American Journal of Engineering Research (AJER) ***2025***
American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-14, Issue-2, pp-140-148
www.ajer.org
Research Paper Open Access

Securing Digital Identities: The Role and Impact of Password Manager

Saleh Muhammad Maruf[1], Mohammad Hasan[2], Abu Hena Md. Mustafa Kamal[3]
[1](*Saleh Muhammad Maruf, Assistant Professor, Department of Computer Science & Engineering, Institute of Science & Technology, Dhaka 1209)*
[2](*Mohammad Hasan, Lecturer, Department of Computer Science & Engineering, Institute of Science & Technology, Dhaka 1209)*
[3](*Abu Hena Md. Mustafa Kamal, Assistant Professor, Department of Electronics & Communication Engineering, Institute of Science & Technology, Dhaka 1209)*
Corresponding Author: Saleh Muhammad Maruf

ABSTRACT : *Passwords continue to be the most important safeguard for users' sensitive data and online accounts in the modern digital world. Yet, the ever-growing number of online platforms coupled with the demand for robust, unique passwords has introduced significant challenges for users. These challenges often manifest as common yet risky behaviors, such as reusing passwords across multiple sites, choosing weak and easily guessable passwords, and facing difficulties in remembering a myriad of complex credentials. Such practices can leave individuals vulnerable to cyber threats, data breaches, and identity theft. To mitigate these issues, password managers have emerged as an essential tool for bolstering cybersecurity. These applications provide users with a secure and efficient method to store and manage their passwords by encrypting them within a digital vault. This not only simplifies the user experience but also enhances security by enabling the creation of strong, unique passwords for each service. This paper examines the multifaceted advantages of password managers, focusing on their role in promoting better password hygiene, improving security protocols, and streamlining user interactions with digital accounts. Furthermore, we delve into the cryptographic foundations that underpin the protection mechanisms within password managers, ensuring that stored credentials remain inaccessible to unauthorized parties. The paper also discusses the critical criteria for selecting a trustworthy password manager, evaluating aspects such as encryption standards, user privacy, and platform compatibility. Finally, we address potential risks associated with password managers, such as vulnerabilities to phishing attacks or master password compromises, and outline best practices for maximizing the benefits while minimizing potential threats.*

KEYWORDS *Password Manager, Cybersecurity, Encrypted Credentials, Secure Storage, Account Security, Data Breaches, Credential Management, Password Storage Database, Password Security Risks, Digital Wallet for Passwords*

Date of Submission: 20-02-2025 Date of acceptance: 03-03-2025

I. INTRODUCTION

A password is a string of characters that serves as a key to access computer applications or online platforms. It acts as a gatekeeper, ensuring that only authorized users can interact with a specific service or application. However, the need to create and manage strong, unique passwords for numerous accounts has become a significant challenge for users. A password manager can be thought of as a digital wallet specifically designed for password storage and management. A secure password manager helps users generate, remember, and store complex and unique passwords in one safe location. Users only need to remember a single master password or secret key to access all their stored credentials. This greatly reduces the cognitive load associated with password management and enhances security by encouraging the use of strong, non-repetitive passwords. Many individuals tend to create weak and short passwords or reuse the same password across multiple accounts because it is difficult

to remember numerous strong, unique passwords. This practice significantly increases the risk of security breaches. Analysis of a leaked password dataset from a Chinese website revealed that many passwords included personal information, such as names and dates of birth. Such patterns make passwords predictable and easy to exploit.

Over the past few years, thousands of passwords have been compromised due to practices such as incorporating personal information into passwords, writing passwords down in insecure locations, or reusing the same password across different accounts. These vulnerabilities have prompted the development of tools like random password generators and password managers to address these issues. Florêncio et al. argue that in the absence of password managers, users resort to grouping accounts and reusing passwords as the only feasible way to manage their credentials. This practice, while convenient, poses significant security risks. Consequently, organizations are encouraged to provide password managers equipped with built-in password generation tools. These tools enable users to create and store robust passwords without having to rely on memory or risky practices.

Password managers are designed to alleviate the challenges of managing multiple passwords. A secure password manager can automatically generate and fill in passwords on websites, sparing users the need to remember them. All stored passwords are encrypted, ensuring that sensitive information remains secure. Access to these encrypted credentials requires a master password or secret key, which serves as the sole piece of information the user needs to remember.

With a password manager, users can securely store all their usernames and passwords for various online accounts. This information is encrypted and stored in a secure database, providing an efficient and secure solution to the complexities of password management. By adopting such tools, users can mitigate the risks associated with weak passwords and improve overall cybersecurity practices. A possible solution to overcome this problem is to build a new system that will be user-friendly, easy to use, web-supported, efficient, and help to secure email and passwords. It will be free and open source so everyone can use it. This solution includes the new system being a web-based system so it can run from both local and web servers. The system objectives are:

- To build a system to store and organize your usernames, and passwords.
- To make it easier to find specific data quickly.
- To facilitate a platform for sharing a specific password with anyone.
- To protect your passwords from being hacked or forgotten.
- To provide multi-step authentication.
- To provide encrypted storage.

II. Literature Review

The growing need for renewable energy solutions has pushed the boundaries of research in solar radiation forecasting. Various studies have applied machine learning models for this purpose, each highlighting the advantages and limitations of different approaches. Traditional methods, such as regression analysis, have been widely used but tend to struggle with complex nonlinear relationships. On the other hand, more advanced methods like Random Forest, SVM, and ensemble learning techniques have shown promising results in capturing intricate data patterns. Recent research also emphasizes the importance of feature selection in enhancing model performance, as well as the role of data quality and size in improving predictive accuracy.

III. Related Work

Bootstrap

Bootstrap is a free and open-source front end development framework for the creation of websites and web apps. The Bootstrap framework is built on HTML, CSS and JavaScript (JS) to facilitate the development of responsive, mobile-first sites and apps. Bootstrap, originally named Twitter Blueprint, was developed by Mark Otto and Jacob Thornton at Twitter as a framework to encourage consistency across internal tools[6].

Laravel

Laravel is an open-source PHP framework, created by Taylor Otwell and intended for the development of web applications" The source code of Laravel is hosted on Github and licensed under the terms of MIT License. Which is robust and easy to understand. It follows a model-view-controller design pattern. Laravel reuses the existing components of different frameworks which helps in creating a web application. The web application thus designed

is more structured and pragmatic. Laravel offers a rich set of functionalities which incorporates the basic features of PHP frameworks[7].

Hash Function

A hash function is any function that can be used to map data of arbitrary size to fixed-size values. The hash value is a summary of the original data. A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length. Values returned by a hash function are called message digest or simply hash values[8]..

IV. METHODOLOGY

Hashing Algorithm

A hashing algorithm is a cryptographic hash function. It is a mathematical algorithm that maps data of arbitrary size to a hash of a fixed size.A hash function algorithm is designed to be a one-way function, infeasible to invert. However, in recent years several hashing algorithms have been compromised. This happened to MD5, for example — a widely known hash function designed to be a cryptographic hash function, which is now so easy to reverse — that we could only use for verifying data against unintentional corruption [9].

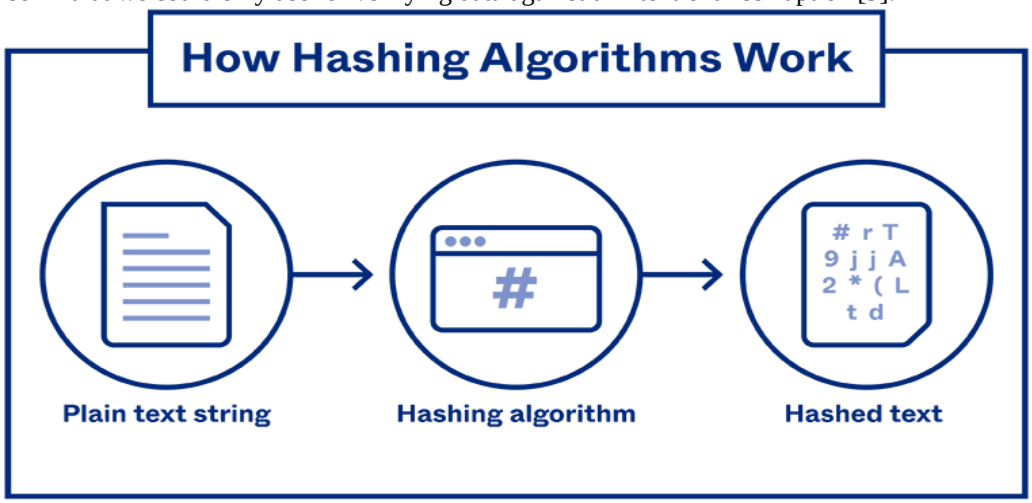

Figure 1: Hashing Algorithm

Workflow Diagram

Our workflow diagram outlines a comprehensive approach to password management, prioritizing security and user convenience. The process begins with the core function of managing passwords. Here, users have two primary options:

Create a New Password:

1. Input Password Information: Users provide essential details such as the website or application, username, and password.
2. Edit or Update: Existing passwords can be modified as needed to reflect changes in account information or security preferences.
3. Delete: Unnecessary or outdated passwords can be securely removed from the system.
4. Database Storage: Once created, edited, or deleted, password information is stored in a secure, encrypted database.

Share Passwords:

5. Invite Users: Users can invite trusted individuals to share specific passwords or groups of passwords.
6. Set Access Preferences: Granular control is provided over shared passwords, allowing users to define whether recipients can view, edit, or delete the shared information.
7. Secure Sharing: Advanced encryption techniques ensure that shared passwords remain confidential and protected during transmission.

By streamlining these processes, our password management solution empowers users to maintain strong security practices while simplifying their digital lives."

Key Improvements:

1. Enhanced Clarity: The revised segment provides a clearer and more concise explanation of the workflow steps.
2. Focus on Security: The emphasis on encryption and secure storage highlights the importance of protecting sensitive information.
3. User-Centric Approach: The description emphasizes how the workflow benefits users by simplifying password management and enhancing security.
4. Stronger Action Verbs: The use of action verbs like "empowers" and "streamlines" makes the language more dynamic and engaging.

This revised segment aims to provide a more compelling and informative overview of the password management workflow. The following two flowcharts outlining processes for password management: one for changing a password and the other for unlocking a system with a password. In the password-changing process, the user initiates the action by pressing the setting button and entering their old password. If the old password is correct, they proceed to enter and confirm a new password. If the two entries match, the password is successfully updated; otherwise, the user is prompted to re-enter. After three failed attempts at entering the old password, an alarm is triggered. In the unlocking process, the user presses the unlock button, enters their password, and confirms it by pressing enter. If the password is correct, the system unlocks; if incorrect, the user can try again. Similar to the password-changing process, three consecutive failed attempts activate an alarm. Both processes emphasize security by incorporating verification steps, limiting failed attempts, and requiring confirmation for changes.

.

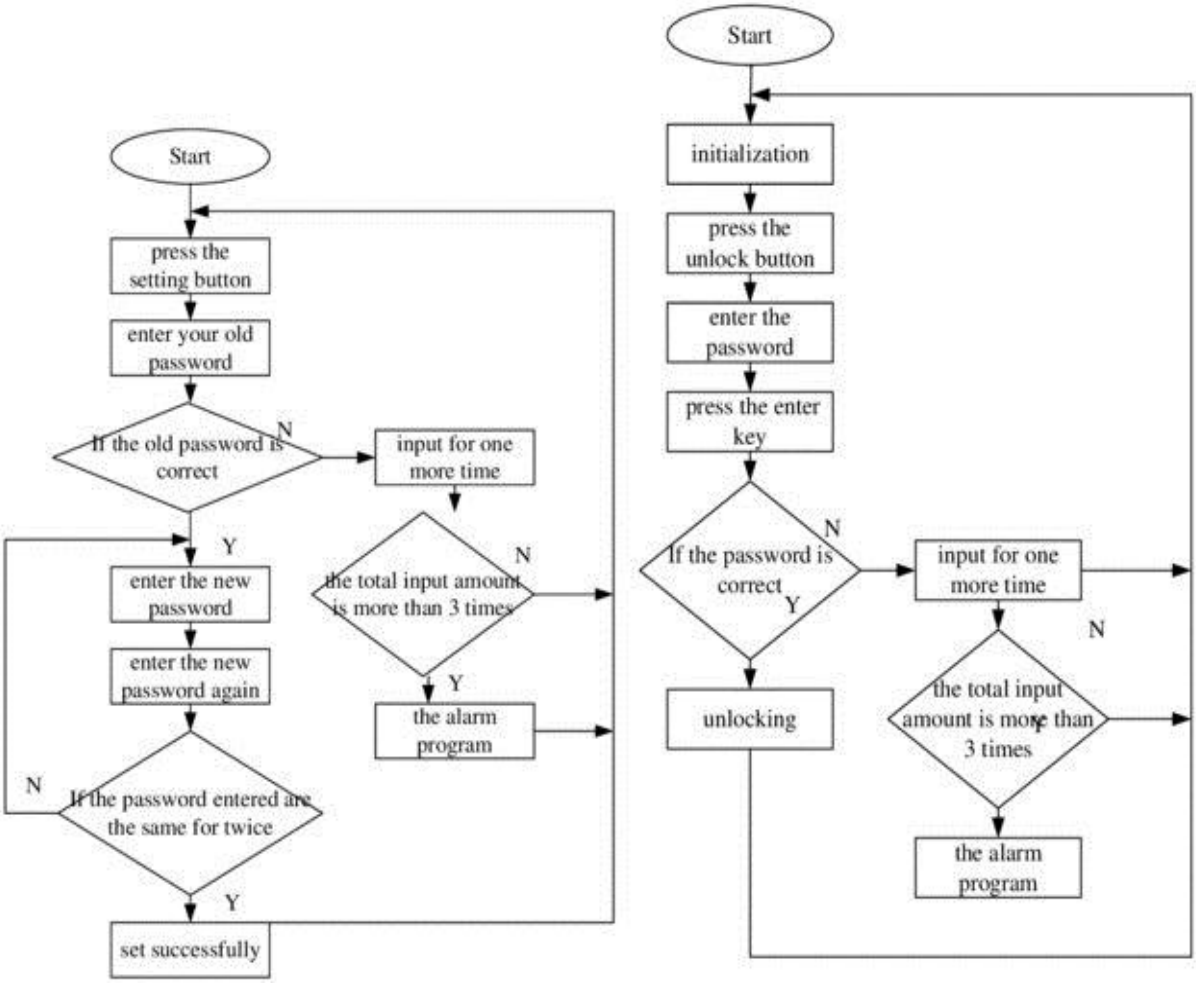

Figure 2: Workflow Diagram

UML Diagram

Our UML diagram illustrates the relationships between various entities within our password management system.

- User: Represents individual users of the system.
- User Log: Tracks login activity for each user, maintaining a one-to-one relationship.
- Password: Stores encrypted passwords for each user. A user can have multiple passwords, establishing a one-to-many relationship.
- Secure Password: Encrypts passwords using robust algorithms, maintaining a one-to-one relationship with each Password.
- Password Sharing: Facilitates password sharing between users. A user can share multiple passwords with other users, creating a one-to-many relationship.
- User Login: Records login attempts and successful logins for each user, maintaining a one-to-one relationship.

- Friend: Represents users who have been invited and added to a user's friend list. A user can have multiple friends, forming a one-to-many relationship.

- Access Preference: Defines the level of access granted to friends, such as view-only or edit permissions. A friend can have only one access preference, establishing a one-to-one relationship.

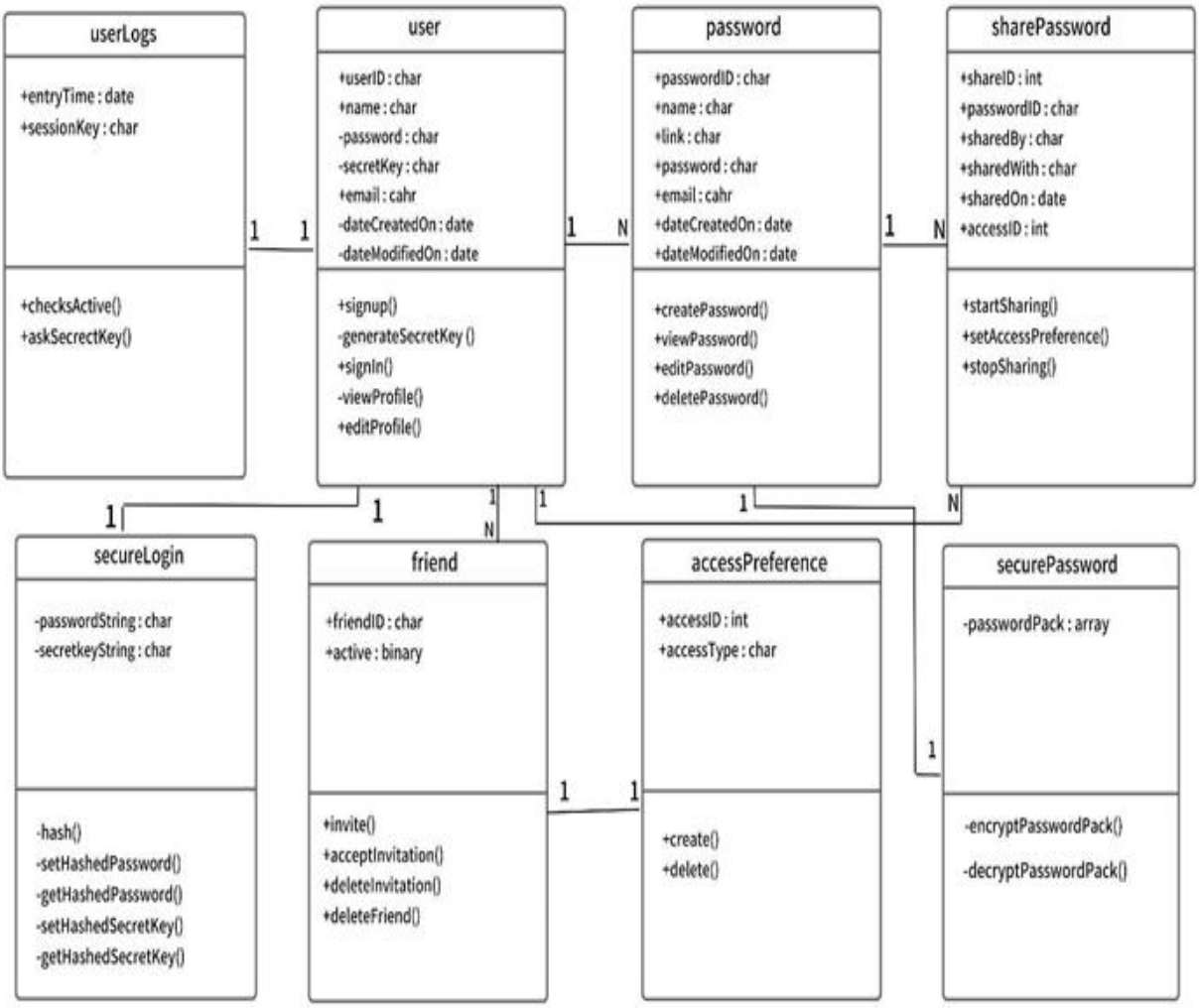

Figure 3: UML Diagram

Base64

Base64 encoding takes the original binary data and operates on it by dividing it into tokens of three bytes. A byte consists of eight bits, so Base64 takes 24 bits in total. These 3 bytes are then converted into four printable characters from the ASCII standard.

The first step is to take the three bytes (24bit) of binary data and split it into four numbers of six bits. Because the ASCII standard defines the use of seven bits, Base64 only uses 6 bits (corresponding to 2^6 = 64 characters) to ensure the encoded data is printable and none of the special characters available in ASCII are used. The algorithm's name Base64 comes from the use of these 64 ASCII characters. The ASCII characters used for Base64 are the numbers 0-9, the alphabets 26 lowercase and 26 uppercase characters plus two extra characters '+' and '/'[10].

Index	Char	Binary	Index	Char	Binary	Index	Char	Binary	Index	Char	Binary
0	A	0	16	Q	10000	32	g	100000	48	w	110000
1	B	1	17	R	10001	33	h	100001	49	x	110001
2	C	10	18	S	10010	34	i	100010	50	y	110010
3	D	11	19	T	10011	35	j	100011	51	z	110011
4	E	100	20	U	10100	36	k	100100	52	0	110100
5	F	101	21	V	10101	37	l	100101	53	1	110101
6	G	110	22	W	10110	38	m	100110	54	2	110110
7	H	111	23	X	10111	39	n	100111	55	3	110111
8	I	1000	24	Y	11000	40	o	101000	56	4	111000
9	J	1001	25	Z	11001	41	p	101001	57	5	111001
10	K	1010	26	a	11010	42	q	101010	58	6	111010
11	L	1011	27	b	11011	43	r	101011	59	7	111011
12	M	1100	28	c	11100	44	s	101100	60	8	111100
13	N	1101	29	d	11101	45	t	101101	61	9	111101
14	O	1110	30	e	11110	46	u	101110	62	+	111110
15	P	1111	31	f	11111	47	v	101111	63	/	111111

Figure 4: Base64

V. Result Analysis & Performance Evaluation

Dimension of the Project

The dimensions of our project are cost effective, secure, time consuming, management encryption, real time data and quick response. It can process a medium length of data within a very short time at a low cost. It can also provide users with information which can be very helpful. As this system takes a large amount of data and processes at a great speed, it responds quickly.

Encryption and Decryption security process

Algorithm 1:

```
from django.db import models

from django.contrib.auth.hashers import make_password, check_password


class Password(models.Model):
    password = models.CharField(max_length=255)  # Adjust max_length as needed
    def save(self, *args, **kwargs):
        # Hash the password before saving
        if not self.password.startswith("pbkdf2_"):  # Avoid rehashing
            self.password = make_password(self.password)
        super().save(*args, **kwargs)
    def verify_password(self, raw_password):
```

```
        # Verify the input password with the stored hash
        return check_password(raw_password, self.password)
    def shared_passwords(self):
        return self.sharedpassword_set.all()  # Assuming a related model named SharedPassword
```

Algorithm 2:

```
from django.db import models
from cryptography.fernet import Fernet
import base64

# Generate a key for Fernet (keep it secret and safe!)
FERNET_KEY = Fernet.generate_key()
cipher = Fernet(FERNET_KEY)
class Password(models.Model):
    password = models.CharField(max_length=255)  # Adjust max_length as needed
    @property
    def decoded_password(self):
        # Decrypt and decode the stored password
        return cipher.decrypt(base64.b64decode(self.password)).decode('utf-8')
    def save(self, *args, **kwargs):
        # Encrypt and encode the password before saving
        encrypted_password = cipher.encrypt(self.password.encode('utf-8'))
        self.password = base64.b64encode(encrypted_password).decode('utf-8')
        super().save(*args, **kwargs)
    def shared_passwords(self):
        return self.sharedpassword_set.all()  # Assuming a related model named SharedPassword
```

VI. CONCLUSION & FUTURE WORK

In today's interconnected world, robust cybersecurity is paramount. A cornerstone of digital security is effective password management. The proliferation of online accounts across various platforms has led to an overwhelming burden on users to remember complex, unique passwords for each. This not only compromises security but also hinders user experience. To address this challenge, we propose a cutting-edge password manager that leverages advanced cryptographic techniques and user-friendly design principles. Our solution aims to:

- **Simplify Password Management:** Automatically generate, store, and autofill strong, unique passwords for every account.
- **Enhance Security:** Employ industry-standard encryption algorithms (e.g., AES-256) to safeguard sensitive information.
- **Enable Secure Sharing:** Facilitate secure password sharing with trusted individuals, such as family members or colleagues.
- **Implement Biometric Authentication:** Offer robust two-factor authentication (2FA) using biometric methods (fingerprint, facial recognition) for added security.
- **Provide Cross-Platform Compatibility:** Ensure seamless access to passwords across various devices (desktop, mobile, tablet) and operating systems.
- **Offer Cloud-Based Synchronization:** Keep passwords synchronized across multiple devices, providing convenient access and automatic updates.
- **Integrate Dark Web Monitoring:** Actively monitor the dark web for compromised credentials associated with the user's accounts, alerting them to potential threats.

By combining these features, our password manager empowers users to maintain a high level of digital security without compromising convenience. It offers a comprehensive solution that protects against the ever-evolving threats in the cyber landscape." Key Improvements expected:

- **Enhanced Security:** Incorporates biometric authentication and dark web monitoring for advanced protection.
- **User-Centric Design:** Emphasizes ease of use and a seamless user experience.
- **Modern Cryptography:** Leverages strong encryption algorithms for robust data protection.
- **Cross-Platform Compatibility:** Ensures accessibility across multiple devices and platforms.
- **Cloud-Based Synchronization:** Offers convenient access and automatic updates.

REFERENCES

[1]. Toaz.info-project-final-year-on-password-management:<https://project-final-year-on-passwordmanagement-systemdocx-4-pdf-free.html.

[2]. Toaz.info-project-final-year-on-password-management:<https://project-final-year-on-passwordmanagement-systemdocx-4-pdf-free.html

[3]. Zhao, R.; Yue, C. All your browser-saved passwords could belong to us: A security analysis and a cloud-based new design. In Proceedings of the Third ACM Conference on Data and Application Security and Privacy,San Antonio, TX, USA, 18–20 February 2013.

[4]. Li, Y.; Wang, H.; Sun, K. A study of personal information in human-chosen passwords and its security implications. In Proceedings of the 35th IEEE International Conference on Computer Communications (INFOCOM 2016), San Francisco, CA, USA, 10–15 April 2016.

[5]. Florencio, D.; Herley, C.; Van Oorschot, P.C. Password portfolios and the finite-effort user: Sustainably managing large numbers of accounts. Proceedings of the 23rd USENIX Security Symposium (USENIX security 14), San Diego, CA, USA, 20–22 August 2014.

[6]. WhatIs.com. 2022. What is a Bootstrap and how does it work?. [online] Available at: <https://www.techtarget.com/whatis/definition/bootstrap> [Accessed 23 September 2022].

[7]. Digitalocean.com. 2022. What is Laravel? | DigitalOcean. [online] Available at: <https://www.digitalocean.com/community/tutorials/what-is-laravel> [Accessed 23 September 2022].

[8]. GeeksforGeeks. 2022. What are Hash Functions?. [online] Available at: <https://www.geeksforgeeks.org/what-are-hash-functions-and-how-to-choose-a-good-hashfunction/> [Accessed 23 September 2022].

[9]. Code Signing Store. 2022. What Is a Hashing Algorithm? A Look at Hash Functions. [online] Available at: <https://codesigningstore.com/what-is-hashing-algorithm- how-it-works> [Accessed 23 September 2022].

[10]. Andrianto, I., 2022. Deno - Base64 Encoding & Decoding Examples. [online] Woolha. Available at: <https://www.woolha.com/tutorials/deno-base-64-encoding-decoding- examples> [Accessed 23 September 2022].