Universidad Creativa

Proyecto Final

UrPass Administrador de usuarios y contraseñas

Asignatura

Fundamentos de la Programación

Docente

Henry Sanchez

Estudiante

Andres Aguilar

Índice

Description del problema 3

Justification 3

Alcance 3

Objetivos 3

Solución 4

Funciones principales: 4

Implementación de la solución 5

Conclusiones 7

Bibliografia 8

# Description del problema

Para muchas personas el uso de diversas plataformas puede representar algunos retos y una de esas tareas puede ser el gestionar nuestras cuentas y contraseñas para esos sitios webs o aplicaciones. Por otro lado las personas tienden a usar contraseñas que no son nada seguras por lo que quedan vulnerables a personas mal intencionadas para evitar este problema y darle solución a este problema se crea esta pequeña app.

# Justification

Urpass nace de la necesidad practica de mantener nuestros datos mas seguros y lejos de esos con un fin no tan bondadoso. Dicha aplicación tiene varias fortalezas como analizar si el password de usuario es lo suficientemente fuerte o no ademas de generar si el mismo lo desea contraseñas que si lo sean y poder respaldar todos esos datos.

# Alcance

Ademas de estas funciones ya mencionadas UrPass podemos extender que esta app tendrá la opción de generar contraseñas de manera aleatoria basados en en multiples criterios para la construcción de la misma lo que hará que su complejidad proporcione seguridad, detalles como minúsculas, mayusculas, caracteres especiales e incluso la longitud del mismo.

Por otro lado la opción de recuperar contraseñas también cuenta con su seguridad ya que el mismo codigo esta diseñado para que se analice el sitio web y usuario para poder recuperar o cambiar el mismo por lo que todos los datos deben de coincidir en su caso contrario mostrara un mensaje de error.

En resumen el alcance de esta aplicación se extiende mucho mas allá de solo analizar y proporcionar contraseñas y todas estas cualidades harán que los usuarios se sientan seguros por las medidas utilizadas para salvaguardar esos datos tan preciados.

# Objetivos

* Analizar passwords actuales y proporcionar retroalimentación
* Generar contraseñas seguras y fuertes
* Proporcionar un medio seguro en el se almacenen las contraseñas
* Brindar un medio en el que se puedan recuperar contraseñas
* Brindar la alternativa de actualizar dichos datos cuando el usuario lo requiera

# Solución

La solución propuesta es un gestor de contraseñas que permite al usuario almacenar sus contraseñas y nombres de usuario de diferentes sitios web en un solo lugar específicamente un archivo .TXT. Está diseñado para ser una aplicación de línea de comandos, y se ha utilizado Python como lenguaje de programación.

Componentes utilizados:

pyfiglet: una librería que se usa para dar formato a un texto con la tipografía de arte ASCII.

random y string: para generar contraseñas aleatorias.

colorama: para imprimir en la consola en diferentes colores.

traceback: para registrar los errores y excepciones de la aplicación.

Estructuras de datos: se utiliza un diccionario llamado passwords para almacenar las contraseñas, nombres de usuario y sitios web.

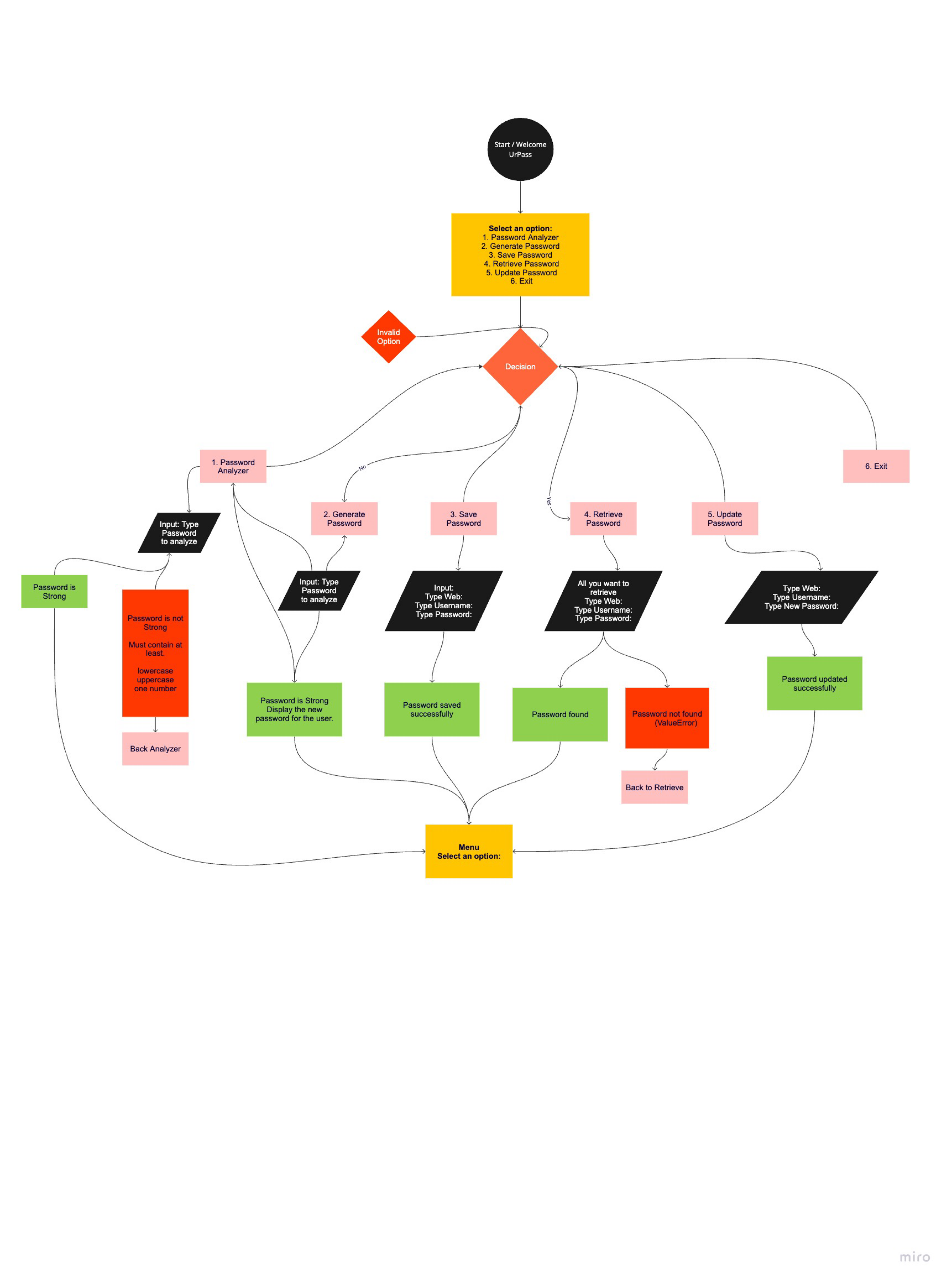
Existen otras librerías que fueron consideradas para este proyecto sin embargo no cumplieron su función.

# Funciones principales:

* menu(): es el menú principal de la aplicación y maneja las diferentes opciones que el usuario puede seleccionar.
* password\_analyzer(password): analiza la fortaleza de una contraseña ingresada por el usuario y verifica si cumple con ciertos criterios como longitud, uso de caracteres en mayúscula, minúscula, dígitos y caracteres especiales.
* generate\_password(): genera una contraseña aleatoria y la verifica con la función password\_analyzer().
* save\_password(app\_web, username, password): guarda las contraseñas y los nombres de usuario en un diccionario passwords y los almacena en un archivo llamado passwords.txt.
* retrieve\_password(app\_web): busca las contraseñas y los nombres de usuario almacenados en el archivo passwords.txt y los muestra en pantalla.
* update\_password(app\_web, password2, username): actualiza la contraseña del sitio web con el nombre de usuario correspondiente.

Diagrama de flujo para demostrar un poco de la usabilidad del mismo. (Fue elaborado en Miro)

<https://miro.com/app/board/uXjVMOutbsk=/?share_link_id=523208473873>



# Implementación de la solución

En este apartado como ya se había mencionado se utilizo el lenguaje de programación Python y las librerías PyFiglet, Random, String, Colorama y Traceback.

El código tiene variosfunciones diferentes, cada una encargada de una tarea específica. La función menu() se encarga de mostrar el menú y ejecutar la opción elegida por el usuario. Las funciones password\_analyzer(), generate\_password(), save\_password(), retrieve\_password() y update\_password() se encargan de realizar las diferentes acciones del gestor de contraseñas.

Para guardar las contraseñas y nombres de usuario, se utiliza un diccionario denominado passwords, el cual guarda la información en memoria RAM. Además, se utiliza un archivo de texto llamado "passwords.txt" para guardar un registro de todas las contraseñas guardadas.

El flujo del programa comienza con la ejecución del archivo principal, el cual llama a la función menu() para mostrar el menú principal. El usuario puede elegir entre diferentes opciones, y el programa ejecutará la función correspondiente. Si el usuario ingresa una opción inválida, se muestra un mensaje de error y se escribe en un archivo de registro llamado "error.log".

La función password\_analyzer() analiza si la contraseña ingresada cumple con los criterios de seguridad necesarios, como la longitud mínima, la inclusión de letras mayúsculas y minúsculas, números y caracteres especiales. Si la contraseña no cumple con estos criterios, se muestra un mensaje de error y se indica cuáles son los criterios que faltan. Si la contraseña es segura, se devuelve un valor verdadero por lo que el usuario validará que pude usar esa contraseña.

La función generate\_password() genera una contraseña aleatoria de 12 caracteres (este detalle lo podemos modificar e incluso dejar 8 caracteres) utilizando caracteres alfanuméricos y especiales, y comprueba si cumple con los criterios de seguridad utilizando la función password\_analyzer(). Si la contraseña no cumple con los criterios de seguridad, se genera otra hasta que se cumplan.

La función save\_password() guarda la contraseña y el nombre de usuario en el diccionario passwords y escribe un registro en el archivo de texto "passwords.txt" con la información guardada. Si el sitio web ya se encuentra en el diccionario, se agrega el nuevo nombre de usuario y contraseña. Si el sitio web no se encuentra en el diccionario, se crea una nueva entrada.

La función retrieve\_password() busca el sitio web en el archivo "passwords.txt" y muestra la contraseña correspondiente si se encuentra en el archivo. Si el sitio web no se encuentra en el archivo, se muestra un mensaje de error pero nunca se menciona que alguno de los inputs es el erróneo.

La función update\_password() busca el sitio web y el nombre de usuario en el diccionario passwords y actualiza la contraseña correspondiente. Si el sitio web y el nombre de usuario no se encuentran en el diccionario, se muestra un mensaje de error en este mismo tampoco se menciona si algunos de las dos variables es la errónea y así no se sugiere nada.

En conclusión, la implementación del gestor de contraseñas UrPass se basa en un conjunto de funciones que trabajan en conjunto para realizar las diferentes acciones del programa, y utiliza un diccionario para guardar la información en memoria RAM y un archivo de texto para guardar un registro de todas las contraseñas guardadas por otro lado también se mantienen los registros de errores en un archivo error.log en el caso de que el admin pueda validar que errores han ocurrido y darle mantenimiento.

# Conclusiones

En conclusion este proyecto ha sido bastante retador no solo por la complejidad de los tópicos sino por tratar de brindar una solución a un problema que muchos tienen.

Por otro lado comprendo que la escalabilidad del proyecto es poco posible debido a donde se almacenan los passwords y usuarios, sin embargo este proyecto funciona como semillero para mi portfolio, siento que me ha brindado la oportunidad de implementar nuevas areas de conocimiento y manipular un lenguaje como Python, ya que mi enfoque es un poco mas visual.

En resumen este proyecto ha sido una buena practica para mi habilidad con Python y potenciales lenguajes a futuro.

# Bibliografia

Stack Overflow. (2011). Python reading text file.

<https://stackoverflow.com/questions/7485458/python-reading-text-file>

Stack Overflow. (2011). How to write to a file using the logging Python module?

https://stackoverflow.com/questions/6386698/how-to-write-to-a

-file-using-the-logging-python-module

b001. (n.d.). YouTube channel.

<https://www.youtube.com/@b001>

Amir Bhb. (n.d.). YouTube channel.

<https://www.youtube.com/@amirbhb>

Python Basics. (n.d.). Join.

<https://pythonbasics.org/join/>

MoureDev. (n.d.). YouTube channel.

<https://www.youtube.com/@mouredev>

Corey Schafer. (2020). Python Tutorial: Logging Basics - Logging to Files, Setting Levels, and Formatting. YouTube video.

<https://www.youtube.com/watch?v=dR_cDapPWyY&t=1577s>

Colegas de trabajo en Gorilla Logic

Juan Camilo Valencia : Senior Python Developer

Reynaldo Guerrero: Senior Java/Python Developer