**ALLSAFE**

**EMPLOYEE MANAGEMENT SYSTEM**



Presented By

Apurv Singh Gautam (15070121506)

Rushin Tilva (15070121547)

**INTRODUCTION**

Employee Management system is an application that enables a company to create and store Employee Records. This application is helpful to department of the organization which maintains data of employees related to an organization. This application also secures the data of Employees from unauthorized access by Encrypting the data in the file.

Our Employee management system contains several functions which are as follow………

1) Adding Employee details,

2) Displaying Employee details,

3) Searching Employee details with the help of Employee ID,

4) Delete the existing Employee details,

5) Modify the existing Employee details,

6) Encrypt the Employee details File,

7) Decrypt the Employee details File.

C++ Concepts

We have used several C++ concepts which are classes, manipulators, functions, graphics, file handling.

**HEADER FILES:-**

A header file is a file with extension .h which contains C++ function declarations and macro definitions to be shared between several source files.

This project contains following Header Files:

1. #include<iostream> - To provide basic input/output for C++

2. #include<fstream> - Represent inpout/output file streams

3. #include<string> - To perform basic String operations

4. #include<iomanip> - To manipulate input/output

5. #include<conio.h> - To provide console input/output

6. #include<process.h> - Contains functions working with process

8. #include<windows.h> - To perform various built-in Windows function

**CLASSES:-**

Class is a way of binding data and its associated function together. It allows data (and function) to be hidden, if necessary, from external use. When defining class, we are creating new abstract data type that can be treated like any other built-in data type.

Generally a class specification has to parts:

1. Class Declaration

2. Class function definition

Syntax of Defining a Class is:

Class class\_name

{

private:

variables;

public:

functions;

};

The class declaration is similar to structure declaration.

In this project only one class is used i.e. class allsafe.

This class controls every operation that is performed in this project like adding data, displaying data, modifying data and deleting data.

**MANIPULATORS:-**

Manipulators are operators that are used to format the data display. The most commonly used manipulator is endl.

The endl manipulator, when used in an output statement, causes a line feed to be inserted .It has the same effect as using the newline character “\n”.

In this project User-Defined Manipulator is also used.

Syntax of Defining a User-Defined Manipulator:

ostream & manipulator(ostream & ostr) //Outstream Manipulator  
 {  
 set of statements;  
 return ostr;   
 }

istream & manipulator(istream & istr) //Instream Manipulator  
{  
 set of statements;  
 return istr;  
}

The User-Defined Manipulator used in this project is:

ostream &nm(ostream &strname)

{

strname<<"\*WELCOME TO ALLSAFE EMPLOYEE DATABASE MANAGEMENT SYSTEM \* ";

return strname;

}

**FUNCTIONS:-**

The basic work of function is to divide a program into parts. It basically states the structured programming. Another advantage of function is that it is possible to reduce the size of program by calling and using them at different places.

The Whole program is governed by main function. Calling of any function or returning the value is done by or in main function. Function call can be done by passing arguments or without passing arguments. We have used both types.

Passing Arguments:-As far as argument is concerned there are two basic type of passing an argument

1) Call by Value

2) Call by Reference

• In call by value, Compiler passed the value of variable by making a copy.

• In call by reference, Compiler doesn't makes copy it passes the address of argument.

In this project following functions are used:

1. void choice() – Contains Menu-Driven Part

2. void login() – Contains Login Part

3. void read() – To Read File

4. void write() – To Write File

5. void modify() – To Modify Data

6. void modifydata() – To Modify Entries from Data

7. void getdata() – To Take Input from User

8. void search() – To Search Data

9. void deleted() – To Delete Data

10. void display() – To Display Front Page

11. void displaydata() – To Display Each Employees Data

13. void displayall() – To Display Single Employee Data

14. void encrypt() – To Encrypt File

15. void decrypt() – To Decrypt File

16. void gotoxy(short,short) – To specify x-axis and y-axis (gotoxy)

17. int getempid() – To Return Employee ID

:New Concepts Learned:

**FILE HANDLING:-**

So far, we have been using the iostream standard library, which provides cin and cout methods for reading from standard input and writing to standard output respectively. This tutorial will teach you how to read and write from a file. This requires another standard C++ library called fstream, which defines three new data types:

A file must be opened before you can read from it or write to it. Either the ofstream or fstream object may be used to open a file for writing or ifstream object is used to open a file for reading purpose only. Following is the standard syntax for open() function, which is a member of fstream, ifstream, and ofstream objects.

void open(const char \*filename, ios::openmode);

Here, the first argument specifies the name and location of the file to be opened and the second argument of the open() member function defines the mode in which the file should be opened.

|  |  |
| --- | --- |
| **Data Type** | **Description** |
| ofstream | This data type represents the output file stream and is used to create files and to write information to files. |
| ifstream | This data type represents the input file stream and is used to read information from files. |
| fstream | This data type represents the file stream generally, and has the capabilities of both ofstream and ifstream which means it can create files, write information to files, and read information from files. |

|  |  |
| --- | --- |
| **Mode Flag** | **Description** |
| ios::app | Append mode. All output to that file to be appended to the end. |
| ios::in | Open a file for reading. |
| ios::out | Open a file for writing. |
| Ios::binary | Opens in binary mode. |

**Closing a File:-**

When a C++ program terminates it automatically closes flushes all the streams, release all the allocated memory and close all the opened files. But it is always a good practice that a programmer should close all the opened files before program termination. Following is the standard syntax for close() function, which is a member of fstream, ifstream, and ofstream objects.

void.close();

Following File Handling Syntax are used in this project:

1. fstream f – To create file pointer named f

2. f.open("input.dat",ios::app | ios::out | ios::binary | ios::in) – To Open file input.dat.

3. f.write((char\*)&c,sizeof(c)) – To Write contents of object c to file

4. f.read((char\*)&c,sizeof(c)) – To Read from File

5. f.tellg() – Tells the current position pointer in file

6. f.seekg(0) – Sets the pointer of file to specified value in braces

**Graphics:-**

We have added text colour to several statements in our program as well as background colour to different screens.

To change the background and text colour, syntax is:

System(“COLOR A0”);

Color Codes are:

0 = Black  
1 = Blue  
2 = Green  
3 = Aqua  
4 = Red  
5 = Purple  
6 = Yellow  
7 = White  
8 = Grey  
9 = Light Blue  
A = Light Green  
B = Light Aqua  
C = Light Red  
D = Light Purple  
E = Light Yellow  
F = Bright White

SYSTEM FLOW CHART

END PAGE

DECRYPT DATA

ENCRYPT DATA

DELETE DATA

SEARCH DATA

MODIFY DATA

INPUT DATA

Admin

No

Yes

**Is Valid?**

Username & Password