

Vault Shield

A Major project report submitted in partial fulfillment of the
requirement for the degree of

Bachelor of Technology
in
Computer Science and Engineering

By

Ansh Agrawal (201300)

Bhavya Chauhan (201142)

UNDER THE SUPERVISION OF

Mr. Prateek Thakral



**Department of Computer Science & Engineering and
Information Technology**

**Jaypee University of Information Technology,
Waknaghat, Solan - 173234 (India)**

CERTIFICATE

This is to certify that the work which is being presented in the project report titled “**Vault Shield**” in partial fulfillment of requirements for the award of the degree of B.Tech in Computer Science & Engineering and submitted to the Department of Computer Science & Engineering And Information Technology, Jaypee University of Information Technology, Solan is an authentic record of work carried out by Ansh Agrawal (201300) and Bhavya Chauhan (201142) during the period from August 2023 to May 2024 under the supervision of Mr. Prateek Thakral (Assistant Professor (Grade II), Department of Computer Science & Engineering And Information Technology)

Ansh Agrawal
(201300)

Bhavya Chauhan
(201142)

The above statement made is correct to the best of my knowledge

Mr. Prateek Thakral
Assistant Professor (Grade II)
Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Solan

DECLARATION

We hereby declare that the work presented in this report entitled “**Vault Shield**” in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Wanknaghat is an authentic record of our own work carried out over a period from August 2023 to May 2024 under the supervision of Mr. Prateek Thakral (Assistant Professor (Grade II), Department of Computer Science & Engineering And Information Technology)

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Ansh Agrawal
(201300)

Bhavya Chauhan
(201142)

This is to certify that the above statement made by the candidates is true to the best of my knowledge.

Mr. Prateek Thakral

Assistant Professor (Grade II)

Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Solan

ACKNOWLEDGEMENT

Firstly, We express my heartiest thanks and gratefulness to almighty God for His divine blessing makes it possible for us to complete the project work successfully. We are really grateful and wish my profound indebtedness to our Supervisor **Mr. Prateek Thakral, Assistant Professor (Grade II)**, Department of CSE Jaypee University of Information Technology, Waknaghat. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

We would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, We might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, We must acknowledge with due respect the constant support and patience of our parents.

Bhavya Chauhan

(201142)

Ansh Agrawal

(201300)

TABLE OF CONTENT

Title	Page No.
Certificate	I
Declaration	II
Acknowledgement	III
Table of Contents	V
List of Tables	VI
List of Figures	VII
List of Abbreviations, Symbols or Nomenclature	VIII
Abstract	IX
Chapter 1 (Introduction)	1 - 6
Chapter 2 (Literature Survey)	7 - 14
Chapter 3 (System Development)	15 - 32
Chapter 4 (Testing)	33 - 36
Chapter 5 (Results and Evaluation)	37 - 42
Chapter 6 (Conclusion and Future Scope)	43 - 48
References	49 - 50
Appendix	51 - 54

LIST OF TABLES

2.1.1 An Enhanced Web Security for Cloud-based Password Management.....	11
2.1.2 Studying the impact of managers on password strength and reuse.....	11
2.1.3 Providing Password Security By Salted Password Hashing Using BCRYPT.....	11
2.1.4 A comparative survey of Symmetric and Asymmetric Key Cryptography.....	12
2.1.5 Toward a secure and usable cloud-based password manager for web browsers.....	12
2.1.6 Security Analysis of Web-based Password Managers.....	12
2.1.7 ADVANCED ENCRYPTION STANDARD Douglas Selent.....	13
2.1.8 PBKDF2: Password-Based Key Derivation Function 2.....	13
2.1.9 Analysis on the Security and Use of Password Managers.....	13
2.1.10 An investigation into users' considerations towards using password managers.....	14

LIST OF FIGURES

1.1	Vulnerabilities found in existing password managers.....	4
1.2	Some commonly used passwords	5
3.1	Data Flow Diagram.....	18
3.2	AES Encryption process.....	19
3.3	How Asymmetric Encryption works.....	19
3.4	Hashing vs Encryption.....	20
3.5	Code to create a key with email and password.....	21
3.6	Code to encrypt the user's password.....	21
3.7	Code to decrypt the user's password.....	22
3.8	Saved Password in the database (encrypted).....	22
3.9	User in the database.....	23
3.10	Development phase of the Application.....	23
3.11	Server Side Dart Code Snippet.....	24
3.12	Password Model Code Snippet.....	25
3.13	User Model Code Snippet.....	25
3.14	Route Handler Code Snippet.....	26
3.15	Login Route Handler Code Snippet.....	27
3.16	Signup Route Handler Code Snippet.....	28
4.1	Login Fail after user created from app and login from postman.....	36
5.1	Login Screen.....	39
5.2	Signup Screen.....	39
5.3	Home Screen.....	39
5.4	Add Password Screen.....	39
5.5	API Testing of Login Route.....	40
5.6	API Testing of Signup Route.....	40
5.7	API for Get saved Passwords.....	41
5.8	Flutter Web Output.....	41
5.9	Flutter MacOS Output.....	42

LIST OF ABBREVIATIONS

- AES - Advanced Encryption Standard
- RSA - Rivest–Shamir–Adleman
- PBKDF2 - Password based Key Derivation Function
- HMAC - Hash-based Message Authentication Code
- SHA - Secure Hash Algorithm

ABSTRACT

It is impossible to overestimate the significance of strong security measures for sensitive and personal data in the digital age. Managing passwords is essential to cybersecurity since they are the first line of protection against illegal access to a variety of online platforms. The advanced password manager program Vault Shield is described in this abstract. Its goals are to improve user security and simplify the handling of complicated authentication credentials.

Vault Shield provides a complete answer to the problems related to managing passwords. Modern encryption techniques are used by the application to protect passwords that are saved, guaranteeing the privacy and security of user data. An additional degree of security is provided by Vault Shield, which uses a zero-knowledge architecture to guarantee that not even the service provider can read or recover user passwords.

The Password manager software has advanced with Vault Shield, which combines strong security features with an emphasis on user experience. In an increasingly linked digital world, Vault Shield seeks to reshape the norms for safe password management by fusing powerful encryption, biometric verification, and collaborative feature

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

In today's age where everyone has an access to a computer, sensitive data protection assumes significance. Security of passwords is what separates private as well as work online space, therefore password management system has to be efficient in terms of security as well as convenience. Diligence went into creating modern password manager Vault Shield and increasing the standard of digital identity protection.

A comprehensive answer to the myriad problems of changing the internet environment, Vault Shield is an eminent leader in the developing password security market. Vault Shield is more than a regular password manager since it employs modern cutting edge functions like Face ID verification with encryption protection of data using military grade standards.

1.2 PROBLEM STATEMENT

This problem, however, is brought about by the fact that it's quite difficult for people to remember and keep the complex passwords needed today as we have several accounts that depend on our privacy and the security of information.

This should have a strong, safe and user friendly password control system. Such an approach would enable people to easily retrieve their passwords and leave them away from harm's way, especially from intruders. Therefore, the software should have an effective encryption mechanism that can protect its storing of passwords and also be easy for usage. LICENSED In certain cases, it might even complement additional security mechanisms like dual factor authentication.

User should just remember one powerful Master Password through which they may access all other passwords for various accounts, and it is the main line of action. It is hard to remember so many passwords if they are strong enough – that is, one has a different password for each platform/account.

First of all, a different password should be stored securely, and it has to be sent in the cloud such that the user will be capable of making an access over and over regardless of the place. However, since sending of passenger's passwords from

the device to the cloud is a complicated task, which involves multiple security issues, we will have to solve some of them to ensure users password remain protected while in transit. This implies that an attacker is just reading out from network logs hence making it possible for them to steal the users' passwords.

1.3 OBJECTIVE

Vault Shield, a cutting-edge password manager program, aims to revolutionize digital identity management by pursuing a wide range of goals. These goals include:

- **Optimizing password security:** Vault Shield applies latest encryption techniques together with zero knowledge structure to protect user's passwords from any illegal access. This entails adopting RSA encryption that is known to be among the most secured algorithms used. Another unique feature is vault shield used a zero knowledge, an architecture meaning that the program did not store any of the user's passwords. This will protect the application against unlawful use in stealing of passwords.
- **Improving user experience:** Vault shield has user friendly interface that can be used by both untrained and trained personnel. The software boasts of a straightforward and user-friendly interface for creation, storage and management of passwords. Also, vault shield will support a wide range of authentication methods including biometrics and facial recognition and hence easy password retrieval.
- **Enabling seamless synchronization:** Vault Shield is a service that enables individuals to sync their password lists with different devices including PCs, tablets and phones. This makes it possible for users to retrieve their passwords from any device wherever they may be located.
- **Encouraging teamwork:** Users can also work together and share their passwords without having any security fears by using Vault Shield. It is also beneficial for a team with shared confidential information.
- **Providing comprehensive security features:** Vault Shield includes a variety of security features to protect users from cyber threats. These features include:

- **Threat detection:** Intelligent algorithm-based scanners are used in vault shield and identify various security threats. In case a threat is identified, the Vault Shield will alert the user, after which it will guard on the passwords.
- **Password health analysis:** The software, Vault Shield, conducts a complete search on user passwords and identifies any weak or compromised pass code. This makes it possible for them to enforce their digital security measures.
- **Secure emergency access:** Vault Shield has been developed and it offers an emergency access protocol through which the user can obtain his/her password even after losing or stealing of a mobile phone.
- **Secure vault for confidential notes and files:** It is safe to store private notes and files in VAULT shield. The same password-encryption technology is also in place on this vault to safeguard sensitive information.

The product is known as Vault Shield and it is a well-rounded, secure and user friendly password management app. Its state of the art security characteristics and user friendly operation, makes it ideal for both business and individual enterprises.

1.4 SIGNIFICANCE AND MOTIVATION

Due to the increased number of interactions that occur worldwide, Vault Shield became an essential item. To protect the ever-growing volume of sensitive and personal data with increased speed, considering the fast upsurge of online activity and shifting nature of threats. Vault Shield was designed to allow people secure access to their passwords and other sensitive information.

This project seeks to utilize new authentication schemes in conjunction with advanced encryption methods in order to address the weaknesses associated with antique password tools. Accordingly, this belief motivates its provision of password manager solutions that do not sacrifice usability for increased strength in protecting digital identities.

Secondly, Vault Shield provides more dependable and resilient password management services using state-of-the-art encryption and authentication combined with cloud

computing. Cloud computing has several advantages such as being cheap, flexible in addition to a high degree of availability. Cloud technology enables the creation of a highly durable platform that will grow at pace with the users.

Vault Shield has made it its mission to give its users more than what they expect under the dynamic cyber environment. This is an effort aimed at setting a new industry standard for secure digital identification, as well as to let individuals own, manage, and trust their online persona with confidence given that there has never been a better moment in history for internet safety.



Fig 1.1: Vulnerabilities found in existing password managers

The Most Popular Passwords Around the World

Most popular passwords appearing in leaks 2019/2020

	2020	change from previous year	2019
1.	123456*	0	123456*
2.	picture1	new	test1
3.	password	0	password
4.	111111	+ 7	zinch
5.	123123	+ 7	g_czechout
6.	senha**	new	asdf
7.	qwerty	0	qwerty
8.	abc123*	+ 65	iloveyou

* or variation ** Portuguese for password

Source: North Pass



Fig 1.2: Some commonly used passwords

1.5 ORGANIZATION OF PROJECT REPORT

Chapter 1 welcomes readers and gives them a first look at the details of our project, Vault Shield. It also acts as a portal to the rest of our report. In these opening pages, we provide context for our project by outlining the issue statement that guides our work. By outlining the goals that direct our project's path and exploring the reasons behind our dedication to it, this part establishes the foundation for the reader's comprehension.

Chapter 2 is a detailed literature survey on our project, Vault Shield. In these pages, we list all relevant search papers, noting their titles, conference dates, authors and short summaries. The present chapter is a storehouse of information which reviews in detail the existing body of literature and presents the basis for our study.

Chapter 3 offers a gripping narrative which takes us through the intricate making of Vault Shield. In this section, we are going to closely examine the system development process starting by providing a detailed justification of the said project requirements. While entering in the analysis phase, we are very cautious with the details that make up our objectives, which lay the foundations for future stages. Subsequently, a part explaining the Vault Shield's architecture outline is provided on this chapter.

As one of the most important parts of our journey, Chapter 4 reveals the painstaking testing process that encompasses the construction of Vault Shield. In these pages, we expose the complex procedure of testing our application—a crucial point at which the robustness and functionality of the app and the API are closely examined. We openly discuss the difficulties we had while building, accepting that mistakes would inevitably occur and highlighting the critical importance of error correction as we work through the complexities of this stage.

Chapter 5 is our project's apex, a contemplative area where we present the results of all of our combined work in making Vault Shield a reality. We painstakingly break down the project's accomplishments on these pages, providing a thorough rundown of what has been accomplished. The story walks readers through the entire process step-by-step and gives a thorough explanation of how we turned our first ideas into a workable and practical solution.

The story of our project reaches a turning point in Chapter 6, which offers a thorough analysis of the path, major obstacles, and potential futures for Vault Shield. In this chapter, we openly disclose the project's intrinsic constraints and the parameters that our solution works inside of.

CHAPTER 2: LITERATURE SURVEY

2.1 Literature survey

- Olajid and Ridwan Olayinka is a research paper which is a cloud-based password management solution that uses the "Cloud" to store and retrieve web passwords. The goal of the study is to identify any vulnerabilities or security issues with the online application (Adekunle Ajasin University, Akungba-Akoko) by looking at and analyzing its present password management state. This study identifies flaws in current web applications and examines how attackers can use them to decipher passwords that users have saved. The research suggests a unique cloud-based password management system in order to accomplish the intended high level of security.
- Studying the impact of managers password strength and reuse by Sanam Ghorbani Lyastani, Michael Schilling, Sascha Fahl, Sven Bugiel, Michael Backes is the second literature review in which we present the first comprehensive analysis of how password managers affect users' actual passwords in this research. We selected 170 respondents from a 476-person online poll on password creation and management techniques, who gave us permission to track their password usage in real time using a browser plugin. Unlike previous study, we gather not only the passwords and their metrics, but also the password entry methods (e.g., human or password manager). We get a more comprehensive view of the variables influencing the strength and reuse of our participants' passwords based on the information we've gathered and the poll we conducted. For the first time, we quantify the benefits of password managers on password strength and originality; however, our findings also imply that these gains are dependent on the managers' methods and the lack of password generators by users really make the issues worse.
- It provides the method of Salted password hashing with BCrypt by P Sriramya R.A. is Karthika uses salted password hashing approach for securing user data. The use of databases storing customer's data in plain text makes it very dangerous. This scenario is therefore avoided by use of

hashing. This project aims at saving user data in its encrypted form on a database rather than its plaintext form. It uses a Bcrypt algorithm to enhance the security of the user's data. The Bcrypt technique involves up to 512 bits of data encryption with its corresponding hashed value. Hash functions are often used in hash tables to locate data records quickly.

- Another study titled “A Comparative Survey of Symmetric and Asymmetric Key Cryptography” was carried out by Sourabh Chand and Smirtha Paira. Moreover, we have compared the importance of these two cryptographic techniques. The proposed algorithms proved to be highly efficient within their realms although some facets of these algorithms were not fully elaborated. This also presents further prospects for exploring these untapped sectors.
- Towards a secure and usable cloud based password manager for web browsers by Rui Zhao and Chuan Yui identifies the weaknesses in the current BPMs and examine the ways in which attackers could utilize them to break users' passwords that they have saved. Furthermore, in order to attain a high degree of security with the required confidentiality, integrity, and availability qualities, we suggest a novel Cloud-based Storage-Free BPM (CSF-BPM) architecture. The paper describes the installation of a CSF-BPM system in Firefox and assesses its accuracy, efficiency, and usability. The study and evaluation results show that CSF-BPM is a practical and effective tool. Further the paper concludes that CSF-BPM is a sensible concept that may be included into other widely used browsers to improve the security, usability, and pleasure of Web users' online experiences.
- In this paper, Zheiwu Li et al evaluate the robustness and integrity of 5 popular online password management systems at the University of California, Berkeley. Different password managers are different because local password managers function outside a browser and web-based password managers work within the browser. Therefore, using our case studies, we enumerate four prominent security risks inherent in web-based passwords and corresponding common vulnerabilities attributed to each

category. Our assaults are serious: An attacker may obtain a user's credentials for any particular website in all but one of the five password managers we considered. We find flaws in many aspects such as the same passwords, bookmarklets, and so on. The common vulnerabilities include CSRF and XSS while the root causes vary significantly – including those due to logical and authorization errors as well as those that result from improper interpretations of the web security architecture.

- However, Seltent's book only briefly highlights on the latest encryption standard of AES for a current secret key. AES has since replaced DES which was developed by Vincent Rijmen and Joan Daemen, both Belgian cryptographers. It consists of the FIPS PUB 197 that defines the uniform process, which is based on Rijndael variant of the AES Algorithm. It applies substitutions in lines and columns, S box as well as XOR. Further, Mixcolumn and rotations. It worked fine. Two disadvantages only, it's very simple to set up without any problems and works for several days on a conventional computer.
- Password manager-Based key derivation function 2 [S.josefsson, Jan. 2011] gives short account of RFC2898 aka PKCS #5. Several protocol like SCRAM uses hash-message and PBKDF2 However, there were no test vectors defined in the first version, but they are often useful to implementers. This part of the paper discusses information related to that paper.
- Analysis on the Security and Use of Password Managers by Carlos luevanos , John elizarraras, Jyh-haw Yeh, khai hirschi briefly discusses three open-source password managers in this essay, each selected for its own distinctive qualities. We'll wrap up our findings with an analysis of the general security of each password manager using a list of known vulnerabilities and the creation of fresh, hypothetical threats. We will also contrast our findings with the scant prior research on password managers. Lastly the paper offers some broad pointers on how to create a stronger and more secure password manager. Than the paper talks about Businesses and individuals alike fail to implement the stringent password

policies that the NIST (National Institute of Standard Technology) advises. Large-scale security breaches have the potential to reveal thousands or even millions of passwords that are saved in files, making users vulnerable to dictionary and rainbow table attacks. These are just two instances of password-cracking assaults.

- Michael Fagan, Yusef Albayram, Mohammad Maif Hassan Khan and Ross Buck conduct an online survey on 137 users and 111 non-users about how they perceive the use of password manager technology. Moreover, according to various findings in psychological and communication studies, which confirm that emotional response plays a role in other high risk decisions (e.g., use of condom during safe sex), we posed a question about how people that use password managers evaluate 45 different emotions. Otherwise, people without using the tool were requested to assess how they would be able to feel the 45 emotions had they used it. “Users” of password managers pointed out a number of reasons why they should adopt this security measure, but these were not in terms of protection offered to them mainly about utility and convenience. Consequently, many people in the world still consider security as one of the least important factors at any decision-making stage despite having more than half of its users employing highly secure measures at any point and time. “Non-users”, however, pointed at security concerns as the primary reason why they would not engage in password management. This signified widespread distrust resulting from lack of knowledge on how the technology worked.

S.no.	Paper Title [Cite]	Journal/ Conference (Year)	Tools/ Techniques/ Dataset	Results	Limitations
1.	An Enhanced Web Security for Cloud-based Password Management. [1]	Ridwan Olayinka and Ajayi Olusola Olajid (2019)	The paper employed cryptographic (secure hash algorithm) and hellman exchange algorithm for better site security.	Providing a more tight security to the site thereby ensuring secure channels through data flows.	The paper does not discuss any limitations of the algorithms.
2.	Studying the impact of managers on password strength and reuse. [2]	arXiv preprint arXiv:1712.08940 (2017)	Use of data collected from paid workers of Amazon's crowd-sourcing service Mechanical Turk .	password managers from this process indeed benefit the password strength and uniqueness	The paper does not show any appropriate limitation to the paper.
3.	Providing Password Security By Salted Password Hashing Using BCrypt algorithm. [3]	ARN journal of engineering and applied sciences 10.13 (2015): 5551-5556.	Methods to prevent online attacks, by using salted password hashing with bcrypt.	Bcrypt hashing provides a higher level of protection against potential data breaches.	The paper does not discuss any limitations of the BCrypt algorithm.

S.no.	Paper Title [Cite]	Journal/ Conference (Year)	Tools/ Techniques/ Dataset	Results	Limitations
4.	A comparative survey of Symmetric and Asymmetric Key Cryptography. [4]	IEEE International Conference on Electronics, Communication and Computational Engineering (ICECCE) (2014)	The authors compare and contrast the different algorithms based on their security, efficiency, key size, and ease of implementation.	Asymmetric key cryptography is more suitable for secure key distribution and digital signatures.	The paper does not discuss any limitations of specific symmetric or asymmetric key cryptography algorithms.
5.	Toward a secure and usable CLOUD-BASED password manager for web browsers. [5]	Computers & Security 46 (2014): 32-47.	Explains Browser password manager which can potentially address the user authentication and password management problems.	Promotion of CSF-BPM i.e. Cloud based storage free BPM to reduce the vulnerabilities of BPM and protect the data stored in the cloud.	Requiring users to migrate their original passwords to hashed passwords is a biggest limitation
6.	Security Analysis of Web-based Password Managers. [6]	23rd USENIX Security Symposium (USENIX Security 14). (2014)	The paper talks about the features like credential encryptions and login bookmarklets and also studies about some famous password manager applications.	Provide access to credentials and auto-fill functionality using bookmarklets, for instance it talks about the results of Mylogin.	The paper does not discuss any limitations as such.

S.no.	Paper Title [Cite]	Journal/ Conference (Year)	Tools/ Techniques/ Dataset	Results	Limitations
7.	ADVANCED ENCRYPTION STANDARD Douglas Selent. [7]	Rivier Academic Journal Volume 6, Number 2, Fall (2010)	Detailed overview of the Advanced Encryption Standard (AES), including its design principles & implementation details.	the paper shows that AES is a secure and efficient encryption algorithm that is suitable for a wide variety of applications..	AES is still under active research, and new vulnerabilities may be discovered in the future.
8.	PBKDF2: Password-Based Key Derivation Function 2. [8]	IETF RFC (2000)	PBKDF2 is designed to be more secure than previous KDFs, such as MD5 and SHA-1, by using a salt and an iteration count.	The paper shows that PBKDF2 is resistant to brute-force attacks, dictionary attacks, and rainbow table attacks.	However, PBKDF2 is designed to be efficient enough for use in real-world applications.
9.	Analysis on the Security and Use of Password Managers.[9]	Published on 2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)	Detailed overview of the Advanced Encryption Standard (AES), including its design principles & implementation details.	the paper talks about Businesses and individuals alike failing to implement the stringent password policies that the NIST (National Institute of Standard Technology) advises.	Currently no limitation is described in this paper.

S.no.	Paper Title [Cite]	Journal/ Conferenc e (Year)	Tools/ Techniques/ Dataset	Results	Limitations
10.	An investigation into users' considerations towards using password managers.[10]	Published on Human-centric Computing and Information Sciences 7, Article number: 12 (2017)	Previously known as Amazon Mechanical Turk (MTurk), Mechanical Turk is a crowdsourcing platform that lets companies and individuals assign Human Intelligence Tasks (HITs) to a large number of workers (referred to as "Turkers") via the internet. Amazon introduced it in 2005.	Results show that "users" of password managers cited utility and convenience over security benefits as their key motivations for using the program, highlighting the fact that even a sizable fraction of users do not view security as the main benefit when making their selection.	No limitations were found to this research paper as well.

CHAPTER 3: SYSTEM DEVELOPMENT

3.1 REQUIREMENTS AND ANALYSIS

A password manager requires an analysis phase that includes identifying, documenting, and analyzing the requirements of software, functionally, restrictions, and expectations that the software will meet. Here are key aspects to consider in the requirements and analysis of a password manager:

- Reliable and high-speed database. MongoDB is a document-based DBMS using JSON-like documents and dynamic schemas. It is freeware, non-proprietary, and portable. MongoDB is a scale-up product designed to offer high availability and it can be used in many applications such as Web application or mobile application, data warehouse.
- A powerful and quick programming language for the encryption and decryption process.
- Web development uses popular languages including JavaScript and Dart. Dart is a client-side language but JavaScript is not; it is a server-side language. Though many people know about JavaScript, Dart gains ground because of its speed and type-security.
- Idea on handling and using tough and unbreakable cryptography.
- Other supplemental characteristics of the algorithm implementation. Symmetric encryption used for data protection is called RSA. One of the most powerful data encrypting algorithms. The Advanced Encryption Standard encryption algorithm, a 128-bit key. However, the attacker has a very difficult time looking through the encrypted information without a key due to this reason.
- Capacity to encrypt and hash a password and a master password for ensuring that no sensitive information leaks out. The system should have capabilities to encrypt and hash user's and master's passwords, so that no information can leak regarding these. This is the process whereby information or data is converted into a form that cannot be easily understood unless the authorized receivers. The process whereby every kind of data is converted into the form that results in giving a unique sequence of characters that cannot be reverted easily is the simple meaning of hashing.
- Give an excellent random password for the users and let them digest it properly. The system thus generates very strong randomized passwords that the user does

not need to remember. At least 12 characters with Upper case/lower case character, special symbol and numbers, passwords generated using cryptographically secure random number generator.

- Measuring how strong the password is. The function should allow users to produce as well as analyse their password's strength. Password has to be scrutinized for its length, complexity, and regular usage of ordinary words or terms.
- The ability to give the user the capability of retrieving saved passwords from any place – the passwords should be in the cloud. It is also advantageous because users will not be required to recall their respective passwords nor worry about the synchronization of such data across various devices.
- The application should perform well, with minimum or no time taken for encryptions, decryptions as well as hashes. Security aspect is essential because it enables the protection of data from hackers.
- Therefore, as a user, I must be able to navigate through the whole platform/application with ease.
- A mobile device that tests the application requirement.
- Web server to run this particular api in a secure manner.
- Ensure high availability, security and integrity of the password management operation. The service should support many clients and their multiple actions at the same time without failures and fast recovery from outage situations.

Some Analysis that were performed while building Vault Shield were:

- Analysis of the encryption and decryption algorithm: The most efficient encryption, decryption, and hashing algorithms to protect our users' passwords and sensitive data.
- Detailed analysis of the Technical Stack as well as the specific libraries that will be utilized in construction of Vault Shield.
- Security analysis to determine if bruteforce attack or the rainbow tables should affect the passwords and master password of the user should also be performed.
- Develop a self-explanatory and easy to use interface for quick password updating.
- Provide consistent and plain user interfaces as well as graphic design.
- Document all requirements, analysis, and design choices for ease of referral while in development stage.

- Assessment of existing solutions with uncovered weaknesses.
- This entails staying up-to-date on the new and emerging encryptions algorithms and authentication methods.

3.2 PROJECT DESIGN AND ARCHITECTURE

3.2.1 Authentication Process:

User Signup:

- When user registers on his system, the client creates a keypair “privateKey” and “publicKey”.
- The privateKey is stored in the client’s device.
- The password is hashed using Bcrypt algorithm.
- User details along with ‘publicKey’ is sent to the server.
- A key is generated from users email and password which is used to encrypt the users private key and send it to server
- A key is again generated using email and the previously generated key, which is sent to the server, where it is hashed and kept in the database.

User Authentication:

- During login, the user enters their credentials.
- The system generates the above 2 keys again from users email and password.
- The passwordKey is sent to server for verification against the stored hash in the db.
- If the hash matches, the server returns a token and the users encrypted privateKey
- The private key is then stored in the device of user.

3.2.2 Password Encryption and Decryption:

Password Addition:

- When a user adds a password, it is encrypted on the frontend using the users privateKey.
- RSA encryption is used with the user’s publicKey as the encryption key before sending it to the backend.

- The encrypted password is stored in the database.
- Password Retrieval:
- The encrypted password is retrieved by the user when they wish to view the password.
- The privateKey is stored and used to decrypt it on the frontend.
- The user's password is recovered by performing the RSA decryption process using the privateKey as the decryption key for user viewing.

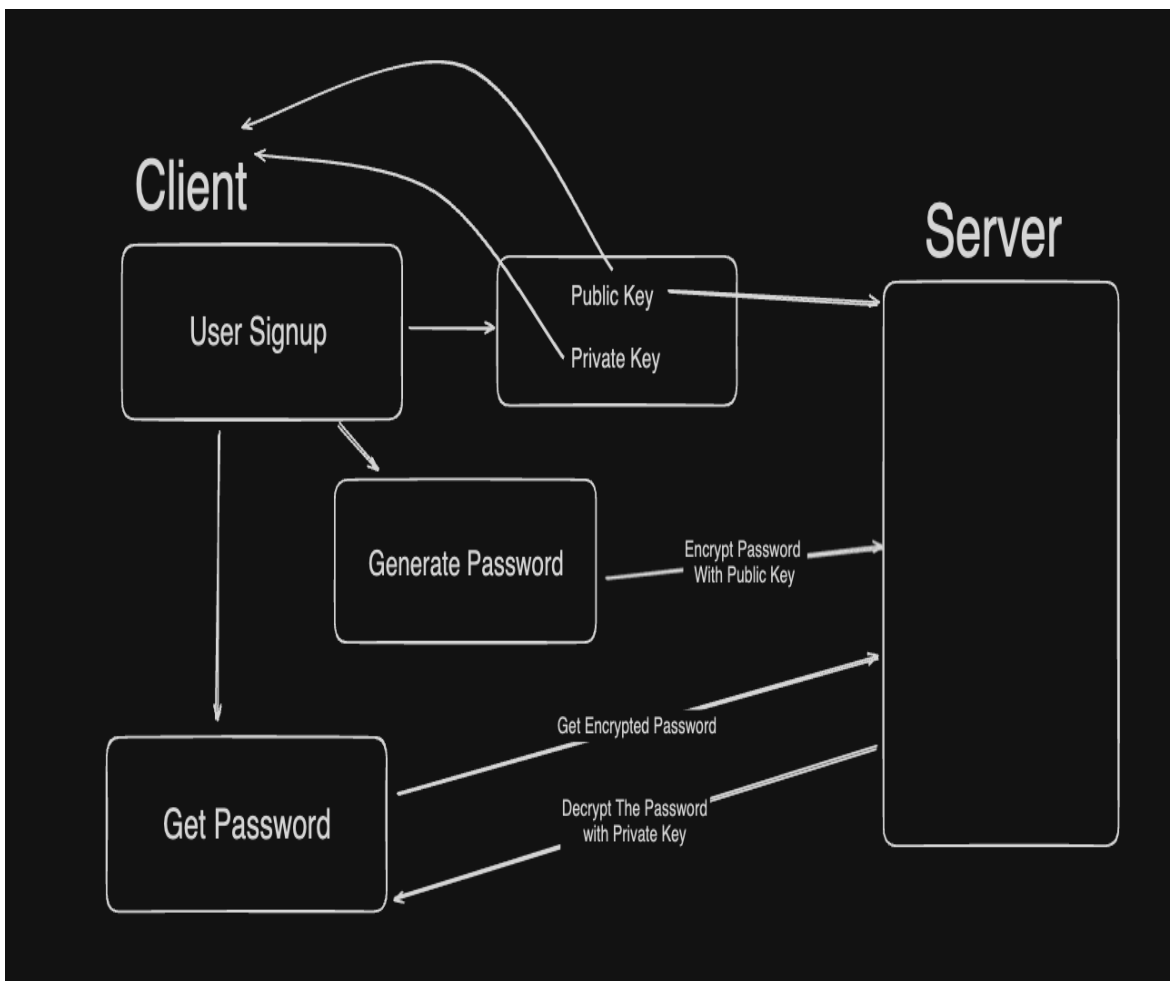


Fig 3.1: Data Flow Diagram

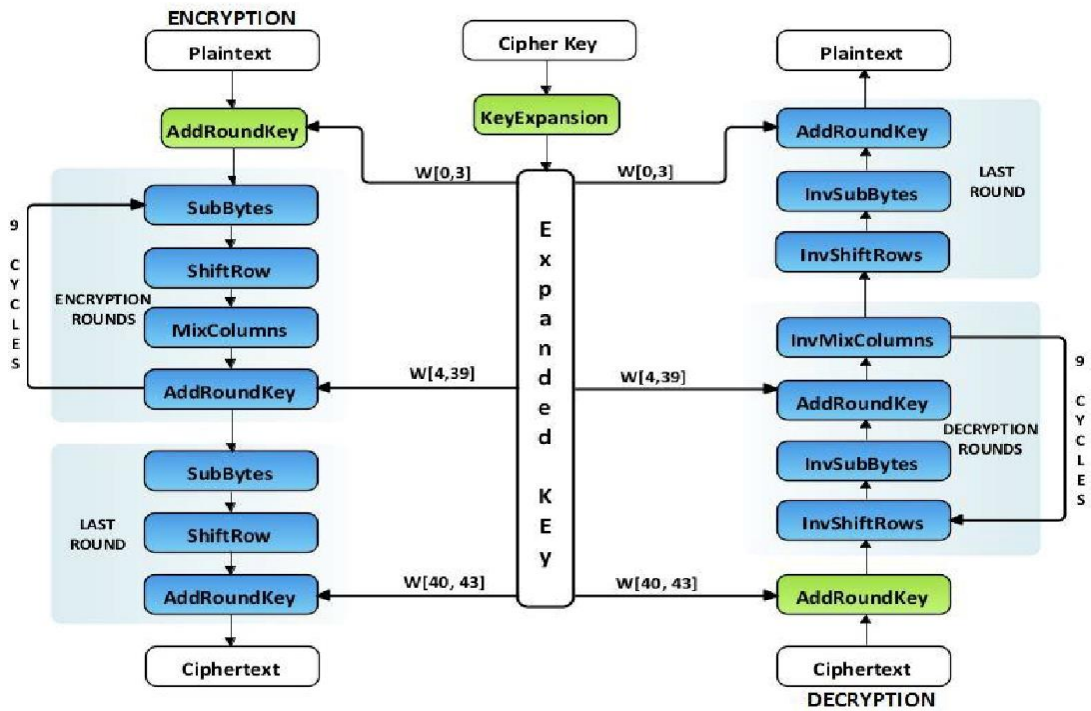


Fig 3.2: AES encryption process

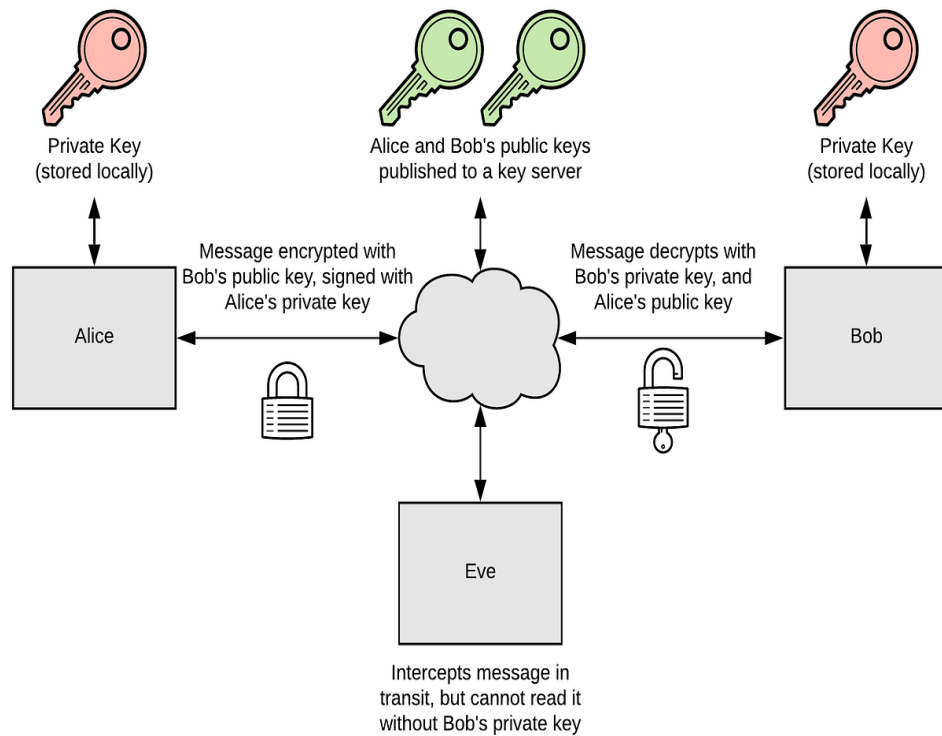
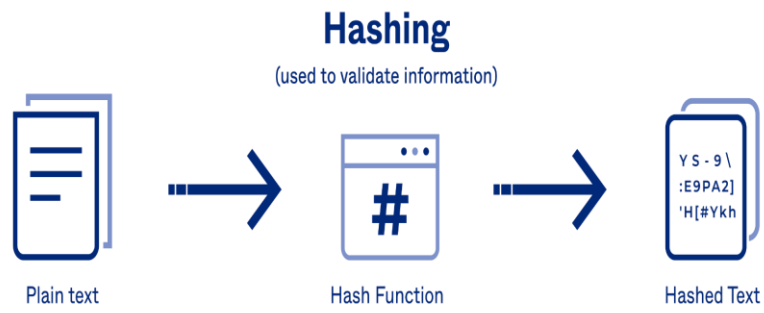
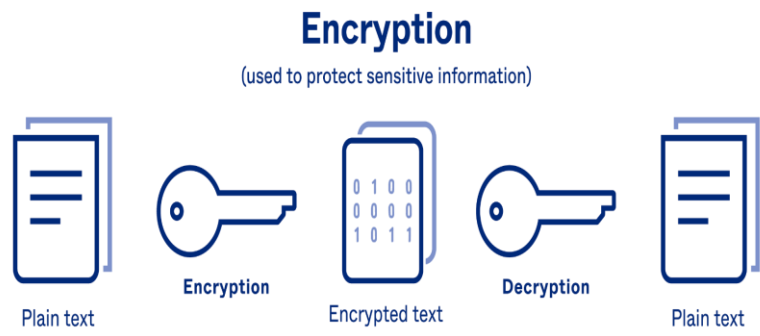


Fig 3.3: How Asymmetric Encryption works



okta

Fig 3.4: Hashing vs Encryption

3.3 Implementation

```
make_encryption_key.dart

Future<String> makeKeyEncryptionKey(
  String email,
  String password,
) async {
  Pbkdf2 pbkdf2 = Pbkdf2(
    macAlgorithm: Hmac.sha256(),
    iterations: 100000,
    bits: 256,
  );

  SecretKey vaultKey = await pbkdf2.deriveKeyFromPassword(
    password: '$email$password',
    nonce: generateSalt(email),
  );
  final bytes = await vaultKey.extractBytes();
  return listToString(bytes);
}
```

Fig 3.5: Code snippet to create a key with email and password

```
JS encrypt.js

Password encrypt(Password password) {
  User user = userSignal.value!;
  if (user.publicKey != null) {
    final publicKey = RSAPublicKey.fromPEM(
      user.publicKey!,
    );
    return password.copyWith(
      password: publicKey.encrypt(
        password.password,
      ),
    );
  }
  return password;
}
```

Fig 3.6: Code snippet to encrypt the user's password


```
_id: ObjectId('663a72707788351d54000000')
name: "ansh"
password: "$2b$10$.YfmFgE8LfInWx2heoqUXe5KhxACV0BgNSxy6aqPZmMLI.e3Kr86m"
email: "a@g.com"
publicKey: "-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQE..."
encryptedPrivateKey: "kEiSwJ/LBmCmbJJXSNROhKh0C5X9AJFLsAkmBww1db2Elnj4YfIi900lE+XtcWly405chy..."
iv: "EmWYuXTVQViiX6D2z9yyFg=="
```

Fig 3.9: User in the database

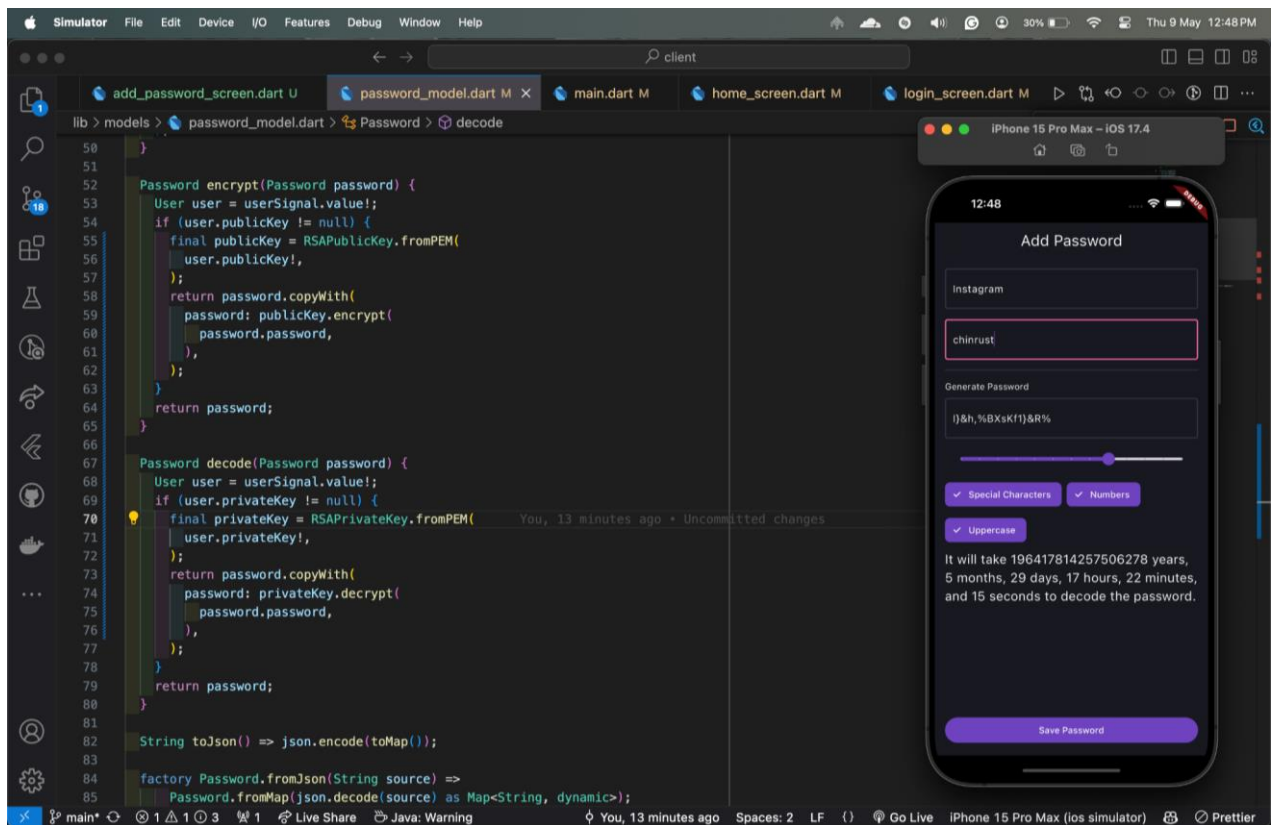


Fig 3.10: Development phase of the Application

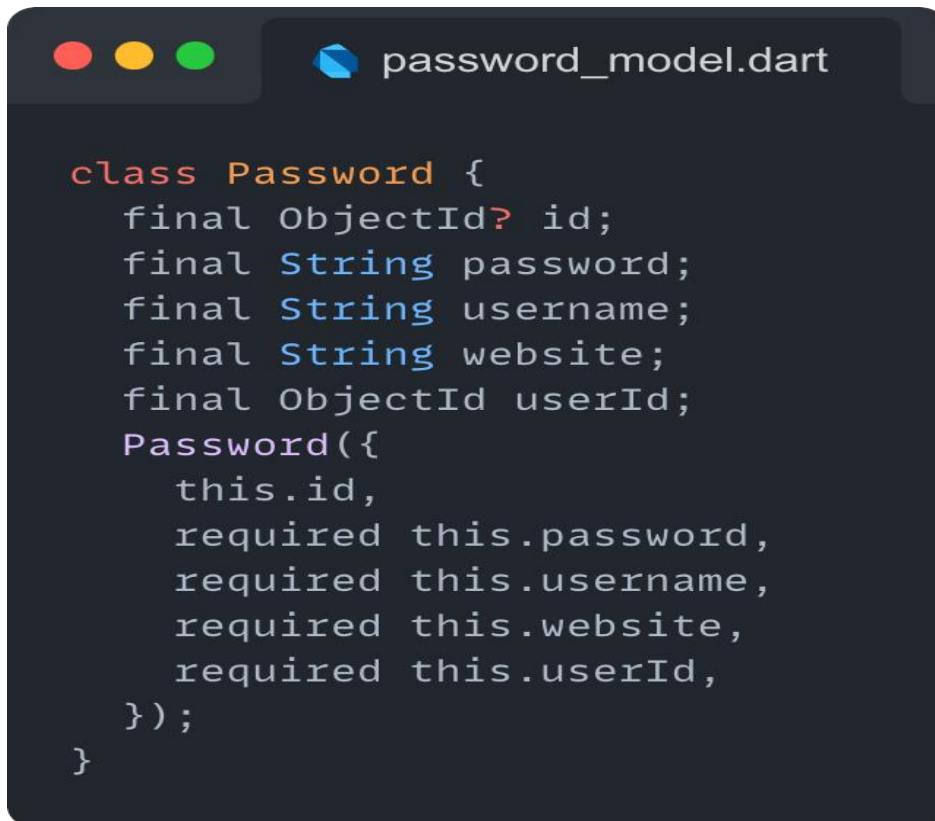
```
server.dart

import 'dart:io';

import 'package:shelf/shelf.dart';
import 'package:shelf/shelf_io.dart';
import 'package:shelf_cors_headers/shelf_cors_headers.dart';
import 'package:vault_shield_server/db.dart';
import 'package:vault_shield_server/service.dart';

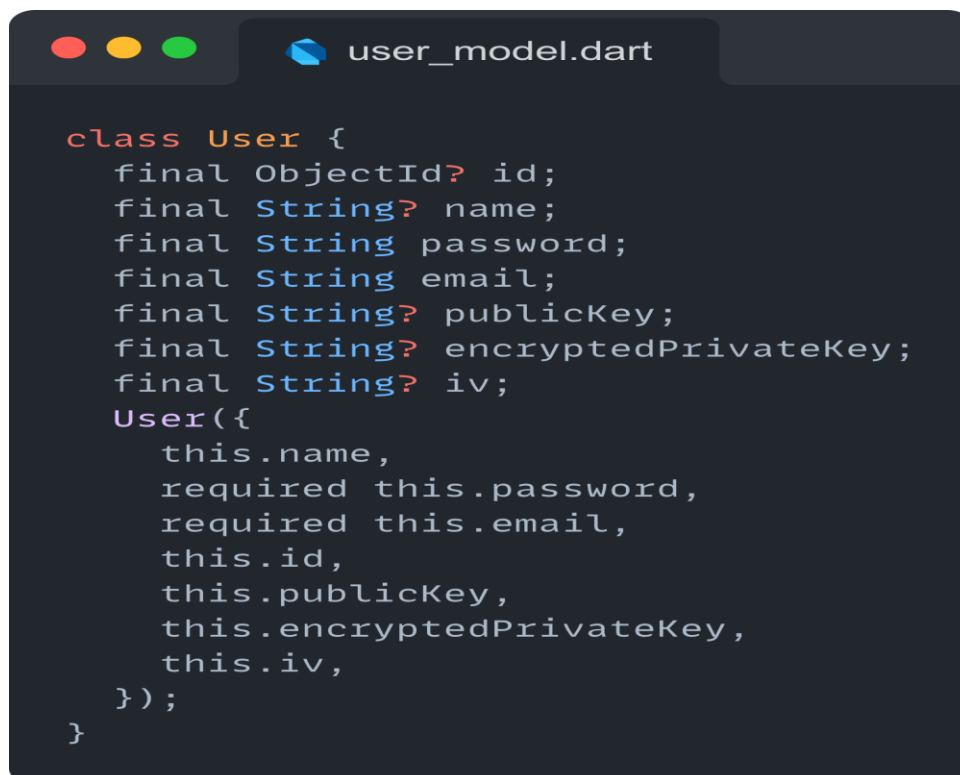
void main(List<String> arguments) async {
  try {
    final ip = InternetAddress.anyIPv4;
    await init();
    await db?.open();
    final PORT = int.parse(Platform.environment['PORT'] ?? '8080');
    final service = Service();
    var handler = const Pipeline()
      .addMiddleware(
        corsHeaders(
          headers: {
            'Access-Control-Allow-Origin': '*',
            'Access-Control-Allow-Methods': 'GET, POST, PUT, DELETE, OPTIONS',
            'Access-Control-Allow-Headers':
              'Origin, Content-Type, X-Auth-Token, Authorization, Accept, X-Requested-With',
          },
        ),
      )
      .addHandler(service.handler);
    final server = await serve(handler, ip, PORT);
    print(
      'Server running at http://${server.address.host}:$PORT',
    );
  } catch (e) {
    print(e);
  }
}
```

Fig 3.11: Server Driver Code Snippet



```
class Password {
  final ObjectId? id;
  final String password;
  final String username;
  final String website;
  final ObjectId userId;
  Password({
    this.id,
    required this.password,
    required this.username,
    required this.website,
    required this.userId,
  });
}
```

Fig 3.12: Password Model Code Snippet



```
class User {
  final ObjectId? id;
  final String? name;
  final String password;
  final String email;
  final String? publicKey;
  final String? encryptedPrivateKey;
  final String? iv;
  User({
    this.name,
    required this.password,
    required this.email,
    this.id,
    this.publicKey,
    this.encryptedPrivateKey,
    this.iv,
  });
}
```

Fig 3.13: User Model Code Snippet

```
route_handler.dart

import 'package:shelf/shelf.dart';
import 'package:shelf_router/shelf_router.dart';
import 'package:vault_shield_server/controllers/login_controller.dart';
import 'package:vault_shield_server/controllers/passwords_controller.dart';
import 'package:vault_shield_server/controllers/public_key_controller.dart';
import 'package:vault_shield_server/controllers/save_password_controller.dart';
import 'package:vault_shield_server/controllers/signup_controller.dart';

class Service {
  Handler get handler {
    final router = Router();
    router.get('/', (Request request) {
      return Response.ok('Hello, World! ');
    });
    router.post('/login', login);
    router.post('/signup', signup);
    router.post('/save', savePassword);
    router.post('/publicKey', publicKey);
    router.get('/passwords', passwords);
    return router;
  }
}
```

Fig 3.14: Route Handler Code Snippet

```
login_handler.dart

import 'dart:convert';

import 'package:dart_jsonwebtoken/dart_jsonwebtoken.dart';
import 'package:bcrypt/dbcrypt.dart';
import 'package:shelf/shelf.dart';
import 'package:vault_shield_server/custom_response.dart';
import 'package:vault_shield_server/models/user_model.dart';

Future<Response> login(Request request) async {
  print('/login ');
  Map<String, dynamic> body = jsonDecode(await request.readAsString());
  try {
    if (body['email'] == null || body['email'] == '') {
      return badRequestJson({'error': 'Email is required'});
    }
    if (body['password'] == null || body['password'] == '') {
      return badRequestJson({'error': 'Password is required'});
    }
    print('${body['password']} ');

    User? user = await User.findOne(body['email']);
    if (user == null) {
      return badRequestJson({'error': 'User does not exist'});
    }
    if (!DBCrypt().checkpw(body['password'], user.password)) {
      return badRequestJson({'error': 'Invalid password'});
    }
    return okJson(
      {
        'message': 'Login successful',
        'token': JWT(
          {
            'email': user.email,
            'name': user.name,
            'id': user.id,
          },
          ).sign(SecretKey('secretKey')),
          ...user.toMap()..remove('password')
        },
      );
  } catch (e) {
    return internalServerErrorJson({'error': e.toString()});
  }
}
```

Fig 3.15: Login Route Handler Code Snippet

```

import 'dart:convert';

import 'package:dart_jsonwebtoken/dart_jsonwebtoken.dart';
import 'package:dbcrypt/dbcrypt.dart';
import 'package:mongo_dart/mongo_dart.dart';
import 'package:shelf/shelf.dart';
import 'package:vault_shield_server/collections.dart';
import 'package:vault_shield_server/custom_response.dart';
import 'package:vault_shield_server/models/user_model.dart';

Future<Response> signup(Request request) async {
  Map<String, dynamic> body = jsonDecode(await request.readAsString());

  try {
    if (body['email'] == null || body['email'] == '') {
      return badRequestJson({'error': 'Email is required'});
    }
    if (body['password'] == null || body['password'] == '') {
      return badRequestJson({'error': 'Password is required'});
    }
    if (body['name'] == null || body['name'] == '') {
      return badRequestJson({'error': 'Name is required'});
    }
    if (body['publicKey'] == null || body['publicKey'] == '') {
      return badRequestJson({'error': 'Some error occurred, please try again'});
    }
    if (body['encryptedPrivateKey'] == null ||
        body['encryptedPrivateKey'] == '') {
      return badRequestJson({'error': 'Some error occurred, please try again'});
    }

    User? user = await User.findOne(body['email']);
    if (user != null) {
      return badRequestJson({'error': 'User already exists'});
    }

    String hashedPassword =
      DBCrypt().hashpw(body['password'], DBCrypt().gensalt());
    user = User(
      email: body['email'],
      password: hashedPassword,
      name: body['name'],
      publicKey: body['publicKey'],
      encryptedPrivateKey: body['encryptedPrivateKey'],
      iv: body['iv'],
    );
    // save user to db
    WriteResult result = await users.insertOne(user.toMap());
    if (result.writeConcernError != null) {
      return internalServerErrorJson(
        {'error': 'Error saving user to DB: ${result.writeConcernError}'});
    }
    return okJson({
      'message': 'User created successfully',
      'token': JWT(
        {
          'email': user.email,
          'name': user.name,
          'id': user.id,
        },
        ).sign(SecretKey('secretKey')),
        ...user.toMap()..remove('password')
      ));
  } catch (e) {
    print(e);
    return internalServerErrorJson({'error': e.toString()});
  }
}

```

Fig 3.16: Signup Route Handler Code Snippet

Tools:

- **Visual Studio Code:** the most popular editor used by web designers. A lightweight app comes with the facilities such as syntax highlighting, code completion, and also debugging tools.
- **Postman:** A tool for testing APIs. It is therefore suitable for sending request of an API and viewing its reply. API testing is an important thing to have and it is how you ensure that the API always plays the right way around.
- **Android Studio and Android Emulator:** It is one of the android development environments. Different build tools, testing kits, and a debugger are included in developing Android apps using the Android Studio. The second approach is when one uses android emulator and no need to purchase an android phone device but one can comfortably download and install these applications on your PC just similar to those of your cell phone.
- **Snappify.com:** Programming in chunks at one place. An app, Snappify, which would enable easy recall of some of the frequently used blocks of codes.
- **Globe.dev:** For deploying Flutter and Dart application
- **MongoDB Atlas:** To host DB online
- **MongoDB Compass:** To visualize data locally
- **iOS Simulator:** For testing Flutter apps on iOS

Additional information:

- Visual studio code is a free, open source code editor. It works with Windows, Mac OS, and most modern Linux distributions.
- Postman is a freemium tool. Basic plans are free; however, the premium plans have collaboration tools and support.
- An efficient, free tool called Android Studio. It can be installed on Windows, Mac or linux.
- The Android Emulator is a free tool. It can be used on window, OS, and Lin.

Tools:

- RSA for encrypting user's passwords
- PBKDF2 (password based key derivation function 2) keys for encrypting private key and for authentication
- Flutter for Frontend, Dart for Backend

3.4 Key Challenges

Enhanced encryption and key management support

This technology supports advanced encryption for keys.

To save user password in a secure way in DBMS, planning for user's authentication & encryptions/decryption procedure. Password encryption and RSA/PBKDF2-based authentication.

Symmetric key RSA is one of the most secure encryption algorithms existing today. An instance of such key derivation function called as PBKDF2 derived a cryptologic key from a password. Combining PBKDF2 and RSA leads to high-level protection for user passwords.

Secure Password Storage

One keyEncryptionKey and authentication Key generation process. Generate a keyEncryptionKey that will securely save the user's privateKey. The key (authkey) to authenticate the user to the server.

On the other hand, authKey, also referred to as secret key, verifies a user's identity against the server. The secret key called the vaultKey is what encryptes and decryptes users' passwords. The keyEncryptionKey is not stored in the server. It is stored only in your device. This guarantees that all passwords of the users are never saved as plain text on the server.

Cross-Platform Compatibility and Accessibility

It should be adaptable using any platform either android, ios, or even as a web application. Moreover, these potential users must be able to login into their passwords even remotely; in fact this can be done by the hand of all devices linked to the internet.

They are a cross-compiled platform framework hence can be used to compile and run on the various devices. It, therefore, makes retrieval of password easier since a user does not have to bother if he/she is on which kind of computer, be it desktop or a mobile device.

Create a design plan for a responsive interface that changes according to screen size and orientation.

Key Management and Key Derivation

Using PBKDF@ for keys generation, privateKey storage at the front-end system.

PBKDF2 is a hashing technique designed for creating a passphrase-generated encryption code. A highly iterated PBKDF2 is used in deriving the keyEncryptionKey value, thus creating much trouble to a potential perpetrator who may want to use the algorithm itself or employ brute force attacks for purposes of recovering the secret key. The user's device stores the privateKey in its encrypted form human written.

Ensuring Comprehensive Testing for Security

Crashing a mobile application by testing it from various test cases. At no time does the app compromise security in the system.

Different types of test cases are used in the application's testing and are carried out on the system to make sure its security. These test cases include:

- Functional testing: It assures that the app is working correctly.
- Security testing: Such kind of testing makes sure that the app is safe with no loopholes or chances of hacking.
- Performance testing: These tests determine how fast an application works and if it exhibits any lags.

There are various devices and operating systems that are used in testing this application to make sure that all platforms are supported.

CHAPTER 4: TESTING

4.1 Testing Strategy

It is important to have effective and trusted testing of password management apps in terms of its safety, security, and ease of use. Here's an outline of the testing strategy and tools that could be employed:

- **Testing encryption/decryption algorithm:** It includes verifying the accuracy and safety of the encryption and decryption programs. One can test against known plaintext and ciphertext and check for weaknesses like timing attacks and side channel attacks.
- **Validating key-derivation functions:** Cryptographic keys are generated from a password or secret using key derivation function. This implies testing of its functions to ascertain that it produces reliable results, without creating vulnerabilities.
- **Testing backend and frontend functionalities:** This refers to ensuring that the parts of the software including the database, web server and user interface are all tested. This may involve manual testing of the application or by employing other automated testing tools.
- **Testing API endpoints for proper communication:** The points of interaction between the front end and back end of an app is known as API endpoints. These endpoints must, therefore, be tested to see if they are functioning and whether they are safe.
- **Ensuring secure data transmission between frontend and backend:** Testing the data transmission between the front end or interface and backend of the app. Such can be achieved through data verification for corruption and lost packets as well as other transmission anomalies.

While testing, we also encountered a problem: For example, what happens if the user loses his/her master password or it gets breached. Together we sat down, brainstormed, and researched on the issues and got some answers. The first solution was that we could apply asymmetric encryption, however, this solution had a problem that was not feasible for this type of encryption across devices. Having the public and private keys in the database exposes this system and it can be easily hacked by anyone who is familiar with these two cryptographic codes. Another one of our solutions we came up with was running a safe database creating an access code after every user creates an account. In

addition to this, two-factor authentication would make our system more secure.

We went forward with storing the public key and private key of the user in the Database, in encrypted format so that only the user can access it. During the signup process, we also generate a recovery key which can be used in case the user loses his/her master password. The recovery key is used to encrypt the private in the Database.

After we were done encrypting the password and private key from the app, we were no longer able to use the postman as the postman would not be able to encrypt the password.

While testing, we also ran into the problem that every time the user was being fetched which was a DB read operation. To reduce this, we introduced JWT token which contained essential information about the user like id which helped us to identify the user without making a DB call.

Tools Used:

- Visual Studio Code which is sort of an Electron JS based text editor could be used like the themes of “coded”.
- Using a widespread tool called Flutter SDK created with Dart, helps to create cross platform apps.
- With regard to this, we could save our data using MongoDB which is a NOSQL database.
- Git/Version control for our application code – This is the name of the software to help us monitor every possible modification we have done to our app’s source codes.
- Also, we used postman in Apis testing.
- Globe.dev for deployment of frontend and backend.
- Android Emulator for testing Android app
- Ios simulator for testing iOS app
- Arc browser for testing the Web App.
- MongoDB compass for viewing the DB while development.
- Pub.dev for packages and dependencies of Frontend and Backend

4.2 TEST CASES AND OUTCOMES

Authentication Testing:

Test case: Attempting login with correct credentials.

Expected outcome: Successful authentication, granting access to the user's vault.

Test case: Attempting login with incorrect credentials.

Expected outcome: Authentication failure, denying access to the user's vault.

Test case: Adding a password to the vault.

Expected outcome: Password is encrypted using the vault key and stored securely in the database.

Test case: Retrieving an encrypted password from the database.

Expected outcome: Authentication failure, denying access to the user's vault.

Test case: Attempting unauthorized access to stored passwords.

Expected outcome: Robust authentication mechanisms prevent unauthorized access to stored passwords.

Test case: Testing encryption strength by decrypting passwords without the vault key.

Expected outcome: Inability to decrypt passwords without the correct vault key.

Test case: Accessing the user data MITM

Expected outcome: Access denied and also can't read the passwords.

Test case: Try to login user from postman using master password

Expected outcome: Access denied, the password is incorrect

Test case: Login user from postman after the user was created from application

Expected outcome: Access denied as password is incorrect.

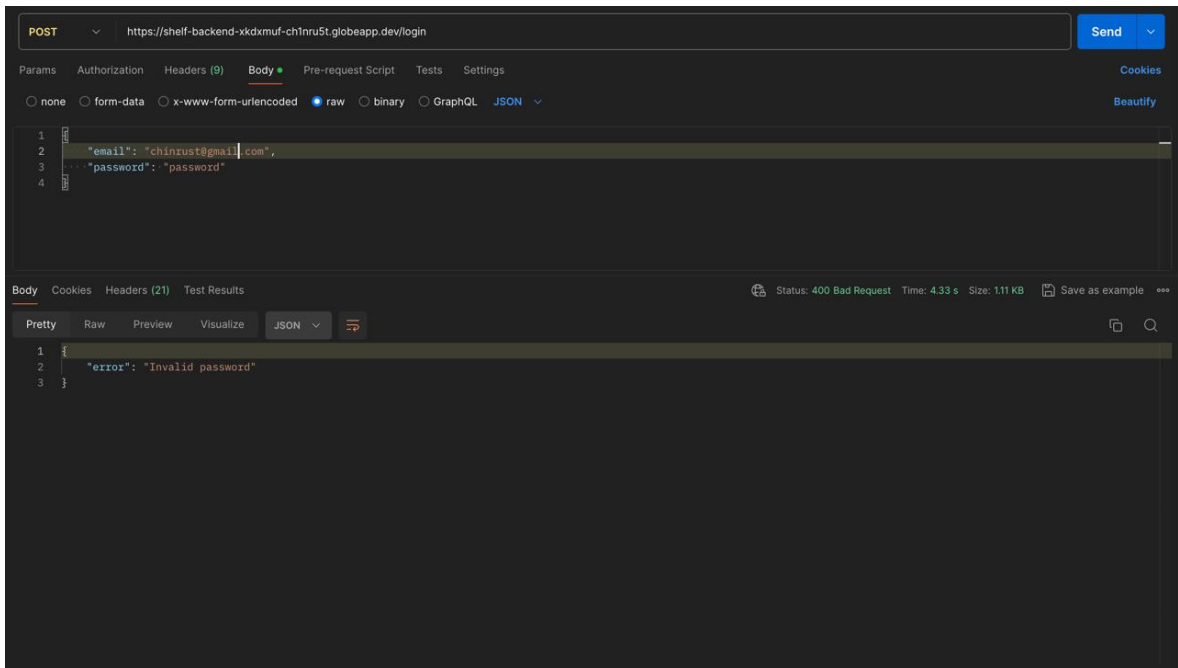


Fig 4.1: Login Fail after user created from app and login from postman

CHAPTER 5: RESULTS AND EVALUATION

5.1 RESULTS

PROJECT GOALS ACHIEVED:

Secure Authentication:

- It employed various robust authentication methods, including tough passwords.
- Centralize authentication to ensure that only users properly identified can access sensitive data.

Encryption & Decryption:

- This is why we employed strong encryption during our password storage process, and decryption was provided with the user's access rights. The site employs a very strong RSA encircryptment (256-bit), which is said to be among the most reliable encryption protocols so far. Passwords are encrypted before they are loaded into the database, and it is only when a user tries to gain entrance by accessing login details that decryption occurs. It helps make it impossible to store these as text and makes it vulnerable to any hacking attack.
- In addition to encrypting the passwords, I also implemented a number of other security measures, such as:
 - Salting the passwords: The password is then mixed up with a random combination of symbols before it is encrypted. As a result, it is harder for an attacker to guess the password even when he could know that.
 - Using a secure hash function: Passwords are hashed and then encrypted. The attacker thus can't recover the original password even when he understands what encryption has been used.

Usability & Accessibility:

- Designed an intuitive user interface that works on different devices to simplify password management. Make it easy-to-use and easy-to-understand by user.
- Interface was designed such that even non-technical users can use it.
- It had provided a password generator as well as another password strength tester among other things that would assist users in creating as well as administration of

strong passwords. Passwords provided by Vault Shield's Password Generator can be considered one of the most secure options and can be made any time desired. According to the user's preference users are able to develop special complex long passwords containing different letter and symbols. Another feature of a password strength checker is instant feedback that allows users to evaluate the power of their existing passwords. The combined skills in this package promote the production of active and accelerated passwords from the app.

Remote Access:

- This means that whenever the user will wish to access their passwords, he/she will manage it from anywhere.
- Such capability was achieved courtesy of symmetric encryption as well as storing the password in an encrypted form in the database.

IMPACT AND BENEFITS:

Enhanced Security

- **Strong password encryption:** These are highly encrypted by a 256 bit RSA key that can only be decrypted by the same key.
- **Regular security audits:** For this reason, our organization carries out periodic assessment to ensure the security systems used by our firm are up-to-date.

User Convenience:

- **Simplified password management:** The users are able to formulate, retain and keep track of their passwords at a single spot.
- **Accessibility across devices:** Users are able to retrieve their password from any device be it their laptops, iPads or mobile phones.
- **Increased security:** Strong encryption and two-factor authentication also allows users to safeguard their login credentials.
- **Peace of mind:** Users can have peace of mind since the credentials they used as well as those for login are safe.

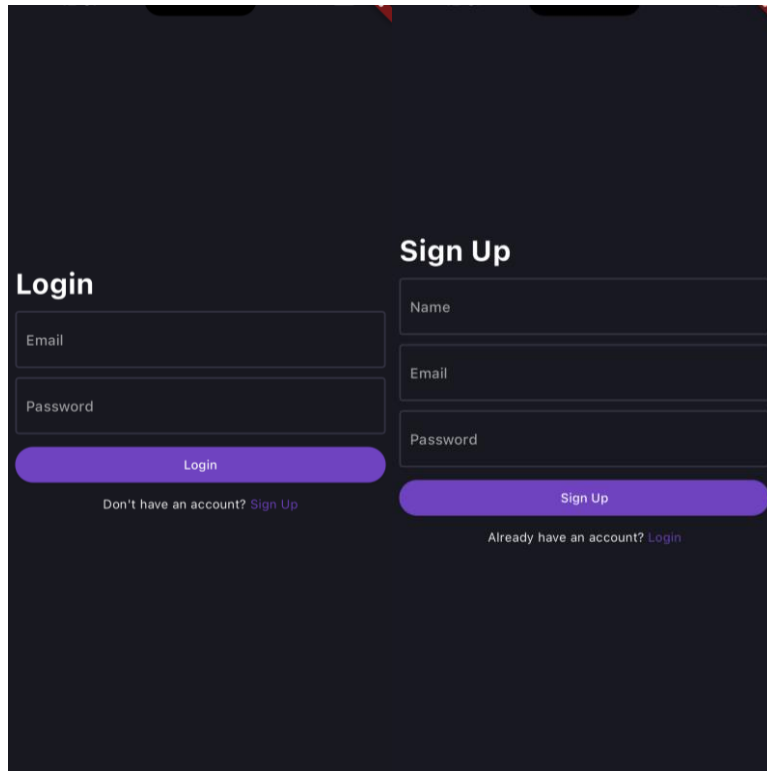


Fig 5.1: Login Screen

Fig 5.2: Signup Screen

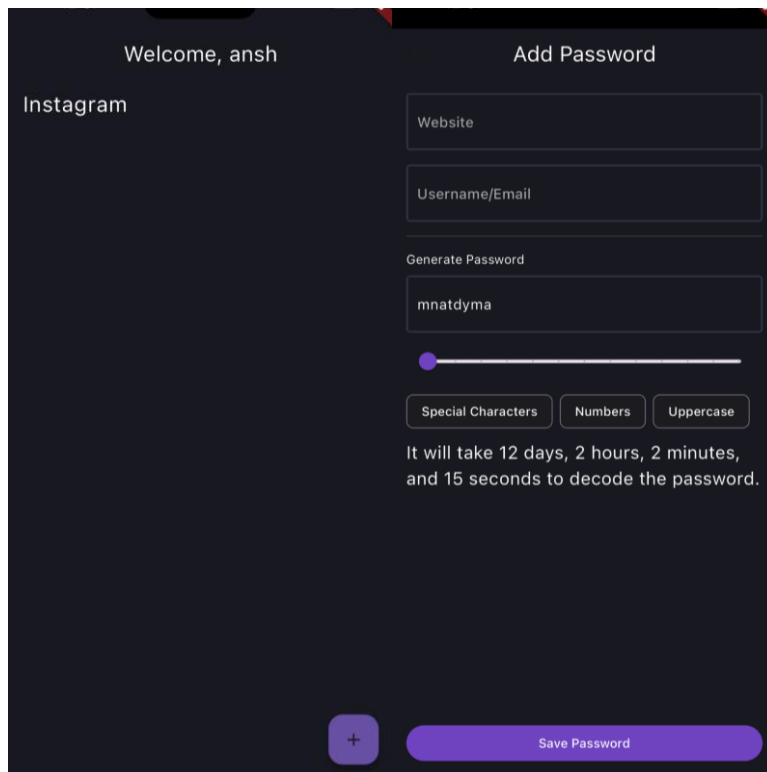


Fig 5.3: Home Screen

Fig 5.4 Add Password

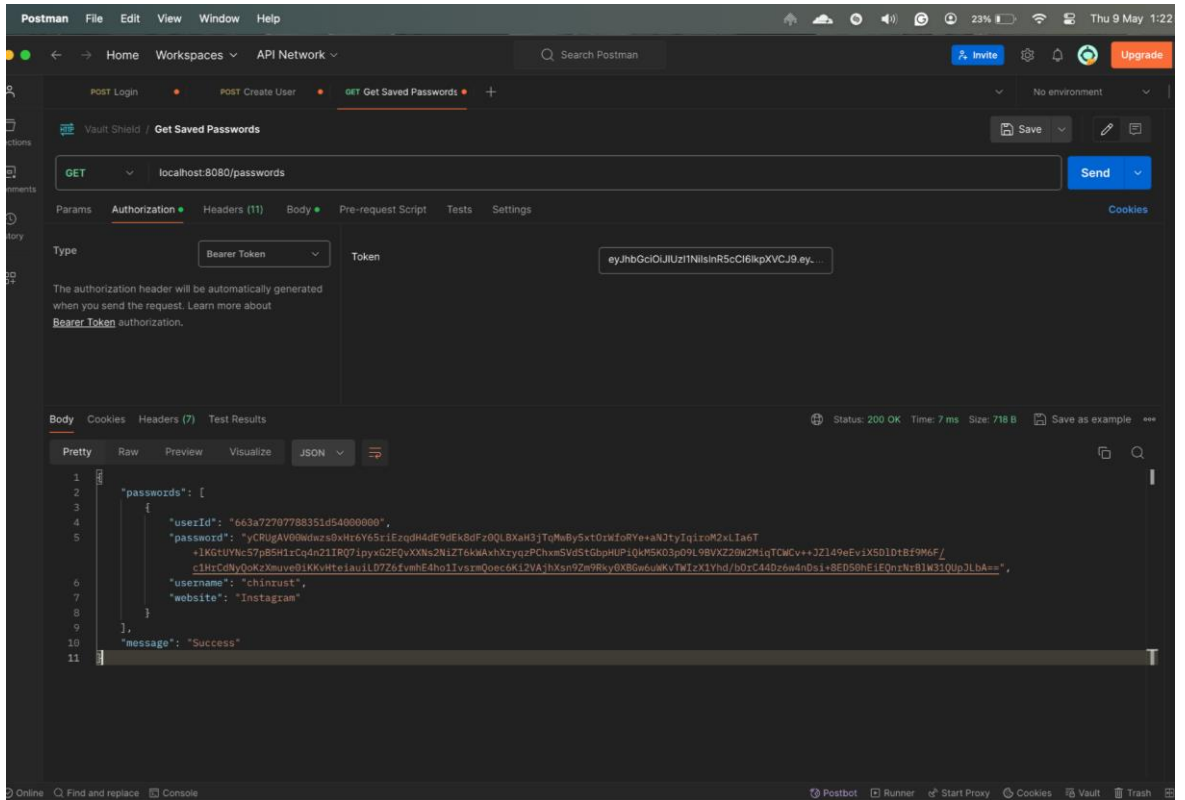


Fig 5.7: API for Get Saved Passwords

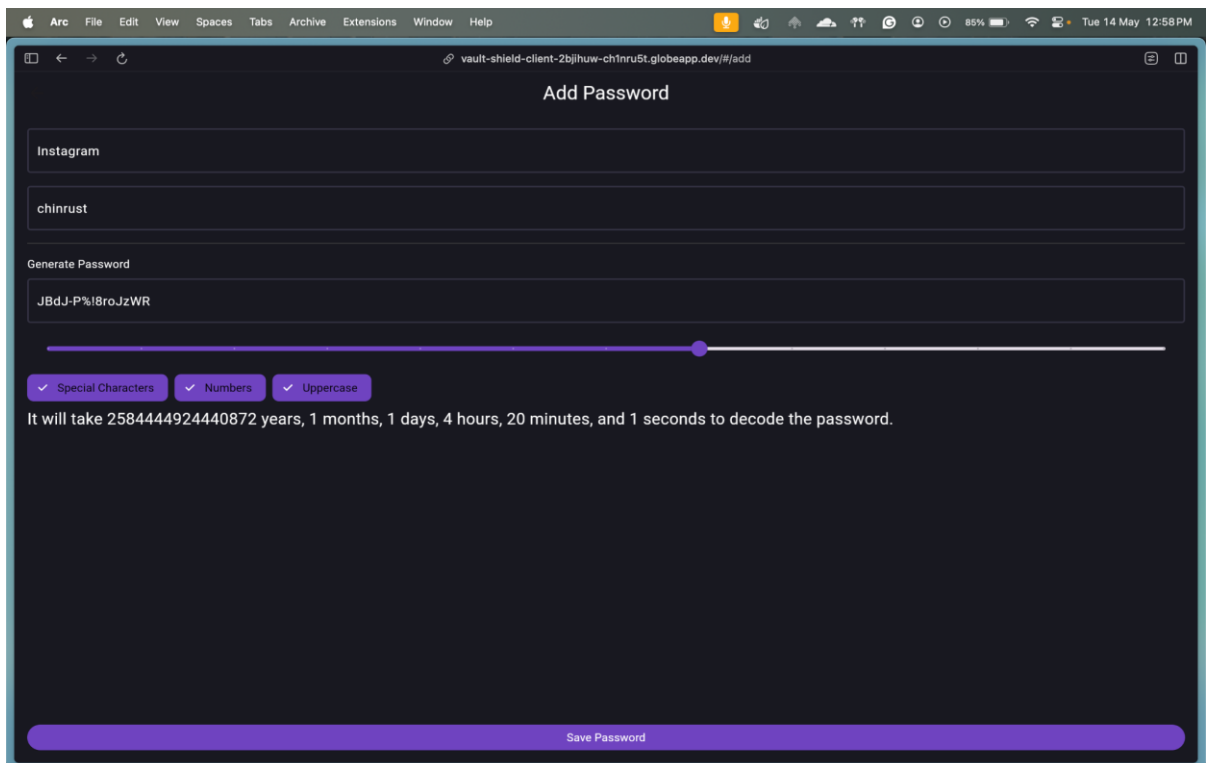


Fig 5.8: Flutter Web Output

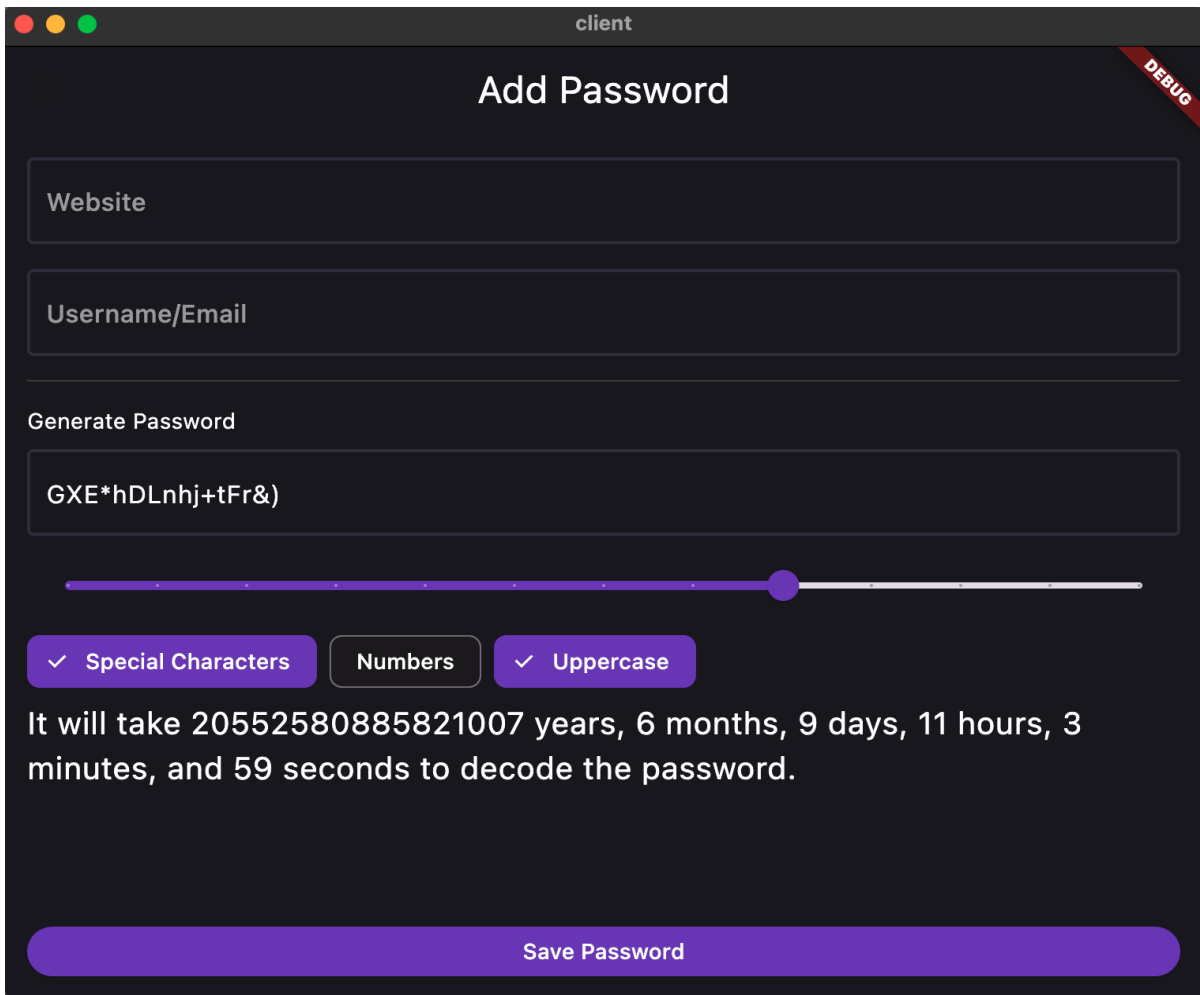


Fig 5.9: Flutter macOS Output

CHAPTER 6: CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

Finally, this password manager application constitutes one of the most useful, safe, easy and trustable solutions offering password management facility to its users. Despite the fact that it brings progress on aspects of security and usability, ongoing improvements as well as adaptations remain vital to remain ahead of the game on the issue of password management technologies.

6.1.1 KEY FINDINGS

- **Security Advancements:** The new password manager implements robust encryption and authentication mechanisms to ensure secure password management. These mechanisms include:
 - **Encryption:** All stored passwords are encrypted with strong encryption methods like RSA. However, in case of a breakdown into the password manager's databases that contains many opened versions of these kept secret passwords.
 - **Authentication:** Users need encryption of their passwords for them to log into the website either through a strong password or their biographic traits. It therefore makes sure that only an authorized person reads the password.
- **Usability Enhancements:** The new password manager offers an intuitive and accessible user interface that is easy to use across diverse platforms. These enhancements include:
 - **A user-friendly design:** Even a user who is not comfortable with technology can easily use this interface. The terminology of its user interface is extremely simple and direct.
 - **Cross-platform compatibility:** It is cross-platform software that can operate on window's macOS os, linux , ios and android. A common password is used by different gadgets which makes it easy for the users to get their passwords while using different gadgets.
- **Performance and Reliability:** The new password manager demonstrates consistent performance even under high user loads. This is due to:
 - **A scalable architecture:** This is a password manager that functions on an

adaptive design to handle billions of account credentials. This ensures that the password manager will continue functioning under pressure – even as many people would be logged in without affecting its speed or efficiency.

- **A robust infrastructure:** The password manager is built within robust infrastructure that minimizes the likelihood if any failures happen. This means that even when there is some unexpected issue relating to the server, or power outage, the users may access their passwords.

All in all, there is no doubt that the updated password manager beats its former. It also makes security enhancement, usability enhancements and secondly, it enhances performance reliability. These changes make the current password managers work better, easier and much safer when handling this aspect.

6.1.2 LIMITATIONS:

- **Scope:** Although the software deals with significant issues related to password management including producing strong passwords, safely storing passwords, and auto-filling passwords; there are additional attributes that would improve its usefulness. For instance, the program could have a facility through which one can send their passwords in a hidden manner or a mode where users are able to monitor the pattern in their use of passwords.
- **Remaining 2-factor authentication:** The application lacks 2-factor authentication that could take it to a higher level of security. Two factor authentication requires the use of two verification means by users in order to login into their accounts, e.g., password and a message sent through sms texting in their mobile devices. Such a situation will make the work of attackers so difficult that they cannot get hold of the users accounts even with the stolen passwords.

In conclusion, the app can be referred to as a great device for password storage in general. Nevertheless, in the next releases there exist a few limitations to be taken care of.

Enhanced Security Measures:

- Implemented advanced encryption procedures in order to boost password security.

- Developed a passwordless authentication system, as opposed to the conventional password-based authentication, that's much safer.
- Constructed a security model that pinpoints as well as resolves security issues.

User-Centric Design:

- Introduced a user-friendly design which works for different types of users.
- Made the platform easy to use and navigate.
- Provided clear and concise instructions.
- Offered a variety of support options to help users troubleshoot problems.
- Gathering insights on how users manage passwords, their pain points, and what features they find most valuable.
- Developing user personas based on the research gaps to create an understanding of the primary user groups of this project.

6.2 FUTURE WORK

6.2.1 More enhanced security features:

Biometric Authentication:

Implementation of biometrics such as fingerprint scanning or facial recognition coupled with Vault Shield could lead to a more secure and convenient experience.

- Multi-Factor Authentication (MFA): Biometric authentication as additional security measure to traditional password based access systems.
- Provide fingerprint authentication option that will allow more security for the users' Vault Shield accounts even when hackers have access to the passwords.

Blockchain Integration:

Explore how block chain can improve password encryption and authentication technologies. Blockchains is a secure-by-design distributed information platform which could store important information like passwords.

- Using blockchain's property of distribution, store password data on various points of an interconnected network so as to ensure there is no single point for compromise.
- Use blockchain's consensus and cryptographic hashing mechanisms to prevent modification and unauthorized access of password data.

2-factor authentication: In this way, the implementation of 2-factor authentication would also take this security aspect to a new high level. Two factors of it would have been 2-FA you would have had to provide two pieces of evidence that could be either passcode or password. So it is very hard for somebody to get an entry into your account through the right password.

- Password + One-Time Codes: Ask the users to provide a single time password sent by mail or SMS or generated using an authenticator.
- This multi-layered technique creates an additional protective layer on top of the master password so as to reduce chances of unauthorized access.

6.2.2 Usability and Accessibility:

Cross-Platform Integration: Optimization should be more enhanced towards easy usage across other devices and different operating systems. This will help in developing a universal software that will work on mobile phones, tablets and PC.

- It is compatible with Linux, Mac OS, and windows. Compatibility: Provide an optimized desktop and laptop experience by making specialized desktop software compatible with most operating systems.
- Take advantage of platform specific features and make sure that feature set is the same in different conditions.

Accessibility Features: Make sure that people with visual disability, hearing problems or those whose mobility is a problem are able to access and use information technology on this site. This incorporates the application of keyboards, and many more.

- **Text-to-Speech Functionality:** Ensuring that other types of devices such as screen reader software like voice over for iOS and talkback for Android support people without sight to browse within text-to-speech mode.
- **Semantic Markup:** When writing for your program, ensure that you use semantic

markups for every one of its elements so it will know exactly what to say when it reads.

6.3.3 Advanced Encryption Techniques:

Post-Quantum Cryptography: Research and adopt encryption methods resistant to attacks from quantum computers. Quantum computers are a new type of computer that is much more powerful than traditional computers. They could be used to break current encryption methods, so it is important to start developing new methods that are resistant to quantum attacks.

- Investigate and construct lattice-based encryption algorithms (such as NTRUEncrypt and LWE-based systems) that are resistant to quantum computer attacks.
- Code-Based Cryptography: Investigate techniques that are believed to be quantum-resistant and rely on error-correcting codes, such as the McEliece cryptosystem.

6.3.4 Improved Recovery Mechanisms:

Decentralized Recovery Systems: This could be useful in a single password based decentralized recovery. Such information as user's data can be accumulated in e-ledgers e.g., blockchain distributed ledger.

- Blockchain-based Recovery: Moreover, the properties of the distributed blockchain's ledger where the encrypted recovery key of users and the data bits can be beneficial in preventing tampering.
- Decentralised Storage: Encrypted pieces of data are spread via many nodes that form a blockchain, thus ensuring their own robustness.

Security Codes: Provide security codes for forgotten master passwords. This is a security code which is a one time use type of code to reset your password. You may use it in case of password forgetting and leakage situations

- One-Time Use: Develop single use security codes which may be kept in a safe place by the users for quick recovery of passwords if such is required under an extreme condition.
- Encrypted Storage: Use strong encryption for these security codes so as to deter an unauthorized entry.

6.3.5 Include More Features

Checking for breach of email:

Vault Shield hopes to provide a proactive security solution in the future by introducing a capability to test user password integrity against known dark web breaches. Secure APIs or databases that cross-reference user passwords against leaked datasets will be employed to create this functionality and warn users if their credentials have been hacked in any known data breaches. This proactive strategy will boost internet security and decrease the dangers connected with possible data breaches by allowing users to immediately update compromised passwords.

Showing an overlay when the user is trying to enter a password to a website or an app:

In future editions, Vault Shield intends to provide an overlay capability to provide an extra degree of security when users enter passwords into websites or applications. When this overlay detects that a user chooses to safely save or manage their passwords within Vault Shield, it will activate and prompt the user. To encourage safe password management practises, the overlay will dynamically appear during password entry and allow users to save their login credentials directly into the password manager. This proactive overlay encourages users to use the password manager regularly for enhanced convenience and security on a range of websites and applications by making it easier for them to remember and store their login credentials.

REFERENCES

- [1] Oladipupo, Ridwan Olayinka, and Ajayi Olusola Olajide. "An enhanced web security for cloud-based password management." (2019).
- [2] Lyastani, Sanam Ghorbani, et al. "Studying the impact of managers on password strength and reuse." arXiv preprint arXiv:1712.08940 (2017).
- [3] Sriramya, P., and R. A. Karthika. "Providing password security by salted password hashing using bcrypt algorithm." *ARNP journal of engineering and applied sciences* 10.13 (2015): 5551-5556.
- [4] Chandra, Sourabh, et al. "A comparative survey of symmetric and asymmetric key cryptography." 2014 international conference on electronics, communication and computational engineering (ICECCE). IEEE, 2014.
- [5] Zhao, Rui, and Chuan Yue. "Toward a secure and usable cloud-based password manager for web browsers." *Computers & Security* 46 (2014): 32-47.
- [6] Li, Zhiwei, et al. "The {Emperor's} new password manager: Security analysis of web-based password managers." 23rd USENIX Security Symposium (USENIX Security 14). 2014.
- [7] Selent, Douglas. "Advanced encryption standard." *Rivier Academic Journal* 6.2 (2010): 1-14.
- [8] Josefsson, Simon. PKCS# 5: Password-based key derivation function 2 (PBKDF2) test vectors. No. rfc 6070. 2011.
- [9] Luevanos, C., Elizarraras, J., Yeh, J.-H., & Hirschi, K. (Y). "Analysis on the Security and Use of Password Managers".2017.
- [10] Fagan, M., Albayram, Y., Khan, M. M. H., & Buck, R. "An investigation into users' considerations towards using password managers.".2017
- [11] Shand, M. and Vuillemin, J., 1993, June. Fast implementations of RSA cryptography. In *Proceedings of IEEE 11th Symposium on Computer Arithmetic* (pp. 252-259). IEEE.
- [12] Barten, D., 2019. Client-side Attacks on the LastPass Browser Extension.
- [13] <https://tacomator.medium.com/use-a-password-manager-do-it-now-2e37cecf8de4>
- [14] <https://jumpcloud.com/blog/password-manager>
- [15] <https://www.filecloud.com/blog/2022/01/should-you-use-a-password-manager/>

[16] https://is.muni.cz/th/sm620/pecuch_bc.pdf

[17] <https://www.sciencedirect.com/science/article/pii/S1574013718302533>

[18] <https://blog.1password.com/how-password-managers-work>

[19] <https://logmeonce.com/resources/how-to-create-your-own-password-manager>

[20] <https://qtanalytics.in/publications/index.php/JoITC/article/view/65>

[21] <https://in.nau.edu/wp-content/uploads/sites/223/2019/11/Password-Manager-Hashing.pdf>

Appendix

Vault Shield

ORIGINALITY REPORT

4%

SIMILARITY INDEX

3%

INTERNET SOURCES

2%

PUBLICATIONS

2%

STUDENT PAPERS

PRIMARY SOURCES

- 1** Sanjay Singla, Aditya Soni, Er. Swati Kashyap, Sandep Singh Kang, Gagandeep Singh, Rahul Bhandari. "From Strings to Secrets: Modernizing Password Management for Enhanced Security", 2023 International Conference on Advanced Computing & Communication Technologies (ICACCTech), 2023
Publication <1 %
- 2** www.cs.uccs.edu
Internet Source <1 %
- 3** Submitted to Accra Institute of Technology
Student Paper <1 %
- 4** www.researchgate.net
Internet Source <1 %
- 5** Submitted to University Of Tasmania
Student Paper <1 %
- 6** inside.mines.edu
Internet Source <1 %

www.ijcaonline.org

7	Internet Source	<1 %
8	Submitted to UT, Dallas Student Paper	<1 %
9	tudr.thapar.edu:8080 Internet Source	<1 %
10	clickup.com Internet Source	<1 %
11	www.arxiv-vanity.com Internet Source	<1 %
12	scholarworks.boisestate.edu Internet Source	<1 %
13	deepai.org Internet Source	<1 %
14	www.slideshare.net Internet Source	<1 %
15	kipdf.com Internet Source	<1 %
16	www.coursehero.com Internet Source	<1 %
17	www.dtic.mil Internet Source	<1 %
18	www.semanticscholar.org Internet Source	<1 %

19

Submitted to PSB Academy (ACP eSolutions)

Student Paper

<1 %

20

uobrep.openrepository.com

Internet Source

<1 %

Exclude quotes On

Exclude matches Off

Exclude bibliography On

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date:

Type of Document (Tick): PhD Thesis M.Tech Dissertation/ Report B.Tech Project Report Paper

Name: _____ Department: _____ Enrolment No _____

Contact No. _____ E-mail. _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	<ul style="list-style-type: none">• All Preliminary Pages• Bibliography/Images/Quotes• 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature

Librarian

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com