Group Number : 09

Lock Box - Offline Password Manager

Domain: Cybersecurity & Data Protection
Guide Name: Dr. C Srinivasan
Date: 23-07-2025

Sl.No	Student Name	Roll No
1	Adithya N S	CB.EN.U4CYS22002
2	Anaswara Suresh M K	CB.EN.U4CYS22007
3	C S Amritha	CB.EN.U4CYS22016
4	R Sruthi	CB.EN.U4CYS22051

Problem Statement Recap

Most existing password managers are either too complex for everyday users or rely heavily on cloud storage, raising privacy concerns. There is a need for a secure, offline password manager that combines strong encryption with essential features like autofill, timeout lock, and an intuitive interface for easy navigation.

Motivation & Significance

- The user interface in tools like KeePassXC is too complex for normal users, making it difficult for everyone to manage their passwords easily.
- Storing passwords offline offers better safety and control compared to cloud-based managers, which come with privacy risks.

- We are developing an **existing software product with added features**, aimed at enhancing usability and security.
- The product is an **offline password manager** that is intended to be **publishable**.
- It solves the need for a **secure and user-friendly** solution by removing complex interfaces and avoiding reliance on cloud storage
- It is particularly useful for **non-technical individuals** who want to keep their data safe and **businesses**
- Key features include **autofill, timeout lock, strong encryption, simple, intuitive user interface**
- It requires only **basic hardware** designed to run efficiently on **most personal computers without external dependencies**

Literature Survey

- **Desktop Framework**
  - *KeePassXC:* Built with Qt (C++) – cross-platform GUI.
  - *Bitwarden:* Built with Electron – uses HTML/CSS/JS for desktop apps.

- **Key Derivation Functions**
  - Both use **PBKDF2** and **Argon2** to derive strong keys from the master password, resisting brute-force attacks.

- **Encryption Mechanism**
  - *KeePassXC:* Uses **AES-256** to encrypt local KDBX database files.
  - *Bitwarden:* Uses **AES-256 + HMAC-SHA256**, encrypts data client-side before syncing to the cloud.

Literature Survey

- **Offline vs Online Model**
  - *KeePassXC:* Fully **offline**; passwords stored locally.
  - *Bitwarden:* **Cloud-based**; syncs only encrypted data (zero-knowledge architecture).

- **Memory Handling & Security**
  - *KeePassXC:* Uses **Salsa20** for in-memory encryption, with locking to reduce leak risk.
  - *Bitwarden:* Holds decrypted data in RAM temporarily; uses auto-timeout/manual logout to manage memory.

- **Zero-Knowledge Architecture**
  - *Bitwarden:* Implements full **zero-knowledge**; servers never see plain text.
  - *KeePassXC:* No server involved, so zero-knowledge not applicable.

Project Objectives

Main Objective

Develop a secure, offline-first password manager without internet dependency or cloud storage.

Sub-Objectives

- Implement AES-256 encryption with key derived using master password
- Create intuitive cross-platform user interface
- Optimize for fast credential retrieval and minimal resource usage

Expected Inputs

- Master Password, Password/ login credentials , Settings input, search query

Expected Outputs

- Decrypted Credentials, Encrypted Database, Authentication Status, Timeout/Auto-lock

Tools, Methodologies, and Technologies

Programming Languages & Frameworks

- **GUI:** Electron
- **Backend:** C++, SQLite for local database, Node.js

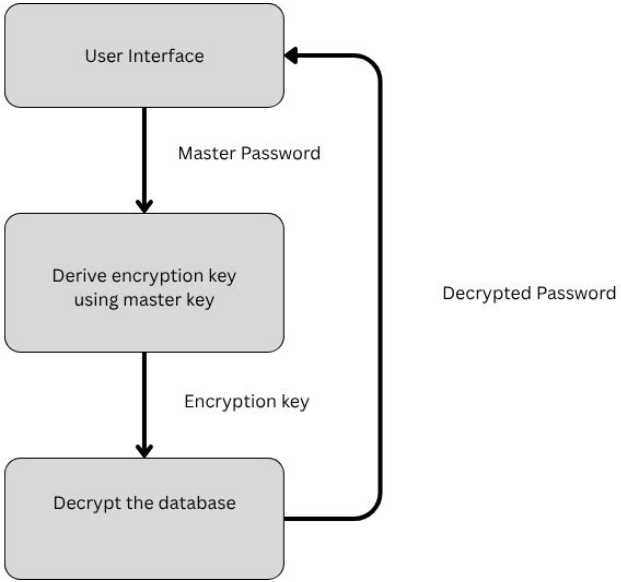
Technologies

- **Libraries:** AES-256-GCM, Salsa20, sqlite3
- **Key Derivation Function:** Argon2id

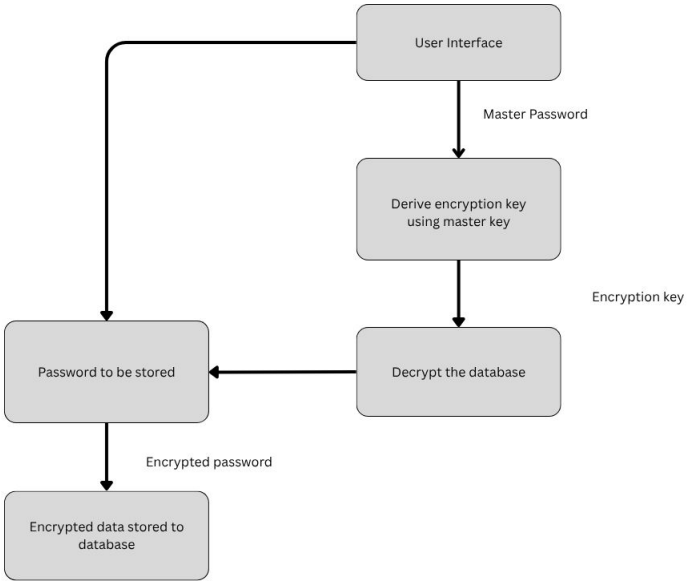
Development Tools

- **IDE:** Visual Studio Code
- **Version Control:** Git, GitHub

Data Flow



Retrieving password



Adding password

Methodologies

Given the modular project structure, we are adopting the **Agile methodology with the Scrum framework** to enable iterative feature delivery, sprint-based task tracking, and continuous feedback-driven improvements.

Deliverables

1. **Source Code Repository:** Complete codebase with documentation
2. **Desktop Application:** Windows
3. **Technical Documentation**
4. **User Manual:** Installation and usage instructions
5. **Demo Videos:** Feature demonstrations and tutorials
6. **Research Paper:** Technical methodology and results documentation

Comprehensive 8-Month Project Plan (July 2025 - February 2026)

Phase 1: Foundation & Research (July - August 2025)

Month 1: July 2025 - Research & Analysis

- Comprehensive study of existing password managers
- Academic paper analysis on cryptographic implementations and secure storage
- User requirement gathering
- Threat modeling workshop using STRIDE methodology
- Technology stack finalization (Python/C++ for desktop, SQLite for database)
- Cryptographic framework selection (AES-256, Argon2id)

Month 2: August 2025 - Design & Prototyping

- Database schema design for credential storage
- Authentication flow
- Key derivation and encryption protocol documentation
- User interface wireframe for desktop application
- Project repository initialization

Phase 2: Core Development (September - October 2025)

Month 3: September 2025 - Core Security Engine

- encryption/decryption module implementation
- Secure random number generation for passwords and salts
- Key derivation function implementation
- SQLite database creation with encryption layer
- Master password validation system implementation
- Encrypted credential storage and retrieval functions
- Basic database CRUD operations for password entries

Month 4: October 2025 - User Interface & Basic Features

- Main application window with login interface
- Credential management interface (add/edit/delete passwords)
- Basic search functionality implementation
- Auto-lock timeout mechanism
- Password strength analyzer and duplicate detection
- Local storage management and backup functionality
- First demo preparation and testing

Phase 3: Advanced Features & Integration (November - December 2025)

Month 5: November 2025 - Enhanced Security & Cross-Platform

- Cross platform - linux
- Two-factor authentication (TOTP) integration

Month 6: December 2025 - Integration & External Services

- Browser extension
- Autofill functionality
- "Have I Been Pwned" API integration for breach detection
- Password health analysis and reporting

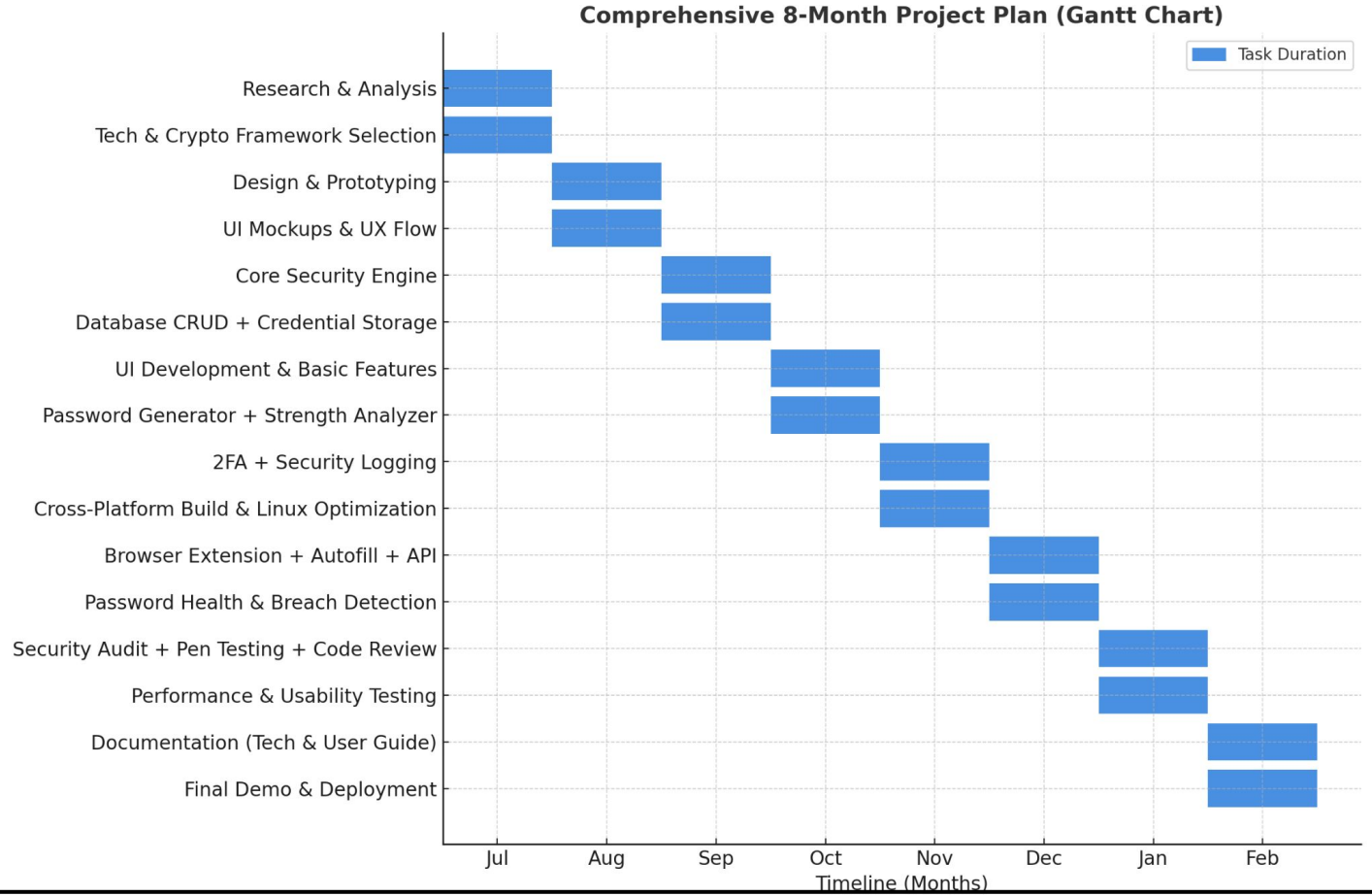
Phase 4: Testing, Documentation & Finalization (January - February 2026)

Month 7: January 2026 - Comprehensive Testing & Security Audit

- Code review
- Performance testing
- Usability testing
- Performance optimization and memory leak fixes

Month 8: February 2026 - Documentation & Final Delivery

- Technical documentation
- User manual with installation and usage guides
- Deployment scripts and installation packages
- Final demo preparation for project delivery


Comprehensive 8-Month Project Plan (Gantt Chart)
Task Duration
Research & Analysis
Tech & Crypto Framework Selection
Design & Prototyping
UI Mockups & UX Flow
Core Security Engine
Database CRUD + Credential Storage
UI Development & Basic Features
Password Generator + Strength Analyzer
2FA + Security Logging
Cross-Platform Build & Linux Optimization
Browser Extension + Autofill + API
Password Health & Breach Detection
Security Audit + Pen Testing + Code Review
Performance & Usability Testing
Documentation (Tech & User Guide)
Final Demo & Deployment
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Timeline (Months)

Individual work load (last 4 weeks)

Member	Task	Progress
Adithya N S	Working of KeePassXC	50%
Anaswara Suresh M K	Working of KeePassXC	50%
C S Amritha	Working of BitWarden	40%
R Sruthi	Working of KeePass	50%
All	• Deciding algorithms, libraries and tools • Literature survey • Deciding features and basic flow	60%

Work for next 3 weeks

Member	**Tasks**
All	• Find potential security problems using STRIDE method • Finalise programming languages and tools • Finalise encryption methods • Design database structure Plan master password system
Anaswara, Adithya	Write encryption rules and process
Amritha	Threat modeling
R Sruthi	Create app UI design

Work for next 3 weeks

Member	**Tasks**
	Create login and password checking
	Build database connection
	Set up testing and documentation
	Start UI

References

- https://github.com/bitwarden/desktop
- https://github.com/bitwarden/clients
- https://github.com/keepassxreboot/keepassxc
- https://scythe-studio.com/en/blog/4-best-frameworks-for-cross-platform-desktop-app-development
- https://en.wikipedia.org/wiki/STRIDE_model
- https://medium.com/@pravallikayakkala123/understanding-aes-encryption-and-aes-gcm-mode-an-in-depth-exploration-using-java-e03be85a3faa
- https://github.com/alexedwards/argon2id