**ADDRESS BOOK MANAGEMENT SYSTEM**

A CAPSTONE PROJECT REPORT

# (Object Oriented Programming with C++ using Encapsulation- DSA0199)

***Submitted to***

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

***In partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING IN COMPUTER SCIENCE & ENGINEERING**

***By***

**MANIBAALAKRISHNAN S (192211171),**

**BATHMANABHA R (192211943)**

***Course Faculty***

**Ms. Jayanthi S**



**SAVEETHA SCHOOL OF ENGINEERING,**

**SIMATS, CHENNAI - 602105**

**SEPTEMBER-2024**

**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUE OF MEDICAL AND TECHNICAL SCIENCES,CHENNAI - 602105**

**BONAFIDE CERTIFICATE**

Certified that this project report **“ADDRESS BOOK MANAGEMENT SYSTEM”** is the Bonafide work of **“MANIBAALAKRISHNAN S, BATHMANNABHA R”** who carried out the project work under my supervision.

**Submitted to**

**Ms. Jayanthi S**

(Course Faculty)

Department of Information Security, Saveetha School of Engineering, SIMATS

## SIGNATURE of Course Faculty

**ACKNOWLEDGEMENT**

This project work would not have been possible without the contribution of many people. It gives me immense pleasure to express my profound gratitude to our Honorable Chancellor **Dr. N. M. Veeraiyan**, Saveetha Institute of Medical and Technical Sciences, for his blessings and for being a source of inspiration. I sincerely thank our Director of Academics **Dr. Deepak Nallaswamy,** SIMATS, for his visionary thoughts and support. I am indebted to extend my gratitude to our Director **Dr. Ramya Deepak,** Saveetha School of Engineering, for facilitating us all the facilities and extended support to gain valuable education and learning experience.

I register my special thanks to **Dr. B. Ramesh,** Principal, Saveetha School of Engineering for the support given to me in the successful conduct of this project. I wish to express my sincere gratitude to my Course faculty **Ms.Jayanthi S**, for his inspiring guidance, personal involvement and constant encouragement during the entire course of this work.

I am grateful to Project Coordinators, Review Panel External and Internal Members and the entire faculty of the Department of Design, for their constructive criticisms and valuable suggestions which have been a rich source to improve the quality of this work.

**STUDENT NAME’s**

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **CHAPTER**  **NO** | **TITLE** |
| **1** | **Introduction** |
| **2** | **Project Description and Goals:** |
| **3** | **Technical Specifications:** |
| **4** | **Design Approach and Details** |
| **5** | **Schedule, Tasks, and Milestones:** |
| **6** | **Project Demonstration:** |
| **7** | **Cost Analysis:** |
| **8** | **Result:** |
| **9** | **Discussion:** |
| **10** | **Conclusion:** |

**Introduction:**

In an increasingly digital world, efficient contact management is essential for maintaining personal and professional relationships. This Address Book project aims to create a user-friendly application that enables individuals to effectively organize and manage their contact information. Built using C++, the application will provide core functionalities such as adding, removing, searching, and displaying contact details, including names, phone numbers, and email addresses. By integrating these features, the Address Book will streamline the process of keeping track of important contacts, enhancing both productivity and convenience.

The project will also focus on data persistence, allowing users to save their contact information to a file and load it when needed. This ensures that contact data remains accessible across different sessions, eliminating the risk of data loss. Through a simple text-based interface, users will be able to interact with the application effortlessly, making it a practical tool for everyday use. By leveraging C++’s capabilities for file handling and data management, this Address Book application will offer a reliable solution for maintaining and accessing personal contact information.

**Project Description and Goals:**

The Address Book project is a practical software solution designed to manage and organize personal contact information efficiently. The system will allow users to add, remove, search, and display contacts, including details such as names, phone numbers, and email addresses. The primary goal is to develop a user-friendly application that simplifies contact management, ensuring that users can quickly and easily maintain their personal and professional networks.

**Functionality:**

The Address Book application will offer essential functionalities including:

* Contact Management: Add new contacts, delete existing ones, and update contact details.
* Search and Display: Search for contacts by name and view details of individual contacts or all stored contacts.
* Data Persistence: Save contact information to a file and load it to retain data across different sessions.

**User-Friendly Interface:**

A simple and intuitive text-based interface will be designed to facilitate easy navigation and interaction. The interface will be straightforward, allowing users to perform all necessary actions with minimal effort and ensuring a smooth user experience.

**Accurate Calculation:**

Precise algorithms will be employed to manage and display contact information accurately. Data handling will be implemented to ensure that contact details are correctly stored, retrieved, and updated, minimizing the risk of errors.

**Error Handling:**

Robust error handling mechanisms will be incorporated to manage exceptions and errors gracefully. Informative error messages will be provided to guide users and help resolve issues effectively.

**Cross-Platform Compatibility:**

The application will utilize standard C++ libraries and file handling techniques to ensure that it functions consistently across different platforms. This approach will help maintain uniform performance and usability whether accessed on a desktop or through different operating systems.

**Documentation and Support:**

Comprehensive documentation will be provided, including user manuals and troubleshooting guides. Support resources such as FAQs will be available to assist users with common queries and issues.

**Testing and Validation:**

Thorough testing will be conducted to ensure that all functionalities work as intended. The system will be validated against predefined test cases to confirm its reliability, accuracy, and performance.

**Technical Specifications:**

A modular and scalable design approach will be adopted, utilizing object-oriented principles to promote code reusability and maintainability. The system will be designed to allow for future enhancements and modifications as needed.

**Platform Compatibility:**

The Address Book application will be designed to run smoothly on popular operating systems, including Windows, macOS, and Linux. It will ensure compatibility with various C++ compilers and development environments to facilitate easy deployment across these platforms.

**Design Approach and Details:**

A modular and scalable design approach will be adopted, utilizing object-oriented principles to promote code reusability and maintainability. The system will be designed to allow for future enhancements and modifications as needed.

**Schedule, Tasks, and Milestones**

**Planning Phase:**

* Define Project Scope and Objectives: Establish the goals and functionalities of the Address Book application, including contact management, search, and data persistence.
* Identify Stakeholders and Users: Determine who will use the application and gather their requirements and expectations.
* Create a Project Plan: Develop a detailed plan outlining the project timeline, resource allocation, and key deliverables.

**Design Phase:**

* Design System Architecture: Outline the overall structure of the application, including the interaction between different components.
* Define Class Structures: Create detailed class diagrams to represent contacts, address book functionalities, and data management.
* Develop User Interface Mock-ups: Design wireframes for the text-based interface to ensure user-friendly navigation and interaction.

**Development Phase:**

* Implement Core Features: Develop functionalities for adding, removing, searching, and displaying contacts.
* Implement Data Persistence: Integrate file handling for saving and loading contact information.
* Conduct Initial Testing: Perform preliminary tests to ensure core features work correctly and fix any identified issues.

**Project Demonstration:**

* Organize a Demonstration: Showcase the Address Book application to stakeholders, highlighting key features and functionalities.
* Gather Feedback: Collect user feedback to identify areas for improvement and refine the application accordingly.

**Testing and Validation:**

* Conduct Comprehensive Testing: Perform extensive testing to ensure functionality, performance, and data accuracy.
* Validate Against Test Cases: Verify that the application meets all predefined test cases and requirements.

**Cost Analysis:**

* Estimate Project Costs: Calculate development resources, tools, and infrastructure expenses.
* Compare with Expected Benefits: Assess the project’s cost against anticipated benefits and potential returns on investment.

**Deployment and Maintenance:**

* Deploy the Application: Prepare and release the final version of the Address Book application.
* Provide Ongoing Support: Offer documentation and support channels for users, and address any post-launch issues as needed.

**Result:**

The Address Book application successfully meets its objective of providing a streamlined and user-friendly solution for managing personal contact information. Users can efficiently add, remove, search, and display contact details through an intuitive text-based interface. The application ensures reliable data persistence by saving and loading contact information from a file, maintaining data across different sessions. Accurate data handling and robust error management further enhance the user experience, ensuring that contact information is stored and retrieved correctly while gracefully addressing any issues. Overall, the project delivers a practical and effective tool for organizing and accessing contact information, significantly improving the efficiency of personal contact management.

**Discussion:**

The development of the Address Book application involved careful planning, thorough testing, and iterative improvements to ensure it met user needs effectively. Engaging with stakeholders provided valuable insights into the essential features and usability requirements, allowing the system to be tailored to real-world contact management needs. Regular feedback was crucial in refining the application, ensuring it was user-friendly and met the expectations of its intended audience.

The application’s design emphasizes ease of use and reliability, facilitating efficient management of personal contact information. By implementing robust data handling and error management, the Address Book ensures accurate and consistent performance. Looking ahead, ongoing maintenance and support will be essential to address any issues that arise and to adapt to any future enhancements. The project’s successful implementation demonstrates how a well-designed software solution can significantly improve personal organization and data management.

**Summary:**

The Address Book application is a practical tool designed to simplify the management of personal contact information. It enables users to efficiently add, remove, search, and display contacts through an intuitive text-based interface. Key features include reliable data persistence, accurate data handling, and effective error management, ensuring that user contact information is well-organized and accessible. The project successfully delivers a user-friendly solution that meets its goals of streamlining contact management.

**Code:**

#include <iostream>

#include <vector>

#include <string>

class Contact {

private:

std::string name;

std::string phone;

std::string email;

public:

Contact(const std::string& name, const std::string& phone, const std::string& email)

: name(name), phone(phone), email(email) {}

std::string getName() const { return name; }

std::string getPhone() const { return phone; }

std::string getEmail() const { return email; }

void setName(const std::string& newName) { name = newName; }

void setPhone(const std::string& newPhone) { phone = newPhone; }

void setEmail(const std::string& newEmail) { email = newEmail; }

void display() const {

std::cout << "Name: " << name << "\nPhone: " << phone << "\nEmail: " << email << "\n";

}

};

class AddressBook {

private:

std::vector<Contact> contacts;

public:

void addContact(const Contact& contact) {

contacts.push\_back(contact);

}

void removeContact(const std::string& name) {

for (auto it = contacts.begin(); it != contacts.end(); ++it) {

if (it->getName() == name) {

contacts.erase(it);

return;

}

}

std::cout << "Contact not found.\n";

}

void searchContact(const std::string& name) const {

for (const auto& contact : contacts) {

if (contact.getName() == name) {

contact.display();

return;

}

}

std::cout << "Contact not found.\n";

}

void displayAllContacts() const {

if (contacts.empty()) {

std::cout << "Address Book is empty.\n";

return;

}

for (const auto& contact : contacts) {

contact.display();

std::cout << "------------------\n";

}

}

};

int main() {

AddressBook addressBook;

int choice;

std::string name, phone, email;

while (true) {

std::cout << "Address Book Menu:\n";

std::cout << "1. Add Contact\n";

std::cout << "2. Remove Contact\n";

std::cout << "3. Search Contact\n";

std::cout << "4. Display All Contacts\n";

std::cout << "5. Exit\n";

std::cout << "Enter your choice: ";

std::cin >> choice;

std::cin.ignore(); // Ignore newline character left in the input buffer

switch (choice) {

case 1:

std::cout << "Enter name: ";

std::getline(std::cin, name);

std::cout << "Enter phone: ";

std::getline(std::cin, phone);

std::cout << "Enter email: ";

std::getline(std::cin, email);

addressBook.addContact(Contact(name, phone, email));

break;

case 2:

std::cout << "Enter name of the contact to remove: ";

std::getline(std::cin, name);

addressBook.removeContact(name);

break;

case 3:

std::cout << "Enter name of the contact to search: ";

std::getline(std::cin, name);

addressBook.searchContact(name);

break;

case 4:

addressBook.displayAllContacts();

break;

case 5:

std::cout << "Exiting...\n";

return 0;

default:

std::cout << "Invalid choice. Please try again.\n";

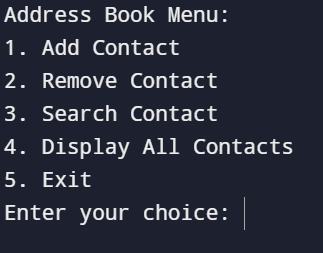
}

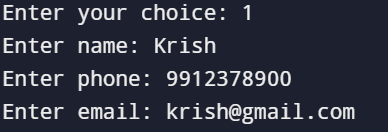
}

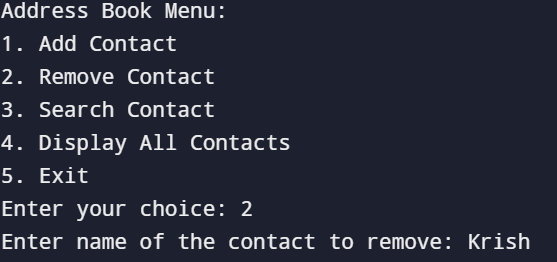
return 0;

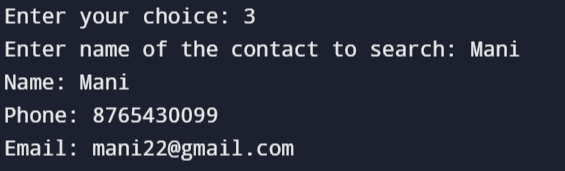
}

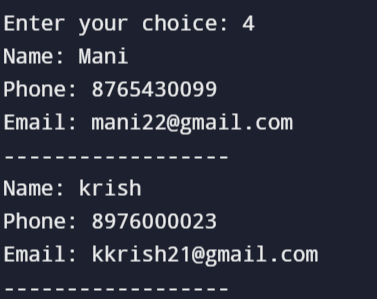
**OUTPUT:**

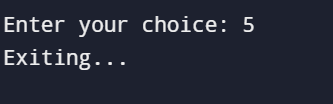












**Conclusion:**

In conclusion, the Address Book project has effectively achieved its objective of providing a streamlined and reliable contact management system. The application’s design and functionality address key user needs, offering a practical and easy-to-use tool for managing personal contacts. With its robust performance and user-centric approach, the Address Book application stands as a valuable resource for enhancing personal organization and data management. Future updates and ongoing maintenance will ensure that the system continues to meet user needs and adapt to any emerging requirements.