Criterion B: Design

Figure 1 shows a general overview of the program as a whole. Each of the predefined processes in the Figure 1 are then shown further in separate flowcharts.

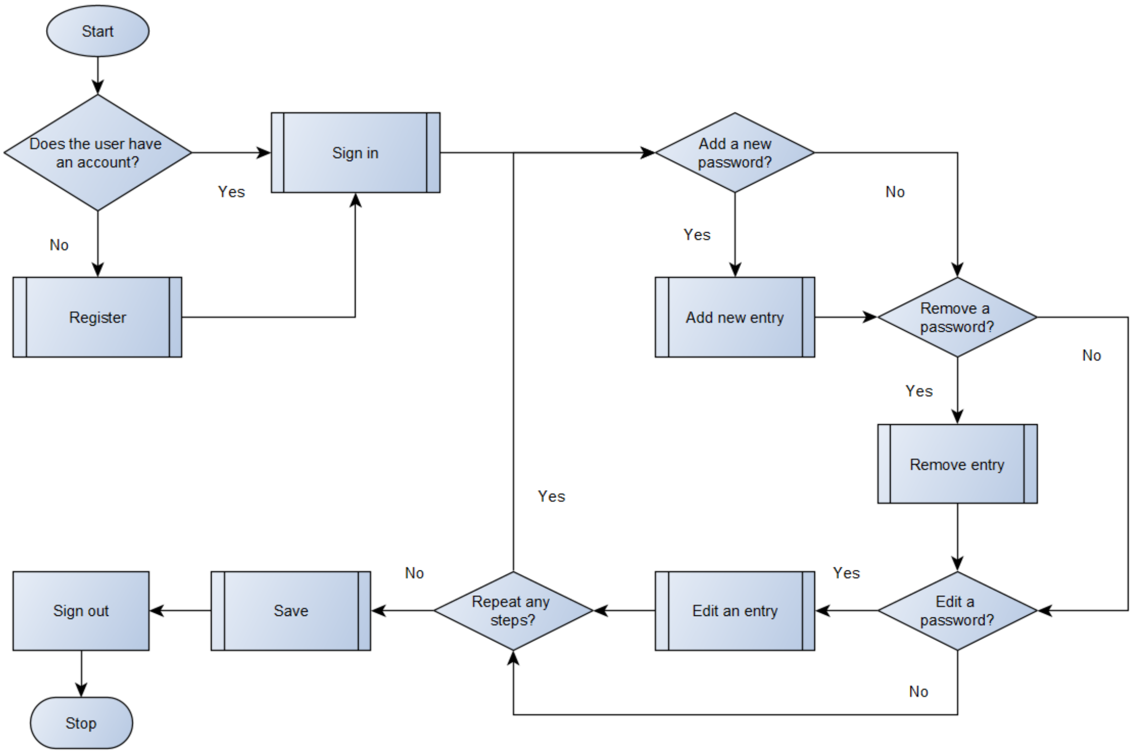


Figure 1: General overview flowchart

When the user starts using the application, they either sign in or register, depending on whether they already have an account or not. If they need to register, they must then sign in once they have created an account.

Once the user is authenticated, they can view their passwords for all websites, and can add new passwords, remove passwords, or edit passwords. Once the user is finished, the passwords are saved and the user signs out.

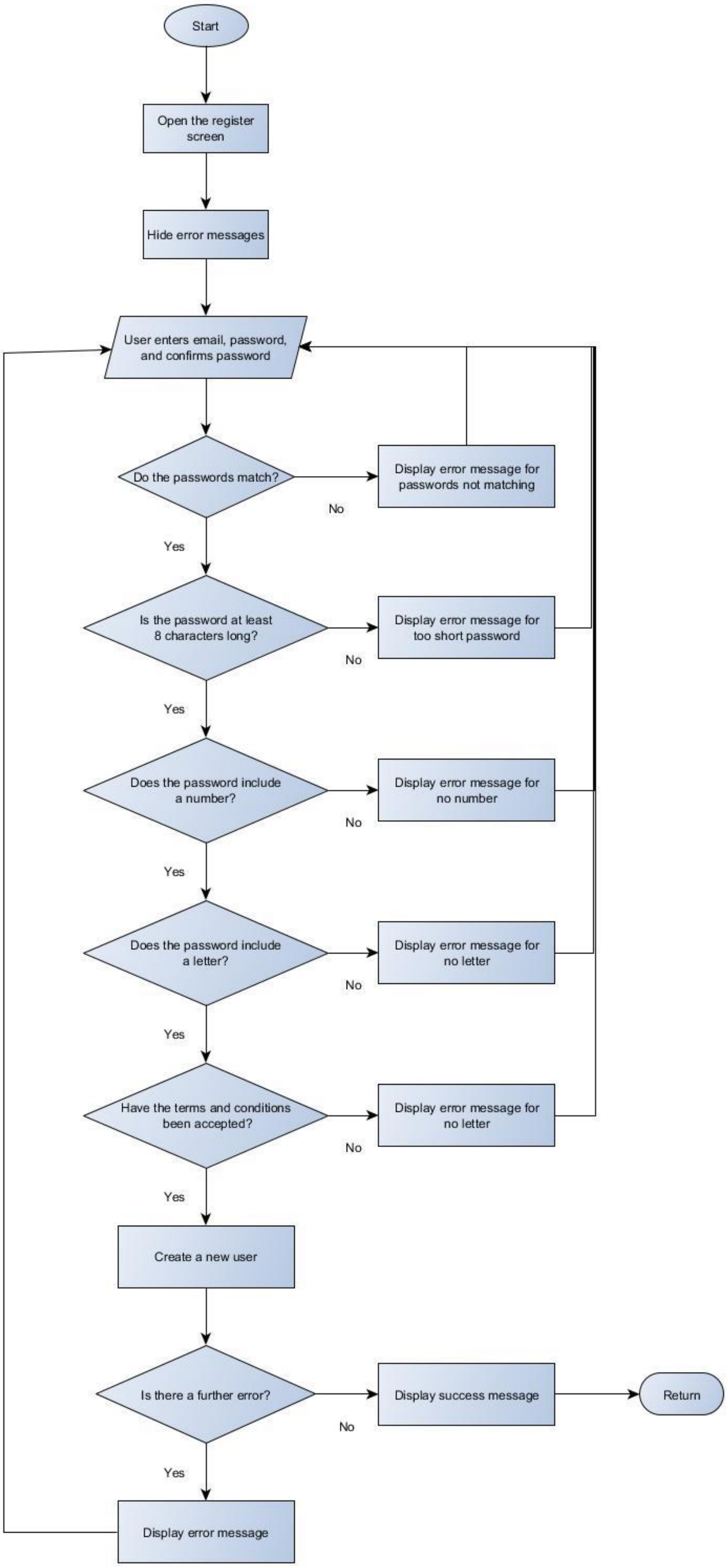

Start
Open the register screen
Hide error messages
User enters email, password, and confirms password
Do the passwords match?
No
Display error message for passwords not matching
Yes
Is the password at least 8 characters long?
No
Display error message for too short password
Yes
Does the password include a number?
No
Display error message for no number
Yes
Does the password include a letter?
No
Display error message for no letter
Yes
Have the terms and conditions been accepted?
No
Display error message for no letter
Yes
Create a new user
Is there a further error?
No
Display success message
Return
Yes
Display error message

Figure 2: Register sub-process flowchart

In the register sub-process, the register screen is first opened and all error messages are hidden. The user enters new user details and these are checked for validity. If any of the checks fail, an error message is displayed and the user has to input new user details. Otherwise a new account is created and a success message is displayed.

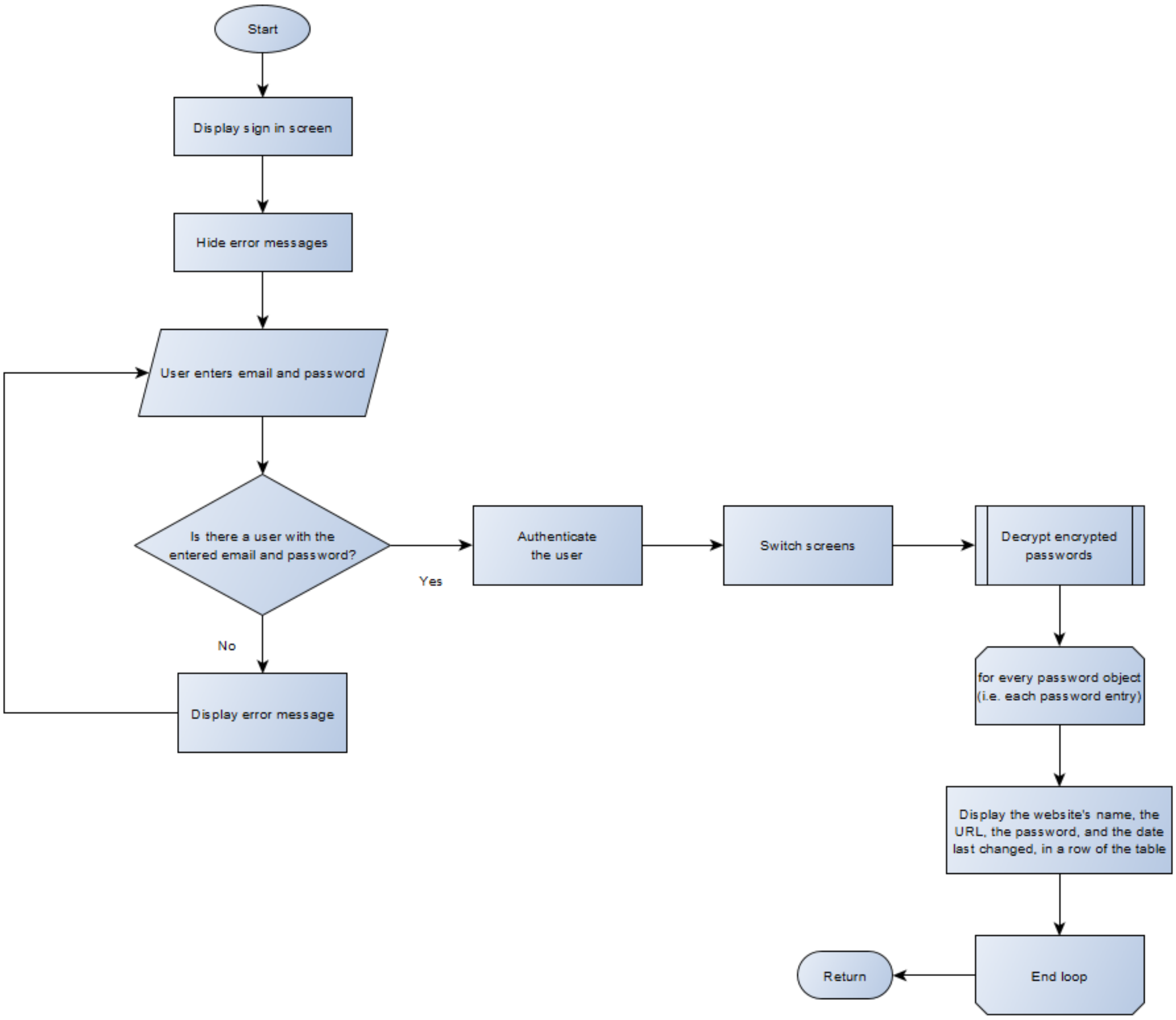

Start
Display sign in screen
Hide error messages
User enters email and password
Is there a user with the entered email and password?
Yes
No
Authenticate the user
Switch screens
Decrypt encrypted passwords
for every password object (i.e. each password entry)
Display the website's name, the URL, the password, and the date last changed, in a row of the table
Display error message
End loop
Return

Figure 3: Sign in sub-process flowchart

In the sign in sub-process, the sign in screen is first displayed, and error messages hidden. The user enters their credentials. If such a user exists, the user is authenticated and taken to a new screen. The user's password list is decrypted, and the passwords and corresponding websites are displayed on the new screen.

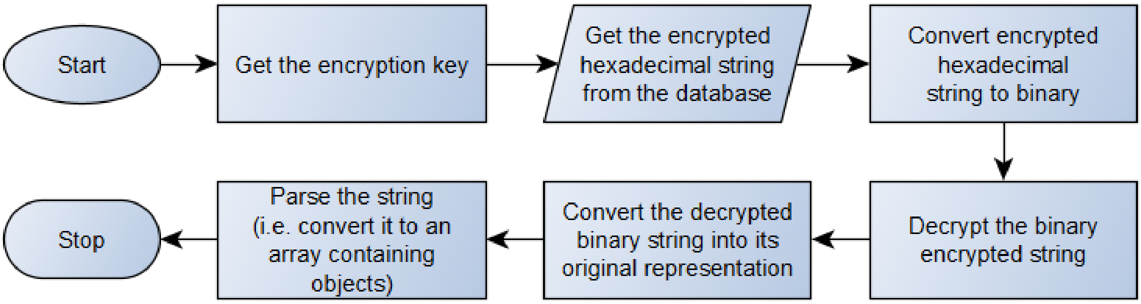

Start
Get the encryption key
Get the encrypted hexadecimal string from the database
Convert encrypted hexadecimal string to binary
Decrypt the binary encrypted string
Convert the decrypted binary string into its original representation
Parse the string (i.e. convert it to an array containing objects)
Stop

Figure 4: Decrypt encrypted passwords sub-process flowchart

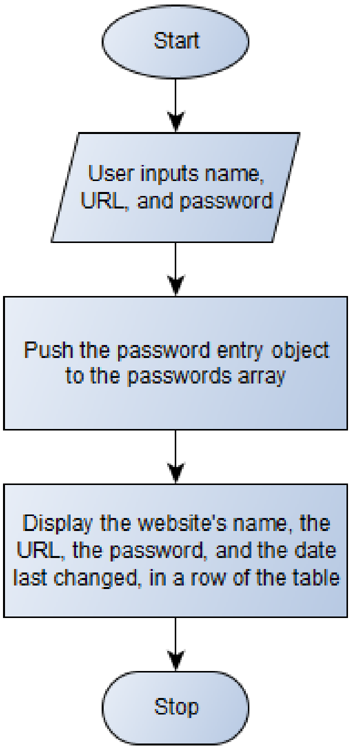

Start
User inputs name, URL, and password
Push the password entry object to the passwords array
Display the website's name, the URL, the password, and the date last changed, in a row of the table
Stop

Figure 5: Add new entry sub-process flowchart

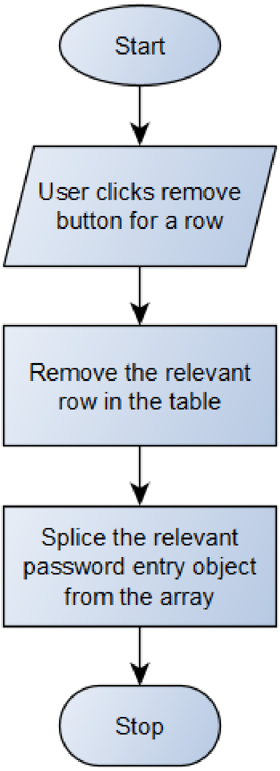

Start
User clicks remove button for a row
Remove the relevant row in the table
Splice the relevant password entry object from the array
Stop

Figure 6: Remove entry sub-process flowchart

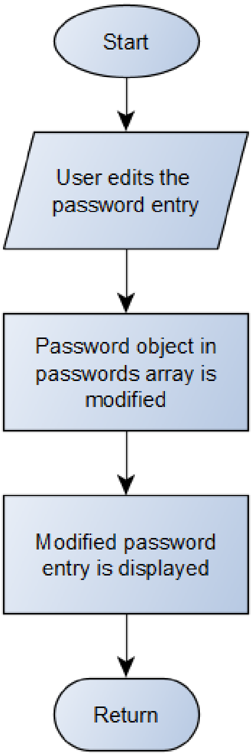

Start
User edits the password entry
Password object in passwords array is modified
Modified password entry is displayed
Return

Figure 7: Edit entry sub-process flowchart

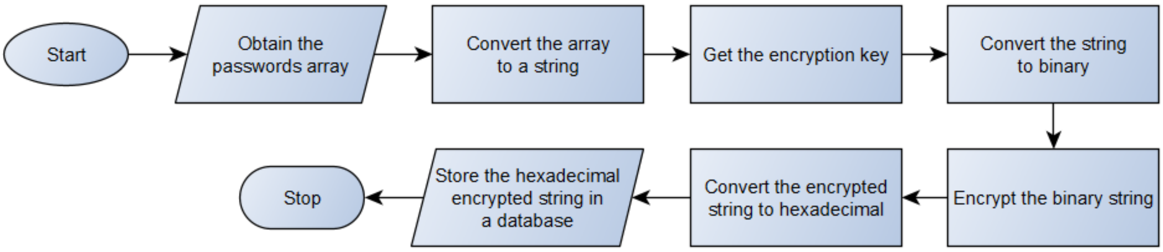

Start
Obtain the passwords array
Convert the array to a string
Get the encryption key
Convert the string to binary
Encrypt the binary string
Convert the encrypted string to hexadecimal
Store the hexadecimal encrypted string in a database
Stop

Figure 8: Save and encrypt sub-process flowchart

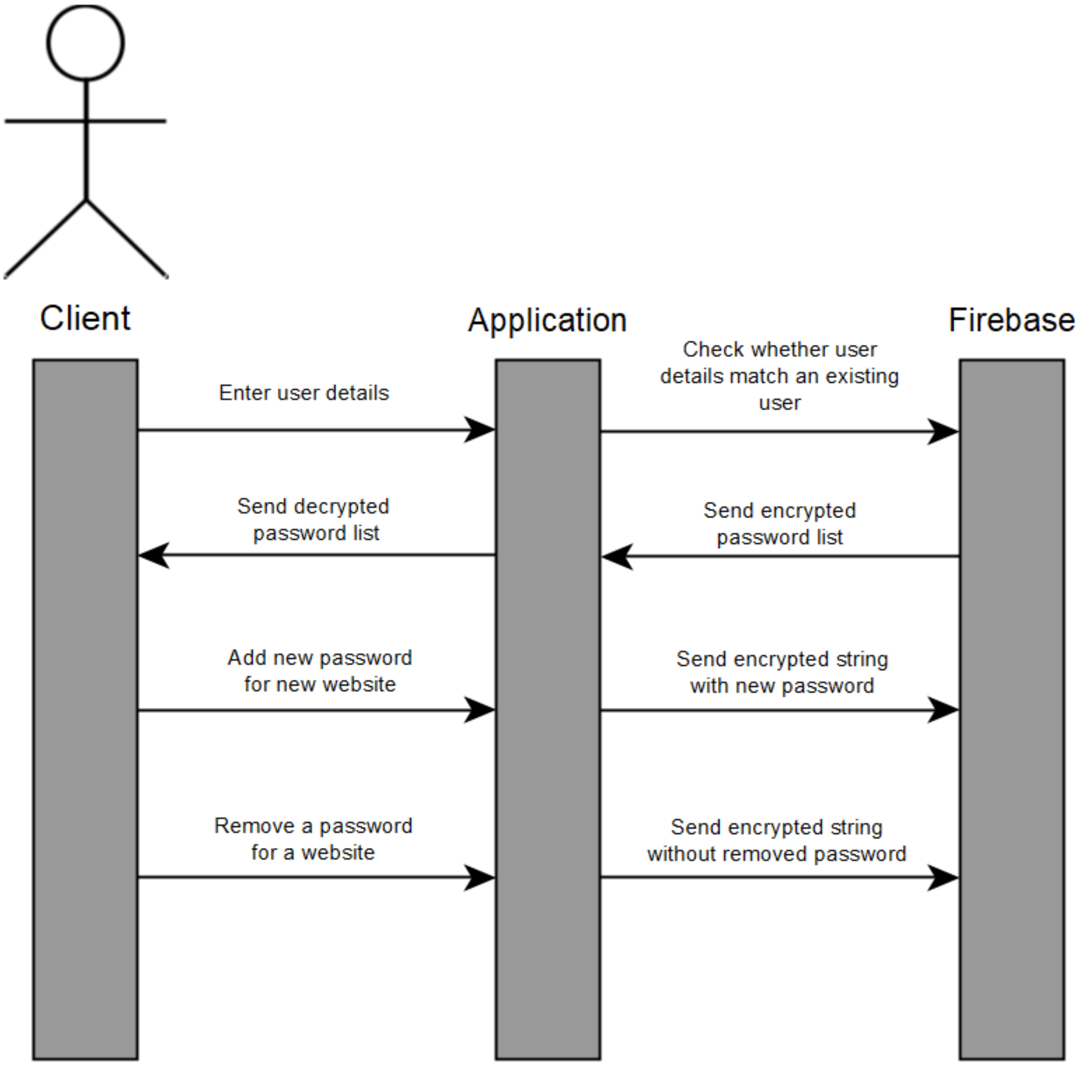
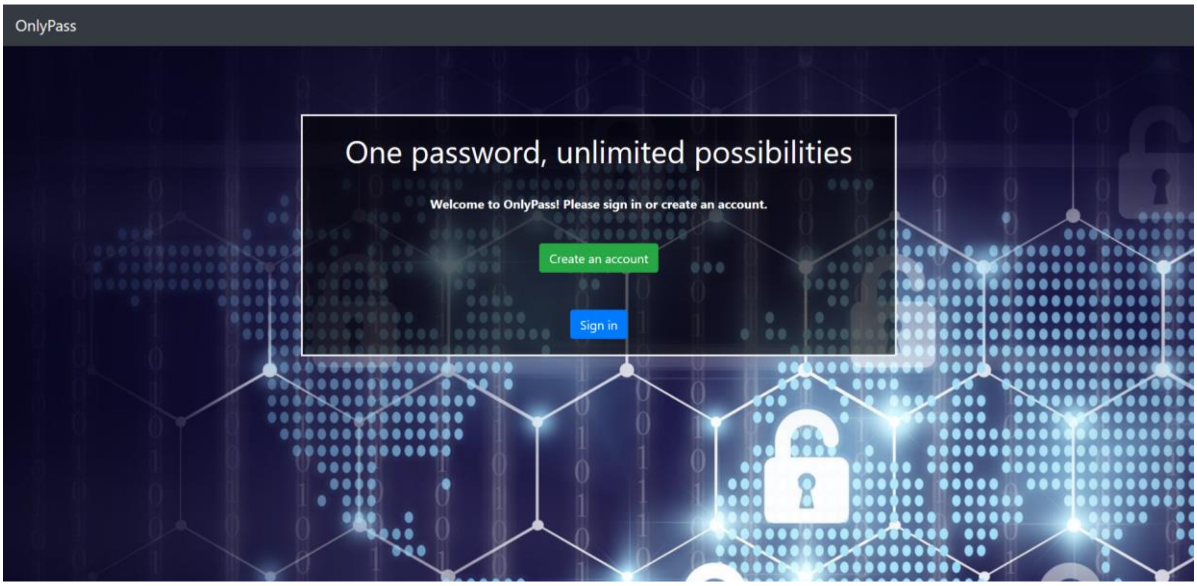

Client
Application
Firebase
Enter user details
Check whether user details match an existing user
Send decrypted password list
Send encrypted password list
Add new password for new website
Send encrypted string with new password
Remove a password for a website
Send encrypted string without removed password

Figure 9: UML diagram

Firebase will be used for user authentication and storing the encrypted strings. Firebase thus acts as the server in the program. The UML diagram shows some of the information flow between the client, the front-end application, and back-end Firebase.

For example, when the client wishes to sign in, he enters his user details. The application then sends a request to Firebase to check whether there is a user with the entered credentials. If so, Firebase finds the relevant encrypted password list, and send it to the front-end application. The application then decrypts the list, and displays it to the client.

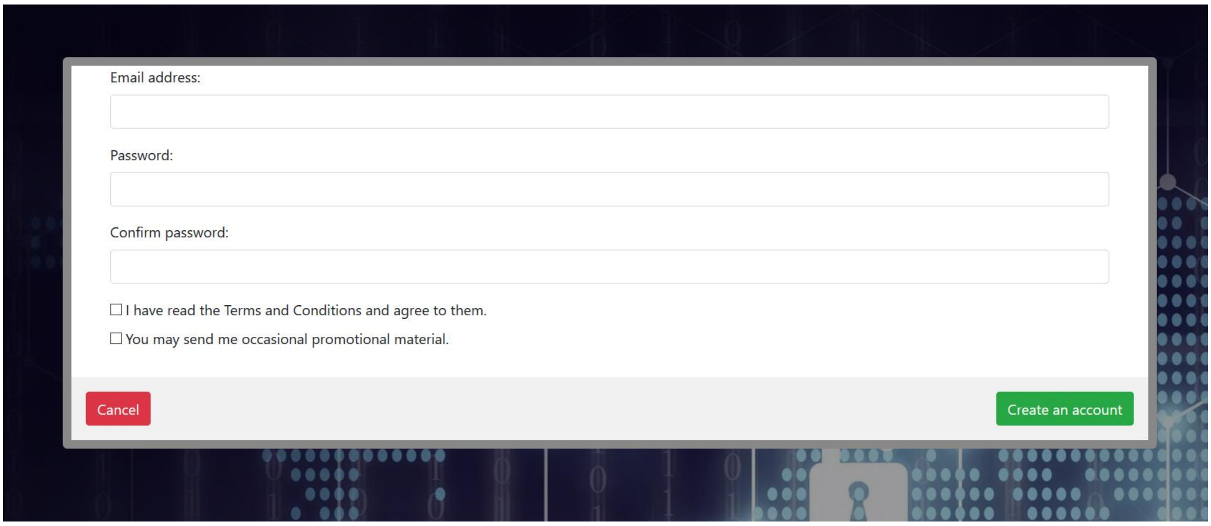
Should the client wish to add or remove a password, this likewise first goes through the application, which modifies the password list and encrypts it. The modified encrypted list then gets sent to Firebase to be stored. It is important that only encrypted information ever gets sent to Firebase, so that the security of the program is not compromised.



One password, unlimited possibilities

Welcome to OnlyPass! Please sign in or create an account.

Figure 10: Starting screen



Email address:

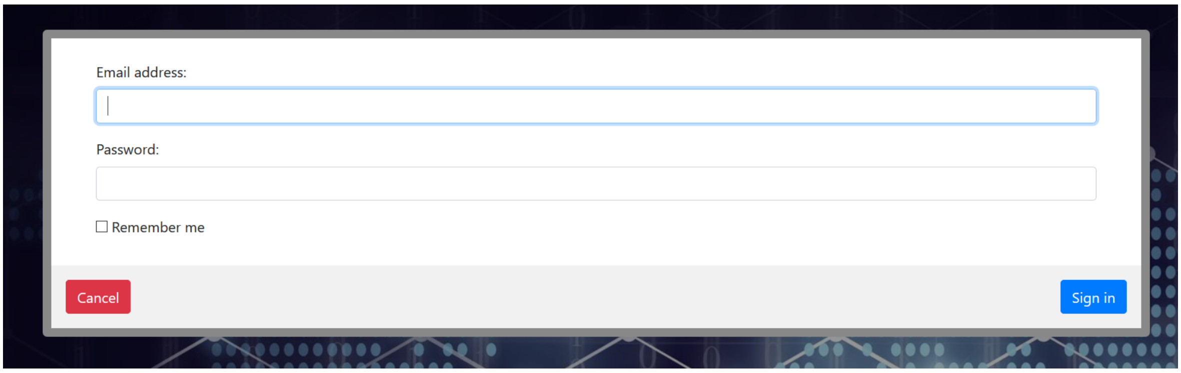
Password:

Confirm password:

□ I have read the Terms and Conditions and agree to them.

□ You may send me occasional promotional material.

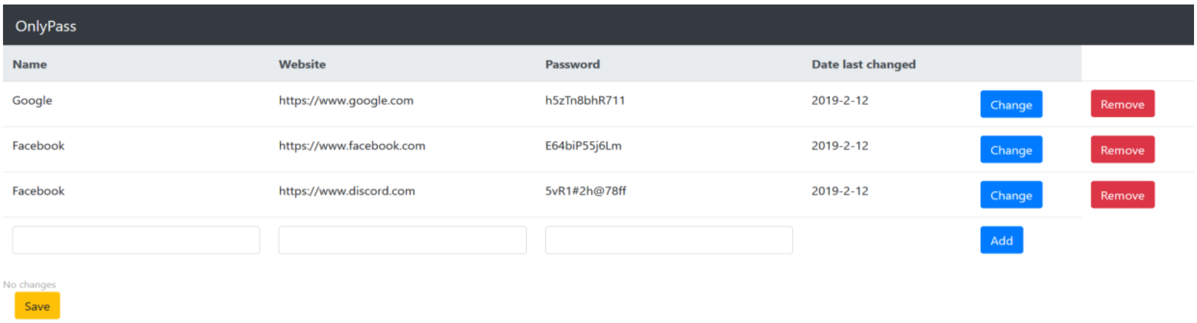
Figure 11: Create an account screen


Email address:

Password:

☐ Remember me

Figure 12: Register screen


OnlyPass

Name	Website	Password	Date last changed
Google	https://www.google.com	h5zTn8bhR711	2019-2-12
Facebook	https://www.facebook.com	E64biP55j6Lm	2019-2-12
Facebook	https://www.discord.com	5vR1#2h@78ff	2019-2-12

No changes

Figure 13: Password list screen

Test Plan

Action to be tested	**Test method**
The client is able to create an account	Attempt to use the register function multiple times with various different inputs, attempting to see whether the system doesn't break down no matter what combination of inputs the client enters.
The client is prevented from choosing an unsecure passphrase	Attempt to choose unsecure passphrases and see whether the program will prevent the user from doing so and throw an error. Attempt to use a passphrase shorter than 8 characters, without a letter and without a number.
Adding passwords for new websites	Attempt to use the add function and add passwords for new websites.

Removing websites	Attempt to use the remove function and remove websites.
The passwords are securely encrypted	Check whether the program produces an unrecognizable hexadecimal string when the passwords are encrypted.
The client finds the product easy to use and user-friendly	Give the product to the client to try out for a few days, and listen to feedback on whether the client found it easy to use.
The program can sync passwords across multiple devices	Add a password for a new website on one device, and then log in on another device and try to obtain the password.
The program's inbuilt password generator functions properly	Attempt to use the inbuilt password generator, and see whether it produces a password at all, and whether that password is sufficiently secure.
Two-factor authentication	Enable two-factor authentication and attempt to log in using it.