**1. INTRODUCTION**

**1.1 IMPORTANCE**

A payroll system is a software designed to organize all the tasks of employee payment and the filing of employee taxes. These tasks can include keeping track of hours. Calculating wages, withholding taxes and deductions, printing and delivering checks and paying employment taxes to the government.

Payroll software often requires very little input wage information then the software calculates the information and performs withholdings automatically. Most payroll software is automatically updated whenever a tax flaw change and will remind employers when to file various tax forms.

**1.2 BACKGROUND**

The Payroll Management System deals with the financial aspects of employee's salary, allowances, deductions, gross pay, net pay etc. and generation of pay-slips for a specific period. The outstanding benefit of Payroll Management System is its easy implement other advantages of Payroll Management System and its extensive features and reports.

**1.3 PROBLEM DEFINATION**

It may be difficult to decide which system to choose, but there are some factors to keep in mind when deciding. First, analyze the size of your business and decide how much you are willing to spend on payroll processing.

While it is possible for smaller businesses to handle payroll duties in-house through a manual process, much time can be wasted while attempting to calculate everything correctly. One miscalculation and the business owner could find themselves in legal or financial trouble. Mid-sized companies with up to employers’ benefit, by investing in a payroll system.

**1.4 OBJECTIVES**

Payroll Management System gives you the power to:

* Manage Employee Information efficiently.
* Define the earnings, deductions, leave etc.
* Generate Pay-Slip at the convenience by a click.
* Generate and Manage the Payroll according to the Salary Structure assigned to the employee.
* Generate all the Reports related to employee, attendance/leave, payroll etc
* Manage your own Security.

**1.5 SCOPE**

**1) Recurring payroll services**

• Gross pay calculation (basic salary, wage supplements, occasional payments, cost reimbursements, etc.).

• Calculation of payroll related taxes and contributions.

• Recording and processing of garnishments (NO SUCH WORD IN ENGLISH DO YOU MEAN DISBURSEMENTS) and other deductions.

• Preparation of payroll slim and other outputs are broken down by employees or by cost centers for managerial and operational use.

• Data recording and processing in connection with voluntary pension and health insurance funds, reporting to the pension funds.

• Recording of wage and labor-related data (registration of personal information. holiday and sick leave balances).

• Providing data and information for posting into General Ledger.

• Data reports and certificates related to payroll processing (to the tax and social security authorities. the Statistical Office. etc.).

**2) Full range of Social Insurance management.**

**3) Tasks related to newcomers and departing employees.**

**4) Annual services connected to payroll processing and social security administration.**

**5) Preparation and submission of tax declarations.**

**6) Payroll disbursement services:**

• Movement of all payments, calculated during the payment processing cycle (net salary. tax. social security. etc.).

• Specified data reports for the Client's accounting.

**7) Other services upon separate written request including:**

• Cafeteria management.

• Tax advisory.

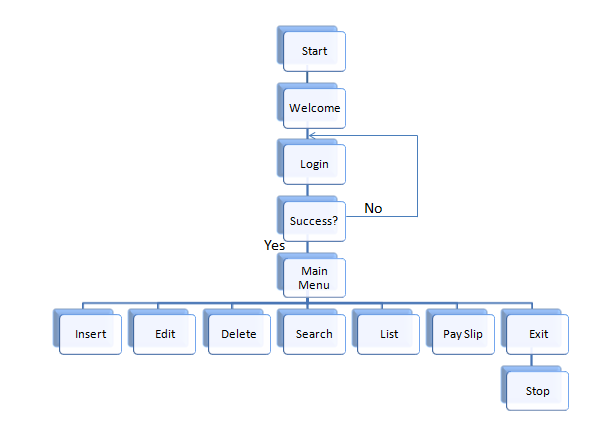
• Labor management.

**2. ARCHITECTURE**

**2.1 SYSTEM ARCHITECTURE**

**Figure 2.1.1**

The System Architecture which consists of details to implement all of the necessary tools to create a Payroll Management System. These are the essential requirements for the program.

****

**2.2 MODULES**

**2.2.1 Admin Login Module**

Here in this module call, user is prompted to enter the login credentials. The Login Module is a portal that allows users to type a user name and password to login. This module is no longer available to users after they have logged in. The Login Module appears to users next to the introduction module.

**2.2.2 Data Entry Module**

After you selected data entry from the main menu you land on this screen. In this module, Data of the employee are inserted. The Fields required here are Name, Id of the employee, Designation, Age, Years of experience, No. of working hours, Loan Status if any.

All the required data is processed and the salary, earnings and deductions of the employee are calculated and finally stored in the files for permanent storage.

**2.2.3 Storing and Retrieving Data Records Module**

Records of all the employees are to be maintained and the records are stored in Files and the information is retrieved from the files. All the Records are separated by new lines, and each field of an individual record is separated by ‘tab’.

**3. SYSTEM REQUIREMENTS**

**1.6.1 Software Requirements**

Operating system: Windows 7 or Later versions of windows

IDE: Dev C++

**1.6.2 Hardware Requirements**

Processor : Intel Pentium core or later versions

Hard Disc : 120GB

RAM : 1 GB or More

**3.1 C++**

C++ is a high-level language that has developed in mid of 1970’s. It is built or derived from C language. It has object-oriented features which allows programmer to create objects within the code. This creation of objects makes programming easier, efficient in terms of time and space. C++ offers power and flexibility over the language.

C++ is a general-purpose programming language and widely used now a days for competitive programming. It has imperative, object-oriented and generic programming features. C++ runs on lots of platform like Windows, Linux, Unix, Mac etc.

**3.2 C++ Classes and Objects**

**Class**

The building block of C++ that leads to Object Oriented programming is a Class. It is a user defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A class is like a blueprint for an object.

When you define a class, you define a blueprint for a data type. This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

A class definition starts with the keyword class followed by the class name; and the class body, enclosed by a pair of curly braces. A class definition must be followed either by a semicolon or a list of declarations.

The keyword **public** determines the access attributes of the members of the class that follows it. A public member can be accessed from outside the class anywhere within the scope of the class object. You can also specify the members of a class as **private** or **protected** which we will discuss in a sub-section.

**Object**

**Object** is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e., an object is created) memory is allocated. A class provides the blueprints for objects, so basically an object is created from a class. We declare objects of a class with exactly the same sort of declaration that we declare variables of basic types.

**3.3 C++ FUNCTIONS**

A function is a group of statements that together perform a task. Every C++ program has at least one function, which is **main()**, and all the most trivial programs can define additional functions.

You can divide up your code into separate functions. How you divide up your code among different functions is up to you, but logically the division usually is such that each function performs a specific task.

A function **declaration** tells the compiler about a function's name, return type, and parameters. A function **definition** provides the actual body of the function. The C++ standard library provides numerous built-in functions that your program can call.

A C++ function definition consists of a function header and a function body. Here are all the parts of a function −

* **Return Type** − A function may return a value. The **return\_type** is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return\_type is the keyword **void**.
* **Function Name** − This is the actual name of the function. The function name and the parameter list together constitute the function signature.
* **Parameters** − A parameter is like a placeholder. When a function is invoked, you pass a value to the parameter. This value is referred to as actual parameter or argument. The parameter list refers to the type, order, and number of the parameters of a function. Parameters are optional; that is, a function may contain no parameters.
* **Function Body** − The function body contains a collection of statements that define what the function does.

**3.4 FILE HANDLING IN C++**

## Opening a File

A file must be opened before you can read from it or write to it. Either **ofstream**or **fstream** object may be used to open a file for writing. And 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.

## Closing a File

When a C++ program terminates it automatically 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.

## Writing to a File

While doing C++ programming, you write information to a file from your program using the stream insertion operator (<<) just as you use that operator to output information to the screen. The only difference is that you use an **ofstream** or **fstream** object instead of the **cout** object.

## Reading from a File

You read information from a file into your program using the stream extraction operator (>>) just as you use that operator to input information from the keyboard. The only difference is that you use an **ifstream** or **fstream** object instead of the **cin** object.

**4. IMPLEMENTATION**

**4.1 PROBLEM STATEMENT**

It may be difficult to decide which system to choose, but there are some factors to keep in mind when deciding. First, analyze the size of your business and decide how much you are willing to spend on payroll processing.

While it is possible for smaller businesses to handle payroll duties in-house through a manual process, much time can be wasted while attempting to calculate everything correctly. One miscalculation and the business owner could find themselves in legal or financial trouble. Mid-sized companies with up to employers benefit, by investing in a payroll system.

**4.2 WORKING PROCEDURE**

**STEP 1**

Main function is executed first and the control to the followed by code.

The function ‘intro()’ got executed and the welcome message is displayed and when the user proceed to the next step by pressing any key.

**STEP 2**

User prompted with a login screen here and a user with valid credentials can have the access to the software.

**STEP 3**

After the user with the valid credentials logged in, the data of previous employee records are retrieved.

**STEP 4**

After the user with the valid credentials logged in and successful retrieval, the user land on the home screen (the Main menu), and based on the choice of the user he land on the requested screen.

**STEP 5**

After the job done, all the modified or created data is stored in the files and the software is successfully exited.

**4.3 CODE**

#include<iostream>

#include<conio.h>

#include<cstdlib>

#include<cstring>

#include<cstdio>

#include<windows.h>

#include<unistd.h>

using namespace std;

class employee{

private:

// Variables for employee details

char name[30];

char id[5];

char designation[10];

int age;

int ctc;

int experience;

// Utility functions

void waitForEnter(void){

cout<<"\n\n\n Press enter to go back \n\n";

cin.get();

cin.get();

}

// Functions to perform desired actions

void listEmployees(void){ //To list total employees with Name, Id and Designation

system("cls");

FILE \*file;

file= fopen("data.txt", "r");

cout<<"\n\t List of Employees\n";

cout<<"\n----------------------------------------------";

cout<<"\n NAME | ID | DESIGNATION\n";

cout<<"----------------------------------------------";

while(fscanf(file, "%s %s %s %d %d %d", &name[0], &id[0] , &designation[0], &age, &ctc, &experience)!= EOF)

cout<<"\n"<<name<<"\t\t"<<id<<"\t\t"<<designation;

fclose(file);

waitForEnter();

}

void showDetails(void){ //Displays all details according to Employee's id

system("cls");

FILE \*file;

char checkId[5];

cout<<"\n\nEnter Employee ID: ";

cin>>checkId;

file= fopen("data.txt", "r");

while(fscanf(file, "%s %s %s %d %d %d", &name[0], &id[0] , &designation[0], &age, &ctc, &experience)!=EOF)

if(strcmp(checkId,id)==0){

cout<<"\n---------------------";

cout<<"\nName: "<<name;

cout<<"\n---------------------";

cout<<"\nId: "<<id;

cout<<"\n---------------------";

cout<<"\nDesignation: "<<designation;

cout<<"\n---------------------";

cout<<"\nAge: "<<age;

cout<<"\n---------------------";

cout<<"\nCTC: "<<ctc;

cout<<"\n---------------------";

cout<<"\nExperience: "<<experience;

cout<<"\n---------------------";

}

fclose(file);

waitForEnter();

}

void editExisting(void){ //edits Designation and CTC of an employee

system("cls");

char checkId[5];

cout<<"\nEnter employee id: ";

cin>>checkId;

char newDesignation[10];

cout<<"\n-----------------------------";

cout<<"\nEnter new designation: ";

cin>>newDesignation;

int newCtc;

cout<<"------------------------------";

cout<<"\nEnter new CTC: ";

cin>>newCtc;

FILE \*file, \*tempfile;

file= fopen("data.txt", "r");

tempfile= fopen("temp.txt", "w");

while(fscanf(file, "%s %s %s %d %d %d", &name[0], &id[0] , &designation[0], &age, &ctc, &experience)!=EOF){

if(strcmp(checkId, id)==0)

fprintf(tempfile, "%s %s %s %d %d %d \n", name, id, newDesignation, age, newCtc, experience );

else

fprintf(tempfile, "%s %s %s %d %d %d \n", name, id, designation, age, ctc, experience );

}

fclose(file);

fclose(tempfile);

int isRemoved= remove("data.txt");

int isRenamed= rename("temp.txt", "data.txt");

waitForEnter();

}

void addNewEmployee(void){ //adding records

system("cls");

cout<<"\n----------------------------------------";

cout<<"\n Enter First Name of Employee: ";

cin>>name;

cout<<"\n----------------------------------------";

cout<<"\n Enter Employee ID [max 4 digits]: ";

cin>>id;

cout<<"\n----------------------------------------";

cout<<"\n Enter Designation: ";

cin>>designation;

cout<<"\n----------------------------------------";

cout<<"\n Enter Employee Age: ";

cin>>age;

cout<<"\n----------------------------------------";

cout<<"\n Enter Employee CTC: ";

cin>>ctc;

cout<<"\n----------------------------------------";

cout<<"\n Enter Employee Experience: ";

cin>>experience;

cout<<"\n----------------------------------------";

char ch;

cout<<"\nEnter 'y' to save above information\n";

cin>>ch;

if(ch=='y'){

FILE \*file;

file= fopen("data.txt","a");

fprintf(file, "%s %s %s %d %d %d \n", name, id, designation, age, ctc, experience );

fclose(file);

cout<<"\nNew Employee has been added to database\n";

}

else

addNewEmployee();

waitForEnter();

}

void deleteEmployeeDetails(void){ //removing records

system("cls");

char checkId[5];

cout<<"\n----------------------------------";

cout<<"\nEnter Employee Id To Remove: ";

cin>>checkId;

char ch;

cout<<"----------------------------------";

cout<<"\n\n\n\n\nCONFIRMATION\nEnter 'y' To Confirm Deletion \n";

cin>>ch;

if(ch=='y'){

FILE \*file, \*tempfile;

file= fopen("data.txt", "r");

tempfile= fopen("temp.txt", "w");

while(fscanf(file, "%s %s %s %d %d %d", &name[0], &id[0] , &designation[0], &age, &ctc, &experience)!=EOF)

if(strcmp(checkId, id)!=0)

fprintf(tempfile, "%s %s %s %d %d %d \n", name, id, designation, age, ctc, experience );

fclose(file);

fclose(tempfile);

int isRemoved= remove("data.txt");

int isRenamed= rename("temp.txt", "data.txt");

cout<<"\nRemoved Successfully\n";

waitForEnter();

}

else

deleteEmployeeDetails();

}

public:

// Function to serve as end point

void options(void){ //menu

int login(); //login declaration

login();//login screen

while(true){

system("cls");

// Options to choose an action

cout<<"\n\t\t\t>>>>>>>>> PAYROLL MANAGEMENT SYSTEM <<<<<<<<<";

cout<<"\n";

cout<<"\n\t\t\t------------------------------------------------";

cout<<"\n\t\t\tENTER 1: To View List of Employees";

cout<<"\n\t\t\t------------------------------------------------";

cout<<"\n\t\t\tENTER 2: To View Employee Details";

cout<<"\n\t\t\t------------------------------------------------";

cout<<"\n\t\t\tENTER 3: To Modify Existing Employee Details";

cout<<"\n\t\t\t------------------------------------------------";

cout<<"\n\t\t\tENTER 4: To Add New Employee Details";

cout<<"\n\t\t\t------------------------------------------------";

cout<<"\n\t\t\tENTER 5: To Remove an Employee Details";

cout<<"\n\t\t\t------------------------------------------------";

cout<<"\n\t\t\tENTER 0: To Exit ";

cout<<"\n\t\t\t------------------------------------------------";

cout<<"\n\n\t\t\t Please Enter Your Choice: ";

// Taking the action input

int choice;

cin>>choice;

// Calling relevant function as per choice

switch (choice) {

case 0:

system("CLS");

cout<<"\n\nPAYROLL MANAGEMENT SYSTEM";

Sleep(10);

return;

case 1:

listEmployees();

break;

case 2:

showDetails();

break;

case 3:

editExisting();

break;

case 4:

addNewEmployee();

break;

case 5:

deleteEmployeeDetails();

break;

default:

cout<<"\n Sorry! I don't understand that! \n";

break;

}

}

}

};

int main(){

// Call the options function

employee e;

e.options();

return 0;

}

int login(){ //login procedure

string pass ="";

char ch;

cout <<"\n\n\n\n\t\t\t\t\tPAYROLL MANAGEMENT SYSTEM";

cout <<"\n\n\n\n\n\t\t\t\t\tEnter Your Password:";

ch = \_getch();

while(ch != 13){//character 13 is enter

pass.push\_back(ch);

cout << '\*';

ch = \_getch();

}

if(pass == "pass"){

cout<<"\n\n\n\t\t\t\t\tLOADING \n\t\t\t\t\t";

for(int a=1;a<8;a++) // Change 'a<?' to how many \* you want

{

Sleep(500);

cout << "...";

}

cout << "\n\n\n\t\t\t\t\tAccess Granted!! \n\n\n";

system("PAUSE");

system("CLS");

}else{

cout << "\nAccess Aborted...\n";

login();

}

}

**4.4 PROJECT ADT**

void gotoXY(int,int); - Sets the cursor position to the specified location.

void Cdelay(int); - Adds the time delay of specified milliseconds.

void border(int, int,int, int); - Draws the borders with specified coordinates.

void borderNoDelay(int, int,int, int); - Similar to border but with no time delay.

void loginFrame(int, int, int, int); - Draws the Login frame .

void intro(); - Introduction of our project is displayed here.

void login(); - Login Authentication goes here.

void menu(); - Main Menu.

void insert(); - Creates a new record.

void edit(); - Edit a record.

void editmenu(); - Display edit options.

void editname(int); - Employee Name will be edited.

void editcode(int); - Employee Code will be edited.

void editdes(int); - Employee Designation will be edited.

void editexp(int); - Employee Experience will be edited.

void editage(int); - Employee age will be edited.

void editsalary(int); - Employee salary will be edited.

void list(); - Lists all the records available.

void deletes(); - Delete a specific record.

void search(); - Search for the record.

void setWindowSize(int,int); - Set Output window to desired size.

void saverecords(); - All the records created/modified will be saved.

void getrecords(); - All the available records will be retrived.

bool isFilePresent(); - Checks weather the Database File is present or not.

void displayPayslip(); - Displays the Pay Slip of the specified employee.

**5. TESTING**

5.1 INTRODUCTION

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. In the testing process we test the actual system in an organization and gather errors from the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed to ensuring that the system works accurately and efficiently.

In the testing process we test the actual system in an organization and gather errors from the new system and take initiatives to correct the same. All the front-end and back-end connectivity are tested to be sure that the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently.

The main objective of testing is to uncover errors from the system. For the uncovering process we have to give proper input data to the system. So we should have more conscious to give input data. It is important to give correct inputs to efficient testing.

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus, the system testing is a confirmation that all is correct and an opportunity to show the user that the system works. Inadequate testing or non-testing leads to errors that may appear few months later.

The testing process focuses on logical intervals of the software ensuring that all the statements have been tested and on the function intervals (i.e.,) conducting tests to 14 uncover errors and ensure that defined inputs will produce actual results that agree with the required results. Testing has to be done using the two common steps Unit testing and Integration testing. In the project system testing is made as follows:

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion. Here system maintenance is done every month or year based on company policies, and is checked for errors like runtime errors, long run errors and other maintenances like table verification and reports.

**5.2. Test Cases**

**5.2.1. Test case for Admin Login**

|  |  |
| --- | --- |
| **Test case #01** | **Priority (H,L): High** |
| **Test Objective** | Admin Login |
| **Test Description** | Admin has to login to the system before modifying or creating data |
| **Expected Results** | Login should be success and success message should display |
| **Actual Results** | Login Success |
| **Problems / Issues** | If the message ‘Login Success’ hasn’t been displayed then ‘Check User Name or Password’ message should be displayed. |
| **Result** | Pass |

Table 5.1 Test case for Admin Login

**5.2.2. Test case for Data Entry**

|  |  |
| --- | --- |
| **Test case #02** | **Priority (H, L): High** |
| **Test Objective** | Employee Data entered |
| **Test Description** | Employee details are to be entered personal and professional details like name, age, designation, working hours, years of experience, loan status. |
| **Expected Results** | Recorded Success message should be displayed. |
| **Actual Results** | Data Recorded Successfully |
| **Problems / Issues** | If the message ‘Data Recorded Successfully’ hasn’t been displayed then ‘Error’message should be displayed. |
| **Result** | Pass |

Table 5.2 Test case for Data Entry

**5.2.3. Test case for Saving Records to file**

|  |  |
| --- | --- |
| **Test case #03** | **Priority (H,L): High** |
| **Test Objective** | Modified data of the employee should be saved |
| **Test Description** | Data of the employee modified or newly created should be saved in the file. |
| **Expected Results** | Data should be saved into the file in the specified format. |
| **Actual Results** | Data Saved successfully. |
| **Problems / Issues** | If saving the data is failed appropriate error message should be displayed. |
| **Result** | Pass |

Table 5.3 Test case for Saving Records

**5.2.4. Test case for Retrieving Data from file.**

|  |  |
| --- | --- |
| **Test case #04** | **Priority (H,L): High** |
| **Test Objective** | Previously saved data of the employee should be restored. |
| **Test Description** | Data of the employee that was previously save is to be loaded into the software. |
| **Expected Results** | Data should be displayed |
| **Actual Results** | Data restored successfully |
| **Problems / Issues** | If restoring the data is failed appropriate error message should be displayed. |
| **Result** | Pass |

Table 5.4 Test case for Retrieving Data

**6. SCREENSHOTS**

Fig 6.1 Home Screen

This is the home screen of the Payroll Management System, where we must enter the password to enter the system.

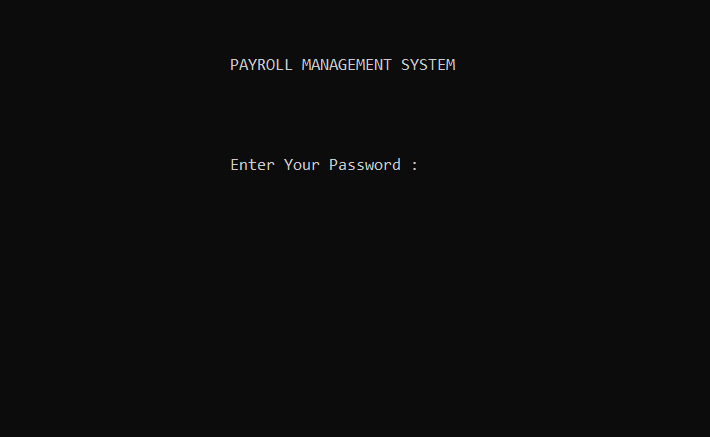


Fig 6.2 Loading Screen

The Payroll Management System has loading screen to load the program.

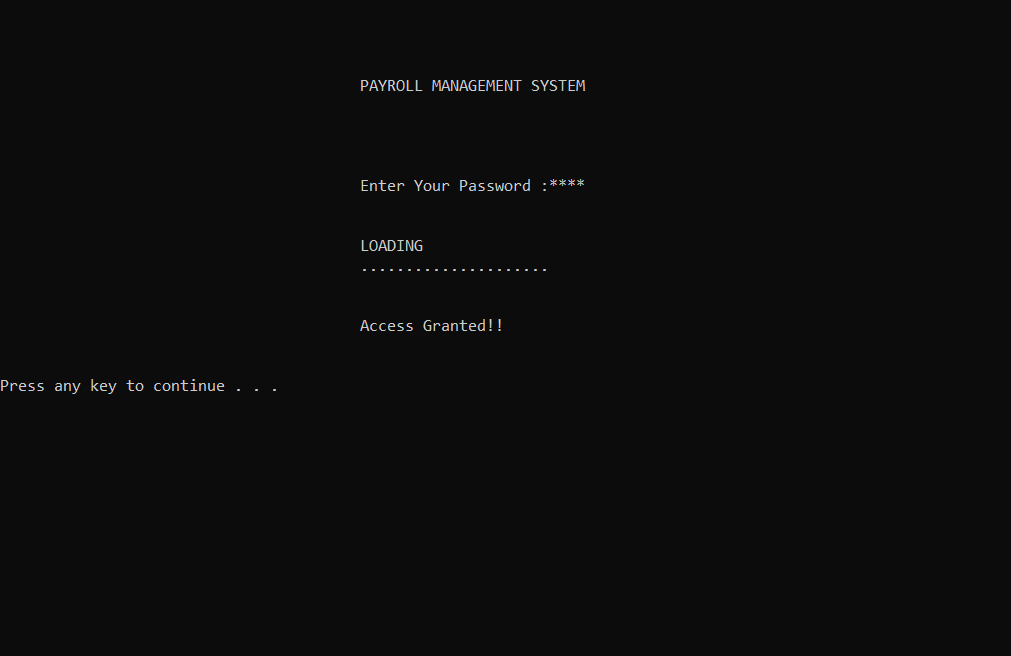


Fig 6.3 Main Menu Screen

The list of the menu and the program options the user can select.

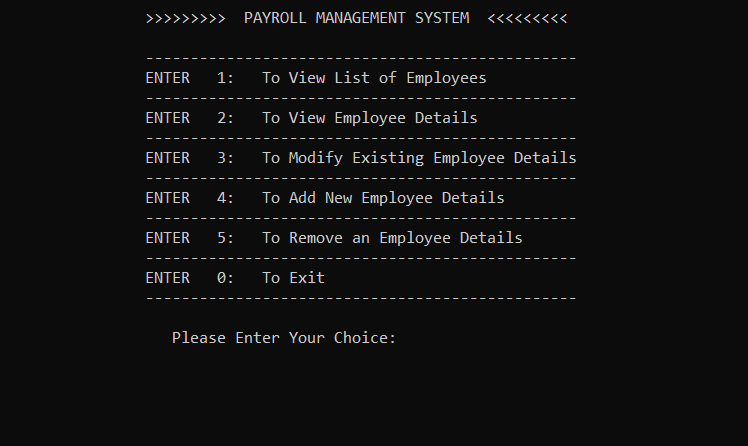


Fig 6.4 Searching for a Record

To search for an employees record for the convenience of the user.

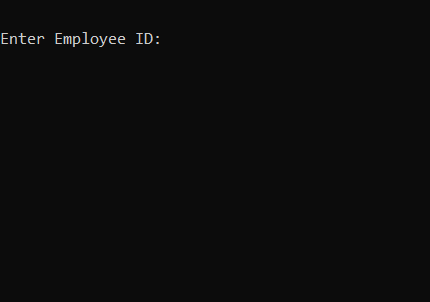


Fig 6.5 Adding a New Record

To Add a Record, we should first add the first name of the employee.

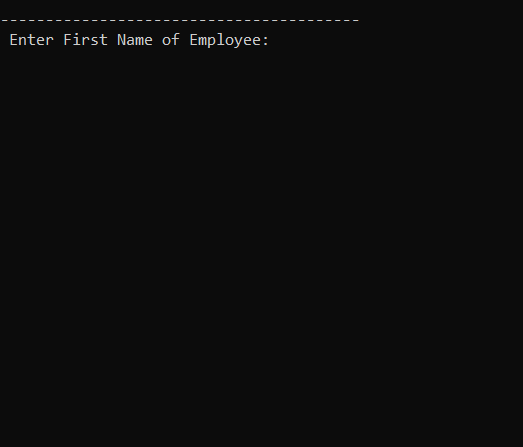


Fig 6.6 Removing Record

To remove an employee record we must enter the employee id.

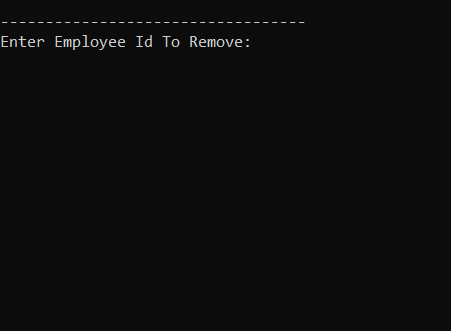
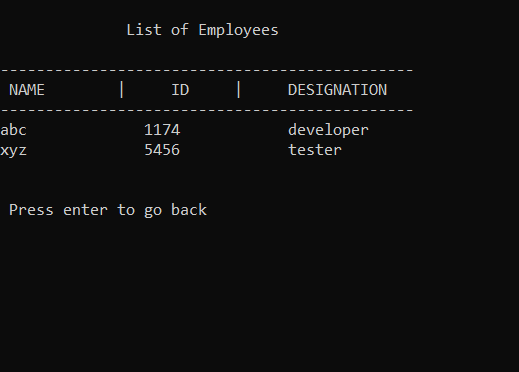


Fig 6.7 List of Employee Table

The full list of employee records and it will display the id as well as the designation.



**CONCLUSION**

The delivered system “PAYROLL MANAGEMENT SYSTEM” software developed for a company has been designed to achieve maximum efficiency and reduce the time taken to handle the payroll activity. It is designed to replace an existing manual record system thereby reducing time taken for calculations and for storing data. The system used C++ as front end.

The system is strong enough to withstand regressive daily operations under conditions where the data is cleaned over a certain time of span. The implementation of the system in the organization will considerably reduce data entry, time and also provide readily calculated reports.

It provides Automation of the entire system improves the efficiency; it provides a friendly Interface which proves to be better when compared to the existing system updating of information becomes so easier.

The System has adequate scope for modification in future if it is necessary.

* Continuous tracking of their activities within the company of every employee can be provided with their individual login credentials.
* Auto update of the details like bank loan update can be provided.
* Administration control over the software is to be improved.

Ex: Administration can able to add the employee credentials such that the employee can know their status and companies’ administration legally.

* Databases can be used for easier and efficient access.
* The proposed system is capable of handling only 100 employee records and can be improved.

**REFERENCES**

**Textbook References**

* Let Us C by Yashavant Kanetkar.
* C Programming A Modern Approach By K.N. King.
* Head First C by David Griffiths.
* Programming Principles and Practice Using C++ By Bjarne Stroustrup.

**Other References**

1. https://www.Stackoverflow.com
2. https://www.Tutorialspoint.com
3. https://www.Geekforgeeks.com