International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 6 Issue 3, March-April 2022 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Enhancing Password Manager Chrome Extension through Multi-Authentication and Device Logs

Karthika Venugopal K, Prof. Priya. N

School of CS & IT, Jain University, Bangalore, Karnataka, India

ABSTRACT

During the early years of the Internet, people used to have a few passwords for a few important web programmes that they used to shop, study, stay connected, and get work done. Things have gotten a lot more complicated. Last-research pass's from 2017 found that users had to remember an average of 191 different passwords—just for work—not to mention their own passwords. Users can save their credentials in an organized manner using the proposed system, such as Finance, Shopping, OTT, social networking, and Work. Users can also create new categories for themselves. Furthermore, upon logging in, the user must go through 2FA verification, which sends a unique six-digit numeric number to the user's authenticator App(s).A Time-Based One-Time Password (TOTP, or OTP) is a sequence of dynamic numbers that changes with the passage of time. Frequently, these take the form of sic-digit numbers that change every 30 seconds. TOTPs are created using a secret seed password that is given to users in the form of a QR code or in plaintext when they register. TOTPs (and their seeds) are stored on either hardware security tokens or soft tokens, which are apps that show the numbers on mobile devices. TOTP deciphers a code from the secret using Greenwich Mean Time (GMT).

KEYWORDS: *Password manager, Authentication, 2FA, Chrome Extension, unauthorized access, Mongo DB, Node.js, Google Authenticator, TOTP*

How to cite this paper: Karthika Venugopal K | Prof. Priya. N "Enhancing Password Manager Chrome Extension through Multi-Authentication and Device Logs" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-6 | Issue-3, April 2022, pp.1564-1570, URL: www.ijtsrd.com/papers/ijtsrd49775.pdf

IJTSRD49775

I. INTRODUCTION

This proposal for a password manager product will be an open-source, cross-platform application that uses PGP encryption standards & 2FA, another outstanding feature of this product is its ability to give consumers with cross-device authentication. Initially, the aim was to implement this project idea as a Google Chrome extension, and then, based on user input, improve the idea as a Desktop Application / Mobile Application in the future. The main goal of this project is to provide users with a secure password manager in the form of a Google extension that can be used from anywhere and in any Google web store compatible browser, as well as a list of device logs that show the number of devices on which the account is active. The Significance of the proposed system is to provide users a secure password manager by overcoming below cons of the existing Chrome extension (Existing System).

- Cross-Device authentication.
- Digital Signature for Recovery and Backup of account.
- PGP Encryption and Decryption standard to store & retrieve users data.
- Multi-factor authentication is implemented to improve the initial security of the application.
- Device logs are made available to the user so that the user can keep track of which device is active & which is not.

Following are the brief explanation about the technology used to build the product: -

A. Angular7

Angular 7 is an open source JavaScript framework for developing web applications and apps in JavaScript, HTML, and Typescript, a JavaScript superset. Angular comes with built-in animation, http service, and materials, which come with features like auto-complete, navigation, toolbar, menus, and more. The

code is written in Typescript, which is then translated into JavaScript and shown in the browser.

B. Node.js

Node.js is a cross-platform runtime environment and framework for executing JavaScript outside of the browser. It's used to make server-side and network web applications. It's free to use and open source.

Many of Node.js' core modules are developed in JavaScript. The most common usage of Node.js is to execute real-time server applications. The following is the official documentation's definition:

Node.js is a framework for easily creating fast and scalable network applications based on Chrome's JavaScript runtime. Node.js is lightweight and efficient because to its event-driven, non-blocking I/O approach, which is ideal for data-intensive real-time applications that operate across multiple devices

C. MongoDB

The technique for storing and retrieving data is provided by a database management system.[2] Different types of database management systems exist:

- RDBMS (Relational Database Management System) (Relational Database Management Systems)
- OLAP (Online Analytical Processing) (Online Analytical Processing)
- NoSQL database (Not only SQL)

Relational databases, such as MySQL, are not the same as NoSQL databases. Before you can actually put data into a relational database, you must first construct the table, specify the structure, and set the data types of fields, among other things. You don't have to worry about it with NoSQL because you can insert and update data on the fly.[2]One of the benefits of NoSQL databases is that they are extremely easy to scale and perform substantially faster in the majority of database operations. There are times when a relational database is preferable to a NoSQL database, however when dealing with large amounts of data, a NoSQL database is the best option.[2]MongoDB is a document-oriented database that is open source and holds data in the form of documents (key and value pairs). Document-based databases are one sort of NoSQL database, as we mentioned in our previous course (NoSQL introduction).[2]

D. Ajax

Ajax is an acronym that stands for Asynchronous Javascript And Xml. Ajax is just a method of retrieving data from a server and modifying sections of a web page without having to reload the entire page. Essentially, Ajax uses the browser's built-in XML Http Request (XHR) object to send and receive data to and from a web server in the background, asynchronously, without blocking the page or interfering with the user's experience. Ajax has grown in popularity to the point where it's difficult to find an application that doesn't use it in some way. Gmail, Google Maps, Google Docs, YouTube, Facebook, Flickr, and a slew of other large-scale Ajax-driven internet apps are examples.

E. Google Authenticator

Google Authenticator is a software-based authenticator developed by Google for authenticating users of software applications using the Time-based One-time Password Algorithm (TOTP; described in RFC 6238) and HMAC-based One-time Password Algorithm (HOTP; specified in RFC 4226).

Authenticator must must be installed on a smartphone before it can be used. It must be configured for each site with which it will be used: the site will provide the user a shared secret key through a secure channel, which the user will store in the Authenticator app. This secret key will be needed to log in to the site in the future. The user enters a username and password to log into a site or service that supports two-factor authentication and Authenticator. The site then calculates (but does not disclose) the six-digit one-time password necessary and prompts the user to enter it. The user launches the Authenticator app, which generates and displays the same password, which the user enters to verify their identity.

II. LITERATURE REVIEW

1. A different technique of authentication is to use devices that are always nearby and personal like mobile phones or a credential server. Password-based authentication protocols are the result of a lot of research and testing. Wu suggests using a strong password, more contemporary protocols such as the PAKE protocols, can function with weak passwords to prevent dictionary attacks on the information transferred. When compromising a service provider, different variants of PAKE protocols prohibit an attacker from impersonating a user. Compact PAKE is a protocol that is similar to EKE2 however, it is more compact and incorporates user key parameter retrieval as well as asymmetric authentication.
2. The PAKE authentication mechanism is used by Better Auth, although it doesn't support password stretching. Van Laer proposes using KDF for password stretching and storing the needed salt parameters at the service provider, which is similar to our method. In comparison to our work, they only use pre-defined KDF factors, such as CPU cost, and only have configurable salt,

making their account aging solution inflexible. They also employ the Schnorrprotocol, which is susceptible to parameter attacks, such as when the provider provides a tampered salt value.

3. An intriguing Key Derivation Function strategy is that it is not susceptible to parameter attacks. When assessing a Halting Key Derivation Function, the user runs a KDF algorithm until it reaches a halting parameter. A password that isn't legitimate can't be ruled out completely. When attempting a dictionary attack, this results in an attacker having to do more than three times the work. The HPAKE authentication protocol employs an HKDF and saves the necessary halting parameter at the predefined provider. External attackers can't see the halting parameters because of a covert credential retrieval technique, which makes offline dictionary attacks unfeasible. A parameter attack would be impossible with HPAKE since the user's HKDF calculation would not finish. A usability issue with HKDFs is that a user who inadvertently enters an erroneous password does not receive a timely notification about the error. Any typical state-of-the-art KDF can be used with our method.
4. Another set of authentication techniques makes use of external devices such as mobile phones or a credential server. No additional device or server is required in our job for the service provider's authentication. Management of passwords. A method that does not require the use of a server. A user's password being reused on several websites is to deduce a hash function that takes the value of the authentication key as inputs, passes the domain name and the password. Since this method is sensitive to dictionary assaults. To strengthen the hash function, it was recommended to use it n times. is the password n, on the other hand, is a hard-coded value. This indicates that the method will not be adaptable to future hardware.
5. Pvault provides two prominent services like cloud storage and password management, with the drawback of using the same password for both data encryption and storage server authentication. Zhao uses KDF to protect a password that is stored on a secure, reliable cloud storage system, but a simple authentication technique is used. Passport uses a similar basic key stretching mechanism as Halderman, except KDF parameters are stored on a Passport server. Passport, on the other hand, uses the Secure Remote Password protocol for authentication, which is vulnerable to parameter attacks.
6. The contributing factors of research carried out in the field of password manager has had a direct impact on cryptography and penetration testing. Hence, the majority of this study focuses on vast collections of academic studies on the security measures taken by most featured password managers and penetration testing experiments performed by security auditing teams. Furthermore, any checks performed on password managers have been extended to the full extent possible.
7. Moving on to the security analysis trends, we learned about featured password manager services like Last Pass and 1Password. Auto-fill is a function present in several popular password managers, and other browser-based password managers like Chrome web browser and Safari web browser. It has been discovered that major flaws exist that takes advantage of the auto-fill feature, such as I Frame sweep attacks, password sync exploits, and injections.

III. PROBLEM FORMULATION

- The available password managers are either Desktop Applications or Web Apps. So, every single time a user needs to launch the application & update the password manager database if there's any change made to the credentials.
- All the available password manager extensions use AES/DES encryption/decryption standards so passing through those encryption keys is not a tedious task for hackers.

IV. OVERVIEW OF THE APPLICATION

A. SYSTEM ARCHITECTURE

The proposed system will be an open-source and cross-platform application that involves PGP Encryption standards. Users are required to submit their digital signature related files to access their profile every time they need to manage their passwords. Another impeccable feature of this product is to provide users easy accessibility through cross-device authentication.

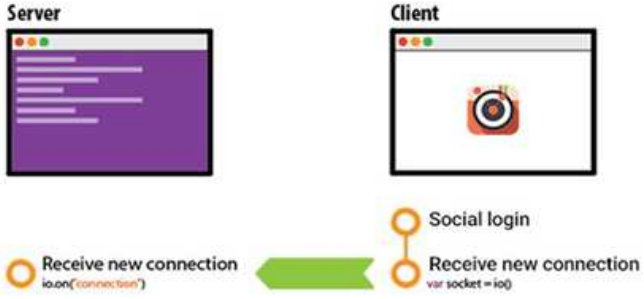
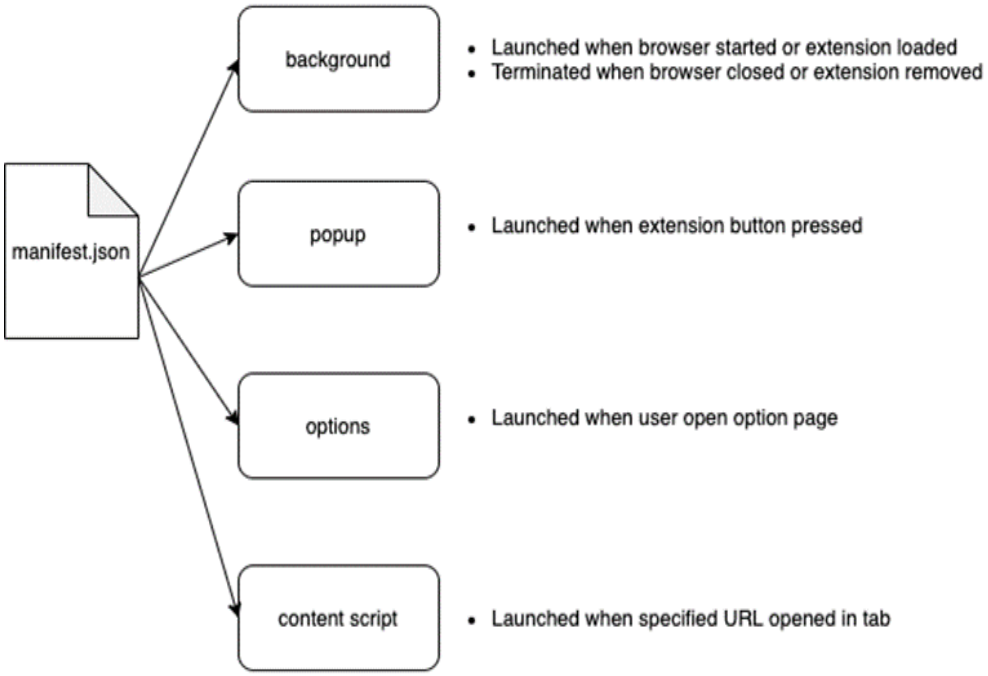


Fig1: System architecture

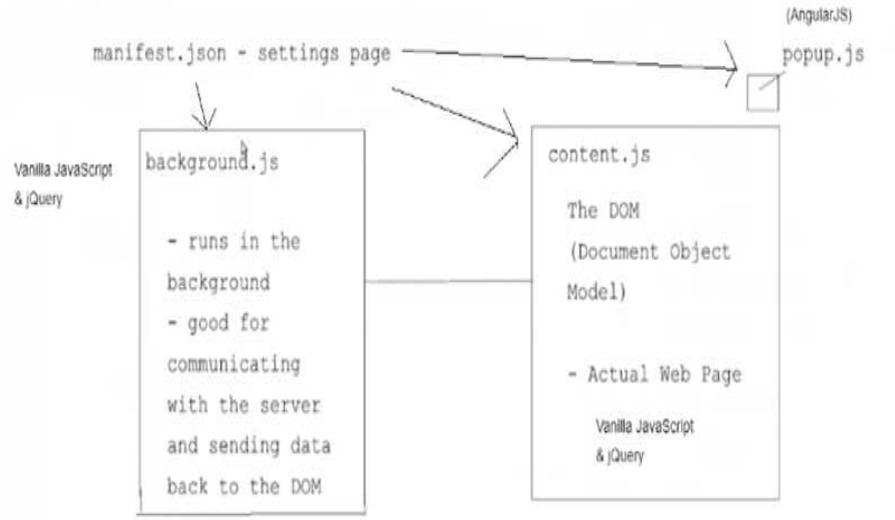
The above diagram shows the relationship between server & client and how the function of a client is dependent on the server for the functioning of the product.

B. Structure of the extension



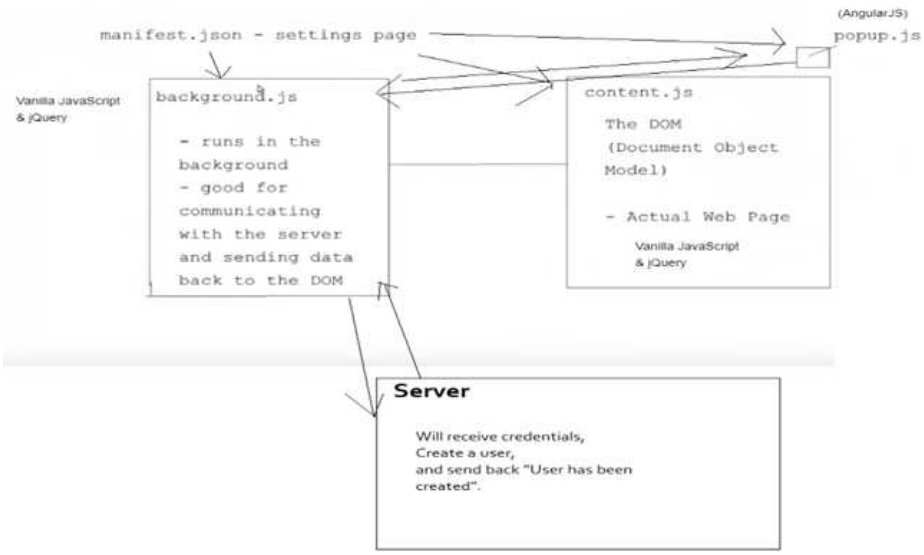
In the above diagram the structure of the extension is read by the manifest. json file which will have the version number, name & description about the extension.

C. Chrome extension workflow structure

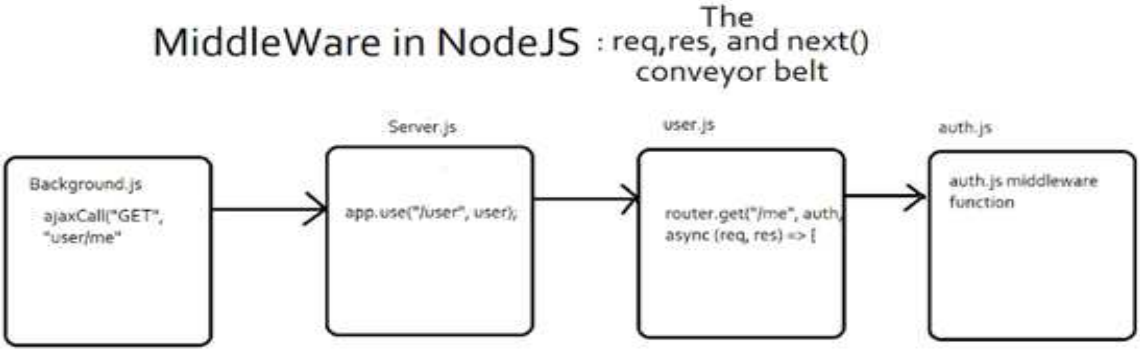


The above diagram shows the workflow of the proposed system from its initial state to end state.

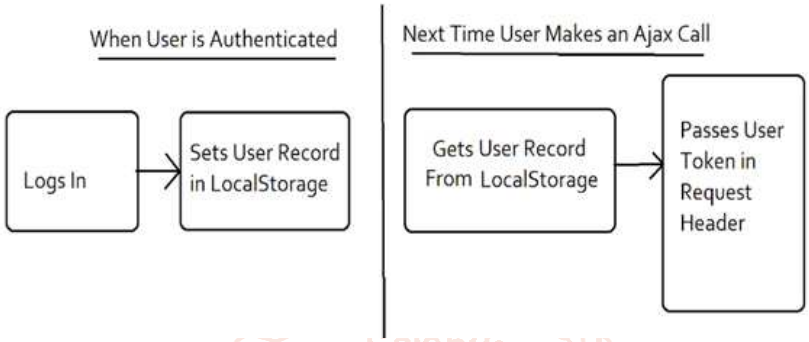
D. Authentication flow structure



E. Node.js middleware flow diagram



F. User token generation process

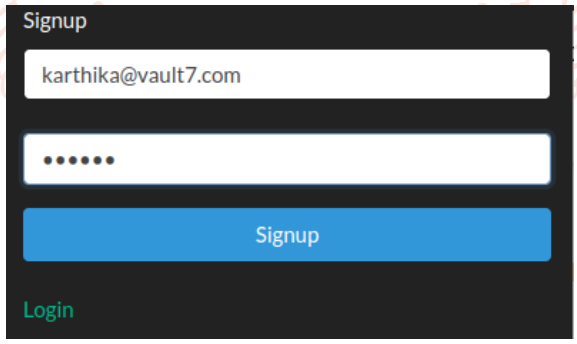


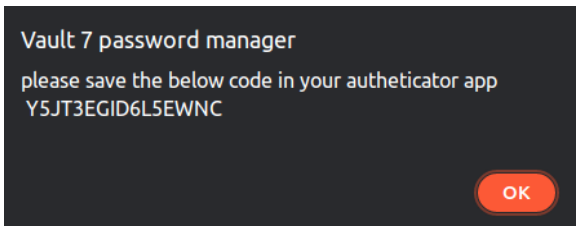
V. Modules

The proposed system involves flowing modules: -

- **Signup module**

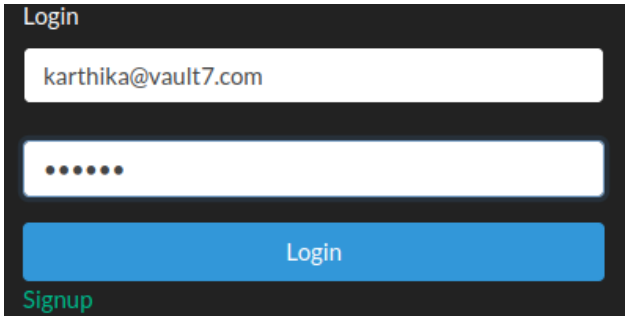
In this module user provides username/email to register/signup to the application. Once user is successfully registered, he will be getting a 16-digit secret key to generate TOTP.

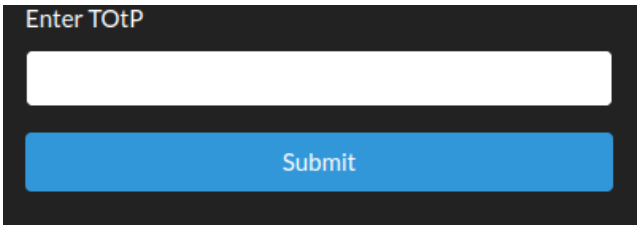




- **Login module**

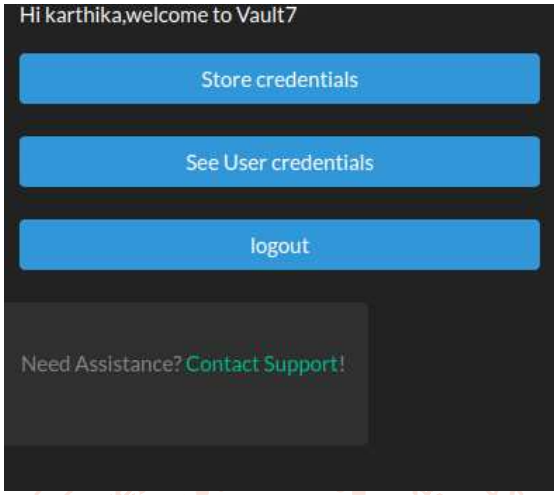
In this module user logs in with his registered username/email to the application. Once login is successful user will be taken to TOTP screen.

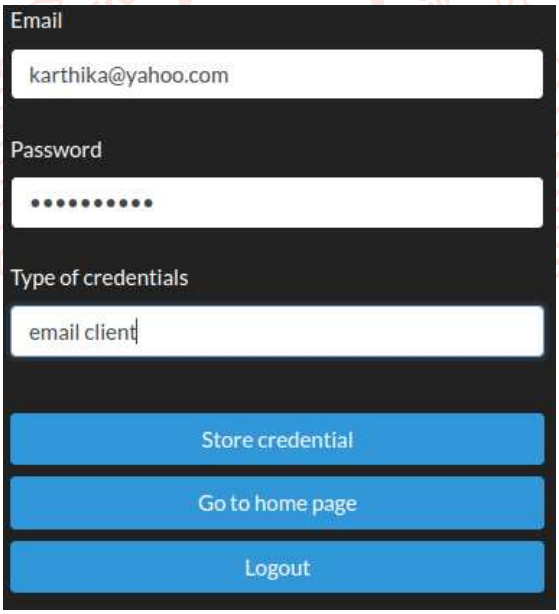


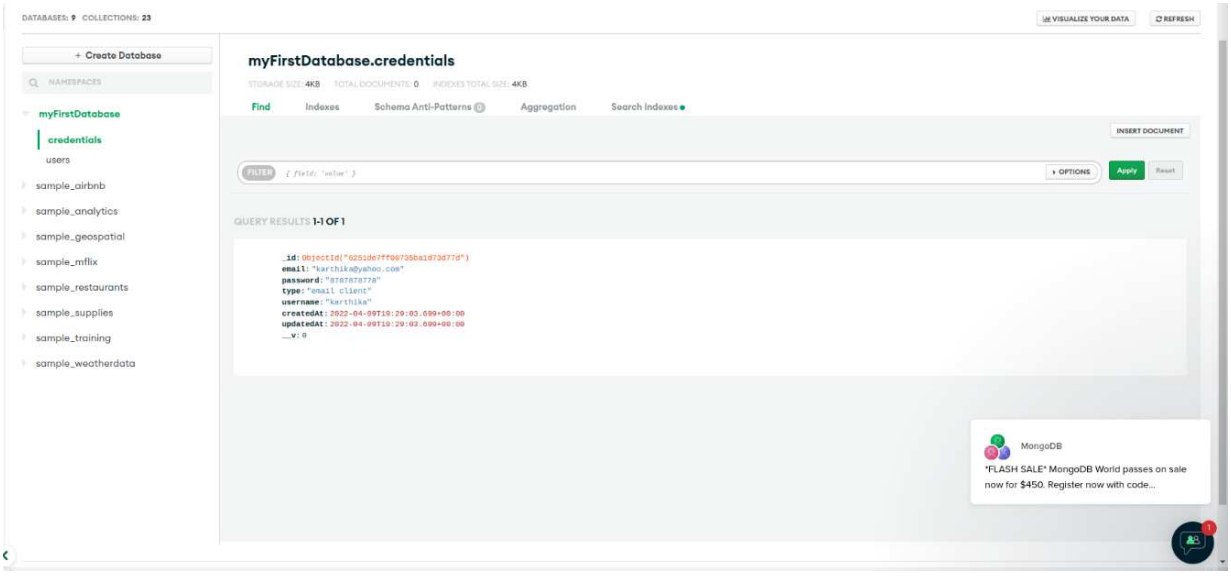


➢ **Store credentials module**

In this module user will store his credentials & stored credentials will be sent to database & stored there for future reference.







VI. CONCLUSION& FUTURE SCOPE

Proposed System is a better password manager as a Google Chrome plugin with 2FA for this research. We demonstrated that when employing PGP encryption technology for secure password-based encryption, a strong authentication approach is required to keep the user's password hidden. Users would simply need their login credentials to log in, and the proposed system would decrypt their data, allowing them to maintain track of their passwords.

A. Future Scope

The future enhancements of the proposed system are:-

- Converting the product into mobile or iOS Applications.
- Using updated encryption & decryption standards
- Providing it as a Desktop application using Electron.js framework
- Enabling the feature of signing up through social networks like Facebook, Google, and LinkedIn.

REFERENCES

[1] M. Bishop and D.V. Klein. Improving system security via proactive password checking. Computers & Security, 14(3):233–249,

[2] J. Bonneau. The Science of Guessing: Analyzing an Anonymized Corpus of 70 Million Passwords. Security and Privacy (SP).IEEE Symposium on, pages 538–552, 2012.

[3] Malone and K. Maher. Investigating the distribution of password choices. In Proceedings of the 21st international conference on World Wide Web, WWW ‟12, pages 301–310, New York, NY, USA, 2012. ACM

[4] https://www.thirdrocktechkno.com/blog/node-js-socket-io-chrome-extension-integration/

[5] https://ieeexplore.ieee.org/document/8456009

[6] https://ieeexplore.ieee.org/document/8326801

[7] https://www.section.io/engineering-education/speakeasy-two-factor-authentication-innodejs/#:~:text=May%205%2C%202021&text=Two%2Dfactor%20authentication%20is%20a,username%2Email%20and%20password%20authentication.

[8] https://www.npmjs.com/

[9] https://www.npmjs.com/package/otplib

[10] https://otplib.yeojz.dev/