PURBANCHAL UNIVERSITY

Biratnagar, Nepal

A Project report on

“CONTACT MANAGEMENT SYSTEM”

In the partial fulfillment for the requirement of the 2st Semester Project-II (subject code- BIT 156 CO) in the completion of Bachelor of Information Technology (BIT) degree at KIST college of Information Technology, under Purbanchal University.

Submitted by:

Priyanshu Kushawaha	**(SN. 321176)**
Taweshal Dev Thakur	**(SN. 321192)**
Bipesh Poudel	**(SN. 321155)**

Under The Guidance of

Mr. Deepak Khadka

HOD, BIT

KIST COLLEGE OF INFORMATION AND TECHNOLOGY

KAMALPOKHARI, KATHMANDU, NEPAL

KIST COLLEGE OF INFORMATION AND TECHNOLOGY

KAMALPOKHARI, KATHMANDU, NEPAL

CERTIFICATE

This is to certify that the project work entitled **“CONTACT MANAGEMENT SYSTEM”** is carried out by **Bipesh Paudel (6548), Priyanshu Kushawaha (6679), Taweshal Dev Thakur (6645),** bona fide students of **KIST COLLEGE OF INFORMATION AND TECHNOLOGY** in partial fulfillment for the award of **BACHELOR IN INFORMATION AND TECHNOLOGY** of the **PURBANCHAL UNIVERSITY, BIRATNAGAR NEPAL**, during the year **2024**. It is certified that all corrections indicated for internal assessment have been incorporated in the report submitted in the department library. The project report has been approved, as it satisfied the academic requirements in respect of the project work prescribed for the said degree.

The details of the students are as follows:-

NAME	REGISTRATION NO.	SYMBOL NO.
Priyanshu Kushawaha	058-3-2-05294-2022	321176
Taweshal Dev Thakur	058-3-2-05311-2022	321192
Bipesh Poudel	058-3-2-05272-2022	321155

Course Semester: - 2nd Semester
Subject: - Project-II

Subject Code: - BIT (156CO)

Mr. Deepak Khadka
Program Coordinator, BIT

KIST COLLEGE OF INFORMATION AND TECHNOLOGY
KAMALPOKHARI, KATHMANDU

Examiner's Certification

The Project Report
On

"CONTACT MANAGEMENT SYSTEM"

Developed by

Bipesh Paudel
Priyanshu Kushawaha
Taweshal Dev Thakur

Is approved and is acceptable in qualified form.

Internal Examiner	**External Examiner**
Name:	Name:
Designation:	Designation:

ACKNOWLEDGEMENT

It is with greatest satisfaction and euphoria that we are submitting our project report entitled **“Contact Manangement System”.** We have completed it as a part of the curriculum of **PURBANCHAL UNIVERSITY.**

We would like to express our deepest appreciation to all those who provided us the possibility to complete this project. A special gratitude to our **BIT Coordinator Mr. Deepak Khadka** who guided us throughout the project. We would also like to thank our friends and family who continuously supported, motivated us and offered deep insight into the study.

We are indebted to all members of **KIST College,** for the valuable support and suggestions provided by them using their specific fields knowledge. We are grateful for their cooperation during the period of our project.

Finally, we would also like to express our gratitude to **PURBANCHAL UNIVERSITY** who gave us the beautiful opportunity to explore on this wonderful project. This project helped us in doing a lot of research and we came to know about so many new things and we are really thankful.

We hope the university will accept our attempt as a successful project.

Thank you!

Priyanshu Kushawaha (321176)

Taweshal Dev Thakur (321192)

Bipesh Paudel (321155)

STUDENT’S DECLARATION

We hereby declare that the project report entitled “**CONTACT MANAGEMENT SYSTEM**” is a result of our own work. If we are found guilty of copying any other report or published information and showing as our original work, we understand that we shall be liable and punishable by **Purbanchal University**.

We further certify that this Project submitted in partial fulfillment of the requirement for the award of Bachelor in Information Technology (**BIT**) of the **Purbanchal University** is our original work and has not been submitted for award of any other degree or other similar title or prize.

NAME	REGISTRATION NO.	SYMBOL NO.
Priyanshu Kushawaha	058-3-2-05294-2022	321176
Taweshal Dev Thakur	058-3-2-05311-2022	321192
Bipesh Poudel	058-3-2-05272-2022	321155

TO WHOM IT MAY CONCERN

This is to certify that, Mr. Priyanshu Kushawaha, Mr. Taweshal Dev Thakur and Mr. Bipesh Paudel of Bachelor in Information Technology (BIT) has studied as per the curriculum of BIT 2nd Semester and completed the project entitled **"CONTACT MANAGEMENT SYSTEM"**. This project is the original work of Mr. Priyanshu Kushawaha, Mr. Taweshal Dev Thakur and Mr. Bipesh Paudel and was carried out under the supervision of Mr. Deepak Khadka as per the guidelines provided by Purbanchal University and certified as per the student's declaration that project "**Contact Management System**" has not been presented anywhere as a part of any other academic work.

The detail of the student is as follows:

Name of Students	:	Priyanshu Kushawaha
	:	Taweshal Dev Thakur
	:	Bipesh Paudel
Semester	:	2nd
Subject Code	:	BIT 156 C0
Project Title	:	**Contact Management System**

…………………………….

Mr. Deepak Khadka

Program Coordinator, BIT

KIST College of Information Technology

Table of contents

Abstract

The purpose of **“Contact Management System”** is to automate the existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

Contact Management System, as declared above, can lead to error free, secure, reliable and fast management systems. It can assist the user to concentrate on their other activities rather than concentrating on the record keeping. Thus it will help organizations in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. Basically the project describes how to manage for good performance and better services for the clients.

CHAPTER 1

INTRODUCTION

1.1 Motivation

We found that we were good at coding and had the ability to reason about the code very well and it didn't take long for us to realize that this is what we wanted to do for the rest of our life. Though when we began college we still wanted to go into programming, as the state of web applications became more interesting in the last few days, we got into that and that's what we are doing now! We really couldn't imagine doing anything else, and programming is definitely one of the top skills too.

1.2 Programming Language

A programming language is a formal language that specifies a set of instructions that can be used to produce various kinds of output. Programming languages generally consist of instructions for a computer. Programming languages can be used to create programs that implement specific algorithms.

1.3 C++ Language

C++ is an extension of the C programming language, adding features such as classes and objects, inheritance, polymorphism, and operator overloading. Developed by Bjarne Stroustrup in the early 1980s, C++ combines the features of high-level languages with the efficiency and flexibility of low-level languages.

1.4 Advantages of C++ Language

C++ language offers several advantages, making it a popular choice for various programming tasks. Here are some of its key advantages:

- Efficiency
- Portability
- Wide Standard Library
- Object-Oriented Programming (OOP)
- Strong Community and Resources
- Highly Customizable
- Legacy Code and Compatibility

- Influence on Other Languages
- Compatibility with Existing Codebases

1.5 Disadvantages of C++ Language

Despite its strengths, C++ also presents certain challenges and drawbacks that developers should consider: Low-Level Language

- Complexity
- Manual Memory Management
- Potential for Bugs
- Limited Standard Library
- Security Risks
- Platform-Dependent

1.6 Project Description

A contact management system in C++ is a software application designed to efficiently store, organize, and manage contact information, including names, phone numbers, email addresses, and more. This system enables users to add, edit, delete, and search for contacts, providing a user-friendly interface for seamless contact management. It incorporates features like data validation, data storage, and retrieval, making it a practical tool for individuals or organizations to streamline their contact-related tasks and maintain an organized contact database.

1.7 Introduction to Contact Management System

The Contact Management System has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and in some cases reduce the hardships faced by this existing system. Moreover this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

The application is reduced as much as possible to avoid errors while entering the data. It also provides an error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus by this all it proves it is user friendly.

CMS, as described above can lead to error free, secure, reliable and fast management systems. It can assist the user to concentrate on their other activities rather than concentrating on the record keeping. Thus it will help organizations in better utilization of resources.

1.8 Advantages of Contact Management System

A contact management system (CMS) offers several advantages for individuals and organizations:

- **Efficient Organization:** CMS helps users efficiently organize contact information, making it easy to access and manage large contact databases.

- **Time-Saving:** It streamlines contact-related tasks like adding, editing, and searching for contacts, saving users time and effort.

- **Improved Communication:**CMS ensures accurate contact details, reducing the risk of communication errors and improving overall communication with contacts.

- **Centralized Data:** It provides a centralized location for storing all contact information, ensuring data consistency and reducing redundancy.

- **Customization:** Many CMS solutions allow users to customize fields and data types to meet specific needs, making it adaptable to various industries and purposes.

- **Data Security:** Most CMSs offer security features to protect sensitive contact data, including user authentication and encryption.

- **Integration:** CMSs can often integrate with other software systems, such as email clients or CRM systems, to enhance functionality.

- **Scalability:** They can scale to accommodate growing contact lists, making them suitable for both small businesses and large enterprises.

1.9 Disadvantages of Contact Management System

While contact management systems (CMS) offer numerous advantages, they also come with some potential disadvantages and challenges:

- **Cost:** Some contact management systems, especially advanced or enterprise-level solutions, can be expensive to purchase and maintain, which might not be feasible for small businesses or individuals with limited budgets.

- **Data Entry:** Initially populating the CMS with contact information can be time-consuming, especially for large contact lists. It may require manual data entry or data migration from existing sources.

- **Data Security:** Storing contact information in a CMS carries the risk of data breaches or unauthorized access if the system is not properly secured.

- **Maintenance:** Regular maintenance, updates, and backups are necessary to ensure the system functions smoothly and that data remains secure and accessible.

- **Customization Complexity:** Highly customizable CMS solutions may require technical expertise to configure and tailor to specific needs, potentially adding complexity to implementation.

- **Limited Features:** CMS may lack certain features or functionalities that users require for their specific needs, leading to limitations in how contacts can be managed and utilized.

- **Data Loss Risk:** If data is not adequately backed up or if there are system failures, there's a risk of losing valuable contact information.

- **User Adoption:** Encouraging all users within an organization to consistently use the CMS can be a challenge. Resistance to change or lack of awareness can hinder adoption.

1.10 Problem Statement

In today's fast-paced digital environment, individuals and organizations are inundated with an ever-expanding network of contacts, including clients, customers, colleagues, suppliers, and personal connections. Managing and maintaining accurate contact information, as well as effectively communicating and collaborating with these contacts, has become a significant challenge. Traditional contact management methods, such as paper-based systems or fragmented digital tools, often result in inefficiency, data errors, and missed opportunities for meaningful interactions. There is a critical need for a comprehensive and user-friendly Contact Management System (CMS) that can efficiently organize, store, update, and provide quick access to contact information while also facilitating communication and engagement. This CMS should address the growing complexities of contact management, ensuring data accuracy, security, and accessibility across various devices and platforms, ultimately enhancing productivity and fostering stronger relationships with contacts.

1.11 Objectives

The primary objectives of a Contact Management System (CMS) are to streamline and optimize contact-related tasks and activities, ultimately improving efficiency, organization, and communication. Here are the key objectives of a CMS:

- **Efficient Contact Management:**The CMS aims to provide a centralized and efficient way to manage a database of contacts, making it easy to add, edit, and organize contact information.

- **Data Accuracy:** Ensuring the accuracy and consistency of contact data is a fundamental objective. The CMS should minimize errors and redundancies in contact information.

- **Quick Access:** Users should be able to access contact details swiftly, enabling faster communication and reducing the time spent searching for contact information.

- **Improved Communication:** Facilitating communication with contacts is a key objective. This includes features like email integration, reminders, and tracking communication history.

- **Customization:** The CMS should be flexible and allow for customization to meet the specific needs of individuals or organizations, including the ability to define custom fields and data types.

- **Security and Privacy:** Protecting sensitive contact information is paramount. The CMS should implement robust security measures to safeguard data against unauthorized access and breaches.

- **User-Friendly Interface**: A user-friendly and intuitive interface is essential to encourage adoption and ease of use for all users.

Overall, the objective of a CMS is to simplify the management of contact information, improve communication and engagement with contacts, and enhance overall productivity and organization for individuals and organizations alike.

1.12 Scope

The scope of a Contact Management System (CMS) encompasses various aspects and functionalities that define what the system will include and achieve. Here is the scope of a CMS:

- **Contact Information Management:**
  - Capture, store, and organize contact details, including names, addresses, phone numbers, email addresses, and additional information.
  - Support for a wide range of contact types, such as clients, colleagues, suppliers, and personal connections.

- **Data Entry and Editing:**
  - Provide user-friendly interfaces for adding, editing, and deleting contact records.
  - Validate and standardize data entry to ensure accuracy and consistency.

- **Search and Retrieval:**
  - Implement robust search and retrieval mechanisms, allowing users to quickly find specific contacts based on various criteria.
  - Support advanced search filters and sorting options.

- **Communication Tools:**
  - Facilitate communication with contacts through integration with email clients, allowing users to send emails directly from the system.

    - Provide features for scheduling calls, meetings, and reminders.

- **Security and Privacy:**
    - Implement user authentication and role-based access control to protect contact data.
    - Comply with data privacy regulations to safeguard the privacy of contact information.

- **Scalability and Performance:**
    - Ensure the system can scale to accommodate a growing number of contacts and users.
    - Optimize performance for efficient data retrieval and management.

- **Cross-Platform Accessibility:**
    - Support access to the CMS from various devices and platforms, including web browsers, mobile apps, and desktop applications.

- **Data Backup and Recovery:**
    - Implement regular data backup and recovery mechanisms to prevent data loss in case of system failures or errors.

- **User Interface (UI) Design:**
    - Develop an intuitive and user-friendly UI that encourages user adoption and ease of use.

The scope of a CMS is essential in defining the system's boundaries, features, and functionalities to meet the needs of individuals and organizations in managing and engaging with their contacts effectively.

CHAPTER 2

MATERIALS AND METHOD

2.1 Header files

<iostream>	This header file is used for input-output stream operations in C++.
<fstream>	This header file is used for handling file input and output operations in C++.
<cstdlib>	This header file is used to include standard library functions like exit() which is used to exit the program.
<cstring>	This header file declares functions for manipulating strings and arrays of characters.
<conio>	This header file provides console input/output functions. It's a non-standard header file and may not be available on all platforms.

2.2 Functions used:

getdata()	This function is a member function of the Details class. It is used to get input data from the user for contact details.
showcontact()	This function is a member function of the Details class. It displays the contact details.
writeonfile()	This function is a member function of the Details class. It writes the contact details to a binary file.
readfromfile()	This function is a member function of the Details class. It reads and displays all contact details from the binary file.
searchfromfile()	This function is a member function of the Details class. It searches for a specific contact detail based on the phone number.

deletefromfile()	This function is a member function of the Details class. It deletes a contact detail based on the phone number from the binary file.
authenticate()	This function is a member function of the LoginManager class. It authenticates the user by comparing the entered username and password with the data stored in the Login_Data.dat file.
signup()	This function is a member function of the LoginManager class. It allows the user to create a new account by entering a username and password.
loginMenu()	This function displays the login menu options and returns the choice made by the user.
main()	The main function of the program. It contains the control flow of the program, including the login process and contact management system.

CHAPTER 3

SYSTEM DESIGN

3.1 Working Principle

The working principle of a Contact Management System (CMS) involves a series of steps and processes that allow users to efficiently manage, organize, and interact with their contact information. Here's an overview of the working principle for a typical CMS:

- **Data Input:**
  - Users input contact information into the CMS. This data can include names, addresses, phone numbers, email addresses, job titles, and other relevant details.
  - The CMS provides user-friendly forms and interfaces for data entry, which can include validation to ensure data accuracy.

- **Data Storage:**
  - The CMS stores the contact information in a structured database. This database is designed to efficiently organize and manage large volumes of contact data.
  - Contacts are typically organized into records or entries, with each record representing an individual or entity.

- **Data Editing and Management:**
  - Users have the ability to edit, update, or delete contact records as needed. This includes making changes to contact details or adding notes and tags for additional context.
  - Contact records can be organized into categories or groups for better management.

- **Customization:**
  - Users can customize the CMS to fit their specific needs. This may involve adding custom fields or defining data types to accommodate unique contact information requirements.

- **Accessibility:**
    - Users can access their contact data from various devices.

- **User Interface (UI):**
    - A user-friendly and intuitive UI design encourages user adoption and makes it easy for users to navigate the CMS.

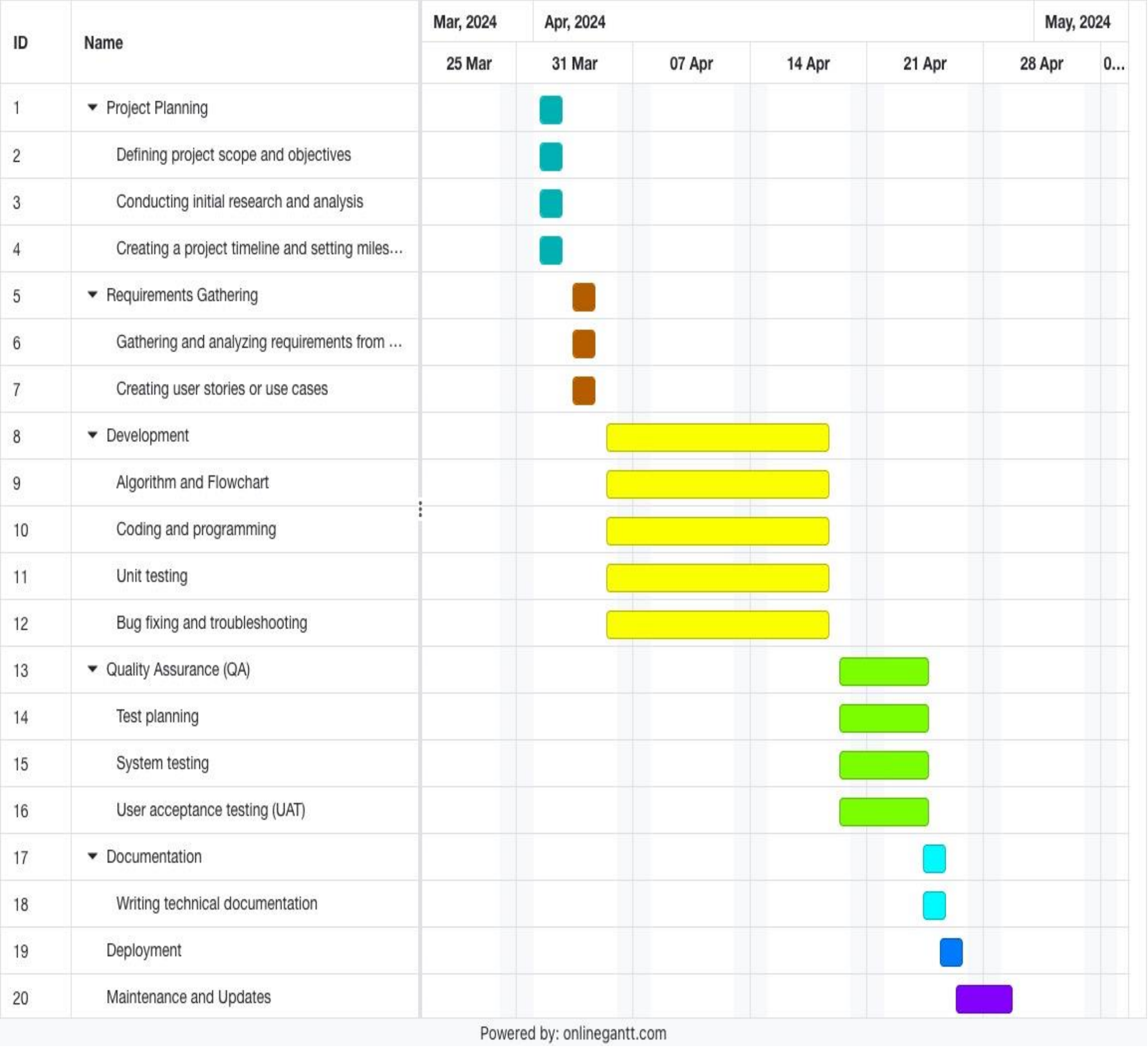
The working principle of a CMS revolves around efficiently managing and leveraging contact information to improve communication, organization, and productivity for individuals and organizations. It offers a structured and centralized approach to contact data management.

3.2 Functional Requirement

In software and system engineering, a functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between input and outputs.

3.3 Gannt Chart

This project required about 1 month to be completed. All time period required for this project for different task are shown in graph.



ID	Name
1	Project Planning
2	Defining project scope and objectives
3	Conducting initial research and analysis
4	Creating a project timeline and setting miles…
5	Requirements Gathering
6	Gathering and analyzing requirements from …
7	Creating user stories or use cases
8	Development
9	Algorithm and Flowchart
10	Coding and programming
11	Unit testing
12	Bug fixing and troubleshooting
13	Quality Assurance (QA)
14	Test planning
15	System testing
16	User acceptance testing (UAT)
17	Documentation
18	Writing technical documentation
19	Deployment
20	Maintenance and Updates

3.4 Algorithm

Step 1: Start

Step 2: Display Login Menu options:

a. Login
b. Signup
c. Exit

Step 3: User selects an option from the Login Menu.

Step 4: If Login is selected:

a. Prompt user to enter username and password.
b. Authenticate user credentials:
c. If credentials are correct, display Contact Management System Menu
d. If credentials are incorrect, display error message and go back to Login Menu.

Step 5: In Contact Management System Menu, show options:

a. Add a new contact
b. List all contacts
c. Search for a contact
d. Delete any contact
e. Exit

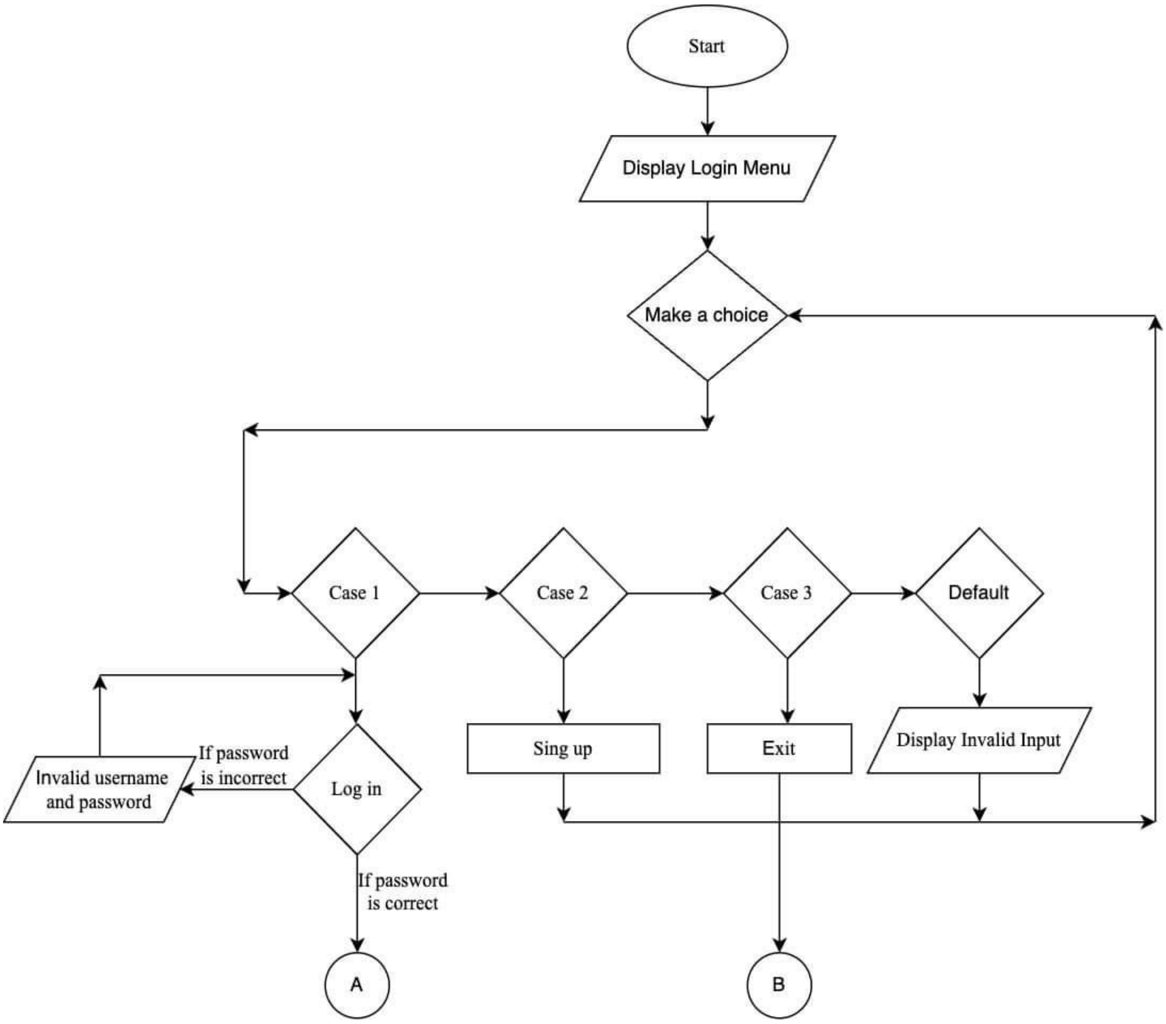
User selects an option from the CMS Menu. - Perform corresponding action based on selected option. - Return to CMS Menu.

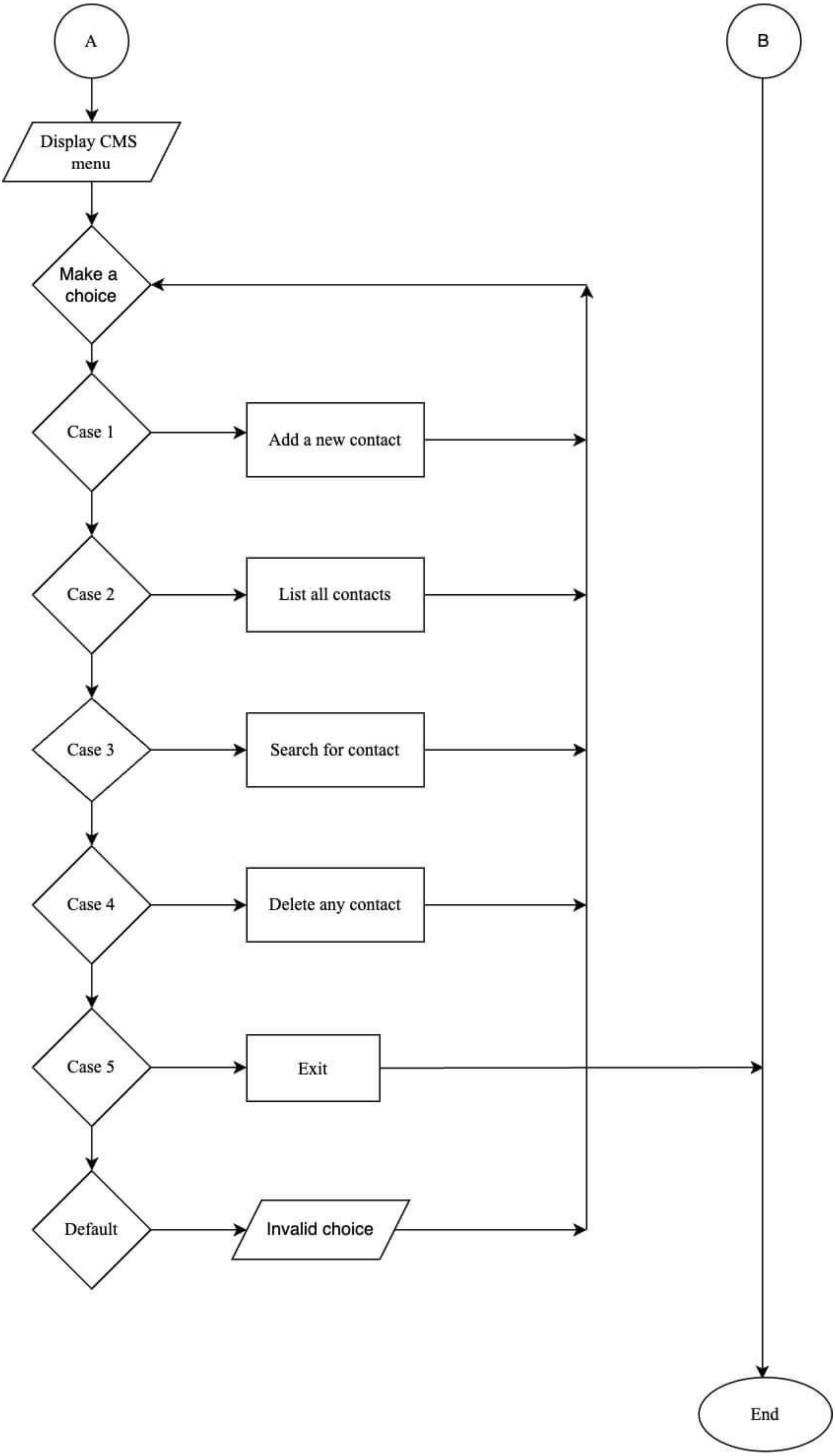
Step 6: If Signup is selected, Prompt user to enter a new username and password. Create a new account with entered credentials. Display success message. - Go back to Login Menu.

Step 7: If Exit is selected, End the program.

Step8: End

3.5 Flowchart




A
B
Display CMS menu
Make a choice
Case 1
Add a new contact
Case 2
List all contacts
Case 3
Search for contact
Case 4
Delete any contact
Case 5
Exit
Default
Invalid choice
End

CHAPTER 4

REQUIREMENT ANALYSIS & IMPLEMENTATION SYSTEM

4.1 Requirements

Hardware: Hardware is the collection of physical parts of a computer system.

- ❖ Memory (RAM): 4GB
- ❖ System Type: 64-bit System
- ❖ Storage Capacity: 10GB of free disk space or above free space available
- ❖ CPU: A clock speed of at least 2GHz and at least 2 cores is recommended

Software: Software is a set of instructions, data or programs used to operate computers and
execute specific tasks. It is the opposite of hardware, which describes the physical aspects of a
Computer.

- ❖ OS: Windows 10 or Higher
- ❖ Applications: Dev C++ or another C++ compiler.

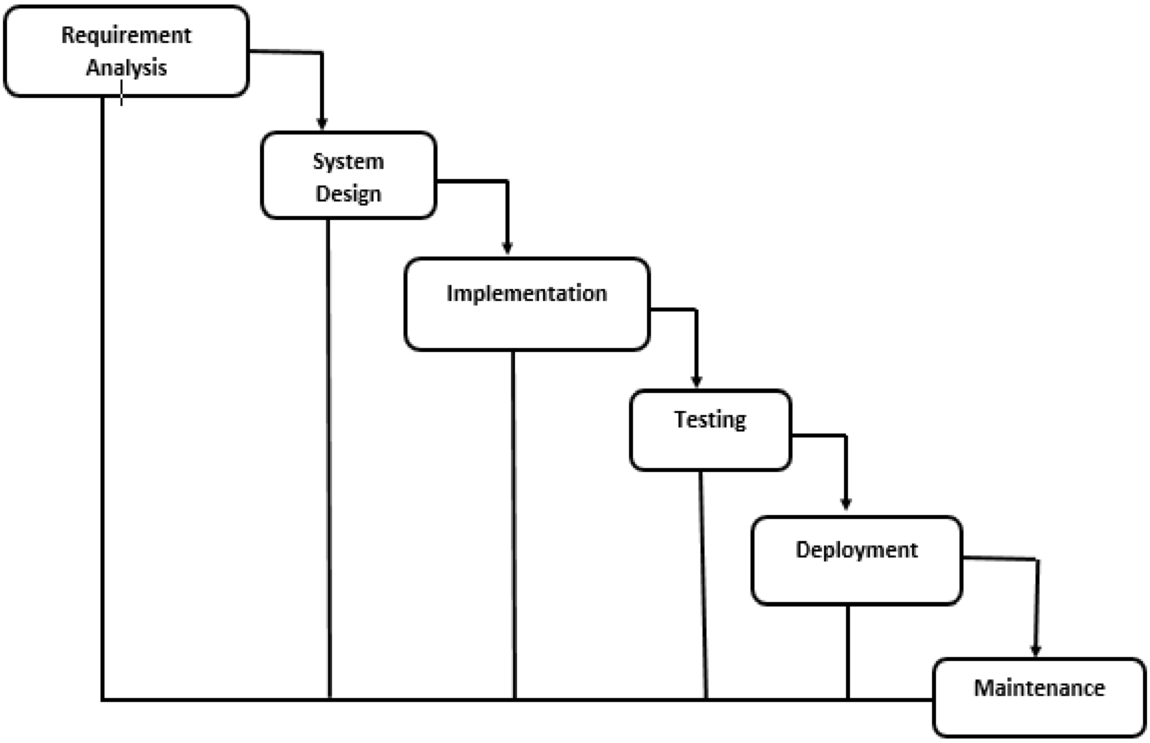
4.2 System Methodology

System Methodology is a methodology for systematically organizing the best ways to develop systems efficiently. It is a step-by-step process for developing any system. There are many system development methodologies. Some of them are Waterfall Model, Iterative Model, Develop Model, V-Model, Spiral Model, Lean and Agile Model, Prototype Model, etc.

In this project, we are going to use the Waterfall Model approach since our project is short and our requirements are fixed. The Waterfall Model is one of the oldest SDLC models but it is best for short projects as this model involves a rigid structure that demands all system requirements be defined at the very start of a project. Only then the design and development stages begin.

Waterfall Model- Design

Waterfall approach was the first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of the software development is divided into separate phases. In this Waterfall Model, typically, the outcome of one phase acts as the input for the next phase sequentially. The following illustration is a representation of the different phases of the Waterfall Model.



The sequential phases in Waterfall Model are: -

- **Requirement Analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document. The detailed requirements of the client like costs, assumption, risks, dependencies, success matrices, and timelines for completion, etc. are done in this phase.

- **System Design:** The requirement specifications from the first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture. Here, the software developers design a technical solution to the problems set out by the requirements. Once this is complete, it is transformed into physical design using specific hardware and software technologies. In this project, we have used C++ programming as a System Design Language.

- **Implementation:** With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit testing. Once the design is complete, technical implementation starts. This might be the shortest phase of the Waterfall Model because the detailed research and design have already been done. In this phase, programmers code applications based on project requirements and specifications, with some testing and implementation taking place as well.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Testing is done to ensure that there are no errors and all the requirements have been completed, ensuring good user experience with the software before releasing the software to the client.
- **Deployment of System:** Once the functional and non-functional testing is done, the product is deployed in the client environment or released in the market.
- **Maintenance:** There are some issues which come up in client environments. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the client environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for the previous phase and it is signed off, so the name “Waterfall Model”. In this model, phases do not overlap.

CHAPTER 5

FUTURE SCOPE AND CONCLUSION

5.1 Future Scope

It may help collect perfect management in detail. In a very short time, the collection will be obvious, simple and sensible. It will help a person to know the management of Passed years perfectly and vividly. It also helps in current work relative to CMS. It will also reduce the cost of collecting the management and collection procedure will go on smoothly. Our project aims at process automation, i.e. we have tried to computerize various processes of CMS. Following are the future scopes of our system:

- In the computer system the person has to fill the various forms & number of copies of the forms can be easily generated at a time.

- In a computer system, it is not necessary to create the manifest but we can directly print it, which saves our time.

- To assist the staff in capturing the effort spent on their respective working areas.

- To utilize resources in an efficient manner by increasing their productivity through automation.

- It satisfies the user requirement.

- Be easy to understand by the user and operator.

- Be easy to operate.

- Have a good user interface

- Be expandable

- Delivered on schedule within the budget.

5.2 Conclusion

Our project is only a humble venture to satisfy the needs to manage the project work. Several user-friendly coding has also been adopted. This package shall prove to be a powerful package in satisfying all the requirements of the school. The objective of software planning is to provide a framework that enables the manager to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

At the end it is concluded that we have made effort on following points:

- A description of the background and context of the project and its relation to work already done in the area.

- The description of Purpose, Scope, and applicability.

- We define the problem on which we are working in the project.
- We describe the requirement Specifications of the system and the actions that can be done on these things.

- We included features and operations in detail, including screen layouts.

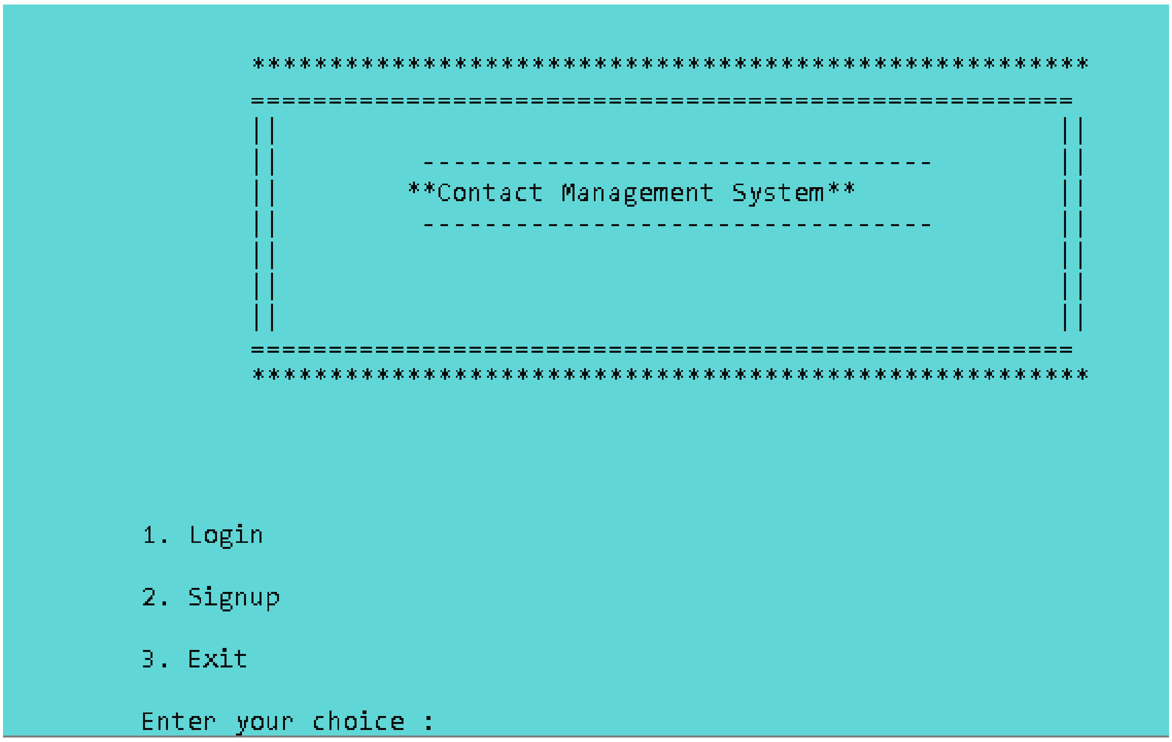
- Finally the system is implemented and tested according to test cases.

CHAPTER 6

Appendix

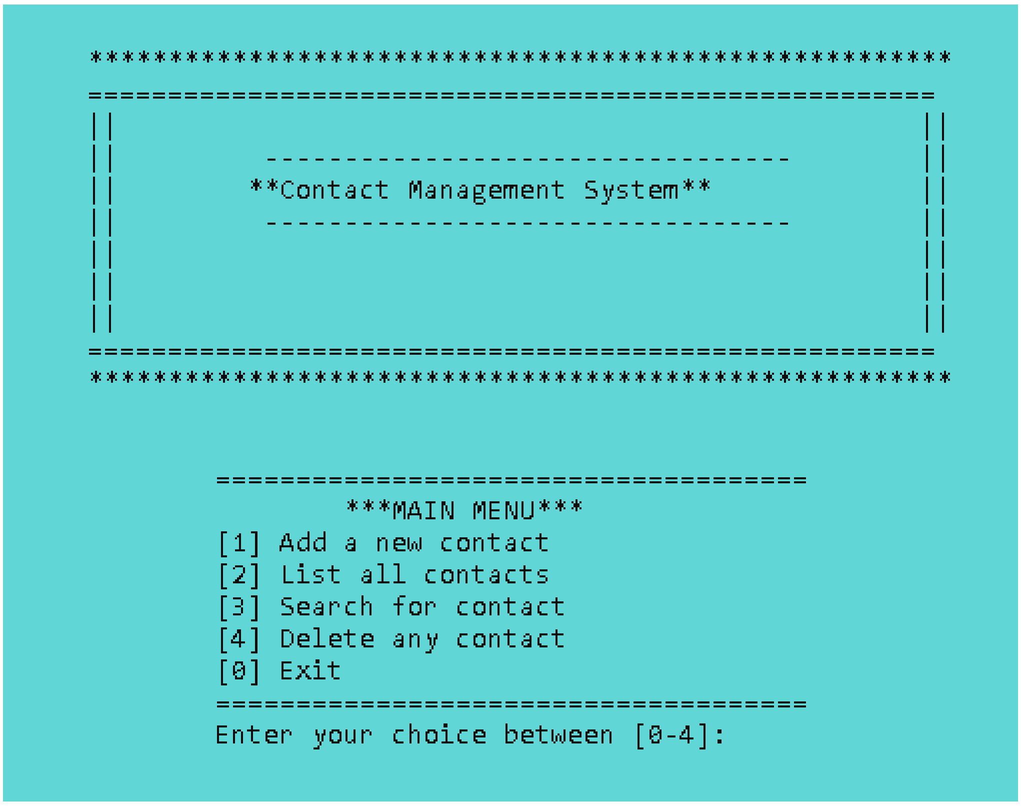
6.1 Screenshots

Login page:



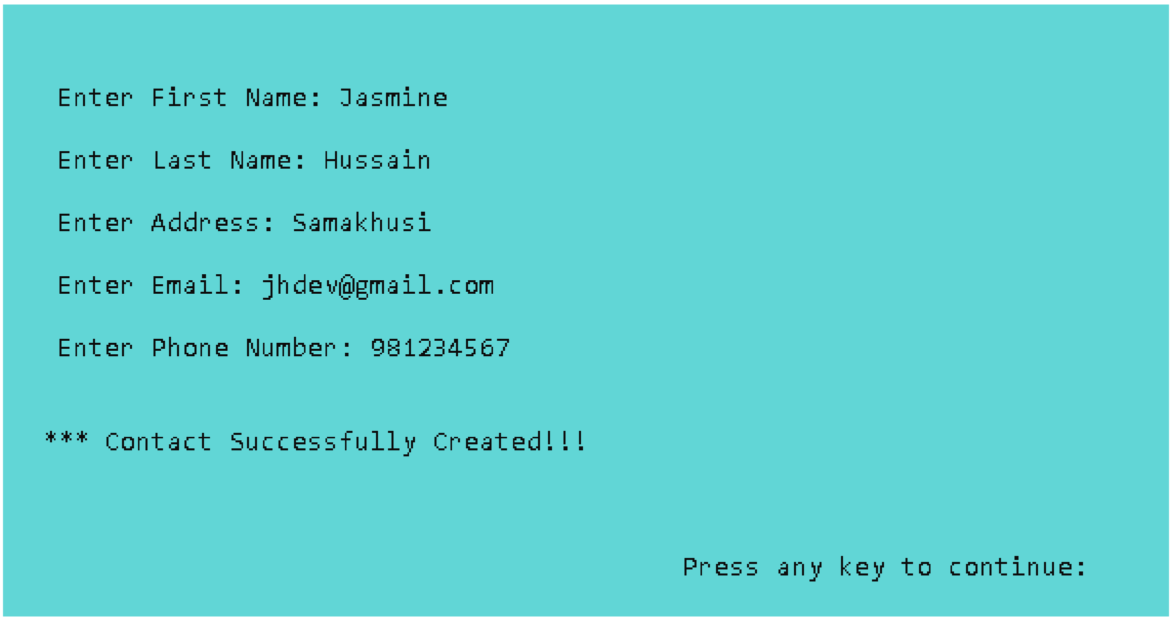
```
                *******************************************************
                ======================================================
                ||                                                   ||
                ||         ----------------------------------        ||
                ||        **Contact Management System**              ||
                ||         ----------------------------------        ||
                ||                                                   ||
                ||                                                   ||
                ||                                                   ||
                ======================================================
                *******************************************************



         1. Login

         2. Signup

         3. Exit

         Enter your choice :
```

Menu Page:



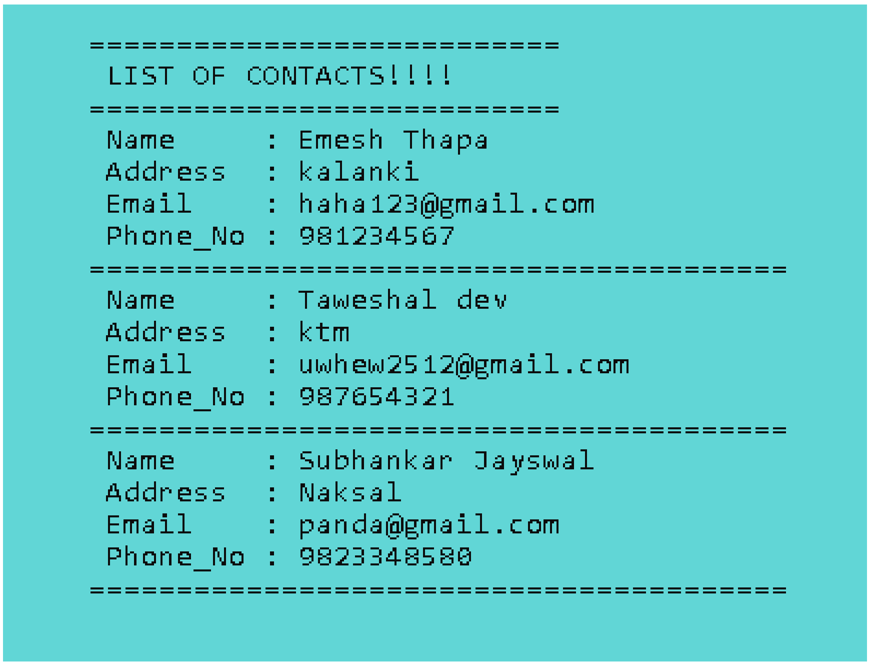
```
    *******************************************************
    ======================================================
    ||                                                 ||
    ||          ---------------------------------      ||
    ||         **Contact Management System**           ||
    ||          ---------------------------------      ||
    ||                                                 ||
    ||                                                 ||
    ||                                                 ||
    ======================================================
    *******************************************************



            ======================================
                    ***MAIN MENU***
            [1] Add a new contact
            [2] List all contacts
            [3] Search for contact
            [4] Delete any contact
            [0] Exit
            ======================================
            Enter your choice between [0-4]:
```

Add contact:



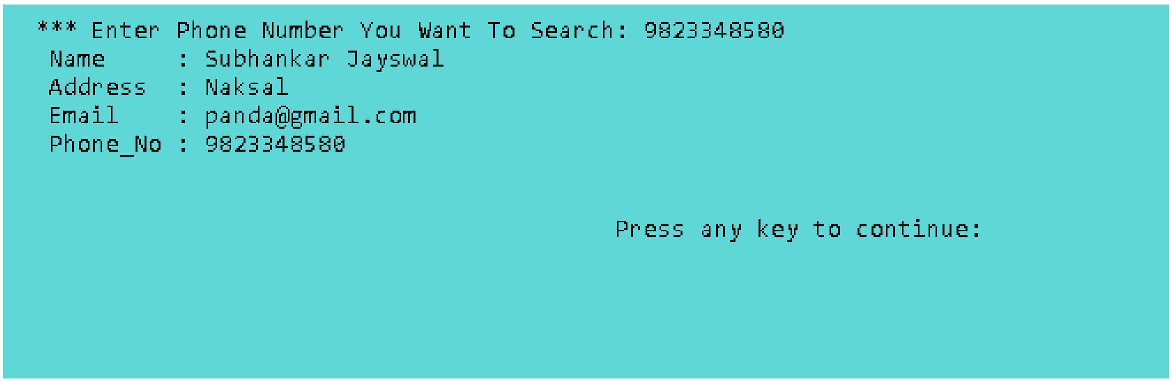
```

  Enter First Name: Jasmine

  Enter Last Name: Hussain

  Enter Address: Samakhusi

  Enter Email: jhdev@gmail.com

  Enter Phone Number: 981234567


 *** Contact Successfully Created!!!



                                         Press any key to continue:
```

List of contact:



```
===========================
 LIST OF CONTACTS!!!!
===========================
 Name      : Emesh Thapa
 Address   : kalanki
 Email     : haha123@gmail.com
 Phone_No  : 981234567
========================================
 Name      : Taweshal dev
 Address   : ktm
 Email     : uwhew2512@gmail.com
 Phone_No  : 987654321
========================================
 Name      : Subhankar Jayswal
 Address   : Naksal
 Email     : panda@gmail.com
 Phone_No  : 9823348580
========================================
```

Search for contact:



```
*** Enter Phone Number You Want To Search: 9823348580
 Name     : Subhankar Jayswal
 Address  : Naksal
 Email    : panda@gmail.com
 Phone_No : 9823348580


                                         Press any key to continue:
```

6.3 Source code:

```
#include<iostream>
#include<fstream>
#include<stdlib.h>
#include<string.h>
#include<conio.h>

using namespace std;
class Title{
        public :
                void gettitle(){
                system("cls");
     system("color B0");
     cout <<
"\n\n\n\n\n\n\n\n\n\n\t\t\t\t\t\t************************************************\n";
     cout <<
"\t\t\t\t\t\t=========================================================\n";
     cout << "\t\t\t\t\t\t\t||                                                ||\n";
     cout << "\t\t\t\t\t\t\t||      ------------------------------------       ||\n";
     cout << "\t\t\t\t\t\t\t||     **Contact Management System**      ||\n";
     cout << "\t\t\t\t\t\t\t||      ------------------------------------       ||\n";
     cout << "\t\t\t\t\t\t\t||                                                ||\n";
     cout << "\t\t\t\t\t\t\t||                                                ||\n";
     cout << "\t\t\t\t\t\t\t||                                                ||\n";
cout<<"\t\t\t\t\t\t\t=====================================================\n";
cout<<"\t\t\t\t\t\t\t********************************************************\n\n\n";
     //cout << " \n\n\n\t\t\t\t\t\t Press any key to continue:";
    //   getch();
                }
};

class Details {
```

```
private:
    char FName[50];
    char LName[50];
    char Address[50];
    char email[50];
    long long phone_No;

public:
    void getdata() {
            system("cls");
            Title head;
        head.gettitle();
        cout << "\n\t\t Enter First Name: ";
        fflush(stdin);
        cin.get(FName, 50);

        cout << "\n\t\t Enter Last Name: ";
        fflush(stdin);
        cin.get(LName, 50);

        cout << "\n\t\t Enter Address: ";
        fflush(stdin);
        cin.get(Address, 50);

        cout << "\n\t\t Enter Email: ";
        fflush(stdin);
        cin.get(email, 50);

        cout << "\n\t\t Enter Phone Number: ";
        fflush(stdin);
        cin >> phone_No;
    }

    void showcontact() {
            //system("cls");
        cout << "\t\t Name     : " << FName << " " << LName << endl;
        cout << "\t\t Address  : " << Address << endl;
        cout << "\t\t Email    : " << email << endl;
        cout << "\t\t Phone_No : " << phone_No;
    }

    void writeonfile() {
        ofstream F1;
        F1.open("CMS.dat",ios::out|ios::app|ios::binary);
        if (!F1.is_open()) {
            cout << "\t\tError opening file!" << endl;
            return;
        }

        F1.write(reinterpret_cast<char*>(this), sizeof(*this));
        cout << "\n\n\t\t*** Contact Successfully Created!!!\n";
        F1.close();
    }

    void readfromfile() {

            system("cls");
            //system("cls");
            Title head;
        head.gettitle();
        ifstream F2("CMS.dat", ios::binary|ios::in);
        if (!F2.is_open()) {
            cout << "\t\tError opening file!" << endl;
            return;
        }

        cout << "\t\t=========================" << endl;
        cout << "\t\t LIST OF CONTACTS!!!!" << endl;
        cout << "\t\t==============================" << endl;

        while (F2.read(reinterpret_cast<char*>(this), sizeof(*this))) {
            showcontact();
            cout << "\n\t\t============================================\n";
        }
        F2.close();
    }

    void searchfromfile() {
            system("cls");
            Title head;
        head.gettitle();
        long long phone;
        cout << "\t\t*** Enter Phone Number You Want To Search: ";
        cin >> phone;

        ifstream F3("CMS.dat", ios::binary);
        if (!F3.is_open()) {
            cout << "\t\tError opening file!" << endl;
            return;
        }

        bool found = false;
        while (F3.read(reinterpret_cast<char*>(this), sizeof(*this))) {
            if (phone_No == phone) {
                showcontact();
                found = true;
                break;
            }
        }

        if (!found)
            cout << "\t\t*** Number Not Found!!!" << endl;
        F3.close();
    }

    void deletefromfile() {
            system("cls");
            Title head;
        head.gettitle();
        long long phone;
        cout << "\t\tEnter Phone Number To delete Contact: ";
        cin >> phone;

        ifstream F4("CMS.dat", ios::binary);
        ofstream F5("temp.dat", ios::binary);

        if (!F4.is_open() || !F5.is_open()) {
            cout << "\t\tError opening file!" << endl;
            return;
        }

        bool deleted = false;
        while (F4.read(reinterpret_cast<char*>(this), sizeof(*this))) {
            if (phone_No != phone) {
                F5.write(reinterpret_cast<char*>(this), sizeof(*this));
            } else {
                deleted = true;
            }
        }

        F4.close();
        F5.close();

        remove("CMS.dat");
        rename("temp.dat", "CMS.dat");
```

```
        if (deleted)
            cout << "\t\tContact Deleted" << endl;
        else
            cout << "\t\tContact Not Found" << endl;
    }
};

class LoginManager {
public:
    bool authenticate() {

        string username, password;
        cout << "\n\n\n\n\n\n\n\n\t\t\t\t\t\t Username : ";
        cin >> username;
        cout << "\n\t\t\t\t\t\t Password : ";
        cin >> password;

        ifstream login_file("Login_Data.dat");
        if (!login_file.is_open()) {
            cout << "\t\tError opening file!" << endl;
            return false;
        }

        string file_user, file_pass;
        while (login_file >> file_user >> file_pass) {
            if (file_user == username && file_pass == password) {
                cout << "\n\n\t\t\t\t\t\t Login successful.\n";
                login_file.close();
                getch();
                return true;
            }
        }
        cout << "\n\n\t\t\t\t\t\t Invalid username or password.\n";
        login_file.close();
        return false;
    }

    void signup() {
        ofstream signup_data;
        string username, password;
        cout << "\n\n\t\t\t\t\t\t Enter new Username : ";
        cin >> username;
        cout << "\n\n\t\t\t\t\t\t Enter new Password : ";
        cin >> password;

        signup_data.open("Login_Data.dat", ios::out | ios::app);
        if (!signup_data.is_open()) {
            cout << "\n\n\t\t\t\t\t\t Error: Unable to create an account." << endl;
            return;
        }
        signup_data << username << ' ' << password << endl;
        cout << "\n\n\t\t\t\t\t\t Signup successfully." << endl;
        signup_data.close();
    }
};

int loginMenu() {
    int choice;
    cout << "\n\n\t\t\t\t\t\t 1. Login";
    cout << "\n\n\t\t\t\t\t\t 2. Signup";
    cout << "\n\n\t\t\t\t\t\t 3. Exit";
    cout << "\n\n\t\t\t\t\t\t Enter your choice : ";
    cin >> choice;
    return choice;
}

int main() {
    // int loginChoice;
    // LoginManager loginManager;
    while (true) {
            flag:
            system("cls");
        Title head;
        head.gettitle();
       /* cout<< " \n\n\n\t\t\t\t\t\t\t Press any key to continue:";
         getch(); */
         int loginChoice;
         LoginManager loginManager;

        loginChoice = loginMenu();
        switch (loginChoice) {
            case 1:
                if (!loginManager.authenticate()) {
                    cout << "\n\n\t\t\t\t\t\t Authentication failed. Try again.\n";
                    getch();
                } else {
                    {
                        system("cls");
                        Details contact;
                        int choice;

                        while (true) {
                                    system("cls");
                              Title head;
                              head.gettitle();
                       //   cout << "\n\n\n\n\n\\n\n\n\t\t*** CONTACT MANAGEMENT
SYSTEM!!! ***\n";
                           cout <<
"\t\t\t\t\t\t\t\t========================================\n";
                           cout << "\t\t\t\t\t\t\t\t\t***MAIN MENU***\n";
                           cout << "\t\t\t\t\t\t\t\t[1] Add a new contact\n";
```

```
              cout << "\t\t\t\t\t\t\t\t[2] List all contacts\n";
              cout << "\t\t\t\t\t\t\t\t[3] Search for contact\n";
              cout << "\t\t\t\t\t\t\t\t[4] Delete any contact\n";
              cout << "\t\t\t\t\t\t\t\t[0] Exit\n";
              cout <<
"\t\t\t\t\t\t\t\t========================================\n";
              cout << "\t\t\t\t\t\t\t\tEnter your choice between [0-4]: ";
              cin >> choice;

              switch (choice) {
                case 1:
                  system("cls");
                  contact.getdata();
                  contact.writeonfile();
                  cout<< " \n\n\n\t\t\t\t\t\t Press any key to continue:";
                  getch();
                  break;

                case 2:
                  system("cls");
                  contact.readfromfile();
                  cout<< " \n\n\n\t\t\t\t\t\t Press any key to continue:";
                  getch();
                  break;

                case 3:
                  system("cls");
                  contact.searchfromfile();
                  cout<< " \n\n\n\t\t\t\t\t\t Press any key to continue:";
                  getch();
                  break;

                case 4:
                  system("cls");
```

```
                        contact.deletefromfile();
                        cout<< " \n\n\n\t\t\t\t\t\t Press any key to continue:";
                        getch();
                        break;

                    case 0:
                        system("cls");
                        cout << "\t\t\t ******* Thank you For Using CMS *******\n\n";
                        cout<< " \n\n\n\t\t\t\t\t\t Press any key to continue:";
                        getch();
                        exit(0);
                        break;

                    default:
                        cout << "\n Invalid choice! Please enter a valid option.\n";
                        cout<< " \n\n\n\t\t\t\t\t\t Press any key to continue:";
                        getch();
                        break;
                }
            }
        }
    }
    break;

case 2:
    loginManager.signup();
    break;

case 3:
    exit(0);
    break;

default:
    cout << "\n\n\t\t\t\t\t Invalid choice. Please try again.\n";
```

```
            getch();
            goto flag;
            break;
        }
    }

    return 0;
}
```

References

- Lafore, R. (n.d.). Object-Oriented Programming in C++. Galgotia publication.
- Programiz. (n.d.). Retrieved from https://www.programiz.com/c-programming
- Roberts, Bucky. “Thenewboston - YouTube.” Www.youtube.com, Bucky Roberts April. 2011, youtube.com/user/thenewboston. Accessed 28 July 2023.
- Jain, Sandeep. “GeeksforGeeks | a Computer Science Portal for Geeks.” GeeksforGeeks, Sandeep Jain, 27 Mar. 2023, geeksforgeeks.org. Accessed 3 Aug. 2023.