# 

# Password Safe With WebDAV Sync

by

Jake Schuenke

Matt Spurr

Tyler Shelton

Kegan Kaiser

Contents

[Password Safe With WebDAV Sync 1](#_Toc339973947)

[1 Executive Summary 4](#_Toc339973948)

[2 Introduction 5](#_Toc339973949)

[3 Client Background 6](#_Toc339973950)

[4 Current System 7](#_Toc339973951)

[5 User and Stakeholder Description 8](#_Toc339973952)

[5.1 Nontechnical user profile and environment 8](#_Toc339973953)

[5.2 Technical user profile and environment 8](#_Toc339973954)

[5.3 Open source developer profile and environment 8](#_Toc339973955)

[6 User Needs 9](#_Toc339973956)

[7 Alternatives and Competition 10](#_Toc339973957)

[7.1 Pen and paper 10](#_Toc339973958)

[7.2 OnePassword 10](#_Toc339973959)

[7.3 Wallet 10](#_Toc339973960)

[8 Features 11](#_Toc339973961)

[9 Feature Mapping 13](#_Toc339973962)

[10 Constraints 14](#_Toc339973963)

[11 Product Overview 15](#_Toc339973964)

[11.1 Product Perspective 15](#_Toc339973965)

[11.2 Elevator Statement 15](#_Toc339973966)

[11.3 Summary of Capabilities 15](#_Toc339973967)

[11.4 Assumptions 15](#_Toc339973968)

[11.5 Rough Cost Estimate 15](#_Toc339973969)

[12 Use Cases and Test Cases 16](#_Toc339973970)

[UC1: Set up Master Password 16](#_Toc339973971)

[UC2: Copy/Paste Passwords 16](#_Toc339973972)

[UC3: Set Password Standards 17](#_Toc339973973)

[UC4: Get code from GitHub 18](#_Toc339973974)

[UC5: Set up connection to WebDAV server 19](#_Toc339973975)

[UC6: Set Notes 21](#_Toc339973976)

[UC7: Lookup Password 22](#_Toc339973977)

[UC8: Add Password 22](#_Toc339973978)

[UC9: Automatic Sync 24](#_Toc339973979)

[UC10: Data Sync 24](#_Toc339973980)

[13 Feature to Use Case Mapping 26](#_Toc339973981)

[14 Supplementary Specifications and Test Cases 27](#_Toc339973982)

[15 Coding Standards 32](#_Toc339973983)

[16 Change Control 33](#_Toc339973984)

[17 Usability Report 34](#_Toc339973985)

[17.1 Process 34](#_Toc339973986)

[17.2 Analysis 34](#_Toc339973987)

[17.3 Findings 35](#_Toc339973988)

[17.4 Presentation 35](#_Toc339973989)

[18 Interaction Architecture 36](#_Toc339973990)

[19 Interface Design 37](#_Toc339973991)

[19.1 Initial Design 37](#_Toc339973992)

[19.2 Revised Design 38](#_Toc339973993)

[20 Version Control 39](#_Toc339973994)

[Index 40](#_Toc339973995)

[Glossary 41](#_Toc339973996)

[References 42](#_Toc339973997)

[Appendix A 43](#_Toc339973998)

## 1 Executive Summary

The goal of this project is to create a “password safe” application. This application will allow users to generate and store passwords with varying levels of strength in a secure location. The user will need only a single password to access their safe, and then they can view any of their previously stored passwords, along with the corresponding username. This product will allow for quick and easy lookup of all stored passwords, as well as directly linking a password to the appropriate site. Passwords can be stored either on a WebDAV server or the Omni Group server.

The client, Curt Clifton, is a developer at Omni Group. Omni Group was founded in 1989 and develops applications for MAC, iOS, and iPad. Omni Group does not have a similar system in place and is not looking for an application for personal use; instead they want a product for general users.

In this document, the features of the system will be described. These features give an outline of the functionality that will be present in the product.

The use cases of the system will be explained in detail in the system. These use cases describe how users will interact with the system to meet the features. The conditions for each case, the basic flow, and the alternate flow are discussed in detail. In addition, supplementary specifications are present to convey requirements that cannot be represented well by use cases.

There are multiple levels of data flow diagrams presented. These demonstrate the flow of the system. In different levels the diagrams will demonstrate the flow of data through the system with varying levels of technical detail.

This document also contains test cases that will be used later in development to assure that the product has a high quality. These test cases are derived from the use cases and supplementary specifications to ensure that all requirements are tested thoroughly.

Finally this document contains a usability study. It describes the process that was used to get feedback from users and also changes that were made based on the findings and feedback of the study.

## 2 Introduction

The goal of this project is to solve a simple problem. In today’s day and age, accounts are required for everything. Each of these accounts has its own separate information: a username and a password. These passwords need to be unique and difficult to hack, so users do not lose sensitive information. The “password safe” application will address this issue by providing users with an easy and secure way to generate and store passwords for each of their various accounts.

The password safe will generate passwords with varying levels of strength. Users can set the strength of the password by choosing whether or not to include numbers, letters, or special characters in the password. The application will also store these passwords in a secure location, either on Omni Group’s server or a WebDAV server set up by the client. Each of these passwords can be linked to a username and an account such as a website. In addition, users will be able to create and store notes that contain text. This will allow them to keep track of any additional important information they desire.

## 3 Client Background

Curt Clifton is a developer with Omni Group. He helps develop applications for Mac, iOS, and iPad. He works with smalls teams on a variety of applications, specifically the framework across applications. He spends his days meeting with colleagues on his team and writing Omni Group, Curt’s employer, was founded in 1989. They are made up of 52 employees and have little to no formal Project Management.

## 4 Current System

Omni Group currently doesn’t have any system in place. The product isn’t for use within the company, but for users who have the need. There are several similar products already on the market. Two of these are Wallet and OnePassword. OnePassword is the one we are going to use to make design decisions. Some positive thing about this software is that it allows its users to get their passwords anywhere, it has strong password generation, and it can operate as a browser extension. The biggest flaw with the software is its dependence on Dropbox. OnePassword uses Dropbox to store the password information, which is a simple solution, but if Dropbox ever goes down, the user loses all their password information. This is the biggest thing Curt and Omni want to change. They want password storage software that allows the user to choose where the information is stored.

## 5 User and Stakeholder Description

### 5.1 Nontechnical user profile and environment

The nontechnical user is a typical person with average skills that will use an app developed for the Mac or iOS markets.  They know enough to download and install an app, and are most likely to use multiple different websites that keep track of passwords.  They are not skilled with setting up their own WebDAV server so they are likely to use OmniGroup’s base sync-server.  The nontechnical user will want an easy to use and flexible system with a low learning curve and quick functionality.   
The environment for a nontechnical user will be the basic graphic user interface found on a Mac computer or an iOS device.  The system will require a data or an internet connection.  This group is defined as mainly themselves, but technical and developing users have the same use environment.

### 5.2 Technical user profile and environment

The technical user will be a user that has the same basic needs and uses as a nontechnical user, but will use the capability to set up and use his/her own personal WebDAV server.  They have pretty good technical skills with good knowledge of setting up a server, and using apps.  They share the same end goals as the nontechnical users.  
The technical user’s environment will still be a Mac or iOS device to run the app.  However, this type of user will develop their own server with whatever they chose on whatever they choose.  They must have a setup guide for the server so it can easily be synced with the app.

### 5.3 Open source developer profile and environment

The open source developer will have very high end technical knowledge.  This project will be an open source, so this user will help develop the code and submit new features, or use this as part of a bigger system.  They have a fully functional knowledge of the code base and all of the API calls and other types of things in the system.  
The technical user has a very specific environment.  While they can use the program their main case is the development and advancement of the system.  They will have develop the code on a MAC as the code will be written in objective C and will be written for MAC devices.

## 6 User Needs

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Need | Effects | Risk | Stability | Possible Solution |
| N1 | Ability to generate strong and unique passwords | All users | Low | High | Software will use a set of options like: lower case only, numbers, symbols, and short words to generate a password of a set length |
| N2 | Host the code to allow it to be open source | Open Source Developers | Low | High | Host the code on a public GitHub repository |
| N3 | Allow the user to choose between a private WebDAV server or Omni Sync Server | Technical users  & Open Source Developers | High | Low | Set up a basic system through abstraction and good API design that will allow the user to easily set up the system with a different backend system |
| N4 | Link generated passwords to usernames and websites | All Users | Med | Med | Allow for a table structure that will link usernames, passwords, and the websites they belong too. |
| N5 | Allow for quick and secure lookup of data | All users | High | Low | Allow for the lookup of secure information quickly using the proper security protocol.  Also, make this encrypted exchange quickly with the least information coming through as possible. |

## 

## 7 Alternatives and Competition

### 7.1 Pen and paper

Every person has the ability to keep track of all of their secure and hard to remember passwords with a pen and paper.  While this is horrible practice, and can lead to a lot of security problems if a malicious person finds the paper, it can be performed by any person.

### 7.2 OnePassword

OnePassword is one of the direct competitors to the system.  It allows users to store all their passwords and allows for browsing through the app itself.  It also allows you to store other sensitive information in the app like credit card information, telephone numbers and other information.  However, this system makes you store all this information on Dropbox.  This is the problem this solution will solve.

### 7.3 Wallet

This is basically a different implementation of OnePassword.  It is another direct competitor

## 8 Features

These are the criteria that will be used to evaluate each feature.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Priority  Indicates the importance of the feature | |  |  | | --- | --- | | Critical | Needed for the project to be a success | | Important | Significantly contributes to the quality of the project | | Useful | Not necessary, but a possible bonus feature | |
| Effort  Indicates the effort required to implement the feature | |  |  | | --- | --- | | High | Requires a lot of effort | | Medium | Moderate effort required | | Low | Little effort required | |
| Risk  Indicates the likelihood that the feature will cause problems | |  |  | | --- | --- | | High | Likely to cause problems | | Medium | Could cause problems | | Low | Unlikely to cause problems | |
| Stability  Indicates the likelihood that the feature will change | |  |  | | --- | --- | | High | Unlikely to change | | Medium | Could change | | Low | Likely to have change in specifications | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Feature | Status | Priority | Effort | Risk | Stability | Reason |
| F1 | Password generator | Approved | Critical | Med | Med | High | The main purpose of the project is to store passwords, so without being able to generate them it will obviously be a failure. |
| F2 | Allow user to set standards for password generation | Approved | Important | Med | Low | Med | It is important for the user experience to allow customization of how passwords are generated so that they can be memorable or very secure. |
| F3 | Source code on Github | Approved | Important | Low | Low | High | It is important to the client for the code to be open source so that others can take the product and improve on it if they wish. |
| F4 | Allow clone of repository on GitHub | Approved | Important | Low | Low | Med | This will allow people to see how the product was programmed, which is important to the client. |
| F5 | Set up fork access on GitHub | Approved | Important | Low | Low | Med | This will allow developers to easily make improvements and make it available to others, which is important to the client. |
| F6 | Store on OmniServer | Approved | Important | Med | Med | High | This should be the default server for non-technical users, so it should be easy for them to use. |
| F7 | Store on chosen WebDAV server | Approved | Important | High | High | High | This will give advanced users the opportunity to store their data wherever they want. |
| F8 | Database links username, password, and website | Approved | Critical | Low | Med | High | These pieces of information needs to be linked, because on their own they have no use. |
| F9 | Encryption | Approved | Critical | Med | Med | High | Without encryption, anyone could get access to the users personal information. |
| F10 | Only sync data that needs to be changed | Approved | Important | High | High | Med | This will keep the time needed for synchronization down, which is important for the satisfaction of the users. |

## 9 Feature Mapping

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | N1 | N2 | N3 | N4 | N5 |
| F1 | X |  |  |  |  |
| F2 | X |  |  |  |  |
| F3 |  | X |  |  |  |
| F4 |  | X |  |  |  |
| F5 |  | X |  |  |  |
| F6 |  |  | X |  |  |
| F7 |  |  | X |  |  |
| F8 |  |  |  | X |  |
| F9 |  |  |  |  | X |
| F10 |  |  |  |  | X |

## 10 Constraints

|  |  |  |
| --- | --- | --- |
| ID | Constraint | Reason |
| C1 | Platform is Mac and iOS | These are the platforms that the client’s company works with.  He would like us to work on them so that the project is more useful to his company, and so that they can help us with technical difficulties we have.  We will have to get loaner machines to do this since we don’t have and Macs.0.0. |
| C2 | Follow coding standards | The client would like for us to follow his company’s coding standards so that the company can easily understand what the code is doing. |
| C3 | Store code on GitHub | The project is required to be open source, and this is the means of storage that the company uses |
| C4 | Passwords must be strong | The passwords must be strong to protect the users’ information.  A third party library will be used for encryption. |
| C5 | Security questions should be used | Security questions provide an additional layer of security, but caution must be used.  With so much information available online, it would be easy for someone to find the answer to the question, so the question should use some sort of code to be more secure. |

## 11 Product Overview

### 11.1 Product Perspective

Our password safe will be a standalone system, save for a possible external server to store the passwords generated. It needs to be accessible to the user whenever needed, so many services will be available without an internet connection.

### 11.2 Elevator Statement

In today’s world everything required an account, and every account requires a password. To maintain a strong level of security these passwords need to be complex. However, remembering numerous gibberish passwords is not easy. Our password safe application would require the user to remember only one password, which gives them access to a list of strong passwords, each randomly generated for their accounts.

### 11.3 Summary of Capabilities

The primary purpose of our password safe is to generate strong passwords for the user and save them to a secure location. It will support multiple options to allow the user to specify the complexity of the password. These options include whether or not to use letters, numbers, or special characters. Each password will allow a username to be stored alongside it.

### 11.4 Assumptions

The client wants the product on Mac and iOS. He is also willing to work with Windows and Android, if the former plan fails,

### 11.5 Rough Cost Estimate

The cost of developing our project will be primarily labor costs, which we are doing for free as students, The client is considering providing Macs for us to work on during the project, which will add to the cost accordingly.

## 12 Use Cases and Test Cases

UC1: Set up Master Password  
  
**Description:** The user sets up a master password to open the application.  
**Actors:** User  
**Main Flow:**  
    1. The system prompts for username and master password.  
           2. The user enters username.  
    3. The user enters password.  
    4. The user verifies password. (c)  
    5. The system sets password.  
**Alternate Flows:**  
    a. The user can quit the process at any time.

b. The app may shut down unexpectedly (user shut off phone, phone lost battery and died, etc), causing this process to cancel at any stage.

c. The user verifies the password incorrectly

**Pre-Conditions:** The system is installed  
**Post-Conditions:** The user has a username and password to log into the application

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 1.1 | The user correctly enters a username and password and verifies the password. (Main flow) | The system is installed. | “Username” “Password” “Password” | The user now has an account that can be logged into. |
| 1.2 | The user incorrectly verifies the password. (Alternate flow c) | The system is installed. | “Username” “Password” “PAssword” | The user is shown an error message and asked to reenter the password. |
| 1.3 | The user correctly enters information, but then cancels the process instead of submitting. (Alternate flow a) | The system is installed. | “Username” “Password” “Password” | No account is created. |

UC2: Copy/Paste Passwords  
  
**Description:** Once the user has access to their password, they may copy and paste their password from the application to the entry field of their browser.  
**Actors:** User  
**Main Flow:**

1. The user highlights password they would like to copy, and selects copy.

2. The system takes the password and places it on the clipboard of the OS. (a)

3. User selects where they would like to paste, and selects paste.

**Alternate Flows:**  
           a. The system fails to store the text in the clipboard

b. The app may shut down unexpectedly (user shut off phone, phone lost battery and died, etc), causing this process to cancel at any stage.

**Pre-Conditions:** User has access to their passwords  
**Post-Conditions:** User is at the password access screen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 2.1 | The user copies a password. (Main flow) | The system is installed. | Highlight the desired password and select copy | The password is stored in the clipboard |
| 2.2 | The user pastes a password. (Main flow) | The system is installed.  The user has already copied a password | Select paste | The password is pasted into the desired area. |
| 2.3 | The system fails to store the text in the clipboard. (Alternate flow a) | The system is installed.  The user has tried to copy a password. | Select paste | No text is stored in the clipboard, so it will be empty and will not paste anything. |

UC3: Set Password Standards  
  
**Description:** The user can choose how strong their password will be by determining what kind of characters to use and password length.  
**Actors:** User  
**Main Flow:**

1. The user navigates to the “Generate Password” screen. (a)

2. The user selects which type of characters to include. (lowercase letters, uppercase letters, numbers, special characters) (b)

3. The user selects a length for the password. (b)

4. The user clicks the “Generate Password” button.

5. A password matching the given constraints appears in the box at the bottom of the screen.

**Alternate Flows:**

a. The device could fail to load the “Generate Password” screen. User is returned to main screen.

b. The process can be canceled at this point by the user or unexpected device shutdown.

c. The app may shut down unexpectedly (user shut off phone, phone lost battery and died, etc), causing this process to cancel at any stage.

**Pre-Conditions:** The user needs the app installed on a device and an account to save the password with.

**Post-Conditions:** The user is on the password creation screen and a password matching the given standards is generated

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 3.1 | The user changes the amount of uppercase letters (Main flow) | The system is installed.  All other parameters are left as the default | Uppercase: 5 Special: 0 Numbers: 0 Length: 10 | Password with at least as many of each type of character |
| 3.2 | The user changes the amount of special characters (Main flow) | The system is installed.  All other parameters are left as the default | Uppercase: 0 Special: 3 Numbers: 0 Length: 10 | Password with at least as many of each type of character |
| 3.3 | The user changes the amount of numbers (Main flow) | The system is installed.  All other parameters are left as the default | Uppercase: 0 Special: 0 Numbers: 4 Length: 10 | Password with at least as many of each type of character |
| 3.4 | The user changes the length of the password (Main flow) | The system is installed.  All other parameters are left as the default | Uppercase: 0 Special: 0 Numbers: 0 Length: 12 | Password with at least as many of each type of character |
| 3.5 | The user changes all parameters. (Main flow) | The system is installed | Uppercase: 2 Special: 5 Numbers: 3 Length: 15 | Password with at least as many of each type of character |
| 3.6 | The user changes parameters but then exits (Alternate flow b) | The system is installed. | Uppercase: 2 Special: 5 Numbers: 3 Length: 15 | Nothing; navigated to the last screen the user was on |

UC4: Get code from GitHub  
  
**Description:** Password Safe is an open source project that stores code on GitHub. Users can download code directly from the repository.  
**Actors:** Developer  
**Main Flow:**

1. The user navigates to the GitHub webpage. (a)

2. The user navigates to the Password Plus repository.

3. The user forks the repository. (b)

**Alternate Flows:**

a. The webpage may not be found (down for maintenance, the user has a bad connection. etc.). The user is redirected by their browser.

b. The fork may fail (incorrect account information, user installed GitHub improperly, failed to download due to other reasons).

c. This process could be canceled at any time by the user.

**Pre-Conditions:** The user must have a GitHub account and an internet connection.  
**Post-Conditions:** The user must have a copy of the code.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 4.1 | The user forks the repository and all information is on their computer (Main flow) | The system is installed.  Navigated to the Password Safe repository | Click the “fork” button on the GitHub website | The user gets all code on their website |
| 4.2 | The user tries to fork the repository, but the information does not download (Alternate flow b) | The system is installed.  User has navigated to the Password Safe repository on GItHub, but no longer has internet connection | User clicks the “fork” button on the GitHub website | The user will get no information |
| 4.3 | The user tries to fork the repository, but is not logged into GitHub (Alternate flow b) | The system is installed. User navigated to Password Safe repository, but is not logged in. | User clicks the “fork”  button on the GitHub website | The user will get no information due to the error of not being logged in |

UC5: Set up connection to WebDAV server  
  
**Description:** The user can set up a WebDAV server to connect to instead of the OmniSync server.  
**Actors:** Advanced User  
**Main Flow:**

1. The user navigates to the “Settings” screen.

2. The user enters information corresponding to the desired server to sync to. (b)

3. Application connects to the server. (a)

**Alternate Flows:**

a. The app may shut down unexpectedly (user shut off phone, phone lost battery and died, etc), causing this process to cancel at any stage.

b. The user enters invalid information

**Pre-Conditions:** User must have a WebDAV server to sync with.  
**Post-Conditions:** Application now syncs with WebDAV servers.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 5.1 | The user enters the correct information to a WebDAV server (Main flow) | The system is installed.  The user has navigated to the “Settings” screen  The WebDav server is properly set up and is currently setup to connect to devices | IP Address: 192.168.0.76  Username: spurrme  Password: P@ssw0rd@1 | The user connects to the server and the data syncs with a the server |
| 5.2 | The user enters the valid IP address for the server but not the correct login information (Alternate flow b) | The system is installed.  The user has navigated to the “Settings” screen  The WebDav server is properly set up and is currently setup to connect to devices | IP Address: 192.168.0.76  Username: spurrme  Password: P@ssw0r21 | The user does not connect and is prompted with a connect error screen |
| 5.3 | The user enters the correct information to a WebDAV server.  During the initial process the phone loses connection for some reason (Alternate flow a) | The system is installed.  The user has navigated to the “Settings” screen  The WebDav server is properly set up and is currently setup to connect to devices.  The initial setup process is in process. | IP Address: 192.168.0.76  Username: spurrme  Password: P@ssw0rd@1 | The server will save state of device credentials immediately.  Phone will save progress at stages to allow for quick recovery of the system. |

UC6: Set Notes  
  
**Description:** The user can store any notes that they wish.  
**Actors:** User  
**Main Flow:**

1. The user navigates to “Notes” screen. (a)

2. The user clicks “Add/Change Note” button.

3. The user is directed to the “Enter Note” screen. (b)

4. The user types a note.

5. The user clicks the “Finished” button. (c)

6. The note is saved.

7. The user is returned to the “Notes” screen.

**Alternate Flows:**

a. The user may click the “Cancel” button at any step prior to step 5. User will be returned to the “Notes” screen and no data will be saved.

b. This process may be canceled at any time by unexpected device shutdown. Data will not be saved unless the process has already completed step 6.

**Pre-Conditions:** The user must have the app installed  
**Post-Conditions:** The note must be saved. The user is returned to “Notes” screen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 6.1 | The user creates a note (Main flow) | The system is installed.  Currently on the “Enter Note” screen | Text:  This is a test note | The note is saved and the user is redirected to the notes screen |
| 6.2 | The user cancels the note he is creating. (Alternate flow a) | The system is installed.  Currently on the “Enter Note” screen | Text:  This is a test note | The note is discarded and memory used is wiped |
| 6.3 | The application crashes or the phone is turned off while user is creating a note (Alternate flow b) | The system is installed.  Currently on the “Enter Note” screen | Text:  This is a test note | The application will not save the note |

### **UC7: Lookup Password**

**Description:** The user looks up passwords and their related accounts  
**Actors:** User  
**Main Flow:**  
    1. The user navigates to Password screen  
    2. The user selects desired account  
    3. The system displays information for that account (b)  
**Alternate Flows:**  
    a. The user can quit the process at any time

b. The app may shut down unexpectedly (user shut off phone, phone lost battery died, etc), causing this process to cancel at any stage.

**Pre-Conditions:** The system is installed.  
    The user has an account.  
**Post-Conditions:** Password is displayed along with username

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 7.1 | The user selects an account to see the password associated with it (Main flow) | The system is installed.  User must be on the “Password” screen | User selects the account | Application displays all passwords associated with that account |

UC8: Add Password  
  
**Description:** The user sets a password and username for a new account  
**Actors:** User  
**Main Flow:**  
    1. The user navigates to Generate Password screen (e)  
    2. The user sets standards for password generation  
    3. The system generates password  
    4. The user clicks button to advance to account creation screen (b)  
    5. The system associates generated password with new account  
    6. The user enters username  
           7. The user clicks “Create” button  
           8. The system stores information (c)  
  
**Alternate Flows:**  
    a. The user can quit the process at any time  
    b. The device may fail to load account creation page. User is returned to main screen.  
    c. The device may fail to create new account. User is notified and returned to main

screen.

d. The app may shut down unexpectedly (user shut off phone, phone lost battery and died, etc), causing this process to cancel at any stage.

e. The user can skip steps 1-3 and instead use a password that they created.

**Pre-Conditions:** The system is installed  
    The user has an account  
**Post-Conditions:** A password is created for a new account

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 8.1 | The user changes one of the settings for the password (Main flow) | The system is installed.  All other settings retain their original values | User moves one of the sliders to a desired setting | That setting is saved and any password generated will obey that setting |
| 8.2 | The user clicks the “Generate” button (Main flow) | The system is installed.  The password must meet all specified criteria | The user selects password settings and clicks the “Generate” button | A password is displayed to the user that meets all the selected criteria |
| 8.3 | The user clicks the “Generate Password and Create Account” button (Main flow) | The system is installed.  The password must meet all specified criteria | The user selects password settings and clicks the “Generate Password and Create Account” button | The user is directed to the account creation page. The generated password is automatically placed in “Password” field |
| 8.4 | The user enters information but then exits (Alternate flow a) | The system is installed. | User moves sliders to desired settings | Nothing; navigated to the last screen the user was on |

### UC9: Automatic Sync

**Description:** User’s data is synced at application startup  
Actors: User  
**Main Flow:**

1. The user enters Master Password at Login Screen.

2. The system checks the information for correctness.  (a)

3. The system then automatically syncs the data with the server.

**Alternate Flow:**

a. The system checks the information and it is incorrect. The system then displays the Log in screen with a message.

b. The app may shut down unexpectedly (user shut off phone, phone lost battery and died, etc), causing this process to cancel at any stage.

**Pre-Conditions:** The user has setup passwords, syncing to server, and has enabled Automatic Data Sync  
**Post-Conditions:** The user is at the Main Menu and data has been synced. Device is in the “ground truth” state.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 9.1 | Data is synced when the application starts (Main flow) | The system is installed.  The device must be able to sync with the server. The “Sync Automatically” setting must be active. | User starts the application and enters their master password | All passwords and data that are allowed to sync are synced with the server |

### **UC10: Data Sync**

**Description:** The user can manually sync data with the server by changing a setting that allows the app to sync automatically.  
**Actors:** User  
**Main Flow:**

1. The user navigates to data sync page

2. The user clicks the “Sync Data with Server” button (a)

3. The app connects to OmniSync or WebDAV server (b)

4. Any changed passwords have their new values saved on the server. Any data on the server that has changed is saved to the device (if applicable)

5. The user is returned to the Sync Data screen

**Alternate Flows:**

a. The user checks the “Sync Data Automatically” box. The app now checks for changes to passwords on startup and saves any changes to the server on shutdown

b. The server cannot be reached. User is returned to the data sync page.

c. The app may shut down unexpectedly (user shut off phone, phone lost battery and died, etc), causing this process to cancel at any stage.

**Pre-conditions:** The user has saved data to sync  
**Post-conditions:** All data is saved, and the user is at the data sync screen.  The device is in the “ground truth” state.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 10.1 | The user clicks the “Sync Data with Server” button (Main flow) | The system is installed.  The device must be able to sync with the server | User clicks the “Sync Data with Server” button | All passwords and data that are allowed to sync are synced with the server |
| 10.2 | The user checks the “Sync Automatically” button (Alternate flow a) | The system is installed.  The device must be able to connect to the server when it attempts to sync | User checks the “Sync Automatically” button | The device now syncs data with the server each time the user logs in |

## 13 Feature to Use Case Mapping

Waiting for a final list and approval of use case before finishing this section

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | F10 |
| UC1 |  |  |  |  |  |  |  |  |  |  |
| UC2 |  |  |  |  |  |  |  |  |  |  |
| UC3 | X | X |  |  |  |  |  |  |  |  |
| UC4 |  |  | X | X | X |  |  |  |  |  |
| UC5 |  |  |  |  |  | X | X |  |  |  |
| UC6 |  |  |  |  |  |  |  | X |  |  |
| UC7 |  |  |  |  |  |  |  | X |  |  |
| UC8 | X | X |  |  |  |  |  | X |  |  |
| UC9 |  |  |  |  |  |  |  |  | X | X |
| UC10 |  |  |  |  |  |  |  |  | X | X |
| UC11 |  |  |  |  |  |  |  |  | X | X |

## 14 Supplementary Specifications and Test Cases

Source:  Users  
Stimulus:  User opens the application for the first time  
Artifact:  The system  
Environment:  During normal operation  
Response:  The User uses the app for the first time after reading the manual.  
Response Measure:  The User will judge that the base features of the application are intuitive and easy to use from application launch. The system is intuitive enough for the user to be able to understand how to use it with little experience. Additionally, the user manual explains the system effectively.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 11.1 | The user tests out the application for the first time | The user is shown a prototype | User testing | User thinks that the application is easy to use and has useful features |

Source:  Users  
Stimulus:  User opens the new password screen  
Artifact:  The system  
Environment:  During normal operation  
Response:  The User creates a new password.  
Response Measure:  The system will be designed to be responsive and easy to use, so the user can create a password in under 2 minutes from system launch.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 12.1 | The user creates a new password | The system is installed. | User follows basic flow of Use Case 8 | A new password is stored, and it took less than 2 minutes |

Source:  Users  
Stimulus:  User opens the new note screen  
Artifact:  The system  
Environment:  During normal operation  
Response:  The User creates a new note  
Response Measure:  The system will be designed to be responsive and easy to use, so the user can create a 4 sentence note in under 5 minutes from system  launch.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 13.1 | The user creates a note | The system is installed. | User follows basic flow of Use Case 6 | A new note is stored, and it took less than 5 minutes. |

Source:  Users  
Stimulus:  User opens the Password screen  
Artifact:  The system  
Environment:  During normal operation  
Response:  The User views a password  
Response Measure:  The system will be designed to be responsive and easy to use, so the user can view a password in under 30 seconds from system launch.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 14.1 | The user views a password | The system is installed. | User follows basic flow of Use Case 7 | User sees note in under 30 seconds. |

Source:  Users  
Stimulus:  User opens the application  
Artifact:  The system  
Environment:  During normal operation  
Response:  The User uses the application from day to day  
Response Measure:  The system will be designed to be responsive and easy to use. 50% of  Users judge the application as useful or more useful than OnePassword.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 15.1 | The user rates the application. | The system is installed.  The user has become familiar with the system. | User’s use of system | 50% of users believe it is as useful as OnePassword |

Source:  Users  
Stimulus:  User does not know something about the system  
Artifact:  The Readme file  
Environment:  During normal operation  
Response:  The User wants to learn more about the system  
Response Measure:  The system will include a Readme file that answers basic instructions on how to use the system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 16.1 | The user reads the ReadMe | The system is installed. | User reads the ReadMe | The users questions are answered due to the content in the ReadMe |

Source:  Internal to the system  
Stimulus:  User navigates screens  
Artifact:  The system  
Environment:  During normal operation  
Response:  The system handles the users input and responds accordingly.  
Response Measure:  Any supported device running the app should be capable of fluid screen navigation at sixty frames per second. Any input given by the user that moves to another screen or opens a dialog box should run at a minimum of sixty frames per second .

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 17.1 | The user uses the system | The system is installed. | The user navigates through screens | The navigations operate at sixty frames per second. |

Source:  Internal to the system  
Stimulus:  User clicks “Generate Password”  
Artifact:  The system  
Environment:  During normal operation  
Response:  The system generates a password based on the given constraints and displays it to the user  
Response Measure:  The time taken to complete each generation should be consistent. The average run should take a hundredth of a second. The worst case run should take no longer than a quarter of a second with an animation to show wait time.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 18.1 | The user generates a password | The system is installed. | User changes standards and clicks button to generate password | The average time should be less than a hundredth of a second |

Source:  Internal to the system  
Stimulus:  User syncs to Omni Sync or a WebDAV server  
Artifact:  The system  
Environment:  During normal operation  
Response:  The system updates passwords on the server and the device to the most recent version  
Response Measure: Syncing data with the server should be a consistently fast process. However, the time taken to sync is dependent on the server cannot be directly specified. To facilitate the syncing process, Password Safe will use coding standards that minimize server interaction. Additional detail on coding standards in general will be given in section 13. The time taken to sync should scale linearly with the number of passwords (and corresponding username and/or note) that have to be synced. This number is generated by discovering all the password that have changed since the last sync of the file. To take internet connectivity out of the problem a user will be able to sync a password in a tenth of a second on a Local Area Network.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 19.1 | Data syncs with the server | The system is installed.  The device must be able to sync with the server | User clicks the “Sync Data with Server” button | Data synced linearly based on size |

Source:  External to the system  
Stimulus:  The server goes offline.  
Artifact:  The system  
Environment:  During normal operation  
Response:  The system cannot access data on the server, but it should still be able to access data stored locally.  
Response Measure:  The system should be able to access local and server data 100% of the time that the server is online and operating correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 20.1 | The application cannot connect to the server | The system is installed. | User uses the application | Local data can be accessed |

Source:  Internal to the system  
Stimulus:  The application fails.  This includes failing to start the application or load a page, getting incorrect information, or not being able to connect to the server when the server is online.  
Artifact:  The system  
Environment:  During normal operation  
Response:  The system should display an appropriate error message.  
Response Measure:  The system should be available again within 2 minutes of restarting it.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 21.1 | The system fails | The system is installed. | User uses application  Cause error | System is available again in 2 minutes |

Source:  Developers  
Stimulus:  They wish to add their own features or update ours.  
Artifact:  Code  
Environment:  Our implementation is complete.  
Response:  The code is made available on GitHub and is well documented to make it easier for developers to make their own changes.  
Response Measure:  The code should be documented enough so that developers can make moderate changes within 2 weeks.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Conditions | Sample Input | Expected Output |
| 22.1 | Developers modify code | The code is available on GitHub | Developers modify code | Developers should find the code well documented and easy to modify |

## 15 Coding Standards

There are two main sources of coding standards. The first is Omni’s Coding Standards that they use for all of their projects. Second is the App Store and Mac App Store Review Guidelines. Both of these standards will lead to better code and also ensure that our software will be able to be on the App Store for easier access.

## 16 Change Control

Requests will be handled by any team member receiving one from an outside source. Github Issues feature will be used for any requests from outside sources. Team members will review Issues regular and resolve them based on importance to the stability of the project.  
  
Changes will be managed by the team as a whole. Any Issues will be discussed by the team as a whole at the following meeting and a decision on the course of action will be determined. How team member changes are managed is dependent on the magnitude of the change. Any change that is less than or equal to 5 lines of code can be altered by the approval of 2 teams members, while any major change greater than 5 lines of code must be approved by the entire team. All changes will reported to the team at the meeting following the change.   
  
Github will be the main component of our source control. All artifacts, including text documents, Milestones, and all code will be kept on the repository and managed using Github’s software. Additionally, all code that is committed and pushed to the repository must be able to build and run with no error and not degrade the functionally of the current version in any way.

## 17 Usability Report

### 17.1 Process

It was attempted to try and keep the flow of each session and each tester as consistent as possible. First they were brought into the testing room with a laptop in front of them and given some background on the product they were testing and the test itself. After, the subject asked if they had any questions and then the test began. The test had 7 steps to it. The steps are as follows:

* Log In
* Generate password with set parameters
* Store the password and associate it with the network account
* Look at a note
* Look at a password
* Edit a password
* Pick the favorite layout

The tasks were organized so that they were related and flowed well together. The first 6 steps were all done in succession on the prototype, and the 7th steps involved the subject viewing 3 different mobile UI layouts and they picked their favorite one and explained why. If at any time a tester struggled with a specific task for too long, they were asked what the problem was and then assisted. Once all the tasks were completed, the test was concluded.

Field testing was conducted by showing the prototype to users and having them experiment with its functionality. They were then asked about the functionality of the application and the ease of user of the interface. Finally, they were asked to choose one of the three interface designs (shown in section 19).

The tester consent form given to each participant in our usability study is included in Appendix A.

### 17.2 Analysis

Testing went well for the most part. Although the demographic of the subjects was varied, the users all experienced difficulty with the same things. The screen that caused the most problems was the Password Generate screen. Tester often struggled with how the screen worked and how to adjust settings. Also, the meaning of the button names was unclear. Navigation between menus and logging in was no problem and users could edit passwords easily. The results of the last step were interesting. Of the three designs (shown in section 19), men preferred the more colorful, minimalist design, while the women prefer the other two.

### 17.3 Findings

This following section details the findings of the conducted usability test and field studies. It is a group of observations made during the tests followed by an explanation on how we plan to incorporate features into the final system to make the system better for the users.

#### 17.3.1 Keyboard shortcuts

During the usability and field tests most users tried using a couple different keyboard shortcuts that are commonly built in to many desktop applications. An example of this would be pressing the enter key when the user wishes to submit information. In the final system the application will have basic shortcuts that are built into many systems.

#### 17.3.2 Label slider bars

Throughout both tests it was clear that users of the system had no idea what the slider bars truly did. The bars were to change the generated password, but the users were not truly able to understand the function of the bars. To fix this we should automatically generate a password of a set length, and default parameters when the user navigates to the screen. Then label the bars and show the changes to the password as the user slides the bar across the screen.

#### 17.3.3 Make screens more intuitive

Some problems with the basic screens of the system were the users were not able to look at the screen and determine how it works. The base idea of the fix would be to take user learning out of most of the system. Limit the items that a user can interact with the least amount it can be.

#### 17.3.4 Put in help screens

Since most of the tasks in the usability study successfully accomplished a task after brief assistance, help screen should be incorporated into the document. The solution would be to put a button with a question mark on the top of each page to explain the individual components on the page and explain what the overall solution is.

#### 17.3.5 Difference between the Usability and Field Tests

The users participating in field tests had an easier time understanding the application by playing with it than those who followed the steps in the usability test. This may be due to many reasons. The usability lab scenario may have made participants nervous, or the steps may have been too complicated. It is also possible that the app is easier to understand through practice than a step-by-step example.

### 17.4 Presentation

The prototype that we tested our on the users will not suffice for the demo to our client. There needs to be added help for users to be able to use the system better. Also, there needs to be a redesign of the user interface design. The team needs to take into account all the positives of the three demo systems and make that the base interface for the system. The flow and architecture of the system is a great design and flow of the system.

## 18 Interaction Architecture

For the initial prototype, the team developed a basic Java application that included most of the functionality. Users could generate passwords with different settings and also store them with a username and website. Users could also create and store notes. The team believed that these features were important to include in the prototype because they represent the typical actions that users will execute in everyday use. Screens that the user could navigate between are shown in the following diagram.

Password Viewer

Note Creator

Note Viewer

Login

Password Creator

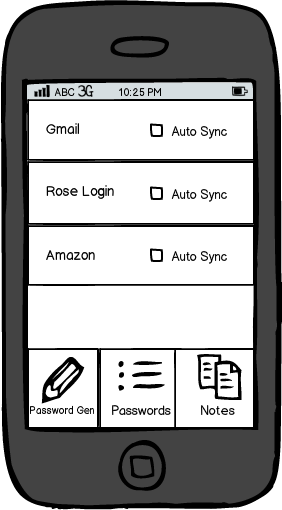
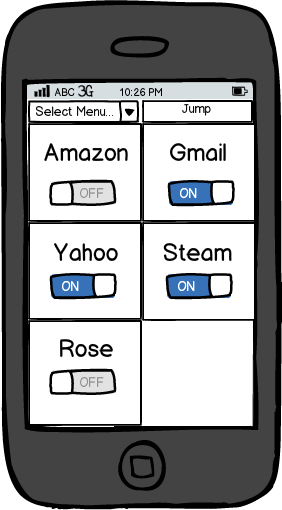
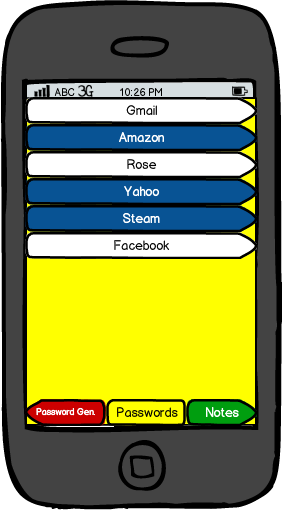
Password Generator

The initial prototype had a very basic user interface. Instead of spending time on improving this, the team decided instead to create several alternative screenshots of what the various pages would look like and show them to the users to get their feedback.

## 19 Interface Design

### 19.1 Initial Design

The initial possible application interfaces appear below.

The first offers a simple table-like design. Users can read off names in the left column, while the sync status of the account appears in the right column (sync status is no longer a supported feature and will not appear in revised designs). Three buttons appear at the bottom of the screen that allow users to jump to one of the other main screens of the application. This simple layout is easy to read and very straightforward.

The second design breaks the screen into squares representing each account. The squares take up a larger portion of the screen. This means that fewer accounts can appear on a screen, but they are easier to read individually. Switches are used to indicate the sync status of each account rather than checkboxes (sync status is no longer a supported feature and will not appear in revised designs). These switches read “On” and “Off” and change color for each state, making them more readable than checkboxes. Instead of three buttons at the bottom of the screen, a single “Jump” button appears in the top right of the screen. Users select a screen with the dropdown box next to the “Jump” button and then click “Jump” to navigate to that screen.

The third design has a single line for each account. This compact layout allows for a large number of accounts on a single page and leaves room for large account names. The color of the account represents the sync status of the account. This forces users to remember which color represents which state, as there is no indicator to remind them. However, because sync status is no longer supported, this is no longer an issue. This layout reverts to using three buttons at the bottom of the screen to jump to other screens.

### 19.2 Revised Design

After collecting user feedback, the first and third screens were found to be the most popular. The key difference between the two is the way they display the sync status of each account. When this feature is removed, the two designs break down into essentially the same design. One of the main benefits of the first design was that it was simple and easy to read. Without the sync column, however, this screen becomes bland. So the third design will be the basis for the final product. It will no longer highlight accounts different colors, which leads to a re-evaluation of the color scheme. A final color scheme has not yet been decided upon.

The other screens of the Password Safe also contain tables of information that will be formatted and colored to match the main screen.

## 20 Version Control

|  |  |  |
| --- | --- | --- |
| Date | Version | What’s New |
| 10/17/12 | 1.0 | Executive Summary Introduction Current System User Needs Features Use Cases |
| 10/18/12 | 1.1 | Test Cases Coding Standards Change Control |
| 10/25/12 | 2.0 | Changes based on PM feedback and updated rubric |

## Index

Curt Clifton...4, 6, 7

GitHub..9, 12, 19, 30, 33

Omni Group..4-9, 12, 19, 25, 29. 32  
  
OnePassword...7, 10, 28

WebDAV...4, 5, 8, 9, 12, 19, 20, 25, 29

## Glossary

API - application programming interface (API) is a specification intended to be used as an [interface](http://en.wikipedia.org/wiki/Interface_%28computing%29) by [software components](http://en.wikipedia.org/wiki/Software_component) to communicate with each other  
  
Data flow diagram -  graphical representation of the "flow" of data through an [information system](http://en.wikipedia.org/wiki/Information_system), modeling its *process* aspects  
  
Encription -  the process of transforming [information](http://en.wikipedia.org/wiki/Information) (referred to as [plaintext](http://en.wikipedia.org/wiki/Plaintext)) using an [algorithm](http://en.wikipedia.org/wiki/Algorithm) (called a [cipher](http://en.wikipedia.org/wiki/Cipher)) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a [key](http://en.wikipedia.org/wiki/Key_%28cryptography%29)  
  
Ground Truth - the state at which a device can be called synched.  It is the state where the file on the device and the file on the server match exactly with data that is set to synched.  
  
Synched - when the device is in the “ground truth” state  
  
Use Case - a list of steps, typically defining interactions between a role (known in [UML](http://en.wikipedia.org/wiki/Unified_Modeling_Language) as an "[actor](http://en.wikipedia.org/wiki/Actor_%28UML%29)") and a system, to achieve a goal  
  
WebDav - an extension of the [Hypertext Transfer Protocol](http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTP) that facilitates collaboration between users in editing and managing documents and files stored on [World Wide Web](http://en.wikipedia.org/wiki/World_Wide_Web) servers

## References

GitHub site - wwww.github.com  
Omni Group site- www.omnigroup.com/  
OnePassword site- www.agilebits.com/onepassword  
Wallet site- www.moxier.com/wallet  
Managing Software Requirements: A Use Case Approach, Second Edition, by Dean Leffingwell and Don Widrig

## Appendix A

**Consent Form for Password Safe Usability Study**

**Why is this research being done?**

The purpose of this study is to get user feedback on the user interface and features of an application that stores passwords and notes.

**What will I be asked to do?**

You will be asked to navigate through the application and test its major features. You will then be asked questions about the application. The questions will focus on your opinions of the design and usability of the application.

The study will take place in the Rose-Hulman Institute of Technology CSSE department’s usability lab. Your time for participation should be from 30 to 45 minutes.

**How will my privacy be protected?**

This research involves recording your opinions and storing your contact information so that you can be contacted in case any answers to questions need to be clarified. You may opt out of giving your contact information by checking the following: \_\_\_\_

All information collected will be stored on personal computers for a maximum of 3 months after the completion of the research and will then be deleted.

Any personal information gathered will only be available to members of the project team.

**Do I have to participate in this research?**

This study is completely voluntary. You do not have to participate. If you do choose to participate, you may decline to answer any question or leave at any time you wish, for any reason.

**Who can answer my questions?**

Any questions regarding the study can be sent to:

Kegan Kaiser: kaiserkp@rose-hulman.edu Jake Schuenke: schuenjr@rose-hulman.edu

Tyler Shelton: sheltotj@rose-hulman.edu Matt Spurr: spurrme@rose-hulman.edu

**Statement of Age and Consent of Subject**

By signing this, you are confirming that you are at least 18 years of age, you understand the study, and you have voluntarily decided to participate in this research.

**Name:**

**Date:**

**Signature:**