Software Requirements Specification

for

Smart Dialer

Version 1.0 approved

Prepared by Cameron Moore, Omar Qasem, Patrick Richardson, Matthew Spedale & Jeffrey Tolliver

CSC 4330

March 11th, 2017

Table of Contents

Revision History

Name		Reason For Changes	Version
Matthew Spedale		Initial draft	1.0

1. Introduction

1.1 Purpose

This Software Requirements Specification document outlines the functional and nonfunctional requirements, and constraints for our project, called “Smart Dialer”. This document compiles all of the documentation of the system and its requirements. It provides a detailed outline of our software. It describes the user interface, hardware requirements, functionality descriptions, and more. It also describes the intended client and audience of our product.

1.2 Document Conventions

This document follows the IEEE conventions provided by the Software Requirements Specification example document provided for the class. More specifically, we used the sections provided by the SRS document to properly divide each section by content. We also used bullet-point lists to indicate statements related to a header subject.

1.3 Intended Audience and Reading Suggestions

The intended audience for this document is both our professor, Dr. Nash, as well as for clients, who are looking to capitalize on an audience of Android cell phone users who are looking for a more powerful calling app than the default app provided on the Android phone. The suggested reading order of the document is to read sequentially from the first page to the last.

1.4 Product Scope

Our product is known as “Smart Dialer”. Smart Dialer is an Android mobile application that is intended to simplify making phone calls. The main feature of the application is an automated reverse phone number (411) lookup. For example, if the user were to call a phone number connecting them to Cox Communications, Smart Dialer will store the number you have called, perform a reverse phone number lookup on it for relevant information (company name, location if applicable, thumbnail if applicable, and email if applicable) and store it in the history list. This way, the user does not need to manually save or lookup phone numbers to make phone calls.

Our strategy is to release this app on the Google Play Store as an alternative to the existing phone dialing application installed on Android products, to offer Smart Dialer as a more powerful and user-friendly application.

1.5 References

This document uses the following resources as references:

- IEEE Std 830-1998 (Source: https://standards.ieee.org/findstds/standard/830-1998.html)
- Google Material Design Guidelines (Source: https://material.io/guidelines/)
- Whitepages Pro Global Identity Verification Data API (Source: https://pro.whitepages.com/)

2. Overall Description

2.1 Product Perspective

Smart Dialer is an original product from our developers. It is not an addition to any other product, and only requires the user to own the mobile platform the application is released on to use. Our development team's only previous experience as a unit was developing a video game application for Android phones, so we have some prior experience working in that environment.

2.2 Product Functions

- Dial phone numbers to make phone calls
- Quick access to the most frequently dialed phone numbers.
- Automatically save previously sent, received, and missed phone calls in a historical list that is ordered chronologically
    - After dialing unsaved phone numbers Smart Dialer discretely uses the 411 API to retrieve information on that give number.
    - The history section can be filterable by sent, received, missed, or all calls.
- Access your contact list conveniently
    - Access multiple contact lists (phone, SIM card, or Google)
    - Ability to edit contact lists such as: Name, Phone Number, Address, Email, and photo.
    - Ability to delete contacts
    - Ability to create contacts
    - Shortcut to send an SMS message using the user's default messaging app
- Personalization Features
    - Multiple color themes for user preference or readability
    - Set distinct ringtones for individual phone numbers

2.3 User Classes and Characteristics

The application only has one type of user class, simply named "user". The user is anyone who accesses the application. The user can utilize all of the outlined features of the application. There is no need for multiple user classes, since there are no higher-level functions that are accessed only by certain users, only one class is necessary.

2.4 Operating Environment

For the scope of the class, Smart Dialer's target platform is the Google Android platform versions 5.0 (Lollipop) and newer, as opposed to iOS, due to our experience with Android Studio and the cost of releasing to iOS. This means that any users will need an Android device to download and

use our application.

2.5 Design and Implementation Constraints

The main constraints of the project are following the specifications laid out by Dr. Nash, as well as time (since the class period is concluded by early May). As previously mentioned, we are also limiting ourselves to the Android platform, with APIs adhering to Android SDK 21 and above.

2.6 User Documentation

Documentation for Smart Dialer is provided within the interface under the settings section. After tapping the help button, a detailed description of the application and its purpose is visible to the user. Upon first-time use, the user is greeted with simple instructions as an overlay that outlines the application's features.

2.7 Assumptions and Dependencies

We are depending on the Android operating system for proper operation of the software. We are also relying on the White Pages Pro API for handling reverse phone number lookup.

3. External Interface Requirements

3.1 User Interfaces

Smart Dialer relies on Google's Material Design philosophy in regards to the interface. The interface is comprised of three main screens, each accessible via tabs on the top of the screen.

Every screen has 3 tabs horizontally laid out on the top, with a discrete, narrow settings button on the very top right to access the settings screen.

3.1.1 Dialpad

The first screen – the default screen which the user first encounters upon opening – opens with the software keyboard, defaulted to a full-sized keyboard with the option to switch to a T9 keyboard (or dialpad). Above the keyboard is a grid of most frequent contacts to act similar to "quick dialing." Above the frequent contacts, just below the tabs is the text field where the user's input is filled in. Here the user can input a name of a contact (from the saved contacts or one that has been reverse-searched via 411) or a phone number.

3.1.2 History

The second screen opens a chronological list of all previously sent, received, or missed phone calls. This is where the main feature of the software is most realized, since this is where any entry with a phone number matched with a successful search has its details populated with the related information. This is accessible via this screen, or the first screen upon entering the name (or number) of the contact. Each item entry has a name (if applicable), location (if applicable), phone number, photo thumbnail (if applicable), and a button to save the phone number to your contact list if it is not already present. The list is filterable via sent calls, received calls, missed calls, or all calls.

3.1.3 Contacts

The third screen opens the contacts list. The layout is similar to the second screen, but sorted alphabetically and separated by their first letters. Tapping on a contact on the list brings up a screen with more detailed information with options to edit and delete the contact. Above the list and below the tabs is a dropdown box that allows the user to switch between contact accounts (for example: Google's saved contacts, the phone's saved contacts, and the SIM's saved contacts)

3.1.4 Settings

The settings screen is comprised of options to give the user control over certain features of the device. In a list format the settings includes: smart search toggle (to enable or disable the phone number search feature), dark mode (a different color theme to help with eye strain or just for preference), about (more information about the app) and rate (the app on the Google Play store).

3.2 Hardware Interfaces

Naturally, a smart dialer application is limited to smartphones, and our first iteration is focused on the Android platform. The application runs on the Lollipop 5.0 operating system and newer.

3.3 Software Interfaces

3.3.1 JSON

All data retrieved from the phone number lookup service (411) is in a JSON format.

3.3.2 Android

Users interact with the Smart Dialer app through the Android operating system.
The Smart Dialer application hooks into multiple built-in Android system applications such as the Android Phone Service when connecting phone calls.

3.4 Communications Interfaces

3.4.1 Internet

411 services are done using HTML requests through HTTP/HTTPS. The data retrieved is in JSON format.

3.4.2 Phone calls

The smart dialer, upon pressing the call button, launches Android's built-in phone app to allow the user to make phone calls.

4. System Features

4.1 Dialpad

4.1.1 Description and Priority

The dialpad is a high priority feature that provides an interface for the user to input a phone number or name entry for the phone call.

4.1.2 Stimulus/Response Sequences

The user uses the dialpad to provide a phone number or a contact name to the app and the dialer populates the text field with that phone number or name. The application also tries to predict the name if there is any match with the input string.

4.1.3 Functional Requirements

REQ-1: The Smart Dialer must be on the Dialpad screen to display the dialpad.
REQ-2: The Smart Dialer must retrieve info from the history list and contact list upon user input to match strings or numbers.
REQ-3: The user must press the call button for the dialpad to launch the Android System phone app.

4.2 Speed Dial

4.2.1 Description and Priority

The speed dial is a medium priority feature that provides an interface that allows the user to save a number to quickly access for calling, similarly to how a speed dial function works on a cell phone.

4.2.2 Stimulus/Response Sequences

The user uses the dialpad to map one phone number per number button, allowing them to quickly call the number when held.

4.2.3 Functional Requirements

REQ-1: The speed dial must display the mapped phone number.
REQ-2: The speed dial must call the displayed phone number when the associated number is held.

4.3 Reverse Lookup

4.3.1 Description and Priority

The reverse lookup is a high priority feature for retrieving information on calls sent, received, or missed.

4.3.2 Stimulus/Response Sequences

The user sends or receives a call from an unknown number and the reverse lookup retrieves the information on the unknown number.

4.3.3 Functional Requirements

REQ-1: The system must check for internet connectivity before making the 411 request.
REQ-2: The user must receive or send a phone call from an unknown number before initiating the reverse lookup service.
REQ-3: The system must store the information received from the 411 request in the Smart Dialer history.

4.4 Smart Dialer History

4.4.1 Description and Priority

The Smart Dialer History is a high priority feature for displaying a history of all calls sent and received.

4.4.2 Stimulus/Response Sequences

The user opens the Smart Dialer History and the application shows information for phone numbers that either exist in the phone’s contact list or has been found from the Reverse Lookup.

4.4.3 Functional Requirements

REQ-1: The Smart Dialer must be on the Smart Dialer History screen to view the history.
REQ-2: The Smart Dialer must retrieve information from the contact list, call history and Reverse Lookup to match phone numbers to contacts.
REQ-3: The Smart Dialer History must display information for history items when the user taps on the item.

4.5 Contact Lists

4.5.1 Description and Priority

Contact list is an integral component to the dialer app and a high priority feature. The contact list contains the user's saved contacts, retrieved from the phone's storage, the SIM card's storage, or their Google account if applicable.

4.5.2 Stimulus/Response Sequences

When the user accesses the Contacts screen via contacts tab, they are presented with their list of contacts, sorted alphabetically and separated by their first alphanumeric character. From there the user can tap on a contact. This will bring up the contact's info in an expanded layout overlay. The user then can edit or delete the contact info.

4.5.3 Functional Requirements

REQ-1: The Smart Dialer must be on the contact list screen to view the contact list.
REQ-2: The user must tap on a contact to view expanded information.
REQ-3: The user must tap on the "edit" button to edit the contact information.
REQ-4: The user must tap on the "delete" button to delete the contact information.
REQ-5: The user must tap on the "done" button or outside the overlay region to close the expanded contact information.

4.6 Contact Edit

4.6.1 Description and Priority

Contact edit feature is an important sub-feature of the contacts list, allowing the user to edit contact information. This is a high priority feature.

4.6.2 Stimulus/Response Sequences

After the user taps on a contact and the more detailed information is presented to the user, an "edit" button is presented. Upon tapping the edit button, each information component becomes an editable text field that the user may tap and change any of those strings of information.

4.6.3 Functional Requirements

REQ-1: The user must tap on the edit button on the contact information overlay to edit the information components.
REQ-2: The user must tap on a text field to edit the the information.
REQ-3: The user must tap on “update” to save the contact information

4.7 Settings

4.7.1 Description and Priority

The settings page is a medium priority feature that contains options for allowing the user to configure the application.

4.7.2 Stimulus/Response Sequences

The user is presented with buttons that allow for configuration of the application. When the user taps on a settings item, the interface will toggle features accordingly.

4.7.3 Functional Requirements

REQ-1: The user must activate the settings button for it to be launched and display a list of options.
REQ-2: User toggles reverse lookup to enable or disable the feature.
REQ-3: User toggles theme option to change colors between dark and light mode.
REQ-4: User activates help option to display images.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Application must be fully initialized within half a second on any device running Android operating system 5.0 (Lollipop) or newer, despite any hardware specifications of that device.
History feature must hold up to 25 unknown numbers. Unknown numbers are any numbers that the reverse look-up feature did not/could not find.
Reverse lookup feature must be completed within 30 seconds after a phone call.
Background processes must be limited to prevent excessive battery drainage.
The application must be stable and not crash.

5.2 Safety Requirements

The main safety concern is providing correct information to users to ensure they are not calling someone that they did not intend to call, such as if the application provides the incorrect information for when a user calls a company. The White Pages API allows us to ensure that the information Smart Dialer provides is as accurate as possible.

5.3 Security Requirements

The security will be dependent on the security of the mobile device (Android) that the application resides on.
Google provides its own security for users.
White Pages API also provides its own security for users.

5.4 Software Quality Attributes

Usability - the application must work completely as intended with all of the features that we have outlined. It also must have a good user interface that is simple for users to understand.
Reliability - the application must return factual information for each number called, so that the user can trust the application works correctly.
Interoperability - the application must interface with the Smart Pages API, as well as Android's caller application to return information and make calls.

5.5 Business Rules

Only the user can access his/her contact list.
Reverse lookup information is only supplied to the user when necessary (i.e. when the user inputs a number)

6. Other Requirements

The development team needs a *WhitePages Pro* account in order to use the API that allows Smart-Dialer access to the WhitePages database of phone numbers, contact names, emails, and locations.
The development team needs an Android developer account in order to publish the application on the Google Play Store.

Appendix A: Glossary

411 - The telephone number for local directory assistance in the United States and Canada. Also used as a shorthand to mean pertinent information relating to who or what it is applied to.
API – Application Program Interface. A set of routines, protocols, and tools for building software applications.
IEEE - Institute of Electrical and Electronics Engineers.
JSON - JavaScript Object Notation
Lollipop - Android's mobile operating system, version 5.
Marshmallow – Android's mobile operating system, version 6.
Mobile Platform – The hardware/software environment for a smartphone.
Nougat – Android's mobile operating system, version 7.
SDK – Software Development Kit.
SIM – Subscriber Identity Module.
SMS – Short Message Service. Basic text messaging functionality in mobile phones.