



Green University of Bangladesh
Department of Computer Science and Engineering (CSE)
Faculty of Sciences and Engineering
Semester: (Summer, Year:2022), B.Sc. in CSE (Day)

Course Title: Employee Record Management
Course Code: 106 Section: DB

Lab Project Name: Employee Management System

Student Details

Name		ID
1.	Gazi Faria	213902067

Submission Date: 11-09-2022

Course Teacher's Name: Farhana Akter Sunny

[For Teachers use only: Don't Write Anything inside this box]

Lab Project Status

Marks: Signature: Comments:

..... Date:

Table of Contents

Chapter 1 Introduction 3 1.1 Introduction 3 1.2 Design Goals/Objective 4

Chapter 2 Design/Development/Implementation of the Project 5 2.1 Implementation 5 2.2 Output 8

Chapter 3 Conclusion 10 4.1 Learning Outcome 10 4.2 Scope of Future Work 10

References 10

Chapter 1

Introduction

1.1 Introduction

Today almost every activity in the world is controlled by computer powered software programs. This trend has been dominated by engineering applications in the past. But as life becomes more complex, each field of human interaction is attacked by different Software systems, such as real time, business, simulation, embedded, web-based, personal and Most recently, artificial intelligence software and so on.

According to the above information, efficient software can now be used to control and maintain the records of an employee. This project focuses on designing efficient and reliable software which controls the transactions of an employee's records

1.2 Goals/Objective

Employee records management system is a web application that automates all kinds of activities in employee records. And this software is based on C programming language. The purpose of this software is to manage all employee data.

Typically, this includes data processing, employee management, and accounting Management. I'm trying to develop this software to keep track of employee information, name, salary, age, address of the employee. This means a process that has a type system that provides benefits provide information to all employees without any complications.

Design OF Employee Management System

- 1. Add Employees Records**
- 2. List Employees Records**
- 3. Modify Employees Records**
- 4. Delete Employees Records**
- 5. Search Employee's Records**
- 6. Sort Employee's Records**
- 6. Exit Button**

Chapter 2

Implementation of the Project

2.1 Implementation

```
/* Employee Record Management
```

```
this project is done by Gazi Faria,
```

```
*/
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#include <math.h>
```

```
#include <stdbool.h>
```

```
typedef struct
```

```
{
```

```
    char name[40];
```

```
    int age;
```

```
    char address[20];
```

```
    double basic_salary;
```

```
} Employee;
```

```
#define TOTAL_EMP 100
```

```
int i = 0, n = 0;
```

```
char choice;
```

```
int read = 0, record = 0;
```

```
FILE *file1, *file2;
```

```
Employee emp[TOTAL_EMP];
```

```
void fileload(Employee emp[]);
```

```
void add_employee(Employee employees[]);
```

```
void list_employee(Employee employees[]);
```

```
void modify_employee(Employee employees[]);
```

```
void delete_employee(Employee employees[]);
```

```
void search_employee(Employee employees[]);
```

```
void sort_employee_name(Employee employees[]);
```

```
void sort_employee_age(Employee employees[]);
```

```
void sort_employee_address(Employee employees[]);
```

```
void sort_employee_salary(Employee employees[]);
```

```
int main()
```

```
{
```

```
    fileload(emp);
```

```
    while (1)
```

```
    {
```

```
        printf("\n :::::::::::::::::::: |EMPLOYEES RECORD MANAGEMENT|  
:::::::::::::::::: \n");
```

```
        printf("1. Add Employee's Records \n");
```

```
        printf("2. List Employee's Records \n");
```

```
        printf("3. Modify Employee's Records \n");
```

```
printf("4. Delete Employee's Records \n");

printf("5. Search Employee's Records \n");

printf("6. Sort Employee's Records \n");

printf("7. Exit System \n");

printf(" \n \n Your Choice: ");

fflush(stdin);

choice = getchar();

switch (choice)

{

case '1':

    system("cls");

    add_employee(emp);

    break;

case '2':

    system("cls");

    list_employee(emp);

    break;

case '3':

    system("cls");

    modify_employee(emp);

    break;

case '4':

    system("cls");

    delete_employee(emp);

    break;

case '5':

    system("cls");

    search_employee(emp);
```

```

        break;

case '6':

    system("cls");


    printf(" \n :::::::::::::::::::: |EMPLOYEES RECORD MANAGEMENT|
:::::::::::::::::: \n");

    printf("1. Sort by Name \n");

    printf("2. Sort by Age \n");

    printf("3. Sort by Address \n");

    printf("4. Sort by Basic Salary \n");

    printf("5. Return Back \n");

    printf(" \n \n Your Choice: ");

    fflush(stdin);

    choice = getchar();

    switch (choice)

    {

case '1':

        system("cls");


        sort_employee_name(emp);

        break;

case '2':

        system("cls");


        sort_employee_age(emp);

        break;

case '3':

        system("cls");

```



```
        sort_employee_address(emp);

        break;

case '4':

    system("cls");


    sort_employee_salary(emp);

    break;

case '5':

    system("cls");

    break;

default:

    system("cls");

    printf("\nInvalid choice & returning back ...\n");

    system ("pause");

}

break;

case '7':

    system("cls");

    exit(0);

    break;

default:

    system("cls");

    printf("Invalid Choice");

    break;

}

}

return 0;
```

```
}
```

```
void fileload(Employee emp[])
```

```
{
```

```
    file2 = fopen("CCsvf.csv", "r+");
```

```
    if (file2 == NULL)
```

```
    {
```

```
        printf("Error opening file2\n");
```

```
        exit(1);
```

```
    }
```

```
    do
```

```
    {
```

```
        read = fscanf(file2, "%39[^,],%d,%19[^,],%lf\n", emp[record].name, &emp[record].age,  
emp[record].address, &emp[record].basic_salary);
```

```
        if (read == 4)
```

```
            record++;
```

```
        if (read != 4 && !feof(file2))
```

```
        {
```

```
            printf("file format error\n");
```

```
            exit(1);
```

```
        }
```

```
        if (ferror(file2))
```

```
        {
```

```
            printf("Error reading from file2\n");
```

```
            exit(1);
```

```
        }
```

```
} while (!feof(file2));
```

```
fclose(file2);
```

```
}
```

```
void add_employee(Employee emp[])
```

```
{
```

```
file1 = fopen("CCsvf.csv", "w+");
```

```
char another = 'y';
```

```
while (another == 'y')
```

```
{
```

```
printf("\nEmployee %d info:\n", record + 1);
```

```
fflush(stdin);
```

```
printf("Name: ");
```

```
scanf("%[^\\n]*c", emp[record].name);
```

```
fflush(stdin);
```

```
printf("Age: ");
```

```
scanf("%d", &emp[record].age);
```

```
fflush(stdin);
```

```
printf("Address: ");
```

```
scanf("%[^\\n]*c", emp[record].address);
```

```
fflush(stdin);
```

```
printf("Basic Salary: ");
```

```
fflush(stdin);
```

```
scanf("%lf", &emp[record].basic_salary);
```

```
printf("\n");
```

```
printf("Add another Employee? (y/n): ");
```

```

    fflush(stdin);

    another = getchar();

    record++;

}

for (i = 0; i < record; i++)

{

    fprintf(file1, "%s,%d,%s,%.2lf\n", emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

    if (ferror(file1))

    {

        printf("Error writing to file1\n");

        exit(1);

    }

}

fclose(file1);

printf("\n%d Employee added to file1\n", record);

}

void list_employee(Employee employees[])

{

    printf("%-7s %-20s %-7s %-20s %-10s\n", "Serial", "Name", "Age", "Address", "Salary");

    for (i = 0; i < record; i++)

    {

        printf("%-7d %-20s %-7d %-20s %-10.2lf\n", i, emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

    }

    system("pause");

}

```

```

void modify_employee(Employee emp[]){

    printf("Enter the serial number of the employee you want to modify: ");

    fflush(stdin);

    scanf("%d", &n);

    printf("\nEmployee %d info:\n", n);

    fflush(stdin);

    printf("Name: ");

    scanf("%[^\\n]*c", emp[n].name);

    fflush(stdin);

    printf("Age: ");

    scanf("%d", &emp[n].age);

    fflush(stdin);

    printf("Address: ");

    scanf("%[^\\n]*c", emp[n].address);

    fflush(stdin);

    printf("Basic Salary: ");

    fflush(stdin);

    scanf("%lf", &emp[n].basic_salary);

    printf("\n");


    file1 = fopen("CCsvf.csv", "w+");

    for (i = 0; i < record; i++)

    {

        fprintf(file1, "%s,%d,%s,%.2lf\\n", emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

        if (ferror(file1))

        {

            printf("Error writing to file1\\n");

            exit(1);

```

```

    }

}

printf("Employee %d modified\n", n);

fclose(file1);

system("pause");

}

void delete_employee(Employee emp[]){

    printf("Enter the serial number of the employee you want to delete: ");

    fflush(stdin);

    scanf("%d", &n);

    for (i = n; i < record; i++)

    {

        strcpy(emp[i].name, emp[i + 1].name);

        emp[i].age = emp[i + 1].age;

        strcpy(emp[i].address, emp[i + 1].address);

        emp[i].basic_salary = emp[i + 1].basic_salary;

    }

    record--;

    file1 = fopen("CCsvf.csv", "w+");

    for (i = 0; i < record; i++)

    {

        fprintf(file1, "%s,%d,%s,%.2lf\n", emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

        if (ferror(file1))

        {

            printf("Error writing to file1\n");

            exit(1);

```

```

    }

}

fclose(file1);

printf("Employee %d deleted\n", n);

system("pause");

}

```

```

void search_employee(Employee emp[]){

    printf("Enter the name of the employee you want to search: ");

    fflush(stdin);

    char name[40];

    scanf("%[^\\n]%*c", name);

    for (i = 0; i < record; i++)

    {

        if (strcmp(emp[i].name, name) == 0)

        {

            printf("%-7s %-20s %-7s %-20s %-10s\n", "Serial", "Name", "Age", "Address",
"Salary");

            printf("%-7d %-20s %-7d %-20s %-10.2lf\n", i, emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

            system("pause");

        }

    }

}

```

```

void sort_employee_name(Employee emp[]){

    int i, j;

```

```

Employee temp;

for (i = 0; i < record; i++)

{

    for (j = 0; j < record - 1; j++)

    {

        if (strcmp(emp[j].name, emp[j + 1].name) > 0)

        {

            strcpy(temp.name, emp[j].name);

            temp.age = emp[j].age;

            strcpy(temp.address, emp[j].address);

            temp.basic_salary = emp[j].basic_salary;

            strcpy(emp[j].name, emp[j + 1].name);

            emp[j].age = emp[j + 1].age;

            strcpy(emp[j].address, emp[j + 1].address);

            emp[j].basic_salary = emp[j + 1].basic_salary;

            strcpy(emp[j + 1].name, temp.name);

            emp[j + 1].age = temp.age;

            strcpy(emp[j + 1].address, temp.address);

            emp[j + 1].basic_salary = temp.basic_salary;

        }

    }

}

file1 = fopen("CCsvf.csv", "w+");

for (i = 0; i < record; i++)

{

    fprintf(file1, "%s,%d,%s,%.2lf\n", emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

    if (ferror(file1))

    {

```



```

        printf("Error writing to file1\n");

        exit(1);

    }

}

fclose(file1);

printf("Employee sorted by name\n");

system("pause");

}

```

```

void sort_employee_age(Employee emp[]){

    int i, j;

    Employee temp;

    for (i = 0; i < record; i++)

    {

        for (j = 0; j < record - 1; j++)

        {

            if (emp[j].age > emp[j + 1].age)

            {

                strcpy(temp.name, emp[j].name);

                temp.age = emp[j].age;

                strcpy(temp.address, emp[j].address);

                temp.basic_salary = emp[j].basic_salary;

                strcpy(emp[j].name, emp[j + 1].name);

                emp[j].age = emp[j + 1].age;

                strcpy(emp[j].address, emp[j + 1].address);

                emp[j].basic_salary = emp[j + 1].basic_salary;

                strcpy(emp[j + 1].name, temp.name);

                emp[j + 1].age = temp.age;

```

```

        strcpy(emp[j + 1].address, temp.address);

        emp[j + 1].basic_salary = temp.basic_salary;
    }

}

}

file1 = fopen("CCsvf.csv", "w+");

for (i = 0; i < record; i++)

{

    fprintf(file1, "%s,%d,%s,%.2lf\n", emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

    if (ferror(file1))

    {

        printf("Error writing to file1\n");

        exit(1);

    }

}

fclose(file1);

printf("Employee sorted by age\n");

system("pause");

}

```

```

void sort_employee_address(Employee emp[]){

    int i, j;

    Employee temp;

    for (i = 0; i < record; i++)

    {

        for (j = 0; j < record - 1; j++)

        {

            if (strcmp(emp[j].address, emp[j + 1].address) > 0)

```

```

    {

        strcpy(temp.name, emp[j].name);

        temp.age = emp[j].age;

        strcpy(temp.address, emp[j].address);

        temp.basic_salary = emp[j].basic_salary;

        strcpy(emp[j].name, emp[j + 1].name);

        emp[j].age = emp[j + 1].age;

        strcpy(emp[j].address, emp[j + 1].address);

        emp[j].basic_salary = emp[j + 1].basic_salary;

        strcpy(emp[j + 1].name, temp.name);

        emp[j + 1].age = temp.age;

        strcpy(emp[j + 1].address, temp.address);

        emp[j + 1].basic_salary = temp.basic_salary;

    }

}

}

file1 = fopen("CCsvf.csv", "w+");

for (i = 0; i < record; i++)

{

    fprintf(file1, "%s,%d,%s,%.2lf\n", emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

    if (ferror(file1))

    {

        printf("Error writing to file1\n");

        exit(1);

    }

}

fclose(file1);

printf("Employee sorted by address\n");

```

```
system("pause");  
  
}
```

```
void sort_employee_salary(Employee emp[]){  
  
    int i, j;  
  
    Employee temp;  
  
    for (i = 0; i < record; i++)  
  
    {  
  
        for (j = 0; j < record - 1; j++)  
  
        {  
  
            if (emp[j].basic_salary > emp[j + 1].basic_salary)  
  
            {  
  
                strcpy(temp.name, emp[j].name);  
  
                temp.age = emp[j].age;  
  
                strcpy(temp.address, emp[j].address);  
  
                temp.basic_salary = emp[j].basic_salary;  
  
                strcpy(emp[j].name, emp[j + 1].name);  
  
                emp[j].age = emp[j + 1].age;  
  
                strcpy(emp[j].address, emp[j + 1].address);  
  
                emp[j].basic_salary = emp[j + 1].basic_salary;  
  
                strcpy(emp[j + 1].name, temp.name);  
  
                emp[j + 1].age = temp.age;  
  
                strcpy(emp[j + 1].address, temp.address);  
  
                emp[j + 1].basic_salary = temp.basic_salary;  
  
            }  
  
        }  
  
    }  
  
}
```

```

file1 = fopen("CCsvf.csv", "w+");

for (i = 0; i < record; i++)

{

    fprintf(file1, "%s,%d,%s,%.2lf\n", emp[i].name, emp[i].age, emp[i].address,
emp[i].basic_salary);

    if (ferror(file1))

    {

        printf("Error writing to file1\n");

        exit(1);

    }

}

fclose(file1);

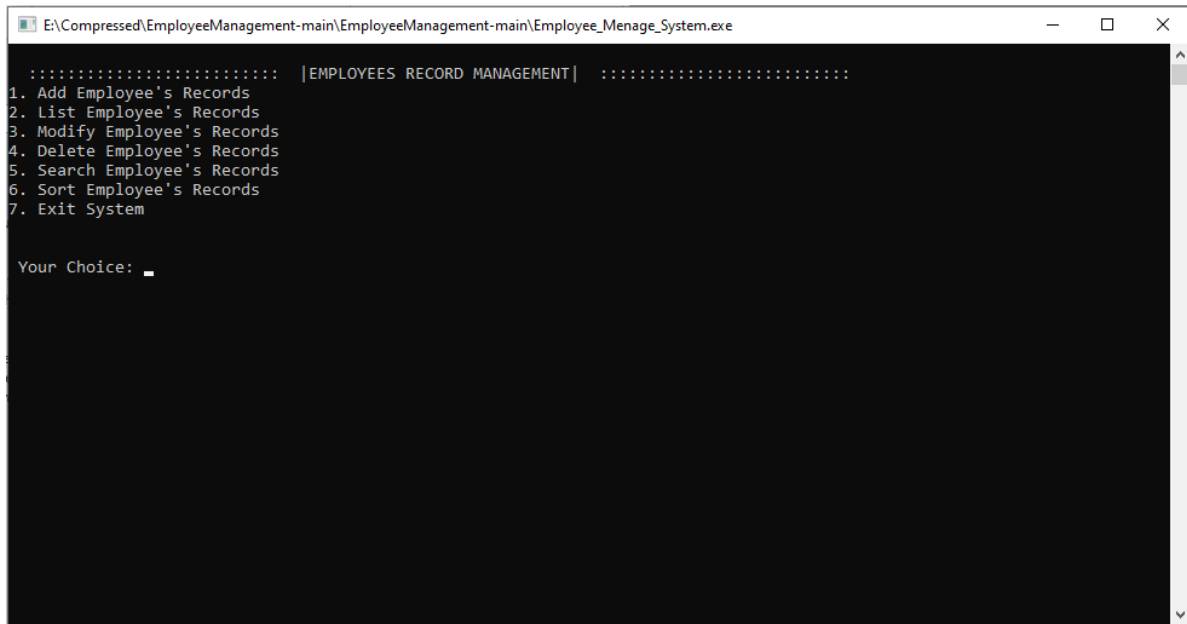
printf("Employee sorted by salary\n");

system("pause");

}

```

2.2 Output



```
E:\Compressed\EmployeeManagement-main\EmployeeManagement-main\Employee_Manage_System.exe

:|EMPLOYEES RECORD MANAGEMENT|:
1. Add Employee's Records
2. List Employee's Records
3. Modify Employee's Records
4. Delete Employee's Records
5. Search Employee's Records
6. Sort Employee's Records
7. Exit System

Your Choice: 
```

Figure 01:: Main Menu interface

Chapter 4

Conclusion

3.1 Learning Outcome

By completing this project, I will be able to understand and visualize the inner workings. The computer system and the architecture and overall concept that drives this C language my project programming. As a programming language, C allows me to write more complex and comprehensive program. I am now able to define and to solve this project manage data structures based on problem domains. Ability to work with text, data, letters and strings. Ability to work with arrays of complex objects. Understanding an idea of object thinking within the framework of a functional model.

3.2 Scope of Future Work

This is an attempt to make the employee information system current inefficient and time consuming the process of finding and storing quality reading material. Currently, clients have to go through a time consuming process to perform aforementioned tasks which cause waste of labor and firm resources. We provide an easy way of searching, finding to employee details. User data are validated and checked for authenticity with the data stored in the system database. All newly created processes will address time consuming, inefficient and inefficient fields existing systems that are wasting the resources of many organizations, such as labor, electricity, equipment, products and services, the management of the company suffers.

References

Six Sigma n.d. : Applying Six Sigma to Software Implementation Projects Retrieved 22 March 2007 from

<http://software.isixsigma.com/library/content/c040915b.asp>

Sommerville, Ian 2004. Object Oriented Design Software

Engineering, 7thEdition .Start your journey the easy way n.d :

Retrieved 4th February 2007 from

<http://www.liverpooljohnlennonairport.com/TravelServices/CarParking.php>

Ramakrishnan, R. and Gehrke, J. 2003. The

Relational Model In Database Management Systems, 3rdEdition|