Industrial Internship Report on

”Password Manager ”

Prepared by

Palavalasa Kusuma

Executive Summary
This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT). This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks’ time. My project was Password Manager. This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship.

TABLE OF CONTENTS

1 Preface

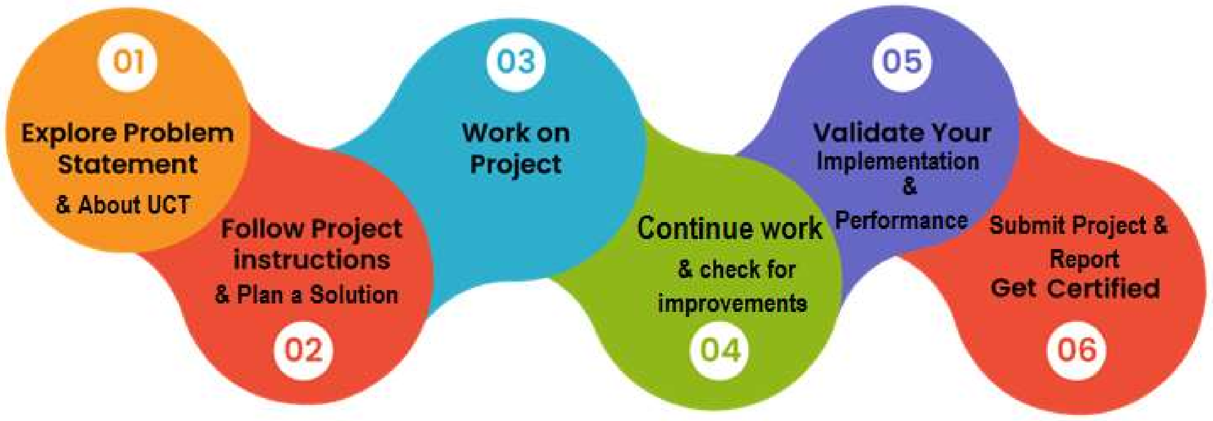
Summary of the whole 6 weeks' work.

About need of relevant Internship in career development.

Brief about Your project/problem statement.

Opportunity given by USC/UCT.

How Program was planned



Your Learnings and overall experience.

Thank to all mentors, who have helped you directly or indirectly.

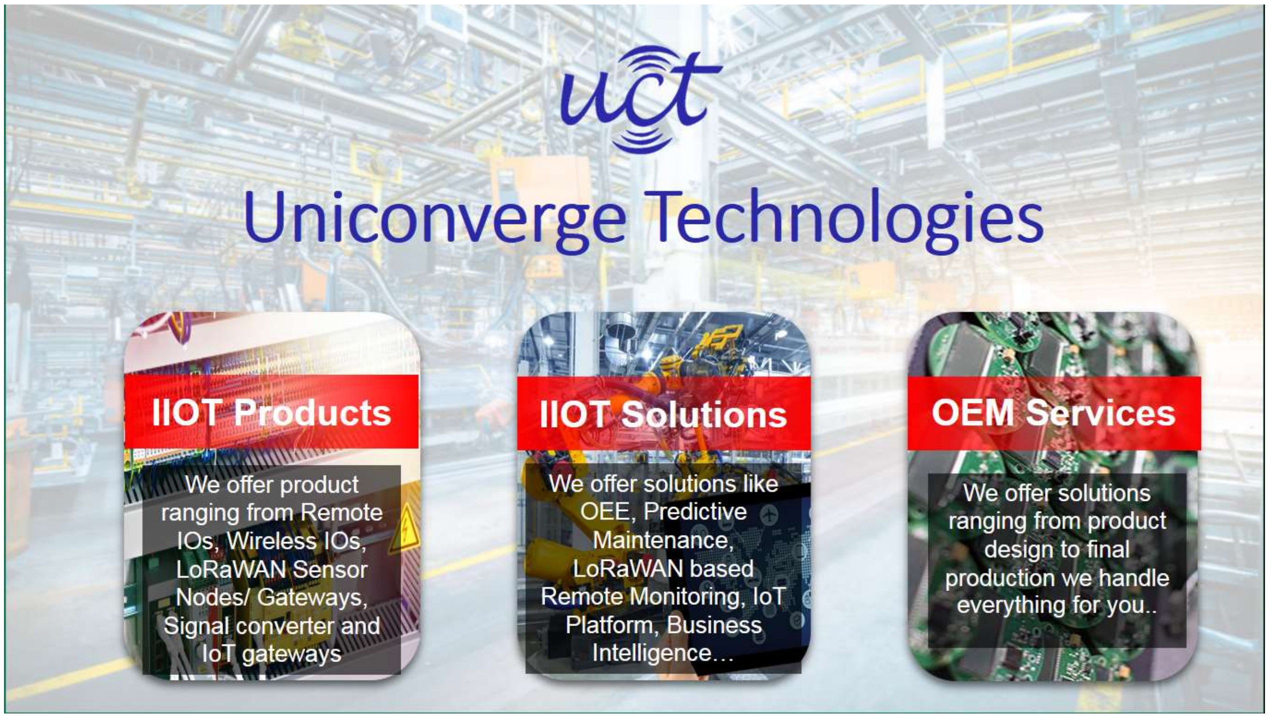
Your message to your juniors and peers.

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.



uct

Uniconverge Technologies

IIOT Products

We offer product ranging from Remote IOs, Wireless IOs, LoRaWAN Sensor Nodes/ Gateways, Signal converter and IoT gateways

IIOT Solutions

We offer solutions like OEE, Predictive Maintenance, LoRaWAN based Remote Monitoring, IoT Platform, Business Intelligence…

OEM Services

We offer solutions ranging from product design to final production we handle everything for you..

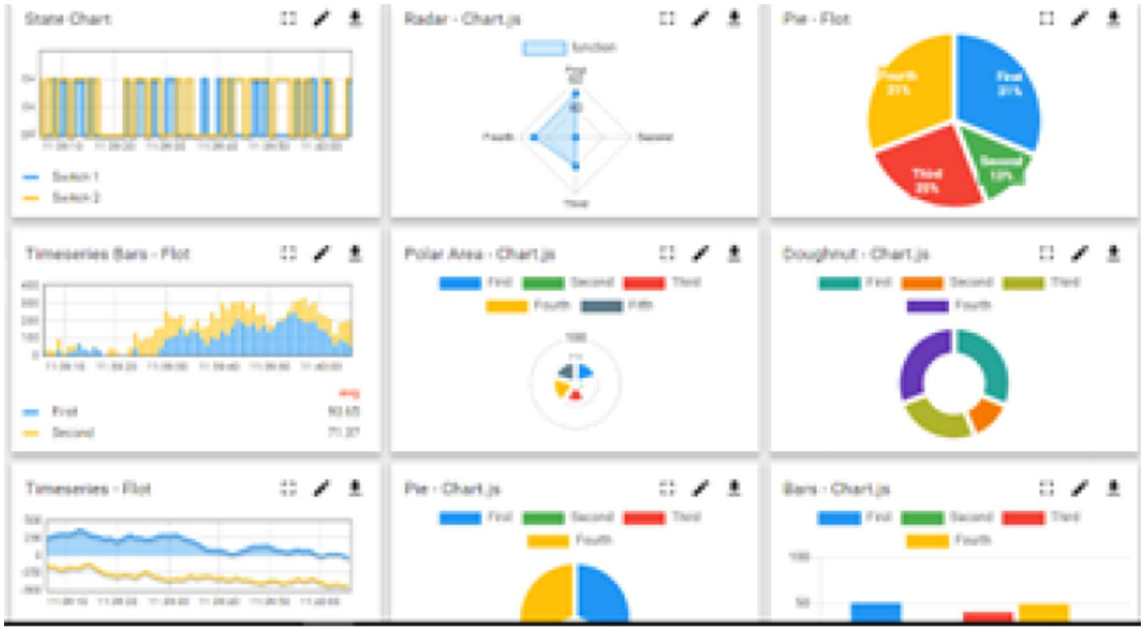
i. UCT IoT Platform (uct Insight)

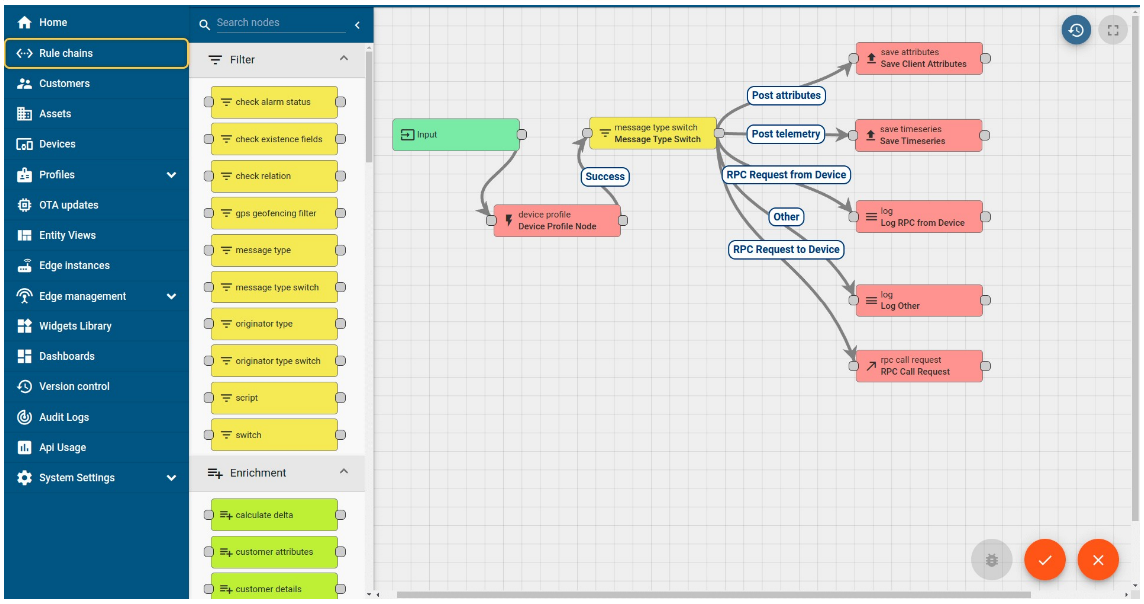
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine





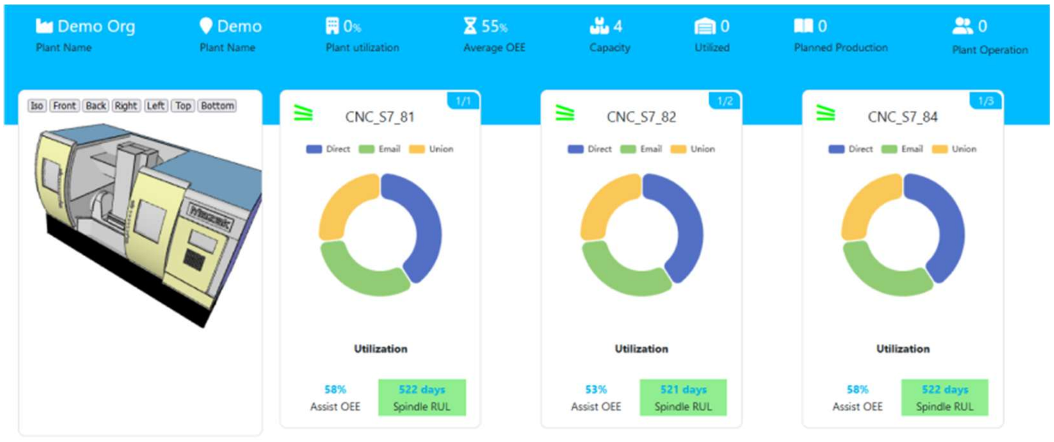
ii. Smart Factory Platform (FACTORY WATCH)

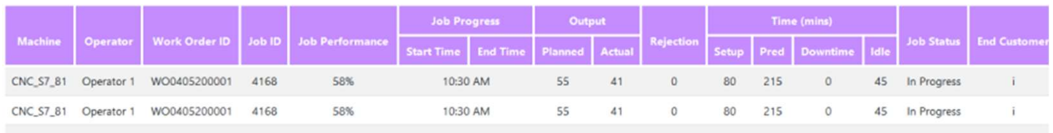
Factory watch is a platform for smart factory needs.

It provides Users/ Factory

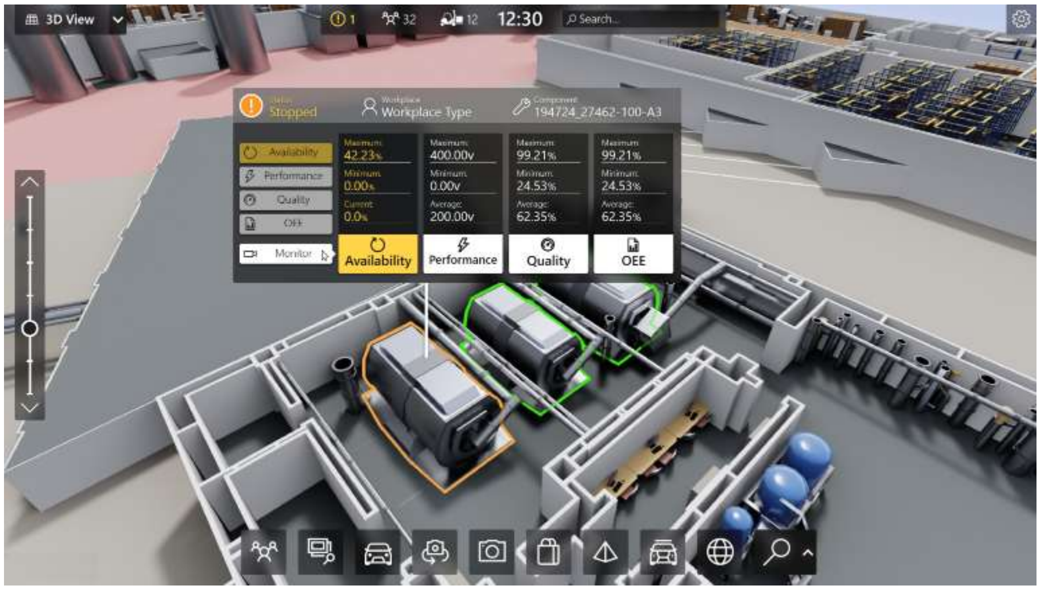
- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.





Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

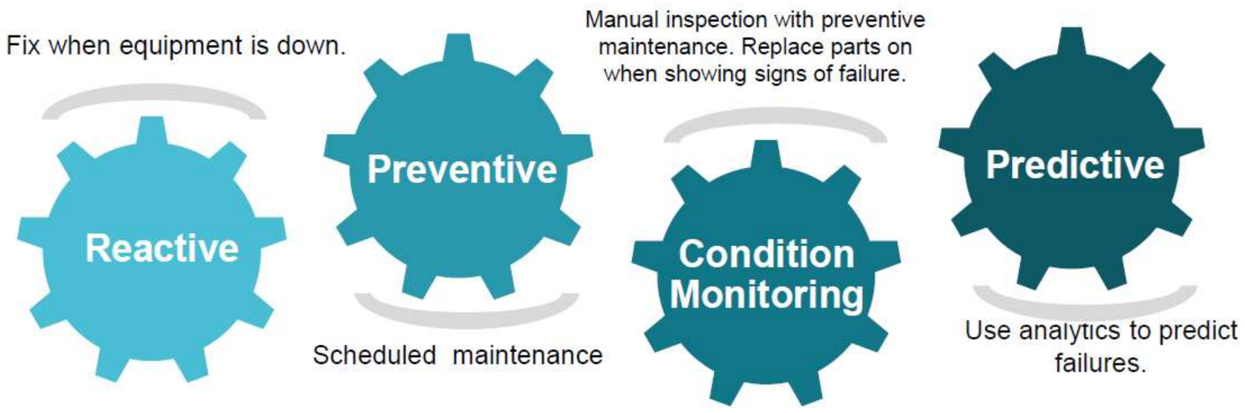


iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

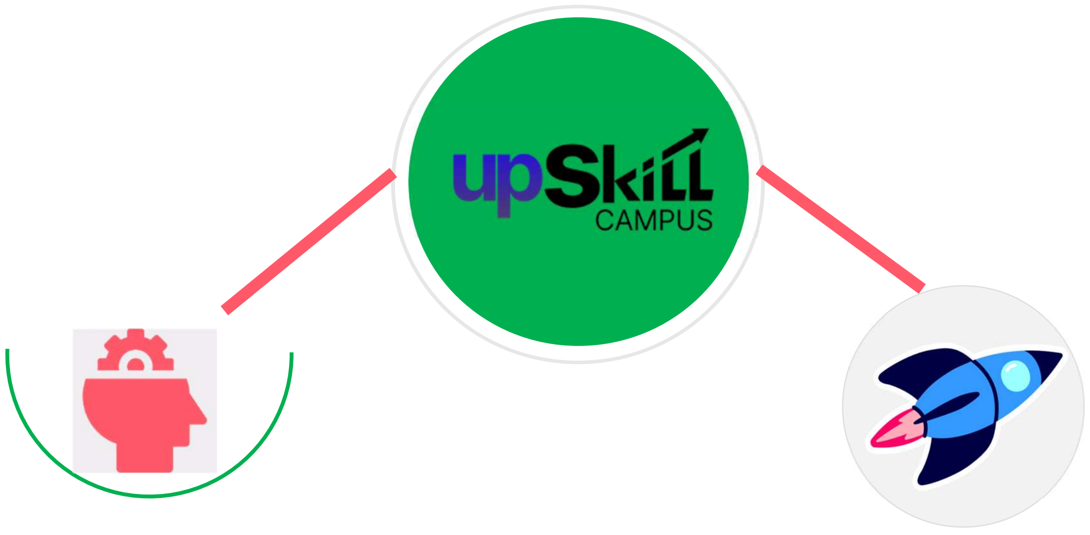
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.


Fix when equipment is down.
Reactive
Preventive
Scheduled maintenance
Manual inspection with preventive maintenance. Replace parts on when showing signs of failure.
Condition Monitoring
Predictive
Use analytics to predict failures.

2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

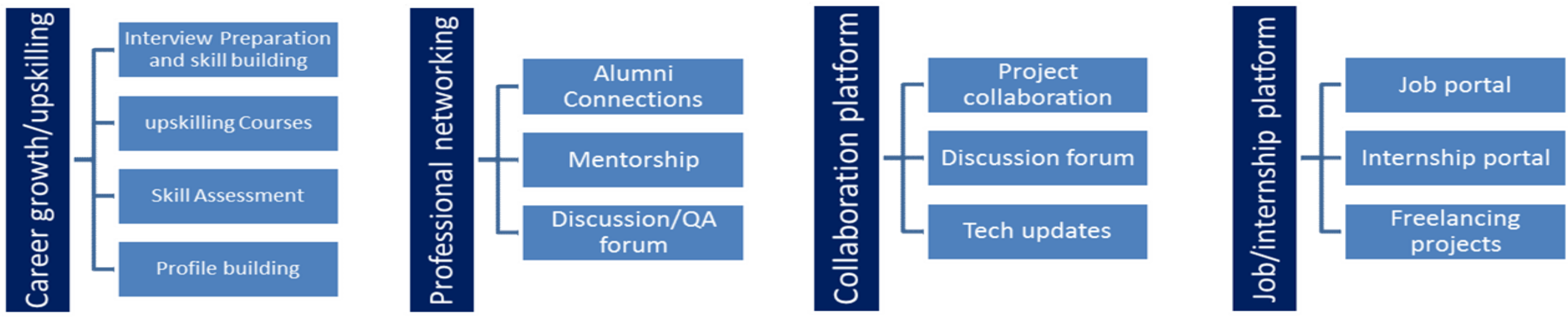
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



Seeing need of upskilling in self paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

☛ get practical experience of working in the industry.

☛ to solve real world problems.

☛ to have improved job prospects.

☛ to have Improved understanding of our field and its applications.

☛ to have Personal growth like better communication and problem solving.

2.5 Reference

1. SQLite3 Python Documentation: Official guide and reference for using the SQLite database module in Python, including connection handling and SQL commands.
   https://docs.python.org/3/library/sqlite3.html

2. Cryptography Library (Fernet) Documentation: Detailed explanations of Fernet symmetric encryption, key management, and cryptographic primitives used for secure encryption and decryption.
   https://cryptography.io/en/latest/fernet/

3. PBKDF2HMAC Key Derivation: Understanding the Password-Based Key Derivation Function 2 (PBKDF2) with HMAC and SHA256 hashing for creating secure cryptographic keys from passwords.
   https://cryptography.io/en/latest/hazmat/primitives/key-derivation-functions/#pbkdf2

2.6 Glossary

Terms	Acronym
SQLite	A lightweight, file-based database engine used to store password records (website, username, encrypted password) locally in a single file
cryptography Library	A Python package providing cryptographic recipes and primitives. Used here for secure encryption and decryption of passwords with Fernet symmetric encryption.
PBKDF2HMAC	Password-Based Key Derivation Function 2 with HMAC, a method to securely derive a cryptographic key from the master password using a salt and many iterations to resist brute-force attacks.
Encryption	The process of converting plaintext passwords into encoded ciphertext using the derived key to protect password secrecy in storage.
Decryption	The process of converting the encrypted ciphertext back to plaintext passwords for viewing, using the same cryptographic key.

3 Problem Statement

In the assigned problem statement

With the increasing number of online accounts and passwords that users need to manage, securely storing and retrieving passwords becomes a critical challenge. Users often resort to unsafe practices such as reusing passwords, writing them down, or storing them in unprotected files, which exposes them to security risks like hacking and data breaches.

This project aims to develop a secure and user-friendly password manager application that allows users to safely store, encrypt, and retrieve their passwords for various websites. The application must safeguard password confidentiality by encrypting the stored passwords using strong cryptographic techniques, ensuring that even if the storage database is compromised, the passwords remain protected. The manager should provide an intuitive interface for adding new credentials and viewing existing ones, accessible only through a master password that derives the encryption key.

The solution must:

- Use robust encryption (Fernet symmetric encryption) combined with a secure key derivation function (PBKDF2HMAC) to protect passwords.
- Store encrypted passwords along with associated website and username details in a local SQLite database.
- Offer a command-line interface for easy password addition, retrieval with decryption, and safe exit.
- Maintain security by using a master password and salt in key derivation to resist brute force and dictionary attacks.

4 Existing and Proposed solution

Existing Solutions: Popular password managers like LastPass, 1Password, and KeePass provide strong encryption, multi-device syncing, and user-friendly interfaces. However, they often depend on cloud storage, may have subscription fees, and can be complex for non-technical users.

Proposed Solution: A lightweight, offline Python password manager that securely stores encrypted passwords in a local SQLite database, accessed via a simple command-line interface with a master password and strong key derivation (PBKDF2HMAC).

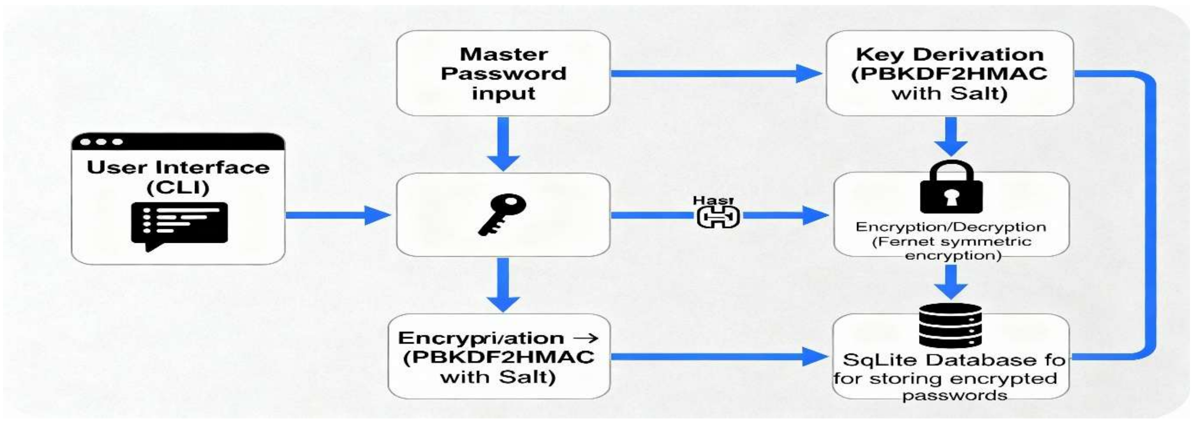
4.1 Code submission (Github link upskillcampus/PasswordManager.py at main · Kusuma-97/upskillcampus

4.2 Report submission (Github link) https://github.com/Kusuma-97/upskillcampus/blob/main/PasswordManager_Kusuma_USC_UCT.pdf

5 Proposed Design/ Model

Given more details about design flow of your solution. This is applicable for all domains. DS/ML Students can cover it after they have their algorithm implementation. There is always a start, intermediate stages and then final outcome.

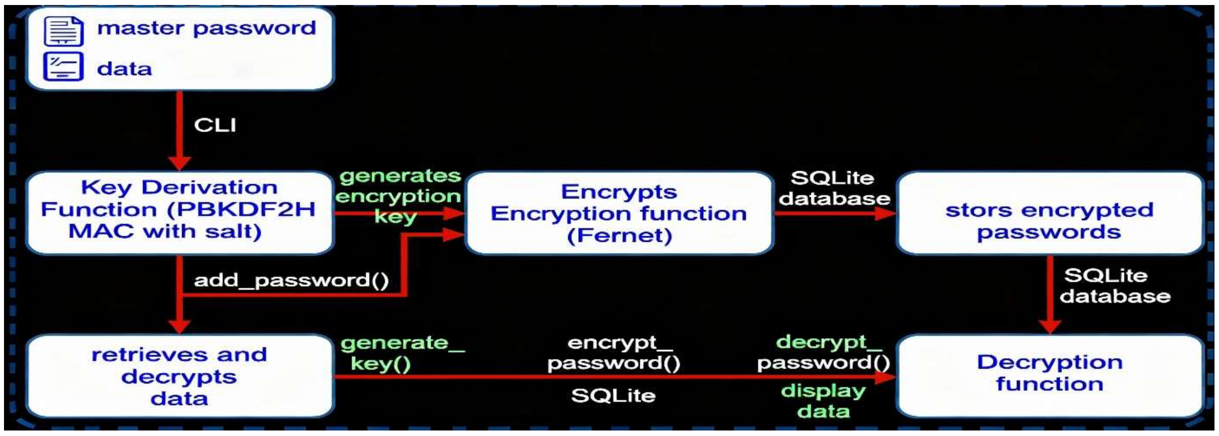
5.1 High Level Diagram (if applicable)



Master Password input
Key Derivation (PBKDF2HMAC with Salt)
User Interface (CLI)
Hash
Encryption/Decryption (Fernet symmetric encryption)
Encrypri/ation → (PBKDF2HMAC with Salt)
SqLite Database fo for storing encrypted passwords

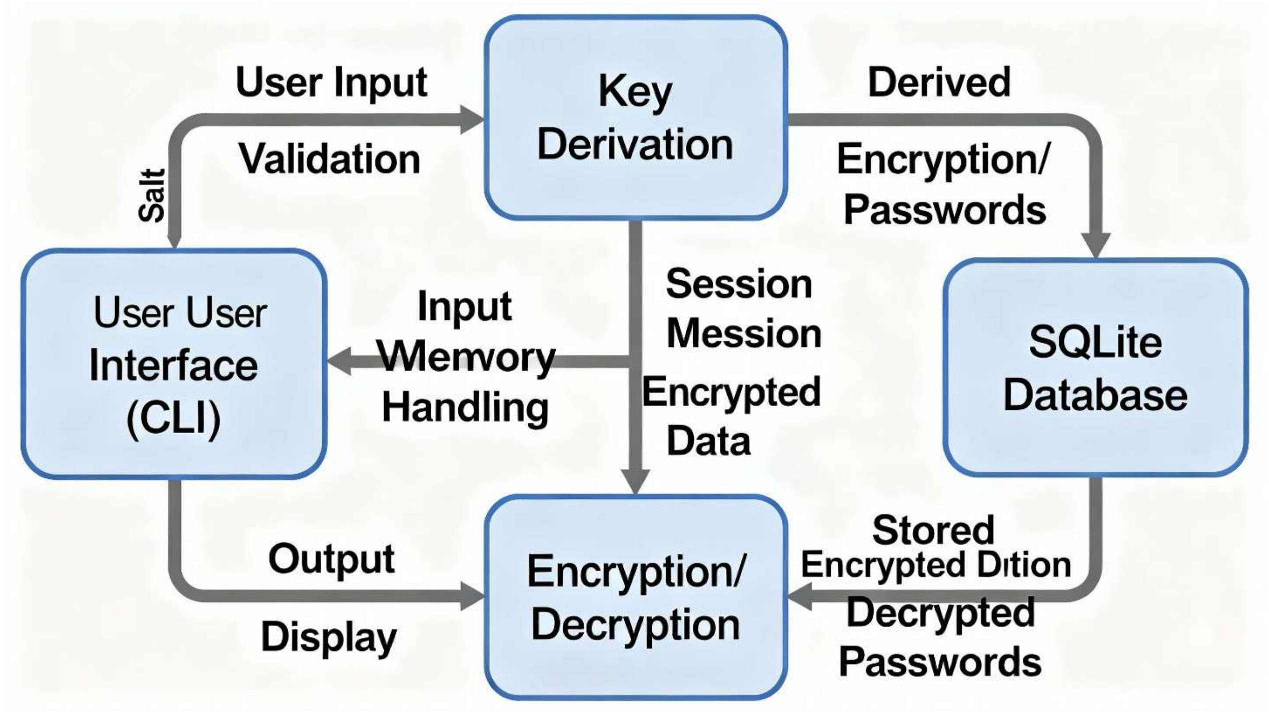
Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

5.2 Low Level Diagram (if applicable)



5.3 Interfaces (if applicable)

Update with Block Diagrams, Data flow, protocols, FLOW Charts, State Machines, Memory Buffer Management.



6 Performance Test

This is very important part and defines why this work is meant of Real industries, instead of being just academic project.

Here we need to first find the constraints.

How those constraints were taken care in your design?

What were test results around those constraints?

Constraints can be e.g. memory, MIPS (speed, operations per second), accuracy, durability, power consumption etc.

In case you could not test them, but still you should mention how identified constraints can impact your design, and what are recommendations to handle them.

6.1 Test Plan/ Test Cases

- Master Password Verification: Test with correct and incorrect master passwords to verify access control.
- Add Password Test: Input valid website, username, and password; verify successful encryption and storage.
- View Passwords Test: Retrieve stored passwords and verify correct decryption and display.
- Input Validation: Test empty and invalid inputs for website, username, and password fields to ensure proper handling.
- Encryption Integrity: Confirm that stored passwords are encrypted, not stored in plaintext.
- Database Operations: Test database table creation, insertion, retrieval, and error handling for missing/duplicate entries.
-

6.2 Test Procedure

- Launch the application and enter the master password.
- Navigate the menu to add new password entries; verify confirmation messages.

- Select view option to display all stored passwords ensuring they match original inputs.
- Attempt invalid inputs such as empty strings and invalid characters and check error messages.
- Verify the persistence of encrypted data by inspecting the SQLite database directly.
- Attempt to start the app with an incorrect master password and confirm restricted access.
-

6.3 Performance Outcome

- Speed: Password encryption, decryption, and database operations complete within milliseconds, providing a responsive user experience.
- Security: Master password-based key derivation with PBKDF2HMAC ensures high resistance to brute-force attacks; passwords stored encrypted using AES-based Fernet.
- Reliability: Database operations execute without loss or corruption of data across multiple add/view cycles.
- Usability: Simple CLI interface supports smooth workflow with clear prompts and validations.
- Resource Use: Lightweight database and cryptography modules have minimal memory and CPU consumption suitable for typical desktop environments.

7 My learnings

- Database Integration: Understanding how to use SQLite in Python for persistent data storage, creating tables, and performing CRUD (Create, Read, Update, Delete) operations securely.
- Cryptography Fundamentals: Gaining practical experience with encryption and decryption using the cryptography library, including symmetric encryption (Fernet) and cryptographic key derivation methods (PBKDF2HMAC).
- Security Best Practices: Learning the importance of protecting sensitive data with secure key generation, use of salts, and proper encryption standards to mitigate risks from data breaches.
- User Authentication Concepts: Implementing a master password mechanism that controls access to all stored credentials and learning how cryptographic keys can be derived from passwords securely.
- Command-Line Interface (CLI) Design: Building a simple interactive menu-driven CLI application for user interactions, which reinforces input/output handling and program control flow.
- Data Encoding: Using base64 encoding to safely handle binary data (encryption keys and tokens) in string format suitable for storage and transmission.

8 Future work scope

- Enhanced Security Features:
    - Implement multi-factor authentication (MFA) for accessing the password manager.
    - Use unique random salts per user and securely store them alongside the encrypted data.
    - Integrate more advanced encryption algorithms or hardware security modules (HSM) for key management.
    - Include password strength analysis and warnings to encourage strong password creation.
- User Interface Improvements:
    - Develop a graphical user interface (GUI) using frameworks like Tkinter, PyQt, or web-based UI for easier interaction.
    - Add features such as password history, password expiry alerts, and audit logs.
- Extended Storage Options:
    - Support cloud synchronization securely (e.g., integration with services like Firebase, AWS, or encrypted cloud storage).
    - Implement databases beyond SQLite (e.g., PostgreSQL, encrypted NoSQL) with remote access options.
- Advanced Functionalities:
    - Incorporate password generation with customizable complexity rules.
    - Enable secure sharing of passwords with trusted contacts.
    - Add import/export options with encryption for data portability.
    - Provide browser extension or API integration for autofill capabilities.

- Cross-Platform and Collaboration:
    - Create mobile app versions (Android/iOS) to access passwords on the go.
    - Enable multi-user support with role-based access control in organizational settings.
- Automated Backup and Recovery:
    - Implement secure automated backups and recovery options to prevent data loss.