```
//index sequential file
#include<iostream>
#include<fstream>
#include <sstream>
#include <string>

using namespace std;

class employee
{
     struct empinfo
    {
        int empid;
        char empName[50];
        char empDesignation[50];
        float empSalary;

    } records;

public :
    void create();
    void Delete(int id);
    void display();
    void print();

};
void employee::print()
{
    cout << "------------Details of Employee---------" << endl;
    cout << "EMployee ID :" << records.empid << std::endl;
    cout << "EMployee Name :" << records.empName<< std::endl;
    cout << "EMployee Designation :" << records.empDesignation<< std::endl;
    cout << "EMployee Salary :" << records.empSalary<< std::endl;
}


void employee::create() {
    fstream file;
    fstream indexfile;
    char ch = 'y';

    file.open("employeenew.txt", ios::app | ios::binary);
    indexfile.open("indexnew.txt", ios::app);

    do {
        cout << "Enter employee Id: ";
        cin >> records.empid;

        cin.ignore(); // clear buffer before getline
        cout << "Enter employee Name: ";
        cin.getline(records.empName, 50);

        cout << "Enter employee Designation: ";
        cin.getline(records.empDesignation, 50);

        cout << "Enter employee Salary: ";
        cin >> records.empSalary;
        file.write((char*)&records, sizeof(records));

        int position = file.tellp();
        indexfile << records.empid << " " << position << endl;

        cout << "Employee added successfully.\n";

        cout << "\nDo you want to add more records? (y/n): ";
        cin >> ch;

    } while (ch == 'y' || ch == 'Y');

    file.close();
    indexfile.close();
}

static int findEmployeePosition(int employeeID) {
    ifstream indexFile("indexnew.txt");
    if (!indexFile) {
        cout << "Error opening index file." << endl;
        return -1;
    }

    string line;
    while (getline(indexFile, line)) {
        istringstream iss(line);
        int id, position;
        if (iss >> id >> position) {
            if (id == employeeID) {
                return position;
            }
        }
    }

    return -1;
}


void employee::display() {
    int empId;
    cout << "Enter employee ID to display: ";
    cin >> empId;

    int pos = findEmployeePosition(empId);
    if (pos < 0) {
        cout << "No matching employee record found." << endl;
        return;
    }

    fstream file("employeenew.txt", ios::in | ios::binary);
    if (!file) {
        cout << "Error opening employee data file." << endl;
        return;
    }

    file.seekg(pos);
    file.read((char*)&records, sizeof(records));

    print();

    file.close();
}

void employee::Delete(int employeeId)
{
    // Open the index file
    fstream indexFile("indexnew.txt",ios::in);
    if (!indexFile) {
        cout << "Error opening index file." << endl;
        return;
    }

    // Open the data file
    fstream file("employeenew.txt", ios::in);
    if (!file) {
        cout << "Error opening data file." << endl;
        indexFile.close();
        return;
    }

    // Create a temporary file to store updated index entries
    ofstream tempIndexFile("tempIndex.txt",ios::out);
    if (!tempIndexFile) {
        cout << "Error creating temporary index file." << endl;
        indexFile.close();
        file.close();
        return;
    }

    string line;
    int id;
    int position;
    bool found = false;
    // Read each line from the index file
    while (getline(indexFile, line)) {
        istringstream iss(line);
        if (iss >> id >> position)
         {
            if (id == employeeId)
             {
                found = true;
                // Skip the record by not writing it to the temporary index file
                continue;
            }
        }

        // Write the index entry to the temporary index file
        tempIndexFile << line << endl;
    }

    // Close the files
    indexFile.close();
    file.close();
    tempIndexFile.close();
    if (!found)
    {
        remove("tempIndexFile.txt");
        cout << "Employee record not found" << endl;
    }
    else
    {
        remove("indexnew.txt");
        rename("tempIndex.txt", "indexnew.txt");

        cout << "Employee deleted successfully." << endl;

    }
    // Remove the original index file and rename the temporary index file

}
int main()
{
    employee emp;
    int employeeId;
    int choice;
    char ans = 'y';
    do
    {
        cout << "1. Add Employee" << endl;
        cout << "2. Delete Employee" << endl;
        cout << "3. Display Employee" << endl;
        cout << "4. Exit" << endl;

        cout << " \n Enter your choice" << endl;
        cin >> choice;

        switch (choice)
        {
        case 1: emp.create();
            break;

        case 2:cout << "Enter employee Id to delete : " << endl;
            cin >> employeeId;
            emp.Delete(employeeId);
            break;

        case 3:
          emp.display();
            break;
        case 4:
            return 0;
        default:
            cout << "Enter  valid choice" << endl;
            break;
        }
    } while (true);

    return 0;

}
```