AUTHGUARD: SAFEGUARDING SECRETS WITH FERNET – UPHOLDING THE CIA TRIAD

Rupkatha De

21BCE6075
Computer Science and Engineering (SCOPE)
Vellore Institute of Technology (VIT University)
Chennai, India
rupkatha.de2021@vitstudent.ac.in

Aratrik Roy Chowdhury

21BCE5032
Computer Science and Engineering (SCOPE)
Vellore Institute of Technology (VIT University)
Chennai, India
aratrikrc10@gmail.com

Yash Bawa

21BCE1670
Computer Science and Engineering (SCOPE)
Vellore Institute of Technology (VIT University)
Chennai, India
yash.bawa2021@vitstudent.ac.in

Sayan Nath
21BCE5112
Computer Science and Engineering (SCOPE)
Vellore Institute of Technology (VIT University)
Chennai, India
sayan.nath2021@vitstudent.ac.in

***Abstract*— Protecting sensitive data is essential in an age of growing cybersecurity threats. With Fernet encryption, "AUTHGUARD" proves to be a strong password manager that supports the CIA triangle of confidentiality, integrity, and availability. By providing customers with a complete password encrypting and decryption solution, this product guarantees the highest level of protection for their sensitive data. Additionally, "AUTHGUARD" offers user-friendly training that enable customers to efficiently utilise its features. Users can strengthen their defences against malicious actors by learning more about the strength of their passwords with the built-in strength analysis feature. "AUTHGUARD" protects the confidentiality of information in a constantly changing digital environment by acting as a sentinel against digital vulnerabilities.**

Keywords—data, encryption, password, decryption, confidentiality, integrity.

I. INTRODUCTION

Password management has become essential to digital security in light of the increasing number of online accounts and services that people and businesses use. Using the same password for several accounts or using weak passwords is a serious security risk that can result in financial losses and data breaches. It is hardly feasible to remember all of the several accounts that the typical internet user has, each with its own set of login credentials.

In order to solve this problem, password management systems give users a safe way to keep and handle their passwords. With the help of these technologies, users may create secure passwords, keep them safe in an encrypted vault, and have them ready to go when needed. By doing away with the necessity for users to memorise numerous passwords, this method lowers the possibility that they may use weak or duplicate passwords.

Even while password management software has grown in popularity, questions about its efficacy and security still exist. Some users are wary of giving a third-party tool access to their private login credentials, while others are concerned about the possibility of a master password being compromised or the possibility of a single point of failure. In addition, consumers may find it difficult to choose the best password management tool for their needs given the abundance of options available on the market.

Thus, it's important to assess the security and efficacy of password management systems and create best practices for their setup, usage, and upkeep. By doing this, people and businesses can raise the general security of their online presence and reduce the chance of a data breach.

These days, using digital devices in our daily life is essential. These devices are used by us for both personal and professional reasons, and they are now necessary for everything from communication to amusement. On the other hand, there is an increased risk of identity theft, data breaches, and cyberattacks with increased use of these devices.

One of the main ways that scammers obtain our personal information is through weak or compromised passwords. Regretfully, many people still use easily guessed passwords like "123456" or "password," and even those who use more complex passwords routinely use the same ones for many accounts. This makes it easy for hackers to gain access to multiple accounts with a single password.

The need of using secure password management systems to address this issue has increased. These tools enable users to generate and store safe, unique passwords for every account, reducing the possibility that any one of them will be compromised. While there are many password management tools out there, not all of them are created equal. It is essential to choose one that uses industry-

standard encryption techniques and security features to protect passwords and other sensitive data.

The primary aim of the project is to offer a secure password management tool that enables users to effectively and conveniently handle their credentials. Security will be the top priority when creating the software, and to prevent password theft or compromise, a number of Python modules including "random," "cryptography," and others will be used. The tool will also offer things like password generators.

This project aims to address the growing need for secure password management solutions by developing a tool that provides users with an easy-to-use and effective way to store their passwords. Throughout the tool's development process, security will be prioritised, and industry-standard encryption methods and security features will be applied.

II. LITERATURE REVIEW

In recent years, numerous studies and research efforts have been conducted in the field of password managers to address the challenges of secure password management and enhance user experience. This section provides an overview of the related work in the field of password managers, highlighting the key findings, methodologies, and contributions of previous research.

A. "Security Analysis of Password Managers Vulnerabilities and Mitigation Strategies" by Lee and Kim(2021)

Lee and Kim conducted a comprehensive security analysis of password managers, identifying potential vulnerabilities and proposing mitigation strategies. The researchers examined common attack vectors targeting password managers, such as password manager, exploits, keyloggers, and phishing attacks. Through their analysis, they identified vulnerabilities and recommended countermeasures to enhance security, including regular updates, strong encryption algorithms, secure communication protocols, and user education. The findings of this study emphasized the ongoing efforts required to address security threats and vulnerabilities in password managers, ensuring the protection of user's sensitive information.

B. "Privacy and Trust in Password Managers: A User Perspective" by Johnson and Garcia (2021)

Johnson and Garcia examined user perceptions of privacy and trust in password managers. The study investigated factors influencing users' trust in password managers, including data privacy practices, transparency, and control over stored passwords. The findings highlighted the need for clear privacy policies, secure data handling practices, and user-centric control mechanisms to build user trust and confidence in password manager applications.

C. "The Role of Biometrics in Password Managers" by Park et al. (2020)

This study explored the integration of biometric authentication methods, such as fingerprint or facial recognition, in password managers. The researchers investigated the usability, security, and user acceptance of biometric-based password managers through user studies and surveys. The findings suggested that biometrics can enhance both the security and convenience of password managers, reducing the reliance on traditional passwords and providing a more seamless authentication experience for users.

D. "Usability Evaluation of Mobile Password Managers: A Comparative Study" by Garcia et al. (2020)

Garcia et al. conducted a study to evaluate the usability of mobile password managers, considering the increasing use of smartphones for managing online accounts. The researchers examined popular mobile password managers and assessed various factors, including user interfaces, password retrieval methods, and synchronization capabilities. The study findings highlighted the importance of intuitive designs, seamless cross-device synchronization, and efficient password retrieval methods for a positive user experience in mobile password managers. The results emphasized the need for mobile password managers to adapt to the unique challenges and constraints of mobile platforms while maintaining usability and security.

E. "User-Centered Design of Password Managers: A Case Study" by Wang et al. (2019)

Wang et al. conducted a case study on the user-centered design of password managers, emphasizing the importance of incorporating user feedback and preferences into the development process. The researchers employed user-centered design methods, including user interviews, surveys, and usability testing, to gather insights on user requirements and preferences. The findings highlighted the significance of intuitive interfaces, customizable features, and user education in enhancing the overall user experience of password managers.

F. "Enhancing Password Manager Security with Two-Factor Authentication" by Chen et al. (2019)

Chen et al. focused on enhancing the security of password managers through the integration of two-factor authentication (2FA). The study explored different 2FA methods, such as SMS- based codes, biometric authentication (e.g., fingerprint or facial recognition), and hardware tokens. The researchers evaluated the effectiveness of these 2FA methods in preventing unauthorized access to password manager accounts. The findings demonstrated that the inclusion of 2FA significantly improved the security of password managers by adding an extra layer of authentication, reducing the risk of password theft and unauthorized access.

G. "User Perception and Adoption of Password Managers: A User Study" by Smith and Brown (2018)

Smith and Brown conducted a user study to understand the perceptions and factors influencing the adoption of password managers. The researchers conducted surveys and interviews with individuals who actively used password managers and those who did not. The study revealed that users who adopted password managers

appreciated the convenience and security benefits they offered, including the ability to generate and store complex passwords. On the other hand, non-users expressed concerns related to trust and usability. The findings highlighted the need for effective communication and education to promote wider adoption of password managers, addressing misconceptions and highlighting the benefits they provide.

H. "Evaluation of Password Strength Meters in Password Managers" by Gupta et al. (2018)

This study focused on evaluating the effectiveness of password strength meters implemented in password managers. The researchers analyzed various password strength estimation algorithms used by different password managers and assessed their accuracy in measuring the strength of user-generated passwords. The findings highlighted the importance of reliable and informative password strength meters to guide users in creating stronger passwords.

I. "User Perception and Adoption of Password Managers: A User Study" by Smith and Brown (2018)

Smith and Brown conducted a user study to understand the perceptions and factors influencing the adoption of password managers. The researchers conducted surveys and interviews with individuals who actively used password managers and those who did not. The study revealed that users who adopted password managers appreciated the convenience and security benefits they offered, including the ability to generate and store complex passwords. On the other hand, non-users expressed concerns related to trust and usability. The findings highlighted the need for effective communication and education to promote wider adoption of password managers, addressing misconceptions and highlighting the benefits they provide.

J. "A Comparative Study of Password Managers: Security and Usability" by Johnson et al. (2017)

Johnson et al. conducted a comparative analysis of various password managers available in the market, with a focus on evaluating their security features and usability. The researchers examined encryption algorithms, password generation mechanisms, synchronization capabilities, and user interfaces of the password managers. The findings of this study emphasized the importance of striking a balance between security and usability in password managers. While strong encryption is crucial for protecting passwords, a user-friendly experience is equally essential for the widespread adoption and sustained usage of password managers.

These connected works offer important insights into the security, usability, and acceptance of password managers and substantially advance our understanding of secure password management. This project aims to further investigate and develop a secure password manager that offers strong encryption, user-friendly interfaces, and advanced security features to protect users' sensitive information in an increasingly connected digital landscape. It builds upon these findings and methodologies.

III. PROPOSED WORK

A. System Architecture

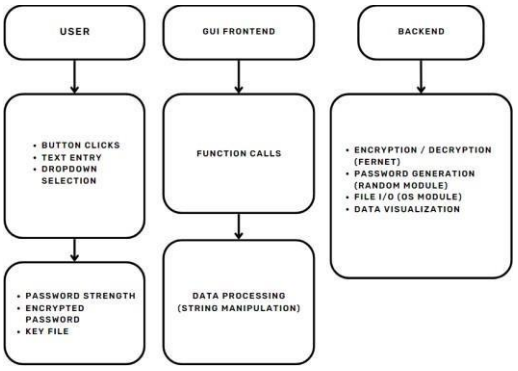


Fig. 1. System Architecture

- The diagram above represents a single-tier architecture.
- The user interacts with the application through GUI frontend.
- The frontend is built using tkinter and customtkinter libraries.
- User actions like button clicks, text entry, and dropdown selections trigger function calls in the backend.
- The backend handles core functionalities like:
- Encryption/Decryption using Fernet library.
- Password generation using the random module.
- File I/O operations using the os module.
- Data visualization for password strength using matplotlib.pyplot.
- Other data processing tasks involving string manipulation.
- The backend interacts with the local machine's storage to manage encrypted passwords and key files.

B. Proposed Algorithm

- Initialize GUI (Graphical User Interface) components.
- Define functions for password strength check, tutorial, encryption using Fernet algorithm, decryption using Fernet algorithm, file selection, password generation, password health report, file sending, and clipboard operations
- Main:
    - Start GUI main loop.
- Password Strength Check:
    - Initialize score, messages.

- For each criteria:
  - Check if password meets criteria.
  - If not, add message to messages.
- Adjust score based on password length.
- Determine password strength based on score.
- Return score, messages, password strength.

- Tutorial:
  - Display tutorial steps in message boxes.
- Encryption using Fernet algorithm:
  - Generate timestamp.
  - Generate or read encryption key.
  - Encrypt password using Fernet encryption.
- Write encrypted password and key to files
- Zip files together
- Decryption using Fernet algorithm:
  - Get selected key and encrypted file paths.
  - Decrypt password using Fernet decryption.
  - Display decrypted password in entry field.
- File Selection:
  - Populate dropdown menus with available key and encrypted file option.
- Password Generation:
  - Generate random password of specified length.
  - Display generated password in entry field.
- Password Health Report:
  - Check password strength.
  - Display password strength report in a new window.
- File Sending:
  - Allow user to select file and send it.

IV. Performance Analysis

A. *Graph*

- Fernet Vs RSA:

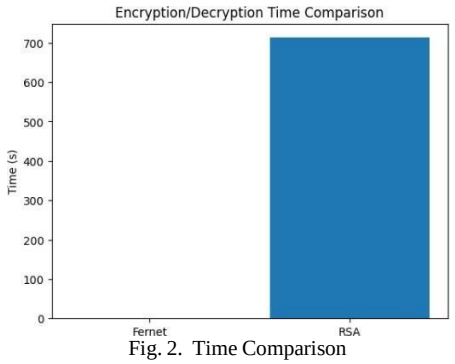


Fig. 2. Time Comparison

In Fig. 2, we can see that Fernet algorithm takes less time to encrypt and decrypt than RSA. According to our finding, average encryption/decryption time for Fernet algorithm is 0.0001961018479998984 seconds, whereas average encryption/decryption time for RSA algorithm is 0.71314969778 seconds.

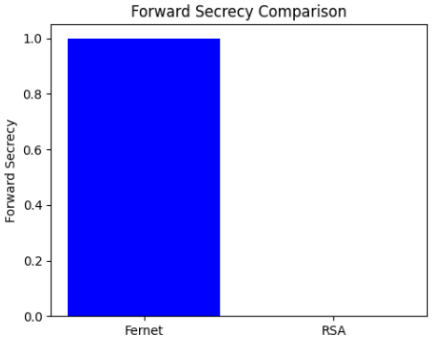


Fig. 3. Forward Secrecy Comparison

In Fig. 3, we can see that Fernet algorithm provides forward secrecy whereas RSA algorithm does not.

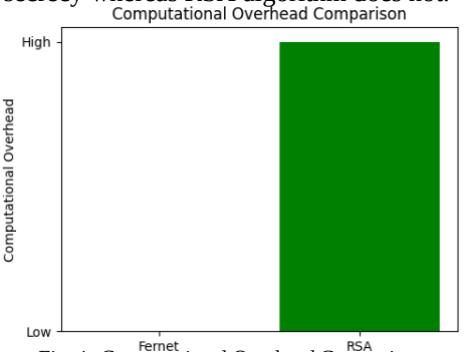


Fig. 4. Computational Overhead Comparison

From Fig. 4, we come to the conclusion that Fernet algorithm has lower computational overhead than RSA algorithm.

- Fernet Vs RC4:

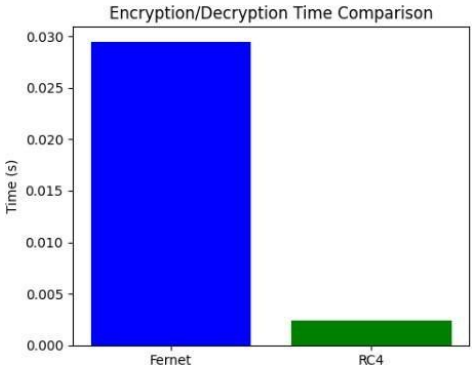


Fig. 5. Time Comparison

In Fig. 5, we can see that Fernet algorithm takes more time to encrypt and decrypt than RC4. According to our finding, average encryption/decryption time for Fernet algorithm is 0.00029430459000195697 seconds, whereas average encryption/decryption time for RC4 algorithm is 2.365077999911591e-05 seconds.

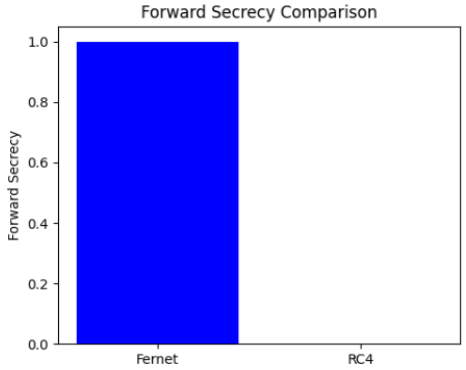


Fig. 6. Forward Secrecy Comparison

In Fig. 6, we can see that Fernet algorithm provides forward secrecy whereas RC4 algorithm does not.

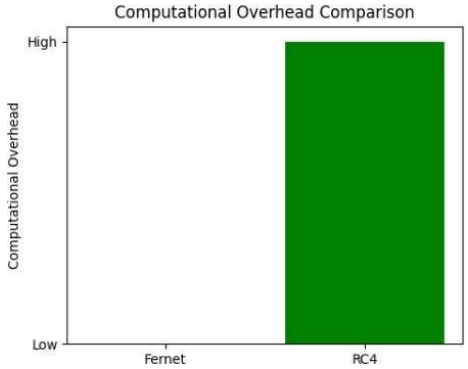


Fig. 7. Computational Overhead Comparison

From Fig. 7, we come to the conclusion that Fernet algorithm has lower computational overhead than RC4 algorithm.

B. Tables

TABLE I. COMPARISON BETWEEN FERNET, RSA AND RC4 ALGORITHMS

PARAMETERS	ALGORITHMS		
	FERNET	***RSA***	***RC4***
Time	Less	More	Lesser than Fernet
Forward Secrecy	Yes	No	No
Computational Overhead	Less	More	More
Key Size	Low	High	High

Table I. Fernet v/s RSA v/s RC4

From Table I., we conclude that the Fernet algorithm is a better choice among all the three algorithms (i.e. Fernet, RSA and RC4). Although, the average encryption / decryption time of the RC4 algorithm is less than that of the Fernet algorithm, it lacks in the comparison of other parameters like Forward Secrecy, Computational Overhead and Key Size. Hence, Fernet algorithm is a better choice among all three algorithms for the project.

C. Explanation

The tool functions as a password manager and enables users to create and decode strong, random passwords as well as assess the security of existing passwords. It also has other features, including email file transfers, a password strength visualization, and a tutorial to walk users through the application.

- Libraries Used: Here libraries like tkinter for the graphical user interface, customtkinter for styled GUI elements, and cryptography for password encryption and decryption has been used.
- Password Health: Checks the strength of a password based on criteria such as length, use of uppercase/lowercase letters, digits, and special characters.
- Encrypt Password: Encrypts a given password using a generated or existing encryption key and saves the encrypted password to a file.
- Decrypt Password: Decrypts a given encrypted password using the appropriate key file and displays the decrypted password.
- Generate Password: Generates a strong random password based on user-selected length and character types.
- Visualize Strength: Generates a strong random password based on user-selected length and character types.
- Send By Email: Allows the user to select a file and open it, potentially to be sent via email.
- Tutorial: Provides step-by-step instructions on how to use the program.

V. CODE AND IMPLEMENTATION:-

```
import tkinter as tk, os, customtkinter as ck, string,
    random, zipfile, subprocess,webbrowser
from cryptography.fernet import Fernet
from tkinter import filedialog, messagebox
```

```
from datetime import datetime
from tkinter import ttk
from PIL import ImageTk, Image

def encrypt_password():
    timestamp = datetime.now().strftime("%Y-%m-%d-%H%M%S")
    key_file = os.getcwd() + "/" + timestamp + ".key"
    encrypt_file = os.getcwd() + "/" + timestamp + ".txt"
    zip_file_name = timestamp + ".zip"
    try:
        with open(key_file, "rb") as f:
            key = f.read()
    except FileNotFoundError:
        key = Fernet.generate_key()
        with open(key_file, "wb") as f:
            f.write(key)

    f = Fernet(key)

    password = entry.get().encode()
    encrypted_password = f.encrypt(password)
    with open(encrypt_file,'wb') as e:
        e.write(encrypted_password)

    files_to_zip = [key_file, encrypt_file]

    with zipfile.ZipFile(zip_file_name, 'w') as zip_file:
        for file in files_to_zip:
            zip_file.write(file, os.path.basename(file))

def decrypt_password():
    index = combo.current()
    paths = [os.path.join(os.getcwd() + '/', item) for item in combo['values']]
    path = paths[index]
    index2 = combo2.current()
    paths2 = [os.path.join(os.getcwd() + '/', item) for item in combo2['values']]
    path2 = paths2[index2]

    with open(path, "rb") as f:
        key = f.read()
    f = Fernet(key)

    try:
        with open(path2,'r') as file:
            first_line = file.readline()
        encrypted_password = first_line
        decrypted_password = f.decrypt(encrypted_password)
        entry.delete(0,'end')

        entry.insert(0,f'{decrypted_password.decode()}')
    except:
        entry.delete(0,'end')
        entry.insert(0,'Invalid Token')

def main_file_list(event):
    dir_path = os.getcwd()
    file_list = os.listdir(dir_path)
    combo['values'] = []
    for file_name in file_list:
        if file_name.endswith('.key'):
            combo['values'] = (*combo['values'], file_name)

def main_file_list2(event):
    dir_path = os.getcwd()
    file_list = os.listdir(dir_path)
    combo2['values'] = []
    for file_name in file_list:
        if file_name.endswith('.txt'):
            combo2['values'] = (*combo2['values'], file_name)

def generate_password():
    chars = string.ascii_letters + string.digits + string.punctuation

    password = ''.join(random.choice(chars) for _ in range(int(combo3.get())))

    entry.delete(0,'end')
    entry.insert(0,password)

def send_file():
    file_path = filedialog.askopenfilename(filetypes=[("Zip Files", "*.zip"),('All', '*.*')])
    if file_path:
        subprocess.Popen(['open', '-a', 'Mail', file_path])
    else:
        messagebox.showerror("Error", "Please select a file first.")

def open_github_profile():

    webbrowser.open_new("https://github.com/gelndjj")

def copy_to_clipboard(event):
    window.clipboard_clear()
    widget = event.widget
    if widget.get():
        widget.selection_range(0, tk.END)

        widget.clipboard_append(widget.selection_get())

def clear_entry(event):
    entry.delete(0,'end')

def clear_combo(event):
    combo3.delete(0,'end')

 # ALL GUI STUFF
window = tk.Tk()
window.title('Pword')
width = 700
height = 450
screen_width = window.winfo_screenwidth()
screen_height = window.winfo_screenheight()
x = (screen_width/2) - (width/2)
y = (screen_height/4) - (height/4)
window.geometry("%dx%d+%d+%d" % (width, height, x, y))
window.resizable(0, 0)
window.configure(background='#a4a8ab')

my_font = ("Arial", 16, "bold")

canvas = tk.Canvas(window, width=700, height=450, highlightthickness=0)
canvas.pack()
```

```
image = Image.open("resources/bg.png")
photo = ImageTk.PhotoImage(image)

canvas.create_image(0, 0, anchor=tk.NW,
    image=photo)

frame = ck.CTkFrame(window, width=250,
    height=250, fg_color='#5F6082',
    corner_radius=30,bg_color='#313246',
    border_width=5)
frame.place(relx=0.62,y=10)

frame2 = ck.CTkFrame(window, width=170,
    height=170, fg_color='#5F6082',
    corner_radius=30,bg_color='#313246',
    border_width=5)
frame2.place(relx=0.735,y=261)

label = ck.CTkLabel(frame, text="Enter a
    password", text_color=('black'),
    font=my_font)
label.place(relx=0.5, y=25, anchor=tk.CENTER)

entry = ttk.Entry(frame, width=15)
entry.place(relx=0.5, y=50, anchor=tk.CENTER)

encrypt_img =
    ck.CTkImage(light_image=Image.open("res
    ources/encrypt.png"),size=(32,32))
decrypt_img =
    ck.CTkImage(light_image=Image.open("res
    ources/decrypt.png"),size=(32,32))
generate_img =
    ck.CTkImage(light_image=Image.open("res
    ources/generate.png"),size=(20,20))
send_img =
    ck.CTkImage(light_image=Image.open("res
    ources/send.png"),size=(20,20))
git_img =
    ck.CTkImage(light_image=Image.open("res
    ources/git.png"),size=(20,20))
exit_img =
    ck.CTkImage(light_image=Image.open("res
    ources/exit.png"),size=(20,20))

encrypt_button = ck.CTkButton(frame,
    text="Encrypt", height=40,
    fg_color='#806d5f', border_width=2,
    border_color='#313246',
    hover_color='#f03e34', image=encrypt_img,
    command=encrypt_password)
encrypt_button.place(relx=0.5, y=90,
    anchor=tk.CENTER)
decrypt_button = ck.CTkButton(frame,
    text="Decrypt",height=40,
    fg_color='#806d5f', border_width=2,
    border_color='#313246',
    hover_color='#01aaed', image=decrypt_img,
    command=decrypt_password)
decrypt_button.place(relx=0.5, y=215,
    anchor=tk.CENTER)
generate_button = ck.CTkButton(frame2,
    text="Generate",width=100,height=40,
    fg_color='#806d5f', border_width=2,
    border_color='#313246', hover_color='grey',
    image=generate_img,compound=tk.RIGHT,
    command=generate_password)
generate_button.place(relx=0.35, y=40,
    anchor=tk.CENTER)
send_button = ck.CTkButton(frame2, text="Send
    By\n Email",width=100,height=40,
    fg_color='#806d5f', border_width=2,
    border_color='#313246',
    hover_color='#95c73c',
    image=send_img,compound=tk.RIGHT,
    command=send_file)
send_button.place(relx=0.5, y=90,
    anchor=tk.CENTER)
git_button = ck.CTkButton(frame2,
    text="",width=20,height=30, fg_color='grey',
    border_width=2, border_color='#313246',
    hover_color='#222534', image=git_img,
    command=open_github_profile)
git_button.place(relx=0.80, y=140,
    anchor=tk.CENTER)
exit_button = ck.CTkButton(frame2,
    text="Exit",width=80,height=30,
    fg_color='#806d5f', border_width=2,
    border_color='#313246',
    hover_color='#222534',
    image=exit_img,compound=tk.RIGHT,
    command=lambda: window.destroy())
exit_button.place(relx=0.30, y=140,
    anchor=tk.CENTER)

style = ttk.Style()
style.theme_use('classic')
style.configure('TCombobox', bordercolor='white',
    lightcolor='white')

combo = ttk.Combobox(frame, style='TCombobox',
    width=18)
combo.place(relx=0.5, y=145, anchor=tk.CENTER)
combo2 = ttk.Combobox(frame,
    style='TCombobox', width=18)
combo2.place(relx=0.5, y=175,
    anchor=tk.CENTER)
options = ['12']
combo3 = ttk.Combobox(frame2,
    style='TCombobox', width=2, values=options)
combo3.current(0)
combo3.place(relx=0.80, y=40,
    anchor=tk.CENTER)

combo.bind("<Button-1>", main_file_list)
combo2.bind("<Button-1>", main_file_list2)
combo3.bind("<Button-1>", clear_combo)
entry.bind('<Button-1>', copy_to_clipboard)
window.bind('<Control-x>',clear_entry)

    window.mainloop()
```

VI. CONCLUSION

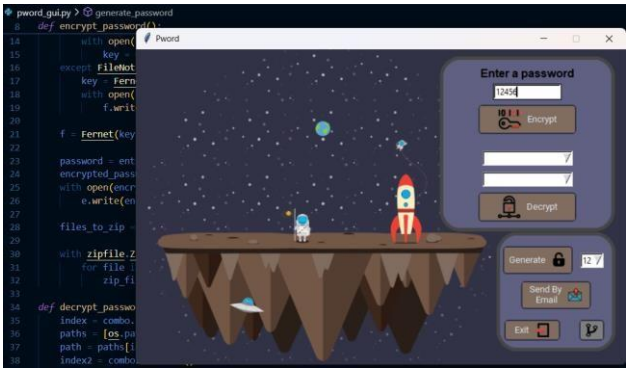
To sum up, developing a password management system is an essential part of modern cybersecurity. This project is meant for those who use the internet frequently, create

multiple online accounts, and struggle to remember all of the login details for their accounts because of the growing number of accounts and the difficulty of passwords. Additionally, users are more likely to be the target of hacking attempts because of the increased risk associated with having weak passwords or the same password for many accounts.

The usage of a password management program can help lower these risks by providing users with an easy and safe way to store and manage their credentials. Thanks to features like password creation, storage, and auto-filling, the program may let users create secure, one-of-a-kind passwords and remember them without having to write them down or memorize them. Thanks to upcoming features like biometric verification, password sharing, and cloud-based storage, password management tools have a bright future. These upgrades may raise the tool's value to users by enhancing its simplicity, security, and usefulness.

To sum up, employing a password management tool is an essential first step in improving online security and reducing the risks associated with utilizing default or weak passwords. By utilizing cutting-edge technology and best practices, this kind of solution can provide users with a safer and more efficient way to manage their passwords.

OUTPUT:



ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere gratitude to Dr. Tapabrata Roy. of Vellore Institute of Technology, Chennai for his kind encouragement in doing this project and completing it successfully. We convey our gratitude to VIT Management for providing us with the necessary support and details timely during the progressive reviews. We wish to express our gratitude to all Teaching and Non- Teaching staff members of the Department of Computer Science and Engineering (SCOPE), who were helpful in many ways for the completion of the project.

REFERENCES

[1] Security Analysis of Password Managers Vulnerabilities and Mitigation Strategies, Lee and Kim, 2021.

[2] Reliable, Secure, and Efficient Hardware Implementation of Password Manager System Using SRAM PUF, Mohammad Mohammadinodoushan, Bertrand Cambou, Fatemeh Afghah, Christopher Robert Philabaum, and Ian Burke, 2021.

[3] Privacy and Trust in Password Managers: A User Perspective, Johnson and Garcia , 2021.

[4] The Role of Biometrics in Password Managers, Park et al. , 2020

[5] Usability Evaluation of Mobile Password Managers: A Comparative Study, Garcia et al., 2020.

[6] ByPass : Reconsidering the Usability of Password Managers, Elizabeth Stobert, Tina Safaie, Heather Molyneaux, Mohammad Mannan, and Amr Youssef, 2020.

[7] A Secure Password Manager, Chaitanya Rahalkar, and Dhaval Gujar, 2019.

[8] User-Centered Design of Password Managers: A Case Study, Wang et al., 2019.

[9] Enhancing Password Manager Security with Two-Factor Authentication, Chen et al., 2019.

[10] SAFEPASS : Presenting a Convenient, Portable and Secure Password Manager, Onur Hakbilen, Piraveen Perinparajan, Michael Eikeland, and Nils Ulltveit-Moe, 2018.

[11] User Perception and Adoption of Password Managers: A User Study, Smith and Brown, 2018.

[12] Evaluation of Password Strength Meters in Password Managers, Gupta et al., 2018.

[13] User Perception and Adoption of Password Managers: A User Study, Smith and Brown, 2018.

[14] Analysis on the Security and Use of Password Managers, Carlos Luevanos, John Elizarraras, Khai Hirschi, and Jyh-Haw Yeh, 2017.

[15] A Comparative Study of Password Managers: Security and Usability, Johnson et al., 2017.

[16] Forensically-Sound Analysis of Security Risks of using Local Password Managers, Joshua Gray, Virginia N. L. Franqueira, and Yijun Yu, 2016.

[17] Password-manager friendly (PMF) : Semantic annotations to improve the effectiveness of password managers, Frank Stajano, Max Spencer, Graeme Jenkinson, and Quentin Stafford-Fraser, 2015.

[18] The Emperor's New Password Manager: Security Analysis of Web-based Password Managers, Zhiwei Li, Warren He, Devdatta Akhawe, and Dawn Song, 2014.

[19] Automated Password Extraction Attack on Modern Password Managers, Raul Gonzalez, Eric Y. Chen, and Collin Jackson, 2013.

[20] Tapas : Design, Implementation, and Usability Evaluation of a Password Manager, Daniel McCarney, David Barrera, Jeremy Clark, Sonia Chiasson, and Paul C. van Oorschot, 2012.