Anonymoose Final Project Report

Jaclyn Bramlette, Jesstina Farrell, Elizabeth Porter, Zack Starrett

Capstone Project Spring, 2021

The University of West Florida

04/28/2021

CIS4592

Dr. Bernd Owsnicki-Klewe

Table of Contents

Executive Summary

This project was completed to be a password management application that also checks for reused passwords and generates random passwords. Currently, there are multiple password management options available that separately have these features, but none that combine them into one single user-friendly application. The goal of this project was to make it easier for users to follow secure password guidelines by creating a central location for their passwords, and to have a random password generator within the application so that users do not have to come up with a password that might be easily guessed. Since attackers may do research on victims before attempting to break passwords, commonly used passwords such as pet names, children’s names, and birthdates are not secure, as many people now post this information on social media.

Final Requirements/Comparison with Initial Requirements

The initial requirements for this project are demonstrated in the use cases below. All of the functionality requirements have been met with the outcome of our final project. A user can create a private account to store sensitive information, a user can securely store their private passwords, and a user can securely retrieve passwords.

Use Cases:

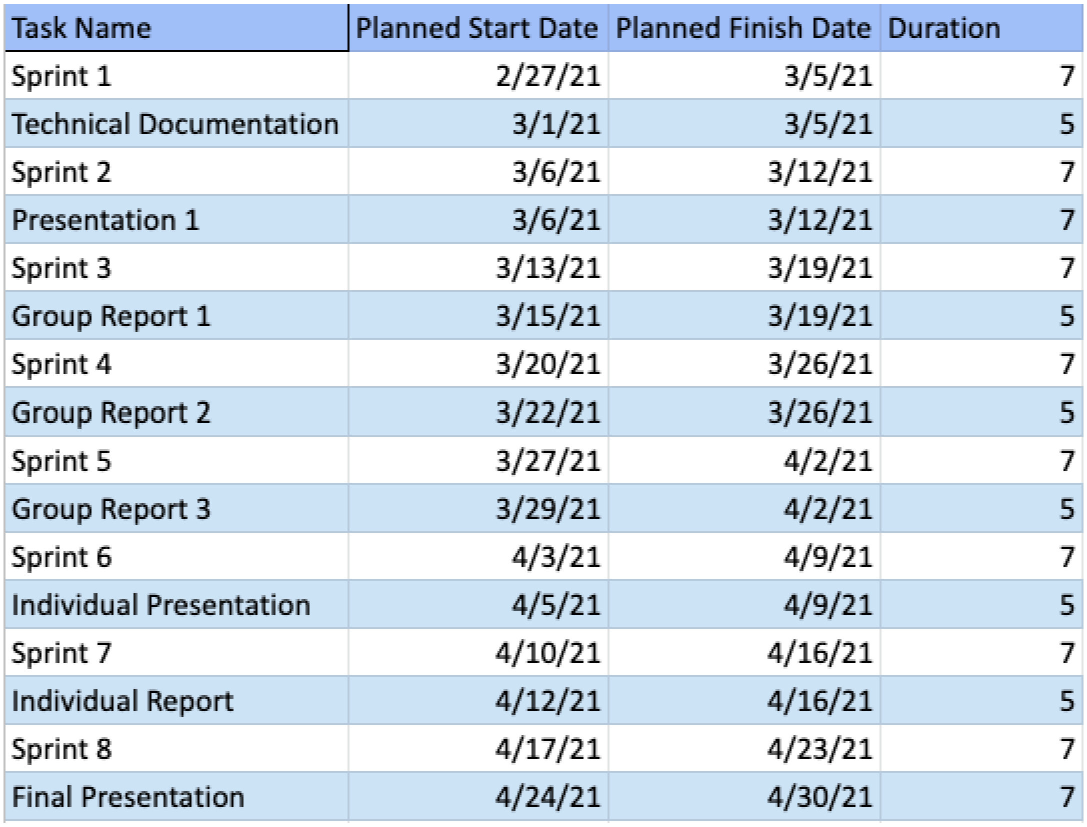
Use Case: Create an account for the application	
Users	All Users
Rationale	Users must be able to create a private account to store their sensitive information.
Preconditions	The user has downloaded the desktop application.
Course of Events	1. The user opens the desktop application 2. The user enters their email address 3. The user creates a new password to be used to access their private information. The user will be required to enter the same password twice as confirmation that it is correct.
Alternate Flow	1. The user enters an email address that has already been registered with the application. The application will inform the user that they must enter a different email address. 2. The user enters a different password for the password confirmation text box. The application will inform the user that they have not entered the same password for both text boxes.
Postconditions	The user has created their own account for the desktop application.

Use Case: Enter a password within the application for storage	
Users	All Users
Rationale	The user must be able to securely store their private passwords through the application.
Preconditions	The user has downloaded and created an account for the desktop application.
Course of Events	1. The user logs in to the application. 2. The user enters the website's name for which they wish to store a password. 3. The user enters the website's username and password.
Alternate Flow	1. The user enters a website that is already stored. The application informs the user and allows them to update their password for that website if needed. 2. The user enters a password that is already in use. The application informs the user and generates a random password to replace it.
Postconditions	The user has securely stored the website name, username, and password.

Use Case: Retrieve a password that was previously stored by the application	
Users	All Users
Rationale	The user must be able to securely retrieve passwords that have been stored by the application.
Preconditions	The user has downloaded the desktop application, created an account, and has stored at least one password.
Course of Events	1. The user logs in to the application. 2. The user searches in the search bar for the website they wish to retrieve the password for.
Alternate Flow	1. The user searches for a website that hasn't been stored. The application will inform the user and allow them to enter a new website.
Postconditions	The application has retrieved the user's password information and presented it to the user.

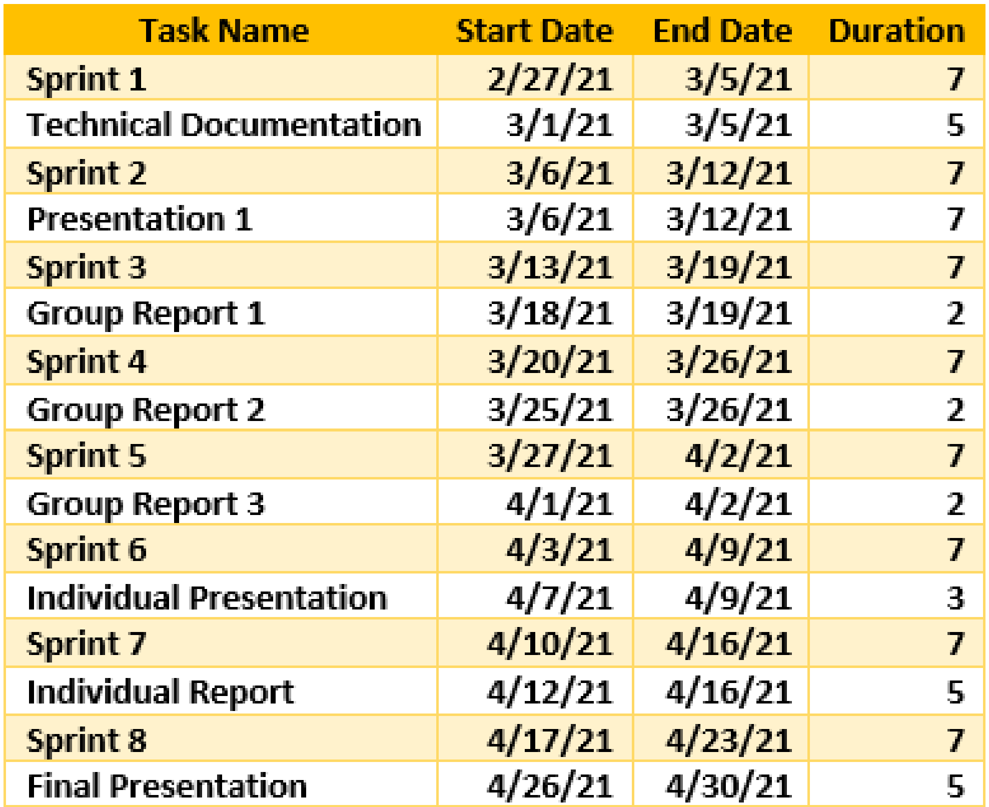
Final Timeline/Comparison with Initial Timeline

Initial Timeline



Task Name	Planned Start Date	Planned Finish Date	Duration
Sprint 1	2/27/21	3/5/21	7
Technical Documentation	3/1/21	3/5/21	5
Sprint 2	3/6/21	3/12/21	7
Presentation 1	3/6/21	3/12/21	7
Sprint 3	3/13/21	3/19/21	7
Group Report 1	3/15/21	3/19/21	5
Sprint 4	3/20/21	3/26/21	7
Group Report 2	3/22/21	3/26/21	5
Sprint 5	3/27/21	4/2/21	7
Group Report 3	3/29/21	4/2/21	5
Sprint 6	4/3/21	4/9/21	7
Individual Presentation	4/5/21	4/9/21	5
Sprint 7	4/10/21	4/16/21	7
Individual Report	4/12/21	4/16/21	5
Sprint 8	4/17/21	4/23/21	7
Final Presentation	4/24/21	4/30/21	7

Final Timeline



Task Name	Start Date	End Date	Duration
Sprint 1	2/27/21	3/5/21	7
Technical Documentation	3/1/21	3/5/21	5
Sprint 2	3/6/21	3/12/21	7
Presentation 1	3/6/21	3/12/21	7
Sprint 3	3/13/21	3/19/21	7
Group Report 1	3/18/21	3/19/21	2
Sprint 4	3/20/21	3/26/21	7
Group Report 2	3/25/21	3/26/21	2
Sprint 5	3/27/21	4/2/21	7
Group Report 3	4/1/21	4/2/21	2
Sprint 6	4/3/21	4/9/21	7
Individual Presentation	4/7/21	4/9/21	3
Sprint 7	4/10/21	4/16/21	7
Individual Report	4/12/21	4/16/21	5
Sprint 8	4/17/21	4/23/21	7
Final Presentation	4/26/21	4/30/21	5

As demonstrated in the above timelines, team Anonymoose kept to the schedule very well. The only days that changed were the start dates for the Group Reports because those took very little time to complete.

Project Results Compared with Expectations

All in all, team Anonymoose performed very well on this project, most of the expectations for this project were met. The only things that changed from our initial expectations throughout the course of creating this project were that we planned to develop this program to have a GUI; however, due to time constraints, that expectation was replaced with command-line functionality instead. Furthermore, initially we planned to upload the stored user data to a server, but changed that to using a Mongo Database instead as it seemed the more reasonable choice for that functionality.

Project Process Review

At the beginning of this project, we anticipated using an Agile development process with weekly team meetings and an iterative development cycle. Team Anonymoose ended up sticking to this process, and had weekly sprint deadlines as well as clear project deliverables. This process worked well for this project as the short deadlines helped keep the team focused. Furthermore, an Agile development process was beneficial as it provided much needed flexibility for our project and allowed for any unexpected changes.

Work To Be Done

As any good software developer knows, a programming project is never really done and can always be improved upon. With that being said, this program would benefit by adding a functionable GUI. Most users do not want a command line interface so having a visual interface is more desirable. Additionally, security could be improved by utilizing a private server to store user data; however this functionality was not quite within our means when developing this project. Furthermore, it would be ideal for this program to be made cross-platform and as a mobile application.