Software Requirements Specification

Secure Messaging App

Project 2 (Updated)

James Bunting

Jonathon Herrera

Jorge Gasca-Cervantes

Justin De Luna

Ricardo Lesmes

May 2, 2017

Table of Contents

[List of Figures 3](#_Toc481529566)

[List of Tables 4](#_Toc481529567)

[SRS 5](#_Toc481529568)

[Section 1: Introduction Summary 5](#_Toc481529569)

[1.1 Purpose of this SRS 5](#_Toc481529570)

[1.2 Scope 5](#_Toc481529571)

[1.3 Overview 5](#_Toc481529572)

[1.4 Architectural Design 5](#_Toc481529573)

[Section 2 7](#_Toc481529574)

[2.1 Use Case Diagram 7](#_Toc481529575)

[2.2 Brief Description of Use Case Diagram 8](#_Toc481529576)

[2.3 Use Case Tables and Descriptions 8](#_Toc481529577)

[Section 4 13](#_Toc481529578)

[4.1. UML Class Diagram 13](#_Toc481529579)

[4.2 UML Class Diagram Description 14](#_Toc481529580)

[Section 5 15](#_Toc481529581)

[5.1 Test Cases 15](#_Toc481529582)

# List of Figures

[Figure 1: 2.1.1 Use Case Diagram 7](#_Toc481529676)

[Figure 2: UML Diagram 13](file:///C:\Users\justi_000\Downloads\NFTFinalSRS.docx#_Toc481529677)

# List of Tables

[Table 1: UC-0001 8](#_Toc481529687)

[Table 2: UC-0002 9](#_Toc481529688)

[Table 3: UC-0003 10](#_Toc481529689)

[Table 4: UC-0004 10](#_Toc481529690)

[Table 5: UC-0005 11](#_Toc481529691)

[Table 6: UC-0006 11](#_Toc481529692)

[Table 7: UC-0007 12](#_Toc481529693)

[Table 8: UC-0008 12](#_Toc481529694)

[Table 9:Test Cases 15](#_Toc481529695)

# 

# SRS

https://github.com/jgascacervantes/Secure-Messenger-App

## Section 1: Introduction Summary

### 1.1 Purpose of this SRS

The purpose of this document is to explain in detail the requirements for the “Not for terrorists secure messaging app”. This document will also explain the purpose of the aforementioned system. In conclusion, this document will act as a proposal to customers as well as a reference for the development team.

### 1.2 Scope

The purpose of this application is to be able to send highly secure messages to those with the same app. Messages will be deleted at most 5 minutes after being read, and will automatically delete after 8 hours. Customers will be able to add encryption to their messages, as well as ten-point security patterns. The makers of this app will not have a way to view these messages, and will have minimal access to the back-end. Passwords will be only be kept on the developers servers, under standard encryption. Accounts can only be created by admins; usernames and passwords will be randomized and be at least 10 characters.

This application needs data-service or Wi-Fi to send messages.

### 1.3 Overview

The document consists of a UML case diagram of all identifiable use cases, with detailed descriptions of each. Furthermore, each use case will have a corresponding sequence diagram. The next section gives a detailed UML class diagram, showing all actors, relations, etc. A brief description will also be provided.

Next will be two state machines for use cases of interest. Functional and non-functional requirements will be documented. Use case stories will be next, followed lastly by the appendix, containing minutes from the stakeholder meetings.

1.4 Architectural Design

The architectural design of the prototype is based off the client-server and the object-oriented style. The object-oriented style was used to hold things like the users and messages, while the client-server style was used for sending and receiving messages. The flow of the program is as follows. The Application class is called on app startup and starts the AWSMobileClient and the PushListenerService. The LoginActivity is displayed on the screen, which takes inputs and compares them with the database on the server. Once login attempt is completed, the mainActivity is started. From the mainActivity, the Messenger class can be called, from which the messages can be created and sent to another device.

## Section 2

### 2.1 Use Case Diagram

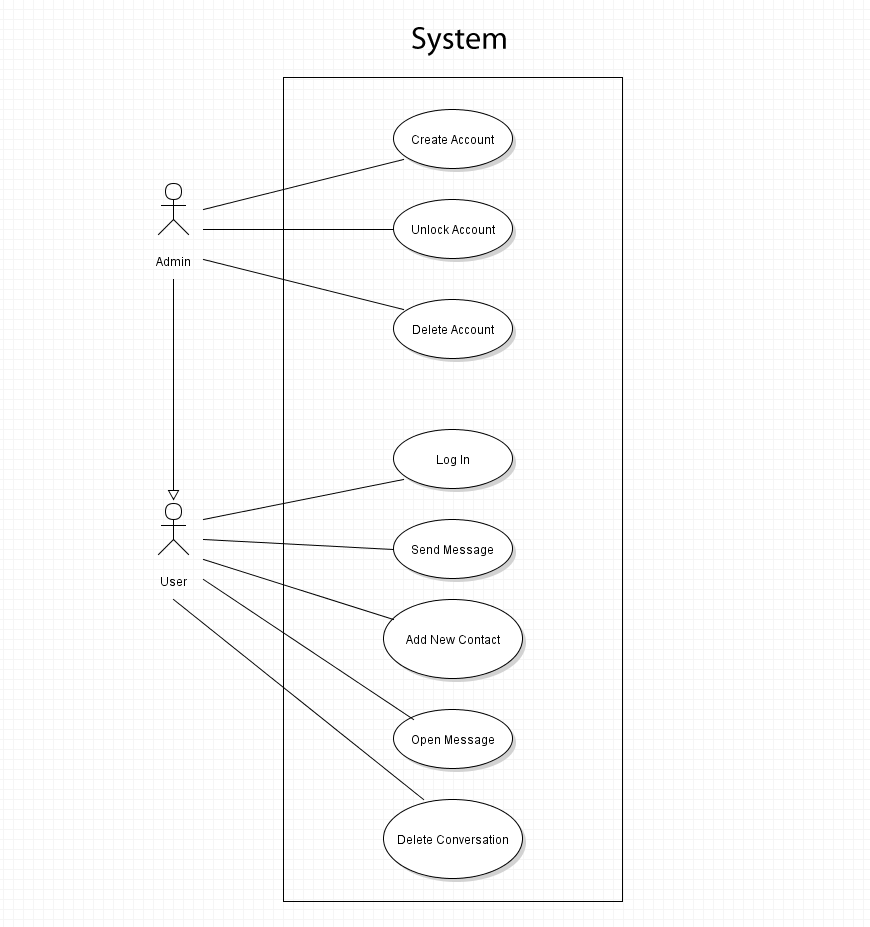


Figure 1: 2.1.1 Use Case Diagram

### 2.2 Brief Description of Use Case Diagram

The Use case diagram displays the 2 actors, the admin and the user and the cases they would be involved in. The list of use cases is as follows: Admins: UC-0001 Create Account; UC- 0002-Lock/Unlock Account; UC- 0003-Delete Account; All Users: UC-0004-Login; UC-0005-Send Message; UC- 0006-Add new Contact; UC- 0007-View Message; UC- 0008-Delete Conversation.

### 2.3 Use Case Tables and Descriptions

For Admins

|  |  |
| --- | --- |
| Use Case Name | Create Account |
| Use Case ID | UC-0001 |
| Description | Creating an account for another User |
| Actors | Admin, System |
| Pre-Conditions | Admin is logged into an Administration account, user who is receiving new profile has passed company security checks. |
| Post-Conditions | Admin has created an account |
| Normal Use Scenario | 1. Admin clicks on ‘Create Account’ button in the toolbar 2. System generates a random 10-15 digit number ID. 3. System checks to see if number matches an existing account\* 4. System generates a random password using numbers letters and special characters 5. System checks if password matches an existing account\* 6. System returns ID and password to Admin. 7. System asks Admin to confirm creation. 8. Admin confirms 9. System creates account with generated ID and password |
| Alternative Use Cases | 3a. System finds a match  3b. System generates new ID  5a. System finds a match  5b. System generates new password |

Table 1: UC-0001

The admin will create an account for a user. They will do so by being logged in to their account and going to the create account button.

|  |  |
| --- | --- |
| Use Case Name | Lock/Unlock Account |
| Use Case ID | UC-0002 |
| Description | Admin locks/unlocks account |
| Actors | System, Admin |
| Pre-Conditions | Admin is logged in and has a list of user accounts. |
| Post-Conditions | Account is locked/unlocked. |
| Normal Use Scenario | 1. Admin opens app 2. Admin types in ID and password 3. System checks account data for a match\* 4. App unlocks and shows received messages 5. Admin navigates to users page. 6. Admin selects user to edit. 7. Admin selects lock/unlock account. |
| Alternative Use Cases | N/A |

Table 2: UC-0002

The Admin locks/unlocks accounts by viewing them in his account. They go to the user data and edits the account accessibility field.

|  |  |
| --- | --- |
| Use Case Name | Delete Account |
| Use Case ID | UC-0003 |
| Description | Admin deletes an account |
| Actors | System, Admin |
| Pre-Conditions | User has an account; Admin has list of user accounts. |
| Post-Conditions | Account is deleted |
| Normal Use Scenario | 1. Admin opens app 2. Admin types in ID and password 3. System checks account data for a match\* 4. App unlocks 5. Admin navigates to users page 6. Admin selects user to edit. 7. Admin selects delete 8. Admin confirms deletion of account |
| Alternative Use Cases | 3a. User incorrectly enters data  3b User incorrectly enters data 3 times  3c. System locks the app  3d. User must contact Admin outside of app to unlock |

Table 3: UC-0003

While logged in, the admin views the profile of a user. The admin then goes to the delete account option and deletes it.

User Use Cases

\*These cases are also usable by Admins

|  |  |
| --- | --- |
| Use Case Name | Login |
| Use Case ID | UC-0004 |
| Description | User logs into their account |
| Actors | User, System, Admin |
| Pre-Conditions | User has an account; User has downloaded the app onto their mobile device |
| Post-Conditions | User is logged into account |
| Normal Use Scenario | 1. User opens app 2. User types in ID and password 3. System checks account data for a match\* 4. App unlocks and shows received messages |
| Alternative Use Cases | 3a. User incorrectly enters data  3b User incorrectly enters data 3 times  3c. System locks the app  3d. User must contact Admin outside of app to unlock |

Table 4: UC-0004

A user with an account opens the app. The first screen visible is the login screen. The user enters their name and password.

|  |  |
| --- | --- |
| Use Case Name | Send Message |
| Use Case ID | UC-0005 |
| Description | User sends a message to another User |
| Actors | User, System |
| Pre-Conditions | Both Users have accounts and are logged in. Both Users must have an agreed upon encryption code and or security pattern |
| Post-Conditions | A message is sent to another user |
| Normal Use Scenario | 1. User types in recipient ID or opens chat with desired recipient 2. User types a message into the message bar 3. User chooses time after message is read that it should be deleted. 4. User chooses whether to add encryption 5. User chooses whether to add pattern 6. User hits send button 7. System sends message to recipient ID |
| Alternative Use Cases | N/A |

Table 5: UC-0005

The user selects the send message button. The button opens up the message creation dialog. The user enters the message and desired encryption and sends the message.

|  |  |
| --- | --- |
| Use Case Name | Add new contact |
| Use Case ID | UC-0006 |
| Description | User adds a new contact |
| Actors | User, System |
| Pre-Conditions | User is logged in and on contact screen |
| Post-Conditions | New Contact is Added |
| Normal Use Scenario | 1. User selects add contact 2. User enters contact data 3. User confirms contact 4. User exits created contact |
| Alternative Use Cases | 3a. User cancels contact creation |

Table 6: UC-0006

The user goes to the contact list and selects the add contact button. The user enters the data and selects create.

|  |  |
| --- | --- |
| Use Case Name | View message |
| Use Case ID | UC-0007 |
| Description | User views message |
| Actors | User, System |
| Pre-Conditions | User is on contacts screen |
| Post-Conditions | User can view message |
| Normal Use Scenario | 1. User selects a contact with an unviewed message 2. User enters encryption key 3. User views message 4. User deletes message 5. User returns to contacts screen |
| Alternative Use Cases | 2a. User enters pattern  2b. User enters pattern and key  3a. Message is deleted due to user failing to enter the correct code  4a. User exits without deleting message |

Table 7: UC-0007

The user receives a message and is notified. The user opens the app and selects the field where the message is then viewed.

|  |  |
| --- | --- |
| Use Case Name | Delete conversation |
| Use Case ID | UC-0008 |
| Description | User deletes conversation |
| Actors | User, System |
| Pre-Conditions | User is on contacts and has had a conversation |
| Post-Conditions | Conversation is deleted |
| Normal Use Scenario | 1. User selects a contact 2. User presses delete conversation |
| Alternative Use Cases | N/A |

Table 8: UC-0008

The user selects a conversation. The user hits the delete conversation button.

## 

## Section 4

### C:\Users\justi_000\Documents\Justins Docs\School\Spring2017\SoftwareEngineering\SecureMessengerApp\SEP2ClassDiagram.jpg4.1. UML Class Diagram

Figure :UML class diagram

Figure 1: 4.1.1 UML Class Diagram

### 4.2 UML Class Diagram Description

The UML Diagram contains 9 classes.

* PushListenerService: Extends GcmListenerService. Listens for push notifications. Functions for receiving messages and decrypting.
* PatterntoShow: Extends AppCompatActivity. Sets the content layout for a pattern.
* MyInboxRecyclerViewAdapter: Extends RecyclerView. Handles views for the inbox.
* ViewHolder: Extends RecyclerView. Is the view class for the other classes.
* Messenger: Extends AppCompatActivity. The class that handles messaging, including sending, creating a pattern, and encrypting.
* MainActivity: Extends AppCompatActivity. The class that holds displays the conacts list and the buttons on that screen. The user can navigate to
* LoginActivity: Extends AppCompatActivity. The class that displays the login screen on app startup. It displays an email field, password field, and login button.
* InboxFragment: Extends Fragment. Handles the creation of the inbox fragment. Creates the view for the main activity.
* Application: Extends MultiDexApplication. The class that is called on creation of the app. It calls initializeApplication(), which then creates the necessary server connections.

## 

## Section 5

### 5.1 Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Scenario | Test Steps | Test Data | Expected Results |
| TC01 | Check for valid Login | 1. Open app  2. Enter User Id  3. Enter Password  4. Tap Sign in or register | Userid= Admin  Password = Password | User should login to application |
| TC02 | Access Contact Info | 1. Be logged in  2. Tap a contact field  3. Edit info | Contact info | User should be able to access contact info |
| TC03 | Send a Message | 1. Be logged in  2. Select a contact or click send a message  3. Type a contact number and message  4. Choose encryption type  5. Click send | contact info | User should send a message to recipient |
| TC04 | View a message | 1. Be logged in  2. Click on a contact with a message to view  3. enter encryption key to view message | Contact info  Encryption key | User should be able to view sent message |

Table :Test Cases