International Journal on Perceptive and Cognitive Computing (IJPCC) Vol 12, Issue 2 (2026)
https://doi.org/10.31436/ijpcc.v12i2.710

Sack Key: An Offline Mobile Password Manager Application

Muhammad Afiq Haikal Ahmad Tarmizi, Muhammad Anwar Mohd Asri, Norlia Md Yusof*

Department of Computer Science, Kulliyyah of Information & Communication Technology,
International Islamic University Malaysia, Gombak, Malaysia

*Corresponding author: norliayusof@iium.edu.my
(Received: 7th June 2026; Accepted: 3rd July, 2026; Published on-line: 30th July, 2026)

Abstract— The rapid growth of online services has significantly increased the number of credentials users must manage, leading to widespread password reuse and weak authentication practices. This issue is particularly severe on mobile devices, where usability constraints often conflict with secure password management. Existing password managers commonly rely on cloud-based storage and subscription models, which introduce privacy concerns and increase exposure to centralized attacks. This paper presents Sack Key, a privacy-centric mobile password manager designed with an offline-first architecture to enhance security and user control. Sack Key securely stores credentials locally using AES-256 encryption and eliminates dependency on cloud infrastructure. The system incorporates multi-layer security mechanisms, including master password, Two-Factor Authentication (2FA), automatic session locking and privacy-preserving features such as manual encryption and decryption. Additionally, Sack Key supports secure peer-to-peer password sharing over local networks without internet connectivity. The results of functional, security and user testing indicate that Sack Key provides reliable password protection, strong user trust in offline security and intuitive user experience. The findings demonstrate that an offline-first password manager can effectively balance usability, security and privacy in mobile environments.

Keywords— Password Manager, Offline-First, AES Encryption, Two-Factor Authentication, Mobile Security, Privacy.

I. Introduction

Mobile devices have become essential tools for communication, online transactions and access to digital services. As a result, users increasingly rely on smartphones to manage sensitive information such as banking credentials, personal accounts and private communications. Despite advancements in mobile security technologies, passwords remain the dominant authentication mechanism across most applications. However, managing strong and unique passwords on mobile devices is challenging due to small interfaces, frequent app switching and user preference for convenience.

Studies indicate that a significant proportion of data breaches are caused by human-related factors, including weak passwords, password reuse and phishing attacks [1]. These risks are amplified in mobile environments, where users often prioritize speed over security. Password managers have emerged as practical solutions to mitigate these issues by securely storing and generating credentials. However, many existing password managers rely heavily on cloud-based storage and continuous internet connectivity. This dependency introduces privacy risks, centralized attack surfaces and trust concerns among users who prefer full control over their sensitive data.

In addition, usability remains a critical challenge. Complex interfaces and advanced security configurations often discourage non-technical users from adopting password managers [2], [3], [4] reducing their effectiveness. Furthermore, some password managers have been shown to expose credentials through insecure autofill mechanisms [5], improper memory handling [6], or insufficient authentication protection.

To address these limitations, this paper proposes Sack Key, an offline-first [7] mobile password manager designed with a strong emphasis on privacy, security and usability. Sack Key stores all credentials locally in encrypted form using AES-256 and does not require cloud synchronization. The system integrates Two-Factor Authentication (2FA), automatic session locking and additional privacy features such as manual encrypt and decrypt to protect users in coercive or high-risk situations. Secure local network password sharing is also supported, enabling controlled credential exchange without internet exposure.

The main contributions of this work are as follows:

1) The design and implementation of an offline-first mobile password manager that eliminates cloud dependency.
2) Integration of multi-layer security mechanisms, including AES-256 encryption and 2FA.
3) Support for secure peer-to-peer password sharing over local networks.
4) Evaluation of usability and security effectiveness through user acceptance testing.

https://doi.org/10.31436/ijpcc.v12i2.710

II. RELATED WORK

Password managers have been widely examined in academic research as an effective countermeasure against weak passwords, password reuse and credential overload. Prior studies consistently demonstrate that password managers enable users to adopt unique and high-entropy passwords, thereby improving overall authentication security [1], [8], [9]. However, recent literature emphasizes that the security of password managers cannot be evaluated solely based on encryption strength, but must also consider architectural design, runtime behaviour and user interaction models [6].

Recent reviews and empirical studies published from 2024 to early 2026 highlight that most modern password managers rely on cloud-based synchronization to support multi-device access [5]. Although these systems typically employ strong encryption algorithms such as AES-256 or modern authenticated encryption schemes, researchers argue that cloud-centric designs introduce centralized trust assumptions and additional attack surfaces [10]. As a result, password manager security must be analysed holistically rather than focusing exclusively on data-at-rest protection.

A. *Security Vulnerabilities and Threat Models*

Recent academic work has expanded the threat model of password managers beyond traditional database breaches. Empirical studies show that decrypted credentials may remain in system memory during runtime, exposing them to memory dump and forensic attacks when a device is compromised [6], [11]. These findings challenge the assumption that encrypted storage alone is sufficient to protect sensitive credentials.

Browser-integrated autofill mechanisms have also been identified as a significant source of credential leakage. Research demonstrates that automated credential injections can expose passwords to malicious or hidden input fields, particularly in phishing and script-based attacks [5], [12]. Consequently, scholars increasingly recommend restricting automatic disclosure and introducing explicit user confirmation or secondary authentication before revealing sensitive information.

On mobile platforms, recent studies further reveal weaknesses in poorly implemented multi-factor authentication mechanisms. Insecure local verification and flawed session handling may allow attackers to bypass One-Time Password (OTP) or biometric checks, undermining the intended security benefits of such mechanisms [10], [13]. These findings highlight the importance of secure local enforcement rather than relying solely on authentication at login.

B. *Usability, Trust and Accessibility*

Despite the security benefits demonstrated, usability remains a major barrier to password manager adoption. Longitudinal studies involving first-time users indicate that perceived complexity, fear of credential loss and limited understanding of security mechanisms significantly reduce long-term usage [2], [3]. Importantly, these challenges are largely behavioural rather than technical.

Human-centered security research further shows that poor usability directly undermines security outcomes, as users adopt insecure coping strategies such as reusing master passwords, disabling security features, or avoiding password managers altogether [3]. Recent accessibility-focused studies reveal that users with visual impairments or low technical proficiency face additional challenges when interacting with password manager interfaces, further limiting the effectiveness of existing solutions [11]. These findings emphasize that secure systems must be designed with usability and accessibility as core requirements rather than secondary considerations.

C. *Offline and Client-Centric Password Managers*

In response to privacy and trust concerns associated with cloud-based synchronization, academic researchers have proposed offline-first or fully local password manager architectures [9], [14]. These designs aim to minimize centralized attack surfaces by ensuring that credentials and cryptographic keys remain solely on the user's device. Studies suggest that offline architecture can significantly reduce exposure to server-side attacks and third-party data handling risks.

However, existing offline solutions discussed in the literature often lack modern usability features, secure authentication layers, or mobile-centric optimization [13]. As a result, their real-world adoption remains limited despite strong theoretical security advantages. This gap indicates the need for practical implementations that balance offline security with usability and accessibility, particularly in mobile environments.

D. *Research Gap and Motivation*

Based on recent academic literature, three key gaps are identified. First, many studies emphasize encryption strength while under addressing runtime exposure risks such as memory-based attacks. Second, usability and accessibility challenges continue to hinder adoption, especially among non-technical users. Third, although offline password managers are theoretically appealing, few mobile-oriented implementations successfully integrate strong security guarantees with user-centered design.

To address these gaps, this research proposes Sack Key, a mobile password manager designed with a fully offline

https://doi.org/10.31436/ijpcc.v12i2.710

architecture, strong local encryption, enhanced authentication during credential access and usability-driven interaction. By grounding its design in recent academic findings (see Table 1), this work aims to contribute toward more secure and accessible password management solutions for everyday users.

TABLE I
COMPARISON OF FEATURES OF PASSWORD MANAGER SYSTEM

Ref	Platform	Encryption algorithms	Storage Architecture	Cloud Dependency	Authentication Support
[14]	Offline	AES-256	Local encrypted vault	None	Master password
[5]	Web	AES-based	Browser-stored credentials	Required	Browser authentication
[10]	Mobile	OS-managed	OS-level credential handling	Not required	2FA / Password-less
[6]	Mobile	AES-based	Local encrypted storage	Optional	Master password
[15]	Mobile	AES-based	Local + runtime memory	Optional	Master password
Sack Key	Mobile	AES-256	Fully local encrypted vault	None	Master password + 2FA

III. METHODOLOGY

This section describes the architectural design and security-oriented development approach adopted for Sack Key. The system was designed to operate entirely offline while maintaining strong protection of sensitive credentials and ensuring usability for non-technical users.

A. Development Methodology

Sack Key was developed following the Object-Oriented System Development (OOSD) methodology [16]. It consists of planning, analysis, design, development and testing phase (see Fig. 1). Security risks such as unauthorized access, insecure local storage and mobile-specific attacks were identified early and addressed through architectural and implementation decisions. This approach ensured that both functional and non-functional security requirements were systematically enforced.

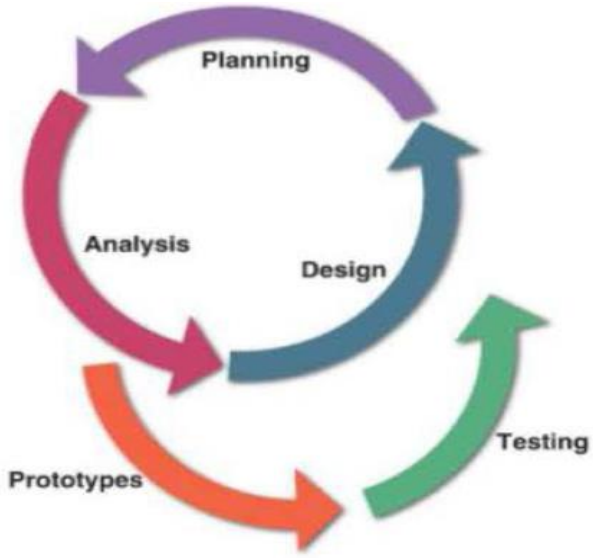

Fig. 1. OOSD adopted in Sack Key

B. System Architecture Design

Sack Key adopts a standalone mobile application architecture with an offline-first design. The architecture is composed of four main layers as illustrated in Fig. 2.

1) Presentation Layer: Provides the mobile user interface developed using Flutter. It manages user interactions, input validation and secure display of credential data by masking sensitive fields by default.

2) Application Logic Layer: Handles core functionalities including authentication, password vault operations, session management and access control. All user requests are validated before any data operation is executed.

3) Data Storage Layer: Utilizes SQLite for local data persistence. All stored credentials are encrypted prior to storage, ensuring that plaintext data is never written to disk.

4) Security Layer: Implements AES-256 encryption, 2FA, automatic session locking and privacy-preserving mechanisms such as manual encrypt and decrypt. This layer operates across all system components to enforce consistent protection of sensitive information.

https://doi.org/10.31436/ijpcc.v12i2.710

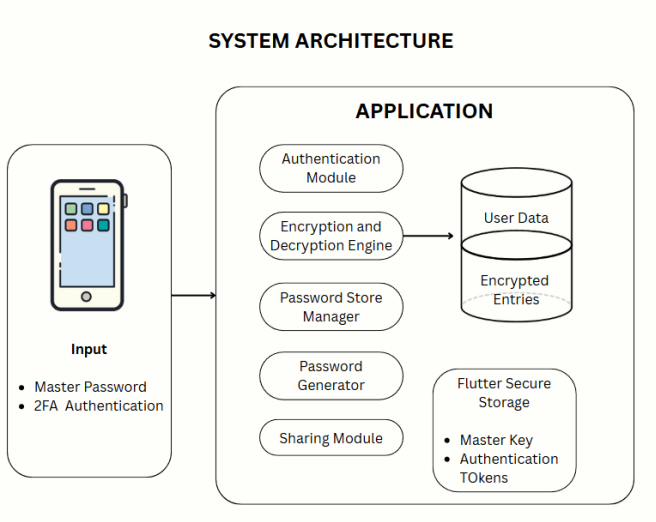

Fig. 2. Overall system architecture of Sack Key

C. Threat Model

To clarify the security scope of Sack Key, this subsection defines the threat model considered during system design. The primary threat explicitly addressed in the current implementation is SQL injection targeting the local SQLite database; all database queries are implemented using parameterized statements combined with input validation at the application logic layer, preventing malicious query fragments from being injected through credential input fields.

Runtime exposure and device-compromise threats are only partially addressed in the current design and warrant more explicit discussion, given their prominence in the literature reviewed in Section II-A [6], [11]. Decrypted vault data exists in memory only while a session is active and is dereferenced once the session ends or the application is locked (Section III-D). However, because Sack Key is implemented in Dart on the Flutter runtime, memory reclamation is handled by a managed garbage collector rather than by explicit, immediate zeroing of memory; consequently, residual data may persist in memory for a short period after dereferencing, and Sack Key does not currently perform explicit memory wiping. This is consistent with the runtime-exposure risk documented for password managers generally in prior work [6], [11], rather than a limitation unique to this system, but it means the memory-dump attack surface identified in the literature is not yet fully closed here. Similarly, the master key and authentication tokens are stored using Flutter Secure Storage (Fig. 3), which delegates to platform-level secure storage (Android Keystore or iOS Keychain) and provides hardware-backed protection against extraction on a non-compromised device. On a rooted or jailbroken device with kernel-level access, however, an attacker may still be able to circumvent these platform protections or perform live memory scraping; Sack Key does not currently implement root/jailbreak detection or anti-tampering measures to detect or respond to such compromise. Cold-boot memory extraction and backup-file extraction were likewise considered during design discussions but are not yet explicitly mitigated or empirically evaluated. Addressing these runtime and device-compromise threats, for example through explicit memory wiping where the Flutter platform permits it, root/jailbreak detection, and anti-tampering safeguards is identified as a priority direction for future development (see Section V).

D. Implementation Overview

The system is divided into four tightly integrated functional modules to support secure password management. The authentication, vault management, encryption and local sharing modules interact through the application logic layer to ensure controlled access and secure data flow.

1) Authentication Module: Manages user registration and login using a master password and optional 2FA via OTP. Credentials and security settings are stored using secure local storage to prevent exposure through local files or memory dumps. Fig. 3 and Fig. 4 depicts the user authentication and OTP verification interface respectively.

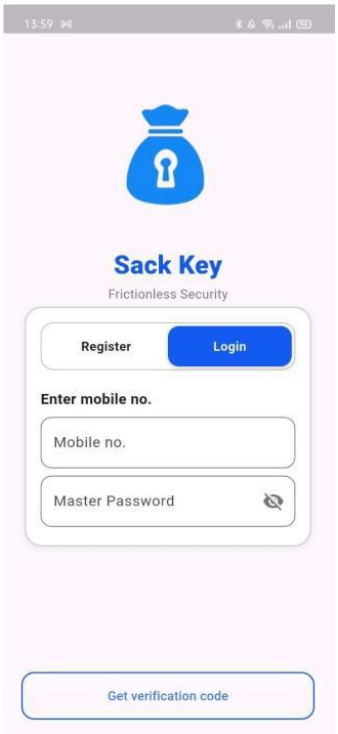

Fig. 3. User authentication and OTP verification interface

https://doi.org/10.31436/ijpcc.v12i2.710

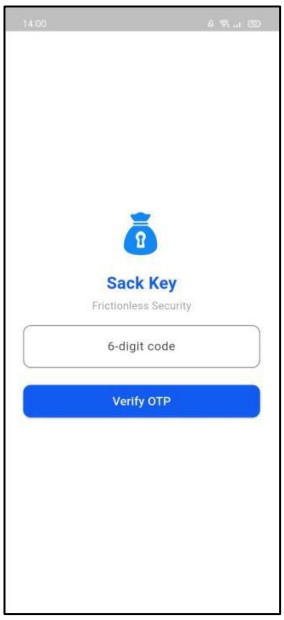

Fig. 4. User authentication and OTP verification interface

2) Password Vault Module: This module is combination of password store manager and password generator in the system architecture. It allows users to generate, view (see Fig. 5), update (see Fig. 6) and delete (see Fig. 7) stored credentials. Vault data is decrypted only in memory after successful authentication and is immediately cleared when the session ends or the application is locked.

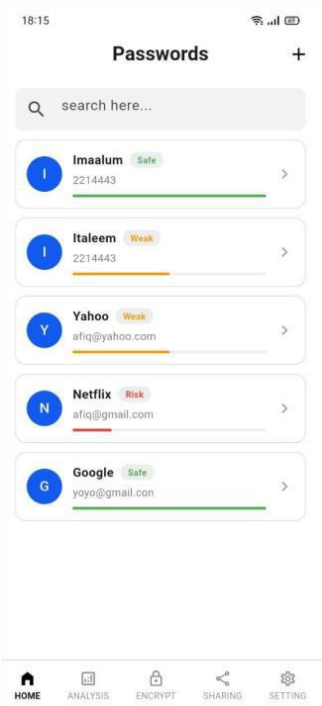

Fig. 5. Encrypted password vault interface

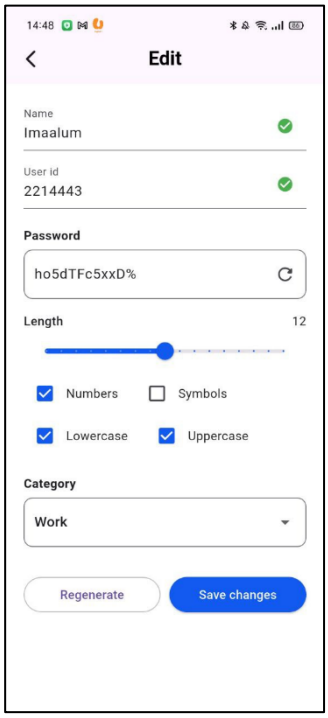

Fig. 6. Edit password interface

Fig. 7. Delete password interface

https://doi.org/10.31436/ijpcc.v12i2.710

3) Encryption Module: Provides AES-256 encryption for all sensitive data (see Figure 8). Encryption keys are derived from the user's master password, ensuring that stored data remains protected even if device storage is compromised.

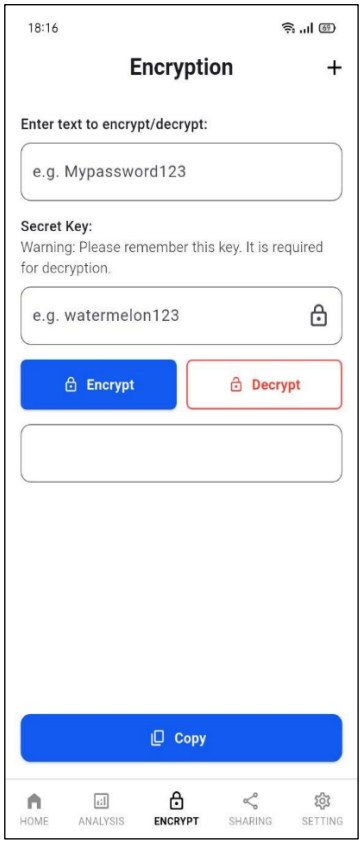

Fig. 8. Encryption and decryption feature output

4) Local Network Sharing Module: Enables secure peer-to-peer password sharing over a trusted local network using Wi-Fi Direct (see Figure 9). Shared credentials are encrypted prior to transmission and are decrypted only after successful authentication on the receiving device.

The architectural design of Sack Key integrates offline-first principles with layered security controls to protect sensitive credentials. By combining secure local storage, strong encryption and modular design, the system achieves a balance between robust security and practical usability.

IV. Results and Discussions

This section presents the results obtained from the development and evaluation of Sack Key. The evaluation focuses on functional correctness, security behavior and usability based on controlled testing and user acceptance testing.

A. *Functional Testing Results*

1) All core functionalities of Sack Key operated as intended during testing. User registration and authentication using a master password and 2FA were completed successfully without critical failures. After authentication, users were consistently granted access to the encrypted password vault.

Fig. 9. Local network password sharing interface

2) Password vault operations, including adding, editing, viewing, searching and deleting credentials, were executed correctly. Stored credentials remained encrypted at rest and decrypted values were displayed only temporarily in memory. The password generator produced strong and random passwords, supporting improved password practices among users.

B. *Security Testing Results*

Security mechanisms were validated through repeated access and usage scenarios. The system successfully enforced encryptions for all stored credentials using AES-256, preventing plaintext exposure in local storage. Automatic session locking was triggered correctly after periods of inactivity, requiring re-authentication before vault access.

https://doi.org/10.31436/ijpcc.v12i2.710

Beyond functional verification, the SQL-injection threat identified in Section III-C was assessed through two complementary checks. First, a code-level review confirmed that all database queries interacting with user-supplied input are implemented using parameterized statements, with no raw string concatenation used to construct SQL queries. Second, manual testing was conducted by submitting common SQL-injection payloads (e.g., ' OR '1'='1, '; DROP TABLE users; --) through the credential input fields. In all cases, the payloads were treated as literal input by the parameterized queries rather than executable SQL fragments; each attempt simply failed as an invalid login rather than altering the query structure or exposing any stored data, and no unauthorized data access or query manipulation was observed.

2FA effectively added an additional security layer during login and sensitive operations. The password vault feature functioned as intended by displaying non-sensitive data when activated, providing protection in coercive scenarios. Manual encryption and decryption feature also performed reliably, allowing users to securely encrypt custom text using user-defined keys.

C. *Local Network Sharing Testing Result*

The feature establishes a direct connection between two devices using Wi-Fi Direct, without relying on an intermediary access point or internet connectivity. The local peer-to-peer password sharing feature was tested within a trusted local network environment. The system successfully established secure connections between devices and transmitted encrypted credential data without internet connectivity. Shared credentials were only accessible after successful authentication on the receiving device, confirming controlled and secure data exchange.

Connection status messages and error notifications were displayed clearly when sharing was successful or interrupted, ensuring transparency during the sharing process.

Because Wi-Fi Direct relies on device-level pairing rather than a centralized trust authority, mitigating man-in-the-middle risk requires an explicit device-verification step during pairing. In Sack Key, the receiving device verifies the sharing device by scanning a QR code displayed on the sharing device before the connection is established, over the same local network, providing mutual device verification and preventing an unverified device from joining the pairing process. A formal evaluation of this QR-based pairing mechanism against relay or proximity-based attacks is noted as an area for future work.

D. *User Testing Results*

User Acceptance Testing (UAT) was conducted with users who had limited prior experience with password manager applications [2]. Table II shows that most users reported that the application was easy to understand and navigate. Authentication flows and password creation processes were rated as intuitive, while the password generator was highlighted as particularly useful.

TABLE II
OVERALL USER ACCEPTANCE TESTING

Test Area	Key Criteria	Average Rating (1–5)
Registration and Login	Ease, clarity, smoothness	4.6
Add Password	Ease, form clarity, experience	4.3
Manage Passwords	Navigation, edit/delete, clarity	3.9
Search Function	Speed, accuracy, ease	4.8
Security Features	Trust, safety, unlocking	4.2
Overall User Experience	Usability, security, acceptance	4.2

It should be noted that the "Security Features" rating in Table II reflects participants' subjective sense of trust and perceived safety while using the application, rather than a formal measure of technical security. Perceived trust and validated security are assessed separately in this study: user trust is captured through the acceptance testing reported here, while technical security is evaluated through the encryption enforcement and SQL-injection validation described in Section IV-B.

Some usability limitations were identified, including limited interaction options within the password list view and the absence of clearly visible delete actions. Despite these minor issues, users expressed strong confidence in the system's offline security model and overall reliability.

The results demonstrate that Sack Key effectively addresses key limitations of existing mobile password managers, particularly in terms of privacy, offline functionality and usability for non-technical users. The offline-first architecture successfully eliminates cloud dependency, reducing exposure to centralized data breaches and third-party trust concerns.

From a security perspective, the integration of AES-256 encryption, 2FA and automatic session locking provides layered protection against unauthorized access [4], [6], [9]. The manual encryption features further strengthen user privacy in coercive or high-risk scenarios. These design choices align with prior findings that emphasize minimizing attack surfaces and securing sensitive data at rest and in memory.

The local network sharing feature demonstrates a practical alternative to cloud-based synchronization by enabling secure credential exchange within trusted environments. While effective, this approach inherently limits scalability and cross-network access, reflecting a deliberate design trade-off favoring privacy over convenience.

UAT results indicate that strong security mechanisms can be implemented without significantly increasing user complexity. Most users were able to perform core tasks such as authentication, password storage and retrieval with minimal guidance. This suggests that security-focused mobile applications can remain accessible when interfaces are carefully designed. However, minor usability limitations observed during testing highlight the trade-off between simplicity and feature visibility.

Overall, the discussion confirms that Sack Key achieves a balanced design that prioritizes user control and data privacy while maintaining acceptable usability. The findings support the feasibility of offline-first password management systems as viable solutions for users who prefer strong security without reliance on external infrastructure.

V. Conclusion

This paper presented Sack Key, a privacy-centric mobile password manager designed to address security and usability challenges associated with cloud-dependent credential management solutions. By adopting an offline-first architecture, Sack Key eliminates reliance on centralized storage while maintaining strong protection of sensitive data through AES-256 encryption and secure local storage mechanisms.

The system integrates multiple security layers, including a master password, 2FA, automatic session locking and privacy-preserving features such as manual encrypt and decrypt. In addition, secure peer-to-peer password sharing over local networks enables controlled credential exchange without internet exposure. The application was implemented using Flutter, Dart and SQLite, ensuring portability and efficient local performance.

Evaluation results demonstrate that Sack Key operates reliably under normal usage conditions and is accessible to non-technical users. Functional testing confirmed correct behavior of core features, while user acceptance testing indicated positive perceptions of usability and trust in the offline security model. Although minor interface improvements were identified, the system successfully achieves a balance between security, privacy and ease of use.

Sack Key demonstrates the feasibility of an offline-first password management approach for mobile environments. Future work may extend this solution through biometric authentication, encrypted backup mechanisms and enhanced sharing controls, further strengthening security while preserving user autonomy.

As identified in Section III-C, future work should also prioritize runtime memory protection (e.g., explicit memory wiping where the platform allows it) and device-compromise safeguards such as root/jailbreak detection and anti-tampering measures, to close the residual attack surface highlighted by prior research on memory-based credential exposure [6], [11].

ACKNOWLEDGMENT

This study received no external funding.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

AUTHORS CONTRIBUTION STATEMENT

All authors contributed equally to this work.

DATA AVAILABILITY STATEMENT

There is no external or third-party data that support the findings of this study.

ETHICS STATEMENT

This study did not require ethical approval

References

[1] Verizon, *2025 Data Breach Investigations Report*, Verizon Enterprise, 2025.

[2] P. A. Cabarcos, M. G. López and J. L. Vázquez, "The more accounts I use, the less I have to think: A longitudinal study on the usability of password managers for novice users," in *Proc. USENIX Symp. Usable Privacy Secur. (SOUPS)*, 2025.

[3] A. Ponticello, R. Kuber and S. Branham, "How blind and low-vision users manage their passwords," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, Nov. 2025, doi: 10.1145/3719027.3765081, 2025.

[4] G. W. W. Mukti and R. Roestam, "Enhancing password manager application security by root detection with usability and security evaluation," in *Proc. 8th Int. Conf. Informatics Comput. (ICIC)*, pp. 1–7, doi: 10.1109/ICIC60109.2023.10382115, Dec. 01, 2023.

[5] Y. Fu and D. Wang, "Leaky autofill: An empirical study on the privacy threat of password managers' autofill functionality," in *Proc. Annu. Comput. Secur. Appl. Conf. (ACSAC)*, pp. 288–303, doi: 10.1109/ACSAC63791.2024.00037, Nov. 2024.

[6] E. Chatzoglou, V. Kampourakis, Z. Tsiatsikas, G. Karopoulos and G. Kambourakis, "Keep your memory dump shut: Unveiling data leaks in password managers," in *IFIP Adv. Inf. Commun. Technol.*, vol. 710. Cham, Switzerland: Springer, pp. 63–78, doi: 10.1007/978-3-031-65175-5_5, 2024.

[7] S. H. Pothineni, "Offline-first mobile architecture: enhancing usability and resilience in mobile systems," *Journal of Artificial Intelligence General Science (JAIGS)*, vol. 7, issue 1, Dec. 2024. [Online]. Available: https://www.researchgate.net/publication/393910615_Offline-First_Mobile_Architecture_Enhancing_Usability_and_Resilience_in_Mobile_Systems

[8] D. Beh, N. Kaur, S. Verma and A. Joshi, "Cryptographic encryption techniques in a password manager," in *IET Conf. Proc.*, vol. 2023, no. 11, pp. 346–353, doi: 10.1049/icp.2023.1803, Jan. 2023.

[9] H. Alsharaya, "Password managers: A critical review of security, usability and innovative designs," *J. Comput. Sci. Inst.*, vol. 36, pp. 328-335, June 2025.

[10] A. T. Mahdad and N. Saxena, "Mobile login bridge: Subverting 2FA and passwordless authentication via Android Debug Bridge," in *Proc. 21st Int. Conf. Privacy, Security Trust (PST)*, pp. 1–12, doi: 10.1109/PST62714.2024.10788081, Aug. 2024.

[11] E. Chatzoglou, V. Kampourakis, Z. Tsiatsikas, G. Karopoulos and G. Kambourakis, "Unmasking the hidden credential leaks in password managers," *Comput. Secur.*, vol. 140, Art. no. 103620, doi: 10.1016/j.cose.2024.103620, 2025.

[12] C. Anliker, M. Staub and D. Basin, "Phishing attacks against password manager browser extensions," in *Proc. USENIX Secur. Symp.*, 2025.

[13] M. Jubur, A. Alsaadi and K. Almarhabi, "An in-depth analysis of password managers and two-factor authentication (2FA) schemes," *ACM Comput. Surv.*, early access, doi: 10.1145/3711117, Jan. 2025.

[14] M. Abdulkadir, S. Alketbi, H. Lamaazi, R. Altamimi, S. Alblooshi and A. Lakas, "Vault-PMS: A vault-based password management system for secure offline data storage," in *Proc. 20th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, pp. 1510–1515, doi: 10.1109/IWCMC61514.2024.10592442, May 2024.

[15] A. Gangwal, S. Singh and A. Srivastava, "AutoSpill: Credential Leakage from Mobile Password Managers," *Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy*, pp. 39–47, doi: https://doi.org/10.1145/3577923.3583658. Apr. 2023.

[16] S. Tilley, *System Analysis and Design*, 12th ed., Boston, USA: Cengage, 2020.