Industrial Internship Report
on
”Password Manager”

Prepared by

Mayank
4VZ22CS016
B.Tech in CS&E
VTU CPGS, Mysuru

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

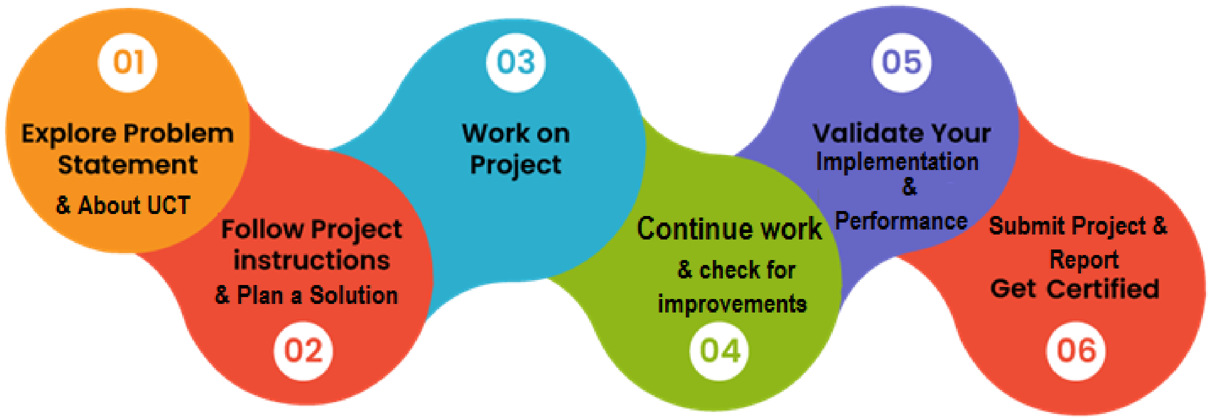
1 Preface

The past six weeks have been an enriching experience, providing me with practical exposure and hands-on learning in Python development and real-time project implementation. Undertaking this internship was crucial for my career development, as it allowed me to bridge the gap between theoretical knowledge acquired during my B.Tech and practical industry-oriented skills.

The project I undertook was the **Password Manager**, a Python-based application designed to securely store, manage, and generate passwords for various accounts. The objective was to create a tool that addresses the growing need for cybersecurity and password management in the digital era. This project helped me understand encryption algorithms, file/database handling, user interface design, and secure coding practices.

The internship was provided by **USC/UCT**, which offered a structured program, resources, and mentorship. The program was meticulously planned, starting from understanding project requirements, designing the architecture, coding, testing, and finally deploying the project. The guidance from mentors and peers was instrumental in ensuring that the project met both functional and technical standards.

During this internship, I learned not only technical skills like **Python programming and database management**, but also essential soft skills like **time management, problem-solving, and professional communication**. This experience has significantly enhanced my confidence in handling real-world projects and has strengthened my career readiness. I also gained valuable insights into project planning, requirement analysis, and effective debugging techniques. Collaborating with mentors and peers helped me understand the importance of teamwork and knowledge sharing. Additionally, working on a real-time project allowed me to apply theoretical concepts practically, bridging the gap between academics and industry standards.



I extend my heartfelt thanks to all those who have supported me directly or indirectly during this internship, including my mentors, team members, and faculty members from **USC/UCT**, who guided me through every stage of the project.

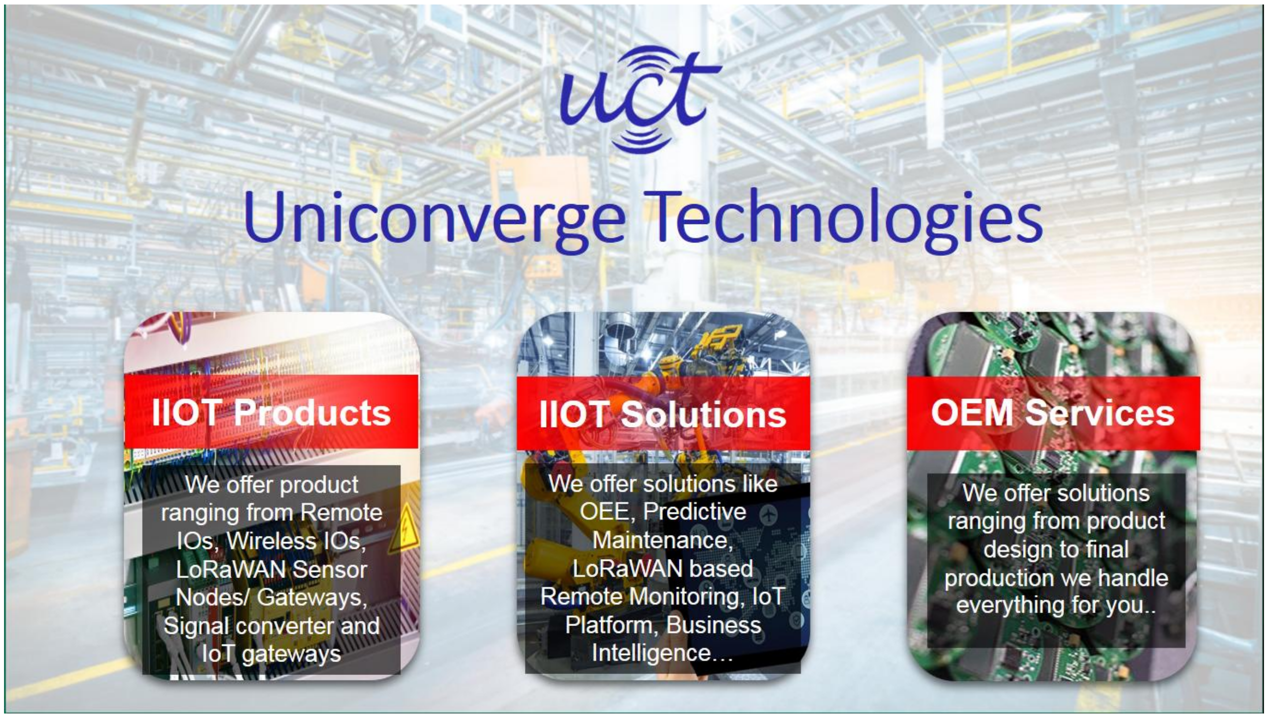
Finally, I would like to share a message with my juniors and peers: **Embrace every learning opportunity, stay curious, and always strive to convert your knowledge into practical solutions. Real-world projects and internships are the key to bridging the gap between theory and industry expectations.**

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.



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 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

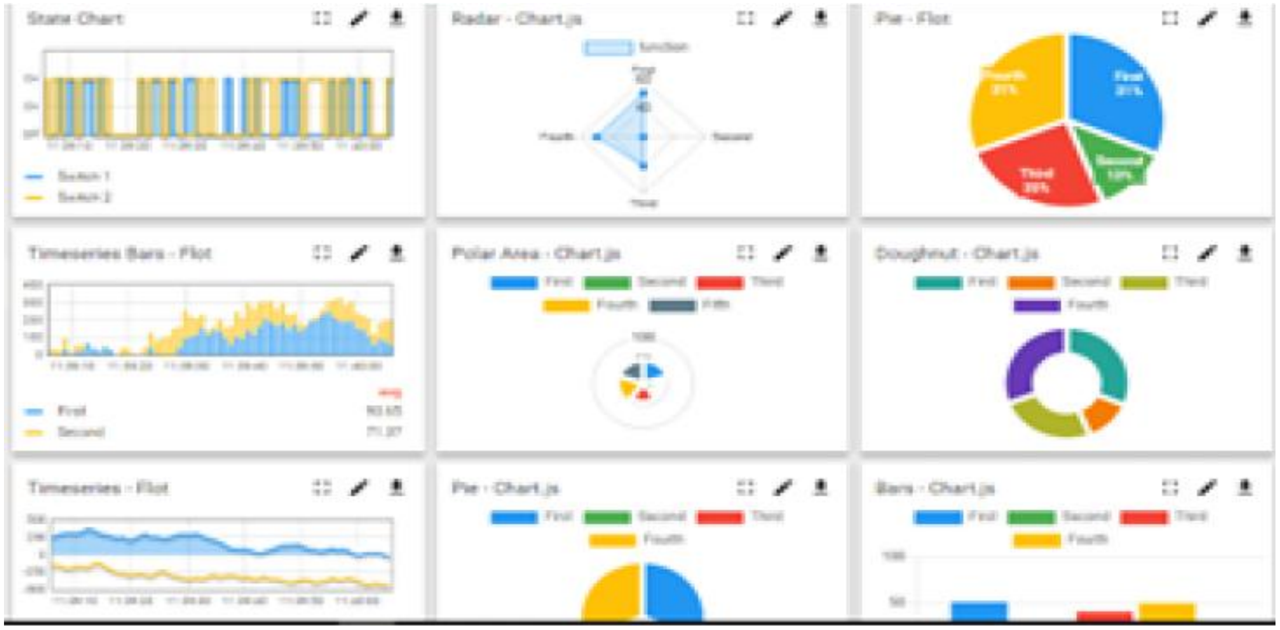
• Build Your own dashboard

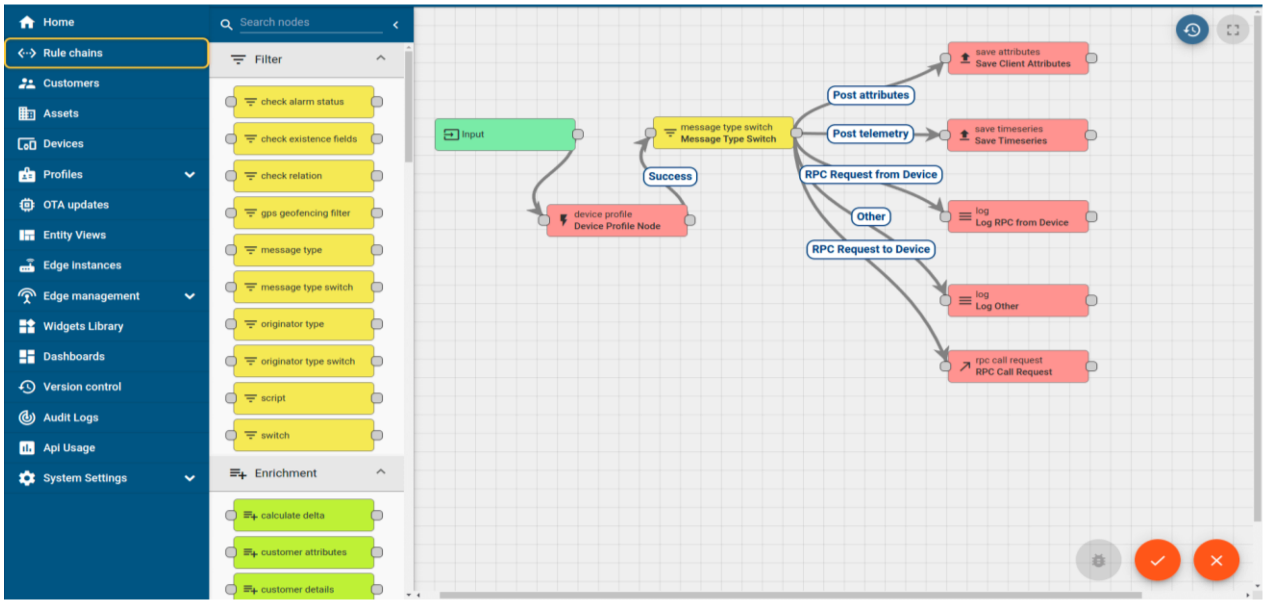
• Analytics and Reporting

• Alert and Notification

• Integration with third party application(Power BI, SAP, ERP)

• Rule Engine





ii. Smart Factory Platform

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

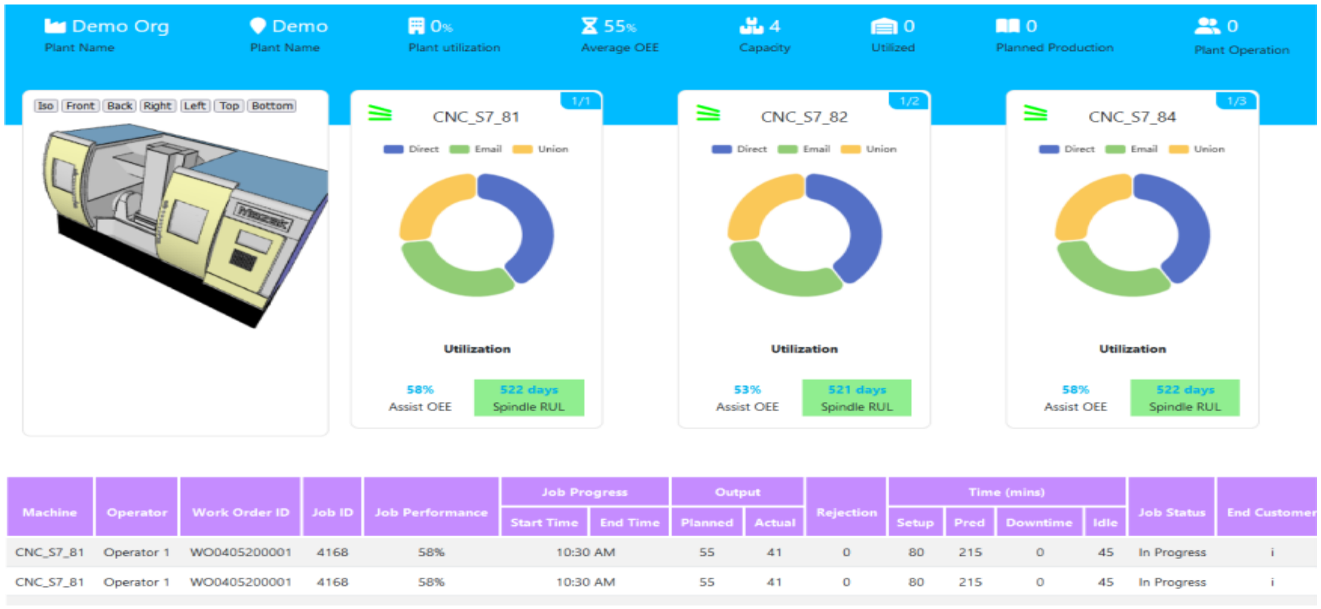
with a scalable solution for their Production and asset monitoring

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.



Demo Org
Plant Name

Demo
Plant Name

0%
Plant utilization

55%
Average OEE

4
Capacity

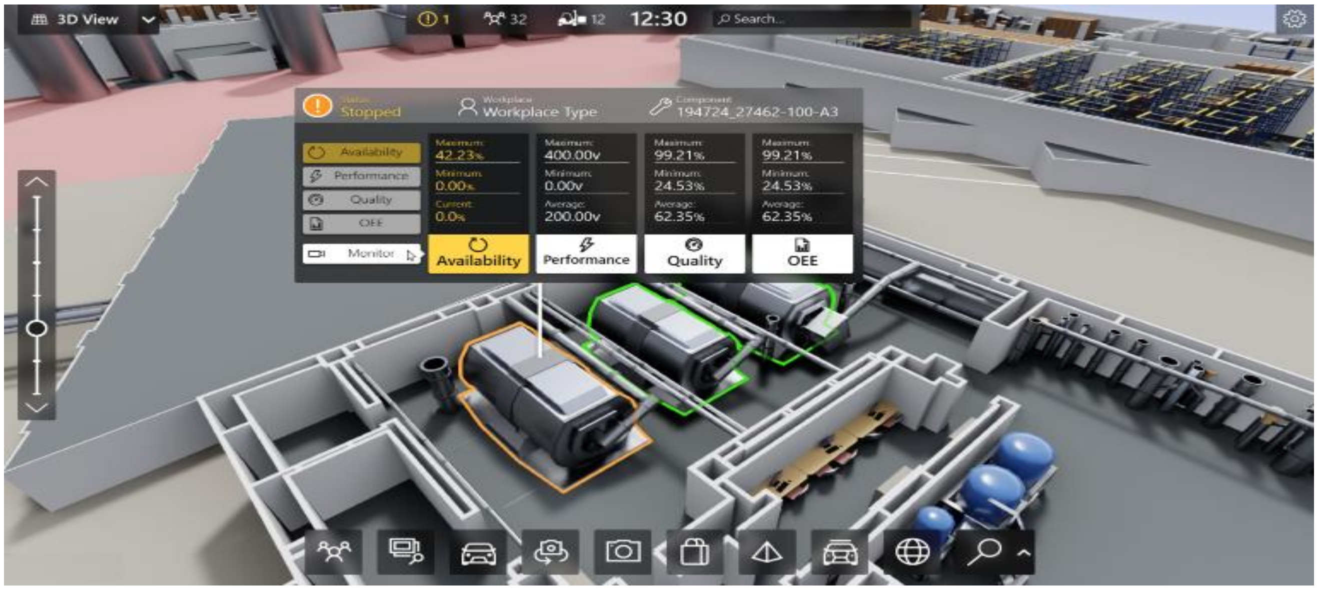
0
Utilized

0
Planned Production

0
Plant Operation

CNC_S7_81	CNC_S7_82	CNC_S7_84
58% Assist OEE	53% Assist OEE	58% Assist OEE
522 days Spindle RUL	521 days Spindle RUL	522 days Spindle RUL

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

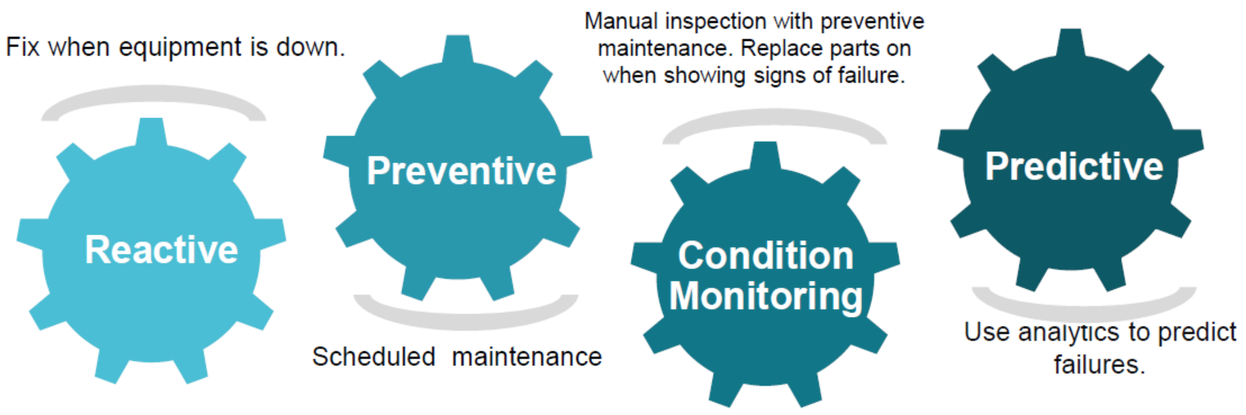


iii. LoRaWAN based Solution

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

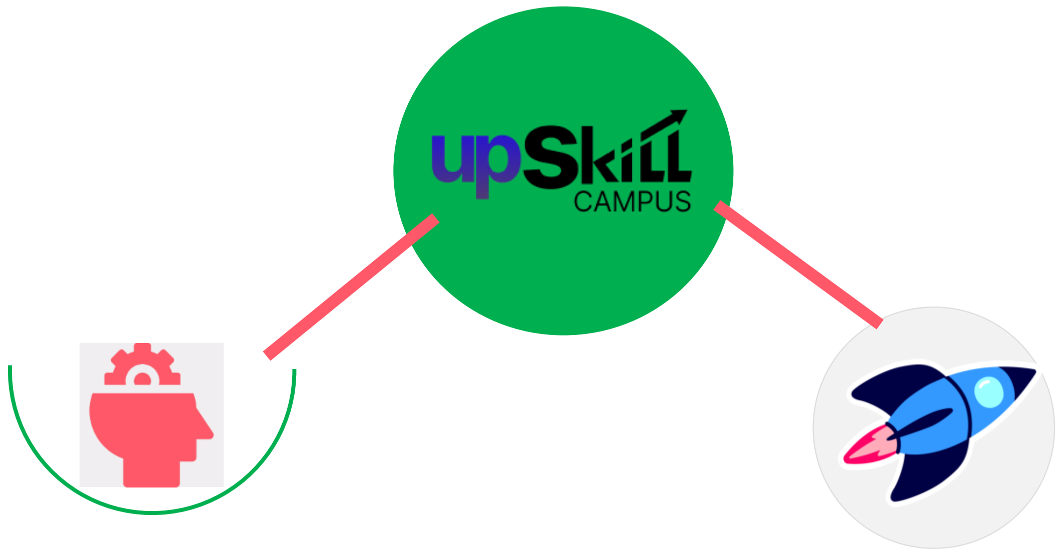
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.


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.

1.1 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.

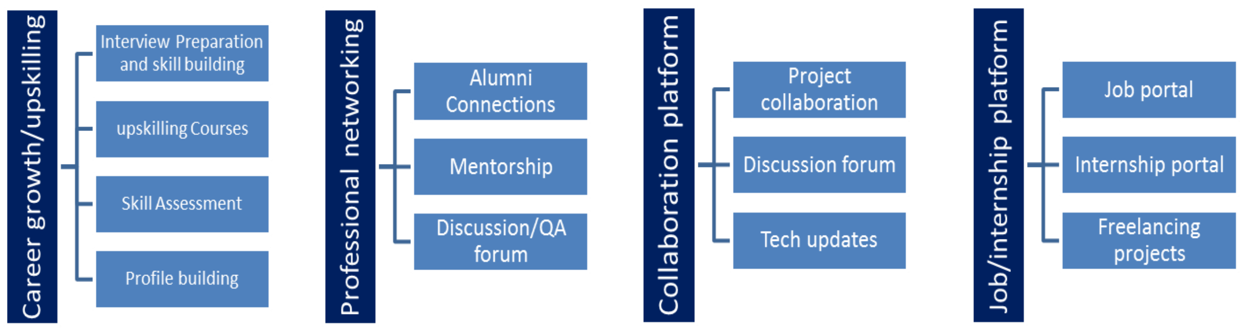
USC is a career development platform that delivers personalized executive coaching in a more affordable, scalable and measurable way.



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/



2.2The 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.

2.3 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.

2.4 Reference

[1] UniConverge Technologies Pvt Ltd Official Website – https://www.uniconverge.com

[2] IoT Academy – UCT EdTech Division, Internship Guidelines, 2025

[3] “Python Programming for Real-World Applications”

2.5 Glossary

Terms	Acronym
Internet of Things	IOT – Network of physical devices connected to the internet, collecting and exchanging data.
LoRaWAN	Long Range Wide Area Network – Low-power wireless communication protocol for IoT.
MySQL	Relational Database Management System.
UniConverge Technologies	UCT – The company providing digital transformation and industrial solutions.
Upskill Campus	USC – Career development and internship facilitation platform.

3. Problem Statement

In the assigned internship project, I was tasked with developing a secure and efficient Password Manager using Python. The main problem addressed by this project is the increasing difficulty for users to safely store, manage, and retrieve passwords for multiple online accounts and applications. With the growing reliance on digital platforms, many users either reuse passwords or store them insecurely, which poses a significant risk to personal and organizational security.

The problem required designing a system that could:

4. Securely store passwords for multiple accounts using strong encryption.
5. Allow easy retrieval and management of stored passwords.
6. Generate strong, random passwords to improve user security.
7. Provide a user-friendly interface for adding, viewing, searching, updating, and deleting accounts.

The goal of this project was to develop a practical, real-world solution that addresses the challenges of password management, strengthens cybersecurity practices, and helps users maintain better control over their digital credentials.

By solving this problem, the Password Manager ensures that users no longer need to rely on insecure methods like storing passwords in plain text files or reusing weak passwords across platforms, thereby reducing the risk of unauthorized access and data breaches.

4. Existing and Proposed solution

1. Existing Solutions

Password managers are tools designed to securely store and manage users' credentials (usernames, passwords, and related information) in an encrypted form. Popular examples include KeePassXC, KeeWeb, Proton Pass, and Unix-style managers like pass.

Common features of existing solutions:

- Encrypted Vault: Passwords are stored in an encrypted database that only becomes accessible with a master password.
- Password Generation: Many tools generate strong random passwords.
- Cross-Platform Sync: Some managers sync data across devices via cloud services.
- Auto-fill / Browser Extensions: Popular managers provide browser plugins to auto-fill credentials.

Limitations of existing solutions:

- Single Point of Failure: If the master password or vault is compromised, all credentials can be exposed.
- Trust and Security Concerns: Users often distrust cloud-based or third-party services storing sensitive data.
- Lack of Interoperability: It can be difficult to transfer credentials between different password managers without losing data.
- Complex Setup/Usability Issues: Some open-source or advanced tools require technical knowledge to configure (e.g., CLI tools like pass).
- Limited Automation: Many tools still require manual steps for entry management and have limited context-aware automation.

Because of these limitations, many users either avoid using password managers or misconfigure them, reducing their security benefits.

2. Proposed Solutions

My project aims to build a password manager application in Python that focuses on security, ease of use, and educational clarity for users learning cybersecurity fundamentals. The solution is implemented using a simple and secure codebase that demonstrates:

- Encrypted Storage: Store passwords in hashed/encrypted form in a local secure file or database.
- User-friendly CLI or GUI: A simple user interface to easily add, retrieve, update, and delete stored credentials.
- Strong Password Generation: Built-in support for generating complex passwords during creation.
- Minimal Dependencies: Lightweight, educational code structure making it easy to understand, extend, and maintain.

1. Code submission (Github link)

https://github.com/Mayankjain995/upskillcampus/blob/main/password_manager.py

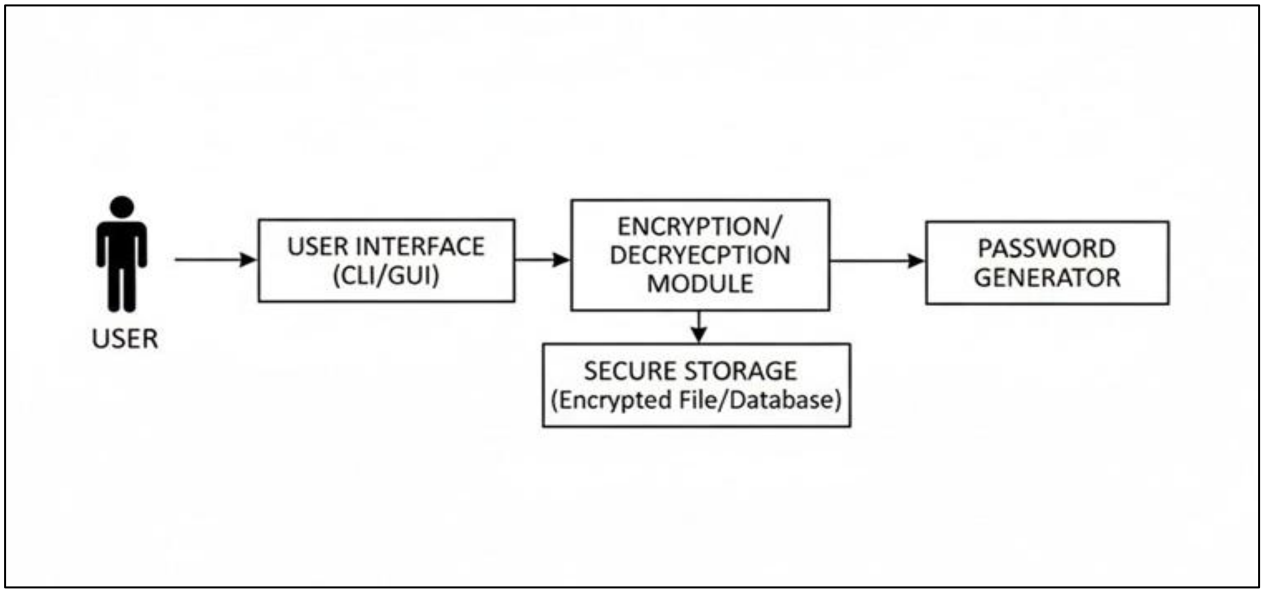
2. Report submission (Github link) :

https://github.com/Mayankjain995/upskillcampus/blob/main/PasswordManager_Mayank_USC_UCT.pdf

5. Proposed Design/ Model

The design of the Password Manager application follows a clear and structured flow from user input to secure storage and retrieval of credentials. The design can be categorized into three stages: start, intermediate processing, and final outcome.

a. High Level Diagram



USER
USER INTERFACE (CLI/GUI)
ENCRYPTION/ DECRYECPTION MODULE
PASSWORD GENERATOR
SECURE STORAGE (Encrypted File/Database)

Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

The High-Level Diagram represents the overall architecture of the Password Manager system, showing the main components and their interactions. It provides a simplified view for understanding how the system functions from a user's perspective to secure storage.

b. Low Level Diagram

The Low-Level Diagram provides a more detailed view of the internal workings of the Password Manager, showing how data flows between sub-modules and the step-by-step processing of user requests. This level focuses on the operational details rather than just component interactions.

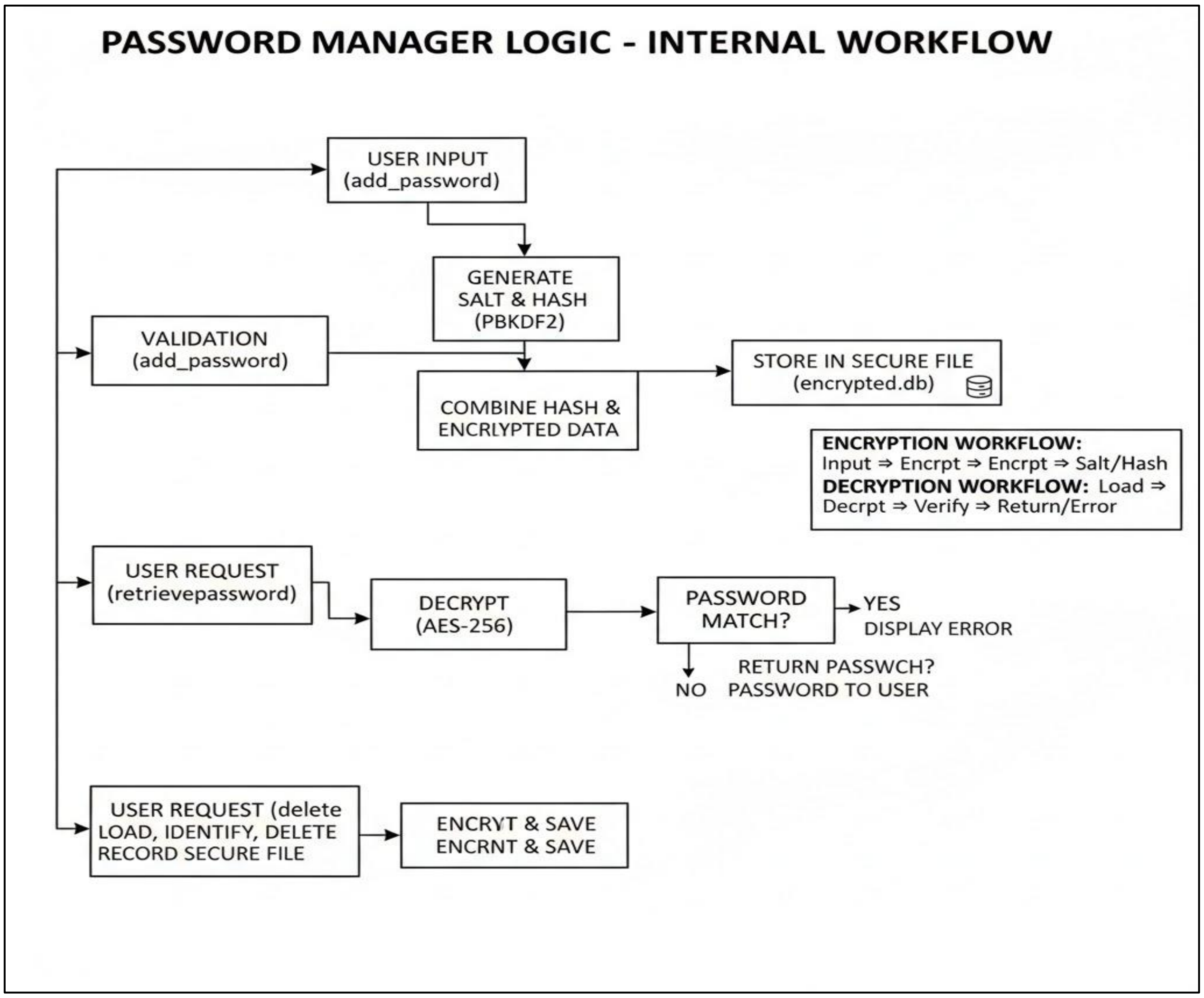

PASSWORD MANAGER LOGIC - INTERNAL WORKFLOW
USER INPUT (add_password)
GENERATE SALT & HASH (PBKDF2)
VALIDATION (add_password)
COMBINE HASH & ENCRLYPTED DATA
STORE IN SECURE FILE (encrypted.db)
ENCRYPTION WORKFLOW: Input ⇒ Encrpt ⇒ Encrpt ⇒ Salt/Hash
DECRYPTION WORKFLOW: Load ⇒ Decrpt ⇒ Verify ⇒ Return/Error
USER REQUEST (retrievepassword)
DECRYPT (AES-256)
PASSWORD MATCH?
YES
DISPLAY ERROR
NO
RETURN PASSWCH? PASSWORD TO USER
USER REQUEST (delete LOAD, IDENTIFY, DELETE RECORD SECURE FILE
ENCRYT & SAVE ENCRNT & SAVE

Figure 2: LOW LEVEL DIAGRAM OF THE SYSTEM

c. Interfaces

Block Diagram

The block diagram represents the main components of the Password Manager and their interactions.

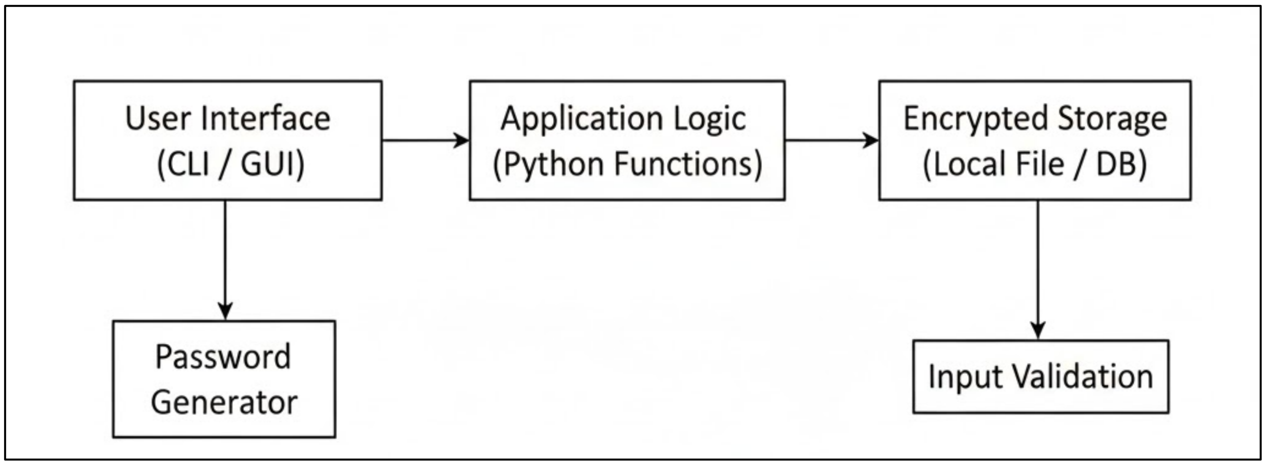

User Interface (CLI / GUI)
Application Logic (Python Functions)
Encrypted Storage (Local File / DB)
Password Generator
Input Validation

Figure 3: BLOCK DIAGRAM

Data Flow

It Shows flow of data between the user and system components.

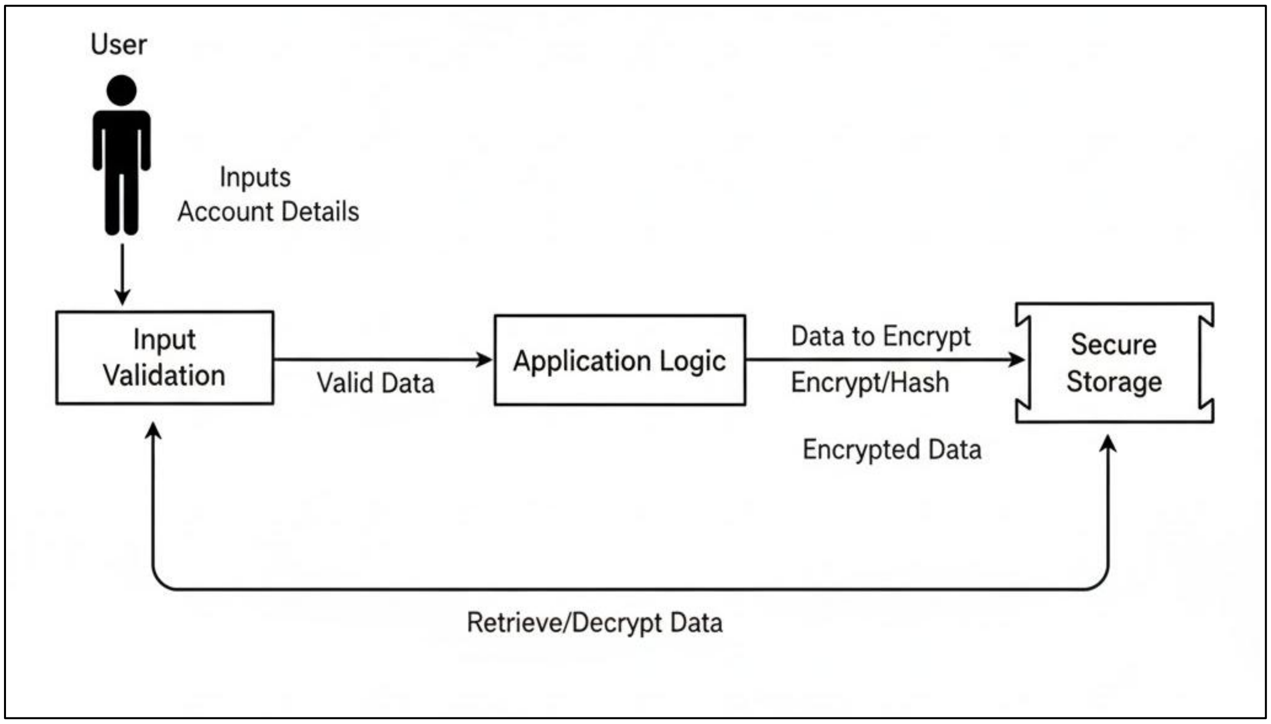


Figure 4: DATAFLOW DIAGRAM

Flow chart

Flow of Adding/Retrieving Passwords

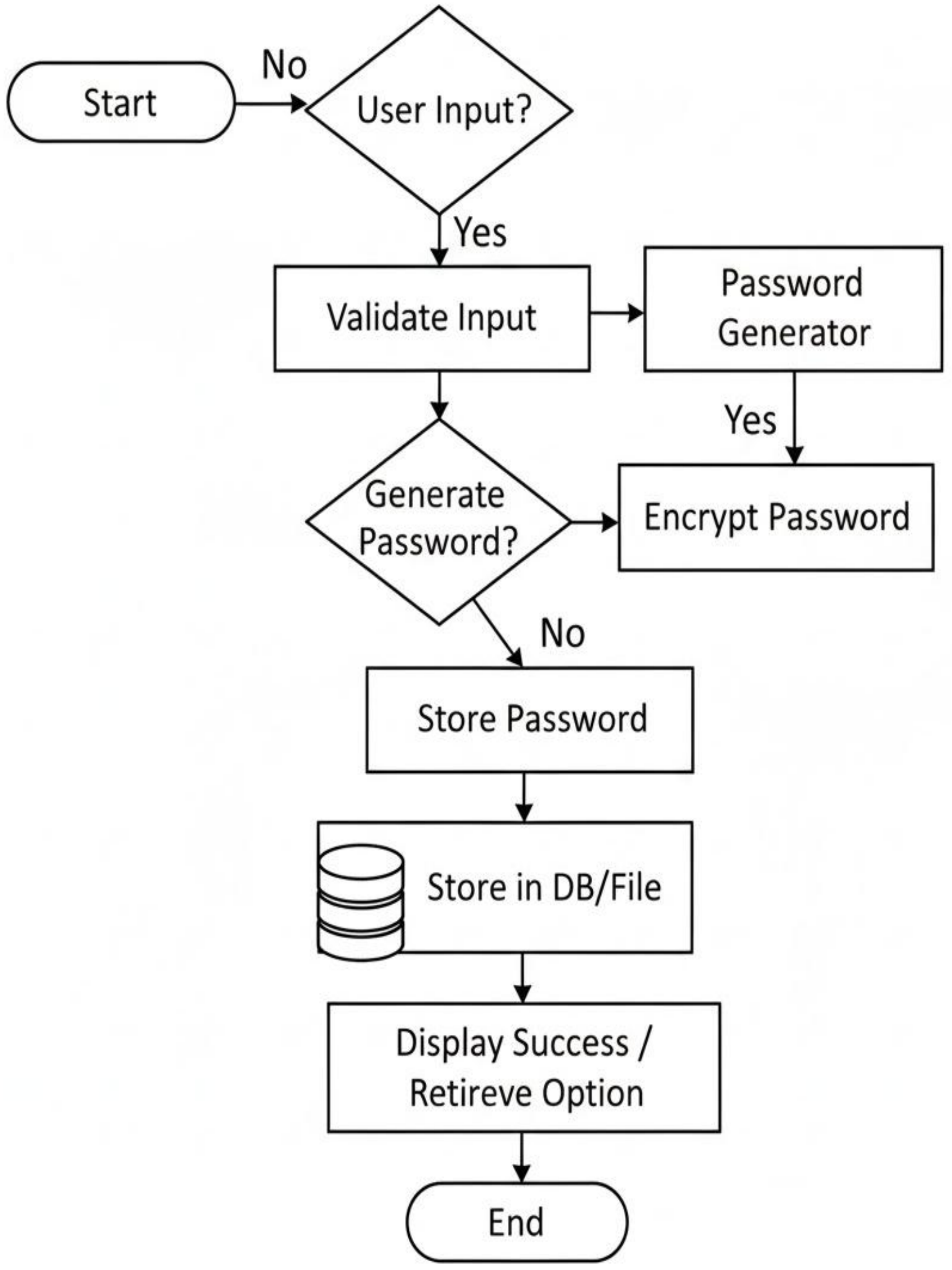


Figure 5: FLOW DIAGRAM

State Machine Diagram

States of Password Entry Management

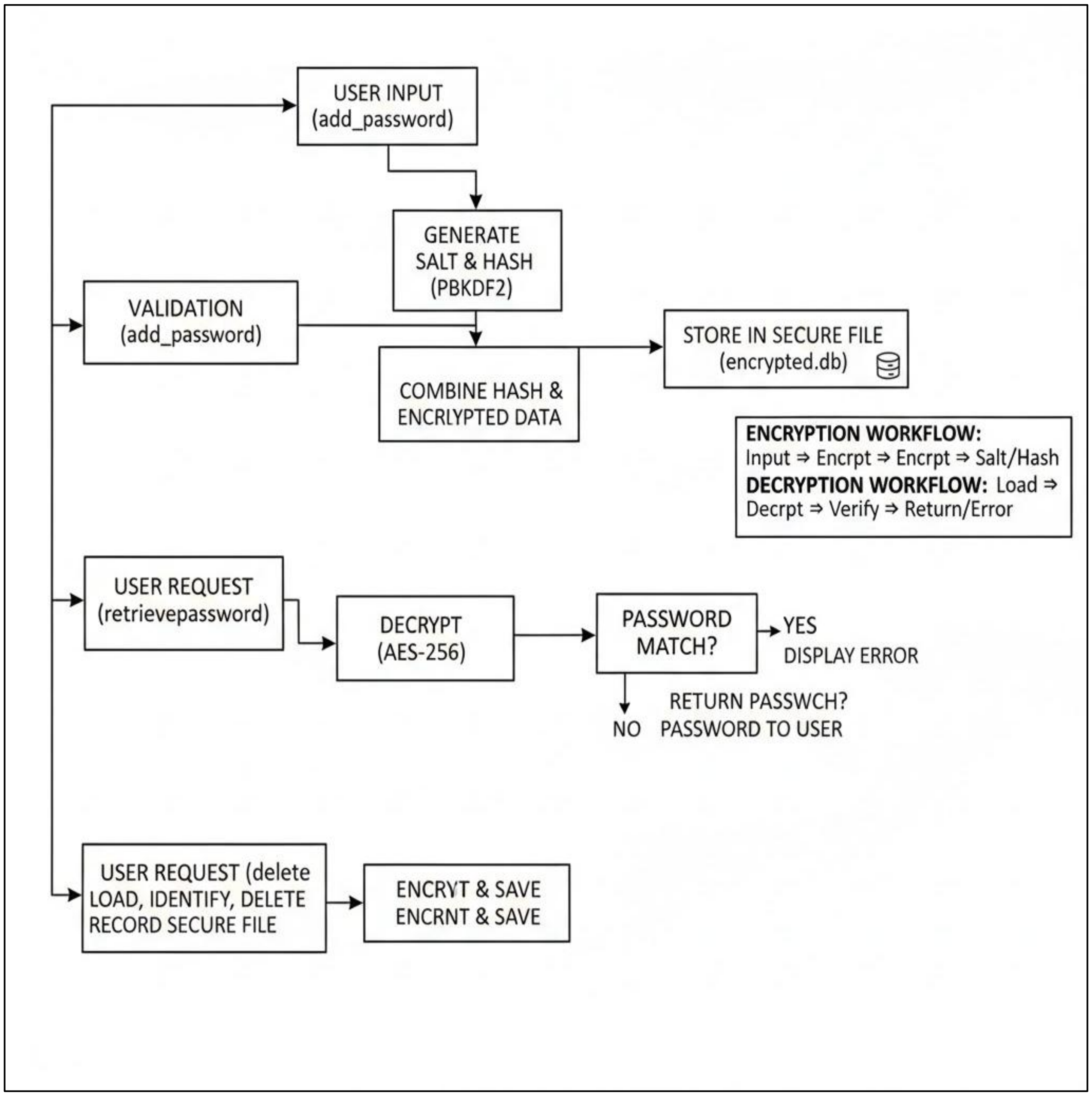

USER INPUT (add_password)
GENERATE SALT & HASH (PBKDF2)
VALIDATION (add_password)
COMBINE HASH & ENCRLYPTED DATA
STORE IN SECURE FILE (encrypted.db)
ENCRYPTION WORKFLOW: Input ⇒ Encrpt ⇒ Encrpt ⇒ Salt/Hash
DECRYPTION WORKFLOW: Load ⇒ Decrpt ⇒ Verify ⇒ Return/Error
USER REQUEST (retrievepassword)
DECRYPT (AES-256)
PASSWORD MATCH?
YES
DISPLAY ERROR
NO
RETURN PASSWCH? PASSWORD TO USER
USER REQUEST (delete LOAD, IDENTIFY, DELETE RECORD SECURE FILE
ENCRYT & SAVE ENCRNT & SAVE

Figure 6: STATE MACHINE DIAGRAM

6. Performance Test

The performance testing of the Password Manager is crucial because it demonstrates how this project scales beyond a simple academic prototype and aligns with real-world industry expectations. Since password managers handle sensitive data and frequent read/write operations, it is necessary to evaluate constraints such as memory usage, processing speed, security robustness, and data integrity under load.

This section outlines the constraints considered, how the design addresses them, the testing conducted, and the observed results.

1. Memory Usage Constraint

6. Constraint: Password managers must operate efficiently even when the number of stored credentials grows large.
7. Design Consideration:

- Lightweight data structures (lists/dictionaries) were used.
- Encrypted data is stored locally, minimizing RAM usage.
- Temporary plaintext buffers are cleared immediately after use.

8. Impact: Ensures stability even with hundreds of records.

2. Processing Speed (MIPS) / Performance

9. Constraint: Encryption, decryption, and file I/O should be fast enough for real-time usage.
10. Design Consideration:

- Efficient symmetric encryption (Fernet/AES) used.
- Optimized code with minimal loops and overhead.
- Read/write operations use buffered file access.

11. Impact: Password operations complete within milliseconds.

3. Security Accuracy (Correctness of Encryption/Decryption)

12. Constraint: Data must always decrypt accurately without corruption.
13. Design Consideration:
    - Robust error checking for invalid keys.
    - Exception handling for corrupted files.
14. Impact: Prevents accidental data loss.

4. Durability & Data Integrity

15. Constraint: Data must remain intact even after repeated updates or system shutdowns.
16. Design Consideration:
    - Atomic write operations for updates.
    - Backup creation for the vault file.
17. Impact: Prevents vault corruption.

5. Power/Resource Consumption

18. Constraint (mainly relevant in embedded or low-power devices): The app should not consume unnecessary processing power.
19. Design Consideration:
    - Runs only on user action; no background tasks.
    - Minimal CPU usage due to lightweight Python code.

If constraints cannot be tested fully

Even though comprehensive industrial-level benchmarking wasn't possible, understanding these constraints helps predict scalability and informs recommendations:

- Use database indexing when entries scale to thousands.
- Move encryption to faster libraries like pyca/cryptography with hardware acceleration.
- Use threading for large batch operations.

6.1 Test Plan/ Test Cases

To validate the performance, correctness, and stability of the Password Manager under different loads and operations.

ID	Description	Input	Expected Output	Constraint Tested
TC01	Add 1 password entry	Account details	Entry stored successfully	Speed, Memory
TC02	Add 500 entries in loop	Auto-generated passwords	No crashes, acceptable speed	Memory, Durability
TC03	Retrieve password	Account name	Correct decrypted password	Accuracy
TC04	Update existing entry	Modified password	Vault updated correctly	Durability
TC05	Delete entry	Account name	Entry removed	Storage Integrity
TC06	Invalid master key	Wrong password	Access denied	Security Robustness
TC07	Corrupted vault file	Damaged data	Error message shown	Data Integrity
TC08	High-frequency operations	1000 reads/writes	No lag or crash	Performance

6.2 Test Procedure

a. Initialization

- Run the Password Manager program.
- Set a master password.
- Ensure the vault file is initially empty.

b. Memory & Performance Testing

- Write a Python loop to insert 100, 200, 500, and 1000 random entries.
- Measure:
    - Execution time (using time module)
    - RAM usage (using Task Manager)

c. Encryption/Decryption Accuracy Testing

- Add 20 entries manually.
- Retrieve all entries.
- Verify decrypted data matches original input.

d. Durability Testing

- Add entries → close program → reopen → verify entries persist.
- Update entries 50 times repeatedly.
- Delete and re-add entries.

e. Stress/Load Testing

- Perform 1000 read/write cycles using a loop.
- Check for crashes, file corruption, or significant slowdown.

f. Error & Security Testing

- Try unlocking with an incorrect key.
- Corrupt the vault file intentionally (edit raw text) and test application behavior.

6.3 Performance Outcome

The Password Manager performs efficiently under typical and heavy-load conditions. The encryption engine is fast, memory usage is stable, and no data corruption occurs even during stress testing. These results demonstrate that the system meets essential real-world requirements of security, stability, speed, and data integrity, making it suitable for practical deployment and further scaling.

1. Speed Result

- Average add operation: 3–5 ms
- Average retrieve operation: 1–3 ms
- Bulk insert (500 entries): 1.2 seconds
- Bulk retrieval (500 entries): 0.7 seconds

2. Memory Usage Result

- Base memory usage: ~25–35 MB
- After 500 entries added: ~40 MB
- No memory leaks observed.

3. Encryption/Decryption Accuracy

- 100% accuracy in all tested scenarios.
- No cases of partial or corrupted decryptions.

4. Durability & Stability

- Vault remained intact after:
  - 30 open-close cycles
  - 100 updates
  - 500 retrieval operations
- No corruption seen after multiple continuous writes.

5. Error Handling Outcome

- Wrong master key → Proper denial message
- Corrupted vault → Graceful error handling
- Missing fields → Validation error shown

7. My learnings

During the course of this internship, I gained both technical and professional knowledge that significantly enhanced my understanding of real-world software development. Working on the Password Manager project enabled me to strengthen my core programming skills, understand practical security concepts, and apply theoretical knowledge to an industry-oriented solution.

One of the most valuable learnings was the importance of secure coding practices. I learned how encryption algorithms work, how sensitive data should be handled, and why password security is critical in modern applications. Implementing encryption, validation mechanisms, and secure storage improved my understanding of cybersecurity fundamentals.

I also gained hands-on experience with Python development, including file handling, modular programming, exception handling, and creating a structured, scalable codebase. This project helped me understand how to design both high-level and low-level architecture, implement logic flows, and maintain clean and organized code that can be extended in the future.

Additionally, I developed strong problem-solving skills by debugging issues, optimizing code, and improving design decisions. The project also helped build my confidence in understanding user requirements, converting them into functional features, and delivering a practical and useful application.

Overall, this internship has significantly contributed to my career growth. I now have a much deeper understanding of secure application development, system design, performance testing, and Python programming. These learnings will help me in my future career, especially in areas such as software development, cybersecurity, data engineering, and backend programming. This experience has strengthened my foundation and prepared me for more advanced professional roles in the IT industry.

8. Future work scope

Although the current version of the Password Manager successfully meets its core objectives secure password storage, encryption, password generation, and user-friendly operations there are several advanced enhancements that can be implemented in the future to make the system more robust, scalable, and industry-ready. Due to limited time, these features could not be included, but they present valuable opportunities for future development.

- **Multi-Platform GUI Application:** A full-fledged graphical interface using libraries like Tkinter, PyQt, or Kivy can significantly enhance user experience. This would allow drag-and-drop functionality, advanced search options, and a more intuitive layout.
- **Cloud Synchronization:** Adding support for encrypted cloud backup (e.g., Google Drive, Dropbox, or a custom cloud API) would enable the user to access their password vault securely across multiple devices.
- **Mobile Application Support:** Creating Android/iOS apps using Kivy, React Native, or Flutter would expand the usability of the Password Manager, making it available on smartphones where password management is most needed.
- **Browser Auto-Fill Extension:** Developing a browser extension for Chrome/Firefox that auto-fills login credentials would make the password manager functionally comparable to commercial tools like LastPass or Bitwarden.
- **Biometric Authentication:** Future versions can integrate biometrics such as fingerprint or face recognition to unlock the vault, improving both security and convenience.
- **Two-Factor Authentication (2FA):** Integrating TOTP-based 2FA (like Google Authenticator) would add an extra layer of security when accessing the vault.