Data structures Project

Employee record management system using linked list

Problem: Create an employee Record Management system using linked list that can perform the following operations:

- Insert employee record
- Delete employee record
- Update employee record
- Show details of an employee
- Search an employee
- Update the salary of an employee

The employee record should contain the following items

- Name of Employee
- ID of Employee
- First day of work
- Phone number of the employee
- Address of the employee
- Work hours
- Salary

Approach:

With the basic knowledge of operations on Linked Lists like insertion, deletion of elements in the Linked list, the employee record management system can be created. Below are the functionalities explained that are to be implemented:

Check Record: It is a utility function of creating a record it checks before insertion that the Record Already Exist or not. It uses the concept of checking for a Node with given Data in a linked list.

Create Record: It is as simple as creating a new node in the Empty Linked list or inserting a new node in a non-Empty linked list.

Smart Search Record: Search a Record is similar to searching for a key in the linked list. Here in the employee record key is the ID number as a unique for every employee.

Delete Record: Delete Record is similar to deleting a key from a linked list. Here the key is the ID number. Delete record is an integer returning function it returns -1 if no such record with a given roll number is found otherwise it deletes the node with the given key and returns 0.

Show Record: It shows the record is similar to printing all the elements of the Linked list.

Update salary: It add 2% of the salary for every extra hour. By default, 32 hours are required for every employee.

Recommendations:

Although the implementation of exception handling is quite simple few things must be taken into consideration before designing such a system:

1. ID must be used as a **key** to distinguish between two different records, so while inserting record check whether this record already exists in our database or not, if it already exists then immediately report to the user an error message.
2. The record should be inserted in sorted order, you can use the inserting node in the sorted linked list seen in the lectures.

Deadline: 04 Feb 2023

```
package employeerecordmanagementsystem;


/**

 *

 * @author Xxfir

 */

import java.util.Scanner;


class Employee {

        int id;

        String name;

        String startDate;

        String phone;

        String address;

        int hours;

        double salary;

    Employee next;
```

```
        public Employee(int id, String name, String startDate, String phone, String address, int
hours, double salary) {
                this.id = id;
                this.name = name;
                this.startDate = startDate;
                this.phone = phone;
                this.address = address;
                this.hours = hours;
                this.salary = salary;
        }

        public void display() {
                System.out.println("ID: " + id);
                System.out.println("Name: " + name);
                System.out.println("Start Date: " + startDate);
                System.out.println("Phone: " + phone);
                System.out.println("Address: " + address);
                System.out.println("Hours: " + hours);
                System.out.println("Salary: " + salary);
        }
}


class LinkedList {
        Employee head;

    public boolean checkRecord(int id) {
      Employee temp = head;
      while (temp != null) {
        if (temp.id == id) {
          return true;
        }
```

```
            temp = temp.next;
        }
        return false;
    }


            public void insert(int id, String name, String startDate, String phone, String address, int
hours, double salary) {


            if (!checkRecord(id)) {
            Employee newEmployee = new Employee(id, name, startDate, phone, address, hours,
salary);
            if (head == null) {
                head = newEmployee;
            } else {
                Employee temp = head;
                while (temp.next != null) {
                    temp = temp.next;
                }
                temp.next = newEmployee;
            }
            System.out.println("Record created successfully.");
        } else {
            System.out.println("Record with ID " + id + " already exists.");
        }
            }


            public void delete(int id) {
                        Employee current = head;
                        Employee prev = null;


                        while (current != null && current.id != id) {
                                    prev = current;
            current = current.next;
        }

        if (current == null) {
            System.out.println("Employee not found.");
            return;
        }

        if (prev == null) {
            head = current.next;
        } else {
            prev.next = current.next;
        }
    }

    public void updateSalary(int id, int newHours) {
        Employee current = head;

        while (current != null && current.id != id) {
            current = current.next;
        }

        if (current == null) {
            System.out.println("Employee not found.");
            return;
        }

        current.hours = newHours;
        current.salary += (newHours - 32) * (current.salary * 2 / 100);
    }
```

```
        public void display() {
                Employee current = head;

                while (current != null) {
                        current.display();
                        current = current.next;
                }
        }

        public Employee search(int id) {
                Employee current = head;

                while (current != null && current.id != id) {
                        current = current.next;
                }

                return current;
        }
}


public class EmployeeRecordManagementSystem {

        public static void main(String[] args) {
         try (Scanner sc = new Scanner(System.in)) {
            LinkedList employees = new LinkedList();

            OUTER:
            while (true) {
               System.out.println("Enter your choice: ");
               System.out.println("1. Insert Employee Record");
```

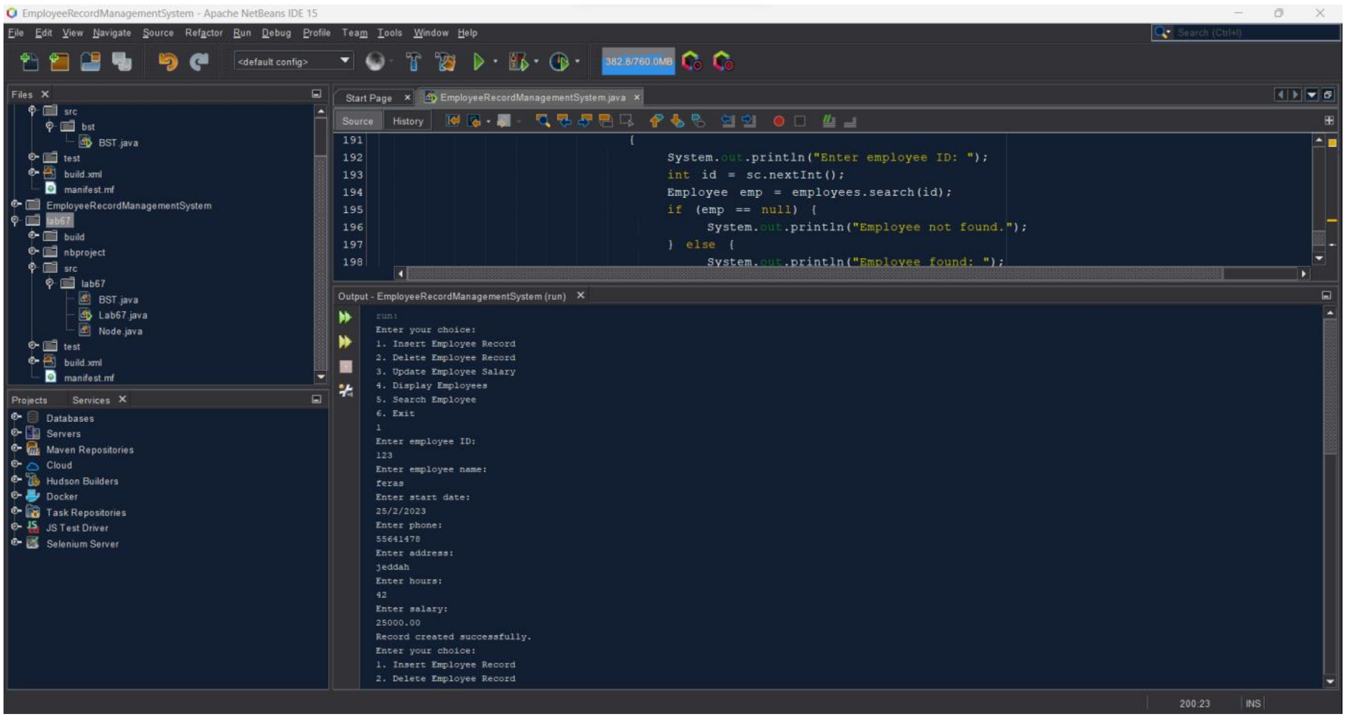
```
 System.out.println("2. Delete Employee Record");
 System.out.println("3. Update Employee Salary");
 System.out.println("4. Display Employees");
 System.out.println("5. Search Employee");
 System.out.println("6. Exit");

int choice = sc.nextInt();

 switch (choice) {
    case 1:
    {
       System.out.println("Enter employee ID: ");
       int id = sc.nextInt();
       System.out.println("Enter employee name: ");
       sc.nextLine();
       String name = sc.nextLine();
       System.out.println("Enter start date: ");
       String startDate = sc.nextLine();
       System.out.println("Enter phone: ");
       String phone = sc.nextLine();
       System.out.println("Enter address: ");
       String address = sc.nextLine();
       System.out.println("Enter hours: ");
       int hours = sc.nextInt();
       System.out.println("Enter salary: ");
       double salary = sc.nextDouble();
       employees.insert(id, name, startDate, phone, address, hours, salary);
       break;
    }
    case 2:
    {
```

```
    System.out.println("Enter employee ID: ");
    int id = sc.nextInt();
    employees.delete(id);
    break;
}
case 3:
{
    System.out.println("Enter employee ID: ");
    int id = sc.nextInt();
    System.out.println("Enter new hours: ");
    int newHours = sc.nextInt();
    employees.updateSalary(id, newHours);
    break;
}
case 4:
    employees.display();
    break;
case 5:
{
    System.out.println("Enter employee ID: ");
    int id = sc.nextInt();
    Employee emp = employees.search(id);
    if (emp == null) {
        System.out.println("Employee not found.");
    } else {
        System.out.println("Employee found: ");
        emp.display();
    }
    break;
}
case 6:
```

```
                        break OUTER;
                    default:
                        break;
                }
            }
        }
    }
}
```



```
{
    System.out.println("Enter employee ID: ");
    int id = sc.nextInt();
    Employee emp = employees.search(id);
    if (emp == null) {
        System.out.println("Employee not found.");
    } else {
        System.out.println("Employee found: ");
```

```
run:
Enter your choice:
1. Insert Employee Record
2. Delete Employee Record
3. Update Employee Salary
4. Display Employees
5. Search Employee
6. Exit
1
Enter employee ID:
123
Enter employee name:
feras
Enter start date:
25/2/2023
Enter phone:
55641478
Enter address:
jeddah
Enter hours:
42
Enter salary:
25000.00
Record created successfully.
Enter your choice:
1. Insert Employee Record
2. Delete Employee Record
```