APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 7 No. 1 (2026) p. 1270-1289
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

PassGuard: Password Manager with Unauthorized Access Detection

Ahmad Amirul Aiman Mohamed Shaharudeen[1], Nordiana Rahim[1*]

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: nordiana@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2026.07.01.068

Article Info

Received: 11 June 2026
Accepted: 23 June 2026
Available online: 30 June 2026

Keywords

Password Management,
Unauthorized Access Detection,
Multi-Factor Authentication,
TrustGuard, waterfall Methodology

Abstract

PassGuard is designed to address concerns about the security and management of sensitive information. It offers a secure, user-friendly password manager to store and organize credentials, secure notes, payment details, and more. With features like a built-in password generator and TrustGuard for improving password strength, it provides comprehensive protection. The application tackles the risks of password theft and unauthorized access by implementing OTP-based validation and intrusion detection, including real-time alerts with OpenCV. Built using Python and PyQt5, PassGuard is intended for Windows users. Eight core modules were successfully developed: secure registration, email verification, credential storage, password analysis, intrusion detection, OTP authentication, TrustGuard analysis, and breach monitoring. Functionality testing achieved 100% success across 13 test categories. UAT with 21 respondents achieved complete acceptance across account security, recovery features, security features, and additional functionalities. Its core functionalities have undergone structured user acceptance testing (UAT) to validate usability, reliability, and effectiveness across real-world scenarios. This ensures secure storage, streamlined access, and proactive threat detection to enhance digital safety.

1. Introduction

In today's digital era, securing sensitive information is critical, as outdated methods like reusing passwords and manual storage expose users to data breaches and financial loss. Surveys reveal that 98% of users reuse passwords across platforms, heightening risks of credential-based attacks [1]. Additionally, 65% of breaches stem from human error [2], emphasizing the need for robust, automated security solutions.

PassGuard is proposed as a secure password manager for Windows OS that integrates unauthorized access detection. It provides users with a centralized, intuitive platform to store credentials, notes, and payment details securely. Unlike many password managers, PassGuard enhances security through features like password strength analysis, multi-factor authentication (MFA), real-time intrusion alerts, and hashed master passwords. The objectives of this project are as follows:

- To design a password manager application with unauthorized access detection.
- To develop the application specifically for a Windows-based environment.
- To evaluate the application's functionality and effectiveness in securely managing sensitive information.

PassGuard combines proactive security features such as detecting unauthorized login attempts, sending email alerts, and locking accounts after multiple failed logins with tools like password strength evaluation and a

strong password generator. Its OTP-based authentication further enhances protection against unauthorized access. The project integrates eight key modules, including secure registration, email verification, credential storage, password analysis, intrusion detection, and OTP authentication, to provide a seamless and secure user experience. By addressing gaps in existing tools, PassGuard offers a reliable, user-friendly solution for managing sensitive data, empowering users to protect their information against evolving threats while fostering better security practices.

This report is structured into six sections. Section 1 introduces PassGuard's background, objectives, scope, and significance. Section 2 reviews relevant research and challenges in credential management. Section 3 outlines the development methodology, tools, and algorithms. Section 4 details the analysis, design, and interface components. Section 5 covers testing and evaluation to ensure security and usability. Section 6 concludes with key contributions, limitations, and future recommendations.

2. Literature review

This section explores key concepts and technologies related to password management, unauthorized access detection, and security practices. The review covers the evolution of cyber threats, the importance of secure password management, methods for detecting unauthorized access, and various authentication techniques. It also includes an analysis of password strength, password generators, and security technologies that underpin the proposed PassGuard application. Additionally, existing password management systems like Dashlane, LastPass, and Bitwarden are reviewed and compared, culminating in a discussion of the innovations introduced by PassGuard.

2.1 Password manager

A password manager is a digital tool that securely stores and manages passwords for various accounts, often protected by master password or authentication methods like biometrics or MFA. It offers features such as password generation, autofill, and cross-platform synchronization, making it essential for secure digital interactions [3]. Password managers mitigate risks from weak or reused passwords by generating strong, unique passwords for each account, enhancing security and convenience. For individuals and organizations, they improve cybersecurity by reducing human errors, enforcing policies, and protecting against password-based attacks [4]. As cyber threats evolve, password managers are crucial for safeguarding sensitive information.

2.2 Unauthorized access detection

Authorized access refers to entry granted to users with valid credentials, allowing them to access specific resources. It becomes unauthorized when someone without proper permissions attempts to bypass authentication systems, often exploiting vulnerabilities. Unauthorized Access Detection involves methods to identify and alert users of unauthorized entry attempts, essential for maintaining data security it helps detect suspicious login locations and behavior patterns, enabling prompt responses to potential breaches. OpenCV feature was used as a main method to capture the intruder’s image by integrating it with host’s webcam.

2.3 Authentication method for authorized access

Authentication is essential for secure access to sensitive information. PassGuard employs strong authentication methods to prevent unauthorized access. This study implements Multi-Factor Authentication (MFA) using OTP-based email verification with traditional password authentication. There are 2 most used types of authentications. The first one is Single-Factor Authentication (SFA) which is a basic method requiring just one credential, such as a password. While easy to implement, it offers limited security as it depends on a single layer of protection.

The next type of authentication is Multi-Factor Authentication (MFA) which improves security by requiring multiple forms of verification, such as a password and a code sent to a user's phone. This extra layer helps prevent unauthorized access, even if one credential is compromised, and is increasingly used to protect accounts [5].

2.4 Password strength analysis

Password Strength Analysis evaluates a password's complexity and resistance to attacks like brute-force and dictionary attacks [6]. It helps users create secure passwords by encouraging diverse characters, avoiding patterns, and using unique passwords for each account. Strong passwords enhance protection against unauthorized access.

2.5 Password generator

A password generator creates strong, unique passwords to enhance security by producing randomized combinations of letters, numbers, and symbols, making them resistant to brute force and dictionary attacks. Many

users create weak passwords based on personally identifiable information (PII), increasing the risk of unauthorized access [7]. Password generators solve this by offering unpredictable credentials, simplifying the adoption of strong passwords and improving cybersecurity hygiene for both individuals and organizations.

2.6 Technology and techniques used

PassGuard integrates key security features like password generation, MFA, unauthorized access detection, real-time password analysis, and advanced encryption to protect user accounts and data. The password generation uses randomized character generation and entropy-based techniques to create secure, customizable passwords. Users can select from various character types to ensure strong, unique passwords. For data protection, PassGuard implements envelope encryption using PBKDF2-based key derivation with 100,000 iterations and Fernet authenticated encryption (AES-128 + HMAC-SHA256). This zero-knowledge architecture ensures that user data remains encrypted even if the database is compromised, as master passwords are never stored and all sensitive data requires the user's master password for decryption.

For MFA, PassGuard employs OTPs and email verification, providing an extra layer of security. OTPs are generated via pyotp and time-sensitive, sent to users' email for identity verification. Unauthorized access is detected through login attempt tracking and unusual pattern identification, locking the device after repeated failed attempts. OpenCV captures intruder images for added security. Real-time password analysis helps users create stronger passwords by evaluating their complexity and providing instant feedback to reduce risks

2.7 Study of Existing Systems and Approaches

Modern password management systems typically share a core set of features aimed at enhancing digital security. These include secure password storage, multi-factor authentication (MFA), password strength analysis, and customizable password generation. While such features are now standard across popular tools like Dashlane, LastPass, and Bitwarden, more advanced capabilities such as unauthorized access detection and real-time alerts are not commonly implemented.

Existing password management systems like Dashlane, LastPass, and Bitwarden offer features such as secure storage, multi-factor authentication, and password generation. However, they lack comprehensive unauthorized access detection and real-time alerts, limiting their proactive security. While these systems provide solid security foundations, their lack of real-time monitoring highlights the need for more robust solutions like PassGuard, which integrates unauthorized access detection, password strength analysis, and customizable password generation. The comparison in Table 1 shows that while Dashlane, LastPass, and Bitwarden offer basic features, they lack real-time unauthorized access detection and alerts. PassGuard addresses these gaps with advanced security measures, offering a more comprehensive solution for protecting sensitive information.

Table 1 *Comparison of existing system with PassGuard*

Feature	Dashlane	LastPass	Bitwarden	PassGuard
Secure password storage	AES-256 zero-knowledge	AES-256 cloud sync	AES-256 open-source	Envelope encryption
Multi-Factor Authentication	TOTP, SMS, biometric	TOTP, SMS, tokens	TOTP, email, keys	Mandatory OTP
Unauthorized access detection	Basic monitoring	Limited dashboard	Reports only	Real-time webcam
Password strength analysis	Health scoring	Basic indicators	Health reports	Pattern detection
Customizable password generation	Standard options	Multiple templates	Full customization	Pattern avoidance
Unauthorized access alerts	Breach notifications	Security alerts	Email only	Webcam alerts

3. Methodology

This section outlines the Object-Oriented Programming (OOP) methodology used for PassGuard's development. OOP emphasizes modularity, reusability, and scalability, ensuring efficient design and seamless component interaction. By applying principles like encapsulation, inheritance, and polymorphism, the approach facilitates secure, maintainable, and adaptable application development. The section also highlights project activities, milestones, and the iterative prototyping model used for continuous refinement through testing and feedback. The goal is to create a secure, user-friendly password manager with advanced unauthorized access detection.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1270-1289

3.1 Project methodology

The waterfall methodology was chosen for its structured and disciplined approach, making it well-suited for security-focused applications like PassGuard. Its linear sequential life cycle model ensures that each phase is completed thoroughly before moving to the next, reducing the risk of overlooking critical security or functional requirements. This predictability and clarity in workflow are especially beneficial when developing tools that require strict adherence to predefined goals, such as data protection and unauthorized access prevention. It is also simple to understand and use [8], allowing for clear documentation, continuous feedback, and incremental improvements. This section outlines the key stages: planning, designing, implementation, testing, and maintenance that ensure the application meets user needs and security requirements. The waterfall workflow, as shown in Fig. 1, illustrates the sequential development phases where each step must be completed before the next beginning, supporting detailed design, organized planning, and traceable progress across the project lifecycle.

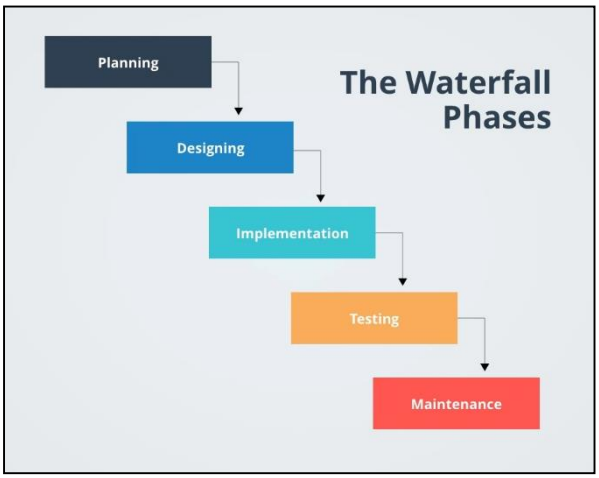


Fig. 1 *Waterfall Methodology*

3.1.1 Planning

During the planning stage, requirements were collected and analyzed by carefully examining cybersecurity risks and current password managers. Significant features have been identified, including multi-factor authentication (MFA), secure password storage, and detection of unauthorized access. The project schedule, development materials, and risk-reduction plans were all developed during this phase. A Software Requirements Specification (SRS) document detailing all functional and non-functional requirements was the outcome of this phase.

3.1.2 Designing

In the design stage, converted the requirements into a template for the application. It contained component interactions, UI/UX wireframes, database schema, and system architecture. Use-case, activity, and sequence diagrams are examples of UML diagrams that were created to show how an application functions. Consistent data flow, secure credential management, and user-friendly interface across TrustGuard, OTP verification, and breach monitoring modules were all guaranteed by the design.

3.1.3 Implementation

Passguard was developed during the implementation phase with MySQL for database storage and Python and PyQt5 for the graphical user interface. According to the design reports, every module including card management, secure notes, login, registration, password creation, and unauthorized access detection was coded. Object-oriented principles for programming were followed, with a focus on modular programming and code reuse.

3.1.4 Testing

The testing phase included User Acceptance Testing (UAT) and Functionality Testing, using detailed test cases to validate each feature against expected outcomes. This stage followed an eight-step process involving requirement review, initial testing, and iterative improvements based on feedback to meet security, usability, and functionality goals [10]. UAT confirmed that all standards were successfully met. As part of this phase, specific attention was given to validating password strength to ensure secure authentication. Fig. 2 illustrates the

password validation process in PassGuard, starting with the user entering a password. The system then checks for minimum length, uppercase and lowercase letters, numbers, special characters, and common password patterns. Any failed criteria generate specific warnings to guide the user toward meeting requirements and improving password strength. If all checks pass, the password is accepted. This validation flow ensures strong password creation, significantly enhancing overall account security and reducing vulnerability to attacks.

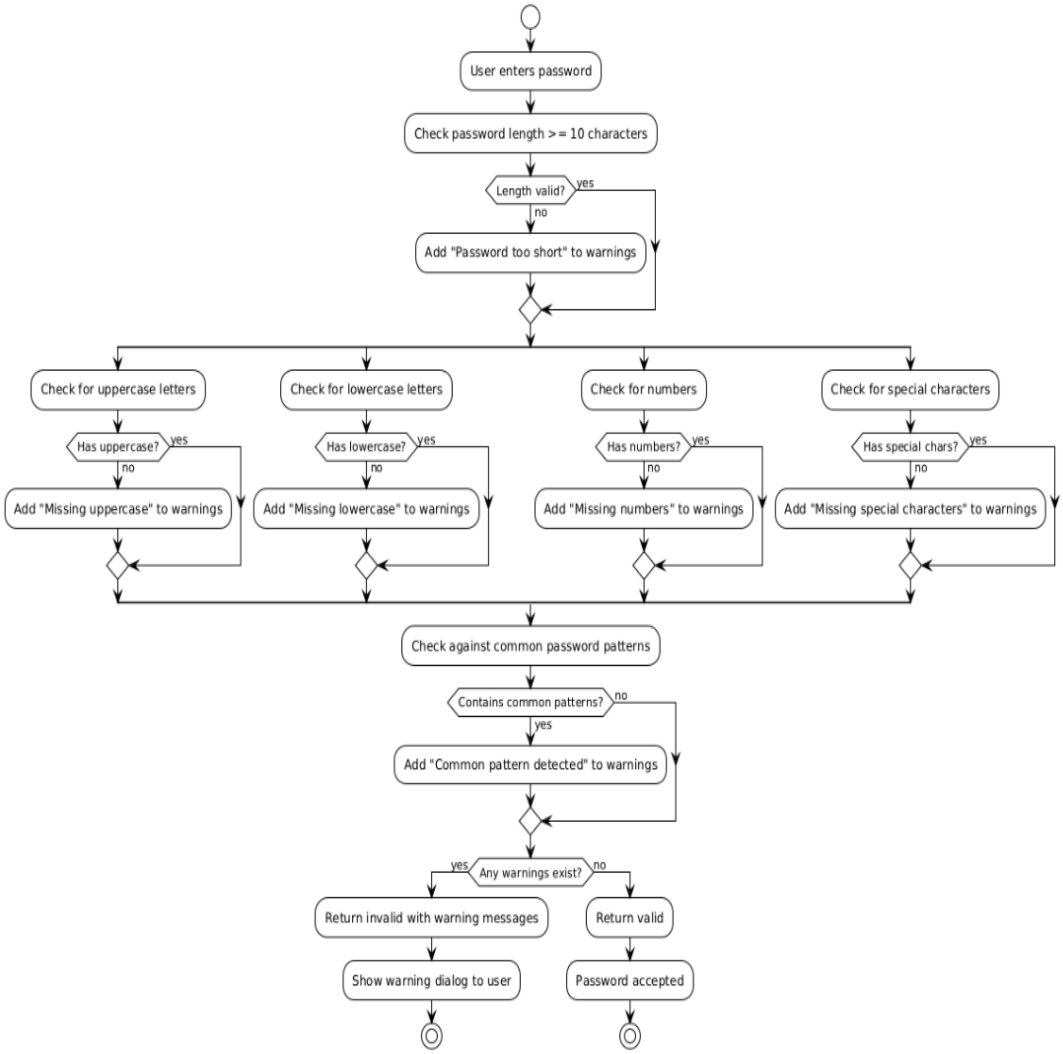


Fig. 2 *Password validation flowchart showing rule-based checks and user feedback for password strength enforcement*

3.1.5 Maintenance

The maintenance phase fixed minor bugs, security patches, and performance optimizations found during testing and feedback, even though the Waterfall model normally limits changes after deployment. The application was made available for Windows platforms, and stability and usability were guaranteed through post-launch monitoring. Improvements based on user input or new threat vectors are part of future maintenance.

3.2 Application development workflow

As shown in Table 2, the development process of PassGuard follows the Waterfall methodology through a series of structured and sequential phases. The Planning phase involves gathering and analyzing requirements to define the application's features and security needs. This is followed by the Designing phase, where the system architecture, UI wireframes, and database schema are developed. During the Implementation phase, the application's core functionalities such as password management, user authentication, and security mechanisms are developed based on the design specifications. The Testing phase ensures that all components function correctly and meet usability and security standards through comprehensive testing. Finally, the Maintenance phase focuses on applying security patches, fixing bugs, and optimizing performance to ensure long-term reliability and protection against emerging threats.

Table 2 *Task and output of each phase*

Phase	Task	Output
Planning	Requirements Gathering & Analysis	Gather and analyze requirements to define application features and functionality.
Designing	application & UI Design	Design the application architecture, including the UI layout, database structure, and backend interactions.
Implementation	Development of Application	Fully developed application modules including authentication, password management, and security mechanisms.
Testing	Functional & User Acceptance Testing	Validated application performance, usability, and security through comprehensive test cases.
Maintenance	Security Patching & Performance Optimization	Enhanced stability, improved performance, and updates based on evolving security needs.

3.3 Project planning

This section outlines the timeline for PassGuard's development using the Waterfall methodology. As shown in Fig. 3, the project runs from September 2024 to June 2025 in five sequential phases: planning (September), design (October to mid-November), implementation (mid-November to early February), testing (February to mid-March), and maintenance (mid-March to June). Each phase is completed before the next, ensuring a structured development process.

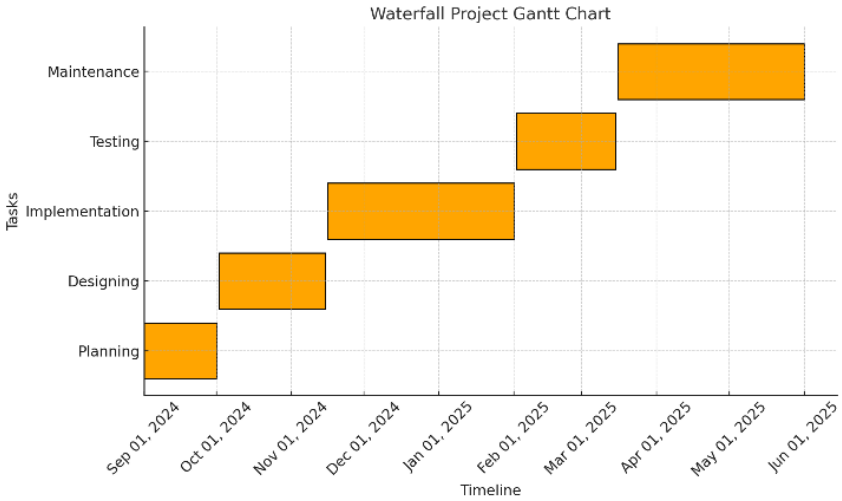


Fig. 3 *Project planning for Passguard*

4. Analysis and design

This section details the analysis and design process for PassGuard, focusing on requirements, architecture, and design methodology. It outlines functionality, performance, and user interface for secure password management. Guided by OOP principles, UML diagrams visualize the application's structure and workflows.

4.1 PassGuard requirements

This section outlines the functional and non-functional requirements for the PassGuard desktop application, defining its core functionalities and quality attributes.

4.1.1 Functional requirements

Table 3 highlights PassGuard's core functionalities, including secure user registration, robust password management, storage for notes and payment information, and a built-in password generator. It features TrustGuard for password analysis, enforced MFA for added security, secure logout, and unauthorized access detection to capture intruders and notify users of suspicious activity.

Table 3 *Functional Requirements*

No	Requirements	Functionalities
1	User Management	Users must be able to register and log in securely.
2	Password Management	The application should allow users to store, edit, and delete passwords.
3	Secure Notes	Users should be able to store, edit, and delete secure notes.
4	Payment Information	Users should be able to store, edit, and delete payment information.
5	Password Generator	A built-in password generator should enable users to create strong, customized passwords.
6	TrustGuard	The application should analyze weak, reused, or old passwords.
7	Multi-Factor Authentication (MFA)	MFA is enforced by default for enhanced security.
8	Logout	Users must be able to log out securely.
9	Unauthorized access detection	The application must capture intruder attempts and notify the user with relevant information.

4.1.2 Non-functional requirements

Table 4 outlines PassGuard's key requirements, including a user-friendly interface, quick performance, robust security, and scalability for growing data. The application must be compatible with Windows, reliable under normal use, easy to maintain, and ensure data integrity across all modules.

Table 4 *Non-Functional Requirements*

No	Requirements	Descriptions
1	Usability	The user interface should be intuitive and easy to navigate.
2	Performance	The application should respond to user actions quickly.
3	Security	Sensitive data must be securely stored, and authentication must prevent unauthorized access.
4	Scalability	The application should efficiently handle increasing amounts of user data.
5	Compatibility	The application must run smoothly on Windows operating systems.
6	Reliability	The application should operate smoothly under normal usage conditions.
7	Maintainability	The application should be easy to maintain and update for future enhancements.
8	Integrity	User data must remain consistent and accurate across all modules of the application.

4.2 PassGuard analysis

This section analyzes the PassGuard application using UML and flow diagrams, including case diagrams for interactions, activity diagrams for login, sequence diagrams for components, and user flowcharts, highlighting its security-focused design and functionality.

4.2.1 Use case diagram

Fig. 4 shows the main features users can access from the PassGuard homepage. Users can manage passwords, secure notes, and payment cards, run security analysis with TrustGuard, monitor data breaches, configure settings, set up account recovery, and logout. These eight core functionalities represent the primary capabilities available through the homepage interface.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1270-1289

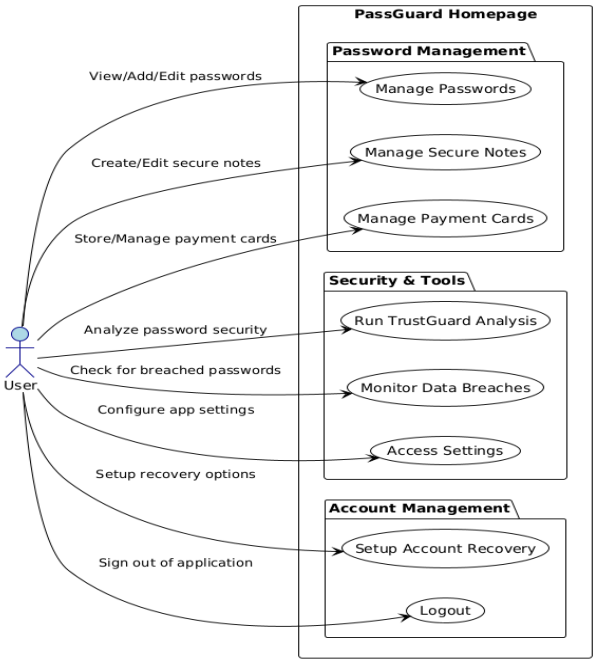


Fig. 4 *Use case diagram of user operations on the PassGuard homepage*

4.2.2 Activity diagram

Fig. 5 shows the activity diagram for the PassGuard login module, illustrating the user authentication flow from credential entry to homepage access. The process includes internet connectivity verification, OTP email sending and verification, and a security mechanism that captures intruder photos and sends alert emails when failed attempts result in account lockout. Upon successful OTP verification, users access the homepage, completing the two-factor authentication process.

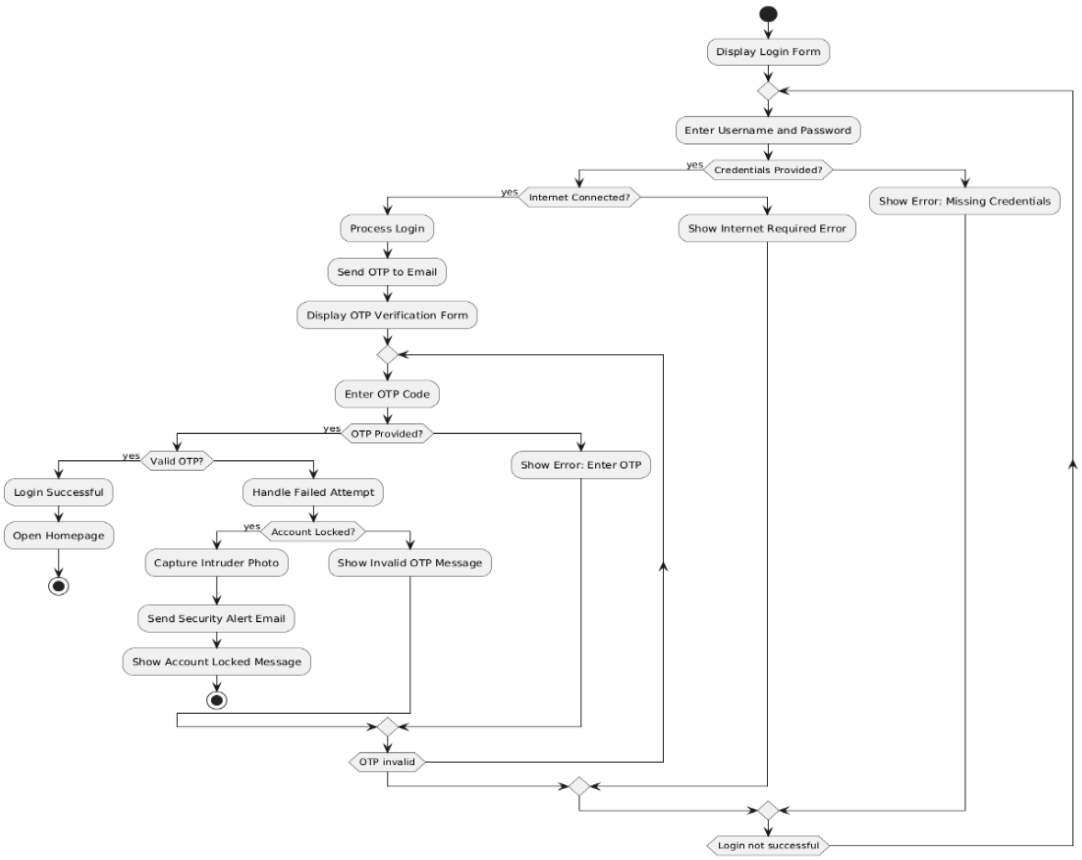


Fig. 5 *Activity Diagram of login process and unauthorized access detection*

4.2.3 Sequence diagram

Fig. 6 illustrates the sequence diagram for the login process in the PassGuard application. It begins when the user enters their username and password. If the input is invalid, an error message is displayed. Upon valid input, the system queries the database to verify the credentials. If the credentials are incorrect, the system checks for failed attempts. After three failed attempts, the webcam captures an image of the potential intruder, and an alert is sent to the user's email. The login process is then locked for three minutes. For valid credentials, the system sends an OTP (One-Time Password) to the user's email. The user enters the OTP, and if it is valid, the homepage is open. If the OTP is invalid, an error message is shown, and similar actions (image capture, email alert, login lock) are triggered after multiple failed OTP attempts.

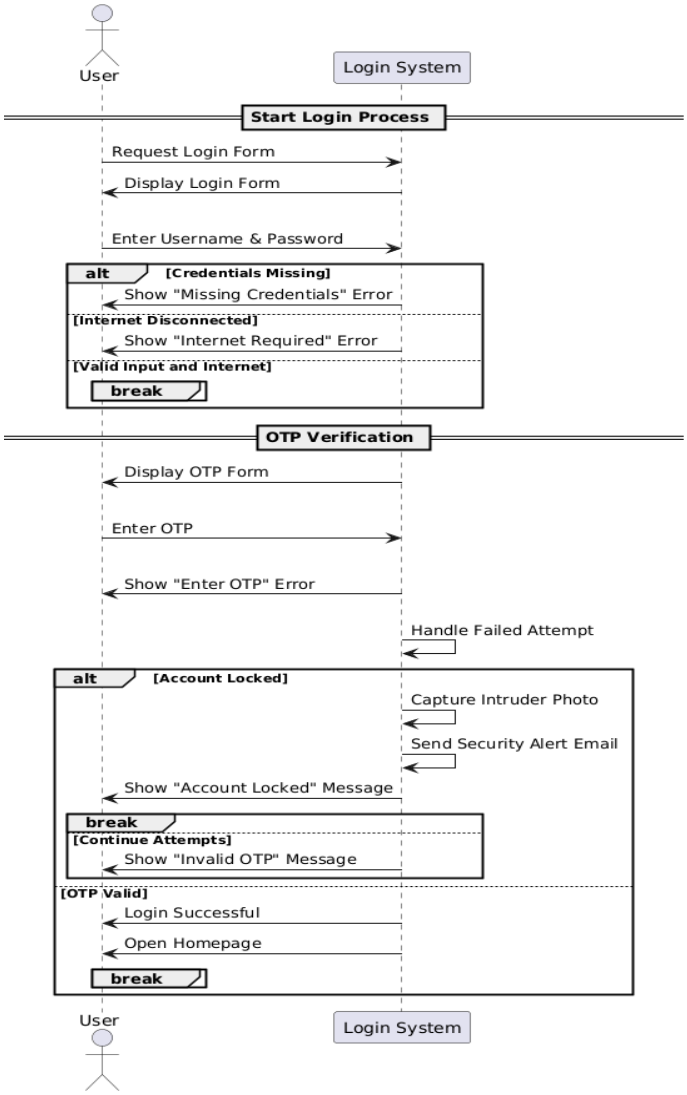


Fig. 6 *Sequence Diagram of login authentication with OTP and security handling*

4.3 PassGuard interface design

Fig. 7 shows the main interfaces of the PassGuard application in designing phase. Fig. 7(a) presents the Login Interface, where users enter their credentials to access the application securely. After three incorrect login attempts, the OpenCV method captures an image of the intruder, and an alert is sent to the account owner's email. Fig. 7(b) displays the Homepage Interface, serving as the central dashboard for easy access to PassGuard's functionalities. Fig. 7(c) illustrates the Passwords Interface, where users securely store, edit, and organize passwords. Finally, Fig. 7(d) presents the TrustGuard Interface, which analyzes password strength and provides improvement suggestions.

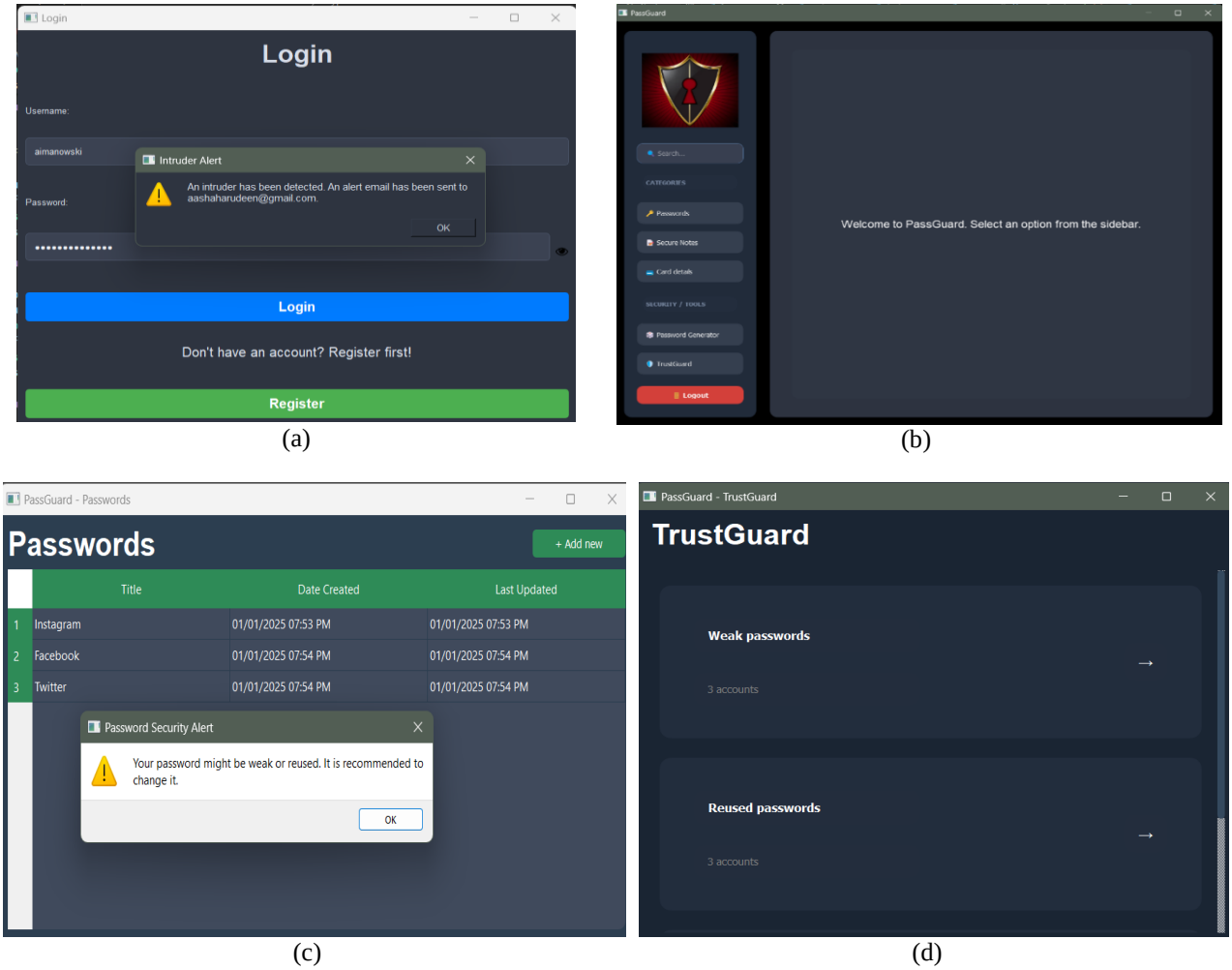


Fig. 7 *PassGuard interfaces (a) Login Interface; (b) Homepage Interface; (c) passwords interface; (d) TrustGuard Interface*

4.4 Database schema

Fig. 8 illustrates the PassGuard database schema, organized into three hierarchical levels to ensure structured and secure data management. At the core, the first level contains the `users` table, which holds essential user information and links to primary data entities such as `passwords`, `payment_cards`, and `secure_notes`, each storing encrypted user-specific content. The second level enhances security through auxiliary tables like `key_seeds` for key management, `otp` for handling one-time passwords, and `intrusion_logs` for logging unauthorized access attempts, including IP addresses and captured photos. The third level consists of independent application-wide configurations and logs, including `global_settings` for app preferences, `verification_codes` for account verification, and `login_attempts` for tracking failed login activities. This layered architecture reinforces data integrity, user association, and robust security controls across the application.

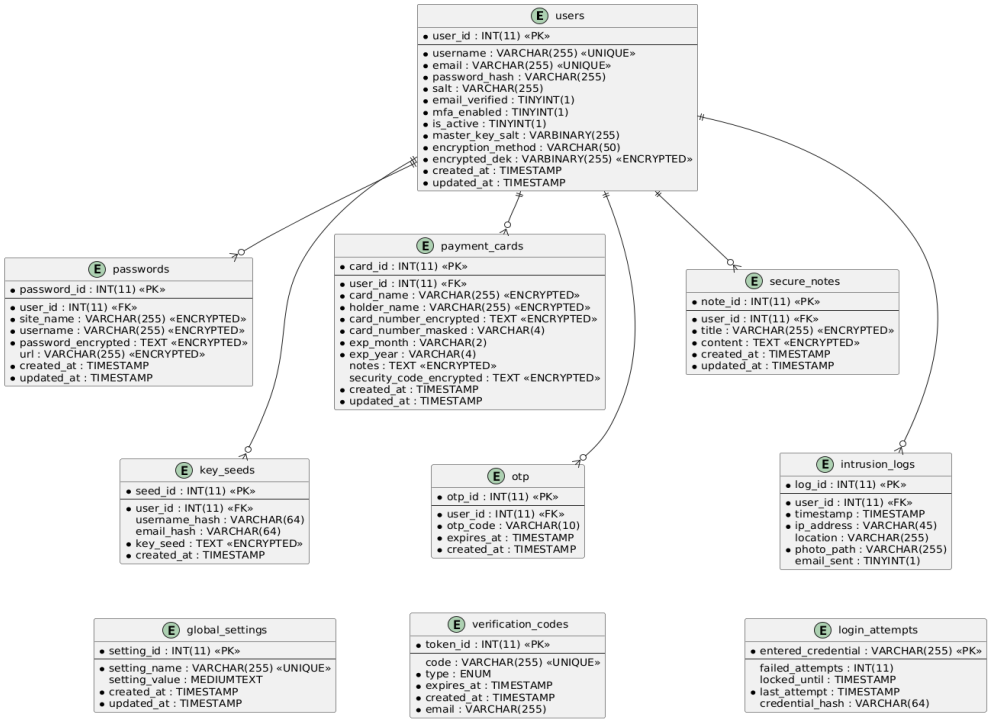


Fig. 8 *PassGuard database schema*

5. Implementation and Testing

This section presents the practical implementation of PassGuard using Python and PyQt5 for the Windows environment. It demonstrates how the design specifications were translated into a functional application, followed by comprehensive functionality and user acceptance testing to validate the application's security and usability.

5.1 PassGuard Implementation

This section demonstrates PassGuard's implemented features through detailed interface walkthroughs and functionality explanations. It showcases all core modules including authentication, password management, security analysis, and unauthorized access detection features.

5.1.1 Register

This interface allows users to create PassGuard account by entering their username, email, password and confirming password. It includes a strong password policy which is a minimum of 12 characters, use of uppercase, lowercase letters, numbers, and special characters. Fig. 9 shows the register interface for PassGuard password manager applications.

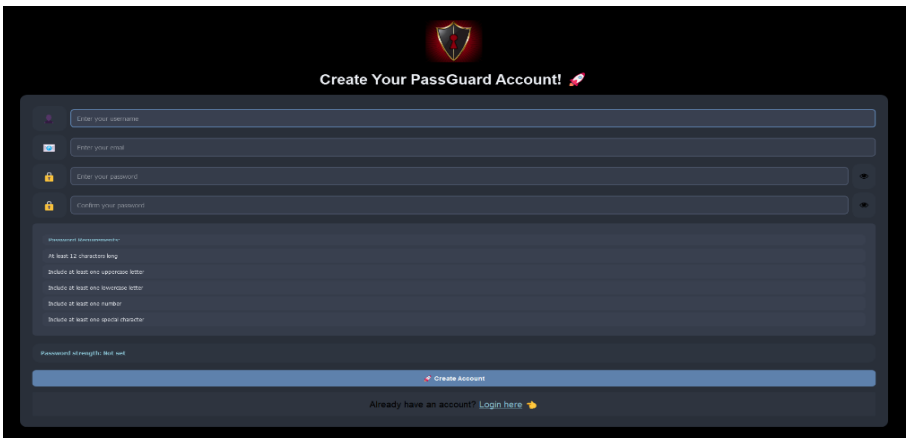


Fig. 9 *Register Interface*

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1270-1289

5.1.2 Login

The login display in Fig. 10 allows users to sign in with their username or email address and a password. It includes password display toggles, a "Can't Access Your Account?" option, and signup links. Fig. 11 shows PassGuard's security alerts, which include an email message of an intruder attempt with an attached image and an app alert letting the user that the account has been temporarily locked owing to several failed login attempts. Fig.12 shows the OTP verification process, which includes a screen for entering the code given via email and an email that comes along with the one-time password required to complete the login verification.

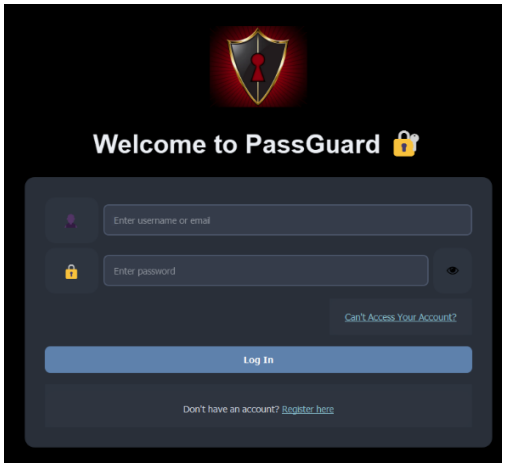
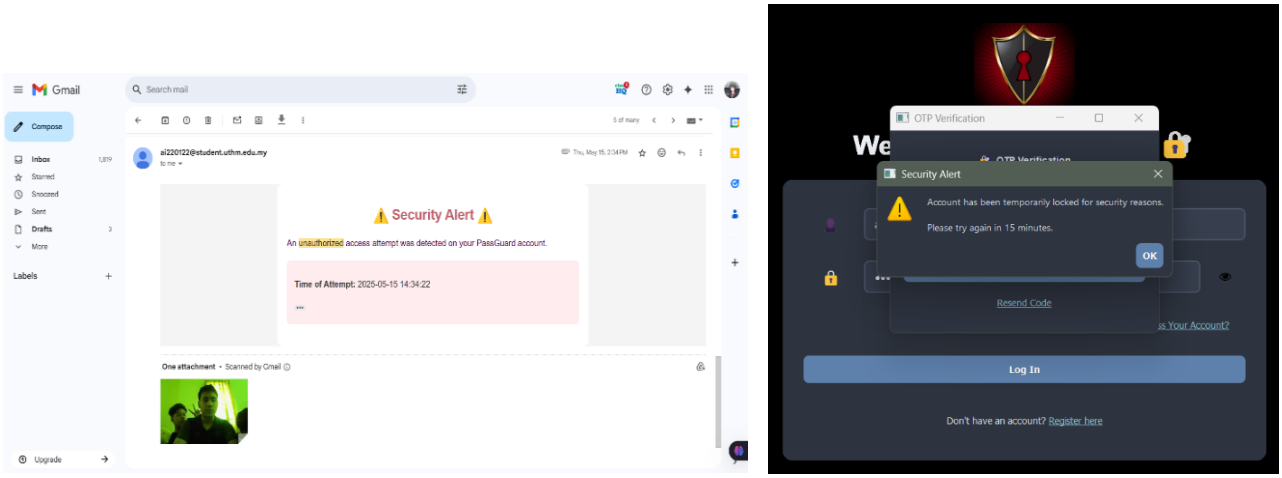
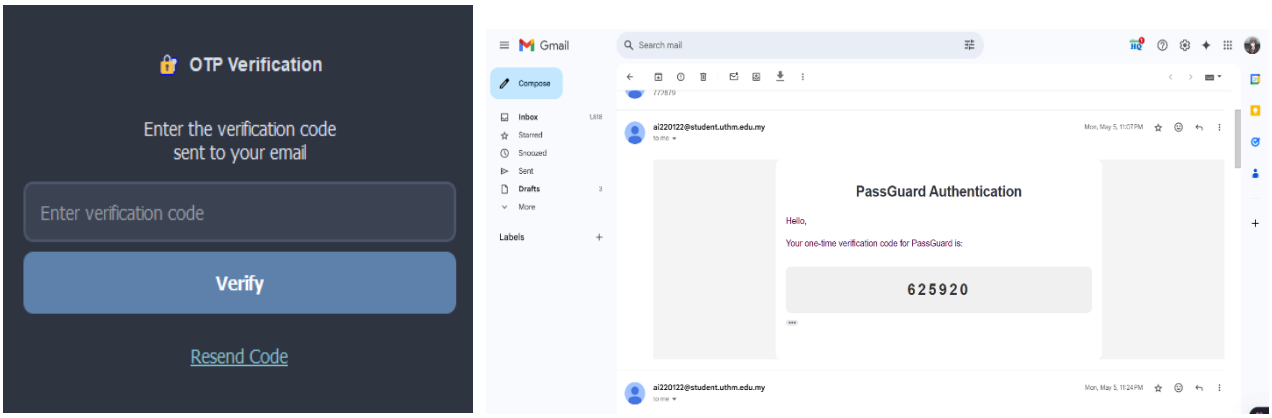


Fig. 10 *Login Interface*



(a) **(b)**

Fig. 11 *PassGuard interfaces (a) Intruder Alert Notifcation: (b) Security Alert Notification*



(a) (b)

Fig. 12 *PassGuard interfaces (a) OTP Verification Interface; (b) Email OTP Notification*

5.1.3 Homepage

The interface in Fig. 13 serves as PassGuard's main dashboard, allowing users to access a variety of services such safe notes, card information, password storage, and security tools. For easy access to all features, it has a clean and organised structure.

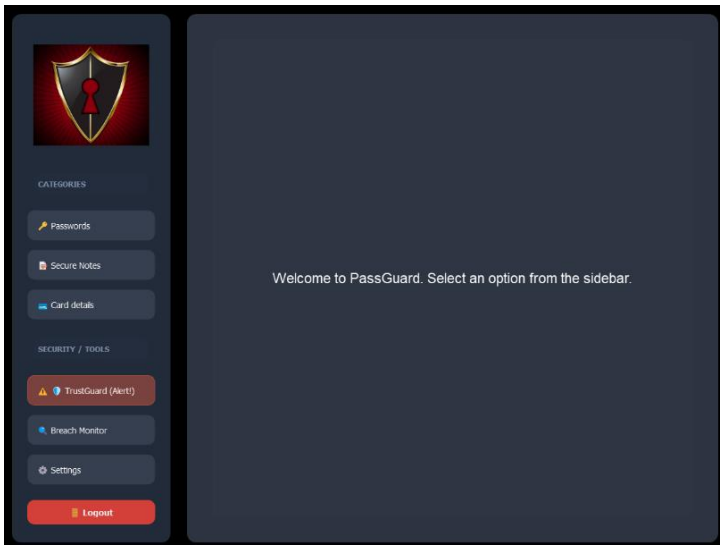


Fig. 13 *Homepage Interfaces*

5.1.4 Account Recovery

Fig.14 shows Users can restore access by entering their registered email address on the account recovery screen. It guides the user through a secure, methodical recovery procedure and displays the possible recovery options.

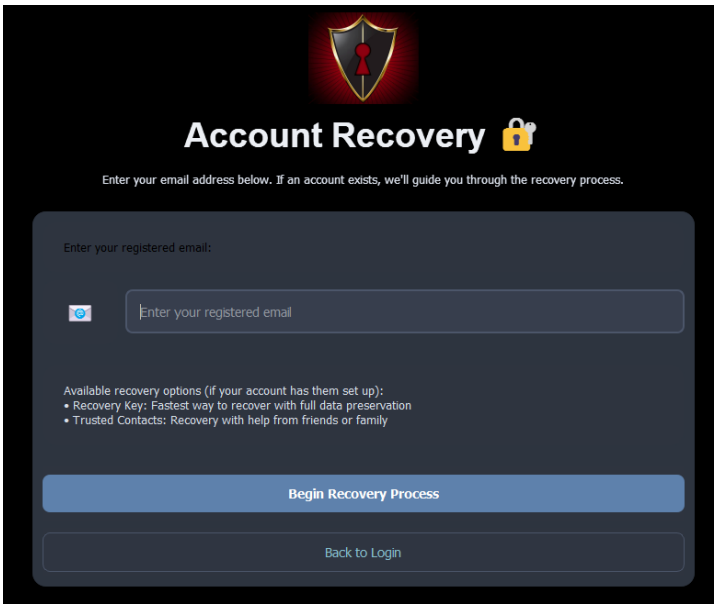
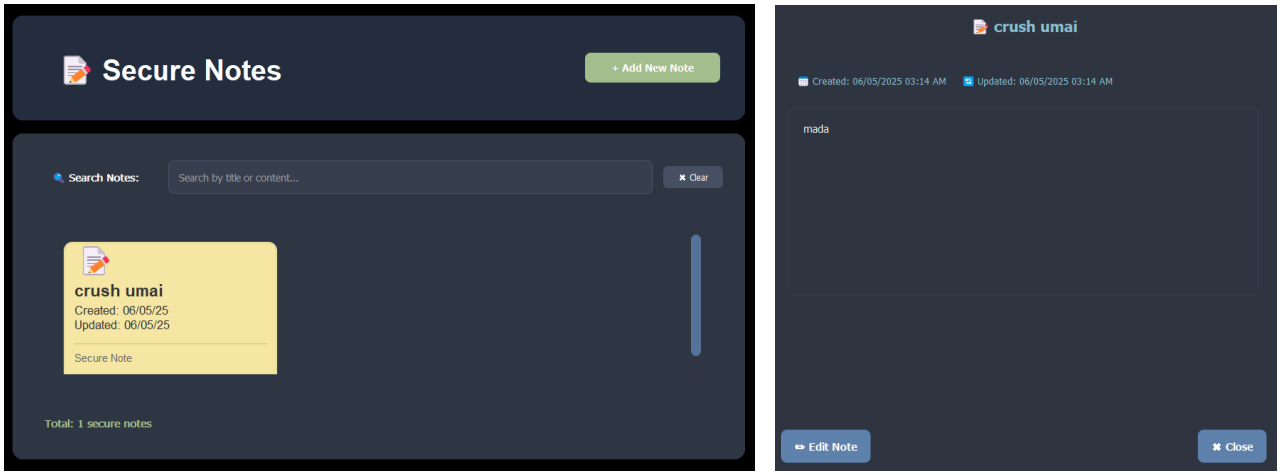


Fig. 14 *Account Recovery Interfaces*

5.1.5 Secure Note

Fig. 15 shows users the ability to securely organise their sensitive data by presenting a list of stored secure notes with search features, the ability to add new notes, and the option to view or edit a selected note.

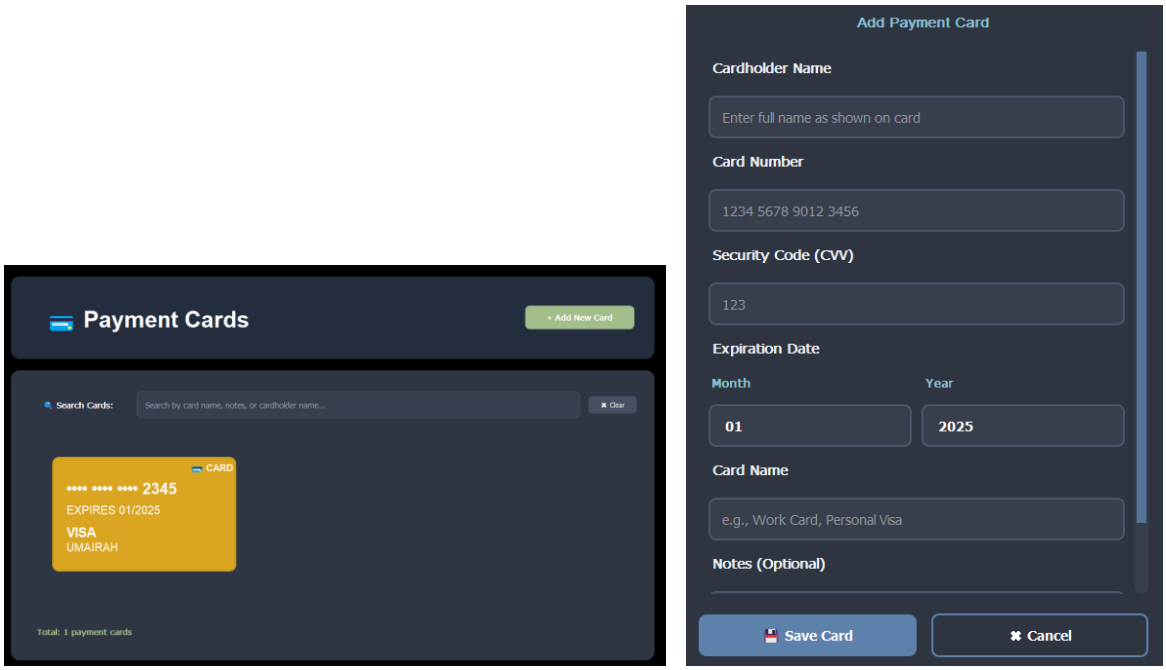


(a) (b)

Fig. 15 *PassGuard interfaces (a) Secure Note Interface; (b) Add New Note Interface*

5.1.6 Card Details

PassGuard allows users to securely store credit and debit card information through a simple form interface. All sensitive card details are automatically encrypted before storage, with card numbers masked to show only the last four digits. Users can easily add, search, edit, or delete their saved payment cards while maintaining complete security through encryption as in Fig.16.



(a) (b)

Fig. 16 *PassGuard interfaces (a) Payment Cards Interface; (b) Add New Card Interface*

5.1.7 TrustGuard

The TrustGuard interface detects for any security flaws by analysing stored passwords. It looks for weak passwords using a predetermined set of security criteria and highlights passwords that do not fit into those criteria. Next, it compares entries to find recurring passwords and uses the rockyou.txt wordlist to find common patterns. Furthermore, users have the option to reset their passwords for increased security if they are more than 90 days old because they have been identified as outdated. Fig.16 shows the interfaces for Trust Guard.

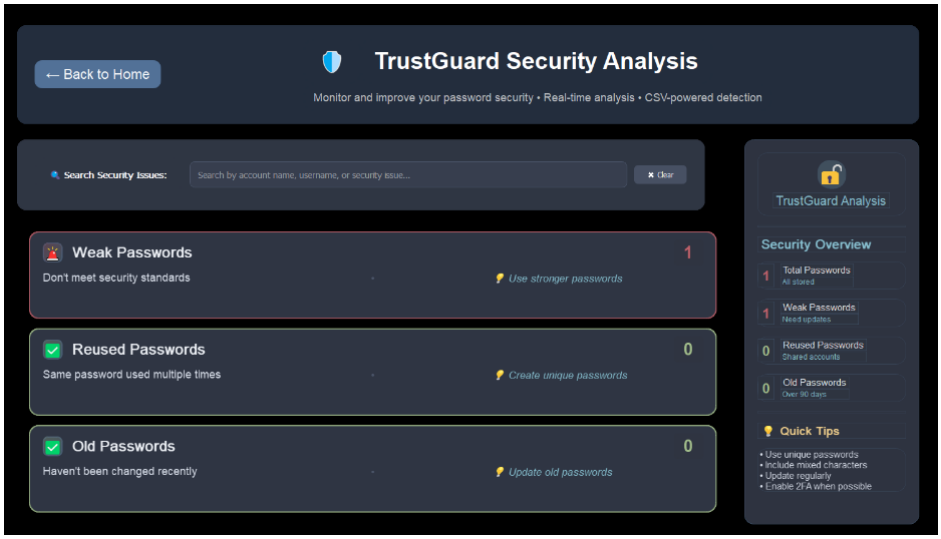


Fig. 17 *TrustGuard Interfaces.*

5.1.8 Breach Monitor

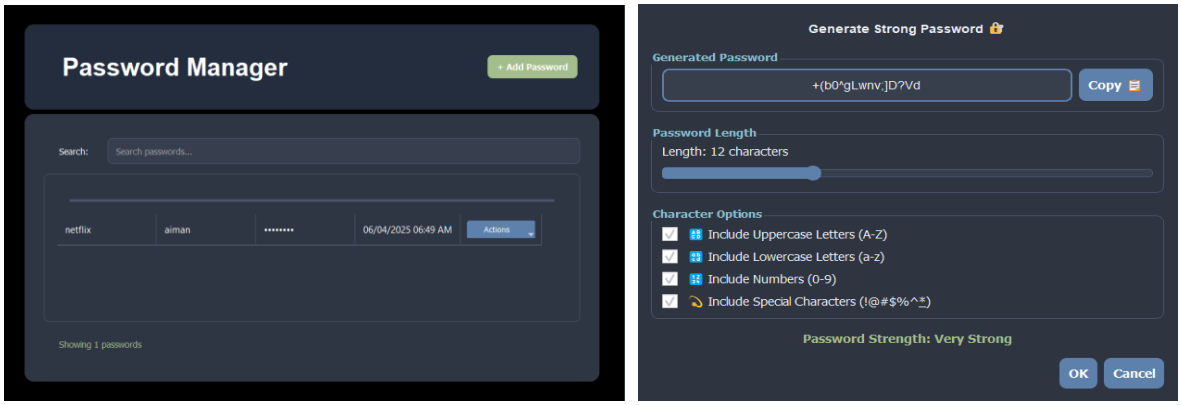
PassGuard's breach monitoring feature checks if user email addresses or passwords have been compromised in known data breaches using real-time security databases. Users can verify their email against breach records or test password security, with the system providing detailed breach information and personalized security recommendations when compromised credentials are detected as shown in Fig. 18.

(a) (b)

Fig. 18 *PassGuard interfaces (a) Breach Monitoring for Email Check Interface; (b) Breach Monitoring for Password Interface*

5.1.9 Password Manager

PassGuard's password manager securely stores login credentials with automatic encryption, allowing users to add, edit, and delete password entries for websites. The system uses envelope encryption with PBKDF2-based key derivation (100,000 iterations) and Fernet authenticated encryption (AES-128 + HMAC-SHA256). The system includes a built-in password generator for creating strong passwords, search functionality to find entries quickly, and one-click options to copy credentials to clipboard or open associated URLs. It validates password strength, detects reused passwords, and highlights weak entries to maintain security standards as in Fig. 19.



(a) (b)

Fig. 19 *PassGuard interfaces (a) Password Manager Interface; (b) Generate New Password Interface*

5.2 PassGuard Testing

This section discusses the testing of PassGuard. There are two types of testing, functionality testing and user acceptance testing. Functionality testing ensures the application fulfils all the requirements by verifying all the test names with the expected result. User acceptance testing was conducted with 21 respondents using form and the result.

5.2.1 Functionality Testing

The results of the functionality testing are presented here. Table 5 shows the result of functionality testing for PassGuard password manager applications.

Table 5 *Functionality Testing Result*

Test ID	Category	Test Name	Test Steps	Expected Result
TC-AUTH-001	Authentication	alid Login & OTP Verification	Enter valid credentials, complete OTP verification	Successful login
TC-AUTH-002	Authentication	Unauthorized Access Handling	Enter invalid credentials, observe generic response	Identical generic response to prevent user enumeration
TC-CRED-001	Credentials	RUD, Search, Copy & Password Generation	Add, edit, delete credentials (password, secure notes, payment), search and copy and generate strong passwords	All operations work with encryption and proper filtering
TC-SEC-001	Security	Breach Detection & Alerts	Detect breached emails and credentials; trigger alerts	Breaches identified and alert emails sent
TC-SEC-002	Security	Password Strength Analyzer	Analyze password strength and detect weak/reused passwords	Accurate strength analysis and reuse warnings
TC-REC-001	Recovery	Recovery Setup & Full Process	Setup recovery key, trusted contacts and verify OTP to complete recovery	Recovery key and contacts functional and recovery succeed
TC-SESS-001	Session	Auto & Manual Logout	Test inactivity timeout; test manual logout	Auto logout triggers: sessions cleared properly
TC-UI-001	UI	Navigation, Notes & Settings	Navigate app to manage secure notes and modify and save settings	mooth navigation: notes and settings saved correctly

Table 5 *(Cont.)*

Test ID	Category	Test Name	Test Steps	Expected Result
TC-EMAIL 001	Email	OTP, Recovery & Alert Emails	Test OTP, recovery, and alert email delivery and validate format	Emails delivered reliably with fallback functionality
TC-EMAIL-001	Email	OTP, Recovery & Alert Emails	Test OTP, recovery, and alert email delivery and validate format	Emails delivered reliably with fallback functionality
TC-INT-001	Intrusion	Lockout & Alert System	Trigger lockout after multiple failed OTP attempts; test photo capture and alert emails	Lockout activates; intruder photos captured, and alerts sent
TC-DB-001	Database	Encryption, Retrieval & Transactions	Store and retrieve encrypted data and test transaction reliability	Data encrypted/decrypted properly ensuring no data loss
TC-ERR-001	Error Handling	Input Validation & Resource Management	Test invalid inputs, network issues, and concurrency	Proper validation and graceful error handling
TC-SET-001	Settings	Account Lockout, Trusted Contacts & Master Email	Configure lockout duration, max attempts; add/remove up to 2 trusted contacts, change master email and erase all stored data	Settings applied correctly, trusted contacts managed, and data erased as requested

5.2.2 User Acceptance Testing

The PassGuard password manager application was tested for user acceptance through the distribution of user acceptance forms to 21 respondents. As shown in Fig. 20, the respondents consisted of students across two education levels and both genders. The form was divided into four sections: Account and Security, Recovery Features, Security Features, and Additional Features. The results were highly positive, with all respondents passing the evaluation. Tables 6 to 9 show the detailed results of the user acceptance testing.

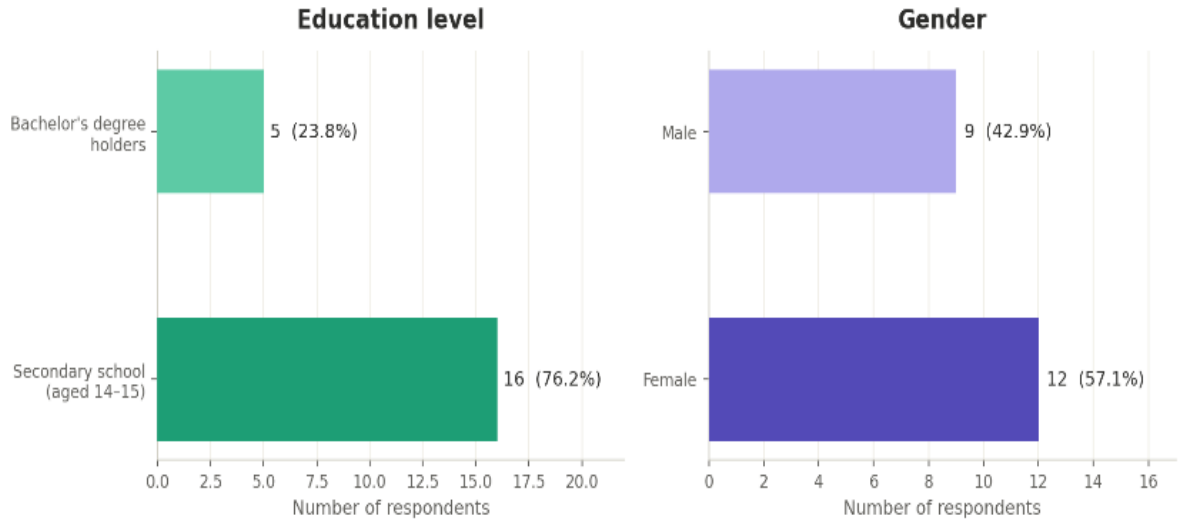


Fig. 20 *User Acceptance Testing demographic.*

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1270-1289

Table 6 *User Acceptance Testing Result for Account and Security*

Feature	Test Actions	Expected Results	Pass/Fail
Register new account	Click Register, enter username, email, password	Account created successfully	Pass
Email verification process	Check email, click verification link	Email verified & login enabled	Pass
Email verified & login enabled	Enter credentials, click Login	Login successful, home screen appears	Pass
OTP verification	Enter OTP code received via email	OTP accepted, access granted	Pass
Master password prompt	Try to access sensitive areas (e.g., complete data deletion)	Master password prompt appears	Pass

Table 7 *User Acceptance Testing Result for Recovery Features*

Feature	Test Actions	Expected Results	Pass/Fail
Set up recovery key	Go to Settings > Recovery, generate key	Recovery key generated & displayed	Pass
Set up trusted contacts	Add contact email in Recovery settings	Contact added, verification sent	Pass
Manage trusted contacts	Add/remove/verify contacts	Changes saved, status updated	Pass
Clear account recovery	Navigate recovery settings, clear setup	All recovery options removed	Pass
Recovery wizard navigation	Open recovery wizard, navigate between screens	Wizard flows logically between steps	Pass
Recover with recovery key	Enter recovery key in recovery wizard	Account recovered with data preserved	Pass
Recover with trusted contacts	Enter verification code from contact	Account recovered with data preserved	Pass

Table 8 *User Acceptance Testing Result for Security Features*

Feature	Test Actions	Expected Results	Pass/Fail
TrustGuard analysis	Open TrustGuard, view analysis	Weak/reused passwords identified	Pass
Password strength meter	Enter passwords of varying strength	Meter accurately reflects strength	Pass
Email verified & login enabled	Enter credentials, click Login	Login successful, home screen appears	Pass
Webcam capture	Trigger failed login attempts	Webcam activates, image captured	Pass
Breach monitoring	Run breach scan	Compromised passwords detected	Pass
Auto-logout	Set timeout, leave app idle	App logs out after timeout period	Pass

Table 9 *User Acceptance Testing Result for Additional Features*

Feature	Test Actions	Expected Results	Pass/Fail
Create secure note	Enter title and note content	Note saved successfully	Pass
Edit/delete note	Modify notes, then delete them	Changes saved, note deleted	Pass
Add payment card	Enter card details	Card saved with masked number	Pass
View card details	Select card, view details	Details displayed correctly	Pass
Password generator	Adjust length slider, toggle character options	Password generated according to specifications	Pass
Character exclusion handling	Generate passwords with no checkboxes selected	Appropriate error message displayed	Pass
Email notifications	Trigger security event	Alert email received	Pass

6. Conclusion

PassGuard effectively addresses the need for secure and efficient password management by integrating advanced features such as multi-factor authentication, unauthorized access detection, and real-time alerts. Developed using object-oriented programming principles and following the structured phases of the Waterfall methodology, the application ensures clarity, maintainability, and strong security standards. It provides key functionalities including secure credential storage, password generation, password strength analysis, and breach monitoring, delivering a reliable solution for protecting user data. In this version, the sign-in method is limited to traditional credentials combined with OTP verification, and the application's scope is focused on essential password management capabilities. Looking ahead, future enhancements include implementing passwordless authentication methods such as passkeys or authenticator-based logins to improve security and user convenience. There are also plans to expand PassGuard into a more integrated platform that can operate across a wide range of websites and services, offering users consistent and secure access wherever credentials are required.

Acknowledgement

The authors wish to express their gratitude to the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, for their support.

Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *A. A. A. Mohamed Shaharudeen, N. Rahim;* ***data collection:*** *A. A. A. Mohamed Shaharudeen, N. Rahim;* ***analysis and interpretation of results:*** *A. A. A. Mohamed Shaharudeen, N. Rahim;* ***draft manuscript preparation:*** *A. A. A. Mohamed Shaharudeen, N. Rahim. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] A. Hanamsagar, S. Woo, C. Kanich, and J. Mirkovic, "How Users Choose and Reuse Passwords." Available: https://publications.isi.edu/trpublic/pdfs/isi-tr-715.pdf

[2] D. Liginlal, I. Sim, and L. Khansa, 'How significant is human error as a cause of privacy breaches? An empirical study and a framework for error management', *Comput Secur*, vol. 28, no. 3–4, pp. 215–228, May 2019, doi: 10.1016/j.cose.2008.11.003.

[3] M. Fagan, Y. Albayram, M. M. H. Khan, and R. Buck, 'An investigation into users' considerations towards using password managers', *Human-centric Computing and Information Sciences*, vol. 7, no. 1, Dec. 2017, doi: 10.1186/s13673-017-0093-6.

[4] J. M. Biju, N. Gopal, and A. J. Prakash, 'CYBER ATTACKS AND ITS DIFFERENT TYPES', *International Research Journal of Engineering and Technology*, 2008, [Online]. Available: www.irjet.net

[5] B. MacIej, E. F. Imed, and M. Kurkowski, 'Multifactor Authentication Protocol in a Mobile Environment', *IEEE Access*, vol. 7, pp. 157185–157199, 2019, doi: 10.1109/ACCESS.2019.2948922.

[6] W. Jia, 'Analysis on Password Attack Model and Password Generation', in *Proceedings - 2022 International Conference on Computers, Information Processing and Advanced Education, CIPAE 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 145–149. doi: 10.1109/CIPAE55637.2022.00038.

[7] S. Murmu, H. Kasyap, and S. Tripathy, 'PassMon: A Technique for Password Generation and Strength Estimation', *Journal of Network and Systems Management*, vol. 30, no. 1, Jan. 2022, doi: 10.1007/s10922-021-09620-w.
[8] Senarath, U. S. (2021). Waterfall Methodology, Prototyping and Agile Development. https://doi.org/10.13140/RG.2.2.17918.72001.
[9] K. Conboy and N. Carroll, 'Implementing Large-Scale Agile Frameworks: Challenges and Recommendations', Mar. 01, 2019, *IEEE Computer Society*. doi: 10.1109/MS.2018.2884865.
[10] A. Anand and A. Uddin, 'Importance of Software Testing in the Process of Software Development', 2019. [Online]. Available: www.ijsrd.com
[11] H. U. Rahman, M. Raza, P. Afsar, H. U. Khan, and S. Nazir, 'Analyzing factors that influence offshore outsourcing decision of application maintenance', *IEEE Access*, vol. 8, pp. 183913–183926, 2020, doi: 10.1109/ACCESS.2020.3029501.