

Safe-Key

Team 4

Ruth Jimenez, Carter Rath, Andrew Le, Jerstine Medrano

Software Design Specification Document

Version: (1)

Date: (12/15/2021)

Table of Contents

1 Introduction	4
1.1 Goals and objectives	4
1.2 Statement of system scope	4
1.3 Reference Material	6
1.4 Definitions and Acronyms.....	6
2 Architectural designs	6
2.1 System Architecture	6
2.2 Design Rational	7
3 Key Functionality design	7
3.1 Safe-Key Log-in.....	7
3.1.1 Safe-Key Log-in Use Cases	7
3.1.2 Processing sequence for Safe-Key Log-in	8
3.1.3 Structural Design for Safe-Key Log-in	9
3.1.4 Key Activities.....	10
3.1.5 Software Interface to other components.....	11
3.2 Safe-Key Account	11
3.2.1 Safe-Key Account Use Cases.....	11
3.2.2 Processing sequence for Safe-Key Account.....	13
3.2.3 Structural Design for Safe-Key Account.....	13
3.2.4 Key Activities.....	14
3.2.5 Software Interface to other components.....	14
3.3 Safe-Key Passwords	14
3.3.1 Safe-Key Passwords Use Cases	14
3.3.2 Processing sequence for Safe-Key Passwords	16
3.3.3 Structural Design for Dealer Census	17
3.3.4 Key Activities.....	17
3.3.5 Software Interface to other components.....	18
4 User interface design	18
4.1 Interface design rules	18
4.2 Description of the user interface.....	18
4.2.1 Login Page.....	18
4.2.2 Account Creation Page	19

4.2.3 Safe-Key Navigation Page	20
5 Restrictions, limitations, and constraints	21
6 Testing Issues	22
6.1 Types of tests	22
6.2 List of Test Cases	22
7 Appendices.....	23
7.1 Packaging and installation issues.....	23
7.2 User Manual.....	24
7.3 Open Issues.....	24
7.4 Lessons Learned	24
7.4.1 Design Patterns.....	24
7.4.3 Team Communications	24
7.4.4 Task Allocations	25
7.4.5 Desirable Changes.....	25
7.4.6 Challenges Faced	26

1 Introduction

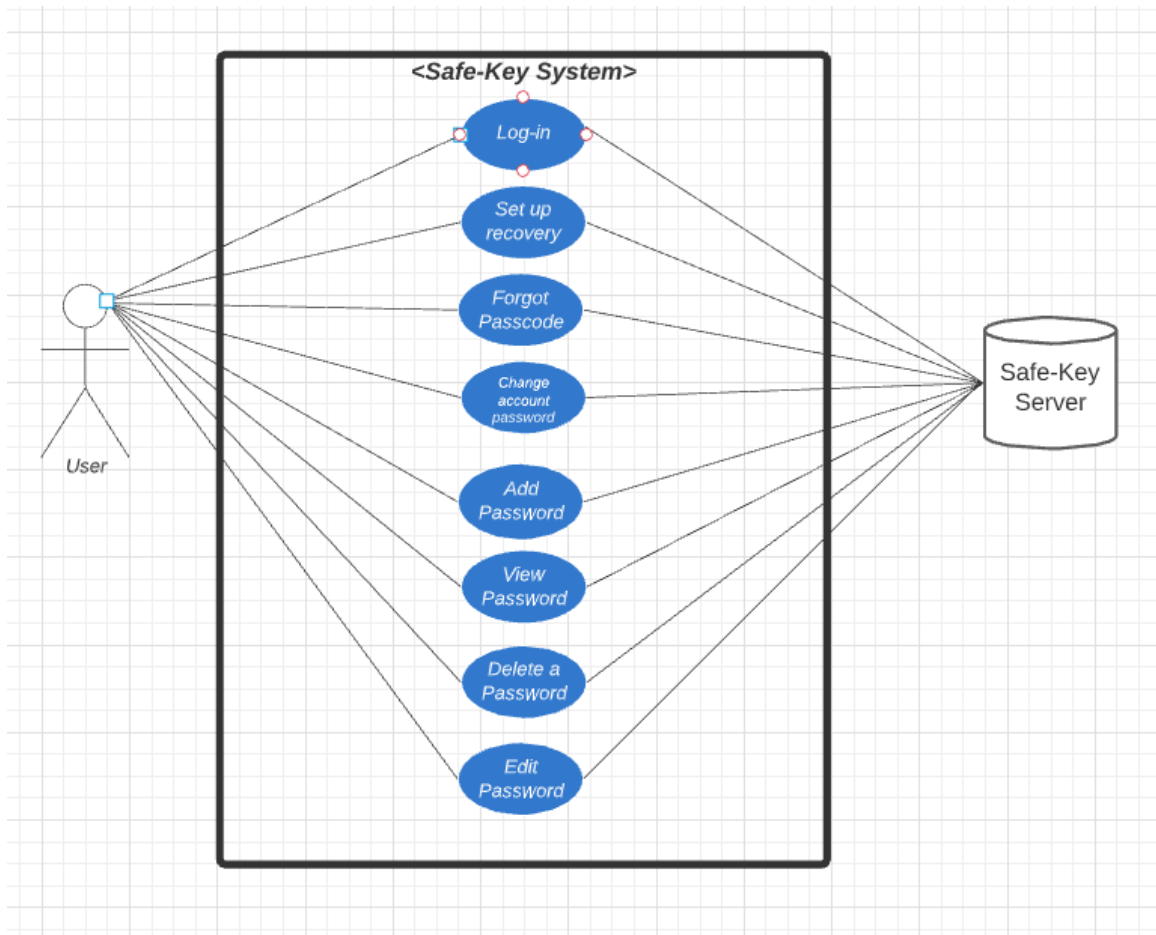
Safe-Key is an application that provides a secure and convenient hub to store any number of passwords that the user desires. This document describes all data, architectural, interface and component-level design for the software.

1.1 Goals and objectives

The objective of Safe-Key is to provide organizational methods of storing specific passwords that pertain to social media or even more important passwords, that way the user will not lose track of which password accesses what service they want to use. The goal is to create the utmost password storage security to ensure the privacy and security of potential users.

1.2 Statement of system scope

The purpose of Safe-Key's software is to provide a secure and convenient hub take the user's passwords, encrypting them, and storing them in the software's password hub. The benefits of having an encrypted password storage are the dependable method of storing the user's passwords in a hub that only the user can access with their own specialized key or pin, which adds another layer of security with our software.



1.3 Reference Material

Lecture Slides:

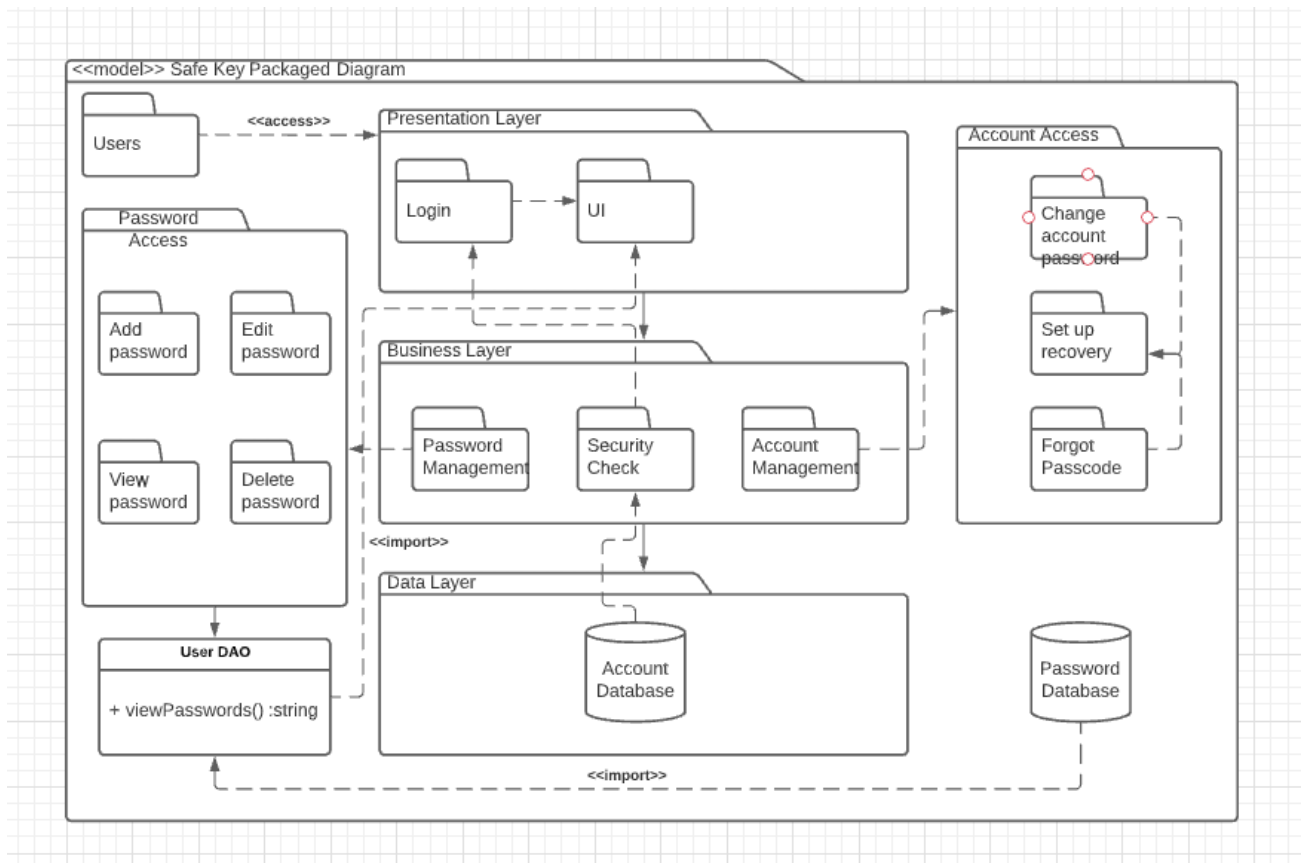
- M7-1
- M7-2
- M8-1
- M8-2
- M9-1
- M9-3
- M10-1

1.4 Definitions and Acronyms

Acronyms	Definition
DB	Data Base
DAO	Data Access Object

2 Architectural designs

2.1 System Architecture



First, the user accesses the presentation layer which presents a UI that displays Login details. After successful login with the security check references the Account Database on the website itself. The business layer is entered which presents account and password management options. For password, there's adding, editing, deleting, and viewing the stored passwords from the database. This is all passed through a User DAO that is then pushed up to the UI. For the account, there are change passwords, forget passcode, and set up recovery. None of them can be accessed until set up recovery is done.

2.2 Design Rational

One of the biggest concerns with the system's architecture was where to store all the data. With a program handling so much data at one time while ensuring user security, it simply was not a possibility to simply put login information into a .txt file or a similar low-level storage option. This is why it was important that access to a database like Firebase or MongoDB is achieved to ensure user safety. Another aspect that was considered was what is displayed for users. It was important to separate both the Login UI and the Password UI as somebody with no initial access to the account should only be able to see the Login UI.

3 Key Functionality design

3.1 Safe-Key Log-in

3.1.1 Safe-Key Log-in Use Cases

The user will launch the application and the login page will be displayed. They will then enter their credentials. The system will authenticate the credentials. If the credentials are entered correctly, the system will display the homepage to the user's account.

Table 1: Safe-Key Log-in Authentication

Project Name	Safe-Key
Use Case ID:	UseCase-002
Use Case Name:	Log-in
User Goal:	A user will log in to Safe-Key
Scope:	The Safe-Key system
Level:	Primary Task
Primary Actor:	User
Precondition:	User has an account

Minimal Guarantee:	User is denied access to their account
Success Guarantee:	User can log-in to their account
Trigger:	The user clicks <<login>>
Success Scenario:	Action
	1. User launches application
	2. User enters username and password
	3. System authenticates user
	4. User is granted access to account
Extensions:	Branching action
2a, 3a, 4a, 5a	1a. System fails to authenticate user <ul style="list-style-type: none"> 1. System prints a warning message 2. User clicks <<forgot password>>

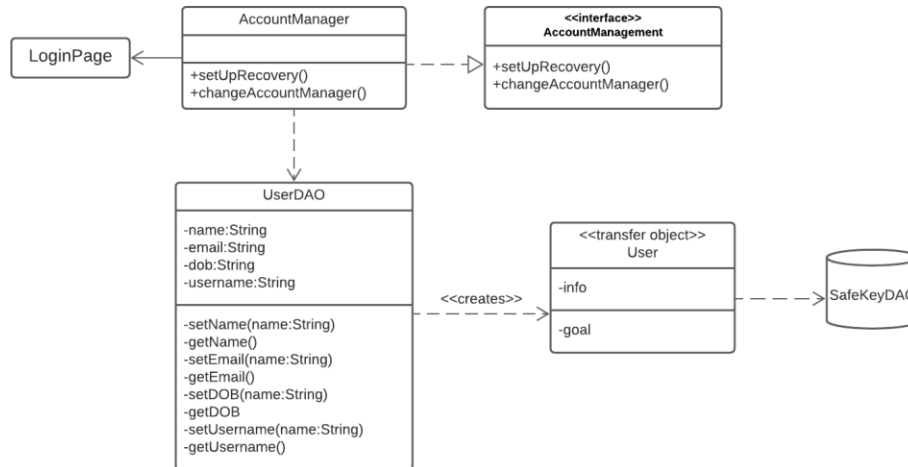
3.1.2 Processing sequence for Safe-Key Log-in

Below, are figures 1 and 2 for Safe-Key's sequence diagrams of Log-in and Forgot Password.

Figure 1: Sequence Diagram - Log-in Authentication



Page 9 of 27 12/15/21



3.1.4 Key Activities

Figure 4 displays the interactions and events related to log-in authentication for Safe-Key.

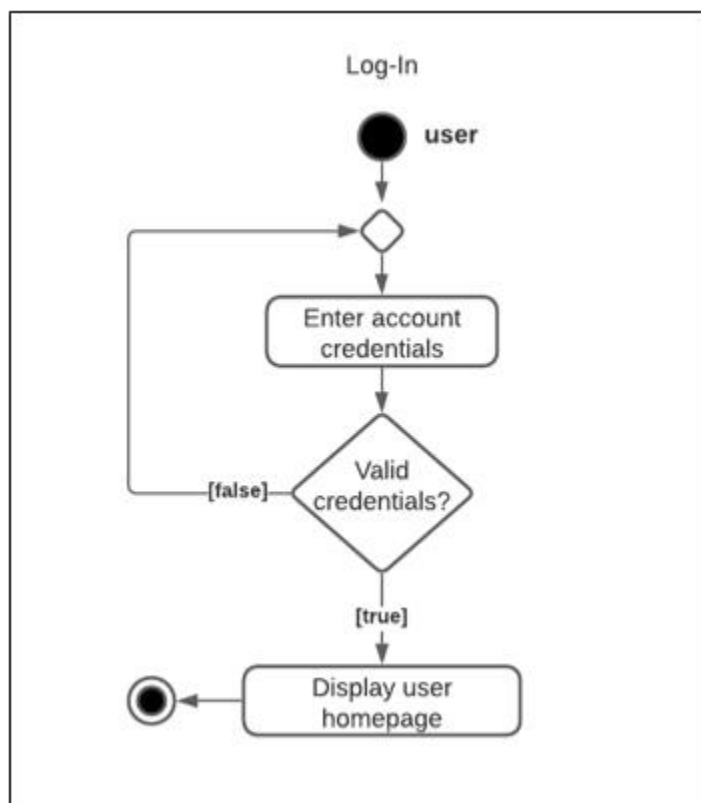
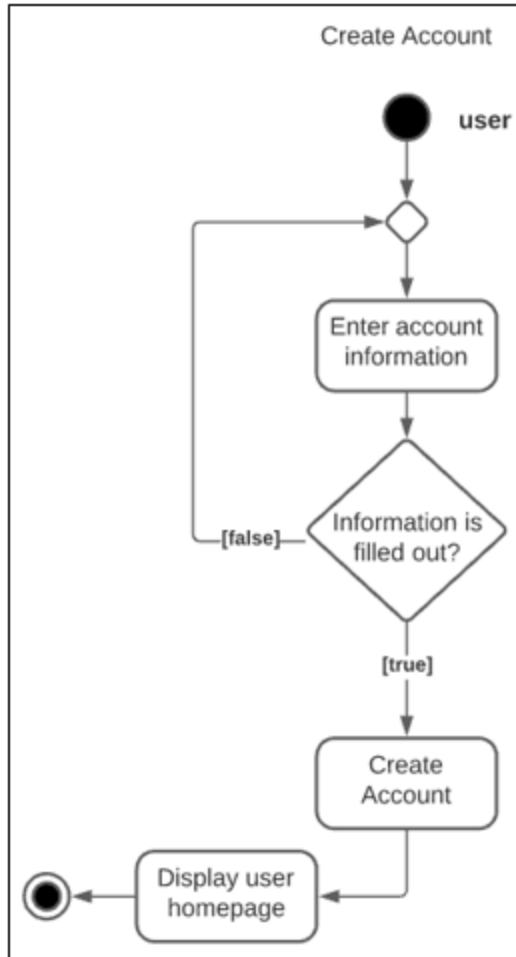


Figure 5 displays the create account feature



3.1.5 Software Interface to other components

The log-in interface uses Account Manager to authenticate the credentials of the user. It will get the account information from the DAO. The Account Manager will create an account object and start interface depending on if the user is logging in, signing up, or if they forgot their password.

3.2 Safe-Key Account

3.2.1 Safe-Key Account Use Cases

The user will set up a recovery for their account for an instance where they cannot remember their credentials to login. The system will prompt them to choose three questions to provide their own answers to. These questions should be ones that only they know the answers to. These questions will be presented to the user when they have failed attempts to login, and they resort to “forgot password.”

Table 2: Set up recovery

Project Name	Safe-Key
Use Case ID:	UseCase-004

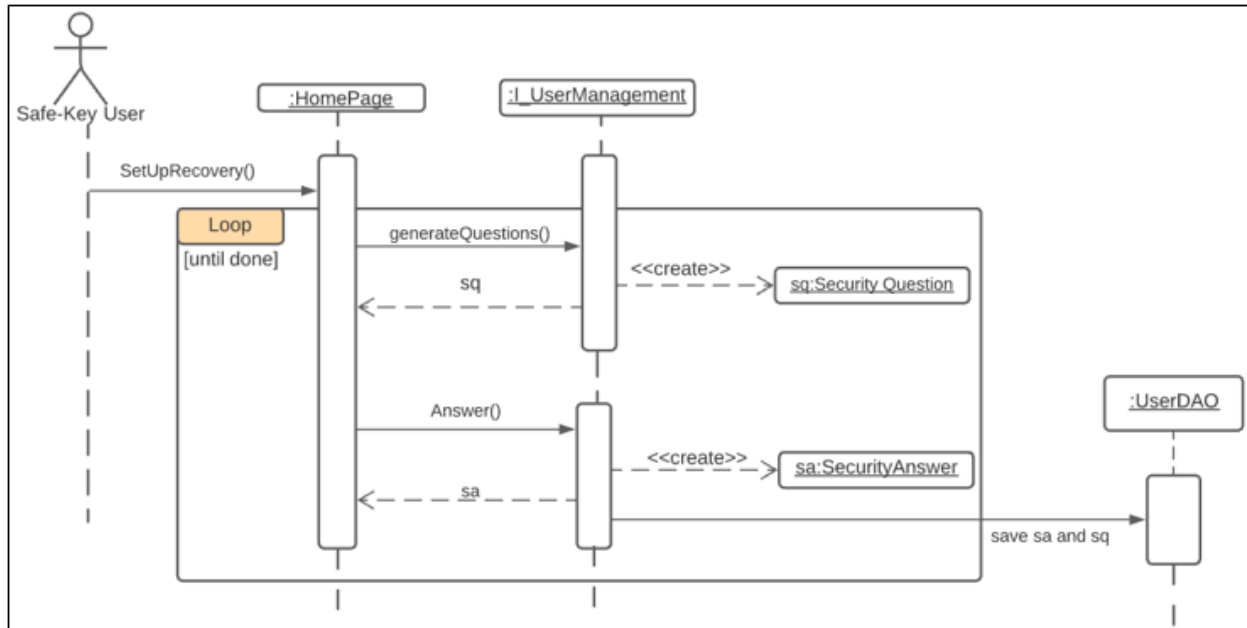
Use Case Name:	Set up recovery
User Goal:	User will create security questions
Scope:	The Safe-Key system
Level:	Primary Task
Primary Actor:	User accessing Safe-Key
Precondition:	User has created account and has access
Minimal Guarantee:	The security questions are not created
Success Guarantee:	The recovery setup is made
Trigger:	User clicks <<Set Up Recovery>>
Success Scenario:	Action
	1. User presses Account Settings
	2. User presses Set Up Recovery
	3. System provides user-related security questions
	4. User chooses 3 security questions
	5. User enters answers to security questions
	6. System saves user's security questions and answers
Extensions:	Branching action
2a, 3a, 4a, 5a	4a. User creates their own questions
	1. User types 3 security questions
	1. User enters answers to security questions

2. System saves user's security questions and answers

3.2.2 Processing sequence for Safe-Key Account

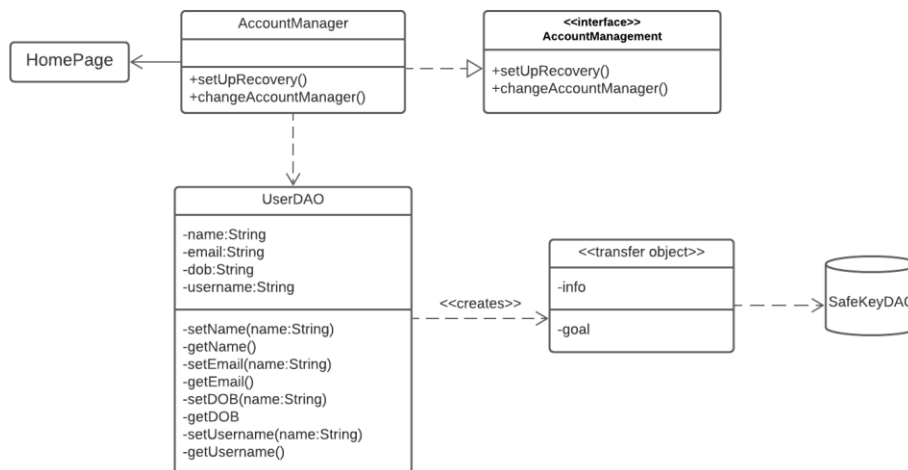
Below is figure 6, which is the sequence diagram for setting up the recovery for a Safe-Key account.

Figure 6: Sequence Diagram – Set Up Recovery



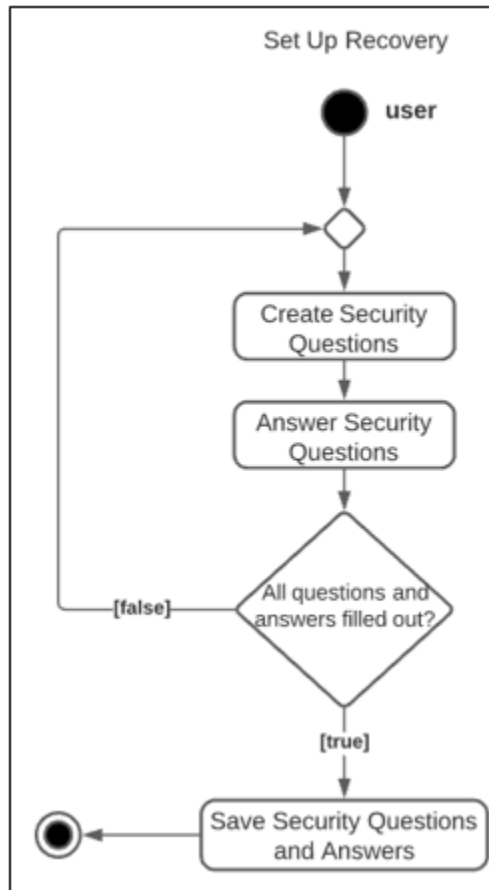
3.2.3 Structural Design for Safe-Key Account

Figure 7 displays the structural design of the Safe-Key account.



3.2.4 Key Activities

Figure 8 displays the interactions and events related to setting up the recovery for a Safe-Key account.



3.2.5 Software Interface to other components

The Safe-Key account uses Account Manager to allow user to set up a recovery for their account. Once the user selects security questions and answers, they will be saved to the DAO. The questions can be retrieved from the DAO whenever the user forgets their password to access the account. This feature also allows the user to change their password to access their account.

3.3 Safe-Key Passwords

3.3.1 Safe-Key Passwords Use Cases

The user will add a password to their account's password list. The user will enter the password for their account and click save. The system will encrypt the password and save it to the database, so the user can access it and view at any time. If the user forgets to press save before exiting, the application will display a warning message to the user.

Table 3: Add Password

Project Name	Safe-Key
---------------------	-----------------

Use Case ID:	UseCase-003
Use Case Name:	Add Password
User Goal:	Encrypt the user's passwords
Scope:	The Safe-Key system
Level:	Primary Task
Primary Actor:	User
Precondition:	User has passed verification and accessed account
Minimal Guarantee:	Password is not saved to account
Success Guarantee:	User's password is encrypted and stored
Trigger:	User clicks <<Add Password>>
Success Scenario:	Action
	1. User types and adds a password
	2. User clicks <<save>>
	3. System encrypts the user's password
	4. System stores the encrypted password
Extensions:	Branching action
2a, 3a, 4a, 5a	2a. User does not click <<save>>
	1. User clicks <<exit>>
	2. System prompts user to save
	3. User clicks <<save>>
	4. System encrypts the user's password

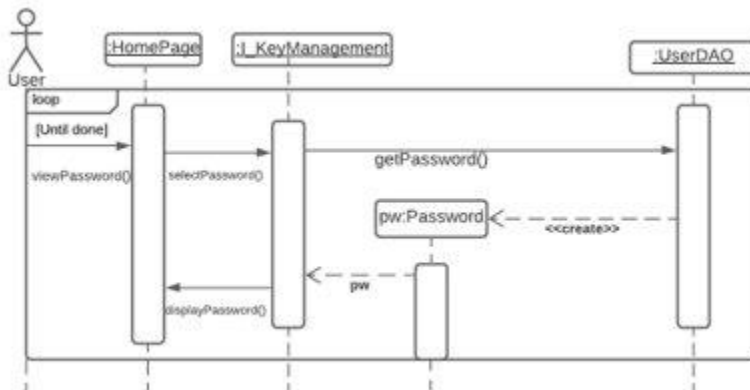
	5. System stores the encrypted password
--	---

Table 4: View Password

Project Name	Safe-Key
Use Case ID:	UseCase-006
Use Case Name:	View Password
User Goal:	User views a single password
Scope:	The Safe-Key system
Level:	Subfunction
Primary Actor:	User accessing Safe-Key
Precondition:	User has passed verification and accessed account
Minimal Guarantee:	System cannot find password
Success Guarantee:	User accesses password
Trigger:	User clicks on <<view password>>
Success Scenario:	Action
	1. User chooses which account to view password
	2. System gets user's encrypted password
	3. System decrypts password
	4. System displays password to screen
Extensions: 2a, 3a, 4a, 5a	Branching action
	System prints alert <<Password was not found.>>

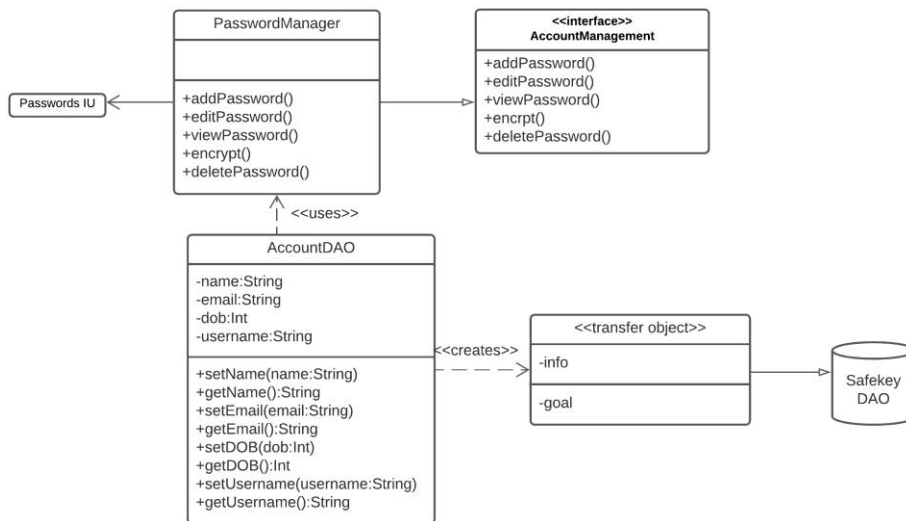
3.3.2 Processing sequence for Safe-Key Passwords

Figure 9 is the sequence diagram for viewing passwords on a Safe-Key account.

Figure 9: Sequence Diagram – View Passwords

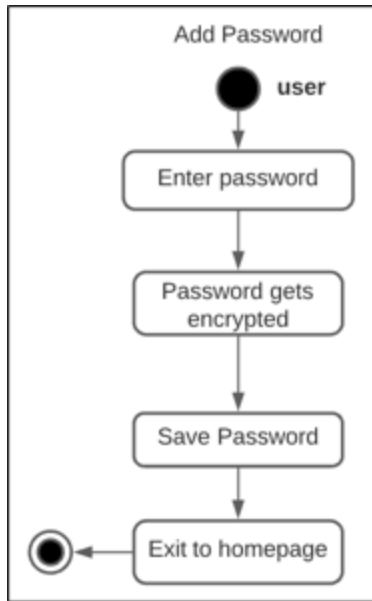
3.3.3 Structural Design for Dealer Census

Figure 10 displays the structural design for the passwords



3.3.4 Key Activities

Figure 11 displays the interactions and events related to adding a password for a Safe-Key account.



3.3.5 Software Interface to other components

The passwords feature of Safe-Key uses the Password Manager to add, edit, delete, and save passwords to the DAO. These passwords the user has created is accessible upon request. When the user requests to view a password, the server will get the password, from the DAO, decrypt it, and display it to the user.

4 User interface design

4.1 Interface design rules

The only standards we are currently following are to conform to a plain but minimalistic color scheme of standard whites, grays, and blacks.

4.2 Description of the user interface

The user interface is an application that provides a visual front-end to the user's stored passwords.

4.2.1 Login Page

The main menu of the application allows the user to create an account or log-in to their existing account. The login page contains fields for the user to fill in with their login information if they have an account with Safe-Key. If the user does not have an existing account with the Safe-Key service, there is a create account button below the login fields for new users.

4.2.1.1 Screen Images

Shown here is the Login interface screen of the Java application.

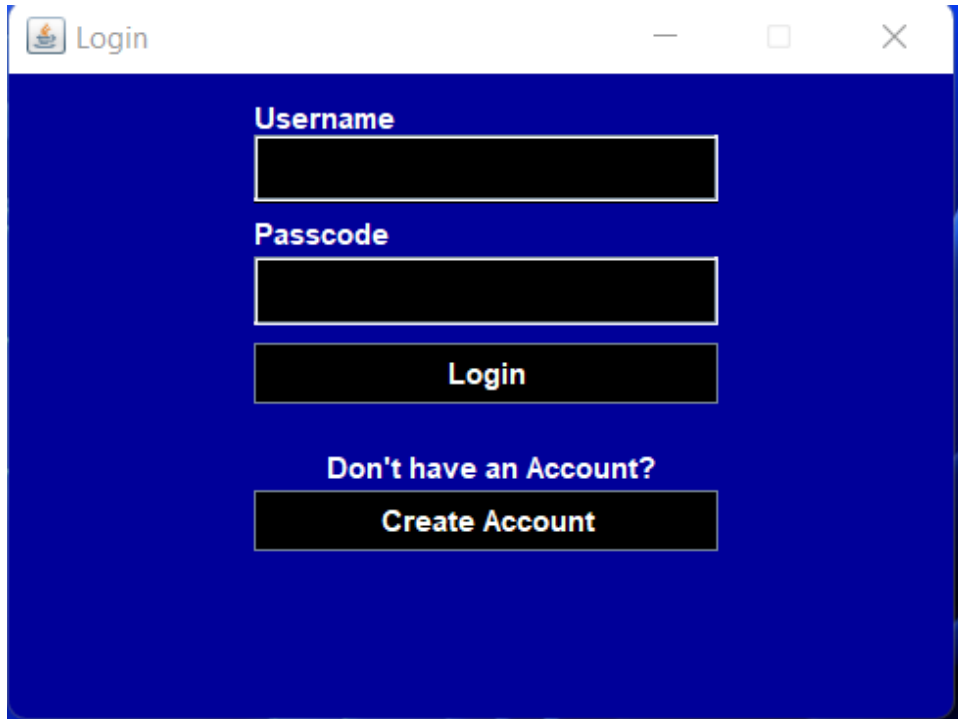


Figure 1 - Login

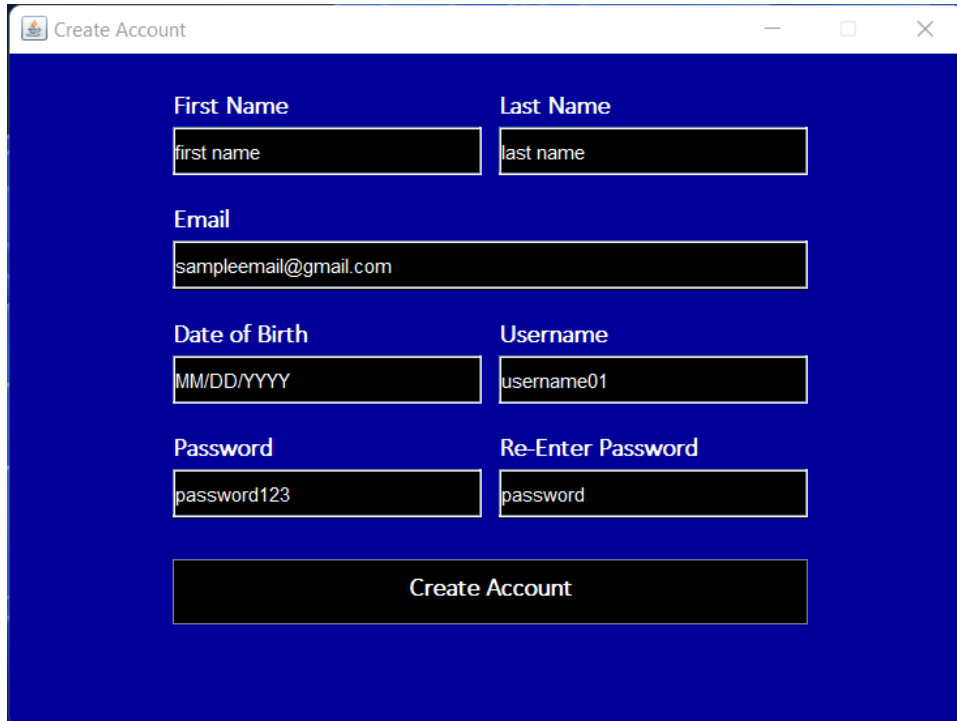
4.2.1.2 Objects and Actions

The user begins with the login screen where they are presented with a username and passcode fields for an existing account. If the user does not have an account with the Safe-Key system, the user has the option to create an account with the button below the login fields.

4.2.2 Account Creation Page

The account creation page has 7 fields that the user needs to input information for. With the account creation page, the user needs to input basic information such as their first and last name, email, and their date of birth. The user will also be prompted to create a username and password for their Safe-Key account.

4.2.2.1 Screen Images



The screenshot shows a window titled "Create Account" with a blue background. It contains several input fields and a button. The fields are arranged in a grid-like fashion. The "First Name" field contains "first name" and the "Last Name" field contains "last name". The "Email" field contains "sampleemail@gmail.com". The "Date of Birth" field contains "MM/DD/YYYY" and the "Username" field contains "username01". The "Password" field contains "password123" and the "Re-Enter Password" field contains "password". At the bottom, there is a large "Create Account" button.

Figure 2 – Account Creation Page

4.2.2.2 Objects and Actions

After the user inputs their needed information in the account creation fields, they can press the “Create Account” button to confirm the creation of their new Safe-Key account.

4.2.3 Safe-Key Navigation Page

The Safe-Key navigation page contains all of the information that the new or returning user would like access to. It has information based on the user’s account management and password management. The user can look through the navigation options and set up their account recovery options under account management, or add a password under password management.

4.2.3.1 Screen Images

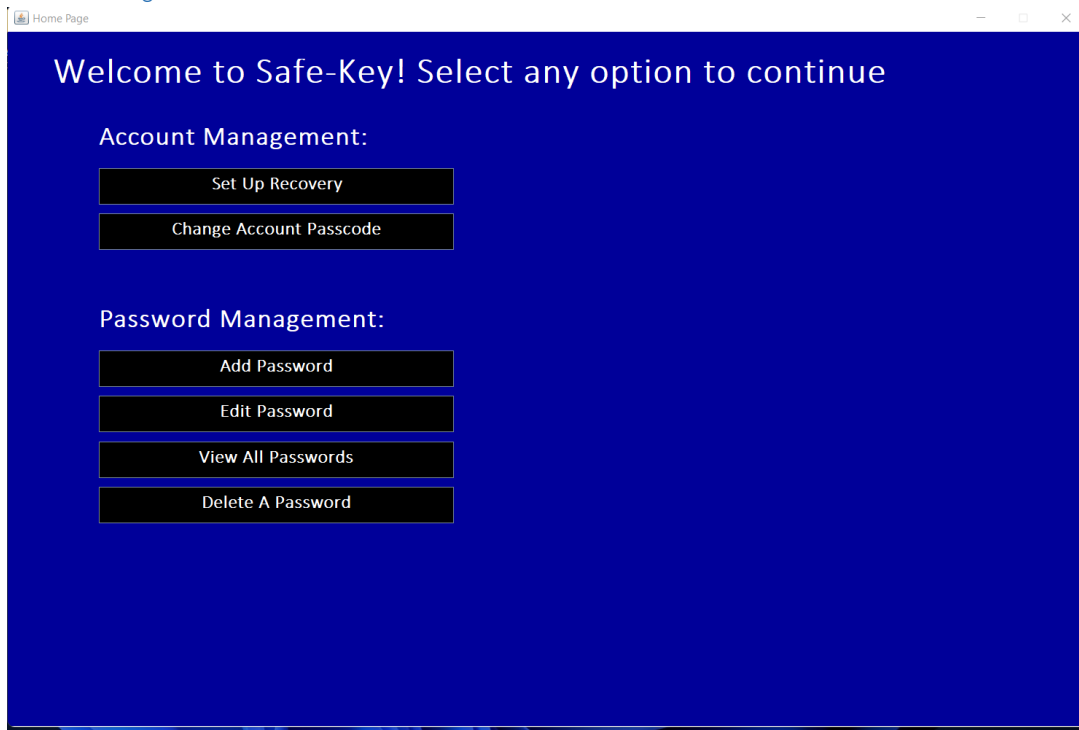


Figure 2 – Safe-Key Navigation Page

4.2.3.2 Objects and Actions

- The Set-Up Recovery button will lead the user to the process of setting up recovery options for their Safe-Key account.
- The Change Account Passcode button allows the user to change their access code to their account.
- Under password management, the add password button allows the user to create a new password to store in the password library.
- The edit password button allows the user to edit an existing password in their password library.
- The view all passwords button allows the user to view the stored passwords in the library.
- The Delete a Password button allows the user to delete an existing password they have in their password library if it is no longer needed or in use.

5 Restrictions, limitations, and constraints

- No HTML language can be used.
- Tool must operate in Microsoft or macOS environment.
- Only have access to free encryption software

- SE 370 knowledge and recourses

6 Testing Issues

Test strategy and preliminary test case specification are presented in this section. Additional test cases are located in the Appendix section in spreadsheet format.

6.1 Types of tests

You may consider the following types of tests:

- (1). **Performance Test** – When searching for a password or getting a password from the data base, It should take no more than a few seconds to return the password. A similar test is made for encryption and decryption.
- (2). **Accuracy Test** – When we press any buttons, the correct display is added or removed. We also tested the accuracy with login and recovery.
- (3). **User Interface Test** – The interface is very simple and straightforward. Any unfamiliar user can use the interface with the instruction provided in the login page and homepage.
- (4). **Security test** – To ensure that users can only perform the tasks specified for their account they must first login. Additionally, the view any added passwords they must be decrypted and are encrypted upon saving.
- (5). **Repeatability Test** – The software returns the same result when repeating the same action.

6.2 List of Test Cases

You should document each test case in the following format:

Test Type	Accuracy Test
Testing range	User login feature
Testing Input	User ID and Wrong password
Testing procedure	Enter user ID and wrong password Click Login
Expected Test Result	Prompt "User is not valid"
Testers	Tom Bridger
Test result	Passed

Note: you may have more than one test cases for each type of tests. If the tester and test results information is not available, you may add it in the final submission.

7 Appendices

7.1 Packaging and installation issues

How to install and prepare the system to run

For example, many teams use a database for your system. Some teams use FireBase, some use Google services, others use customized servers (located in dorm). Whatever techniques you are using, you need to describe how you set up the DB and how to create DB connections (showing some code snippet would help).

This part will be used by the instructor to evaluate your self-learning ability. When you describe a technique (even it is a small tool like Postman), provide sufficient information such that it can be used as a tutorial for a novice to quick get it up and running.

7.2 User Manual

1. If you are a new user:
 - a. Press create account
 - b. Fill in all the necessary information to create the account.
 - c. Set up a recovery by creating personalized questions and answers
 - d. Change your account password by entering your old password and entering your new password.
2. If you are a returning user:
 - a. Login:
 - i. Add a password by pressing “Add Password”
 - ii. Remove a password by pressing “Delete Password”
 - iii. Edit a password by pressing “Edit Password”
 - iv. View all password by pressing “View All Passwords”
 - b. Forgot Password:
 - i. Answer security questions. If the questions are correct, you will have access to your account.

7.3 Open Issues

Having a search bar for easy password navigation.

7.4 Lessons Learned

Very instructions for project before starting, this will avoid assignments being done wrong due to misunderstanding.

7.4.1 Design Patterns

Creational Design Patterns. We used this design pattern in order to implement and associate classes.

7.4.3 Team Communications

I believe our team communication will definitely improve in the future. We learned how to manage our time and how to support each other through our assignments.

7.4.4 Task Allocations

We allocate tasks but by assigning different tasks to each group member depending on our strengths in course material. We also believed it was a great learning opportunity for leadership from one student to another regarding each assignment.

7.4.5 Desirable Changes

Assume that you have another month to work on the project, what aspects of the system you would like to improve? What are the additional features you want to add to the system? [Each student should use a separate paragraph to respond to the questions]

Ruth:

If we had another month, I would definitely like to add more features to help with the interface. While the application seems very simple, maybe adding a few more instructions would be helpful for people who have difficulties understanding web applications.

Jerstine:

If we had another month to work on the project, the aspects of the system that I would like to improve are the add, edit, and delete password features. The additional feature I would have wanted to add to the system is being able to view all the passwords while adding, editing, or deleting passwords just so the user can see if they're adding a duplicate password on accident or so they can see which password they are specifically deleting without having to navigate back to the view all passwords button.

Carter:

If we had another month to work on the project, I would want to add a tutorial that displays to the user after the account is first created. We indicated our key users would be those with memory issues due to old age, disabilities, etc. and they may not be too comfortable with using this application. An additional feature I would have liked to add is to connect the accounts to their email, so the forgot password feature could send a link to change their password to their email after security questions were answered correctly. This would be an additional security measure to ensure that the person trying to access the account is really the owner of the account.

Andrew:

If we had another month, I would want to make the program more secure. With the amount of time there was this semester, I found it quite difficult to learn Java in a small amount of time and also learn implementation of password security and hashing. I think features specifically would be the relationship of user_ids with password and making it more like a regular hash table data structure instead of having access to all passwords freely.

7.4.6 Challenges Faced

Among requirements specification, system design, and system implementation, which one you think is the hardest task? Why? [Each student should use a separate paragraph to respond to the questions]

Ruth:

In my opinion, system design was the hardest to complete. There are many complex ways to create a system design more seem simpler than others but in the end it's going with what works best for the application.

Jerstine:

I think the requirements specification was the hardest task because in addition to creating the system, the documentation had to stay consistent with updates and changes to the system. The documentation of the system was difficult to get correct at first but with valuable feedback on our labs and during class time, we utilized that to our advantage to deliver the best we could for our application.

Carter:

For me, system design was the hardest task to complete because this was a brand-new concept. Even though there are many ways to proceed with a system design, there are also many rules. This resulted in many mistakes and updates, but my group remained attentive and responsive to feedback.

Andrew:

The biggest challenge for myself was allocating time to learn and also implement the code. With password management and security being a new concept for myself, it took a while for me to understand hashing and also how to work with a database. The database implementation took me

a while as at first, I was working with MongoDB, then Google Firebase, and then finally MySQL. Overall though, I've learned a lot from this project.