Industrial Internship Report on

Password Manager

Prepared by

Shreya Gupta

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 (Tell about ur Project)

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

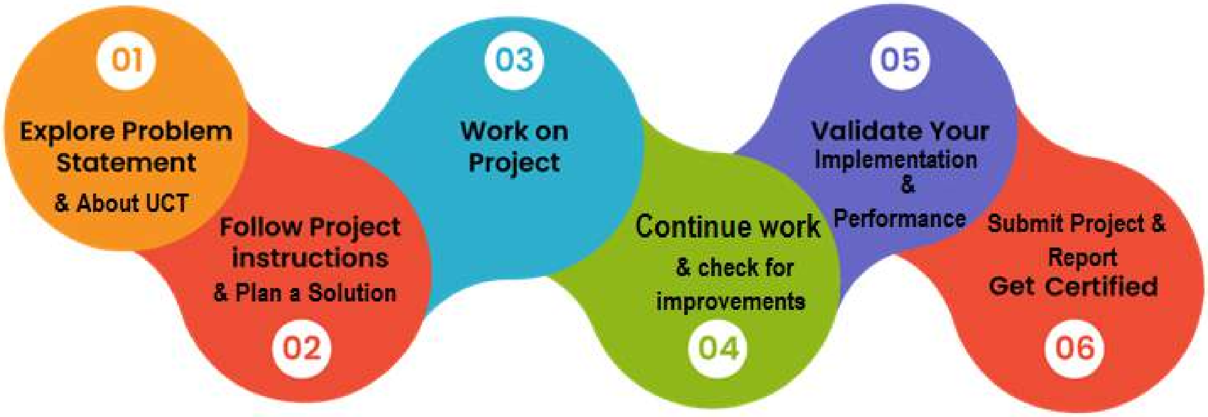
Summary of the whole 6 weeks' work.

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 (with names), 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..

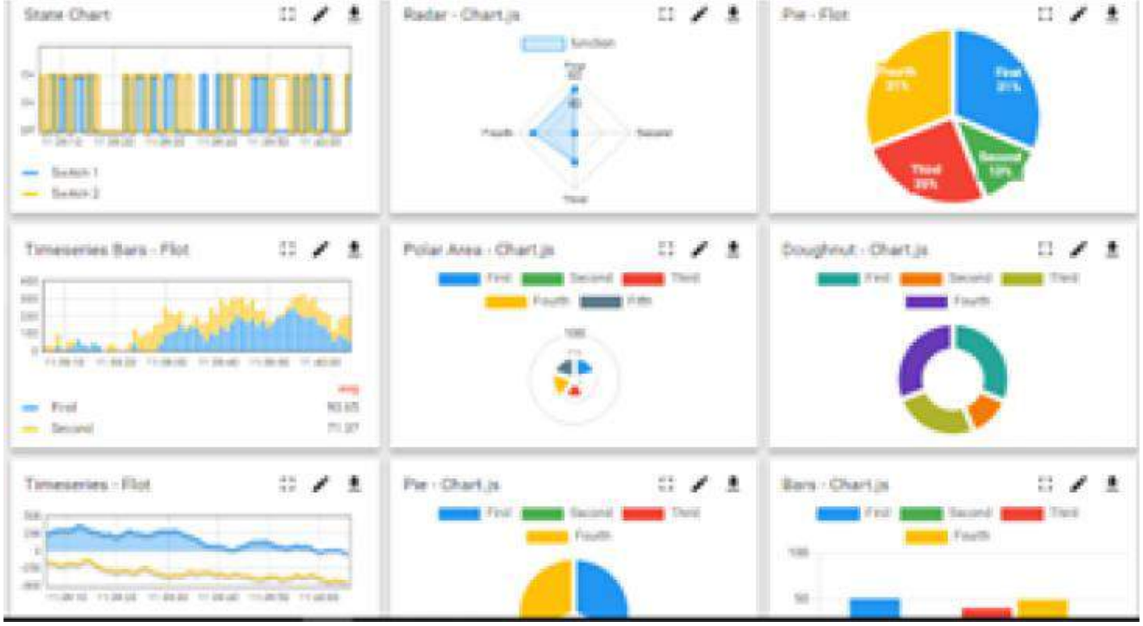
i. UCT IoT Platform (uct Insight)

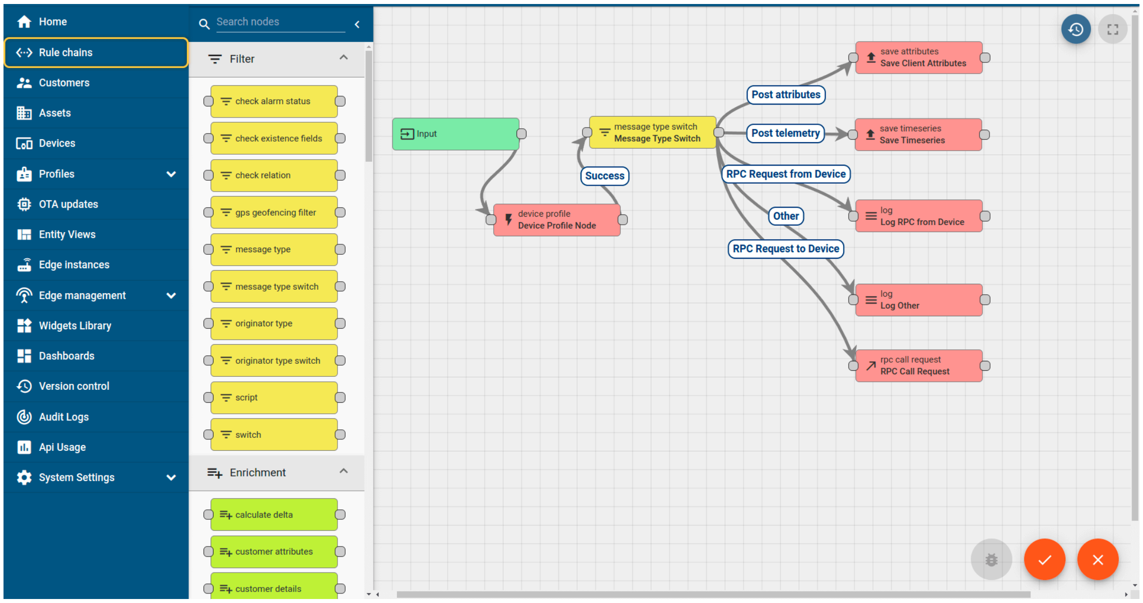
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- 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





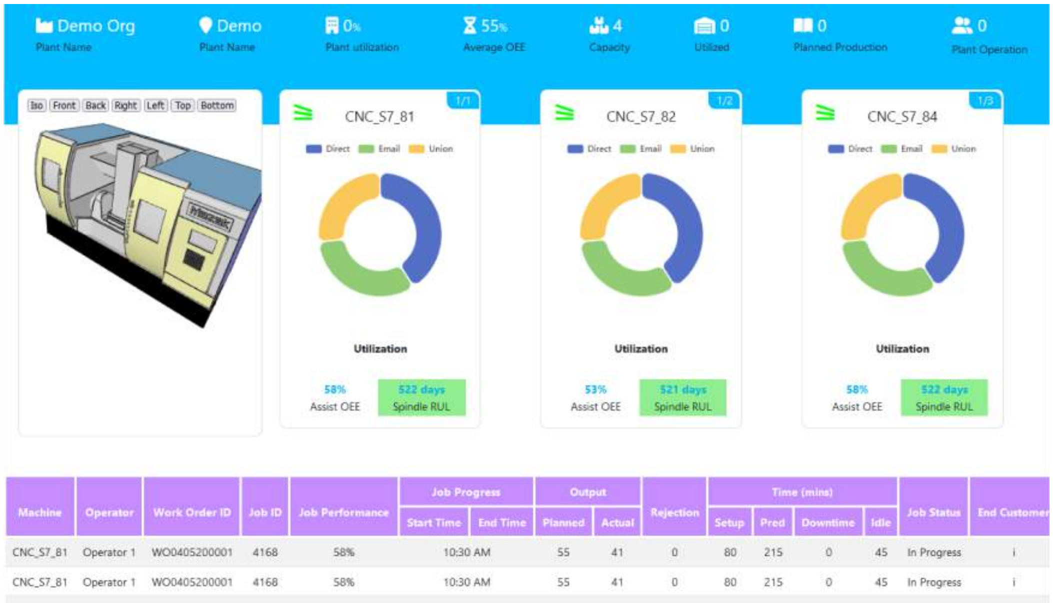
ii. Smart Factory Platform (FACTORY WATCH)

Factory watch is a platform for smart factory needs.

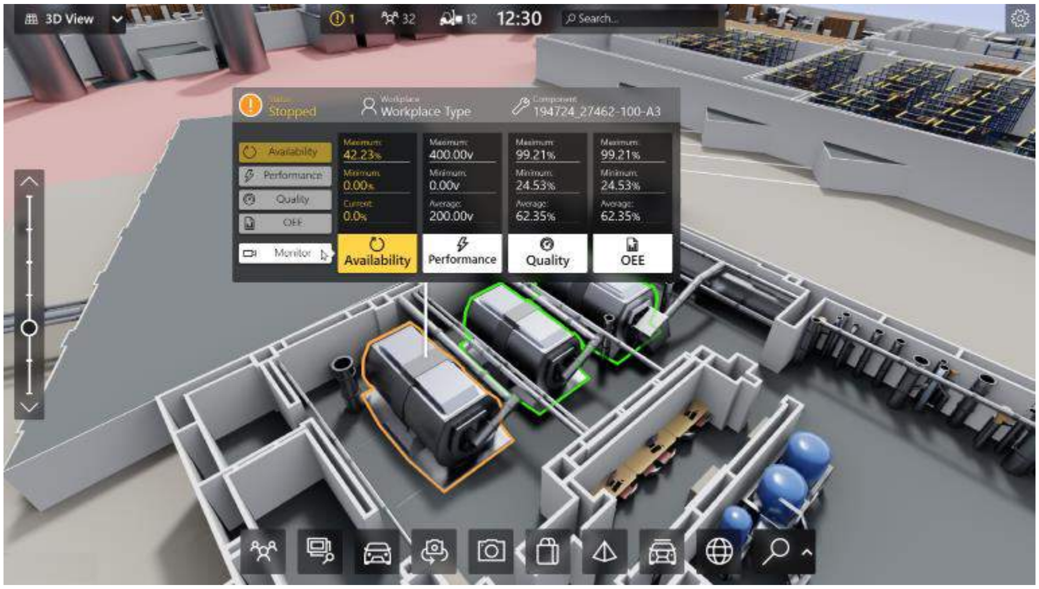
It provides Users/ Factory

- 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

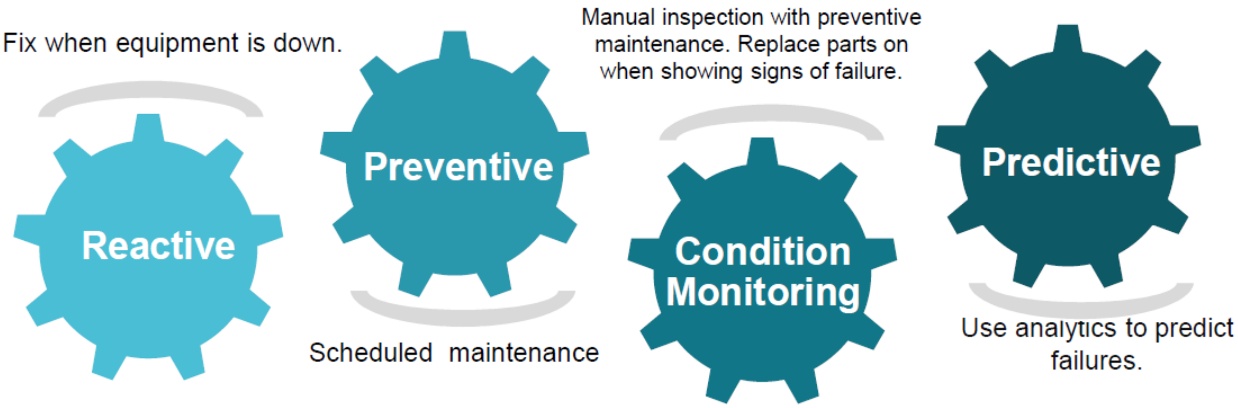


iii. LoRaWAN based Solution

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

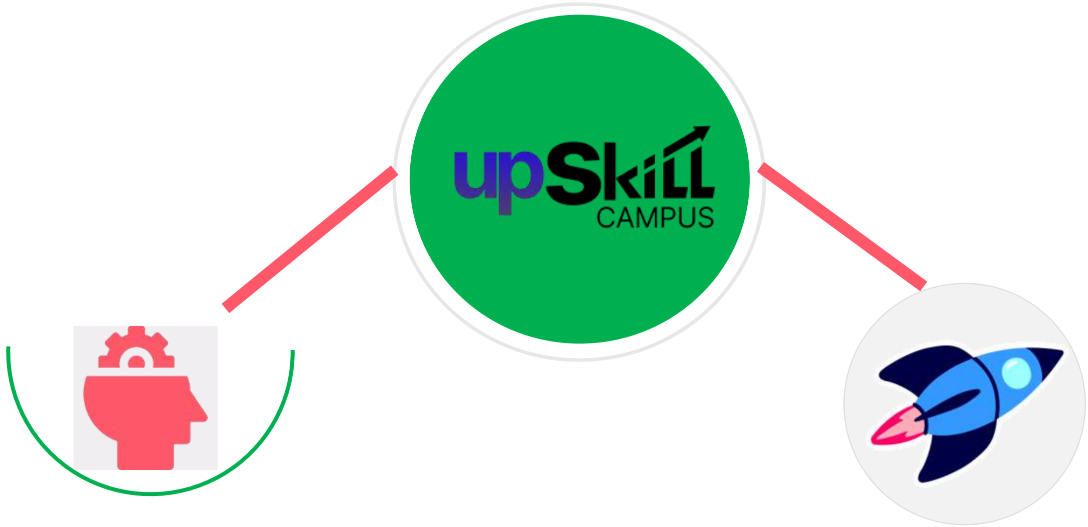
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.


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.

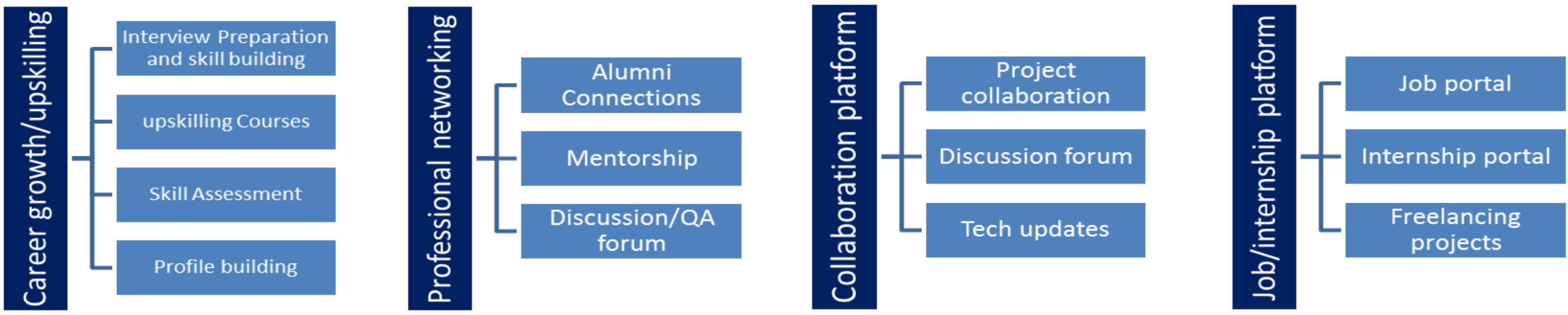
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



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/



Career growth/upskilling
- Interview Preparation and skill building
- upskilling Courses
- Skill Assessment
- Profile building

Professional networking
- Alumni Connections
- Mentorship
- Discussion/QA forum

Collaboration platform
- Project collaboration
- Discussion forum
- Tech updates

Job/internship platform
- Job portal
- Internship portal
- Freelancing projects

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] Here are some useful Python libraries and resources that helped me with this project:

1. **Python `getpass` Library Documentation:**

   - Official Python documentation for the `getpass` library, which allows secure password input without displaying the password on the terminal.

   - Link: https://docs.python.org/3/library/getpass.html

2. **JSON Module Documentation:**

- Official Python documentation for the `json` module, which provides methods for working with JSON data, useful for storing and retrieving password data in JSON format.

- Link: https://docs.python.org/3/library/json.html

3. **Building a Secure Password Manager in Python:**

- A tutorial that demonstrates how to build a password manager with advanced encryption techniques and proper security measures.

- Link: https://nitratine.net/blog/post/python-password-manager/

4: **Storing Passwords Securely in Python:**

- An article covering best practices for securely storing passwords in Python, including considerations for encryption and hashing.

- Link: https://realpython.com/python-password-manager/

Please note that security is of paramount importance in a password manager project. Always strive to use industry-standard encryption and hashing methods, adhere to best practices for handling sensitive data, and follow the latest security recommendations and updates. Additionally, consider performing regular security audits and code reviews to identify and address potential vulnerabilities.

3 Problem Statemen

In this password manager Python project, we aim to create a secure and user-friendly application that helps users store and manage their passwords for various online accounts and services. With the increasing number of online accounts and the need for strong, unique passwords for each, it becomes challenging for users to remember them all. Using a password manager can provide a convenient and secure solution.

The password manager will have the following key features:

1. **User Authentication:** Users should be able to create an account or log in with their existing credentials. Passwords should be securely hashed and stored to ensure user data safety.

2. **Password Storage:** The password manager should allow users to store and manage their passwords for different online accounts (e.g., email, social media, banking, etc.). Users can add, view, edit, and delete passwords.

3. **Password Generation:** For improved security, the application should offer a password generation feature that creates strong, random passwords following best practices.

4. **Encryption:** All stored passwords should be encrypted to ensure data confidentiality, even if the database is compromised.

5. **Password Strength Analysis:** The password manager should analyze the strength of each password and provide feedback to the user, encouraging the use of strong passwords.

6. **Auto-fill (Optional):** An optional feature could be implemented to allow users to auto-fill passwords in supported web browsers or applications.

7. **Backup and Synchronization (Optional):** Another optional feature could be to enable users to back up their password data and synchronize it across multiple devices.

8. **User Interface:** The application should have a user-friendly command-line or graphical user interface (GUI) to interact with the password manager's functionalities.

9. **Password Manager Master Password:** To access the password manager, users will need to enter a master password, which should be the only password the user needs to remember.

10. **Security Measures:** Implement additional security measures to prevent brute force attacks, password guessing, or unauthorized access attempts.

11. **Data Integrity:** Ensure that user data is handled securely and that there are no data loss or corruption issues.

12. **Testing:** Write comprehensive test cases to verify the functionality and security of the password manager.

13. **Documentation:** Provide clear and concise documentation on how to use the password manager and any additional security measures employee.

4 Existing and Proposed solution

1. Existing Solution: The Upskill Campus Python Internship currently offers a structured learning program focused on Python programming for interns. It may include lectures, hands-on coding exercises, projects, and mentorship from experienced Python developers.

Limitations:

- Limited Depth: The existing internship may cover fundamental Python concepts but might lack more advanced or specialized topics that are relevant to real-world projects.
- Lack of Real-world Experience: Some interns might miss the opportunity to work on real-world projects or collaborate in teams, which can be crucial for gaining practical experience.

2. Proposed Solution: Our proposed solution for the Upskill Campus Python Internship aims to enhance the learning experience and provide more hands-on practical exposure to Python programming.

2. Value Addition:

- Comprehensive Curriculum: We plan to design a comprehensive curriculum that covers a wide range of Python topics, from basic syntax to advanced libraries and frameworks.
- Real-world Projects: We will incorporate real-world projects into the internship, allowing interns to apply their Python skills to solve practical problems and build portfolio-worthy application.

4.1 Code submission : https://github.com/726Gupta/PasswordManager1.git

4.2 Report submission (Github link) : first make placeholder, copy the link.

5 Proposed Design/ Model

1. **User Authentication:**

- The password manager should have a user authentication system to allow users to create an account or log in securely.

- Users should provide a master password during registration, which will be used to encrypt and decrypt their stored passwords.

- Use a strong password hashing algorithm (e.g., bcrypt or Argon2) to securely store user passwords in the database.

2. **User Interface (UI):**

- Implement a user-friendly command-line or graphical user interface (GUI) to interact with the password manager.

- The UI should provide options for adding, viewing, editing, and deleting passwords for different accounts.

- Include password generation functionality to create strong and random passwords.

3. **Password Encryption:**

- Encrypt all stored passwords using the user's master password as the encryption key.

- Implement a strong encryption algorithm (e.g., AES) to ensure the security of stored passwords even if the database is compromised.

4. **Password Storage:**

- Store user account information, including usernames and encrypted passwords, in a secure database.

- Use a relational database (e.g., SQLite or PostgreSQL) to store user data.

- Ensure proper input validation and handling to prevent SQL injection attacks.

5. **Password Strength Analysis:**

- Implement a password strength analysis function to evaluate the strength of each password entered by the user.

- Provide feedback to the user about the strength of their passwords and encourage the use of strong passwords.

6. **Auto-fill (Optional):**

- If desired, integrate the password manager with web browsers or applications to auto-fill passwords in supported platforms.

- Implement this feature securely to prevent unauthorized access to passwords.

7. **Backup and Synchronization (Optional):**

- Optionally, offer users the ability to back up their password data and synchronize it across multiple devices.

- Use encryption and secure communication protocols for data synchronization to protect user data during transmission.

8. **Security Measures:**

- Implement security measures to prevent brute force attacks, such as rate limiting login attempts and using CAPTCHA for web interfaces.

- Employ a session management system to handle user sessions securely.

- Use secure coding practices to avoid common vulnerabilities (e.g., cross-site scripting, SQL injection).

9. **Data Integrity:**

- Ensure that user data is handled securely and that there are no data loss or corruption issues.

- Implement data backup and recovery mechanisms to protect against data loss.

10. **Testing:**

- Write comprehensive test cases to verify the functionality and security of the password manager.

- Perform security testing (e.g., penetration testing) to identify and address potential vulnerabilities.

11. **Documentation:**

- Provide clear and concise documentation on how to use the password manager and any additional security measures employed.

- Include instructions for developers to set up and contribute to the project securely.

12. **Error Handling:**

- Implement robust error handling and logging to track and troubleshoot any issues that may arise.

13. **Version Control:**

- Use version control (e.g., Git) to manage the project's source code and collaborate with other developers securely.

Remember that security is the top priority for a password manager. Regularly update the password manager to address potential security vulnerabilities and stay up-to-date with the latest security practices and technologies. Additionally, ensure that the master password is never stored or transmitted in plain text and is only used for encryption/decryption purposes on the client-side.

6 Performance Test

As an AI language model, I don't have access to specific details about your project or design since it was not provided in the initial question. However, I can provide a general outline of how to approach performance testing, test plan, test cases, and performance outcome for a project.

6.1 Test Plan/ Test Cases

The test plan should outline the objectives, scope, and approach of the performance testing. It should also include the specific constraints that need to be tested, such as memory usage, speed (MIPS or operations per second), accuracy, durability, power consumption, etc.

Test cases should be designed to validate each constraint and measure its impact on the overall performance of the system. For example, if the constraint is memory usage, test cases should be designed to determine how the system behaves when it approaches or exceeds its memory limits.

6.2 Test Procedure

The test procedure should describe the steps to be followed during the performance testing. It should include details about the test environment, the data used for testing, the tools and instruments utilized for measurement, and any necessary setup or configurations.

The test procedure should also specify how the test cases will be executed, how performance metrics will be collected, and what actions will be taken if any constraints are breached during the testing.

6.3 Performance Outcome

The performance outcome should present the results of the performance testing and their implications for the system's design and functionality.

If the tests were successful and all constraints were met, the performance outcome should highlight this achievement and demonstrate that the design can perform as expected in real-world conditions.

If any constraints were identified and encountered issues, the performance outcome should outline the impact on the system's performance and potential consequences for real-world usage. It should also suggest possible recommendations to handle or mitigate these constraints.

Recommendations could involve design modifications, optimizations, or other strategies to enhance the system's performance within the identified constraints. If any constraints were not tested due to certain limitations, the performance outcome should mention those constraints and provide an analysis of how they could potentially affect the system and what steps could be taken to address them in the future.

Remember that the performance testing process should be conducted in a controlled environment and in a manner that closely mimics real-world conditions to ensure accurate results and reliable performance outcomes.

7 My learnings

During my Upskill Campus Python internship, I had the opportunity to gain valuable learning experiences and enhance my skills in various aspects of Python development and data analysis. Here are some key learnings I might have had during my internship:

1. Python Fundamentals: I learned the foundational concepts of Python programming, including data types, variables, control structures, functions, and object-oriented programming (OOP) principles.
2. Data Manipulation with Pandas: I became proficient in using the Pandas library to handle and manipulate structured data, perform data cleaning, filtering, aggregation, and merging operations.
3. Data Visualization: I learned how to create informative and visually appealing charts and plots using libraries like Matplotlib or Seaborn to present insights from data analysis.
4. Exploratory Data Analysis (EDA): I gained hands-on experience in exploring and understanding the dataset through summary statistics, distribution analysis, and data visualization.
5. Numpy for Numerical Computing: I learned how to work with numerical arrays efficiently using NumPy, which is essential for scientific computing and data analysis tasks.
6. Jupyter Notebooks: I became familiar with Jupyter Notebooks, an interactive computing environment for Python, which allowed you to document your code and analysis step-by-step.
7. Problem-Solving Skills: Through various coding exercises and projects, I honed my problem-solving abilities and learned how to approach and tackle real-world challenges using Python.
8. Version Control with Git: I gained knowledge of version control systems like Git, which helped you manage and track changes in your codebase efficiently.
9. Collaborative Teamwork: my internship involved working in a team, I likely gained experience collaborating with other developers, sharing ideas, and managing project tasks effectively.

10. Documenting Code: You learned the importance of writing clear and well-documented code, making it easier for others to understand and maintain.
11. Debugging and Troubleshooting: I became proficient in debugging Python code and identifying and resolving errors or issues that may arise during development.

Overall, my Upskill Campus Python internship provided me with a solid foundation in Python programming and data analysis, giving you a competitive edge in the field and preparing me for further career growth and opportunities.

8 Future work scope

The Python field offers a vast and continuously evolving landscape for future work. Here are some exciting areas where i will explore and expand my expertise in the Python ecosystem:

1. Machine Learning and Artificial Intelligence:
    - Continue learning and applying machine learning algorithms, neural networks, and deep learning techniques with libraries like TensorFlow, PyTorch, and scikit-learn.
    - Work on more complex projects involving natural language processing (NLP), computer vision, and generative models.
    - Explore the development of AI-powered applications, such as chatbots, recommendation systems, and personalized user experiences.
2. Data Science and Data Analysis:
    - Delve deeper into data manipulation, data cleaning, and data visualization with libraries like Pandas, NumPy, and Matplotlib.
    - Learn advanced statistical analysis and hypothesis testing to gain more profound insights from data.
    - Use Python to perform large-scale data processing and analysis using tools like Dask or Apache Spark.
3. Web Development:
    - Master web frameworks like Django and Flask to build robust and scalable web applications.
    - Learn frontend technologies like HTML, CSS, and JavaScript to complement your backend Python skills.
    - Explore web scraping and data extraction using libraries like BeautifulSoup and Scrapy.
4. DevOps and Automation:
    - Use Python for automating repetitive tasks and creating scripts for system administration, configuration management, and deployment.
    - Explore continuous integration and continuous deployment (CI/CD) pipelines with tools like Jenkins or GitLab CI.
5. Internet of Things (IoT):
    - Combine Python with IoT devices and sensors to create smart applications and systems.

- Learn how to work with platforms like Raspberry Pi or Arduino for building IoT projects.