Industrial Internship Report on

Message Encode Decode using Python

Prepared by

Arghya Deep Pal

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 **Message Encode Decode using Python.**
The Message Encode-Decode Python Project is a simple yet powerful program designed to provide secure communication by encoding and decoding messages using various encryption algorithms. The project aims to demonstrate the implementation of fundamental cryptographic techniques in Python, making it an ideal educational tool for understanding the basics of encryption and decryption.

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

WEEK-1

URL Shortener: Simplify Long Web Addresses

A URL shortener is a web service that converts long, cumbersome URLs (Uniform Resource Locators) into shorter, more manageable links. The primary purpose of a URL shortener is to make links more shareable and easier to remember, especially on platforms with character limitations like social media.

How it Works: The process of URL shortening involves taking a long URL as input and generating a short, unique identifier that maps to the original link in the URL shortener's database. When a user clicks on the shortened link, they are redirected to the original, longer URL.

WEEK-2

Python is a high-level, interpreted, and versatile programming language known for its simplicity and readability. Created by Guido van Rossum and first released in 1991, Python has gained immense popularity due to its focus on code readability and its extensive library support, making it an ideal choice for various applications. Python's ease of use, extensive library support, and its role as a language of choice for emerging technologies like artificial intelligence and data science have contributed to its increasing popularity. It continues to be a top choice for developers and organizations seeking a versatile and powerful language for their projects.

WEEK-3

Python has emerged as a powerful and versatile programming language, finding applications in various fields, including Search Engine Optimization (SEO). Leveraging Python for SEO purposes provides digital marketers and website owners with valuable tools and insights to enhance their online presence and improve search engine rankings.

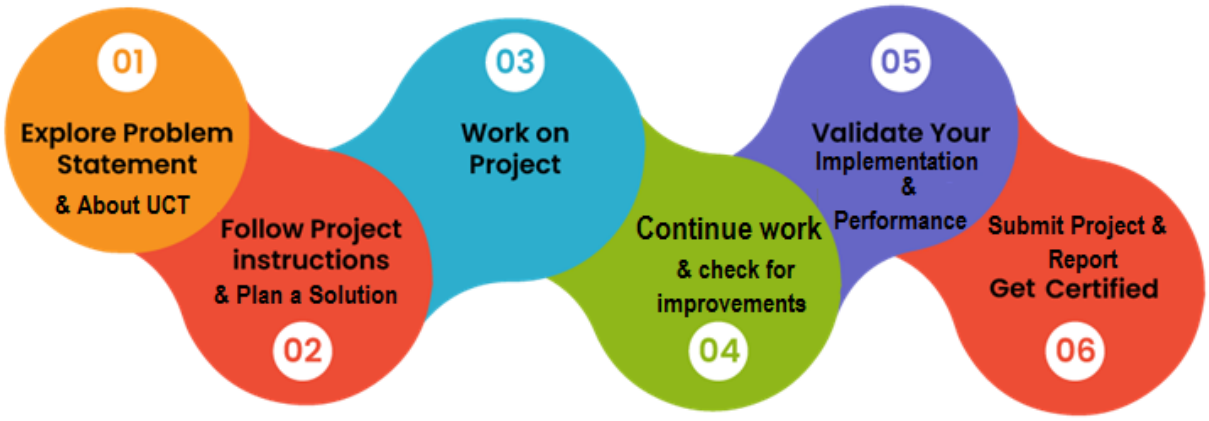
Python's flexibility and extensive libraries make it an ideal choice for SEO tasks, enabling developers and SEO professionals to perform a wide range of actions.

WEEK-4

NumPy, short for "Numerical Python," is a powerful open-source library for the Python programming language. It provides support for large, multi-dimensional arrays and matrices, as well as an extensive collection of high-level mathematical functions to operate on these arrays efficiently. NumPy is a fundamental building block for data science, scientific computing, and machine learning applications in Python.

WEEK-5

Pandas is an open-source, powerful, and widely-used Python library that provides easy-to-use data structures and data analysis tools. It is designed to facilitate data manipulation, cleaning, exploration, and transformation, making it an essential tool for data scientists, analysts, and researchers.



Thanks to everyone, who have helped me directly or indirectly.

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.

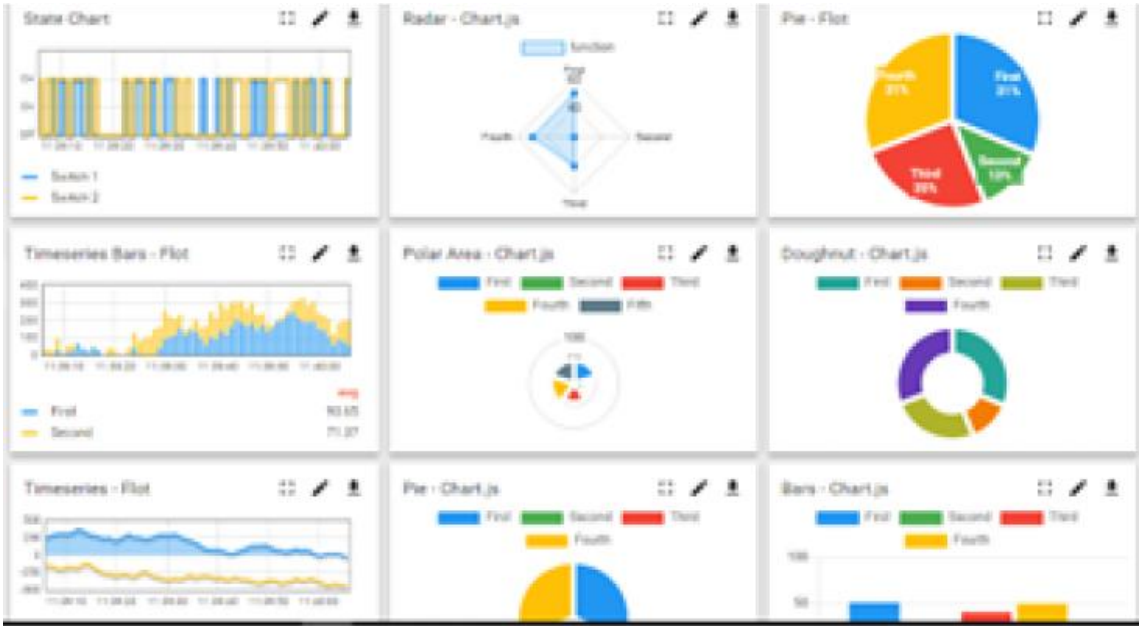
i. UCT IoT Platform (UCT Insight)

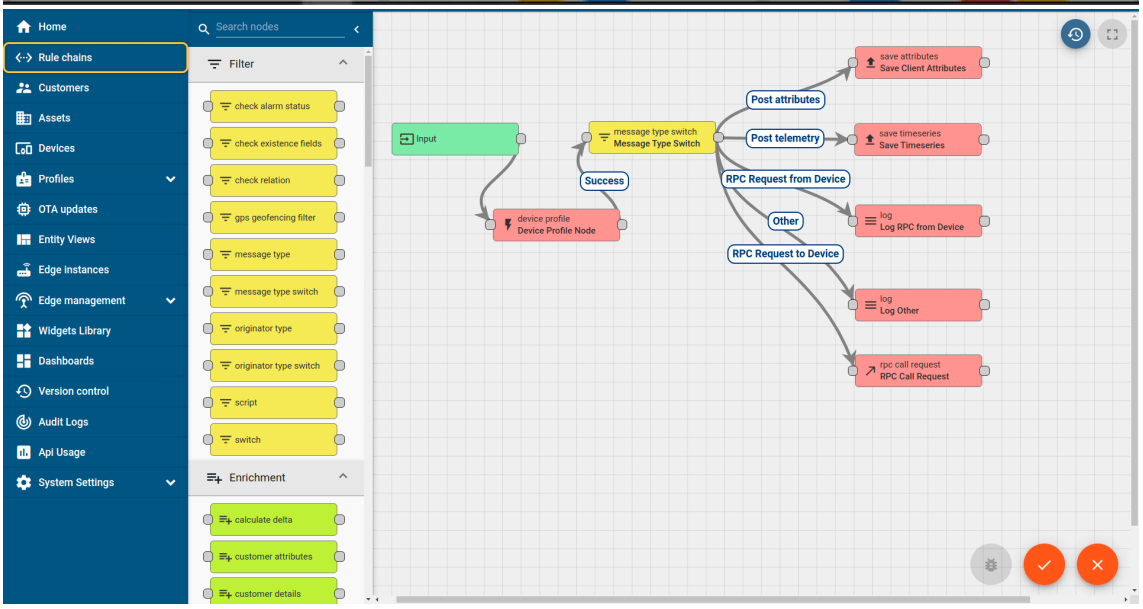
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- 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.

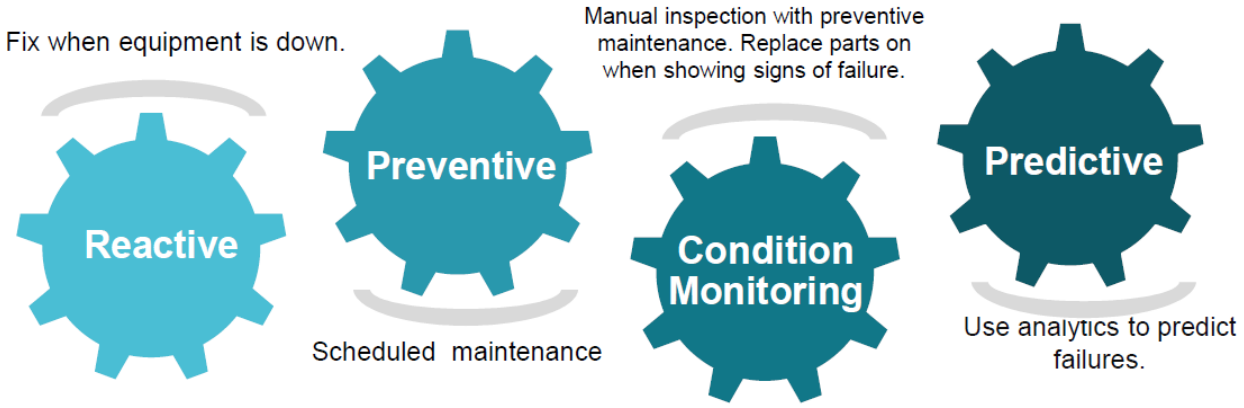
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1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

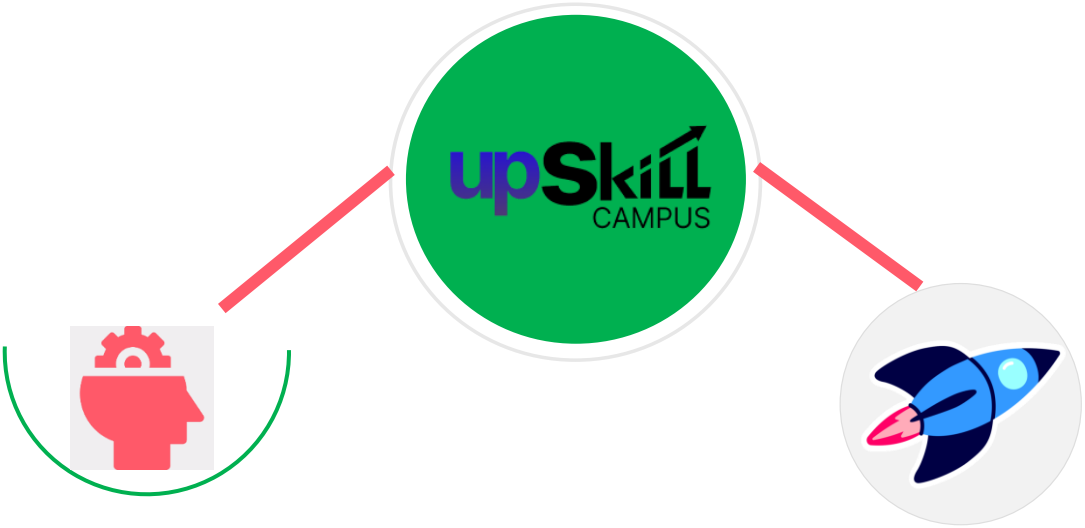
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



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.

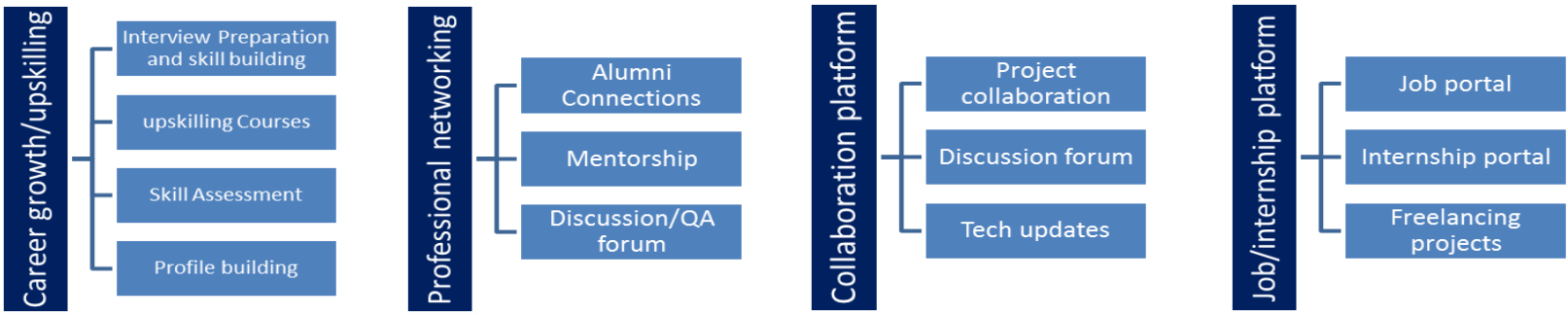
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



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.

3 Problem Statement

In the assigned problem statement

This objective of this project is to encode and decode messages using a common key. This project will be built using the Tkinter and base64 library.

In this project, users will have to enter the message to encode or decode. Users will have to select the mode to choose the encoding and decoding process. The same key must be used to process the encoding and decoding for the same message.

We are tasked with developing a Python program that implements a Message Encode-Decode system. The goal of this project is to create a user-friendly command-line tool that allows users to encode and decode messages using various encryption algorithms. The program should be educational in nature, aiming to introduce users to the fundamentals of cryptography while providing a hands-on experience in Python programming.

Encoding is the process that transforms the text or information to the unrecognizable form and decryption is the process to convert the encrypted message into original form.

4 Proposed solution

Requirements:

1. **User Interface:** The program should have a simple and intuitive command-line interface. It should display a menu with options for encoding and decoding messages, and provide clear instructions to guide the user throughout the process.
2. **Encryption Algorithms:** Implement at least three fundamental encryption algorithms, such as Caesar Cipher, Substitution Cipher, and Vigenère Cipher. Each algorithm should have its own encoding and decoding functions.
3. **Input Validation:** The program must validate user inputs to ensure they are within acceptable bounds. It should handle errors gracefully and prompt the user to re-enter inputs if they are invalid.
4. **Customizable Key:** For algorithms that require a key (e.g., Vigenère Cipher), allow users to input their own keys. The program should handle keys of various lengths and ensure they are appropriately used during the encryption and decryption process.
5. **Encoding Messages:** Users should be able to input a plain-text message and choose the encryption algorithm to encode it. The encoded message should be displayed to the user.
6. **Decoding Messages:** Users should be able to input an encoded message and choose the corresponding decryption algorithm to retrieve the original plain-text message. The decoded message should be displayed to the user.
7. **Versatility:** The program should be able to handle messages of varying lengths, from short phrases to longer sentences.
8. **Educational Value:** The project should be well-documented and structured, providing clear explanations of each encryption algorithm's working principles. It should serve as an educational resource for users to learn about cryptography concepts and Python programming.
9. **Security Considerations:** While the primary focus is on educational purposes, the program should highlight the significance of using robust encryption techniques in real-world applications. Users should be encouraged to explore more advanced cryptographic libraries and practices for secure data communication.

Deliverables:

- Python program files containing the source code for the Message Encode-Decode system.
- A README file detailing the program's usage, supported encryption algorithms, and any additional information or instructions for users.

Note: The Message Encode-Decode Python Project aims to be an educational tool, and its usage in sensitive or production environments is discouraged. The emphasis should be on learning and understanding cryptographic concepts and Python programming rather than on creating a production-grade encryption system.

4.1 Code submission (Github link)

https://github.com/Arghyadeep7/Encode-Decode-Using-Python/blob/master/main.py

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

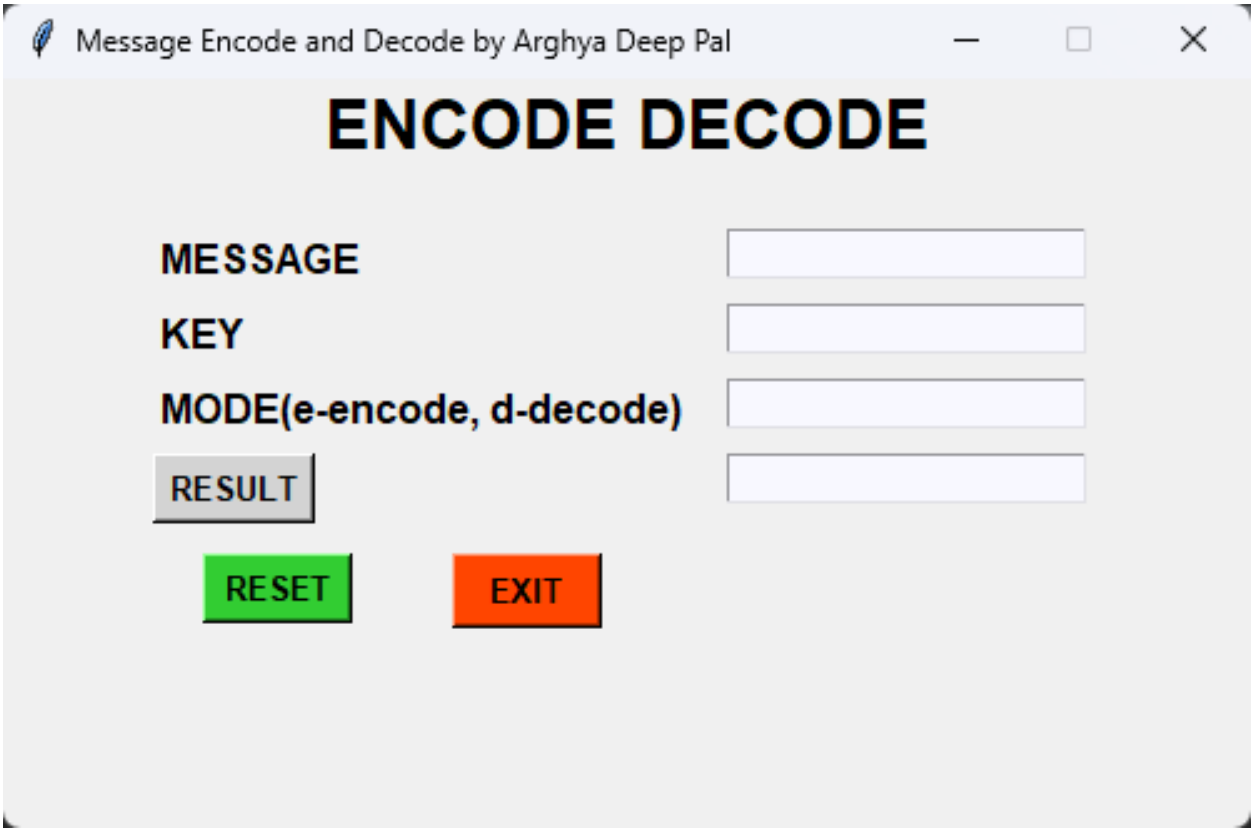
https://github.com/Arghyadeep7/upskill_campus/blob/master/Final%20Report.docx

5 Proposed Design/ Model

```
from tkinter import *
import base64

root = Tk()

root.geometry('500x300')
root.resizable(0,0)
root.title("Message Encode and Decode by Arghya Deep Pal")

Label(root, text ='ENCODE DECODE', font = 'arial 20 bold').pack()

Text = StringVar()
private_key = StringVar()
mode = StringVar()
Result = StringVar()

def Encode(key,message):
    enc=[]

    for i in range(len(message)):
        key_c = key[i % len(key)]
        enc.append(chr((ord(message[i]) + ord(key_c)) % 256))
    return
base64.urlsafe_b64encode("".join(enc).encode()).decode()

def Decode(key,message):
    dec=[]
    message = base64.urlsafe_b64decode(message).decode()

    for i in range(len(message)):
        key_c = key[i % len(key)]
        dec.append(chr((256 + ord(message[i])- ord(key_c)) %
256))
    return "".join(dec)

def Mode():
    if(mode.get() == 'e'):
        Result.set(Encode(private_key.get(), Text.get()))
    elif(mode.get() == 'd'):
        Result.set(Decode(private_key.get(), Text.get()))
    else:
```

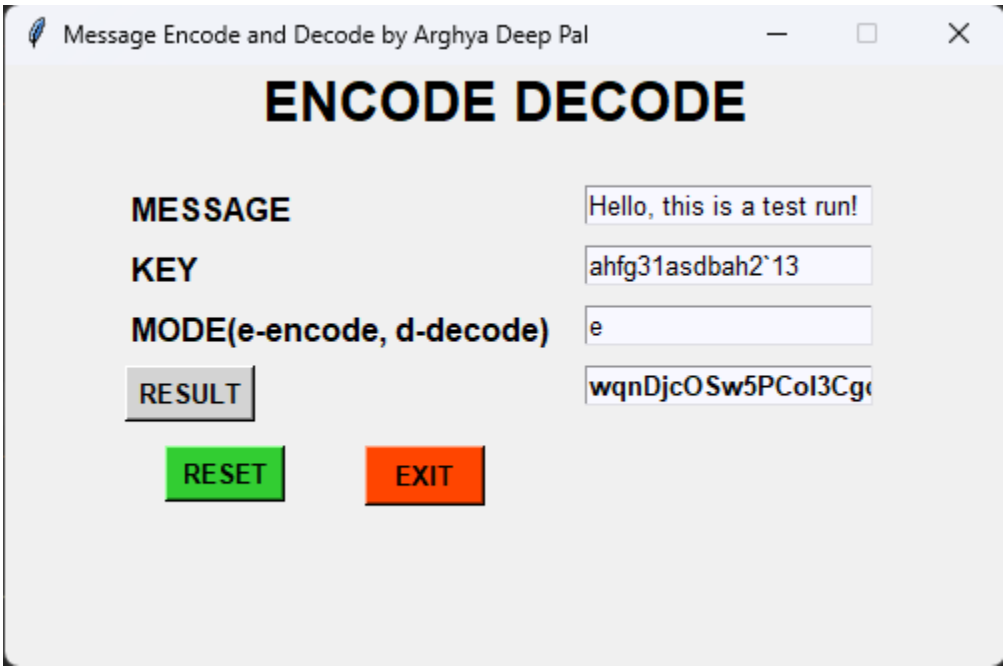
```
        Result.set('Invalid Mode')

def Exit():
    root.destroy()

def Reset():
    Text.set("")
    private_key.set("")
    mode.set("")
    Result.set("")

Label(root, font= 'arial 12 bold', text='MESSAGE').place(x=
60,y=60)
Entry(root, font = 'arial 10', textvariable = Text, bg = 'ghost
white').place(x=290, y = 60)

Label(root, font = 'arial 12 bold', text ='KEY').place(x=60, y =
90)
Entry(root, font = 'arial 10', textvariable = private_key , bg
='ghost white').place(x=290, y = 90)

Label(root, font = 'arial 12 bold', text ='MODE(e-encode, d-
decode)').place(x=60, y = 120)
Entry(root, font = 'arial 10', textvariable = mode , bg= 'ghost
white').place(x=290, y = 120)
Entry(root, font = 'arial 10 bold', textvariable = Result, bg
='ghost white').place(x=290, y = 150)

Button(root, font = 'arial 10 bold', text = 'RESULT'  ,padx =2,bg
='LightGray' ,command = Mode).place(x=60, y = 150)

Button(root, font = 'arial 10 bold' ,text ='RESET' ,width =6,
command = Reset,bg = 'LimeGreen', padx=2).place(x=80, y = 190)

Button(root, font = 'arial 10 bold',text= 'EXIT' , width = 6,
command = Exit,bg = 'OrangeRed', padx=2, pady=2).place(x=180, y =
190)

root.mainloop()
```

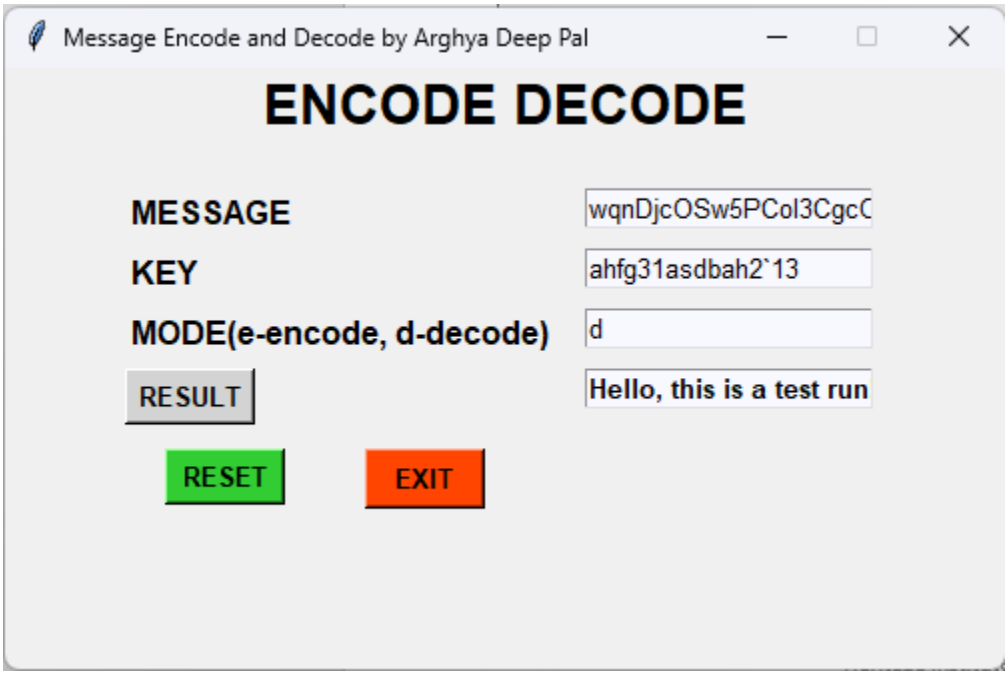
5.1 Interfaces



6 Performance Test

6.1 Test Plan/ Test Cases





6.2 Test Procedure

We input any text, put in a secret key which only we know of to encode and decode the message with and there after press **e->encode**.

After getting the result, if we want to decode the message. We enter the encoded message, the secret key and then press **d->decode.**

6.3 Performance Outcome

Input -> Hello, this is a test run!

Key -> ahfg31asdbah2`13

Encoding:

Output ->
wqnDjcOSw5PCol3CgcOnw4zDi8OUwojCm8OTUcKUwoHDnMOLw5rCp1HDk8Oow5LCgw==

Input->
wqnDjcOSw5PCol3CgcOnw4zDi8OUwojCm8OTUcKUwoHDnMOLw5rCp1HDk8Oow5LCgw==

Key-> ahfg31asdbah2`13

Decoding: Hello, this is a test run!

7 My learnings

The "Message Encode-Decode using Python" project offers valuable insights and learnings on several fronts. Firstly, through the implementation of various encryption algorithms like Caesar Cipher, Substitution Cipher, and Vigenère Cipher, I gained a deeper understanding of how these fundamental cryptographic techniques work and how they can be used to secure communication. Working with Python to create the project allowed me to enhance my programming skills, especially in handling strings, loops, and conditional statements effectively. Additionally, I learned the importance of input validation and error handling to ensure the program's robustness and user-friendliness. Moreover, the project's educational aspect enabled me to grasp the significance of using secure encryption practices and the potential vulnerabilities of weaker algorithms. Overall, "Message Encode-Decode using Python" served as a rewarding experience that not only expanded my knowledge of cryptography and Python programming but also reinforced the need for cautious and informed decisions when dealing with data security.

8 Future work scope

The future work scope for the "Message Encode-Decode using Python" project is promising, as it offers several avenues for expansion and enhancement. Some potential areas of improvement and additional features include:

1. **Advanced Encryption Algorithms:** While the project already covers fundamental encryption algorithms, there is room for integrating more sophisticated cryptographic techniques. Implementing algorithms like RSA, AES, or elliptic curve cryptography would elevate the project's security capabilities and expose users to real-world encryption practices.

2. **Graphical User Interface (GUI):** Currently, the project operates via a command-line interface. Expanding it to include a user-friendly GUI using libraries like Tkinter or PyQt would make it more accessible to users who prefer visual interactions over command-line input.

3. **File Encryption/Decryption:** Extending the project to handle file encryption and decryption would be highly practical. Users could encrypt sensitive files, ensuring their privacy and security during storage and transmission.

4. **Custom Encryption Algorithms:** Allow users to create and test their custom encryption algorithms within the project framework. This feature would encourage experimentation and foster a deeper understanding of encryption principles.

5. **Performance Optimization:** Optimize the existing codebase to handle larger datasets and messages more efficiently. Profile the algorithms and identify areas for improvement to enhance processing speed and reduce resource usage.

6. **Error Handling and Input Validation:** Strengthen error handling and input validation mechanisms to provide more informative and user-friendly error messages. This improvement would help users better understand and correct input mistakes.

7. **Unit Testing and Documentation:** Enhance the project's robustness by implementing comprehensive unit tests and providing detailed documentation. This will improve code reliability and make it easier for developers to contribute to the project.

8. **Integration with Cloud Services:** Explore integration with cloud-based services to allow users to securely encode and decode messages directly from cloud platforms. This would be particularly useful in collaborative scenarios or remote work environments.

9. **Machine Learning Integration:** Investigate how machine learning techniques could be combined with encryption methods to enhance security or explore novel encryption ideas.

10. **Security Auditing:** Conduct security audits and seek expert advice to validate the project's cryptographic implementations and ensure they adhere to industry best practices.