EMPLOYEE MANAGEMENT SYSTEM

Submitted by

Name of the Student: Semanti Datta
Enrolment Number: 12022002002193
Section: A
Class Roll Number: 90
Stream: CSE Core
Subject: Programming for Problem Solving with C
Subject Code: ESC103(Pr)

Under the supervision of
Prof. Dr. Indrajit De

Academic Year: 2022-26

PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE SECOND SEMESTER

DEPARTMENT OF BASIC SCIENCE AND HUMANITIES
INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA

CERTIFICATE OF RECOMMENDATION

We hereby recommend that the project prepared under our supervision by **SEMANTI DATTA,** entitled Employee Management System be accepted in partial fulfillment of the requirements for the degree of partial fulfillment of the second semester.

Head of the Department Supervisor Basic Sciences and Humanities IEM, Kolkata	Project

1 Introduction:

In this project we create various modules using C to create databases in which we can collect information about employee details, adding an employee, updating about an existing employee, deleting an employee.

1.1 Objective:

To create various C modules for a **Employee Management System** Create a employee's database of his/her details.

1.2 Organization of the Project:

The project is organized mainly in 5 different modules namely adding an employee, updating an employee, deleting an employee and displaying employees. The first module holds the control of the entire portal. It asks us our choice:-

1.Add an employee
2.Update an employee
3.Delete an employee
4.Display employees
5.Exit

If we go for the first option we will get into the "add an employee" module and perform various operations on the file which will result in changes in the "update an employee", "delete an employee" and "display employees" modules. If we go for the second option we will get into the "update an employee" module and perform various operations on the file which will result in changes in the other modules. If we go for the third option we will get into the "delete an employee" module and perform various operations on the Batch.csv file which will result in changes in the "display employees" module. If we go for the fourth option we will get into the "display employees" module and perform various operations on the file then it will display all the names of the current employees present. If you go for the final option we exit the program control.

2 Database Descriptions:

1. "Add an employee" module stores all the information of

an employee that is to be added such as “Employee ID”, “Employee name”, “Employee age”, “Employee salary”.

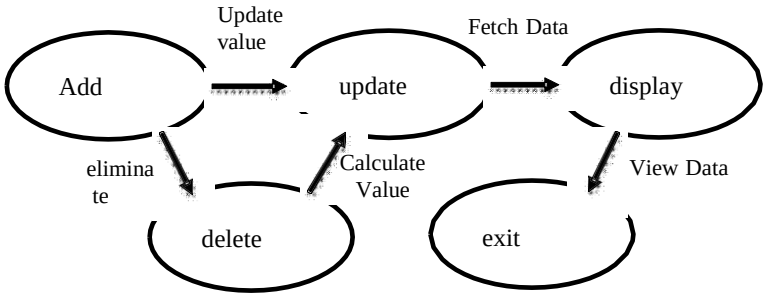
2. “Update an employee” module helps us to update about an employee by providing the stored employee ID
3. “Delete an employee” module allows us to delete the name of a particular employee from the stored list of employees.
4. “Display employees” module shows all the names and details of the employees currently present in the stored list of employees.

FUNCTIONS USED:

1. Struct Employee()
2. Void addEmployee(int)
3. Void updateEmployee(int)
4. Void deleteEmployee(int)
5. Void displayEmployees(int)

3 Data Flow and ER Diagrams:

Demonstrate the dependency of all the C modules written using data flow diagrams



4 Programs:

Provide the python programs of the various modules

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
struct Employee {
int id;
char name[100];
int age;
float salary;
};
void addEmployee(struct Employee *employees, int *count) {
if (*count >= 100) {
printf("Employee database is full. Cannot add more employees.\n");
return;
}
struct Employee newEmployee;
printf("Enter employee details:\n");
printf("Employee ID: ");
scanf("%d", &newEmployee.id);
printf("Employee Name: ");
scanf(" %[^\n]", newEmployee.name);
printf("Employee Age: ");
scanf("%d", &newEmployee.age);
printf("Employee Salary: ");
scanf("%f", &newEmployee.salary);
employees[*count] = newEmployee;
(*count)++;
printf("Employee added successfully.\n");
}
void updateEmployee(struct Employee *employees, int count) {
int employeeId;
int found = 0;
printf("Enter the employee ID to update: ");
scanf("%d", &employeeId);
for (int i = 0; i < count; i++) {
if (employees[i].id == employeeId) {
printf("Enter new details for the employee:\n");
printf("Employee Name: ");
scanf(" %[^\n]", employees[i].name);
printf("Employee Age: ");
scanf("%d", &employees[i].age);
printf("Employee Salary: ");
scanf("%f", &employees[i].salary);
printf("Employee details updated successfully.\n");
found = 1;
break;
}
}
if (!found) {
printf("Employee not found.\n");
}
}
void deleteEmployee(struct Employee *employees, int *count) {
int employeeId;
int found = 0;
```

```
printf("Enter the employee ID to delete: ");
scanf("%d", &employeeId);
for (int i = 0; i < *count; i++) {
if (employees[i].id == employeeId) {
for (int j = i; j < *count - 1; j++) {
employees[j] = employees[j + 1];
}
(*count)--;
printf("Employee deleted successfully.\n");
found = 1;
break;
}
}
if (!found) {
printf("Employee not found.\n");
}
}
void displayEmployees(struct Employee *employees, int count) {
if (count == 0) {
printf("Employee database is empty.\n");
return;
}
printf("Employee Database:\n");
printf("ID\tName\tAge\tSalary\n");
for (int i = 0; i < count; i++) {
printf("%d\t%s\t%d\t%.2f\n", employees[i].id, employees[i].name, employees[i].age,
employees[i].salary);
}
}
int main() {
struct Employee employees[100];
int count = 0;
int choice;
printf("Employee Management System\n");
while (1) {
printf("\nSelect an option:\n");
printf("1. Add an employee\n");
printf("2. Update an employee\n");
printf("3. Delete an employee\n");
printf("4. Display employees\n");
printf("5. Exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);
switch (choice) {
case 1:
addEmployee(employees, &count);
break;
case 2:
updateEmployee(employees, count);
break;
case 3:
deleteEmployee(employees, &count);
break;
case 4:
displayEmployees(employees, count);
```

```
break;
case 5:
printf("Exiting Employee Management System. Goodbye!\n");
exit(0);
default:
printf("Invalid choice. Please select a valid option.\n");
}
}
return 0;
}
```

5 Outputs:

Sample Outputs to demonstrate the functionalities in program

```
Output                                                    Clear
/tmp/C8bGiGjpV6.o
Employee Management System
Select an option:
1. Add an employee
2. Update an employee
3. Delete an employee
4. Display employees
5. Exit
Enter your choice: 1
Enter employee details:
Employee ID: 12345678
Employee Name: Semanti Datta
Employee Age: 19
Employee Salary: 75,000
Employee added successfully.
```

```
Select an option:
1. Add an employee
2. Update an employee
3. Delete an employee
4. Display employees
5. Exit
Enter your choice:2
Enter employee details:birthdate: 30th dec

Employee ID: Employee Name: Employee Age: Employee Salary: Employee added
    successfully.
```

END