Computer Science Department

Capstone Project Proposal

**webb**

Mitch Rodriguez

Course Number and Course Title: CSC 520

Presentation Date: December 14th, 2020

Project Supervisor: Professor Kentros

webb

webb is a data management web application that has a focus on security. The app allows the customer to be able to import and export private files and information and be able to set encrypted passwords on any imported files with an algorithm that’s not only secure but very hard to break. New accounts created in the application will have strict password requirements, security questions for verification, and a two-factor authentication system for whenever an account is created, or when a user attempts to login at a new IP. The app will also feature something called History, which allows the logged in user to view what changes have been made to and within the account, including login attempts if someone tries to break into the account. webb gives the customer the peace of mind knowing that they are the only ones viewing the information imported within the app. Managing data with security and confidence.

Problem Specification: Whenever a user creates an account for most online platforms, password requirements aren’t very strict, and security preferences for an account aren’t enforced by the platform. Most people aren’t actively looking to make their account more secure, and it’s not because they don’t know how to or don’t care, its partially because they don’t understand what can happen once a hacker breaks into a one of their accounts. As someone who has been working in a customer facing environment for over five years, and a large portion of that time spent is troubleshooting, and assisting customers in resolving issues they have, there’s something that’s abundantly clear that is a similarity shared with hundreds of customers I’ve assisted. There passwords are either too simple and can be easily brute forced with very little computation resources to break their password, or they don’t know their passwords to their respective online platform. In some extreme cases, I have spoken to and helped customers who have had several accounts breached and believe there experiencing someone attempting to steal their identity. From seeing first-hand how bad a lousy password can cause a slew of problems when the user has his or her account breached, I not only have setup multiple thirty-character passwords for my own personal accounts, but I’ve applied these similar security mentalities with webb. A user can easily create an account with webb, however they will need secure account information to do. Password requirements for webb are set the way they are so that a user cannot create a short password, and fourteen seemed like a good number to help prevent. In just one day, an eight-character password can be cracked to break into a user’s account, whereas fourteen characters will be significantly harder to break and take more time. Although fourteen characters doesn’t completely secure a user’s information, the cap limit for a password to be created with webb is one hundred. A user will have plenty of flexibility to create a longer password if they feel like they need it. To make a user’s account as secure as possible, not only will the password be encrypted using a very complicated algorithm, but account creation also asks for the setup of two factor authentication and security questions. So even if a user has a simple fourteen-character password that has been cracked, two factor authentications will stop the hacker from continuing to break into the account. The user can make their account as secure as they want, while keeping even the minimum requirements more secure than most online platforms minimum account requirements. When a user creates and logs into webb, the application opens to its main page which allows the user to import, export, create a folder, set password encryption, check account preferences, and check History. webb allows a user to take a private document or set of documents, import them into webb and have peace of mind knowing their eyes are the only ones viewing that document. For exporting files, as long as the files don’t have a password set on them, a user will be able to export any file and download it. And to export a file that does have a password on it, the user will just have to enter their password to unlock the file, which then can be exported. Although webbs purpose is very simple, the account creation, verification process, and encryption method for passwords and files within the application make importing and exporting important information easy and safe.

webb is a data management web application that allows users to import, export, and secure documents, files, and photos with having an emphasis on security for both logging into the account and securing files by encrypting them using the Argon2 algorithm. The languages used to help create webb will be written in Python with the Django Framework, HTML and CSS. The front end and back end of webb will be managed and developed with the help of the framework, making development faster and more secure. All passwords, including account and files encrypted, will be secured using the Argon2 algorithm. Both accounts and individual files will be extremely secure, and a database will store users’ information and saved IP addresses for the two-factor authentication system.

A user can get started with webb by first signing up within the web application. Said user will enter their first and last name, their email, and a secure password. The password requirements are as follows: minimum of fourteen characters, maximum characters being one hundred, one capital letter required, one number, and one special character. Once this has been completed, the user will then be prompted to enter the code sent to their email address, which will the initiate the two-factor authentication system. Afterwards, the user will finally be prompted to setup their security questions, which will be for authentication if the user does not remember their passwords. These case sensitive responses will be for no less than three, to no greater than eight security questions. The purpose of having these higher end, high amount choices like for passwords and security questions is so that the user could make their account as secure as they possibly can. Once the user has properly created their account, the user will be in webb, and have access to all features such as import, export, and encrypt.

There are currently no prototypes for the project yet, but the list of deliverables, case diagram and use case checklist will be below.

Student Objectives:

* Learn different encryption methods and find which will be the most secure. As of right now that is Argon2.
* Become more familiar with web development and successfully implement the Django framework in making my capstone.
* Gain more experiences and work with databases and how it will be implemented into my project.
* Be able to use those databases and store customers information and stored IP address within them.
* Have a few accounts set up for webb, with different information and files stored within them.
* Properly implement a two-factor authentication system, with a code being sent to the customers email address, and the customer has sixty seconds to retrieve that code and enter it.
* Become more familiar with how AWS will be implemented into my project.
* Ability to build the application and focus on security, creating the strict requirements for account creation, and learn how Argon2 will encrypt the account password and any imported file password that is set.
* Build web. This is the biggest project I’ve worked on and I’ve never worked on something like this before. Become knowledgeable in Python/HTLM/CSS/Any other programming language I need to learn to create and finish my project.

List of deliverables:

* Two factor authentication system for new accounts and new log ins on different IP addresses.
* Strict requirements for account creation regarding password requirements and security question setup.
* User can import a file or photo within the web application and move the file to and from folders within the app.
* User can export a file or photo, assuming there is not password encryption set on the file.
* User can setup password encryption for files within the application that are already imported.
* Passwords for the account as well as the files with the encryption set on them will be encrypted using the Argon2 algorithm.
* User should be able to log in to the app, open their account settings, and be able to change password, security questions, and amount of security questions, as well as use a different email address for the account, which will then prompt to setup the two-factor authentication system again.
* User will have the ability to access History, which will show any changes that have been made to the account while logged in, as well as if any log in attempt have been made for the account.

Use case checklist:

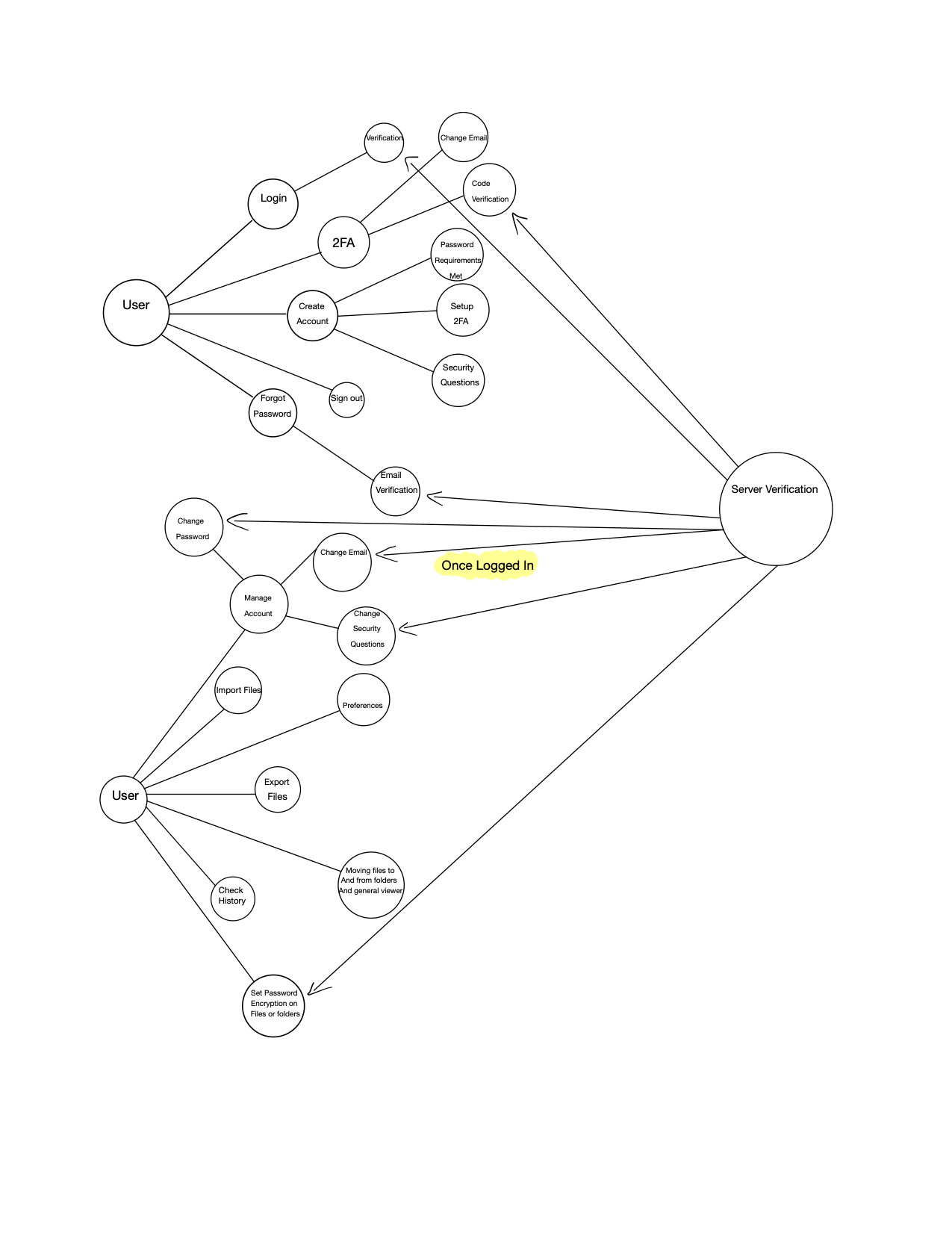
* Login splash screen will consist of two fields, email, and password, with create account and forgot password link on page.
  + Log in successful and user is entered into account.
  + Login is not successful, prompting “incorrect email or password, please try again”.
    - In the rare case that an account has four unsuccessful logins with a registered email, the account will be locked temporarily due to multiple failed attempts.
      * Account lock timers are as follows: 15 minutes for first failed attempt, 30 minutes for second failed attempts, 60 minutes for third failed attempt, and for the final attempt/any attempts after third failed attempt until log in is successful, 180 minutes. After the first failed log in timer, user can attempt password entry twice until next timer and lock engages.
  + Create account
    - Enter email/password (minimum of twelve characters with one number, one capital letter, and one special character required.
      * Entered account credentials do not meet requirements, prompt user to try again.
      * Entered account credentials meet requirements.
        + Prompt user for two factor code sent to entered email.

User enters code correctly and continues.

System prompts user to create three to eight security questions for verification.

* + - * + User enters code incorrectly, and either tries again or requests a new code.
        + User doesn’t enter code in time, new code is resent, and user must enter within sixty seconds.
  + Forgot password.
    - Prompt user to enter email address for account.
      * User enters email properly and is sent a six-digit verification code to enter within the application to allow for new password creation.
        + Code entered correctly, prompts user for new password to be entered twice with same requirements as log in.
        + Code entered incorrectly, prompt user to try again, or resend a code to email with maximum attempts being three.
      * Email entered by user that does not have an account associated with it will receive a prompt stating the entered email is not linked with an account, and prompt to try again.
* Once logged in.
  + Import files.
    - Successfully imported file.
  + Export files.
    - Successfully exported file.
    - Attempt to export file that is encrypted with password, “Please remove password encryption before exporting.”, prompt user to remove password encryption by asking for password to be entered.
  + Check history
    - Shows user what changes have been made to the account including imports/exports/etc as well as if the account had any attempted log ins.
  + Set password encryption on either files or folders.
    - User sets password encryption on individual files and folders containing multiple files.
  + Moving files to and from folders and general viewer
    - Successfully prompts for moving files.
  + Top left of system is systems menu, showing “Import”, “Export”, “History”, and “Preferences”.
  + If files that are encrypted need to be moves to and from folders or out of folders, the password encryption set on the file or folder needs to be entered before continuing.
  + Within preferences will be account settings, like “Change password”, “Edit security questions”, “Change email”, and possibly more.
    - Changing email; for this to be successful to be successful, two factor authentications will be setup again with new email.
    - Changing passwords will require the user to answer the security questions again, ensuring the user requesting password reset is the owner of the account.
    - Editing security questions will require the user to re-enter their password.
* Exit app.

Use case diagram:



ER Diagram/Solution Design:

webb

Contains

1

M

1

1

M

History

Export

Import

Customer

Account

1

PW Encryption

1

Creates

N

Has

1

N

Has

M

Preferences

Solution Design Cont.: Above is an ER diagram for my project, and I will start by going from the top entity down in terms of the responsibilities of different components. For the top entity, webb, is the general application. Webb contains all the functions above regarding importing and exporting, as well as the customers who use the application. Within the import entity is the “imFile” which is short for import file, which will be the function for importing files into webb. Within the export entity, the “exFile” is for the exporting of files within webb. For the history entity here, the five attributes within history are for specific changes that happen when a user is logged in and making changes to files, as well as attempts to sign into a user’s account. For “hImpo”, this will be the history for any files that will be imported into webb associated with a user’s account. As you can guess, “hExpo” is also the history for any files that have been exported from webb and downloaded by the user, and will also display when a export attempt has been made on a file that is encrypted by a password. For the attribute “hPWEChange”, this will be the function that tracks and displays within history when the password has been changed for the account the user is signed in with. Similarly, “hPWAttp” is a history of when the specific account that is logged in has had any attempts at trying to login. This is in place for transparency for the user so that they see the information quickly and respond quickly if they find that someone may be trying to access the account. Finally, for “hFChange”, this will be a tracked history for any files that have been moved, encrypted with a password, or decrypted with a password. This is another feature built for transparency to have a record of what changes the user is making to their sensitive data. The next entity to look at is PW Encryption, which encapsulates the encrypting and decrypting algorithm for files that have been imported, and folders created within webb. The “dCrypt” attribute will specifically be decrypting a file or folder that already has password encryption set on it. The “nCrypt” attribute is strictly for setting a password encryption on a specific file or folder. These attributes will be responsible for encrypting and decrypting files and folders that have been imported into webb. For the customer entity above, the customer has different features they can take advantage of when logged into their account. Above, the customer can create a specific folder within webb to contain files, which said folder can be encrypted with a password, or contain several files with set password encryption. Creating folders within webb will give the user the ability to keep their important information organized while also allowing the user to encrypt that folder if they feel they need to. For the customer entity, we can look at the account and what attributes are stored within it which are integral to keeping the account secure. For “cSecQues”, this is the customers security questions and answers they had provided and confirmed upon creating the account. This function will store the information securely, help verify a user when logging in, and to communicate with the “chSecQues” whenever the user is attempting to make account changes and will help verify that the entered information is what belongs to the account holder. Similarly, the “cPassWD”, “cEmail”, and “cUName” are respectively for when the user created their account and what they had entered for their password, email, and username. The password and email functions will be communicating with the respective functions within the preferences whenever a user is trying to make any account changes. Within the customer entity is the account preferences. The customer has their account and can manage different preferences within webb, which can help improve security of the account depending on the user’s information entered. Within the preferences entity is different methods of changing a user’s information with webb. For “chSecQues”, this is the attribute that will allow a user to change the number of security questions they may have enabled, while also changing any answers to those security questions. To do this though, a two-factor authentication code will be requested of the user to be entered, which will then verify that they are the owner of the account and allow the user to make the changes they desire. For “chPassWD”, this will be the attribute that allows the user to change the password for their account. To do this though, they will be requested to enter their current password, answer their security questions for their account, and be asked to enter a two-factor authentication code. The purpose of this is to ensure that the person changing the password is the owner of the account, and that the password is the most important thing to keep a user’s account secure. Once they have entered that information, they will be able to change their password assuming it meets the requirements. For “chEmail”, the customer will have the option to change their email associated with the account. To accomplish this, the user will be prompted to enter their password and security question answers to verify that the changes being made are by the account holder. Once the account information has been entered and a new email has been provided by the user to be made the account main email, a two-factor code will be sent to the users email address for verification. If the authentication code isn’t entered in time for verification, the original email will be reinstated as the account main email and the user will have a thirty-minute block to change the account main email address. The attribute “resAcct” will be an attribute in charge of allowing the user to be able to completely factory reset their account so that all information, files, passwords, etc. are to be erased from the account. This is in place in the rare case that the user either wants to remove all information from the account for specific reasons, or just does not have a need for the information stored within webb and wants to make sure that it is all completely erased. For the final attribute to go over, the “savedIPs” attribute will stored within the preferences and will save any IP addresses that the user will log in to. This will be used for when a user is attempting to log in, which webb will check if it is a saved IP. If it is not, they will be requested to enter their two-factor authenticator code, which will then allow them to log in.

Tools List: Currently for the tools list for my project, I will be using PyCharm for the IDE for Python and will be using the Django framework to assist in development and any HTML/CSS code that would need to be written. I may use ReactJS as well depending on how well development of the front end goes, cannot say for certain now since I haven’t started programming the project yet. I will be using AWS for the cloud storage portion of my project, and MySQL for the databases which will have integration with the framework.

Benchmark Specifications: Listed below is the measurable objectives I have for webb that will be completed once webb is finished.

* Customer can open webb, view the splash page with “Create account” or “Login”. Clicking on create account will display appropriate text boxes for a username, email, password. For login, the user will have two text fields, one being for email and another for password, and a “Forgot Password” clickable link to assist the user if there are issues logging in.
* Customer logins in/creates account and imports any file into webb.
* Customer can export that same file that had been imported previously.
* Customer can create a folder within webb, import a file successfully into that folder.
* Customer can set a password encryption on both the folder and the file.
* Customer can attempt to open the folder, which prompts for password to be entered. After, customer can attempt to export file, which should yield “Please remove the encrypted password from the file before exporting.”, then clicks “Return”.
* Customer can then remove both password encryptions from the file and the folder, and successfully export the file that was previously encrypted.
* Customer can access the preferences which can allows the user to change their email address, security questions, and passwords related to the account.
* Customer can check History, which will show that the user has imported a file, exported that file, created a folder, and imported the file and set password encryption on said file, display that password entry was correct, etc. History should be fully functional.

I am still working on my benchmark specifications, here is a place holder for where I am at.

Time Schedule: The current timeline I want to have for developing webb will be as follows.

* December 20th, 2020, I would like to have started my programming portion of this project. This will be the first time I will ever be creating a web application with databases and a cloud server, so there is going to be a lot of research when getting my project started.
* January 15th, 2021, I would like to have the databases architecture to be laid out for webb, in preparations for the data bindings of importing, exporting, saving IP’s linked to a user, etc.
* February 15th, 2021, webb’s front end should either be developed or nearly developed, in preparation for laying out the UI for the project, making development more enjoyable, and having full readiness for the rest of development
* April 1st, 2021, I would like to either be nearly finished with the backend of webb or finished, which allows for user authentication and authorization, and proper communication to the front end.
* April 20th, 2021, webb should have AWS successfully implemented, and importing and exporting files should be possible under two different accounts within webb.
* May 1st, 2021, webb should either be nearly completed or finished, with potentially some minor issues and bugs that need tending to after May 1st. webb should be ready for a user to create an account, login, import and export a file, logout, change account information; all operations of webb should be complete.

Grading Scheme and Conclusion: My grading scheme will be based off the components of my project listed above in the time schedule component. Considering that this will be the first time I will be creating a project of this scale for my final graduation assignment, I will be grading my project with the intention of having similarly equal components. For the first component of the grading scheme, database architecture, I believe the weight for this portion should be 10%. The databases will be used for saving certain information regarding a user’s account, and important functions like allowing the user to import or export their files, while also having webb save the IP addresses for the user when they log in. For both the front end, back end, and cloud server portions of the rest of the project, I believe it is reasonable to associate these sections of the project with 20% of a grade weight to each respectively. The front end and back end both have their own complexities for allowing a user to create an account, providing proper implementations for said user to have their file not only be imported into a functional cloud server, but to then have those files encrypted using an algorithm that has been proven incredibly hard to break. I have personally never worked with a cloud server, and so not only will it be new for me to learn but will also push my boundaries for what I am able to create for a project.

I am still working on the grading scheme and conclusion, here is a place holder for where I am at.

I am also working on finishing up my written portions earlier in the project, adding more to my use case checklist and list of deliverables. I hope the progress I have made so far is sufficient and lays good groundwork for my capstone.