|  |
| --- |
| **EMPLOYEE RECORD SYSTEM**  **21CSS101J – PROGRAMMING FOR PROBLEM-SOLVING**  **Mini Project Report**  *Submitted by*  **Student Name :Y.P.V.S.SAI KUMAR [Reg.No.:RA2311003011103**  **B.Tech. CSE – CLOUD COMPUTING**  **SRMIST-01.jpg**  **SCHOOL OF COMPUTING**  **COLLEGE OF ENGINEERING AND TECHNOLOGY**  **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**  **(Under Section 3 of UGC Act, 1956)**  S.R.M. NAGAR, KATTANKULATHUR – 603 203  CHENGALPATTU DISTRICT  **November 2023**  **COLLEGE OF ENGINEERING AND TECHNOLOGY**  **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**  **(Under Section 3 of UGC Act, 1956)**  S.R.M. NAGAR, KATTANKULATHUR – 603 203  **SRMIST-01.jpg**  **BONAFIDE CERTIFICATE**  Certified that Mini project report titled \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is the bonafide work of Reg.No\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.  **SIGNATURE SIGNATURE**  **(GUIDE) (HEAD OF THE DEPARTMENT)** |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S No.** | **Title** | **Page No.** |
| 1 | Problem Statement | 2 |
| 2 | Methodology / Procedure/ Algorithm | 3 |
| 3 | Flowchart | 4 |
| 4 | Coding (C/Python) | 5 |
| 5 | Results/Screenshots | 9 |
| 6 | Results/Screenshots Modules of the proposed work | 13 |
| 7 | Conclusion | 16 |
| 8 | References | 17 |

PAGE1

**1.PROBLEM STATEMENT**

Create a program to demonstrate and access the details of the employees in a company .

PAGE2

**2.ALGORITHM OF THE PROGRAM**

STEP 1: start

**Step 2:** Display a menu with the following options:

1. Add a record
2. List records
3. Modify a record
4. Delete a record
5. Exit the program

**Step 3:** If the user selects "Add a record", prompt the user to enter the employee's name, age, and salary, and write this information to a file.

**Step 4:** If the user selects "List records", read all of the records from the file and display them to the user.

**Step 5:** If the user selects "Modify a record", prompt the user to enter the name of the employee whose record they wish to modify, and then prompt them to enter the new age and salary for that employee. Update the employee's record in the file with the new information.

**Step 6:** If the user selects "Delete a record", prompt the user to enter the name of the employee whose record they wish to delete. Remove the employee's record from the file.

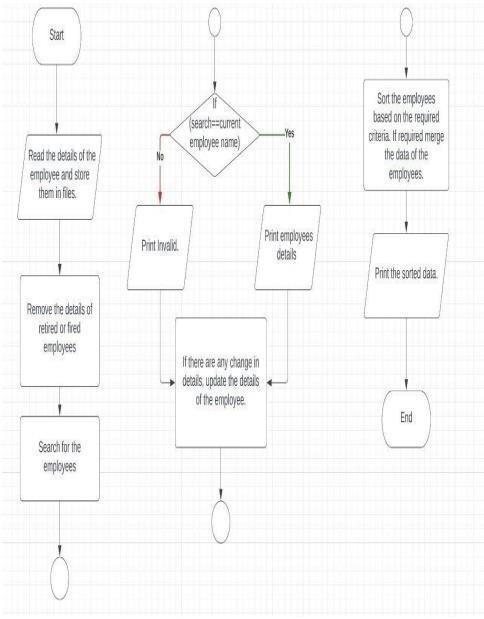
**Step 7:**If the user selects "Exit the program", terminate the program.

**Step 8:** If the user enters an invalid option, display an error message and return to the menu.

**Step 9:** Repeat steps 2-8 until the user chooses to exit the program.

**Step 10:** End

PAGE3



**3.FLOW CHART**

PAGE4

**4. Coding (C/Python)**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct emp {

int id;

char name[50];

float salary;

int age;

} Employee;

// Function prototypes

void addEmployee(Employee\* employees, int\* numEmployees);

void listEmployees(Employee\* employees, int numEmployees);

void modifyEmployee(Employee\* employees, int numEmployees, int id);

void deleteEmployee(Employee\* employees, int\* numEmployees, int id);

// Function to perform user login

int login() {

char username[50];

char password[50];

int attempts = 0;

do {

printf("\nEnter username: ");

scanf("%s", username);

printf("Enter password: ");

scanf("%s", password);

if (strcmp(username, "admin") == 0 && strcmp(password, "password") == 0) {

printf("\nLogin successful!\n");

return 1; // Successful login

} else {

printf("\nInvalid username or password. Please try again.\n");

attempts++;

}

} while (attempts < 3);

printf("\nExceeded maximum login attempts. Exiting program.\n");

exit(1); // Exit program if login attempts are exceeded

} PAGE5

int main() {

// Create an array to store employee records.

Employee employees[100];

// Initialize the number of employees to 0.

int numEmployees = 0;

// Perform login

if (login()) {

// Display a menu to the user.

int choice;

do {

printf("\nEmployee Record System");

printf("\n1. Add employee");

printf("\n2. List employees");

printf("\n3. Modify employee");

printf("\n4. Delete employee");

printf("\n5. Exit");

printf("\n\nEnter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

addEmployee(employees, &numEmployees);

break;

case 2:

listEmployees(employees, numEmployees);

break;

case 3:

int id;

printf("\nEnter employee ID: ");

scanf("%d", &id);

modifyEmployee(employees, numEmployees, id);

break;

case 4:

printf("\nEnter employee ID to be deleted: ");

scanf("%d", &id);

deleteEmployee(employees, &numEmployees, id);

break;

case 5:

exit(0);

break;

default:

printf("\nInvalid choice!");

} PAGE6

} while (1); // Infinite loop until the user chooses to exit

}

return 0;

}

// Function implementations

void addEmployee(Employee\* employees, int\* numEmployees) {

printf("\nEnter employee ID: ");

scanf("%d", &employees[\*numEmployees].id);

printf("\nEnter employee name: ");

scanf("%s", employees[\*numEmployees].name);

printf("\nEnter employee salary: ");

scanf("%f", &employees[\*numEmployees].salary);

printf("\nEnter employee age: ");

scanf("%d", &employees[\*numEmployees].age);

(\*numEmployees)++;

}

void listEmployees(Employee\* employees, int numEmployees) {

printf("\nEmployee Records\n");

printf("ID\tName\tSalary\t\tAge\n");

for (int i = 0; i < numEmployees; i++) {

printf("%d\t%s\t%f\t%d\n", employees[i].id, employees[i].name, employees[i].salary, employees[i].age);

}

}

void modifyEmployee(Employee\* employees, int numEmployees, int id) {

int index = -1;

for (int i = 0; i < numEmployees; i++) {

if (employees[i].id == id) {

index = i;

break;

}

}

if (index == -1) {

printf("\nEmployee record not found!");

return;

} PAGE7

printf("\nEnter employee name: ");

scanf("%s", employees[index].name);

printf("\nEnter employee salary: ");

scanf("%f", &employees[index].salary);

printf("\nEnter employee age: ");

scanf("%d", &employees[index].age);

}

void deleteEmployee(Employee\* employees, int\* numEmployees, int id) {

int index = -1;

for (int i = 0; i < \*numEmployees; i++) {

if (employees[i].id == id) {

index = i;

break;

}

}

if (index == -1) {

printf("\nEmployee record not found!");

return;

}

for (int i = index; i < \*numEmployees - 1; i++) {

employees[i] = employees[i + 1];

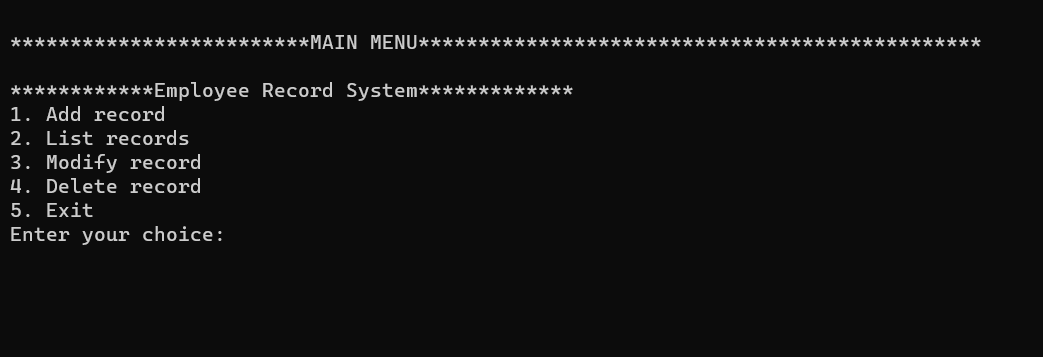
}

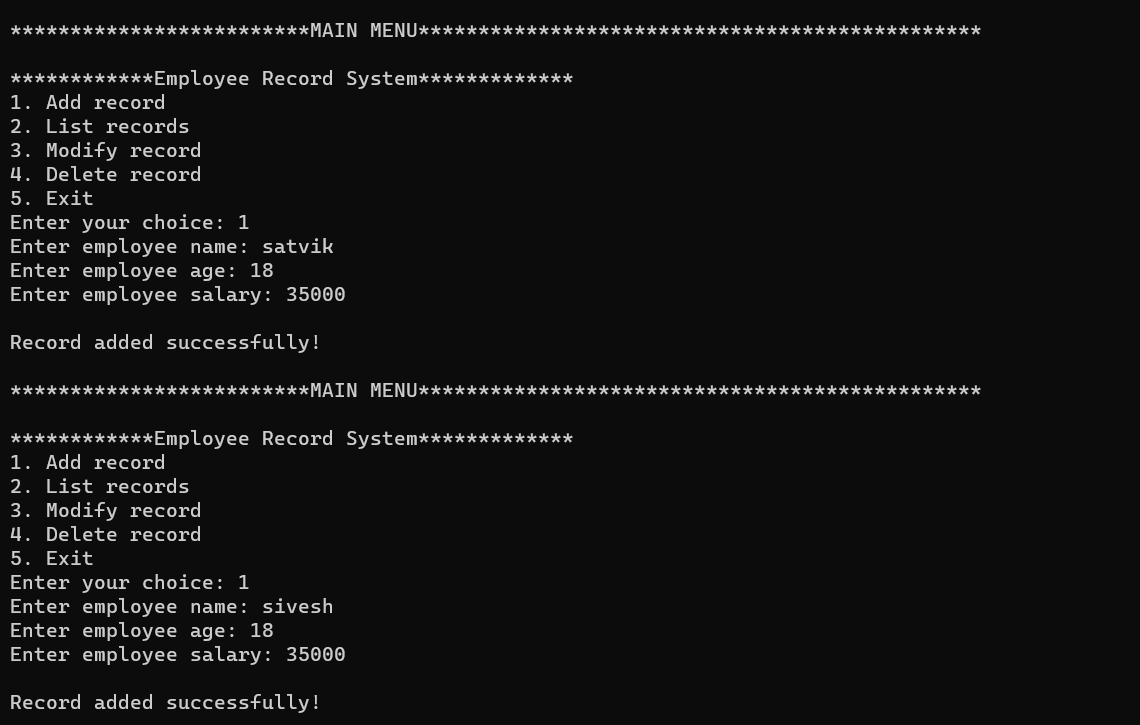
(\*numEmployees)--;

}

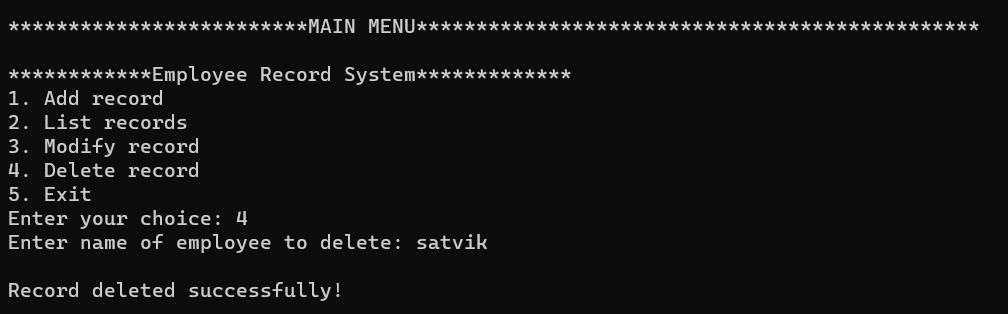
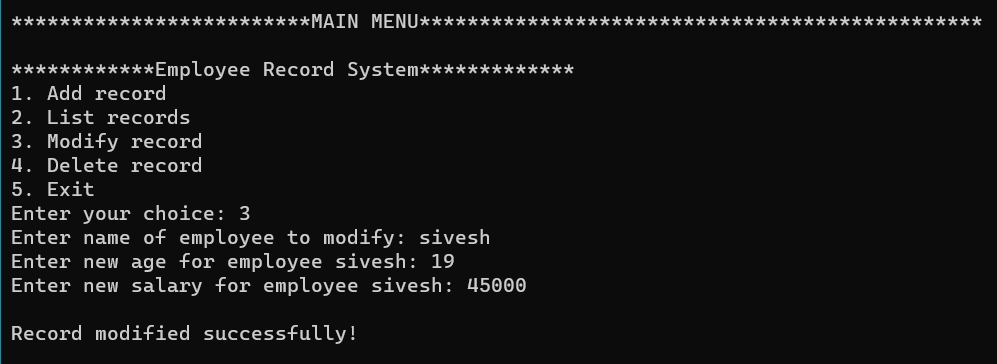
PAGE8

OUTPUT :

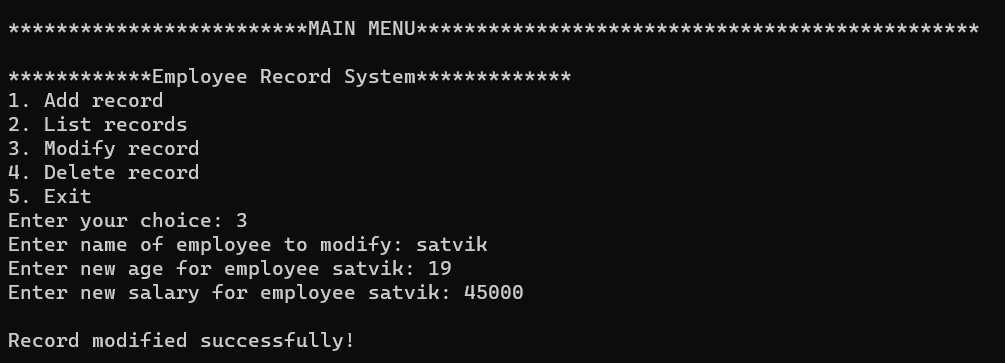
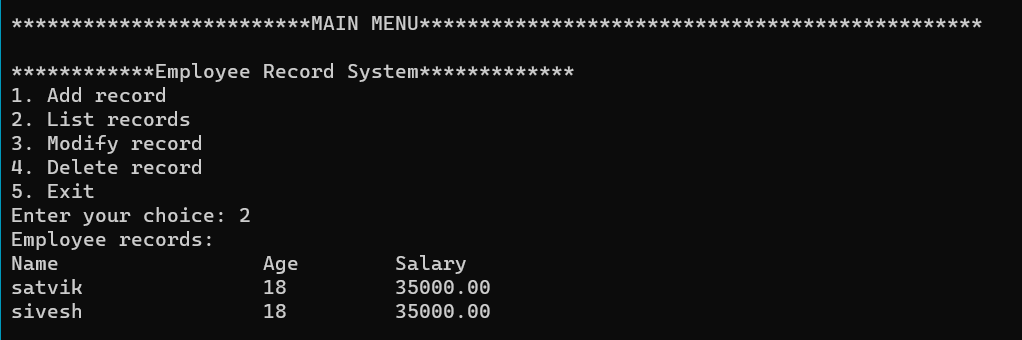




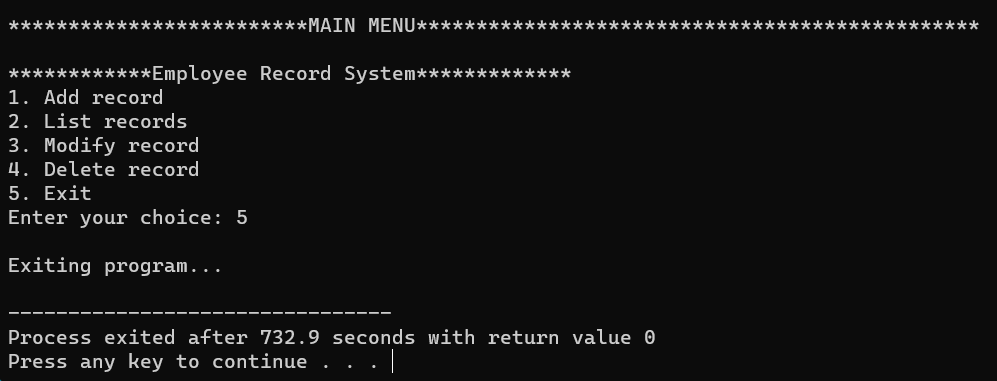
PAGE9



PAGE10



PAGE 11



PAGE12

**MODULES OF THE PROPOSED WORK :**

1. login:

- Purpose: Performs user login.

- Input: None

- Output: Returns 1 for a successful login, exits the program if login attempts exceed the limit.

2. addEmployee:

- Purpose: Adds a new employee to the system.

- Input:

- Employee\* employees: Array of employee records.

- int\* numEmployees: Pointer to the number of existing employees.

- Output: Modifies the array of employees and increments the number of employees.

3. listEmployees:

- Purpose: Lists all existing employees.

- Input: PAGE13

- Employee\* employees: Array of employee records.

- int numEmployees: Number of existing employees.

- Output: Prints the details of each employee.

4. modifyEmployee:

- Purpose: Modifies the details of an existing employee.

- Input:

- Employee\* employees: Array of employee records.

- int numEmployees: Number of existing employees.

- int id: Employee ID to be modified.

- Output: Modifies the details of the specified employee.

5. deleteEmployee: - Purpose: Deletes an existing employee from the system.

- Input:

- Employee\* employees: Array of employee records.

- int\* numEmployees: Pointer to the number of existing employees.

- int id: Employee ID to be deleted. PAGE14

- Output: Modifies the array of employees and decrements the number of employees.

6. main:

- Purpose: The main function where the program starts execution.

- Input: None

- Output: Manages the flow of the program, including the login process and the employee record system menu.

PAGE15

**RESULT :-**

The program is successfully completed

**Conclusion :-**

We construct an employee record system in this C Projects source code series (ERS). We can keep track of the information of a company's or organization's personnel. As soon as you hear the name, you'll have an idea of what this project is about. Afile handling mechanism is used to save the data in a specified file.

We successfully completed developing our project “Employee record system” in c language, with insertion, modify and deletion of data, without any errors.

PAGE16

**Refrences:**

This is a guide to employee record in c .Here we are discussing the definition, syntax,How employee record program work in C? examples with code implementation, you may also have a look at the following articles to learn more-

1.Linear search in C

2.Circular Linked List in C

3.Circular Queue in C

4.Binary Search in C

Student Name :K. SRI VARDHAN REDDY[Reg.No.:RA2311028010066]

B.Tech. CSE – CLOUD COMPUTING

PAGE17