Company maintains employee information such as employee ID, name, designation and salary. Allow users to add, delete information of employees. Display information of a particular employee. If an employee does not exist an appropriate message is displayed. If it is, then the system displays the employee details. Use index sequential file to maintain the data.

CODE:

```
#include <iostream>
#include <fstream>
#include <cstring>

using namespace std;

const char FILE_NAME[] = "employee_data.dat";
const char INDEX_FILE_NAME[] = "employee_index.dat";

struct Employee {
    int id;
    char name[50];
    char designation[50];
    float salary;
};

struct IndexRecord {
    int id;
    int offset;
};
```

```
// Function to display employee details
void displayEmployee(const Employee& emp) {
    cout << "Employee ID: " << emp.id << endl;
    cout << "Name: " << emp.name << endl;
    cout << "Designation: " << emp.designation << endl;
    cout << "Salary: " << emp.salary << endl;
}

// Function to add an employee to the file
void addEmployee() {
    Employee emp;
    cout << "Enter Employee ID: ";
    cin >> emp.id;
    cout << "Enter Name: ";
    cin.ignore();
    cin.getline(emp.name, sizeof(emp.name));
    cout << "Enter Designation: ";
    cin.getline(emp.designation, sizeof(emp.designation));
    cout << "Enter Salary: ";
    cin >> emp.salary;

    ofstream dataFile(FILE_NAME, ios::binary | ios::app);
    ofstream indexFile(INDEX_FILE_NAME, ios::binary | ios::app);

    if (!dataFile || !indexFile) {
        cout << "Error opening file!" << endl;
        return;
    }
```

```
    indexFile.seekp(0, ios::end);
    int offset = dataFile.tellp();

    dataFile.write(reinterpret_cast<char*>(&emp), sizeof(Employee));

    IndexRecord indexRecord;
    indexRecord.id = emp.id;
    indexRecord.offset = offset;
    indexFile.write(reinterpret_cast<char*>(&indexRecord), sizeof(IndexRecord));

    dataFile.close();
    indexFile.close();

    cout << "Employee added successfully!" << endl;
}

// Function to delete an employee from the file
void deleteEmployee() {
    int id;
    cout << "Enter Employee ID to delete: ";
    cin >> id;

    fstream dataFile(FILE_NAME, ios::binary | ios::in | ios::out);
    fstream indexFile(INDEX_FILE_NAME, ios::binary | ios::in | ios::out);

    if (!dataFile || !indexFile) {
        cout << "Error opening file!" << endl;
        return;
```

```
}

IndexRecord indexRecord;
bool found = false;
while (indexFile.read(reinterpret_cast<char*>(&indexRecord), sizeof(IndexRecord))) {
    if (indexRecord.id == id) {
        found = true;
        break;
    }
}

if (found) {
    // Clear the record in the data file
    Employee emp;
    memset(&emp, 0, sizeof(Employee));
    dataFile.seekp(indexRecord.offset);
    dataFile.write(reinterpret_cast<char*>(&emp), sizeof(Employee));

    // Clear the record in the index file
    indexRecord.id = -1;
    indexRecord.offset = -1;
    indexFile.seekp(indexFile.tellg() - sizeof(IndexRecord));
    indexFile.write(reinterpret_cast<char*>(&indexRecord), sizeof(IndexRecord));

    cout << "Employee deleted successfully!" << endl;
} else {
    cout << "Employee not found!" << endl;
}
```

```
    dataFile.close();
    indexFile.close();
}

// Function to display the details of a particular employee
void displayEmployeeDetails() {
    int id;
    cout << "Enter Employee ID: ";
    cin >> id;

    ifstream dataFile(FILE_NAME, ios::binary);
    ifstream indexFile(INDEX_FILE_NAME, ios::binary);

    if (!dataFile || !indexFile) {
        cout << "Error opening file!" << endl;
        return;
    }

    IndexRecord indexRecord;
    bool found = false;
    while (indexFile.read(reinterpret_cast<char*>(&indexRecord), sizeof(IndexRecord))) {
        if (indexRecord.id == id) {
            found = true;
            break;
        }
    }

    if (found) {
        Employee emp;
```

```
        dataFile.seekg(indexRecord.offset);
        dataFile.read(reinterpret_cast<char*>(&emp), sizeof(Employee));
        displayEmployee(emp);
    } else {
        cout << "Employee not found!" << endl;
    }

    dataFile.close();
    indexFile.close();
}

int main() {
    int choice;

    while (true) {
        cout << "1. Add Employee" << endl;
        cout << "2. Delete Employee" << endl;
        cout << "3. Display Employee Details" << endl;
        cout << "4. Exit" << endl;
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                addEmployee();
                break;
            case 2:
                deleteEmployee();
                break;
```

```
            case 3:
                displayEmployeeDetails();
                break;
            case 4:
                cout << "Exiting..." << endl;
                return 0;
            default:
                cout << "Invalid choice! Try again." << endl;
                break;
        }

        cout << endl;
    }

    return 0;
}
```

OUTPUT:


main.cpp
employee_data.dat
employee_index.dat
1
Tanusri
HR
@


main.cpp
employee_data.dat
employee_index.dat
1
yyyyyyyy
1