PassUSB

Team Number 10: Sonali Benni, Rey Jairus Marasigan, Chidiebere Otuonye, Kaiya Roberts, and Gentman Tan

Florida International University

CEN4010 U02: Software Engineering I

Kianoosh G. Boroojeni

March 11, 2022

PassUSB

Introduction

With the prevalence of information technologies, there exists an ever-increasing need for individuals to secure one's own access to online accounts. The typical method of doing so requires the user to create a secret passphrase that they would then be responsible for memorizing in order to access a given system. However, several factors make such a task difficult and unsafe; first, the exponential rise in computational power has led to the feasibility of "brute force attacks", in turn forcing IT administrators to enforce increased password length and complexity. Another effect that the increase in password length has is making the memorization of multiple different passwords difficult, thereby incentivizing individuals to unsafely reuse their own passwords. With these shortcomings in mind, we propose a system that would solve all of these issues in a single package.

Purpose of system

Solve the security issues facing PC end users in the realm of password authentication

Scope of the system

In scope: Multi-platform mobile application, password management, multi-platform USB keyboard emulation, mitigation against man-in-the-middle, replay and spoofing attacks Out of scope: Application data security, side-channel attacks

Objectives and success criteria of the project

- Provide a mobile app for users to create and store passwords
- Provide a USB hardware dongle that can be paired with the mobile app which can type in passwords in lieu of keyboard input

Definitions, acronyms, and abbreviations

- PassUSB: the project's USB dongle solution that emulates a USB keyboard
- Password manager: a computer program that generates, stores and retrieves passswords for its users
- HID (Human Interface Device): a computer device that facilitates communications between a computer user and a computer
- App: a computer application
- Pairing: the process of recognition and acknowledgement between the mobile device and USB dongle

Overview of document

The project will consist of two types of coding assignments, one for frontend development i.e. mobile app development, and the other for backend development i.e. microcontroller programming. The mobile app will prompt the user to create a new password database, in which he/she will then enter a master password that is to be used to secure the database. The user will be given an option to pair the PassUSB with the app. Should the user choose to or not choose to pair the PassUSB, the user is then able to utilize the app's password generation, management and storage features.

Current System

Without password managers, users are succeptable to password leaks which can lead to the exploitation of other accounts that the user owns if the same password is used. Also, the memorization of increasingly entropic passwords for many accounts is becoming an increasingly difficult challenge. Password managers presently exist to solve these issues; however, the login process can be tedious if the application or website that is to be logged into is on a device that has not been set up with a user's password manager and account database.

Requirements Elicitation

Use Cases

Use Case ID:	UC-1-AA
Use Case Name:	Add account
Created By: Rey Jairus Marasigan	Last Updated By: Rey Jairus Marasigan
Date Created: February 21, 2022	Last Revision Date: February 21, 2022
Actors:	• User
Description:	This use case is used for specifying the different elements of our system that the DeviceUser goes through to add an entry to a list of account entries within our app. The ideal outcome of this use case is that the account is added to the database without any adverse effects.
Trigger:	The user presses the plus icon on the top right of the UI.
Preconditions:	• The User has created a database • The User has opened the app and unlocked their database • The User is in the correct directory to be able to see and touch the plus icon.
Postconditions:	• At minimum: the system retains consistency and integrity in case of an error or deviation from the flow of events i.e. the user can “add account” again without repercussions from previous try. • Everything in harmony and works without error: Account is added to the database. • The account added can be accessed after being saved to the database.

Normal Flow:	(assumes the user wants to generate password when adding the account) 1. The DeviceUser presses the plus icon on the top right 2. The view switches to a display that shows 3 text fields: a website field, a username and a password field. The first two will be required to be filled using the pop-up keyboard. The password field can either be manually filled or automatically generated using the associated button located to its left. <generate password use case> 3. The user presses a button that opens to another view that shows multiple fields, boxes, and sliders to be filled, checked, and adjusted, respectively, to meet a certain criteria associated with the password. 4. The user enters the field with a pop-up keyboard on their phone. 5. Below the fields, the users checks various boxes or switches that configures the password generated by the apps such as: • contains special characters- an option that includes special characters in the password • contains numbers- an option that includes number • etc. 6. Below the switches, there exists a slider that controls how long the password will be. When it is adjusted towards the right, it increases the length. When it is adjusted towards the left, it decreases the length. The maximum length of the slider occupies the entire width of the screen with some deadzone. The minimum length is 8 characters and the maximum is 32 characters. Initially the slider is set to 12. 7. As the user changes the length of the desired password, a text box that shows the password itself changes. The password is shown in its pure form i.e. the actual password to be used. Here, the password is changeable by pressing on the box and using the pop-up keyboard. Initially, the password generated here is from the combination of the slider and checkboxes. 8. The user then presses the "save password" button below to generate the password. <generate password use case> 9. The user finally then presses the “add account” button below to add the account to the list. 10. A notification pops up that lets the user know that the account has been added. 11. The state of the app then returns to its default view.

Alternative Flows:	2b. In step 2 of the normal flow, if the user chooses to manually enter the password rather than generate a new password, 1. The entire normal flow can skip to step 9 7b. In step 7 of the normal flow, if the user changes the generated password and then slides the password slider, 1. An entirely different password is generated. 2. The new password is instantly displayed within the text box. 3. The normal use case continues thereafter on step 7 where the user can edit the newly generated password. 9b. In step 9 of the normal flow, if the website field and username field is not filled 1. A notification pops up to let the user know about the required fields 2. The app is scrolled the field needed to be filled 3. The fields are highlighted red to indicate requirement. 4. The keyboard pops up automatically 5. Once the fields are filled, the use case enters step 8 of the normal flow.
Exceptions:	Exceptions to adding account to list 9b. In step 9 of the normal flow, if the user does not enter a website or username.
Includes:	editAccount- steps 3 to 7 would be required to edit an entry within the list and can be separated in its own use case
Frequency of Use:	On-demand
Special Requirements:	• Users should not wait >20 seconds for an action to be performed • Animations should last <250ms • Colors should be made to be accessible to colorblind • Application should be made available on all major mobile platforms
Assumptions:	• The Customer has installed and is using the application • The application is readable to the Customer • The client has the ability to use the keyboard and interact with the screen.

Notes and Issues:	• Is the minimum and maximum possible length of the password generator feasible for the users? • Should there be a limit to the amount of accounts added to the list? • What are the different switches that we can include in the forum? • should the app request for the master password (the password required to enter the app)or other biometrics for the ability to add an entry to the list? • Should we use other characters outside of the printable ascii characters?

Use Case ID:	UC-2-AB
Use Case Name:	Login to Database
Created By: Sonali Benni	Last Updated By: Sonali Benni
Date Created: February 22, 2022	Last Revision Date: February 22, 2022
Actors:	• DeviceUser
Description:	This use case is for logging into the app to access passwords. Ideally, the actor would have the app on their phone. They would log into the app and have access to their home screen/ main menu.
Trigger:	The DeviceUser clicks on the app on their personal device.
Preconditions:	• The DeviceUser is logged in on their personal device such as a cellphone where the app is located. • The DeviceUser opens the app
Postconditions:	• The DeviceUser is able to get into the app and is brought to a screen that allows them to enter login information. • The DeviceUser has successfully entered their information and is brought to a screen that shows their home screen.
Normal Flow:	• The user turns on their personal device and is logged in. • The user then locates the app on their device. • The user clicks on the app. • The app then opens and displays a login screen. • The user then enters a username and password into the app. • The information is successfully processed and the user can see their home screen. • From this point, the user is free to update, add, or delete any accounts.

Alternative Flows:	6a. In step 6 of the normal flow, the login information is not successfully processed and the user can't see their home screen. 1. The app will prompt the user with an error, try again message 2. Once the information is correct and processed, the user is able to log in 3. Use Case resumes on step 6 7a. Once the user is successfully in the app, if the personal device were to turn off/shut down/force closed: 1. The user would be automatically signed out of the app. 2. Use Case resumes on step 4
Exceptions:	4a. In step 4 of the normal flow, the login screen is not displaying 1. If the app is down for maintenance or needs to be updated, a message will be displayed when the app is clicked on. 5a. The user forgets their password 1. The user hits forget password 2. Then a recovery key could be entered or security questions could be answered to login 3. User is prompted to create a new password
Includes:	Included in any Login use case scenarios
Frequency of Use:	On-demand: Depends on how many accounts the user needs to login into on that particular day. This use case will most likely be used multiple times a day.
Special Requirements:	The app should be organized and users should be able to quickly find what they are looking for. Color themes matter, color affects readability.
Assumptions:	• The user understands English and knows their username and password for the app.
Notes and Issues:	• Should the app have the ability to use Face ID or any other form of biometric authentication?

Use Case ID:	UC-2-AB
Use Case Name:	Pair Dongle
Created By: Gentman Tan	Last Updated By: Gentman Tan
Date Created: February 22, 2022	Last Revision Date: February 22, 2022

Actors:	• DeviceUser • Mobile Device • PassUSB
Description:	Initialize a secure connection between the DeviceUser's MobileDevice to the PassUSB for future device communication.
Trigger:	The DeviceUser initializes the pairing subroutine
Preconditions:	• The DeviceUser is logged in on their personal device such as a cellphone where the app is located • The DeviceUser has the app opened • The DeviceUser has created a database • The DeviceUser has logged into a database
Postconditions:	• The DeviceUser has successfully paired the PassUSB dongle to their database
Normal Flow:	• The DeviceUser selects the "Pair Dongle" option in the drop down menu • AccountHolder is prompted to enable bluetooth discovery mode • The app searches for a PassUSB dongle • Once found, the Bluetooth dongle's name, serial number and PIN number is displayed • AccountHolder is prompted to either allow or decline the pairing process • Once the AccoutHolder presses "Allow", they are prompted to enter the PIN number displayed on the PassUSB • Once the PIN number is entered and confirmed, the devices are subsequently paired
Alternative Flows:	1a. In step 1 of the normal flow, a Bluetooth dongle is already paired 1. The app will prompt the user to either unpair the already paired dongle or cancel the pairing process 2. If the unpair button is pressed, the currently paired dongle is unpaired and PairDongle resumes on step 2 3. If the cancel button is pressed, the pairing dialog box is closed and the Pair-Dongle use case exits

Exceptions:	3a. In step 3 of the normal flow, a Bluetooth dongle is not found 1. The app will prompt the user to either retry or cancel the pairing process 2. If the retry button is selected, PairDongle resumes on step 3 3. If the cancel button is pressed, the pairing dialog box is closed and the PairDongle use case exits
Includes:	N/A
Frequency of Use:	On-demand
Special Requirements:	• The app should filter out other unrelated Bluetooth devices that are discovered
Assumptions:	• The DeviceUser understands English and knows their username and password for the app. • The DeviceUser has a funcitonal and powered PassUSB dongle
Notes and Issues:	• Should the user have the ability to pair multiple different PassUSB dongles simultaneously?

Use Case ID:	UC-3-AC
Use Case Name:	Generate Password
Created By: Kaiya Roberts	Last Updated By: Kaiya Roberts
Date Created: February 22, 2022	Last Revision Date: February 22, 2022
Actors:	• DeviceUser
Description:	The reason behind this use case is to prevent the user from having to re use the same password or a variation of one combination of password. The ideal outcome of this use case will be the creation of a unique password with the requested character amount, letter amount, number amount, and special character amount.
Trigger:	The user clicks on the create new password button on their personal device
Preconditions:	• The DeviceUser is logged in on their personal device such as a cellphone where the app is located • The DeviceUser has the app opened • The DeviceUser has created a database • The DeviceUser has logged into a database
Postconditions:	• The account holder has decided to not generate a password, but include their own password instead. Their own password will be stored in the account, and the account holder will be alerted that their password was stored • The password is generated with the conditions specifed by the accountHolder, and the accountHolder will be notified that the password was successfully created and it is stored into the system.

Normal Flow:	1. The user has added the desired account that they would like to generate a password for (mentioned in the Add Account use case). 2. The user has pressed the create a password button for the desired account. 3. The user device switches to a view where they have the option to click an eye icon that will show the password as it generates. 4. On the same view mentioned above the user has the option to choose between certain parameters that they can adjust based on the parameters of the password they would like to generate. 5. The user clicks the parameter they would like to add to the password. The examples are provided below. • length of password • upperCase letters • lowerCase letters • digits • Minus • underscore • space • special characters • bracket variations 6. The user adjusts the desired length of password by moving the slider icon to their desired amount. • The slider starts at 12 characters and it can be adjusted to another number. • The maximum number of characters in the password is 32. • When the slider is moved towards the right it increases the desired number • When the slider moves towards the left it decreases the desired number 7. As the parameters are updated, the user will see the changes in the password box where the new password will be generated. 8. The user updates the password with the desired parameters and has created a final password 9. The user presses save password 10. The password is saved into the application's database

Alternative Flows:	• At any time, the user can cancel the Generate Password dialog box and exit the Generate Password use case
Exceptions:	N/A
Includes:	N/A
Frequency of Use:	On-demand
Special Requirements:	• Options chosen should reflect immediately in the password (the DeviceUser should not need to wait >500ms for a password to be generated)
Assumptions:	• The DeviceUser understands English and knows their username and password for the app • The DeviceUser is logged into a database • The DeviceUser has elected to either add a new account entry or edit an existing account entry
Notes and Issues:	• Which kinds of cryptographic algorithms should be used in the process of password generation? • Should passwords be shown in plaintext to the DeviceUser by default?

Use Case ID:	UC-5-AC
Use Case Name:	Delete Account
Created By: Chidiebere Otuonye	Last Updated By: Chidiebere Otuonye
Date Created: February 21, 2022	Last Revision Date: February 22, 2022
Actors:	• DeviceUser
Description:	This use case is for outlining the behavior of DeviceUser when they're deleting an account from the USB dongle, via the app.
Trigger:	The user presses the "edit" button.
Preconditions:	• The DeviceUser is logged in on their personal device such as a cellphone where the app is located • The DeviceUser has the app opened • The DeviceUser has created a database • The DeviceUser has logged into a database
Postconditions:	• The system retains all data from accounts that user intends to keep and integrity of control flow is preserved • The correct account is deleted and the other accounts are preserved, control flow is preserved and the user can execute other actions and navigate to other parts of app

Normal Flow:	1. The accountHolder presses the 'edit' icon. 2. Selection bubbles appear next to all accounts. 3. The accountholder can select multiple accounts or just 1. 4. The accountholder selects the trash icon in the top right-hand corner. 5. A dialogue box pops up asking if user is sure they want to delete selected accounts. 6. App authenticates identity before deleting account.

Alternative Flows:	3a. In step 3 the user decides not to delete any accounts after pressing 'edit'. 1. User selects 'cancel' in top left-hand corner where the 'edit' button previously was. 2. Selection bubbles disappear from next to account fields. 3b. In step 3 the user decides not to delete any accounts after pressing 'edit' and selecting account(s). 1. User selects 'cancel' in top left-hand corner where the 'edit' button previously was. 2. Selection bubbles disappear from next to account fields. 3c. In step 3 the user decides to delete all accounts after pressing 'edit' button. 1. User presses the 'select all' button. 2. User presses the trash icon. 3. User is prompted to confirm if they're sure they want to delete 'ALL accounts'. 4. User is prompted to authenticate using physical password. 5. User is then prompted once more if they'd like to delete 'ALL accounts as the action is irreversible' 6. User selects 'delete' and all accounts are deleted. 5a. In step 5 user decides not to delete accounts after selecting the trash icon. 1. User selects the 'Never mind' option when presented with the options to delete or not delete selected accounts.

Exceptions:	6a. In step 6 user isn't able or chooses not to authenticate their identity to complete deletion process. 1. User fails authentication and is prompted to authenticate biometrically 2 additional times. 2. User is then asked to enter physical password. 3. User enters physical password incorrectly 3 times and is locked out of account for 30 minutes. 4. User enters physical password incorrectly 1 more time and is locked out for 45 minutes. 5. User enters physical password incorrectly 1 more time and is locked out for 1 hour. 6. User enters physical password incorrectly 1 more time and is locked out permanently until they recover account. N/A
Includes:	N/A
Frequency of Use:	On-demand
Special Requirements:	• Customer can organize accounts with different methods to make deleting quicker.
Assumptions:	• The user is able to use the application without issue on their device(s). • The user understands which button to choose to select and delete accounts
Notes and Issues:	• Should account objects that are to be deleted undergo lazy deletion or completely dereferenced?