PROFESSIONAL TRAINING REPORT

entitled

Intuitive Contact Manager: A Tkinter-Based Graphical Phonebook

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering with specialization in Internet Of Things

by

SHAIK MAHAMMAD THOFIQ

Reg.No.41732022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A++" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI – 600119

OCTOBER 2023

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with A++ Grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai,
Chennai – 600 119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training is the bonafide work of **Mr. SHAIK MAHAMMAD THOFIQ, Reg.No.41732022** who carried out the project entitled **'Intuitive Contact Manager: A Tkinter-Based Graphical Phonebook'** under my supervision from June 2023 to October 2023.

Internal Guide
Dr.D.Menaka,

Head of the Department
Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner **External Examiner**

DECLARATION

I, **SHAIK MAHAMMAD THOFIQ, Reg.No.41732022,** hereby declare that the Professional Training Report-I entitled **'Intuitive Contact Manager: A Tkinter-Based Graphical Phonebook'** done by me under the guidance of **Dr.D.Menaka,**is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering with specialization in Internet Of Things.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D.**, **Dean**, School of Computing, **Dr. S.Vigneshwari M.E., Ph.D., Head of the Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Internal Guide **Dr.D.Menaka,**for his/her valuable guidance, suggestions and constant encouragement which paved way for the successful completion of my phase-1 professional Training.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

COURSE CERTIFICATE

MORNING SESSION (FN)	National Programme on Technology Enhanced Learning	NPTEL
Hall Ticket For	Programming, Data Structures And Algorithms Using Python - Online	

Candidate Name	SHAIK MAHAMMAD THOFIQ				
Roll No	NOC23CS95S15332027				
Date of Birth	22-06-2003				
PwD Status	No	Compensatory Time Required	N.A	Scribe Required	N.A
Exam Date	Saturday, 23 September, 2023				
Reporting Time	08:00 am	Gate Closure	09.30 am		
Exam Timing	09:00 am	Shift	FN		
Test Centre Name	iON Digital Zone iDZ Kundrathur				
Test Centre Address	Chennai Institute of Technology, 2nd Floor, iON Digital Zone, Sarathy Nagar, Kundrathur, Chennai, Tamil Nadu, India - 600069				

NPTEL Coordinator

NPTEL EXAM - 23 SEPTEMBER, 2023
General instructions for candidates - FN
(All timings mentioned here are in IST)

DRESS CODE: Candidates are expected to come in professional attire to write the exams. Candidates wearing SHORTS will NOT be permitted inside the exam hall

AT THE EXAM CENTRE, IF YOU ENCOUNTER ANY ISSUES WITH RESPECT TO THE COMPUTER OR EXAM OFFICIALS, KINDLY CONTACT THE NPTEL EXAM REPRESENTATIVE, WHO WILL BE AVAILABLE AT THE CENTRE.

1. The Hall Ticket must be presented for verification along with one original photo identification (not photocopy or scanned copy). Examples of acceptable photo identification documents are School ID, College ID, Employee ID, Driving License, Passport, PAN card, Voter ID, Aadhaar-ID. Printed copy of the hall ticket and original photo id card should be brought to the exam centre. Hall ticket and id card copies on the phone will not be permitted.
2. This Hall Ticket is valid only if the candidate's photograph and signature images are legible. To ensure this, print the Hall Ticket on A4 sized paper using a laser printer, preferably a colour photo printer.
3. Please report to the examination venue by 08:00 am; CANDIDATES WILL NOT BE ALLOWED TO ENTER THE EXAMINATION HALL AFTER 09.30am.
4. Candidates will be permitted to appear for the examination ONLY after their credentials are verified by center officials.

P.T.O.

ABSTRACT

In a digital age where efficient contact management is crucial, the Python-based Phone Book application presented here stands out as a powerful and user-friendly solution. This advanced phone book offers several features that make it a valuable tool for users seeking a seamless and intuitive way to manage their contacts.

One of the key strengths of this application is its robustness in preventing data redundancy. Unlike conventional phone books, it employs intelligent data validation checks to ensure that no two contacts share the same phone number. This unique feature prevents clutter and confusion, making it exceptionally reliable for users.

Furthermore, the application excels in its ability to handle contact entries with a higher level of granularity. Users can not only input essential details like first name, last name, and phone number but also enjoy the freedom to add and update contact information effortlessly. This makes it adaptable to a variety of contact management scenarios, from personal to professional use.

Moreover, the phone book offers a convenient search functionality, empowering users to quickly locate specific contacts by either their name or phone number. This time-saving feature enhances productivity, especially in scenarios where time is of the essence.

Visually, the application is designed with user experience in mind. Its modern and intuitive Graphical User Interface (GUI) ensures that users can navigate the features effortlessly. The interface's clean and appealing layout contributes to a pleasant and efficient user interaction.

Overall, this Phone Book application sets itself apart as an advanced and indispensable tool for contact management. Its robustness in preventing data duplication, flexible data entry capabilities, efficient search functionality, and user-friendly interface collectively position it as a standout choice for users seeking a hassle-free and effective means of managing their contacts. In comparison to traditional phone books, it excels in both functionality and user experience, making it a valuable asset for modern users.

CHAPTER NO.	TABLE OF CONTENTS	PAGE NO.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 WHAT IS PHONEBOOK

A phonebook, in its simplest form, is a directory or a physical book that contains a list of names, addresses, and phone numbers of individuals, businesses, and organizations. Its main purpose is to help people find and contact others quickly and easily. Phonebooks have been an integral part of our daily lives for decades, serving as an essential reference for connecting with friends, family, colleagues, and various services.

Traditionally, phonebooks were printed on paper and distributed to households and businesses. People would flip through its pages to locate the contact information they needed. These physical phonebooks were often categorized by alphabetical order or geographical location, making it convenient to search for specific entries. In many ways, they were a symbol of communication and connectivity in a pre-digital era.

1.2 WHAT IS DIGITAL PHONEBOOK

In programming, a phonebook is a digital application or system designed to store and manage contact information, such as names, phone numbers, email addresses, and other relevant details. Much like its real-world counterpart, a digital phonebook acts as a centralized repository for organizing and retrieving contact information for individuals, businesses, or entities.

The primary purpose of a programming phonebook is to provide users with a convenient and efficient way to store, search, and access their contacts' details. Users can typically add, edit, and delete entries, making it easy to maintain an up-to-date list of contacts.

Phonebooks in programming often incorporate advanced features and functionalities, such as a graphical user interface (GUI) to enhance the user experience. A GUI allows users to interact visually with the application, making it more user-friendly and accessible. Users can navigate through the contact list, perform searches, and perform actions like adding or updating contacts with ease.

1.3 How Does Digital PHONEBOOK Works?

A high-end programming phonebook leverages advanced technology to offer a seamless and feature-rich user experience. Here's how it typically works:

1.3.1 Graphical User Interface (GUI):

Unlike traditional phonebooks, high-end programming phonebooks offer a sophisticated Graphical User Interface (GUI). This GUI provides an interactive and visually appealing platform for users to interact with the phonebook. It includes input fields, buttons, lists, and search bars, making it user-friendly and easy to navigate.

1.3.2 Data Storage:

The phonebook stores contact information in a structured database. This database can be a file, a cloud-based service, or a dedicated server. It efficiently manages large volumes of data, including names, phone numbers, addresses, and additional details.

1.3.3 Data Retrieval:

Users can search for contacts using various criteria, such as name, phone number, or address. The phonebook employs advanced search algorithms to quickly retrieve relevant results, ensuring that users find the information they need without delay.

1.3.4 Error Handling:

High-end phonebooks are equipped with robust error-handling mechanisms. They validate user input to prevent common mistakes and inconsistencies. For instance, they may check for duplicate entries or missing information and provide informative error messages to guide users.

1.3.5 User Interaction:

Users can add, update, or delete contacts seamlessly through the GUI. The phonebook validates user inputs, allowing only valid and properly formatted data to be stored. It also offers features like batch import/export to manage contacts efficiently.

CHAPTER 2

LITERATURE REVIEW

2.1 AIM:

To develop a user-friendly Python GUI phonebook for easy contact management.

2.2 SCOPE:

The scope of this project extends beyond traditional phonebook applications by leveraging the power of Python and GUI development. This phonebook aims to offer an intuitive and visually appealing interface for managing contacts. Users can add, update, delete, search, and load contact details effortlessly. Additionally, the phonebook will implement features to prevent duplicate entries, ensuring data accuracy. It will also be compatible with CSV file storage for seamless data retention. This project's scope is not only to create a basic phonebook but to provide an advanced, user-centric, and efficient tool for organizing contacts, setting it apart from conventional phonebook applications.

CHAPTER 3

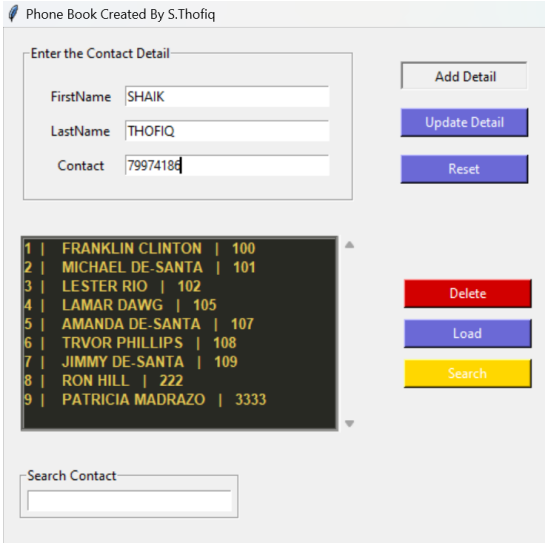
REQUIREMENTS ANALYSIS

3.1 OBJECTIVE OF THE PROJECT:

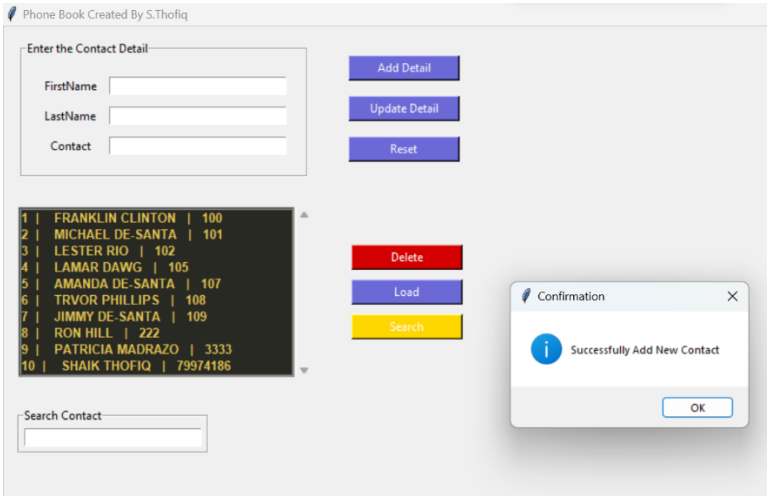
The objective of the above phonebook code is to develop a feature-rich, user-friendly, and efficient phonebook application using Python's graphical user interface (GUI) capabilities. This application aims to simplify contact management by allowing users to:

3.1.1 Add Contacts:

Users can easily add new contacts, providing both first names, last names, and contact numbers.



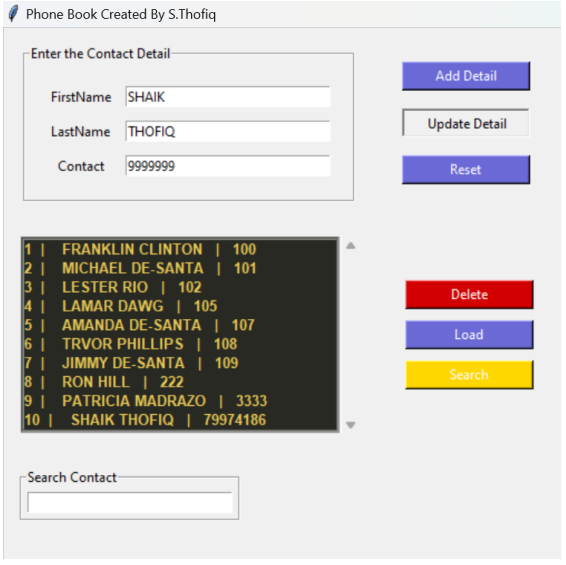
FIG(1): Before Adding New Contact



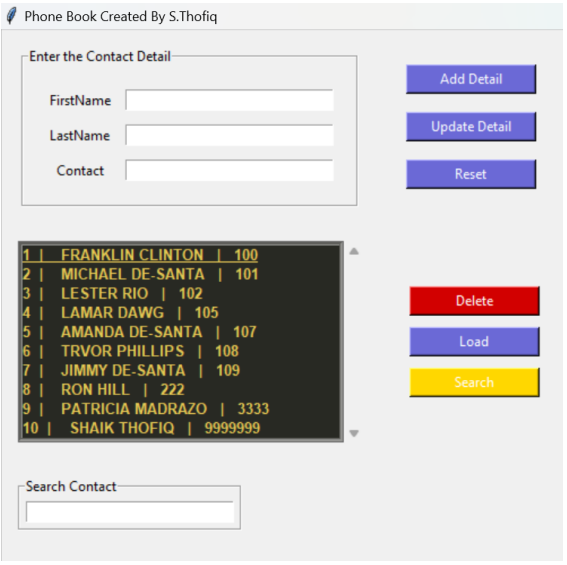
FIG(2): After Adding New Contact

3.1.2 Update Contacts:

The phonebook facilitates updating contact details, ensuring that users can maintain accurate and up-to-date information.



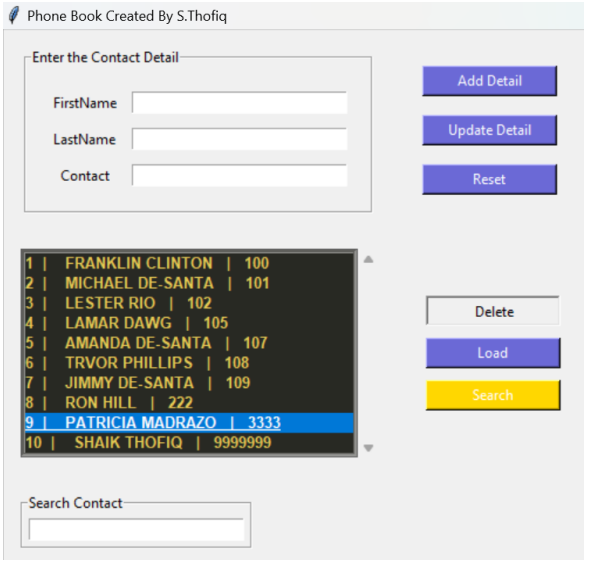
FIG(3): Before Updating Existing Contact



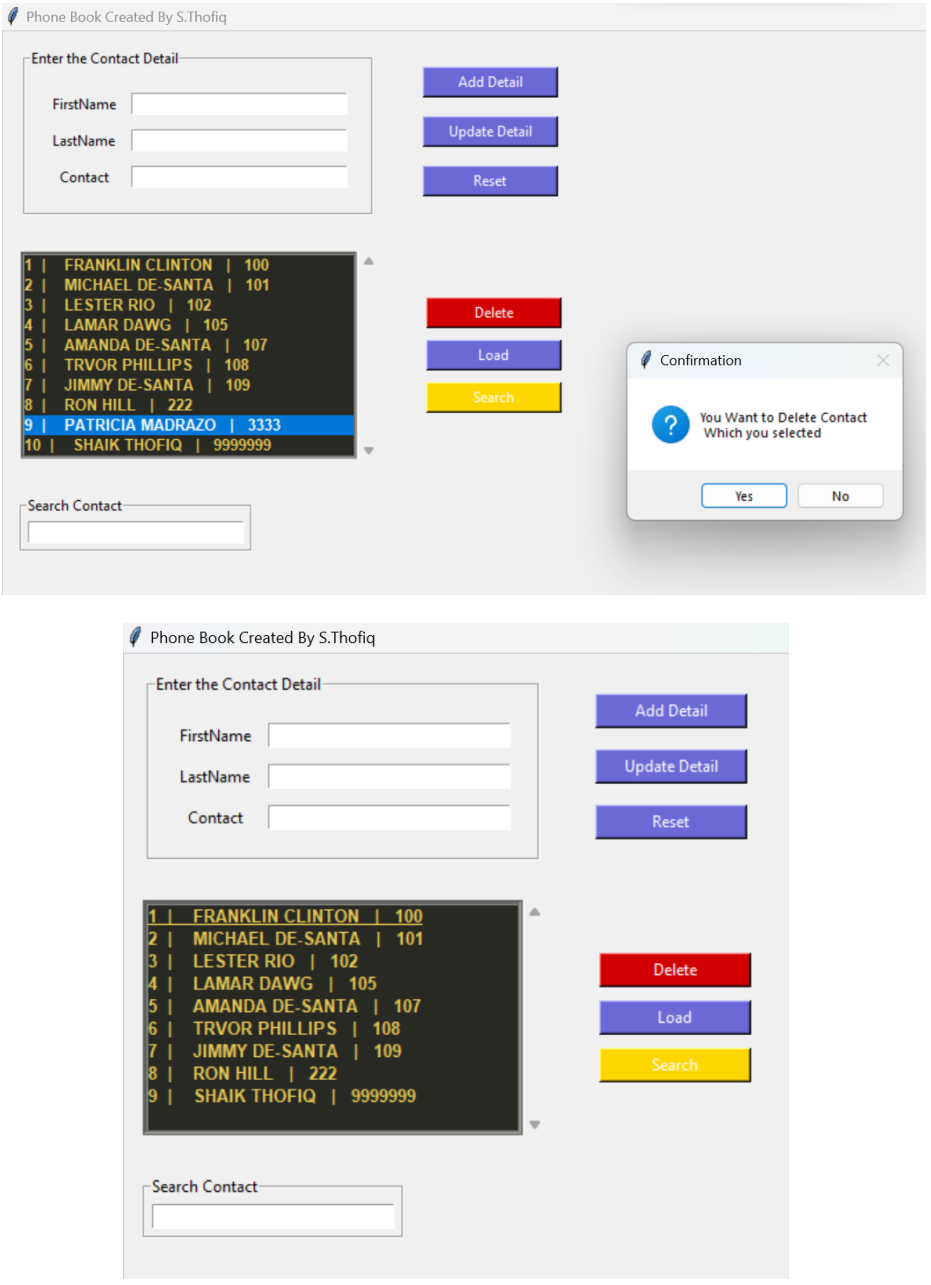
FIG(4): After Updating Existing Contact

3.1.3 Delete Contacts:

Users have the ability to delete unwanted contacts with a simple click.



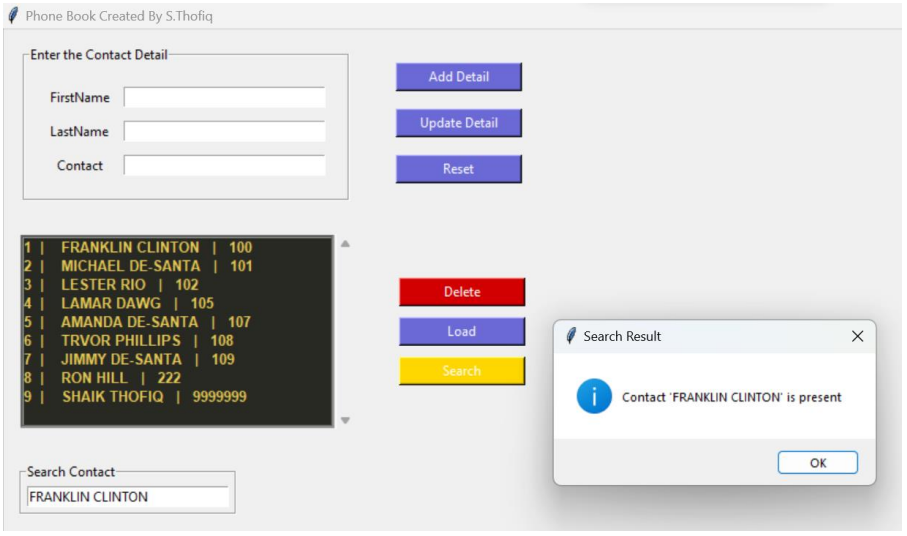
FIG(5): Before Deleting Existing Contact



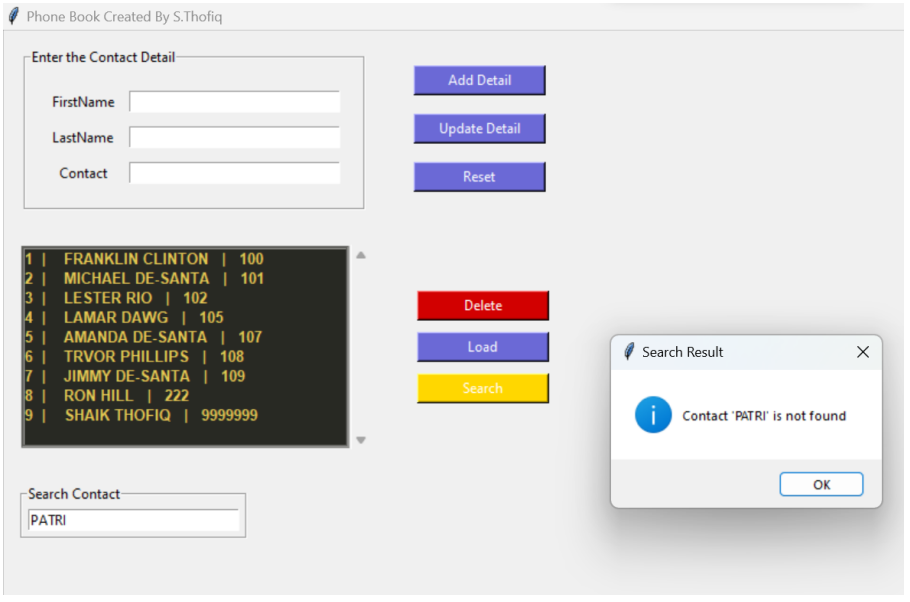
FIG(6): After Deleting Existing Contact

3.1.4 Search Contacts:

The application provides a search functionality, enabling users to find contacts quickly by name or phone number.



FIG(7): Search Succes



FIG(8): Search Failure

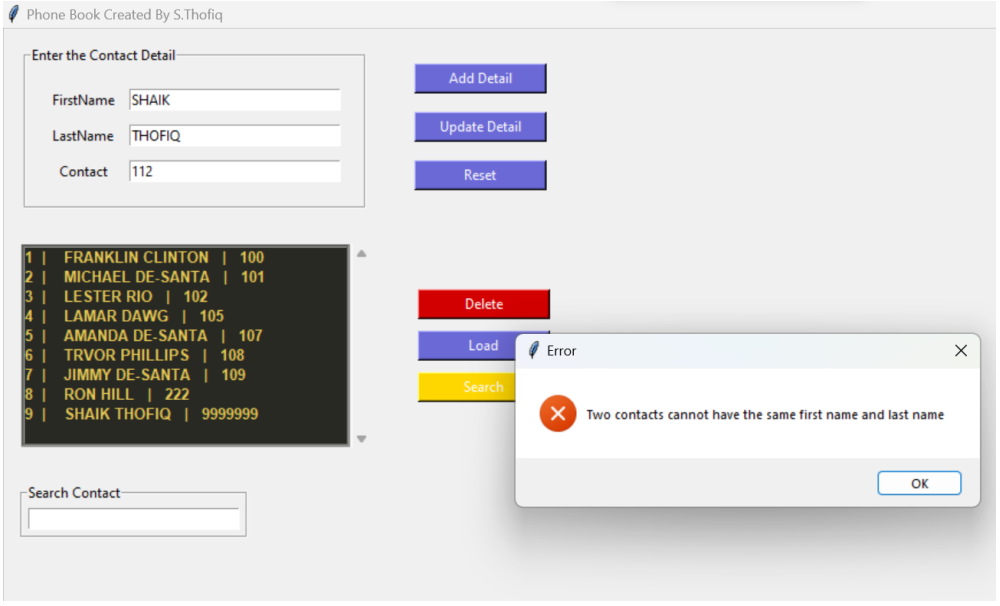
3.1.5 Load Contacts:

Users can load selected contacts' information into the input fields for editing or viewing.

FIG(9): Load Contact

3.1.6 Prevent Duplicates:

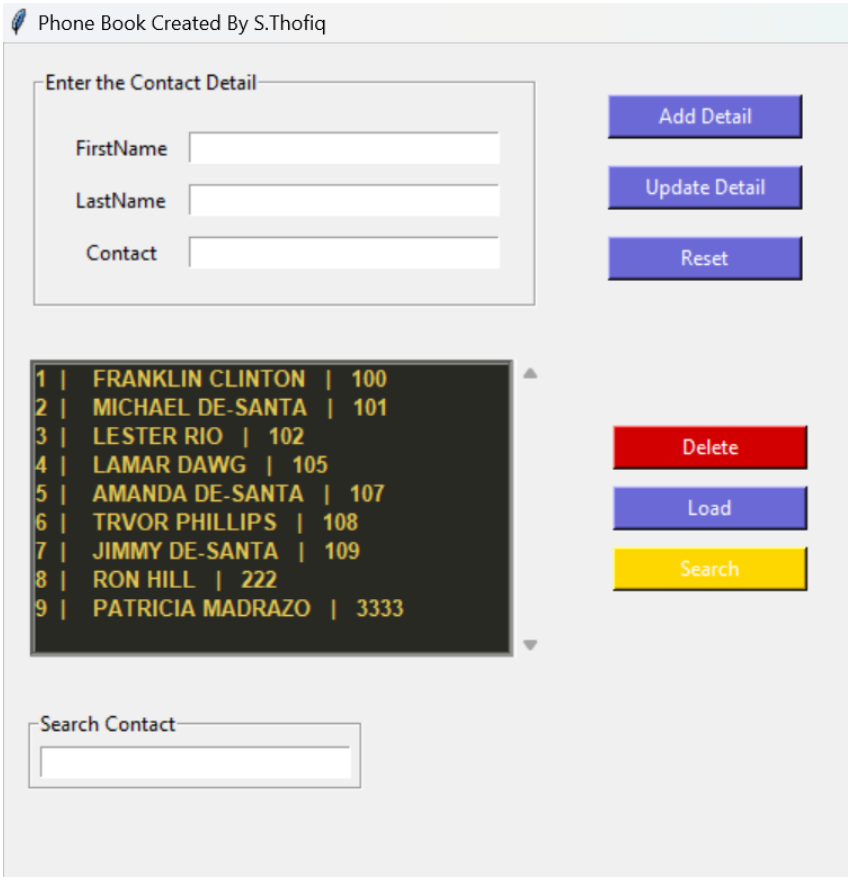
The phonebook includes checks to prevent duplicate contact numbers and names, enhancing data integrity.



FIG(10): Duplicate Handling

3.1.7Visual Appeal:

It offers an attractive and intuitive GUI, making it easy for users to navigate and interact with the application.



FIG(11): User Interface

3.2 REQUIREMENTS

The requirements for the phonebook project provided in the code are as follows:

1)Python:

The project requires Python, a programming language used to develop the phonebook application.

2) Tkinter Library:

Tkinter is a standard GUI library for Python. It is used for creating the graphical user interface of the phonebook application.

3) CSV File:

The phonebook stores contact data in a CSV (Comma-Separated Values) file format. The application should have read and write access to this file for data storage and retrieval.

4) User Input:

The application requires user input for adding, updating, and searching for contacts. It includes fields for first name, last name, and contact number.

5) Graphical Elements:

The GUI elements, including labels, entry fields, buttons, and frames, are used to create an appealing and functional user interface.

3.2.1 HARDWARE REQUIREMENTS:

The phonebook project described in the provided code has minimal hardware requirements since it's a simple desktop application. Here are the basic hardware requirements:

1) Computer:

You'll need a desktop or laptop computer to run the Python code and the graphical user interface (GUI) of the phonebook application.

2) Storage:

You need enough storage space to store the Python script and the CSV data file. Since the application deals with a small dataset, storage requirements are minimal..

3) Processor (CPU):

A standard modern CPU is sufficient, and the project does not require high processing power.

3.2.2 SOFTWARE REQUIREMENTS:

The phonebook project described in the provided code has some software requirements that you should meet to run the application successfully. Here are the software requirements:

1) Python:

You need Python installed on your computer. You can download Python from the official website (https://www.python.org/downloads/) and ensure that it's added to your system's PATH during installation.

2) Python Libraries:

The code uses several Python libraries for GUI development and working with CSV files.

You may need to install these libraries if they are not already present on your system.

You can install them using the Python package manager, pip, with the following command:

```
pip install tkinter
```

The ‘tkinter’ library is part of the standard Python library for creating graphical user interfaces. You may not need to install it separately on most systems.

3) Operating System:

The code is compatible with various operating systems, including Windows, macOS, and Linux. Ensure that you have a supported operating system installed.

4) Integrated Development Environment (IDE):

You can use a text editor or a Python-specific IDE like Visual Studio Code, PyCharm, or IDLE to edit and run the Python code.

5) CSV File:

The code reads and writes data to a CSV file named 'StudentData.csv.' Ensure that this file exists in the same directory as the Python script or update the file path accordingly.

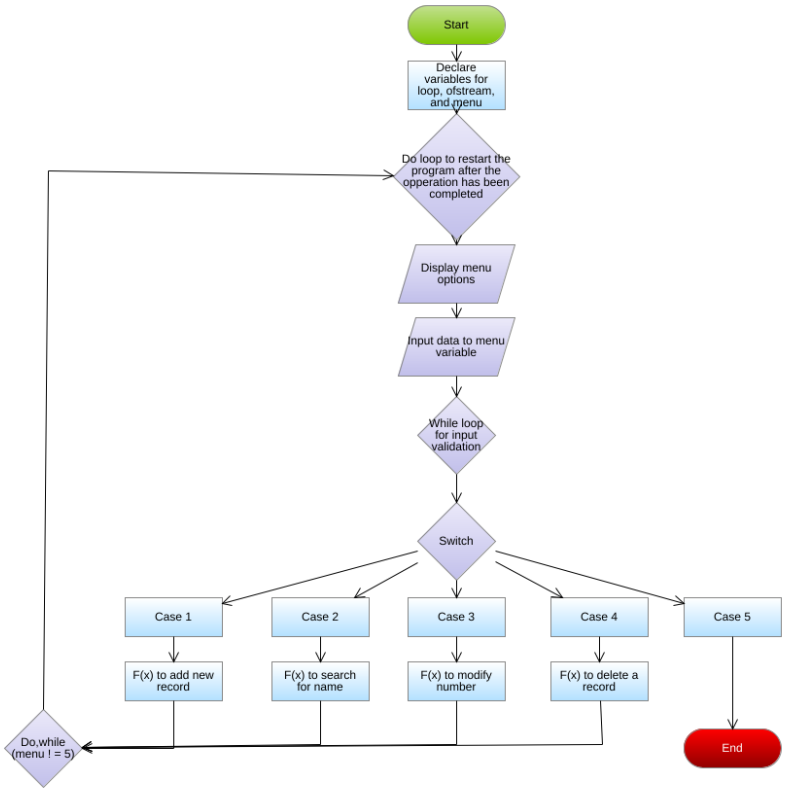
CHAPTER 4

DESIGN DESCRIPTION OF PROPOSED PROJECT

4.1 PROPOSED METHODOLOGY

The provided Python code creates a phonebook application with a graphical user interface (GUI) using the tkinter library. The application allows users to manage contact details stored in a CSV file. It follows a structured methodology: It reads data from the CSV file, displays it in a listbox, and offers functionalities such as adding, updating, and deleting contacts. When adding or updating a contact, the code checks for duplicate entries based on contact numbers and first/last name combinations. It also includes a search feature to find specific contacts. The GUI provides a user-friendly interface for interacting with the phonebook, while the code ensures data integrity by preventing duplicates and offers feedback to users through message boxes.

4.1.1 Ideation Map/System Architecture



FIG(12): WORKFLOW

4.1.2 Various Stages

1) Imports and Initialization:

The code begins by importing the necessary modules: ‘tkinter’ for GUI components, csv for handling CSV files, and ‘messagebox’ for displaying message boxes. An empty list named ‘phonelist’ is created to store contact information.

2) Reading CSV File (ReadCSVFile Function):

The ReadCSVFile function is responsible for reading data from a CSV file called 'StudentData.csv.'It uses the 'csv.reader' to open and read the CSV file.The first row of the CSV file (usually containing headers) is read as 'header'.The contact details from the CSV file are read and appended to the 'phonelist'.The 'set_select' function is called to populate the listbox in the GUI with the loaded contacts.The loaded 'phonelist' is printed for reference.

3) Writing to CSV File (WriteInCSVFile Function):

The 'WriteInCSVFile' function handles writing data back to the CSV file.It opens the 'StudentData.csv' file for writing using 'csv.writer'.The 'header' (previously read from the file) is written as the first row in the CSV.Then, it iterates through the 'phonelist' and writes each contact's information as a row in the CSV.

4) WhichSelected Function:

The 'WhichSelected' function checks which item is selected in the listbox (select).If no item is selected (length of selection is zero), it displays an error message asking the user to select a name.If an item is selected, it returns the index of the selected item.

5) AddDetail Function:

The 'AddDetail' function adds a new contact to the phonebook. It retrieves the first name, last name, and contact number entered by the user from the GUI.It checks if all three fields are filled; if not, it displays an error message.It then checks for duplicate entries based on contact numbers and first/last name combinations. If a duplicate is found, it displays an error message.If no errors are encountered, the new contact is added to phonelist, written to the CSV file, and the GUI is updated.A confirmation message is shown upon successful addition.

6) UpdateDetail Function:

The 'UpdateDetail' function allows the user to update an existing contact.It retrieves the updated first name, last name, and contact number from the GUI.It checks if all three fields are filled, and if not, it displays an error message.It checks for errors related to the selection or missing information and displays appropriate messages.If no errors are encountered, it updates the contact in 'phonelist', writes the changes to the CSV file, and displays a confirmation message.

7) EntryReset Function:

The 'EntryReset' function clears the input fields (first name, last name, and contact number) in the GUI.

8) DeleteEntry Function:

The 'DeleteEntry' function allows the user to delete a selected contact.It asks for confirmation before deleting.If confirmed, it removes the contact from 'phonelist', updates the CSV file, and refreshes the GUI.

9) LoadEntry Function:

The 'LoadEntry' function loads the details of a selected contact into the input fields for editing.

10) set_select Function:

The 'set_select' function populates the listbox in the GUI with contacts from 'phonelist', sorted by contact numbers.

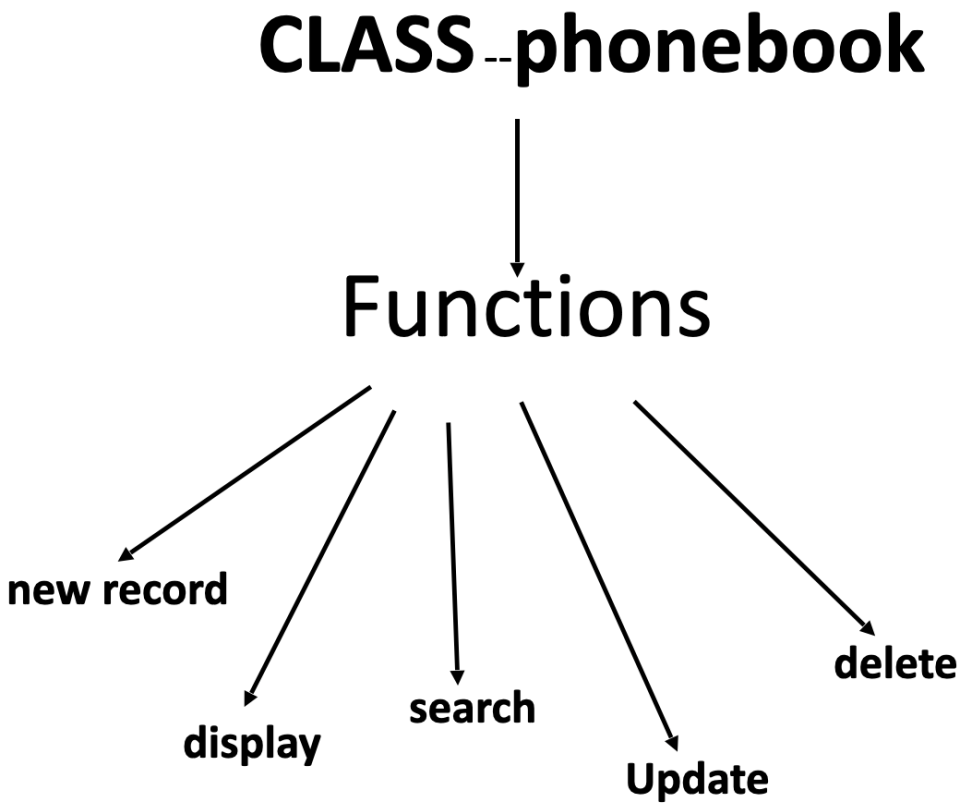
11) SearchContact Function:

The 'SearchContact' function enables users to search for specific contacts.It retrieves the search query from the GUI and performs a case-insensitive search through the 'phonelist'.If a contact matching the search query is found, it displays a message indicating its presence; otherwise, it informs the user that the contact was not found.

12) GUI Layout and Main Loop:

The code creates a graphical user interface (GUI) with various widgets like labels, entry fields, buttons, and a listbox to display contacts.The layout and functionality of the GUI allow users to interact with the phonebook application.The main loop ('window.mainloop()') keeps the GUI responsive and functional.

4.1.3 Internal or Component design structure



4.2 FEATURES

1)CSV Data Handling

2)Graphical User Interface (GUI)

3)Add New Contacts

4)Update Contacts

5)Delete Contacts

6)Search Functionality

7)Data Sorting

8)Confirmation Messages

9)Data Integrity

10)Reset Entry Fields

11)Responsive GUI

12)Load Contacts on Startup

4.2.1 Novelty of the proposal

The phonebook project, based on the provided code, introduces several novel features that enhance its utility and usability. One of the key aspects is the incorporation of a Graphical User Interface (GUI) using the Tkinter library, which significantly improves user-friendliness. This GUI provides an intuitive and visually appealing platform for managing contact information, making it accessible to a wide range of users, including those with minimal technical expertise.

Additionally, the code employs robust CSV data handling capabilities, allowing users to efficiently manage and persist their contact data. This feature facilitates easy import and export of contacts, ensuring that users can back up their information or switch to other applications seamlessly.One of the most significant advantages of this phonebook is its comprehensive functionality. Users can add, update, and delete contacts effortlessly. The addition of a search functionality simplifies the process of finding specific contacts by name or phone number, a feature that can save significant time when dealing with a large contact list.

Moreover, the code ensures data integrity through data validation checks. It prevents the addition of duplicate contacts based on both phone numbers and names, safeguarding the accuracy of the contact list. Users are provided with confirmation messages upon successful contact management actions, adding to the overall user experience.The phonebook's responsive GUI design ensures smooth navigation and interaction, further enhancing its usability. Contacts are automatically loaded from a CSV file upon startup, guaranteeing that users have immediate access to their data. In cases of incomplete or incorrect input, the code includes error messages to guide users, making the application more user-friendly.this phonebook project not only introduces novel features but also leverages these features to provide a highly useful tool for managing and organizing contact information.

CHAPTER 5

CONCLUSION

In conclusion, the phonebook project, based on the provided code, represents a significant leap in contact management and organization. Its novelty lies in its ability to seamlessly blend a Graphical User Interface (GUI) with robust data handling, making it a versatile and user-friendly tool. This phonebook isn't just a digital address book; it's a comprehensive solution that caters to a myriad of user needs and surpasses conventional phonebook applications in several ways.

The GUI, created using the Tkinter library, stands out as a testament to modern software development practices. It offers an attractive, intuitive, and accessible interface that simplifies the task of managing contacts. Unlike traditional phonebooks, this project doesn't require users to navigate complex menus or commands. Instead, it provides a visually appealing platform that is inviting to users of all technical backgrounds.

The core functionality of this phonebook is where it truly shines. Users can effortlessly add, update, and delete contacts. With a few clicks, you can organize your entire contact list, ensuring it remains up-to-date and accurate. The addition of a search feature significantly streamlines the process of locating specific contacts, a valuable time-saver for users with extensive contact lists.Another notable aspect of this project is its commitment to data integrity. It employs checks to prevent the addition of duplicate contacts based on both phone numbers and names, ensuring the utmost accuracy in your contact list. Users are continuously informed about the status of their actions through confirmation messages, making it an excellent choice for those who demand precision and reliability in their contact management.

The responsive design of the GUI enhances usability further. Contacts are automatically loaded from a CSV file upon startup, guaranteeing immediate access to your data. Error messages guide users in cases of incomplete or incorrect input, showcasing the project's user-centric approach.

In essence, this phonebook project transcends the limitations of conventional address books. It offers a modern, efficient, and user-friendly solution for managing contact information. Whether you're a professional seeking a reliable business contact management tool or an individual looking to simplify your personal contacts, this project is an invaluable asset. Its novelty, combined with its utility, makes it a standout choice in the realm of contact management applications. With this project, managing your contacts has never been easier or more enjoyable.

REFERENCES

1. Alan D. Moore. He's developed both open source and private code using frameworks like Django, Flask, Qt, and of course Tkinter, and is known to contribute to various open-source Python and JavaScript projects
2. B. M. Harwani. His recent publications include jQuery Recipes, published by Apress, Introduction to Python Programming and Developing GUI Applications with PyQT, published by Cengage Learning.
3. Shipman, John W. (2010-12-12), Tkinter reference: a GUI for Python, New Mexico Tech Computer Center, retrieved 2012-01-11.
4. Eby, Phillip J. (7 December 2003). "PEP 333 – Python Web Server Gateway Interface v1.0". Python Enhancement Proposals. Python Software Foundation. Archived from the original on 14 June 2020. Retrieved 19 February 2012.
5. Alfred Aho, John Hopcroft, and Jeffrey Ullman, Data Structures and Algorithms, Addison-Wesley, 1983, ISBN 0-201-00023-7
6. Ellis Horowitz and Sartaj Sahni, Fundamentals of Data Structures in Pascal, Computer Science Press, 1984, ISBN 0-914894-94-3
7. Dinesh Mehta and Sartaj Sahni, Handbook of Data Structures and Applications, Chapman and Hall/CRC Press, 2004, ISBN 1584884355
8. Gao, Leo; Schulman; Hilton, Jacob (2022). "Scaling Laws for Reward Model Overoptimization". *arXiv*:*2210.10760*