Industrial Internship Report on

URL Shortener

Prepared by

Chris Daniel Wilson

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.

The project I chose to undertake for the Python Internship offered upskill Campus was the URL Shortener, an application developed in Python that will allow users to shorten the links they input into it. The application will take long URLs and convert them into smaller and much more manageable links using an API. Apart from that, the overall scope of this project involved implementing a database to store the mapping between original and shortened URLs, and developing functions to generate unique shortened URLs and handle redirection.

This internship gave me a very good opportunity to get exposure to Industrial problems, and to design and then implement solutions for those. It was an overall great experience to have this internship.

TABLE OF CONTENTS

1 Preface

During the first week of the internship, I did some research on Python, its syntax and how it worked in comparison to the other programming languages I had worked with so far. I also analyzed all the options for projects we were given to choose from, and elected to work on creating a URL Shortener. Consequently, I kept the project's primary goals and its problem statement in mind as I began to develop my chosen project.

In the second week, I decided to solidify the core functionality of my project, namely its URL Shortening functionality. After familiarizing myself with the *pyshorteners* library in python, I created a test program to see It functioned as intended, which it did. I also decided to primarily use the *customtkinter* library for creating the UI my project needed.

The third week was when I got to work on the appearance of my application, and the implementation of the URL Shortening feature in it. Much of the widgets that were used in *tkinter* and customtkinter are very similar to the ones used in Visual Basic, a language that I have some experience with. I learnt how to use simple widgets like *Labels*, *Entries* and *Buttons* and created a simple program using all three. With the knowledge I then had, I made a simple URL Shortener with a basic but adequate user interface that accepted inputs in the form of long URLs, and gave output in the form of shortened ones. Besides the URL Shortening function, I also created a function that validated the inputted links by checking if they replied with "200 OK" to a HEAD request. This was made possible by importing the *Requests* library.

In the fourth week, I addressed the project's requirement of database functionality. Through the use of the *sqlite3* package, I was able to use some standard *SQL* commands in Python, which I tested out in a sample program. In the sample program, I inserted some information into a database in *SQLite*. The information was clearly viewable in the database using a tool called *DB Browser*. I was now able to successfully add this database functionality to the code I had developed the previous week, allowing for the storage as well as the mapping of shortened and original URLs in a locally embedded database.

I neared finalization of the coding aspect of my project in the fifth week and added some entries to my projects non-functional requirements, since I had satisfied all the functional requirements. I improved the application by adding more elements, and used a different method to more effectively place most of them. I also reworked the URL validation function I added earlier, enabling it to allow users to shorten more URLs than they previously could.

The sixth week is when I thoroughly reviewed all of my progress thus far. I conducted various tests to gauge the real-world performance of my application. I made minor changes to my code after extensive troubleshooting. This is also the week in which I will submit my final report detailing all of my work and progress so far, which brings me to where I am right now, as of the time I am writing this report.

Internships, such as the one offered by upskill Campus that I am undertaking, play a significant role in career development and can have various benefits for individuals seeking to build their professional paths. First and foremost, they provide an opportunity to gain hands-on experience in a specific field or industry. They will allow individuals to apply theoretical knowledge obtained from academic studies to real-world situations. Internships offer a chance to gain valuable insights into a particular industry or professional field. You can learn about industry practices, trends, and the day-to-day operations of the workplace. This knowledge can help you make informed decisions about your career path and develop a better understanding of the industry's expectations.

Internships allow you to explore different career paths and industries. They help you understand your interests, strengths, and weaknesses in a practical setting. This firsthand experience can help you make more informed decisions about your future career and prevent potential career changes down the road. Internships also provide excellent networking opportunities. Building relationships with professionals in your chosen field can lead to mentorship, future job opportunities, and references. Networking with colleagues, supervisors, and industry professionals can expand your professional network and open doors for future career growth.

Internships also help develop a wide range of skills, both technical and soft skills. Technical skills may include specific software proficiency, data analysis, project management, or laboratory techniques, depending on the field. Soft skills such as communication, teamwork, time management, and problem-solving are also honed through internships. In addition to the skill you've acquired, having internships on your resume demonstrates to potential employers that you have practical experience and are proactive in pursuing professional development. Internships showcase your ability to apply classroom knowledge to real-world scenarios and make you a more competitive candidate when applying for jobs.

The project I chose for my internship was the **URL Shortener**. The problem statement pertaining to my project describes a scenario in which one wishes to **shorten a URL** for the purpose of convenience, avoiding overcomplication of documentation, or simply adhering to character limits in certain social media platforms. The core features of my project are as follows:

- Dedicated functions to generate unique shortened URLs. These functions will also handle redirection.
- An aesthetically pleasing, modern and intuitive user interface.
- A database to store data relevant to the mapping of the original URLs to their shortened counterparts and vice versa.

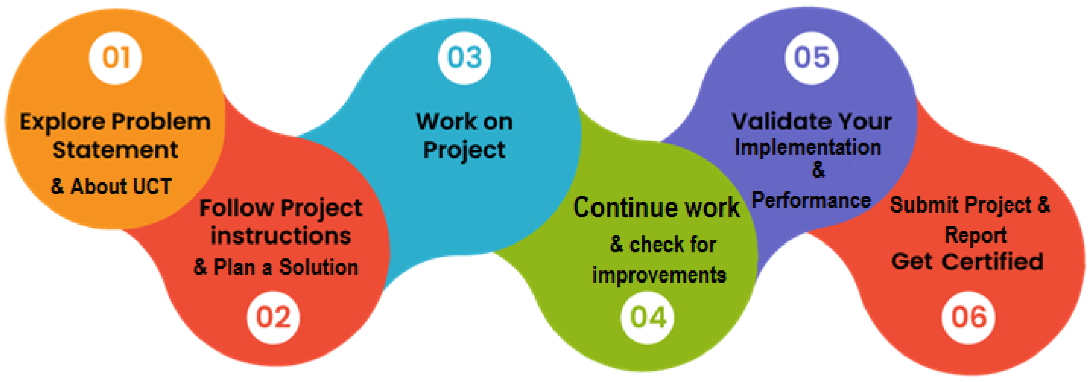
Keeping with its original problem statement, however, the basic objective of my project will be to shorten URLs and convert them into smaller links that are easier to use. The shortened links can be implemented into memos, emails, announcements on social media and even other projects. These links will redirect users to the original link.

When I was required to take an internship by my college during my vacation, I was overjoyed to find a free internship offered by Upskill Campus in multiple interesting subjects like Core Java, App Development, Digital Marketing, etc. One course I was particularly interested in was Python since I had wanted to learn the language over my vacation.

Participating in this internship or collaboration program with Upskill Campus and utilizing this opportunity will provides a number of benefits. Upskill Campus and The IoT Academy, in partnership with UniConverge Technologies offers internship programs that focus on in-demand skills and technologies in the field of IoT (Internet of Things) and other areas of programming and application design. Interns are ensured that they acquire knowledge and skills that are directly applicable in the industry.

Collaborating with established companies like UniConverge Technologies provides interns with the opportunity to work with experienced professionals and subject matter experts in the field. Interns can learn from their expertise, receive guidance, and gain insights into industry best practices. This exposure can enhance their learning experience and help them develop a deeper understanding of the subject matter.

Having experience with recognized organizations such as the likes of UniConverge Technologies will make your resume stand out and demonstrate your ability to work in a professional setting. It also signifies that you have received training and exposure to industry-specific skills, making you a more attractive candidate to potential employers. Building connections with such industry experts and like-minded individuals, can and will open doors to future career opportunities, mentorship, and professional references.



Over the course of this internship, I gained valuable insight into the real-world applications of Python in areas like Data Science and related topics like Search Engine Optimization and NumPy with its usage and significance. I also gained some very valuable experience in Python in a practical and hands-on approach since I had to develop a project myself using the language, a project that will be reviewed by professionals. I learnt how to implement a functional user interface into an application strictly using code, and not another GUI like in Visual Studio. I also learnt how to embed a functional database in that application. Finally, I got to create, develop and design that very application myself.

To that end I would like to thank Upskill Campus for providing me with the opportunity that made all this possible.

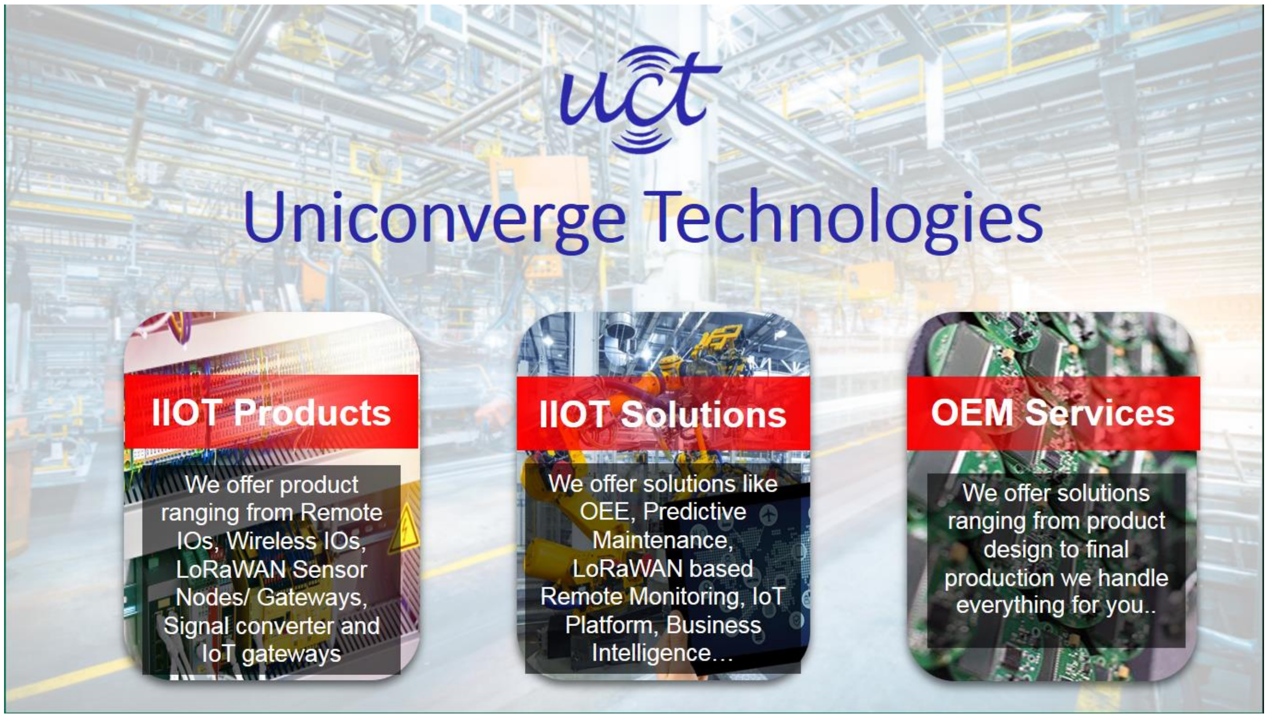
I would also like to thank my friend and fellow intern Joel Abhishek for referring me to Upskill Campus.

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..

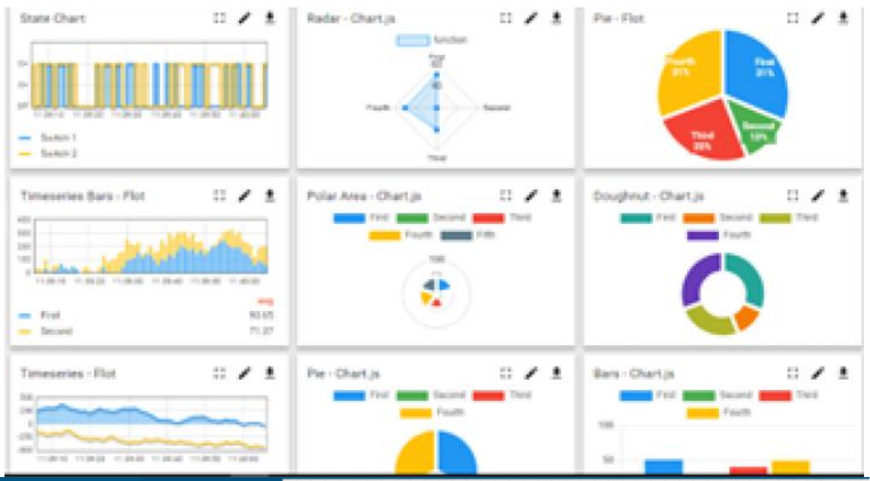
i. UCT IoT Platform (Insight)

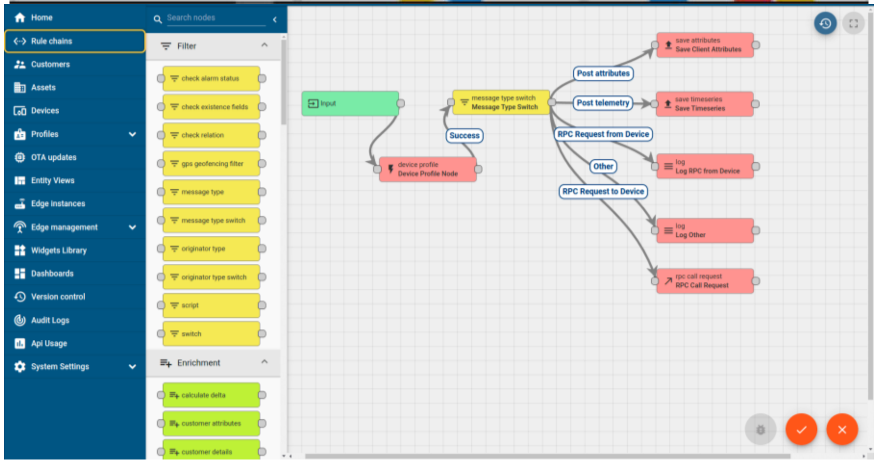
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- 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





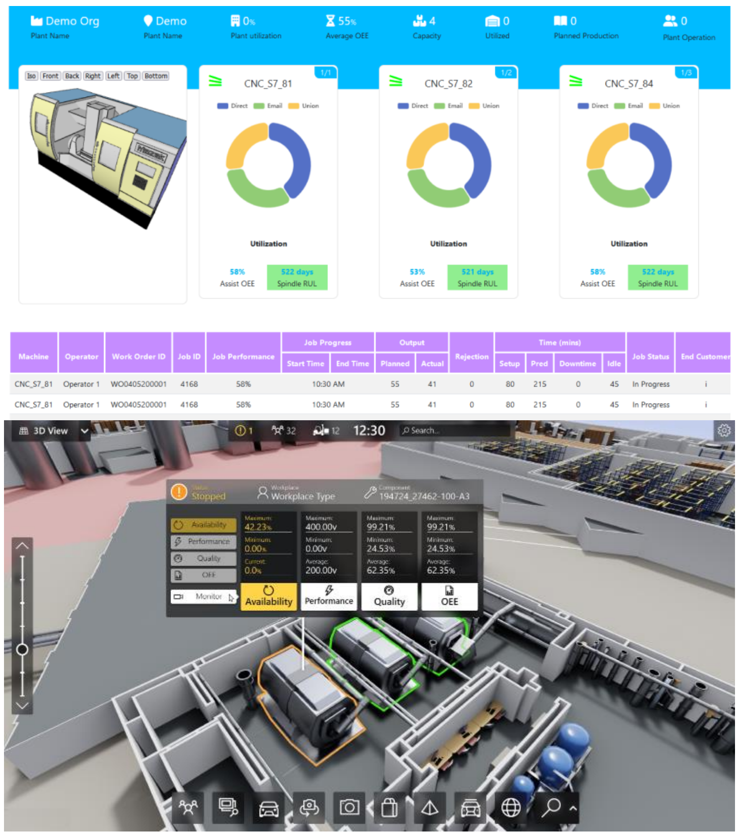
ii. Smart Factory Platform (FACTORY WATCH)

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.



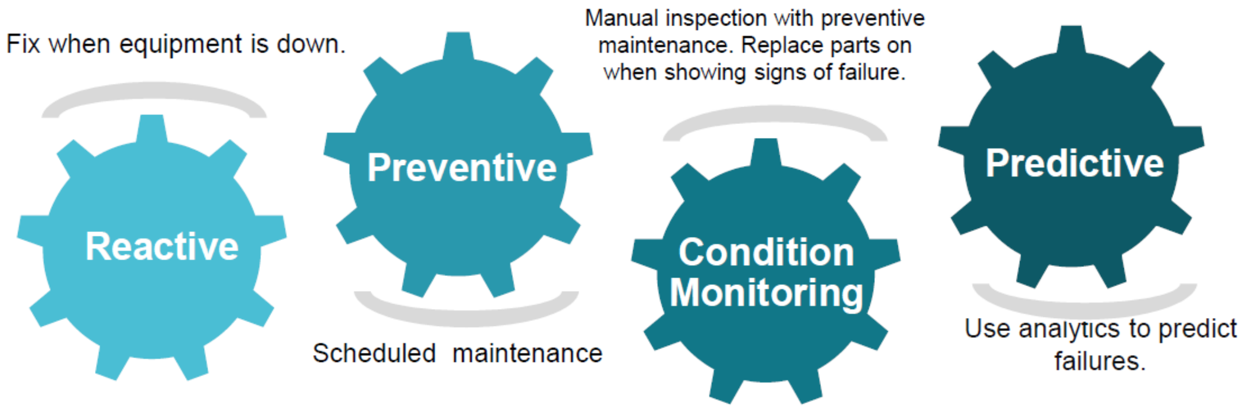
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1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

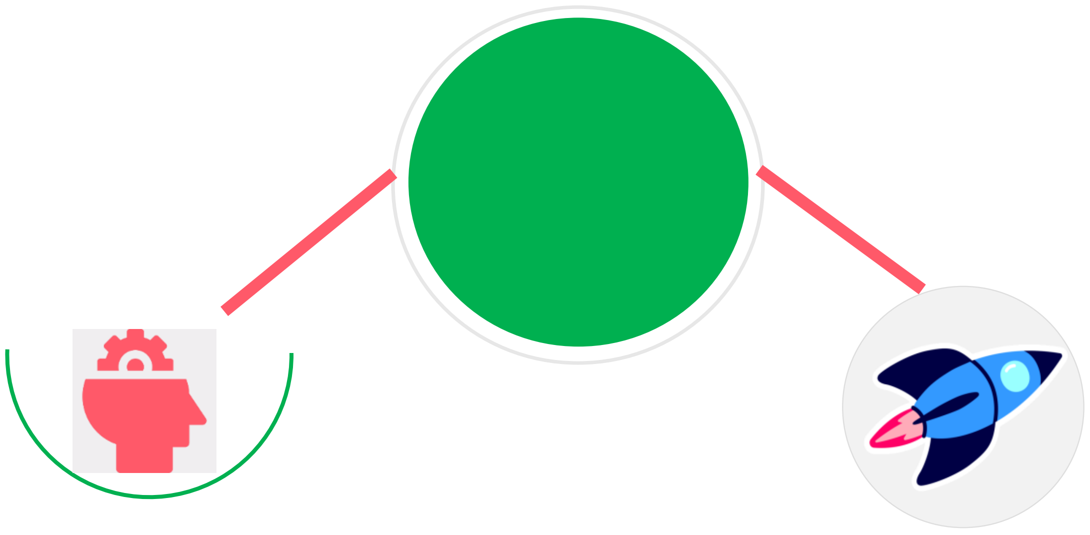
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.


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.

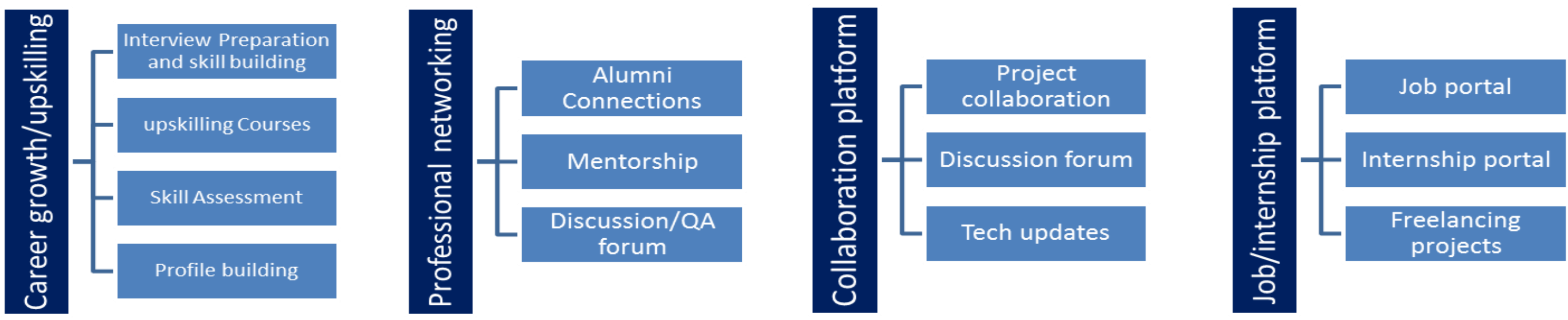
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



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 About 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

- https://medium.com/analytics-vidhya/create-url-shortner-with-python-50129714f044
- https://pypi.org/project/requests/
- https://customtkinter.tomschimansky.com
- https://pyshorteners.readthedocs.io/en/latest/

2.6 Glossary

Terms	**Acronym**
URL	A URL (Uniform Resource Locator) is a unique identifier used to locate a resource on the Internet. It is also referred to as a web address. URLs consist of multiple parts -- including a protocol and domain name -- that tell a web browser how and where to retrieve a resource.
redirection	It refers to the process of forwarding a user from one web address (URL) to another. I**t** is often used for various purposes, including website maintenance, content reorganization, and URL shortening.
PyShorteners	PyShorteners is a Python library that allows developers to integrate URL shortening functionality into their Python applications or scripts without having to deal with the complexities of interacting with each individual URL shortener's API.
API	An API, or an Application Programming Interface, is a set of rules and protocols that allows different software applications to communicate and interact with each other. It defines the methods and data formats that applications can use to request and exchange information.
Tkinter	Tkinter is and has been the standard Python library used to create graphical user interfaces (GUIs). It provides a set of tools and widgets to build windows, dialogs, buttons, menus, and other elements for desktop applications.
CustomTkinter	CustomTkinter is a python UI-library based on Tkinter, that provides new, modern and fully customizable widgets.
requests	The requests library is a popular Python module used for making HTTP requests. It provides a simple and user-friendly API for sending HTTP requests to web servers and handling their responses.
SQL	SQL is short for Structured Query Language. It is a programming language used to manage and manipulate relational databases. SQL allows users to create, retrieve, update, and delete data in databases.
SQLite	SQLite is a self-contained, lightweight, and embedded relational database management system. It implements SQL and provides a C library interface to interact with databases.

3 Problem Statement

I designed and implemented a URL Shortener application in Python that takes long URLs as input and generates shortened versions, making it easier for users to share links in various contexts and situatuions where character count is limited.

The application should address the following key requirements:

- Users should be able to input a long URL, and the application should generate a shortened version of the link. The shortened URL should be unique and relatively short in length to be easily shareable.
- When users access the shortened URL, the application should seamlessly redirect them to the original long URL, ensuring a smooth user experience.
- The application should store the mappings between the original long URLs and their corresponding shortened versions in a database or data store. This ensures that the redirection works correctly, even across application restarts.
- The application should be able to handle various scenarios, such as invalid URLs, and database failures, by providing appropriate error messages.
- The URL Shortener should have a user interface for non-technical users to interact with the application, input long URLs, and view their shortened versions.

The problem statement pertaining to my project would practically describe a scenario in which one wishes to shorten a URL for the purpose of convenience, avoiding overcomplication of documentation, or simply adhering to character limits in certain social media platforms.

4 Existing and Proposed solution

The current existing solution in the programming and developing space right now, in terms of URL Shortening, consists of separate domains online such as bit.ly or owl.ly where you can enter URLs and receive shortened URLs. While they are functional for the most part, they come with a lot of issues that can hinder a user's experience with them, especially a user with specific and highly frequent use cases.

- Using a third-party URL shortener means relying on the service's availability and continued operation. If the service experiences downtime or shuts down, all shortened URLs associated with that service could become inaccessible.
- When using external URL shorteners, the service provider can potentially track the users who click on the shortened links, leading to privacy concerns. Additionally, some users might be hesitant to click on shortened links due to security risks, as they may not know the destination before clicking.
- The performance and speed of redirection for shortened URLs can vary based on the service's infrastructure and server locations. Some services might experience slower redirection times compared to others.
- Certain URL shortening services display ads and ask end-users to wait for certain periods of time when they click on the shortened links, which can be inconvenient for them and negatively impact their experience with the service.
- For users who rely on the URL shortener's API to programmatically generate shortened links, some services might have rate limits or restrictions on the number of requests allowed within a specific time period.
- Some URL shorteners may have limitations on link accessibility based on the user's geographical location, which could affect global reach.
- Additionally, many of these existing URL Shortening services lack any sort of provision that allows users to see the links they shortened in the past. Any services that do have this functionality, most likely reserve it behind a premium subscription.

My proposed solution is to build a URL Shortener entirely in Python, utilizing the SQLite database for storage functionality, interfacing with the TinyURL API through the pyshorteners library for URL shortening, and developing a graphical user interface (GUI) using tkinter.

The following is its key components:

- Backend: The URL Shortener is written entirely in Python, making it platform-independent and easy to maintain, and to further be improved by other Python developers if they wish to do so.
- API Integration: The URL Shortener utilizes the pyshorteners library to interact with the TinyURL API for generating shortened URLs. The TinyURL API offers a simple and reliable way to create short links.
- User Interface: The user interface of the URL Shortened was designed using tkinter, a standard Python library for creating graphical user interfaces, and CustomTkinter, another library of user interface widgets based in Tkinter that aims to improve upon Tkinter's slightly lackluster appearance. The GUI will allow users to input long URLs, generate shortened links, and view the results in a user-friendly manner.
- Database: SQLite serves as a lightweight and embedded database for storing the mappings between the original URLs and their corresponding shortened versions. SQLite provides an efficient and scalable solution for data storage and is ideal for the scale of my project.

Its functionality is as follows:

- URL Shortening: Users can input long URLs through the GUI, and upon clicking the "Shorten" button, the application will interact with the TinyURL API to generate a unique and shortened version of the URL.
- Database Storage: The original long URL and its shortened version will be stored in the SQLite database. This ensures that the mappings persist across application restarts, making the shortened links accessible over time.
- Redirection: When users click on a shortened URL, the TinyURL API will handle the redirection by looking up the original URL in the database and redirecting the user's browser to the corresponding long URL.
- Error Handling: The application will include error handling to deal with potential issues, such as invalid URLs, database failures, or API communication errors. Users will receive appropriate error messages in case of any problems.

As such, these are the clear advantages of my project:

- Full Control: Since I built the URL Shortener myself from scratch, I have complete control over the application's functionalities, security, and user experience.
- Privacy: As the TinyURL API is used for URL shortening, users' privacy concerns are addressed since the actual redirection happens on the TinyURL server. Any information collected by the application itself will only be stored locally on an embedded database.
- Customization: Developing a custom GUI with Tkinter and CustomTkinter allowed me to design a user interface tailored to specific needs and branding.
- Lightweight: Using SQLite as the database provided a lightweight and self-contained storage solution that requires no additional setup.
- Pythonic: The entire application is written in Python, making it easy to maintain and extend by Python developers.

4.1 Code submission (Github link)

https://github.com/ChrisDanielW/UpSkill-URL_Shortener-CARGO/blob/master/CARGO-URL_Shortener.py

4.2 Report submission (Github link)

https://github.com/ChrisDanielW/UpSkill-URL_Shortener-CARGO/blob/master/Report_URL-Shortener_Chris-Daniel-WIlson.pdf

5 Proposed Design/ Model

Some designs and models directly pertaining to my project have been provided below:

5.1 High Level Diagram

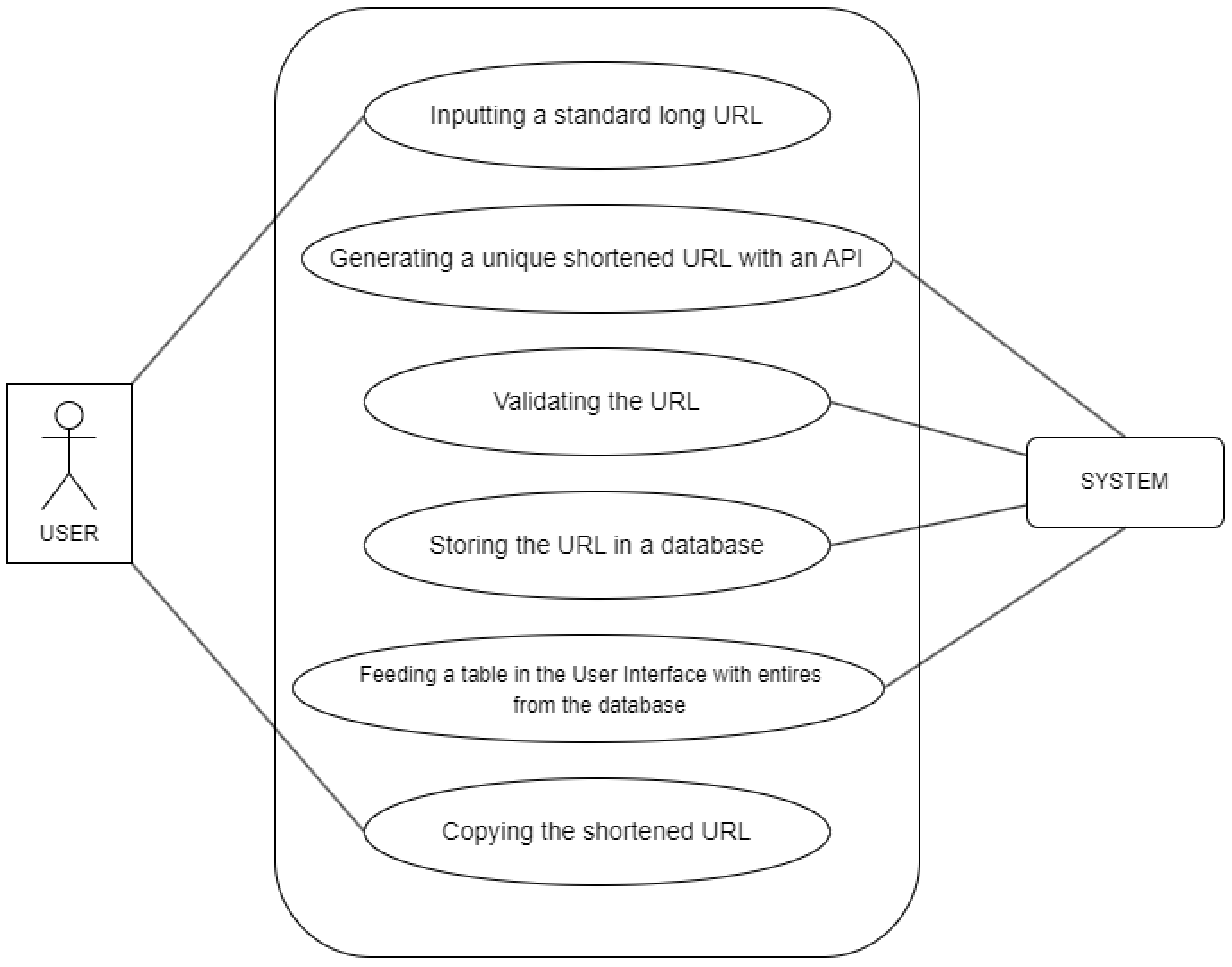

Inputting a standard long URL
Generating a unique shortened URL with an API
Validating the URL
Storing the URL in a database
Feeding a table in the User Interface with entires from the database
Copying the shortened URL
USER
SYSTEM

FIGURE 1: HIGH LEVEL USE CASE DIAGRAM

5.2 Interfaces

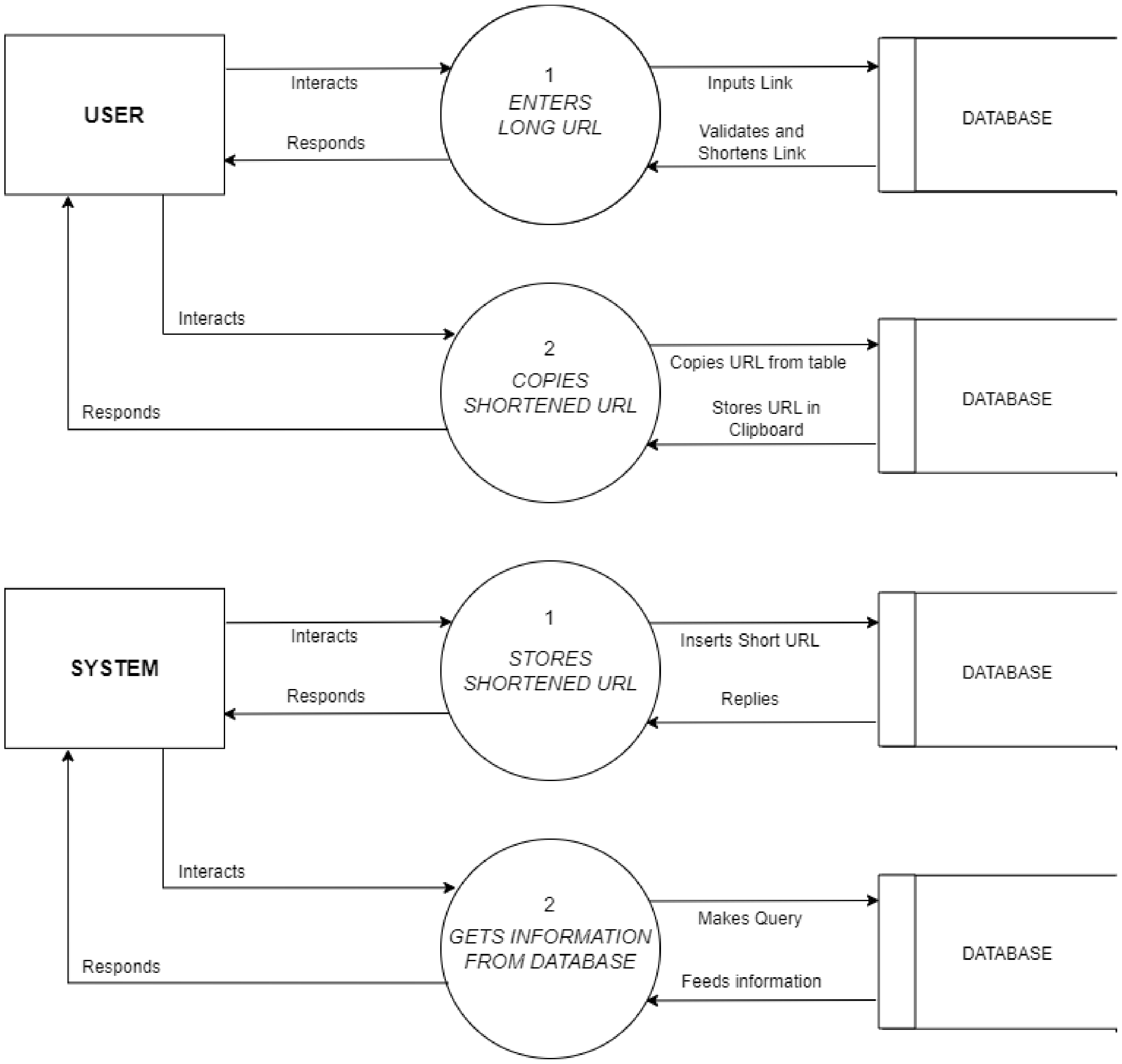


FIGURE 2: LEVEL-1 DATA FLOW DIAGRAM

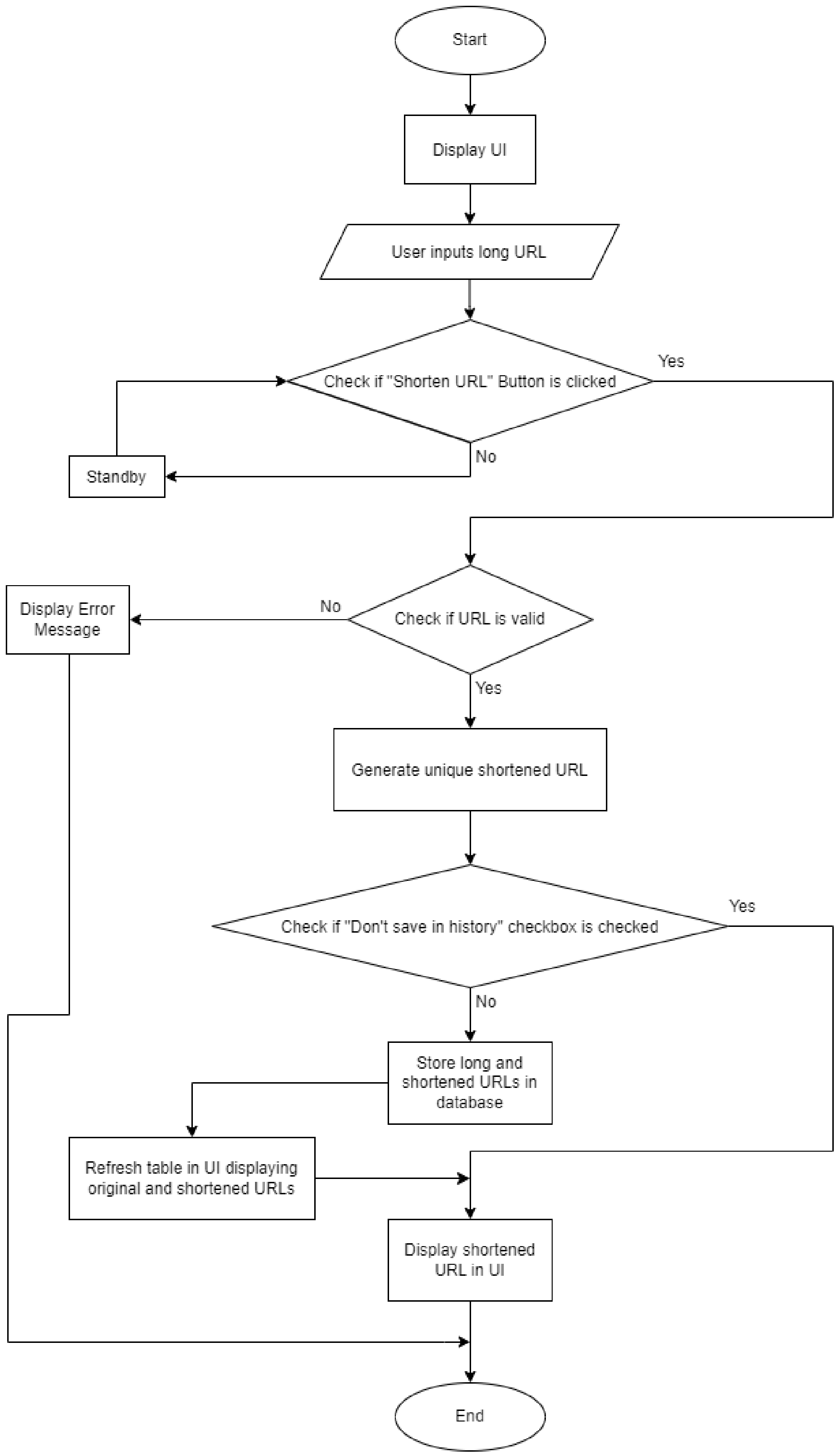

Start
Display UI
User inputs long URL
Check if "Shorten URL" Button is clicked
Yes
No
Standby
No
Check if URL is valid
Display Error Message
Yes
Generate unique shortened URL
Check if "Don't save in history" checkbox is checked
Yes
No
Store long and shortened URLs in database
Refresh table in UI displaying original and shortened URLs
Display shortened URL in UI
End

FIGURE 3: FLOWCHART

6 Performance Test

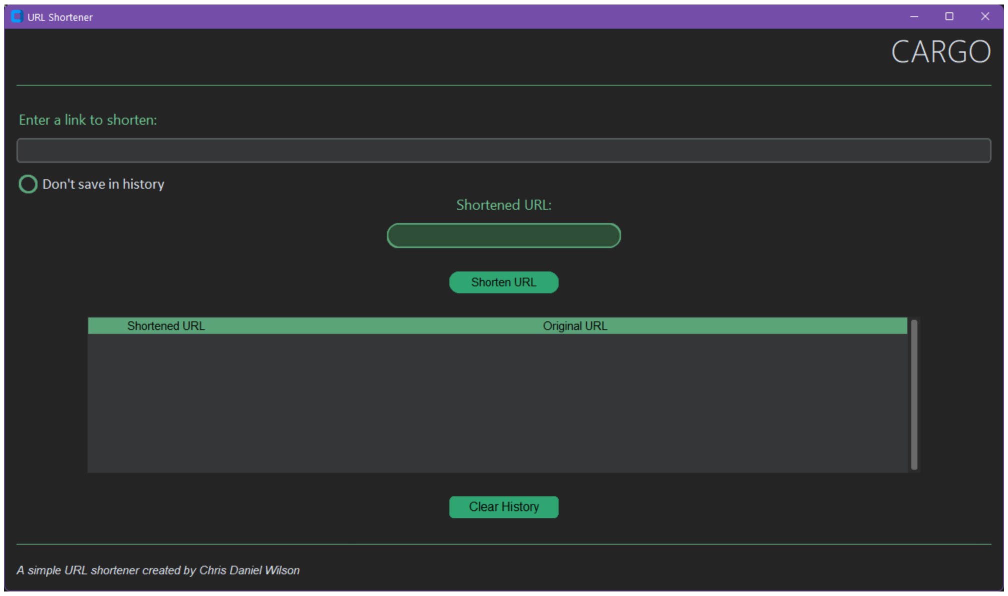
Performance testing holds immense significance in any project, ensuring optimal functionality and an excellent user experience. By subjecting the system to rigorous evaluations, valuable insights are gained into its responsiveness, scalability, and efficiency. Performance testing identifies and eliminates potential bottlenecks, ensuring proper functioning of database storage, API integration, and the user interface.

6.1 Test Cases

The following are some tests and test cases representing all the possible interactions a user could have with the application:

Serial No.	**Test Scenario Name**	**Test Data**	**Expected Result**
1	Nothing-I/P	No input (empty field) is entered and the “Shorten URL” button is clicked.	Error message displaying that an invalid URL was entered.
2	False-URL	Arbitrary string of characters with illegal white-spacing and the “Shorten URL” button is clicked.	Error message displaying that an invalid URL was entered.
3	Invalid-URL	URL entered is expired or incorrectly entered and the “Shorten URL” button is clicked.	Error message displaying that an invalid URL was entered.
4	Double-Short	URL entered is a shortened URL and the “Shorten URL” button is clicked.	Error message displaying that an invalid URL was entered.
5	Correct-URL-Save	A valid URL is entered, the “Don’t save in history” checkbox is unchecked and the “Shorten URL” button is clicked.	The URL is shortened and displayed on the user interface. Both the shortened and original URLs are then stored in the database.
6	Correct-URL-No_Save	A valid URL is entered and the “Don’t save in history” checkbox is checked, and the “Shorten URL” button is clicked.	The URL is shortened and displayed on the user interface. Neither the shortened nor the original URLs are stored in the database.
7	Copy-URL	An entry in the table in the GUI is selected and right-clicked on. The user then clicks the “Copy shortened URL” option.	The shortened URL selected in the table is copied to the clipboard.
8	Clear-DB	The “Clear History” button is clicked.	A dialog box opens and asks the user for confirmation. All entries from the table and database are cleared upon receiving it.
9	Clear-DB-RC	The table is selected and right-clicked on. The user then clicks on the “Clear History” option	A dialog box opens and asks the user for confirmation. All entries from the table and database are cleared upon receiving it.

6.2 Test Procedure



While the procedures for the performing the tests listed above are varied and situation-specific, they collectively involved:

- Launching the Python file titled “CARGO_URL-Shortener.py”.
- Interacting with both the Entry widgets, i.e., the ones for entering the long URL and the one for displaying the shortened URL.
- Interacting with the “Shorten URL” button while keeping the “Don’t save in history” checkbox checked and unchecked respectively.
- Interacting with the Treeview widget that displayed long and shortened URLs directly via a right-click.
- Interacting with the “Clear History” button below the Treeview widget.

Through this stress testing, the system's robustness under extreme conditions is confidently assessed, enabling informed decisions on resource allocation and optimizations. A well-executed performance testing strategy will fine-tune this project, guaranteeing reliability and performance.

6.3 Performance Outcome

The final results and outcomes of all the test cases listed earlier are in the below table

Serial No.	**Test Scenario Name**	**Actual Result**	**Final Outcome**
1	Nothing-I/P	Error message displaying that an invalid URL was entered.	Pass
2	False-URL	Error message displaying that an invalid URL was entered.	Pass
3	Invalid-URL	Error message displaying that an invalid URL was entered.	Pass
4	Double-Short	Error message displaying that an invalid URL was entered.	Pass
5	Correct-URL-Save	The URL is shortened and displayed on the user interface. Both the shortened and original URLs are then stored in the database.	Pass
6	Correct-URL-No_Save	The URL is shortened and displayed on the user interface. Neither the shortened nor the original URLs are stored in the database.	Pass
7	Copy-URL	The shortened URL selected in the table is copied to the cilpboard.	Pass
8	Clear-DB	A dialog box opens and asks the user for confirmation. All entries from the table and database are cleared upon receiving it.	Pass
9	Clear-DB-RC	A dialog box opens and asks the user for confirmation. All entries from the table and database are cleared upon receiving it.	Pass

From these outcomes, it can be concluded that the application has performed desirably and within expectations, causing no program-halting errors or any minor syntax-related issues of the sort.

7 My learnings

During my internship, I had the invaluable opportunity to explore the real-world applications of Python. I delved into diverse areas, such as Data Science, where Python plays a pivotal role in analyzing and interpreting large datasets. I also explored the concept of Search Engine Optimization, understanding how Python can be harnessed to optimize website content and boost search rankings. Additionally, my exploration of NumPy, a powerful numerical computing library, emphasized Python's significance in scientific computing and data manipulation for me.

An area where I experienced significant growth was in developing functional user interfaces from scratch, exclusively through code and not through other GUIs like I did previously in Visual Basic. This aspect of the internship sharpened my skills in crafting intuitive and user-friendly interfaces that cater to end-users' needs

Another aspect that enriched my learning journey was the implementation of a fully functional database within the application. Integrating the database provided insights into the intricacies of data management and storage, crucial aspects of any real-world software application. It was really satisfying, personally, to see how a well-designed database could locally facilitate efficient data retrieval and manipulation, enhancing the overall user experience.

A defining aspect of my internship was the hands-on experience I gained in Python. The practical approach allowed me to grasp Python's nuances, enhancing my problem-solving abilities and coding proficiency. This was a wonderful opportunity to develop a real-world project using Python, a task that required creativity, critical thinking, and meticulous planning. The process not only enhanced my technical prowess but also instilled in me a sense of ownership and accomplishment. As I iteratively refined the application, I grew more confident in my abilities as a Python developer, ready to take on future challenges with enthusiasm and expertise.

In conclusion, my internship journey provided me with an immersive experience in the world of Python, its real-world applications, and its potential to transform ideas into functional software solutions. The combination of theoretical knowledge and practical application allowed me to grasp Python's usefulness and significance in various domains.

8 Future work scope

Due to technical limitations and a lack of sufficient time, I unfortunately wasn't able to implement a few features that I wanted to add to my final product, such as:

1. A better table implementation for displaying original and shortened URLs: I did manage to design a functioning Treeview Widget that acted as a live representation of the all the data in the database, however, I still feel that it could be improved upon. For instance, the widget as it is right now does not scale to the edges for the frame like many of the other widgets when the main window's size is changed. It is also not possible to select any single cell, with any all selections from the user's size being limited to the entire row of the cell they click on.

2. The option to choose from multiple APIs for URL Shortening: The project currently uses the TinyURL API to shorten URLs, exclusively. Making a provision to choose from multiple APIs was slightly tedious given the fact that most other APIs needed dedicated API keys to be used. If this issue was addressed, the user could have had more control and flexibility over the services they preferred using to shorten their URLs.

3. Customization for URLs: TinyURL and other APIs offer users the ability to name the shortened URLs they create, allowing for more personalization in the URLs generated and for the removal of the ambiguity that comes with the arbitrary names assigned to shortened URLs. The ignorance of this feature was an oversight on my end, and given more time, I could implement it into my project along with the choice to use other APIs.

4. Estimated expiration dates: Shortened URLs generated in some services will only work as intended for a certain period of time. Knowing this window of operation will greatly benefit most users, giving them some foresight as to when they should stop using those URLs.

5. Dedicated dark and light themes: While most people today frequent dark themes in the applications they use, there is still a good percentage of users who use light themes for various reasons like personal preference or eyesight issues. My project defaults to a dark theme and lacks any option for a brighter color scheme, since changing the theme of the application will require a function that changes the colors of all almost the widgets.

All of the above features could be added to my project in a future update, should I receive the time and resources for it.