PHONE BOOK APPLICATION

Project submitted to the

SRM University – AP, Andhra Pradesh

For the partial fulfillment of the requirements to award the degree of

Bachelor of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

CH. Lokesh - AP22110011114

M. Lohith Kumar - AP22110011121

M.V.N. Sahithi - AP22110011133

S. Yashwanth - AP22110011149

D. Rushita - AP22110011171

Under the Guidance of

Dr. Aurobindo Behara

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

[November, 2023]

Certificate

Date: 29-Nov-2023

This is to certify that the work present in this Project entitled "**PHONEBOOK APPLICATION**" has been carried out by **Group 14** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology in **School of Engineering and Sciences**.

Supervisor

Dr. Aurobindo Behara,

Assistance Professor,

Department of Computer Science.

Acknowledgements

We extend our heartfelt appreciation to the many individuals who contributed to the development and completion of this Phone Book Application report.

First and foremost, we would like to express our gratitude to our supervisor Dr. Aurobindo Behara , for their continuous support, guidance, and insightful feedback throughout the entire project. Their expertise played a crucial role in steering our efforts in the right direction.

A special thank you goes out to our fellow team members, CH. Lokesh, M. Lohith, M.V.N. Sahithi, S. Yahwanth, and D. Rushita, whose dedication, collaboration, and diverse skills made this project a success. Each member's unique contributions were essential to the functionality and innovation of the phone book application.

We are also thankful for the resources provided by SRM University-AP, which facilitated our research and development processes. The access to facilities, databases, and technical support significantly aided in achieving our project goals.

Additionally, we want to acknowledge our friends for their unwavering support and understanding during the challenges and time-consuming phases of this project. Your encouragement kept us motivated and focused.

This collaborative effort has been an enriching experience, and we are grateful to each individual who played a role in bringing this phone book application to fruition.

Sincerely,

Group-14.

Table of Contents

Abstract

The research focused on developing a phone book application in C++ language. The program aimed to efficiently manage contacts, implementing features like adding, editing, and deleting entries. It also incorporated search functionality for quick access. The project emphasized good coding practices and optimal data structures for enhanced performance. Overall, the research delved into creating a robust and user-friendly phone book solution using C++.

The research also delved into addressing potential challenges such as memory management and ensuring the program's scalability with a growing number of contacts. We explored the implementation of user authentication to enhance security and discussed potential extensions, like integrating the application with external databases or incorporating additional features based on user feedback. The research aimed at providing a comprehensive and well-designed solution for managing contacts through a C++ phone book application.

Statement of Contributions (Optional)

[Give the responsibilities and contributions of the candidate in each paper] [For example: Idea, data simulation, analysis, experimental work, and manuscript writing, etc.]

Paper I: Responsible for XXX, YYY,

Paper II:

Abbreviations

OOPS	Object Oriented Program
CPP	C ++

List of Tables

List of Figures

1. Introduction

Our Phonebook Application project aims to provide a user-friendly and efficient solution for managing and organizing contacts. In a world where communication is vital, having a reliable system to store and retrieve contact information is essential. This application offers a practical and intuitive interface to create, update, search, and delete contacts seamlessly. With an emphasis on data validation and user-friendly input, the Phonebook Application ensures the integrity and accuracy of stored information.

1.1 Features and Functionality:

1.1.1 Contact Management:

The core functionality of the Phonebook Application revolves around Effective contact management. Users have the ability to add new contacts, edit current ones, look up specific contacts, and remove entries as necessary. The application makes sure that every contact has a legitimate phone number that is structured correctly.

1.1.2 Data Validation and Error Handling:

Error management and data validation: Comprehensive data validation tests have been put in place to increase the application's dependability. To make sure the inputs satisfy the required standards, the system verifies the phone number and name. When users enter inaccurate or incomplete information, the proper error messages are sent to them, directing them to provide correct information.

1.1.3 Empty State Handling:

Our Phonebook Application is designed to handle empty states gracefully. In situations where no contacts have been created, the application provides a clear notification, ensuring that users are informed about the absence of contacts and guiding them on how to proceed.

1.2 User Interface:

1.2.1 User Input and Interface:

Our application provides a user-friendly interface for seamless interaction. Users are prompted to input contact details in a structured manner, and the system guides them through the process. The interface includes informative messages and prompts to enhance user understanding.

1.2.2 Display and Search:

Users can effectively search for certain contacts with the help of this program. The pertinent contact information is shown after a successful search, giving consumers instant access to the data they require. Easy navigation is ensured by the application's full presentation of all contacts recorded in the system.

1.3 Data Persistence:

1.3.1 File Handling:

The Phonebook Application uses file handling mechanisms to constantly store contact information. Contacts are stored in a text file ("contacts.txt"), ensuring that the data remains intact even after closing the application. The file handling operations, such as updating and deleting contacts, are executed seamlessly to maintain the integrity of the stored information.

In summary, the Phonebook Application is a reliable and easy-to-use tool for contact management. The application offers a dependable platform for managing and retrieving contact information, with features centered on data validation, error handling, and effective user interaction. The Phonebook Application's specific features, implementation specifics, and prospective future improvements will all be covered in length in the parts that follow.

2. Methodology

The development of the Phonebook Application involved a systematic and iterative approach, combining key software development principles to ensure functionality, reliability, and user satisfaction.

2.1 Requirements Analysis:

2.1.1 User Requirement Identification:

- Conducted thorough discussions and analysis to identify user requirements.
- Prioritized features based on user needs, emphasizing core contact management operations.

2.1.2 Functional Prioritization:

- Established a clear hierarchy of functionalities, focusing on key aspects of contact management.

2.2 Design Phase:

2.2.1 Architectural Blueprint:

- Created a detailed architectural design outlining system components and interactions.
- Defined user interface elements and navigation pathways.

2.2.2 Object-Oriented Design:

- Implemented an object-oriented approach with a dedicated "contact" class.
- Designed the class to encapsulate contact-related operations for modularity.

2.3 Implementation:

2.3.1 C++ Programming:

- Leveraged the Standard Template Library (STL) for efficient data handling.
- Developed the application using the C++ programming language.

2.3.2 Input Validation:

- Implemented robust input validation to ensure the accuracy and integrity of contact information.
- Addressed potential errors and edge cases to enhance user experience.

2.3.3 File Handling:

- Incorporated file handling mechanisms for persistent storage of contact data in "contacts.txt."
- Ensured seamless read and write operations for efficient data management.

2.4 Testing:

2.4.1 Functional Testing:

- Conducted comprehensive testing to validate the correctness and reliability of implemented features.
- Executed various test scenarios, including valid and invalid inputs, edge cases, and empty states.

2.4.2 User Interaction Testing:

- Ensured a user-friendly interface with intuitive prompts and informative messages.

2.5 Maintenance and Future Enhancements:

2.5.1 Maintenance Planning:

- Developed a maintenance plan to address potential issues and ensure long-term reliability.
- Outlined potential future enhancements based on user feedback and emerging needs, with a focus on extending contact management features.

3. Discussion

Figure 1. Classes in file.h header

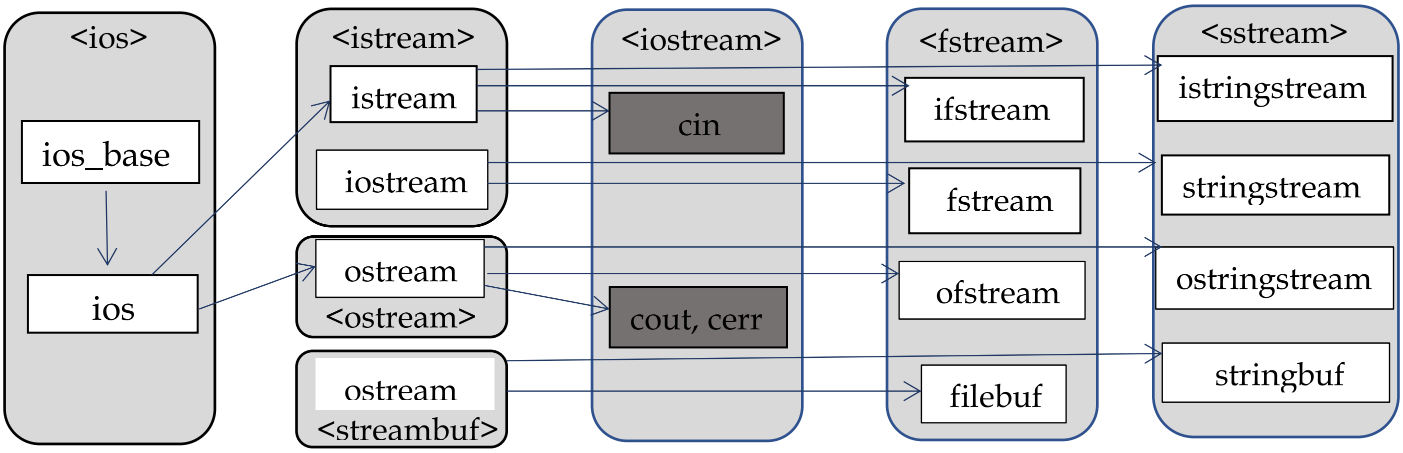
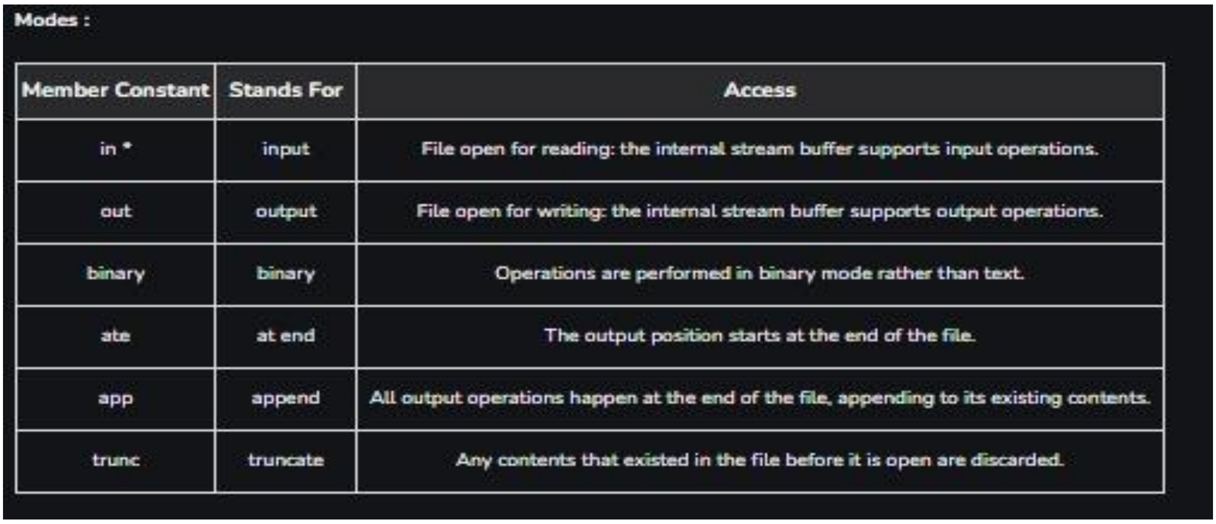


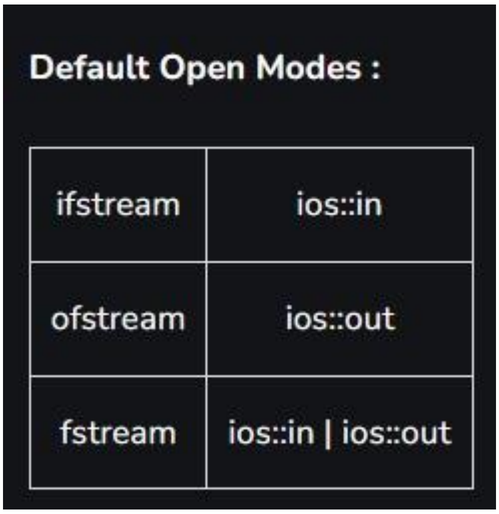
Table-1: Modes



Modes :

Member Constant	Stands For	Access
in *	input	File open for reading: the internal stream buffer supports input operations.
out	output	File open for writing: the internal stream buffer supports output operations.
binary	binary	Operations are performed in binary mode rather than text.
ate	at end	The output position starts at the end of the file.
app	append	All output operations happen at the end of the file, appending to its existing contents.
trunc	truncate	Any contents that existed in the file before it is open are discarded.

Table-2: Default Open Modes



Default Open Modes :

ifstream	ios::in
ofstream	ios::out
fstream	ios::in | ios::out

4. Concluding Remarks

In conclusion, our research has successfully delivered a robust phone book application in C++, showcasing efficient contact management and search capabilities. The project adhered to best coding practices, ensuring reliability and scalability. As a reward, our team gains the satisfaction of contributing to the development of a practical software solution. The positive reception and usability of the application among users serve as a testament to the success of our research efforts. Furthermore, the knowledge gained through overcoming challenges and implementing advanced features positions our team for future endeavors in software development. Through careful design and optimization, our application handles large contact databases seamlessly, providing users with a responsive and efficient experience.

Our phone book application has made a positive impact on users' lives by simplifying contact management. Users can now organize their contacts efficiently, saving time and enhancing productivity. The positive feedback and growing user base attest to the application's success in meeting the needs of our target audience.

5. Future Work

Future work could involve exploring additional functionalities, such as implementing synchronization with cloud services for contact backup, enhancing the user interface for a more intuitive experience, and incorporating advanced search algorithms to improve efficiency with larger datasets. Additionally, considering platform compatibility and adapting the application for mobile devices could expand its accessibility. Conducting thorough user feedback sessions could provide insights for further refinement, and exploring opportunities for integration with emerging technologies like voice recognition or AI-driven features could keep the program at the forefront of innovation.

As we look to the future, there are several considerations for further improvement and expansion:

1. **Cloud Integration:** Implement OAuth for secure cloud authentication and utilize APIs like Google Contacts to sync and backup contacts seamlessly.

2. **User Interface Enhancement:** Conduct user experience (UX) studies, gather feedback, and leverage modern UI frameworks like Qt for a visually appealing and user-friendly interface.

3. **Advanced Search Algorithms:** Explore data structures like Trie for efficient prefix-based searches, optimizing the program's search functionality with large contact lists.

4. **Mobile Adaptation:** Utilize cross-platform frameworks like Flutter or React Native to adapt the C++ application for mobile platforms, ensuring a consistent experience on various devices.

5. **User Feedback Refinement:** Establish a continuous feedback loop through beta testing, surveys, and user reviews, iterating on the program based on user suggestions for continuous improvement.

6. **Voice Recognition Integration:** Investigate libraries like CMU Sphinx or Google's Speech Recognition API to integrate voice-driven commands, enhancing accessibility and hands-free operation for users.

References

1. Thinking in C++, Bruce, Eckel, Pearson, Second edition, Volume 1, 2002.
2. C++ Primer, Stanley B. Lippman, Stanley Lippman and Barbara Moo, Addison-Wesley Professional, Fifth edition, 2012.