Project Proposal: Secure Password Manager

1. Project Title: Secure Password Manager

2. Team Members: Asad Ali (2022903), Ammaid Saleem(2022344), Mursaleen (2022401),Hamayun Nasir (2022189)

3. Introduction: With the increasing number of online accounts, users struggle to remember multiple passwords. Many resort to weak or reused passwords, making them vulnerable to cyberattacks. This project aims to develop a **Secure Password Manager**, which will allow users to securely store, retrieve, and manage their passwords with strong encryption techniques.

4. Objectives:

- Develop a user-friendly password manager with a secure authentication system.
- Encrypt all stored passwords using **AES-256 encryption**.
- Implement secure user authentication with **PBKDF2/Argon2 password hashing**.
- Provide an auto-fill and password generation feature for strong password recommendations.
- Ensure secure clipboard handling to prevent keylogging attacks.
- Implement role-based access control for multi-user scenarios.

5. Security Requirements & Planning:

- **Secure Authentication:** Users must log in with a master password (hashed with PBKDF2/Argon2).
- **Data Encryption:** Passwords will be encrypted before storage using AES-256.
- **Secure Storage:** Store encrypted passwords in a local SQLite/MySQL database.
- **Auto Logout:** Sessions will expire after inactivity to prevent unauthorized access.
- **Brute Force Prevention:** Implement account lockout after multiple failed login attempts.

6. Threat Modeling & Risk Assessment:

- **Potential Threats:**
    - Brute force attacks on master passwords.

    - Database breaches exposing sensitive credentials.
    - Phishing attacks targeting users.
    - Keyloggers capturing entered passwords.
- **Risk Mitigation Strategies:**
    - Enforce strong master passwords with complexity rules.
    - Use AES-256 encryption for data security.
    - Implement 2FA (Two-Factor Authentication) for additional security.
    - Secure clipboard handling to prevent password theft.

7. System Architecture & Secure Design:

- **Frontend:**
    - Desktop App: **Python (Tkinter/PyQt)**
    - Web App: **HTML, CSS, JavaScript (React.js optional)**
- **Backend:**
    - **Python (Flask/Django) or Node.js (Express.js)**
- **Database:**
    - **SQLite/MySQL with encrypted password storage**

8. Secure Coding & Implementation:

- Implement **input validation** to prevent SQL Injection and XSS.
- Use **JWT-based authentication** for web applications.
- Ensure **secure API endpoints** for password retrieval and storage.

9. Security Testing & Vulnerability Analysis:

- **Perform penetration testing** using OWASP ZAP and Burp Suite.
- **Conduct static code analysis** using SonarQube.
- **Implement fuzz testing** to check input validation security.

10. Final Implementation & Secure Code Review:

- Review **encryption methods and access control measures**.

- Conduct **peer code reviews** to ensure secure coding practices.
- Optimize system for **performance and scalability**.

11. Final Report & Presentation:

- Include **threat model, security features, testing results**.
- Submit source code with proper documentation.
- Provide a **live demo showcasing security features**.

12. Expected Outcome:

- A fully functional **Secure Password Manager** that securely stores and manages user credentials.
- **Improved security practices** for users by enforcing strong passwords and encryption.
- A **user-friendly interface** with secure authentication and password management features.

Conclusion: This project will provide a **practical and real-world solution** to the problem of password management while incorporating **secure software development principles**. It will help users manage their credentials securely, reducing the risk of password-related cyber threats.