EMPLOYEE MANAGEMENT SYSTEM

Submitted By

Name of the Student : **ARUNIMA SAMANTA**
Enrollment Number : **12022002003186**
Section: H
Class Roll Number: 60
Stream: ECE

Subject: Programming for Problem Solving Using C
Subject Code: ESC103(Pr)
Department: Basic Science and Humanities

Under the supervision of
Dr.Swarnendu Ghosh
Mrs.Sumana Sinha

Academic Year: 2022-2026

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 **Dr.Swarnendu Ghosh**, entitled **EMPLOYEE MANAGEMENT SYSTEM** be accepted in partial fulfillment of the requirements for the degree of partial fulfillment of the first semester.

Dr. Prabir Kumar Das

Head of the Department
Basic Sciences and Humanities
IEM , Kolkata

Dr. Swarnendu Ghosh

Project Supervisor

INTRODUCTION

The Employee Management System is a program designed to efficiently manage and organize employee data within an organization. It provides a user-friendly interface for adding, updating, deleting, and displaying employee information. The system keeps track of essential employee details such as their name, employee ID, salary, and job title. With this system, employers can easily maintain an organized database of their employees and perform necessary operations on the data.

The program utilizes a struct called "Employee" to store employee information, including their ID, name, age, and salary. It allows for the addition of new employees, updating existing employee details, deleting employees, and displaying the complete employee database.

The system provides an interactive menu where users can select various options based on their requirements. The available options include adding an employee, updating employee information, deleting an employee, displaying all employees, and exiting the program.

By implementing the Employee Management System, organizations can streamline their employee data management processes, ensuring accurate and up-to-date information while simplifying administrative tasks.

OBJECTIVE

The objective of the Employee Management System is to provide a user-friendly program to manage employee data efficiently. The system allows users to add new employees, update their information, delete them if needed, and easily view the complete employee database. By keeping track of important details such as employee names, IDs, salaries, and job titles, the system aims to help organizations organize and access employee information quickly and easily. Overall, the objective is to simplify and streamline administrative tasks and keep employee records organized and up to date.

REQUIRED DATA TYPE WITH BRIEF DESCRIPTION

int: It is a datatype used to store integer values. It represents whole numbers without any decimal points. For example, employee ID can be stored as an int.

char: It is a datatype used to store individual characters. It can represent letters, numbers, and special characters. For example, the employee's name can be stored as an array of char to hold multiple characters.

float: It is a datatype used to store floating-point numbers. It represents numbers with decimal points. For example, the employee's salary can be stored as a float to accommodate decimal values.

struct: It is a user-defined datatype that allows you to group different variables together under a single name. It enables you to create a custom data structure with multiple components. In this case, the struct Employee is defined to hold employee information such as ID, name, age, and salary.

int*: It is a pointer to an integer variable. Pointers are used to store memory addresses. In this program, the int* is used to pass the count of employees by reference, allowing the function to update the value of the variable in the main program.

char[]: It is an array of characters used to store multiple characters in a contiguous memory block. In this program, arrays of char are used to store employee names.

ALGORITHM

1. Declare the necessary variables and data structures, such as an array of Employee structures and a count variable to keep track of the number of employees.

2. Display a welcome message and start a loop to present the menu options to the user.

3. Inside the loop, display the menu options and prompt the user to enter their choice.

4. Use a switch statement to handle the different menu options.

a. For option 1 (Add an employee) :

- Check if the employee database is full. If it is, display a message indicating that no more employees can be added.
- Prompt the user to enter the employee details (ID, name, age, salary).

- Create a new Employee structure and assign the entered values to its corresponding members.
- Add the new employee to the employees array and increment the count variable.
- Display a success message.

b. For option 2 (Update an employee) :

- Prompt the user to enter the employee ID to update.
- Search for the employee with the matching ID in the employees array.
- If found, prompt the user to enter the updated details (name, age, salary).
- Update the corresponding member values of the found employee structure.
- Display a success message.
- If not found, display a message indicating that the employee was not found.

c. For option 3 (Delete an employee) :

- Prompt the user to enter the employee ID to delete.
- Search for the employee with the matching ID in the employees array.

- If found, shift the remaining employees in the array to remove the employee's data.
- Decrement the count variable.
- Display a success message.
- If not found, display a message indicating that the employee was not found.

d. For option 4 (Display employees) :

- Check if the count variable is 0, indicating an empty employee database.

CODE

```
#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;
}
```

OUTPUT

```
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: 5
Employee Name: Arunima
Employee Age: 20
Employee Salary: 300000
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: 4
Employee Database:
ID      Name    Age     Salary
5       Arunima 20      300000.00
```