIPPA, which many applications do. In addition, human error often occurs and health care data is sent to individuals who are not supposed to view it. Although Message Wave will not be dealing with patient information the regulations outlined in HIPPA could serve as an outline for addressing privacy concerns. Message Wave will have names and phone numbers of numerous individuals and ensuring this information stays private is necessary. User data could be disclosed to unauthorized users by numerous means as individual devices can be lost, stolen or passwords and pins could have weak requirements, or the device could be left unlocked and expose the data to outside sources. Another concern is unencrypted message traffic. Unencrypted messages can expose the data to unauthorized access. Within the European Union the General Data Protection Regulation (GDPR) aims to give citizens greater control over and protection of their data. (Masoni & Guelfi, 2020) The GDPR regulates where user data can be stored and that organizations must know where and how data is stored. (Masoni & Guelfi, 2020). Unfortunately, the U.S. does not have an overarching data protection or privacy act. The U.S. Privacy Act only applies to Federal Organizations and only a handful of individual states have enacted data protection laws. Due to a lack of a federal policy on privacy, Message Wave will inform users how their data will be used and stored in the form of a Terms of Use Agreement. It will also utilize built in Android security measures but will unfortunately be unable to encrypt messages.

## Intellectual Property

Intellectual property, like personal property is characterized by things that only the owner, or those that have the owner’s permission can do. These include giving it away, selling it or licensing it for others to use. They may also prevent it from being reproduced, copied or sold. (Rosen, 2005) In regard to Message Wave any software that was produced by someone that owns the intellectual property rights requires their permission to use it. A license agreement is a way to grant permission to use software and Android Studio “grants a limited, worldwide, royalty-free, non-assignable, non-exclusive, and non-sublicensable license to use the SDK solely to develop applications for compatible implementations of Android.” (Google, 2024). This will allow the use of Android Studio for development and implantation of the application. In addition, any open-source software can be used be used by anyone, anywhere, for any purpose, even if it is licensed under academic or reciprocal licenses. (Rosen, 2005). Therefore, any open-source software can be used. Certain versions of SQL databases are open source, and the application will use an open source version of SQL.

## Building Trust

ADD IF NESSARY Capturing the attention and getting an appropriate response has been covered previously. Integral to this is building trust with the user.

## What Has Not Been Investigated

ADD IF NESSARY List the things that have not been used for text messaging. Alerts, fax..

## Case Studies

ADD IF NESSARY Case study A: This emergency happened and this many people received the message and did the right thing

Case study B: …

## Fundamental Issues

ADD IF NESSARY Interfacing with SMS service, app stores, storing data, privacy issues

## Development Environments and Deployment

ADD IF NESSARY Using swift, android app

What have apps been deployed on before or what language were they written in? Java,

## Testing Methods

1. Local emulator testing
2. Internal Testing
   1. Publishing to App store
      1. Signing APK and ABB files
      2. Creating privacy policy
         1. Publish website
      3. Publish initial release to internal testers
   2. Download and test on real device
      1. Testing Procedure (Maybe make this into a different format)
         1. Features
            1. App display compatibility

Ensure each screen is properly displayed and that necessary information and buttons are available where they should be.

* + - * 1. Group

Add mulitiple

Edit names

Delete group

Group screen shows members info and is scroll able

* + - * 1. Contacts

Manually

Via CSV in local filesystem

Edit contact info

Delete Contacts

* + - * 1. Send Messages

Message customization

Contact selection

Select all

By group

Unselect all

By member type

header works and grabs appropriate name selection

First name

Full name

custom

none

footer

Saves last footer to load for next new message

message body

Messages send individually to each member with correct message information

Message data stored in message history

Failed messages are flagged in message history

Message History Screen

History properly displayed in correct order

Clickable messages to see failed recipients and message content

Sortable by group

Tutorial Screen

Loads and displays tutorial information

Reset functions

Reset all data

Reset group data

Reset messages

Reset member data

Make sure all data is saved and reloaded properly when closing the app, and reopening the app.

* 1. Stakeholder/Tester Feedback
     1. Functions
        1. Add debug symbols and debug info
     2. Display
        1. Home Screen
        2. New Message Screen
        3. Message History Screen
        4. Tutorial Screen
        5. Group Screen
  2. PreLaunch - Issues that caused the app to crash, or stop responding.
     1. Issues
        1. 5 issues found at first in “Stability” crashing on certain devices
     2. Fixes
  3. Launch

# CHAPTER III: System Design

## Introduction

Introduction text to be written last.

## Overall Approach

The project will use a plan based, also known as the waterfall methodology for this project as requirements are largely clear and fixed. Requirements will not change in any drastic way and when requirements are unlikely to change a plan-based methodology has proven to be the best approach (Butler et al., 2020). Agile development will not be implemented as the user will not be heavily involved, requirements are not likely to change, nor will multiple deliverables be produced which are hallmarks of agile development. (Dawson & Dawson, Software Development Process Models: A Technique for Evaluation and Decision-Making, 2014)

## Goals

Message Wave has numerous goals. The first is to build a free to use application capable of storing a database of contacts and sending messages to a select group of those contacts through SMS services in the form of a text message. Another goal is to make the application easy to use with an intuitive interface. The final goal is to ensure any messages sent through the application are received, capture the attention of the user, the message is understood, and appropriate action is taken. This will ensure the application is beneficial to both the stakeholder and the recipient of any messages.

## Functional Requirements

* Upload and hold a database of contacts
* Add a contact to the database of contacts
* Delete a contact from the database
* Update a contact from the database
* One message payload entry
* Individual multiple messages sent to the members using their individual names in the greeting
* Using local SMS from device service if possible
* TBD based on stakeholder meeting

## Non-Functional Requirements

* Apple and Android compatible
* Database of contacts capable of holding 250 contacts
* Security to be determined
* Encryption to be determined

## Specifications

Android studio will be used for development of the user interface and backend. Android studio uses Java and Dart programming languages. Local storage on the device will store the database of contacts with the option to manipulate and delete contacts. Hardware specifications require a modern smartphone, tablet or personal computer with Operating system: 64-bit Microsoft Windows 8/10/11, or macOS 10.14 Mojave or newer.

## Tasks and Timelines

| **Task Name** | **Task Description** | **Task Assignment** | **Completion** | **Notes** |
| --- | --- | --- | --- | --- |
| Requirement Analysis | Gather and analyze project requirements | Team | Complete |  |
| Design Phase | Create design documents and prototypes | Team | Complete |  |
| Development Setup | Setup IDE and development environment | Team | Complete |  |
| Feature Development | Implement core features | Team - Various | Complete |  |
| Account Management | Building interface of account management | Schmidt | Canceled | Not building a login screen |
| Database of Contacts | Building database and Contacts | Schmidt | Complete |  |
| Message Payload | Message payload from user | Blauw | Complete |  |
| Processing | Processing members list and selection | Team | Complete |  |
| SMS Service | SMS service subscription/Local Device SMS connection | TBD | Ongoing | Awaiting physical device |
| Message Payload | Message Payload distribution | TBD | Ongoing | Awaiting physical device |
| User Interface | Design UI | Blauw | Complete |  |
| Testing | Conduct thorough testing of application | Schmidt | Ongoing | Awaiting one test to complete: send SMS message with physical device |
| Stakeholder Review | Present to stakeholders for feedback | Team | Incomplete | Awaiting build and meeting |
| Final Adjustments | Make necessary adjustments post-feedback | Team | Incomplete | Awaiting build |
| Deployment | Deploy the final product | Team | Incomplete | Awaiting build |

Table 1: Tasks and Timelines

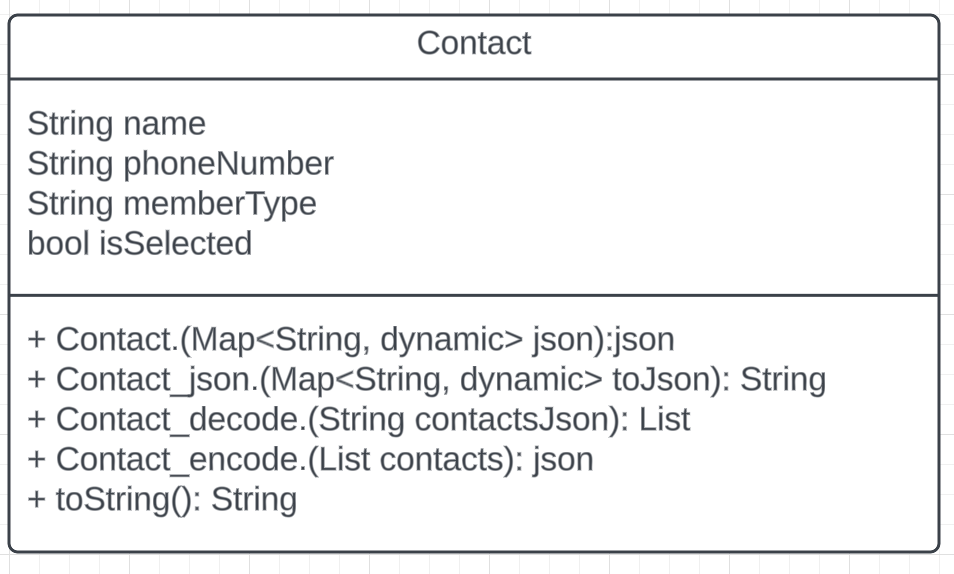
## Development

The first stage in development is identifying requirements. This was accomplished first with a meeting with the stakeholders to identify user requirements, confirm problems and goals. This was furthered by research into what had been done prior with emergency notification systems and health care alert systems. The next stage was to install the proper IDE and software packages for development. Android Studio was identified as the IDE and Java, then Dart was identified as the programming language that would be used within the Flutter framework.

## Modeling

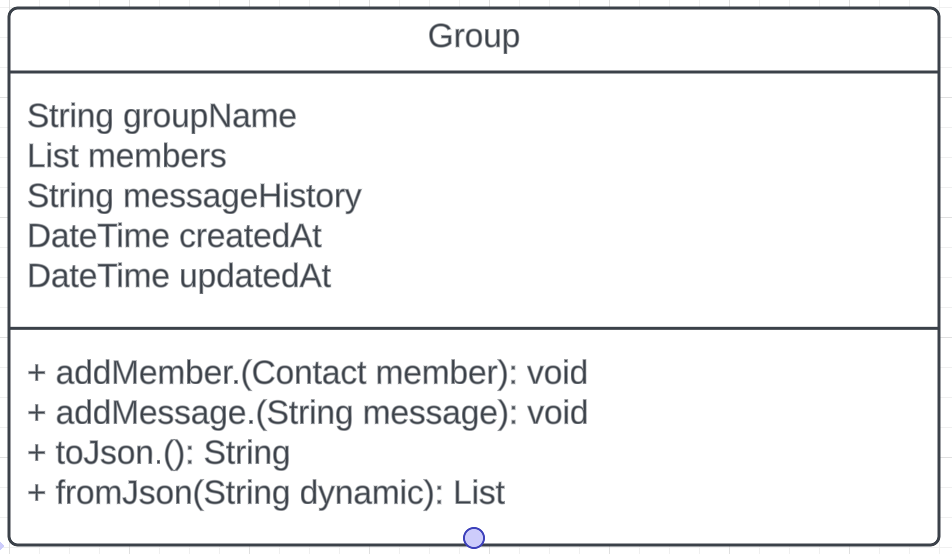
The application will follow a modular design, where each module can be developed and tested independently. The user interface will be designed for simplicity and ease of use, ensuring accessibility for all users.

Figure 5



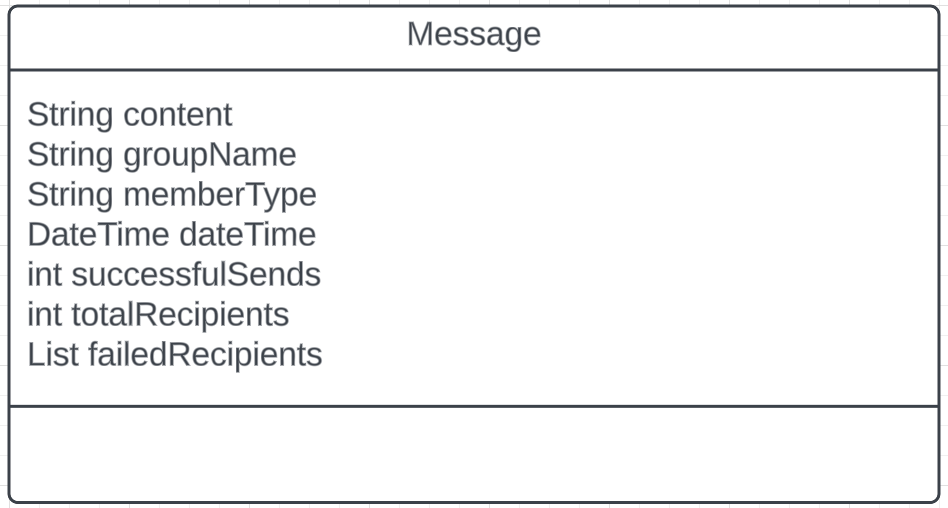
Note. Model of Contact Class. This class will construct each contact entered into the application. This class can be created from .csv import.

Figure 6



Note. Model of Group Class. This class will construct a group in order to put contacts into. Contacts can be loaded into the group via individual input or .csv input.

Figure 7



Note. Model of Message Class. This class will construct the messages used in the application. It will also store the date and time the message was created and the status of send results.

Figure 8

A screenshot of a computer

Description automatically generated

Note. Model of import .csv service. This interface takes a .csv file and imports contacts into a group.

Figure 9

A group of text on a white background

Description automatically generated

Note. Model of group service interface. This interface allows functionality within a group to load groups, save, delete, add contacts and load contacts.

Figure 10

A message on a computer

Description automatically generated with medium confidence

Note. Model of messaging interface. This interface allows sending of messages, storing of message history, personalization of a message and loading previous message history.

Figure 11

A screenshot of a computer

Description automatically generated

Note. Model of reset service interface. This interface allows the application to reset all data or chose which type of data should be reset.

# Chapter IV: Implementation

#### Introduction

#### To be written.

#### Account Creation and Management

The next stage in development was system and software design. This included building the applications features. Per the requirements account creation and management was needed as an application feature. This requirement was determined not to be needed and will not be included in the final build.

#### Database of Contacts

The next requirement to be addressed was a database of contacts. The database was created by using Flutter framework and the Dart programming language. The first iteration used Dart to create an SQL database where contacts could be added, removed and updated.

Figure 12: Creation of SQL Database

A screen shot of a computer code

Description automatically generated

Figure 13: Entering Contact into DB

A screen shot of a computer code

Description automatically generated

Figure 14: Updating Contact

A screen shot of a computer

Description automatically generated

Figure 15: Deleting Contact

A screen shot of a computer

Description automatically generated

As development progressed it was decided that local storage would be a better option for the application. The home screen was re designed with a “Create New Group” button to allow for the creation of groups of contacts that would be stored locally. For uploading large amounts of contacts, a comma separated value (csv) upload feature was implemented. Within the project the file was named csv\_service.dart and the class was CsvService Class. This feature allowed a csv file to be uploaded into the contacts, minimizing the time required to input large amounts of contacts.

Figure 16

A screen shot of a computer program

Description automatically generated

Figure 17: Home Screen with Group Creation

A screenshot of a video game

Description automatically generated

Figure 18: Creating New Group

A screenshot of a group

Description automatically generated

Once the new group was created contacts could be added in local storage and these contacts would include the name, phone number and the member type of the contact.

Figure 19: Contact

A screenshot of a computer code

Description automatically generated

Figure 20: Local Storage of Contact

A black background with colorful text

Description automatically generated

#### Message Payload

The next requirement called for a message Payload from the user. The first iteration of the development the user interface and message prompt was programmed using Java, the second and third iteration was programmed in Flutter.

Figure 21: Message Payload from User in Java Iteration 1

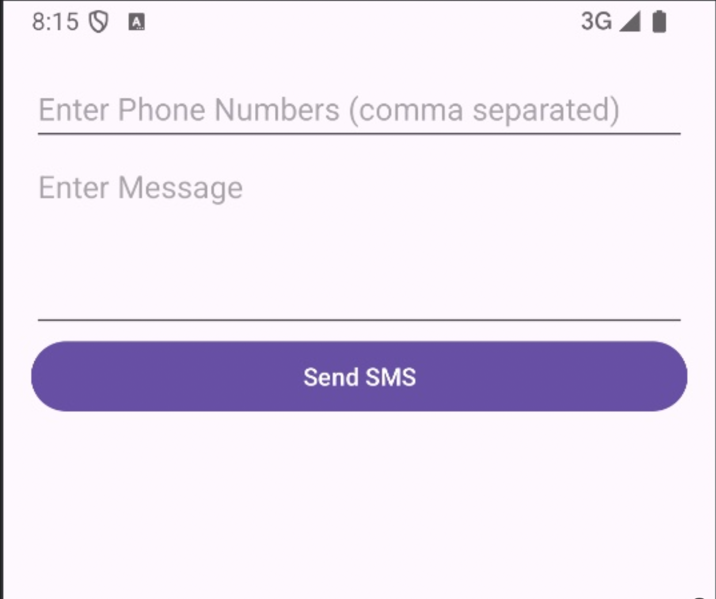
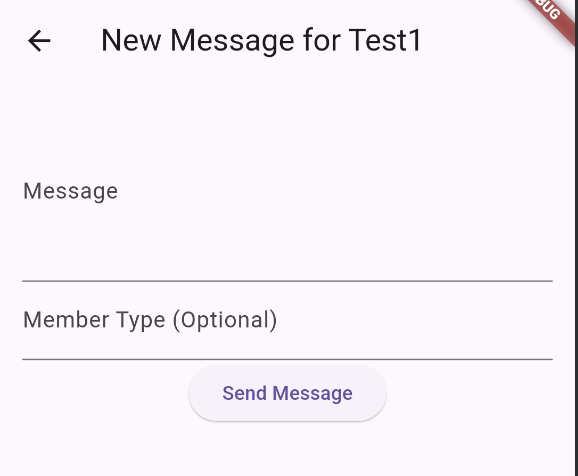


Figure 22: Message Payload from User in Flutter iteration 2



Figure 23: Message Payload 3rd Iteration



The third iteration went away from adding phone numbers as part of the message payload to just the message and an option to label the Member Type.

Figure 24: Fourth Iteration

A screenshot of a phone

Description automatically generated

The fourth iteration implemented a feature to select group members, either all of them or individually as well as a dropdown to include their first name, last name of full name in the message. The SMS service subscription/Local Device SMS connection development is still ongoing.

#### User Interface

The User Interface requirement has five screens, one home screens, a group screen, message history, new message and tutorial screen. The group screen allows the building of groups with name, phone number and a member type. The home screen allows a user to reset all the data within the application, this includes deleting all group data, log history and all messages. A user is also able to look at historical messages and get help on using the application. The group screen allows adding contacts, editing existing contacts and deleting contacts. The message history screen allows a user to select a group and see all messages sent to that group. Within this screen there is data available on how many recipients there were, the message itself and if there was any failures.

The new message screen will allow the creation of a new message for the selected group and a customized greeting. The tutorial screen will include a tutorial on how to use the different screens.

Figure 25: User Interface Screens

A screenshot of a computer

Description automatically generated

# Chapter V: Testing

#### Testing Concepts

Testing will involve multiple phases: unit testing, integration testing, system testing, and user acceptance testing (UAT). The aim is to ensure that each component works correctly individually and as part of the whole system.

#### Testing Process

* Unit Testing: Each module will be tested independently to ensure functionality.
* Integration Testing: Combined modules will be tested to ensure they work together seamlessly.
* System Testing: The entire application will be tested in a production-like environment to ensure it meets all requirements.
* User Acceptance Testing: Selected users will test the application to ensure it meets their needs and is user-friendly.

#### Unit testing

A series of five unit tests were completed on the application. This testing was done at the class level to ensure functions and classes withing the application were functioning properly. Where possible at least 10 tests were run with different inputs for each test. Tests that involved no input to test were only executed once. Testing the validation of user input failed several times and will need to be addressed with additional development.

EST CASE

UNIT TESTING

TEST NAME

PREDICTION

ITEMS BEING TESTED

PREDICT

SAMPLE INPUT

ACCURATE

EMOTIONS(AUDIO)

EXPECTED OUTPUT

EMOTION RECOGNITION

IS DONE BASED ON THE

INPUT FILES.

ACTUAL OUTPUT

SUCCESSFUL SPEECH

EMOTION RECOGNITION

REMARKS

PASS

EST CASE

UNIT TESTING

TEST NAME

PREDICTION

ITEMS BEING TESTED

PREDICT

SAMPLE INPUT

ACCURATE

EMOTIONS(AUDIO)

EXPECTED OUTPUT

EMOTION RECOGNITION

IS DONE BASED ON THE

INPUT FILES.

ACTUAL OUTPUT

SUCCESSFUL SPEECH

EMOTION RECOGNITION

REMARKS

PASS

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Number** | **Test Case** | **Unit Testing** | **Notes** |
| Unit Test 1 | Test Name | Validate Input into Create Group |  |
| Items being tested | \_createGroup() |  |
| Sample input | “Group 1” | Test iterated 10x with number, symbols, letters and a combination of each. Max length of input tested was 100 characters |
| Expected output | “Group 1” group created and displayed on screen |  |
| Actual output | “Group 1” group created and displayed on screen | Displayed each test as entered |
| Remarks | Pass |  |
| Unit Test 2 | Test Name | Invalid Input within Create Group |  |
| Items being tested | \_createGroup() |  |
| Sample input | No input |  |
| Expected output | Group is not created |  |
| Actual output | Group was not created and returned to home screen |  |
| Remarks | Pass |  |
| Unit Test 3 | Test Name | Validate Input into Add Contact |  |
| Items being tested | \_addContact() |  |
| Sample input | “Jon Doe”  123-456-7890 | 10x iteration with different names and different length of phone numbers |
| Expected output | “Jon Doe”  123-456-7890 |  |
| Actual output | “Jon Doe”  123-456-7890 |  |
| Remarks | Fail | Unit should prompt user to enter a valid phone number. Any combination of numbers from 1 to a string of 20 numbers was accepted |
| Unit Test 4 | Test Name | Invalid Input |  |
| Items being tested | \_addContact() |  |
| Sample input | No input |  |
| Expected output | Contact not created |  |
| Actual output | Contact was created with no name, phone number or contact type |  |
| Remarks | Fail | Unit should prompt user to enter values into Add Contact Screen |
| Unit Test 5 | Test Name | Individual name is included in message |  |
| Items being tested | buildCompleteMessage(Contact) |  |
| Sample input | “this is a test message” |  |
| Expected output | “Hello Brian Ignite this is a test message” |  |
| Actual output | “Hello Brian Ignite this is a test message” |  |
| Remarks | Pass |  |

Table 2: Test Case 1

#### Integration testing

Integration testing was completed on various interfaces and components to ensure they were integrated and functional. Two tests were completed and both passed. Multiple components from the application worked together to achieve the desired outcome.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Number** | **Test Case** | **Integration Testing** | Notes |
| Integration Test 1 | Test Name | Reset Data |  |
| Items being tested | \_resetData()  \_loadGroups()  resetServices.resetData() |  |
| Sample input | Clicking Reset Data Button |  |
| Expected output | Confirm reset and ability to check: Group Data, log history, messages, and reset all data |  |
| Actual Output | All data was reset, groups were deleted, message history and messages were all deleted |  |
| Remarks | Pass |  |
| Integration Test 2 | Test Name | Message History |  |
| Items being tested | \_MessageHistoryScreenState()  \_loadMessageHistory()  \_filterMessageHistory()  \_getUniqueGroupNames() |  |
| Sample input | Clicking History Button |  |
| Expected output | Readout of all previous messages sent |  |
| Actual Output | All test messages were displayed |  |
| Remarks | Pass |  |
|  |  |  |  |

Table 3

#### System testing

Two system tests were completed to verify the system meets requirements and works as a whole. A third test is planned to test sending an actual message through SMS. Both tests that were completed passed and no errors were found.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case** | **System Testing** | Notes |
| System Test 1 | Test Name | System testing in different operating system |  |
| Sample Input | Execute on Android and iOS device | \_createGroup()  \_addContact()  \_editContact()  \_NewMessageScreenState()  \_deleteContact()  groupService.deleteGroup() |
| Expected Output | Equal performance on both Android and iOS for creating group, adding contact, editing contact, building a new message, deleting contact and deleting a group | No noticeable difference between Android and iOS |
| Actual Output | Created group, added contact, edited contact, built new message, deleted contact, deleted group | No noticeable difference between Android and iOS |
| Remarks | Pass |  |
| System Test 2 | Test Name | Storing database of contacts |  |
| Sample Input | CSV sheet | .csv sheet with 270 names was uploaded into virtual device in Android Studio |
| Expected Output | 270 names added to group from .csv file |  |
| Actual Output | 270 names were added to group |  |
| Remarks | Pass |  |
| System Test 3 | Test Name | Sending SMS message |  |
| Sample Input | Sample message |  |
| Expected Output | Sample message is sent via SMS to another phone |  |
| Actual Output | To be determined, test pending |  |
| Remarks | Pending |  |

Table 4

#### Static testing:

Static testing was completed using the Dart Analysis tool within Android Studio. Analysis of this test is still ongoing.

Table 5: Dart Analysis

Dart Analysis

info: The prefix 'groupService' isn't a lower\_case\_with\_underscores identifier.

info: Invalid use of a private type in a public API.

info: Don't use 'BuildContext's across async gaps.

info: The prefix 'groupService' isn't a lower\_case\_with\_underscores identifier.

info: The prefix 'csvService' isn't a lower\_case\_with\_underscores identifier.

info: The prefix 'resetServices' isn't a lower\_case\_with\_underscores identifier.

info: Don't use 'BuildContext's across async gaps.

info: Don't use 'BuildContext's across async gaps.

info: Don't use 'BuildContext's across async gaps.

info: Don't use 'BuildContext's across async gaps.

info: The prefix 'messageService' isn't a lower\_case\_with\_underscores identifier.

info: Invalid use of a private type in a public API.

info: The prefix 'messageService' isn't a lower\_case\_with\_underscores identifier.

info: The prefix 'groupService' isn't a lower\_case\_with\_underscores identifier.

info: Invalid use of a private type in a public API.

info: Don't use 'BuildContext's across async gaps.

info: Don't invoke 'print' in production code.

info: Don't invoke 'print' in production code.

info: Don't invoke 'print' in production code.

#### Training Plan

The training plan includes the following steps:

User Manual: A comprehensive user manual will be provided, detailing how to use the application. This user manual is Appendix A.

Support: Ongoing support will be available to help users with any issues they encounter.

#### Updates

Updates will be produced at a later time.

## Support

Ongoing support will be available to help users with any issues they encounter.

# CHAPTER VI: RESULTS

A result is the final consequence of actions or events expressed qualitatively or

quantitatively. Performance analysis is an operational analysis, is a set of basic

quantitative relationship between the performance quantities.

A result is the final consequence of actions or events expressed qualitatively or

quantitatively. Performance analysis is an operational analysis, is a set of basic

quantitative relationship between the performance quantities.

A result is the final consequence of a series of actions that is demonstrated qualitatively or quantitatively. The application …..

# CHAPTER VII: CONCLUSION

To be written.

# REFERENCES

Bish, T. (2013). *Instant Apache ActiveMQ messaging application devleopment how-to: devleop message based applications using Active MQ and the JMS.* PACKT Publishing.

Butler, C., Vijayasarathy, L., & Roberts, N. (2020). Managing Software Development Projects for Success: Aligning Plan-and Agility-Based Approaches to Project Complexity and Project Dynamism. *Project Managment Journal*, 262-277.

Dawson, C. (2014). Software Development Process Models: A Technique for Evaluation and Decision-Mmaking. *Knowledge and Process Management* , 42-53.

Dawson, C., & Dawson, R. (2014). Software Development Process Models: A Technique for Evaluation and Decision-Making. *Knowledge and Process Management*, 42-53.

Falcao, J., Krebs, J., Kumar, S., & Erdogmus, H. (2018). OpenAlerts: A Software System to Evaluate Smark Emergency Alerts and Notifications. *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, (pp. 1250-1255).

Google. (2024, June 22). Retrieved from Terms and conditions : https://developer.android.com/studio/terms#:~:text=3.1%20Subject%20to%20the%20terms,for%20compatible%20implementations%20of%20Android.

Harris, C., Zerylnick, J., McCarthy, K., Fease, C., & Taylor, M. (2022). Breaking the Code: Considerations for Effectively Disseminating Mass Notifications in Healthcare Settings. *International Journal of Enviornmental Research and Public Health*.

Jenkins, S. (2009). *The Truch about Email Marketing.* FT Press.

Liu, X., Sutton, P., McKenna, R., Sinanan, M., Fellner, B., Leu, M., & Ewell, C. (2019). Evaluation of Secure Messaging Applications for a Health Care System: A Case Study. *Applied Clinical Informatics*, 140-150.

Masoni, M., & Guelfi, M. (2020). WhatsApp and other messaging apps in medicine: opportunities and risks. *Internal and Emergency Medicine*, 171-173.

NewsRx. (2011). Dairy Farming; Avtex Enhances its Emgerency Notification Software. *Investment Weekly News*, 159-.

NewsRX LLC. (2011). Avtex Enhances its Emergency Notificaiton Software. *Investment Weekly News*, p. 159.

Pielot, M., Church, K., & de Oliveria, R. (2014). An in-situ study of mobile phone notificaitons. . *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services*, 233-242.

Ray, J. (2010). The secret to high email open rates: how to write email subject lines that increase open rates and drive clickthrough percentages. *Campaigns and Elections*, Vol 33, Issue 1, 48-.

Repanovici, R., & Nedelcu, A. (2021). Mobile emergency notification apps: current state, barriers and future potential. *IOP Conference Series* (p. 12049). Materials Science and Engineering.

Rosen, L. (2005). *Open Source Licensing: software freedom and intellectual property law.* Prentice hall PTR.

Tazegul, G., Bozoglan, H., Ogut, T., & Balci, M. (2017). A Clinician's Artificial Organ? Instant Messaging Applications in Medical Care. *International Journal of Artificial Organs*, 477-480.

Toshev, M. (2015). *Learning RabbitMQ: build and optimize effecient messaging applications with ease.* Packt Publishing.

# Appendix A: User Manual

User Manual