Phone Calling Improvement Application

by

Matthew Richter, Tapash Phuyal, Xiang Qiu

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Matthew Richter	4/9/2019 Date
Tapash Phuyal	4/9/2019 Date
Xiang Qiu	4/9/2019 Date
Abdou Fall-Advisor	4/9/2019 Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 2019

Table of Contents Page

List of Illustrations

Tables

Figures

Acronyms and Abbreviations

UI	User Interface
API	Application Program Interface
XAML	Extensible Application Markup Language
OWASP	Open Web Application Security Project

Abstract

Many people dread calling up their cell phone company, electric company or any other company with an automated answering system to fix an issue. Currently, the only way to call them is to call and listen through the options before you hear the option you need which is a waste of time. Based on an article on www.westuc.com, the average consumer can expect to spend around 43 days of their life waiting on hold listening to the automated answering machine. Our Phone Call Enhancement App is a way to make this process easier. The app will have a list of companies and their phone number with department information. Once you select the department the number will open in your phone app ready to be dialed. Once you hit the call button the call will be sent to the specific department of your choice, so you do not have to listen through all the options. Having this application available to users will avoid time wastage waiting when calling companies with automated answering machines.

1. Problem Statement

1.1 Introduction

Calling companies with automated systems is a troublesome task. There is a lot of time wasted on the phone listening to options. This time should be able to be used better. The Phone Calling Improvement Application was built to make calling companies with automated systems easier.

1.2 Project Description

Phone Calling Improvement Application allows users to dial companies with automated systems more easily. The app has a list of companies and their department's direct line configured and when user select their choice the call will be sent to the specified department without having to listen to the answering machine.

1.3 Problem

When you dial a Toll-Free number or some businesses like Duke Energy, Cincinnati Bell, etc., the robot will start giving you automated options on what to dial to get to a department. It will be a few minutes before you figure out the options and reach the department you want to. Since people are busy with their daily lives, it's just a waste of time just listening to the automated options, instead, that time could be used to do some other productive work.

1.4 Solution

Our App avoids the time of waiting on the answering machine. When a user dials a number, for example, Duke Energy to report an outage, usually he will listen to multiple options before selecting the one he desires, which is the outage option. Our App has a list of companies with their phone number and department direct line configured. Once the user finds the company and selects their desired department the call will be sent to that specific department. This will not only save the user some significant time, but it also provides a digital phonebook which has

several companies and their departments' direct line. We are also giving the users the ability to add a new company's information, which is saved in the database, so others can benefit from it.

1.5 User Profile

The end user will be the only type of user that will be interacting with our mobile application. The potential users for the application will be any user who wants to make the process for calling companies with automated phone systems easier. Users will find the information for the company of their choice and will be presented with departments that they can choose from, which will send the call to that specific department which saves the time for a user since they don't have to wait to listen to the automated answering machine.

1.5.1 Project Title

Phone Call Experience Improvement App

1.5.2 Potential Users

Anyone looking to improve their experience when calling companies with automated answering systems with multiple options.

1.5.3 Software and Interface Experience

This application does not require any software, interface or related experience to use. The application will have a simple user interface which does not require any knowledge to get the full value out of the application.

1.5.4 Frequency of Use

Whenever a customer calls a number with an automated system.

1.5.5 Key Interface Design Requirements that the Profile Suggests

-Simple UI which lists companies and their departments available to the user on the screen

- UI will display options available to the user and what that option will bring to them (ex: Customer service to call the customer service department of a company)

1.6 Use Case Diagram

Figure 1: Use case diagram illustrates the role of users (Users with account and Guest User) and administrator.

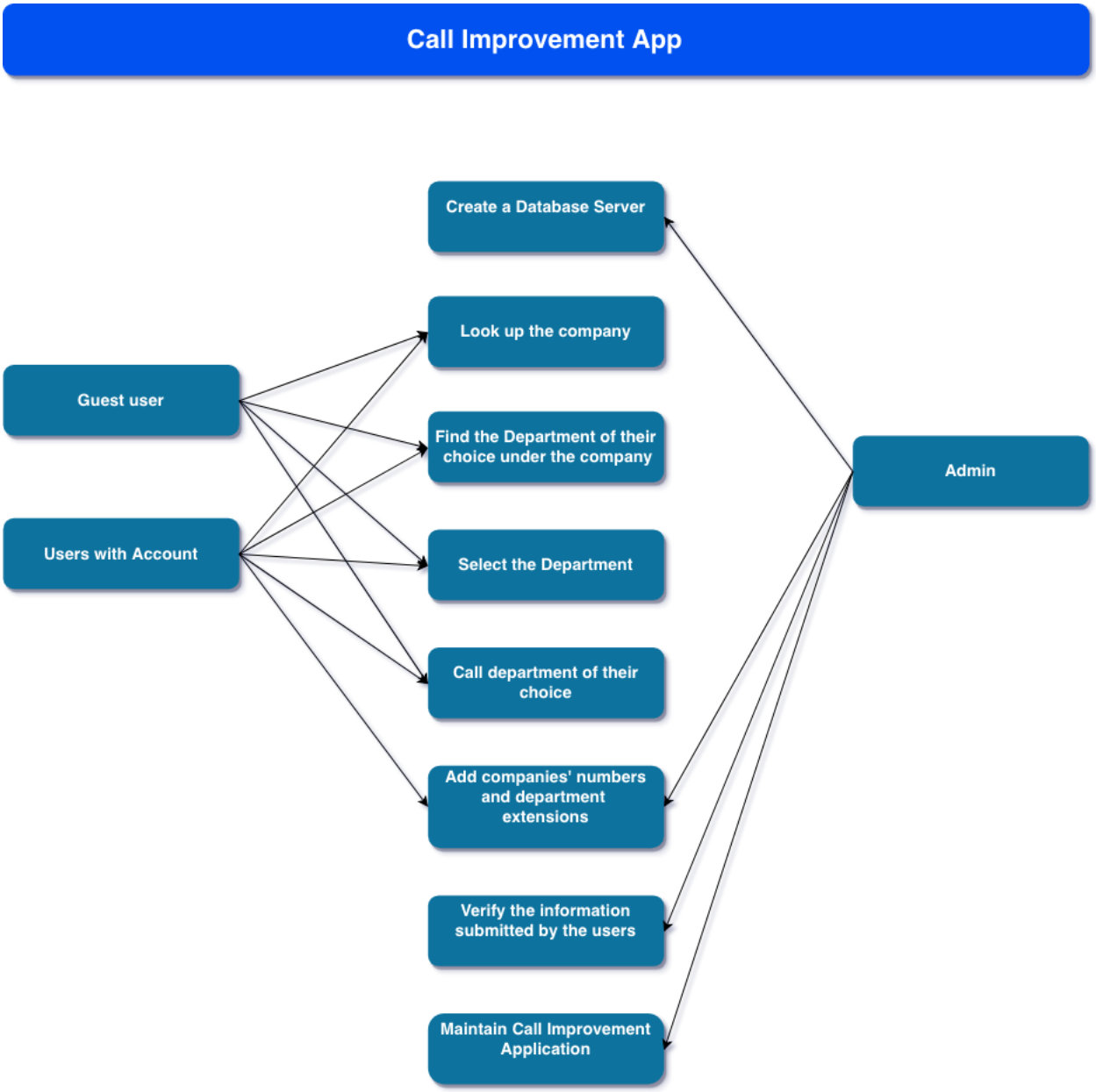


Figure 1: Use Case Diagram

2. Project Management

2.1 Budget

Table 1: Project Budget, displays the budget for our project. It includes the labor for all of the group members (2 developers, 1 cybersecurity) for creating the application on IOS and Android platforms. This doesn't reflect what it would cost for this application since we are not employed to create this and are working for free.

ITEM	UNIT #	COST	TOTAL
Labor	1000 Hours	$30	$30,000
TOTAL			$30,000

Table 1: Project Budget

2.2 Objectives/Deliverables

The project deliverables and deadlines are presented in Table 2

MAJOR PROJECT MILESTONES(Deliverables)			
FALL OF 2018 MILESTONES			
Planning	09/6/2018	User Interface Design	11/14/18
Development	11/09/2018	Presentation	12/03/18
SPRING OF 2019 MILESTONES			
Database Setup	01/15/2019	Application Backend	01/25/2019
Final Development	02/30/19	QA Test	03/7/19
Deployment to Store	03/15/19	IT Expo	04/9/19

Table 2: Project Objective/Deliverables

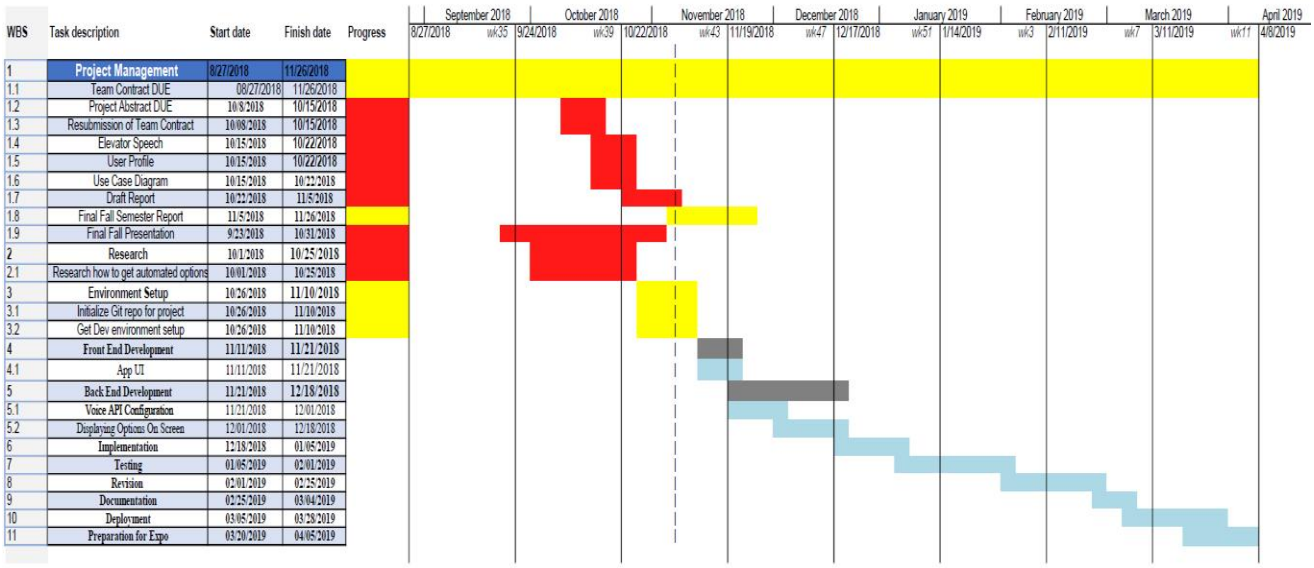
2.3 Project Schedule and Gantt Chart

Figure 2: Project Schedule lists our major tasks and dates for us to accomplish specific tasks.

1	Project Management	8/27/2018	11/26/2018
1.1	Team Contract DUE	08/27/2018	11/26/2018
1.2	Project Abstract DUE	**10/8/2018**	**10/15/2018**
1.3	Resubmission of Team Contract	**10/08/2018**	**10/15/2018**
1.4	Elevator Speech	**10/15/2018**	**10/22/2018**
1.5	User Profile	**10/15/2018**	**10/22/2018**
1.6	Use Case Diagram	**10/15/2018**	**10/22/2018**
1.7	Draft Report	**10/22/2018**	**11/5/2018**
1.8	Final Fall Semester Report	**11/5/2018**	**11/26/2018**
1.9	Final Fall Presentation	**9/23/2018**	**10/31/2018**
2	**Research**	**10/1/2018**	**10/25/2018**
2.1	Research how to get automated options	**10/01/2018**	**10/25/2018**
3	**Environment Setup**	**10/26/2018**	**11/10/2018**
3.1	Initialize Git repo for project	**10/26/2018**	**11/10/2018**
3.2	Get Dev environment setup	**10/26/2018**	**11/10/2018**
4	**Front End Development**	**11/11/2018**	**11/21/2018**
4.1	App UI	11/11/2018	11/21/2018
5	**Back End Development**	**11/21/2018**	**12/18/2018**
5.1	Voice API Configuration	11/21/2018	12/01/2018
5.2	Displaying Options On Screen	12/01/2018	12/18/2018
6	**Implementation**	**12/18/2018**	**01/05/2019**
7	**Testing**	**01/05/2019**	**02/01/2019**
8	**Revision**	**02/01/2019**	**02/25/2019**
9	**Documentation**	**02/25/2019**	**03/04/2019**
10	**Deployment**	**03/05/2019**	**03/28/2019**
11	**Preparation for Expo**	**03/20/2019**	**04/05/2019**

Figure 2: Project Schedule

Figure 3: Gantt Chart, shows the timeline of when each milestone will be done.



WBS	Task description	Start date	Finish date	Progress
1	**Project Management**	8/27/2018	11/26/2018	
1.1	Team Contract DUE	08/27/2018	11/26/2018	
1.2	Project Abstract DUE	10/8/2018	10/15/2018	
1.3	Resubmission of Team Contract	10/08/2018	10/15/2018	
1.4	Elevator Speech	10/15/2018	10/22/2018	
1.5	User Profile	10/15/2018	10/22/2018	
1.6	Use Case Diagram	10/15/2018	10/22/2018	
1.7	Draft Report	10/22/2018	11/5/2018	
1.8	Final Fall Semester Report	11/5/2018	11/26/2018	
1.9	Final Fall Presentation	9/23/2018	10/31/2018	
2	Research	10/1/2018	10/25/2018	
2.1	Research how to get automated options	10/01/2018	10/25/2018	
3	Environment Setup	10/26/2018	11/10/2018	
3.1	Initialize Git repo for project	10/26/2018	11/10/2018	
3.2	Get Dev environment setup	10/26/2018	11/10/2018	
4	**Front End Development**	11/11/2018	11/21/2018	
4.1	App UI	11/11/2018	11/21/2018	
5	**Back End Development**	11/21/2018	12/18/2018	
5.1	Voice API Configuration	11/21/2018	12/01/2018	
5.2	Displaying Options On Screen	12/01/2018	12/18/2018	
6	**Implementation**	12/18/2018	01/05/2019	
7	**Testing**	01/05/2019	02/01/2019	
8	**Revision**	02/01/2019	02/25/2019	
9	**Documentation**	02/25/2019	03/04/2019	
10	**Deployment**	03/05/2019	03/28/2019	
11	**Preparation for Expo**	03/20/2019	04/05/2019	

Figure 3: Gantt Chart

3. Technical Elements

3.1 Network (Hardware/Infrastructure)

We offer application on both Android and IOS platform. We have a server hosted in Microsoft Azure which hosts our SQL database for the backend.

3.2 Application

We have developed the application with Xamarin and C# programming language because this offers cross-platform development for both Android and IOS. Xamarin uses XAML pages for the frontend and C# pages for the backend. For UI design, we used Photoshop for logo creation and image processing. The application relies on a database hosted on our server in Microsoft Azure. Upon opening the app, a call is made to the database which loads all companies stored in the database. The app uses XAML pages and C# to support adding of additional companies to the database and allowing direct calling of companies right from your phone.

Figure 4: Technical Diagram, provides visual illustration of how this application works.

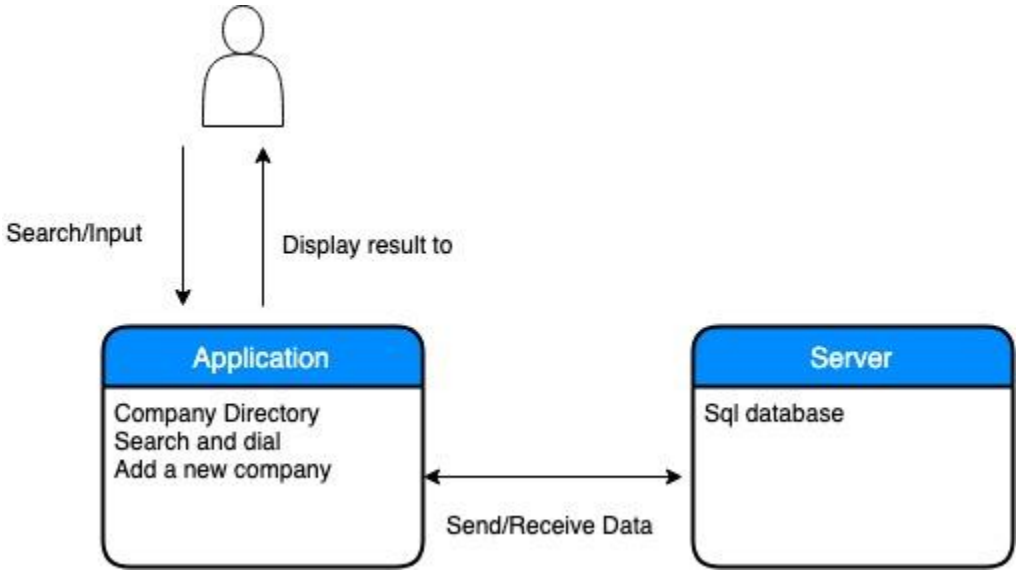


Figure 4: Technical Diagram

3.3 Database

Currently, for our database, we are using Microsoft SQL server 2016. The database will store company directories, user account information, and new numbers contributed by users.

3.4 Security

Safe transmission of data between user-end and database is very important to the integrity of our app.

4. User Interface

Figure 5: User interface, illustrates how users interact with the application

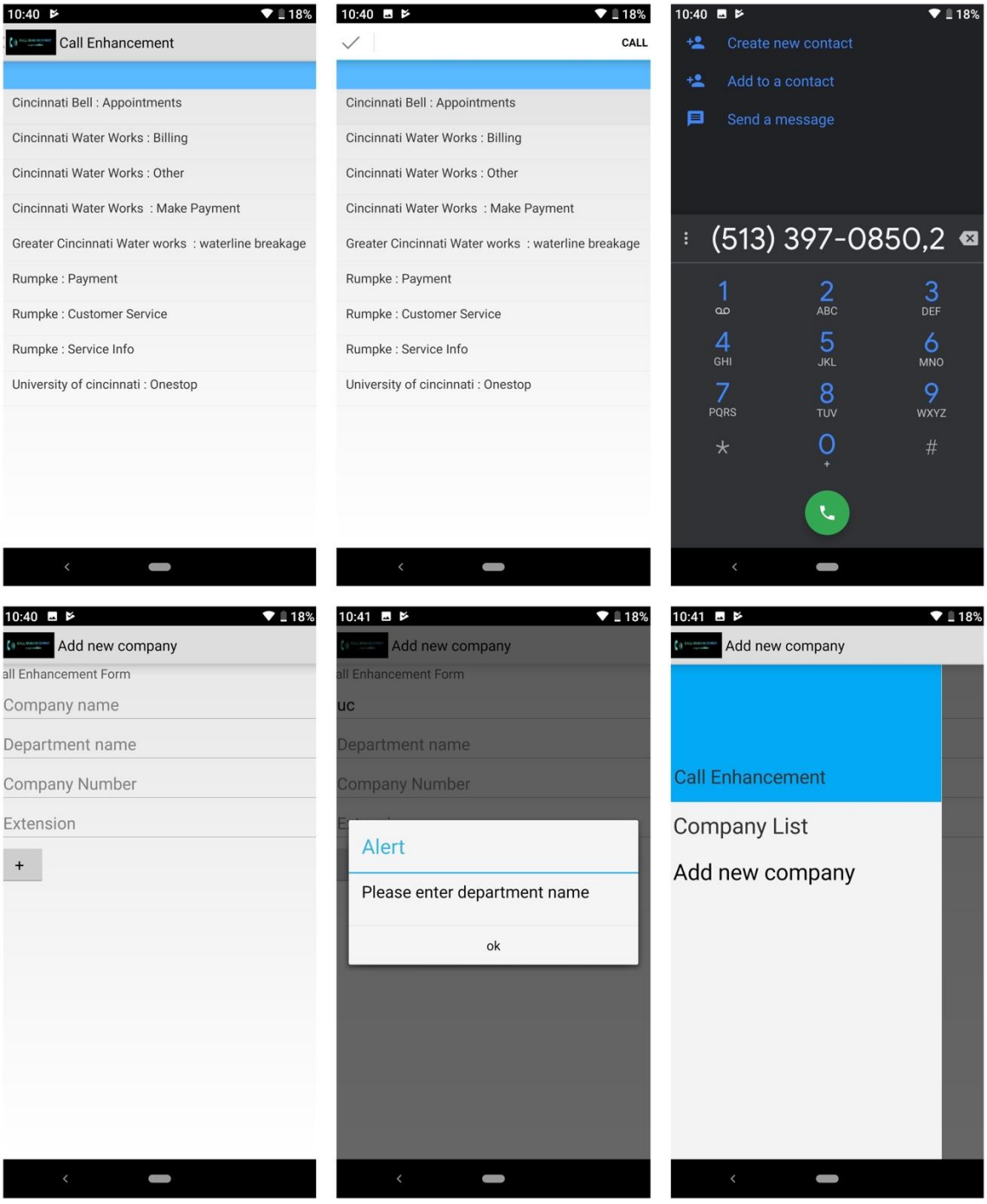


Figure 5: User Interface

Company List: This page will show the information of the company and list of departments.

Figure 6: Company List, shows company list within the application

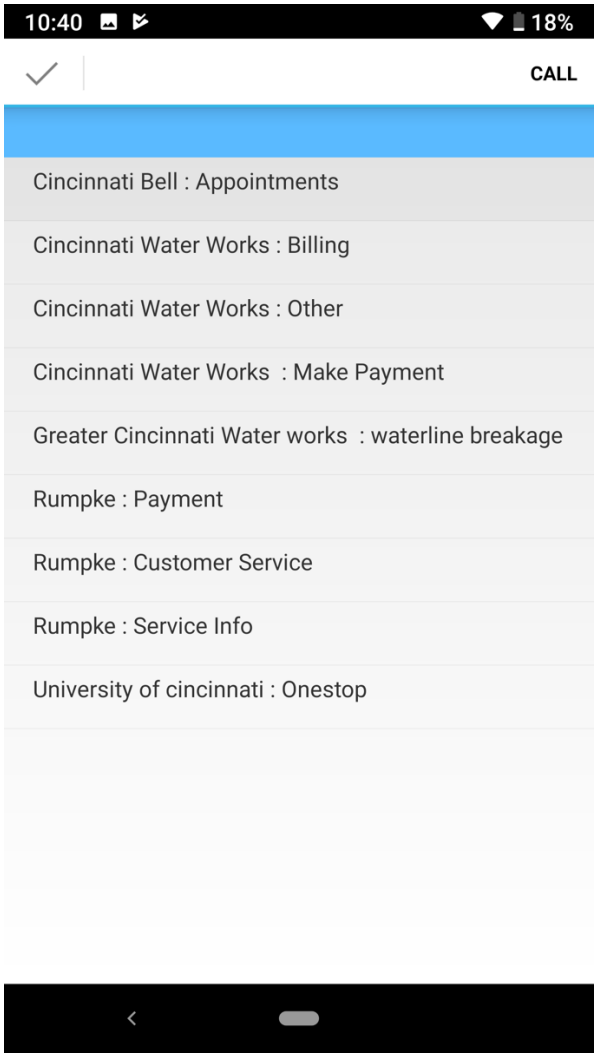


Figure 6: Company List

Add a company: Using this form users will be able to add new company's information to add to the database.

Figure 7: Add a company, shows add a company page within the application

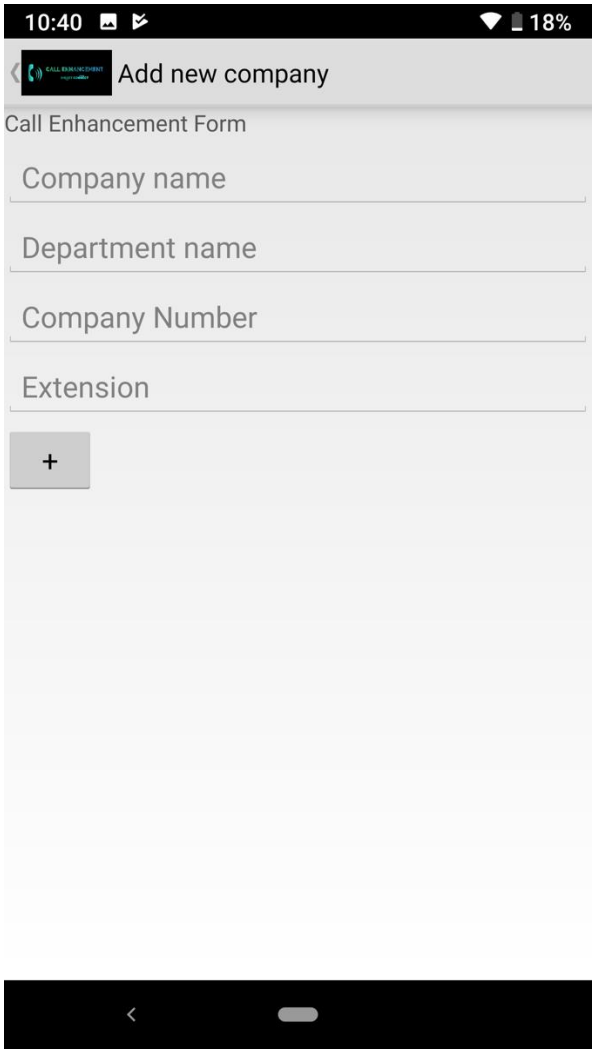


Figure 7: Add a company

5. Test Plan

Overview

For our testing, we have manually tested application. We may add in some automated testing but with our app, there isn't much that could be automated for testing. The main functionality of our app is creating phone calls, which to our knowledge is unable to be automated, this is why we manually tested. This section will explain the testing methodology we used for the Call Enhancement App. The following individuals will use this section as test guidance:

1. End Users
2. Developers

Scope

The scope of testing is to see the functionalities, user interface of this application on an Android device and to test the accessibility of devices included in the process. We ran the test for the app on Android phones running Android 6.0 and higher. The tests were to ensure everything in the app is working correctly. We chose to go with Android 6.0 and higher as most Android devices run on at least this version of Android. The tests are organized based on the application's requirements.

Objective

The objective was to verify that the app runs and works as expected. We did the test for the following:

1. Creating an account
2. Logging into the created account
3. Dialing existing numbers in the address book
4. Adding a new number with the form
5. Verifying new number shows in the address book for current and other users
6. Dialing new number added with forms works as expected

Entry and Exit Criteria

Entry Criteria:

- Complete building the application
- Testing the design process is written and validated

Exit Criteria:

- The application is tested and is working in an acceptable manner
- Bugs and issues are documented and fixed
- Fixes are approved by software developer
- Make sure the solution proposed for the project is fulfilled

Logging Test and Procedures

If a bug is found while testing it will be documented and developers will investigate recreating the bug and work on a fix or solution for the issue. For procedures, we will have our

tester go through the above steps and document any feedback that they may have about UI/ functionality of the app.

Problem Encountered and Solution

While in the process of developing the application, we were having trouble finding a database which we could use to get the numbers for different companies. As we progressed further, we had to make a slight change to our initial idea and created a database which stores the numbers for different companies. Some of the entries were made by the developer team and on users have an option to add new numbers if that does not already exist in the database and it will be available for other users.

What We Learned During Testing

We learned that we had a few bugs in the create account portion of the app initially, so we focused on getting those fixed. We also thought it might be a good idea to implement a verification step with sign up such as an email or text message to check that the person trying to sign up exists. This can help prevent bots from creating multiple fake accounts in our app.

Test Results

We ran the mobile app security test for vulnerability scanning through the ImmuniWeb (AI for application security) online. This online application tests the security and privacy of the mobile application, it detects the top 10 Open Web Application Security Project (OWASP) and other weaknesses when the APK file provided.

Figure 8: Audit Summary, shows an audit summary for our app

As shown in the figure below, in general, no issues were found for Mobile Application Behavior, Software Composition Analysis, External Communication, and OWASP. It requires attention for the Backend APIs and Web Services.

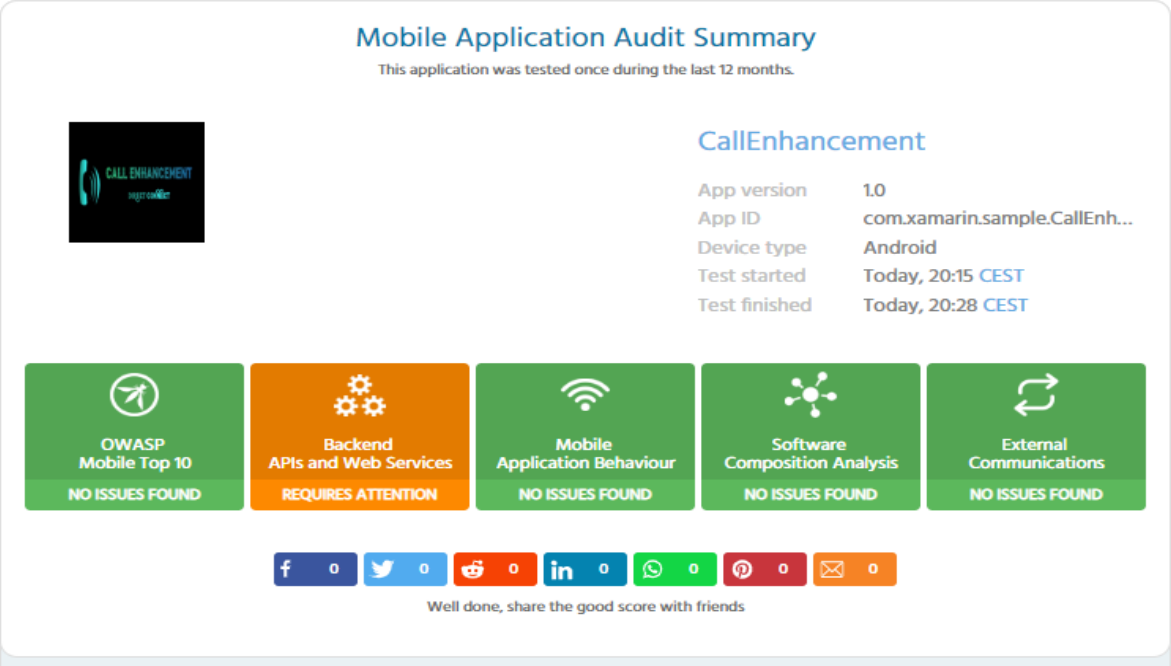


Figure 8: Audit Summary

Figure 9: Warnings found, shows 2 warnings found within the application

It detected two warnings for security flaws and weaknesses, but not any kind of risks.

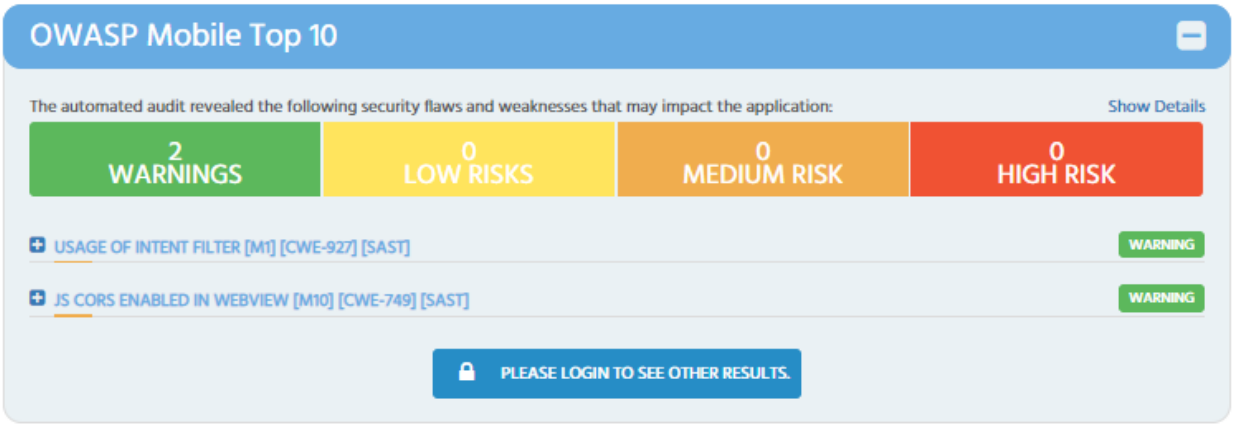


Figure 9: Warnings Found

6. Conclusion

6.1 Fall Semester

The focus of this project is on creating both Android and IOS applications which will allow users to search company numbers and save times by skipping the answer machines. We have been developing our application into Xamarin cross-platform. So far, we have spent a huge amount of time researching what kind of app we are creating and what practical features we are including.

6.2 Spring Semester

During the Spring Semester, we detailed the UI of our app by organizing functionalities into several sub-pages. Also added input-required validation on the Add Company Page. We finished setting up a database hosted in Microsoft Azure. We created a mobile service backend which is called by the app to get the database connection to load companies/add companies etc. In Azure, we have a server, and this server holds our Microsoft SQL server. We chose Azure over AWS (Amazon Web Services) because we encountered issues with permissions on the student account when going to implement with AWS.

Appendix A. Additional Information

The team contacted the following professionals for clarification and additional insight.

1. Abdou Fall, Advisor – University of Cincinnati
2. Eric Kornau, IT Consultant – Keyedge IT Auditing & Consulting

Appendix B. References

[1] Microsoft. “Part 1 – Understanding the Xamarin Mobile Platform - Xamarin.” *Microsoft Docs*, Mar. 2017, docs.microsoft.com/en-us/xamarin/cross-platform/app-fundamentals/building-cross-platform-applications/understanding-the-xamarin-mobile-platform.

[2] Microsoft. “Using SQLite.NET with Android - Xamarin.” *Microsoft Docs*, Apr. 2018, docs.microsoft.com/en-us/xamarin/android/data-cloud/data-access/using-sqlite-orm.

[3] Xamarin. “Xamarin/Xamarin-Android.” *GitHub*, 5 Oct. 2018, github.com/xamarin/xamarin-android.

[4] McMahon, Kevin. “Research Reveals How Much Time We Spend on Hold.” *West Unified Communications Services*, 17 Nov. 2016, www.westuc.com/en-us/blog/managed-voice-services/research-reveals-how-much-time-we-spend-hold.

[5] Poster