Software Requirements Specification

for

Iris: Email Reader Application

Version 1.06

Scott Arnette

Joseph LaCava

Derek Ouzia

Bryan Smith

University of Virginia’s College at Wise

Department of Mathematics and Computer Science

April 15th 2016

Table of Contents

Table of Contents ii

Revision History iii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 2

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 3

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 3

2.6 User Documentation 3

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 4

3.1 User Interfaces 4

3.2 Hardware Interfaces 4

3.3 Software Interfaces 4

3.4 Communications Interfaces 5

4. Functional Requirements 5

5. Other Nonfunctional Requirements: 6

5.1 Performance Requirements 6

5.2 Safety Requirements 6

5.3 Security Requirements 7

5.4 Software Quality Attributes 7

6. Other Requirements 7

Appendix A: Glossary 8

Appendix B: To Be Determined List 8

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Scott Arnette | 11/18/15 | Initial document creation | 1.0 |
| Derek Ouzia | 11/22/15 | Complete sections 3.3, 3.4, 5.3, and 5.4 | 1.01 |
| Bryan Smith  Scott Arnette | 12/2/15 | Completed initial document. | 1.02 |
| Bryan Smith | 12/6/15 | Update definitions | 1.03 |
| Bryan Smith | 1/26/16 | Add new requirements for features being added in second semester | 1.04 |
| Bryan Smith | 3/31/16 | Removing Multiple Voice and visual cue requirements | 1.05 |
| Bryan Smith | 4/15/16 | Update requirement 5.1.2 to make it clearer. | 1.06 |

# Introduction

## Purpose

The purpose of this Software Requirements Specification is to define the requirements for the Email Reader Application in its entirety. We shall refer to the project as Iris, or simply the application henceforth.

## Document Conventions

This document does not feature any fonts or highlighting that will signify any special circumstances. All high level requirements will be represented by additional detailed requirements, with priorities implemented when all known requirements have been listed and this document established as a baseline.

## Intended Audience and Reading Suggestions

This document is intended to be read by all stakeholders of this project, including developers, testers, and users. The remainder of this document provides information about the system being developed, such as its description, intended functionality, operational environment, and any interfaces it may use. Finally, this document will detail the requirements this system shall adhere to, such as functionality or any nonfunctional requirements that the system shall follow.

## Product Scope

Iris will provide users the ability to be read emails received to the Google Account they choose to use during application use on a device running the Android operating system (version 3.0 or newer). After being read an email, users will additionally be prompted whether to keep or delete the previously read message. Users will also be provided settings to change how often emails are synchronized to Iris, whether Iris should currently read messages to the user, and whether to keep the device’s screen on (not go into sleep or low-power mode) during application use.

## References

Android Support Libraries: <http://developer.android.com/tools/support-library/features.html>

# Overall Description

## Product Perspective

The Iris project is a new, self-contained project.

## Product Functions

Iris allows users to be read incoming email messages via synthesized voice and respond whether to delete or keep the read message:

* Iris uses the Gmail API for Android and reads emails sent to the user’s Google Account:
  + The user must allow permission for Iris to be installed and access email data when selecting their account.
  + At launch, only a valid Google Account can be used in Iris. Other accounts are not supported.
* Iris can be toggled whether or not to read incoming messages.
* The user is provided settings to customize how frequently new messages are checked for and whether to force the application to keep the device’s screen on.
* The user will be prompted to keep or delete messages as they are being read via synthesized electronic voice.

## User Classes and Characteristics

Due to the simplistic nature of the Iris, the user classes will not be distinct enough to have individual user classes. The users will all be performing the same actions and there will not be any security levels. This simply leaves the only user class as “User”.

## Operating Environment

Iris shall run on devices with Android 3.0 or newer as their operating system. The device will also have to support Google Play Services v8.3.

## Design and Implementation Constraints

The Iris is constrained by the operating system and devices it is created for. Iris is only compatible with devices run Android 3.0 or newer with Google Play Services. An internet connection is required for synchronization of emails that Iris reads and users respond to. Additionally, only one Google account is supported by the application currently.

## User Documentation

Due to the simplistic nature of Iris, no user documentation will be produced outside of in-application guidance.

## Assumptions and Dependencies

* Network speed is assumed to be at minimum close to average. (100kb/s)
* Assumed that the user will not use Iris as a complete Email client replacement.
* Application is dependent upon the user having a Android device that will support Iris
* Application is dependent upon the user having a Gmail (Google) account.

# External Interface Requirements

## User Interfaces

Iris is used on the Android Operating System. Users interact with the application through its graphical user interface and touch inputs. At launch, Iris will provide the user with commands to set up their account for use with the application. The application will provide a toggle option for users to halt/resume synthesized voice reading of incoming email messages. The application will provide the user with a settings menu allowing for customization of synchronization frequency and to toggle the device’s screen to remain at during application use. The application will allow users to provide an input to immediately perform a partial synchronization of messages.

## Hardware Interfaces

Iris will run on an Android device which shall have allow touch input or mouse/keyboard input.

## Software Interfaces

Iris will run on the Java Virtual Machine on the Android operating system. The application will use the built in SQLite services provided by the Android OS and store email message data in a local SQLite database. The application will use the external libraries listed below:

* Android Support v4
* Android Support v7
* Android Support Design v23
* Android Play Services Identity v8.3
* Google API Client for Android v1.2
* Google Gmail API v1.2
* Javax Mail v1.4.1

Messages outbound from the application are for requesting message data from the Gmail API and authenticating with Google. Messages inbound are the email message data requested.

## Communications Interfaces

Iris shall communicate with Google and Gmail servers via the respective libraries. These libraries handle the HTTP requests and responses that we request from them. The data is sent and received via HTTP using Gmail’s REST API. Security is handled server side by Google and in order to retrieve data from the API, the client’s (in this case Iris) fingerprint must be specified in the Google Developer Console. Iris’s fingerprint is the SHA1 hash of the signing key for the application.

# Functional Requirements

## The application shall require at least a single valid Google Account from the user for use by the application.

## The application shall provide a centralized list of emails upon application launch.

## The application shall provide user notifications in the form of reading the email via synthesized electronic voice.

## The application shall provide input options to the user while a message is being read.

### The application shall provide a Keep button to keep an email.

### The application shall provide a Delete button to delete an email.

## The application shall provide an input option for the user to pause and resume reading of email via synthesized electronic voice.

## The application shall provide an input option for the user to perform an immediate, manual synchronization of email messages.

## The application shall provide a settings screen accessible from the main screen.

### The application shall provide a setting to modify the frequency that messages are synchronized while the background voice service is *not* running.

### The application shall provide a setting to modify whether the device screen remains on during application use.

### The application shall provide an option to logout for each of the Google accounts that are signed into the application.

### The application shall provide a setting to modify the frequency that messages are synchronized while the background voice service is running.

## The application shall provide support for deleting a range of emails.

## The application shall provide support for archiving a range of emails.

## The application shall allow the user to sign into multiple Gmail accounts.

# Other Nonfunctional Requirements:

## Performance Requirements

### The application shall load within 4 seconds.

### The application shall retrieve email message data from the Gmail API within 3 minutes or less. (I.E Data from Gmail API should be transferred over the network to the device in 3 minutes or less.) This does not mean a sent message should be received within that time range as we have no control over the Gmail infrastructure to minimize delays from sending until the Gmail API receives the message.

### The application shall not exceed more than 250 Gmail API requests per second.

## Safety Requirements

### The application will never be allowed to read, write, or delete any files other than those that are created by the application.

### The application, regardless if user specified, should never interact with files belonging to the operating system.

### The application shall not modify, send, or delete, any email messages of the signed in Google Accounts unless directly specified by the user.

## Security Requirements

### The application shall only access the user’s Google Account Gmail data after the user explicitly grants the application permission.

### The application shall never store the user’s Account passwords.

### The application shall only receive data from the Gmail API if signed with the key store file provided with this project.

## Software Quality Attributes

### The application shall provide a simple, intuitive and familiar GUI.

### The application shall be easily portable to new versions of the Android OS. Often with no changes needed.

### The application is stand-alone and will be non-interoperable with other systems other than the OS.

# Other Requirements

* The application shall be developed in the Java Programming Language.
* The application shall run on Android 3.0 (API 11) or higher.

Appendix A: Glossary

* ADB – Android Debug Bridge. Versatile command line tool that lets you communicate with an emulator instance or connected Android-powered device
* APK - Android application package is the package file format used by the Android operating system for distribution and installation of mobile apps and middleware.
* API – Application Programming Interface. A set of routines, protocols, and tools that govern a software specification.
* GUI – Graphical User Interface. An interface that allows users to interact with electronic devices through icons and visual indicators.
* Iris – The name of the Email Reader Application. The project that this SRS is for.
* OS – Operating System. The software that the product runs on.
* REST API - Representational state transfer application programming interface. A light weight web based API. The client does not need to know the structure of the API but the server provides the information the client needs to interface with the service.
* SDD – Software Design Document
* SRS – Software Requirements Specification.
* STP – Software Test Plan

Appendix B: To Be Determined List

1. NONE