Part B Micro-Project Report
Employee Record Management System

1. Rationale:

The program starts with the declaration of an Employee class that has instance variables such as empid, age, name, designation, salary, and contact_no. It also has several methods to get employee details from the user, search an employee by their ID, update an employee by their ID and delete an employee by their id.

2. Course Outcomes Achieved:

a) Develop programs using Object Oriented Methodology in Java.
b) Implement Exception Handling.

3. Literature Review:

3.1 Employee Record Management System Using Java and MySQL by Deepak Saini and Nitesh Kumar (2020): This paper presents the design and implementation of an ERMS in Java that uses JavaFX for the user interface and MySQL for the database management system. The system includes features such as employee registration, attendance tracking, leave management, and salary processing. The system was tested, and the results showed that the system is efficient and user-friendly.

3.2 Java-Based Human Resource Management System by R.N. Saranya, A. Santhoshkumar, and S. Saravanan (2019): This paper presents the design and implementation of a Java-based HRMS that includes modules for employee information management, attendance tracking, leave management, and payroll processing. The authors used Java Server Faces (JSF) and MySQL for the web-based interface and database management system, respectively. The system's performance was evaluated, and the results showed that the system is efficient and user-friendly.

3.3 Employee Management System in Java by Amit Kumar (2018): This paper proposes an ERMS in Java that uses Java Server Pages (JSP) and MySQL for the web-based interface and database management system, respectively. The system includes features such as employee registration, attendance tracking, and leave management. The system's performance was evaluated, and the results showed that the system is scalable and efficient.

3.4 Java Employee Record Management System (JERMS) by Anuja P. and Chavan S. (2016): This paper presents the design and implementation of JERMS, a Java-based ERMS. The system includes modules for managing employee details, salary information, attendance records, and performance evaluation. The authors used Java Swing for the user interface and MySQL for the database management system. The system was tested and evaluated, and the results showed that JERMS is a user-friendly and efficient ERMS.

4. Actual Methodology Followed:

- The getdata() method uses a Scanner object to get employee details from the user. It prompts the user to enter employee ID, name, salary, designation, age, and contact number. It also validates the user inputs for age and contact number. If the user enters an age less than zero or a contact number that is not ten digits long, it prompts the user to enter a valid value.

- The search() method takes an array of employees, the number of employees, and an employee ID to search for an employee in the array. If it finds the employee with the given ID, it displays the details of the employee in a formatted table. If the employee is not found with the given ID, it displays a message.

- The update() method takes an array of employees and the number of employees to update an employee record by their ID. It prompts the user to enter the employee ID to update and the field they want to update (salary, age, contact number, or designation). It then finds the employee in the array and updates the corresponding field as per user input. If the user enters an invalid contact number, it prompts the user to enter a valid number.

- The delete() method takes an array of Employee objects and the total number of employees in the array as parameters. It prompts the user to input the ID of the employee they wish to delete. The method then searches for the employee with the given ID in the array by comparing it with the empid field of each Employee object in the array. If the employee is found, the method stores the index of the employee in a variable called index. Next, the method shifts all the remaining elements in the array to the left starting from the index of the deleted employee. This effectively removes the deleted employee from the array. The method also decrements the total number of employees by one and returns the updated count to the caller. If the employee with the given ID is not found in the array, the method returns the original count of employees without making any changes to the array.

Source Code:

```
import java.util.*; // For creating object of Scanner Class

class Employee
{
    // instance variables
    int empid,age;
    String name, designation;
    float salary;
    long contact_no = 0;

    void getdata()   // method to get employee details from user
    {
        Scanner s = new Scanner(System.in);

        System.out.println("-----Enter Details of Employee-----");

        // input employee ID
        System.out.print("\nEnter Employee ID: ");
        empid = s.nextInt();

        // input employee name
        System.out.print("\nEnter Name: ");
        s.nextLine();  // consume the newline character from the previous input
        name = s.nextLine();

        // input employee salary
        System.out.print("\nEnter Salary: ");
        salary = s.nextFloat();

        // input employee designation
        System.out.print("\nEnter Designation: ");
        designation = s.nextLine();

        // input employee age with validation
        System.out.print("\nEnter Age: ");
        age = s.nextInt();
        if(age<0) // check if age is less than zero, if yes then prompt user to enter a valid
age
        {

                System.out.print("Enter a valid age:");
                age=s.nextInt();

        }

        // input employee contact number
        System.out.print("\nEnter Contact Number: ");
        contact_no = s.nextLong();
        // check if contact number is 10 digit long, if not prompt user to enter a valid
number
        while(String.valueOf(contact_no).length() != 10)
          {
            System.out.print("\nContact number should be of 10 digits \nEnter a valid
Number:");
            contact_no = s.nextLong();
          }

        System.out.println("************************************");
    }

    // method to search an employee by their id
    void search(Employee e[], int n, int ei)
     {
        int flag = 0;

        for (int i = 0; i < n; i++)
        {
            if (ei == e[i].empid)
            {
                flag = 1;

                String format = "| %-10s | %-20s | %-15s | %-20s | %-5s | %-15s |\n";
                // display the details of the employee in a formatted table

System.out.println("____________________________________________________
____________________________________________________");
                System.out.format(format, "Empid", "Name", "Salary", "Designation",
"Age", "Contact_no");

System.out.println("____________________________________________________
____________________________________________________");
                System.out.format(format, e[i].empid, e[i].name, e[i].salary, e[i].designation,
e[i].age, e[i].contact_no);

System.out.println("____________________________________________________
____________________________________________________");

                break;
            }
        }

        if (flag == 0)
        {
           // if employee not found with the given ID, display a message
           System.out.println("Employee Not Found");
        }
    }

     // method to update an employee by their id
    void update(Employee e[], int n)
    {
        int eID, flag = 0, a;
        float sal;
        String desg;
        long con;

        Scanner s = new Scanner(System.in);
        // input employee id to update
        System.out.println("Enter Employee's ID to Update:");
        eID = s.nextInt();

        System.out.println("\n<<What you want to update>>");
        System.out.print("\n1.EMPLOYEE'S     SALARY     \n2.EMPLOYEE'S     AGE
\n3.EMPLOYEE'S CONTACT NO \n4.EMPLOYEE'S DESIGNATION");
        System.out.print("\nEnter Your Choice:");
        int N = s.nextInt();

        // loop through the array of employees to find the employee with the given ID
        for(int i=0;i<n;i++)
        {
           if(eID==e[i].empid)
           {
           for (  i = 0; i < n; i++)
               {
                   if (eID == e[i].empid) // if employee id matched
                     {
                        switch (N)
                        {
                           case 1: // update employee salary
                              System.out.print("\nEnter salary to update:");
                              sal = s.nextFloat();
                           for (int j = 0; j < n; j++)
                            {
                                 if (eID == e[j].empid)
                                  {
                                     e[j].salary = sal;
```

```
                flag = 1;
            }
      }
     break;

     case 2:  // update employee age
     System.out.print("\nEnter AGE to Update:");
     a = s.nextInt();
     for (int j = 0; j < n; j++)
     {
         if (eID == e[j].empid)
         {
             e[j].age = a;
             flag = 1;
         }
     }
     break;

     case 3: // update employee contact number
     System.out.print("\nEnter Contact you want to update:");
     con = s.nextLong();
    // check if contact number is 10 digit long, if ot prompt user to enter
a valid number
     if (String.valueOf(con).length() != 10)
     {
         System.out.println("Contact number should be of 10 digits");
     }
     else
     {
         for (int j = 0; j < n; j++)
         {
             if (eID == e[j].empid)
             {
                 e[j].contact_no = con;
                 flag = 1;
             }
         }
     }
     break;

 case 4:  // update employee designation
     System.out.print("\nEnter Designation to Update:");
     desg = s.next();
         for (int j = 0; j < n; j++)
```

```
                                {
                                    if (eID == e[j].empid)
                                    {
                                        e[j].designation = desg;
                                        flag = 1;
                                    }
                                }
                            break;

                            default:
                              System.out.println("Invalid choice!");
                           break;
                      }
                   }
                  }
                 }
                 else
                 {
                   // if employee not found with the given id, display message
                   System.out.println("\nEmployee Not Found");
                   break;
                 }
             }
         if (flag == 1)
         {
             // if employee record updated, display message
             System.out.println("\nInformation Updated");
         }
         else
         {
             // if employee record not updated, display message
             System.out.println("\nInformation Not Updated");
         }
     }

    // method to display employee details in table format
     void display()
     {
         String format = "| %-10s | %-20s | %-15s | %-20s | %-5s | %-15s |\n";

         System.out.format(format, empid, name, salary, designation, age, contact_no);

System.out.println("________________________________________________________
________________________________________________________");
```

```
    }

     //method to delete employee details with entered employee id
    int delete(Employee e[], int n)
    {
        Scanner s = new Scanner(System.in);

       //  Prompt the user to enter the ID of the employee to be deleted
        System.out.println("Enter Employee ID to Delete:");
        int empID = s.nextInt();

        //Initialize a variable to store the index of the employee to be deleted
        int index = -1;

         //Loop through the array of employees to find the index of the employee to be
deleted
        for (int i = 0; i < n; i++)
        {
            if (empID == e[i].empid)
            {
                 // Store the index of the employee to be deleted
                index = i;
                break;
            }
        }

        // If Employee id found to be delete
        if (index != -1)
        {
            // If the employee was found, shift all elements after the deleted employee by
one position to the left
            for (int i = index; i < n - 1; i++)
            {
                e[i].empid = e[i+1].empid;
                e[i].name= e[i+1].name;
                e[i].designation= e[i+1].designation;
                e[i].age= e[i+1].age;
                e[i].contact_no= e[i+1].contact_no;
                e[i].salary= e[i+1].salary;
            }

            // Decrease the number of employees by one and inform the user that the
employee was successfully deleted
            System.out.println("Employee with ID " + empID + " has been deleted.");
            return --n;
        }
        else
        {
            // If the employee was not found, inform the user and return the current number
of employees
            System.out.println("Employee Not Found");
            return n;
        }
    }

    // This is the main method of the program
    public static void main(String args[])
    {
        // Loop to create and initialize the array of Employee objects
       while(true)
      {
        // Create a scanner object to read input from the user
        Scanner s = new Scanner(System.in);

        // Initialize variables
        int n = 0; // Number of employees
        int ch; // Menu choice

        // Prompt the user to enter the number of employees to create
        System.out.print("\nEnter Number of Records to Create:");
        n=s.nextInt();

        // Create the array of Employee objects
        Employee e[] = new Employee[n];

        // Initialize each Employee object in the array
        for(int i=0;i<n;i++)
        {
            e[i] = new Employee();
        }

        // Display the menu and prompt the user to choose an option
        while(true)
        {


            System.out.println("\n\n*****<<<<<MENU>>>>>*****");
```

```
                  System.out.println("1 : ACCEPT THE EMPLOYEE
INFORMATIOON");
                  System.out.println("2 : DISPLAY THE EMPLOYEE
INFORMATION");
                  System.out.println("3 : SEARCH THE EMPLOYEE
INFORMATION");
                  System.out.println("4 : UPDATE THE EMPLOYEE
INFORMATION");
      System.out.println("5 : DELETE");
                  System.out.println("6 : EXIT");

      System.out.print("Enter your choice:");
      ch = s.nextInt(); // Read the user's choice

      switch(ch) // Execute the appropriate option based on the user's choice
      {
        case 1: // Accept the employee information from the user
            for(int i=0;i<n;i++)
              e[i].getdata();
            break;

        case 2: // Display the employee information
        String format = "| %-10s | %-20s | %-15s | %-20s | %-5s | %-15s |\n";

System.out.println("______________________________________________________
______________________________________________________");
        System.out.format(format, "Empid", "Name", "Salary", "Designation",
"Age", "Contact_no");

System.out.println("______________________________________________________
______________________________________________________");
          for(int i=0;i<n;i++)
            e[i].display();
           break;

        case 3:  // Search for an employee by ID
            System.out.println("Enter Employee id to search:");
            int ei = s.nextInt();
            for(int i=0;i<n;i++)
            {
              e[i].search(e, n, ei);
              break;
            }
            break;
```

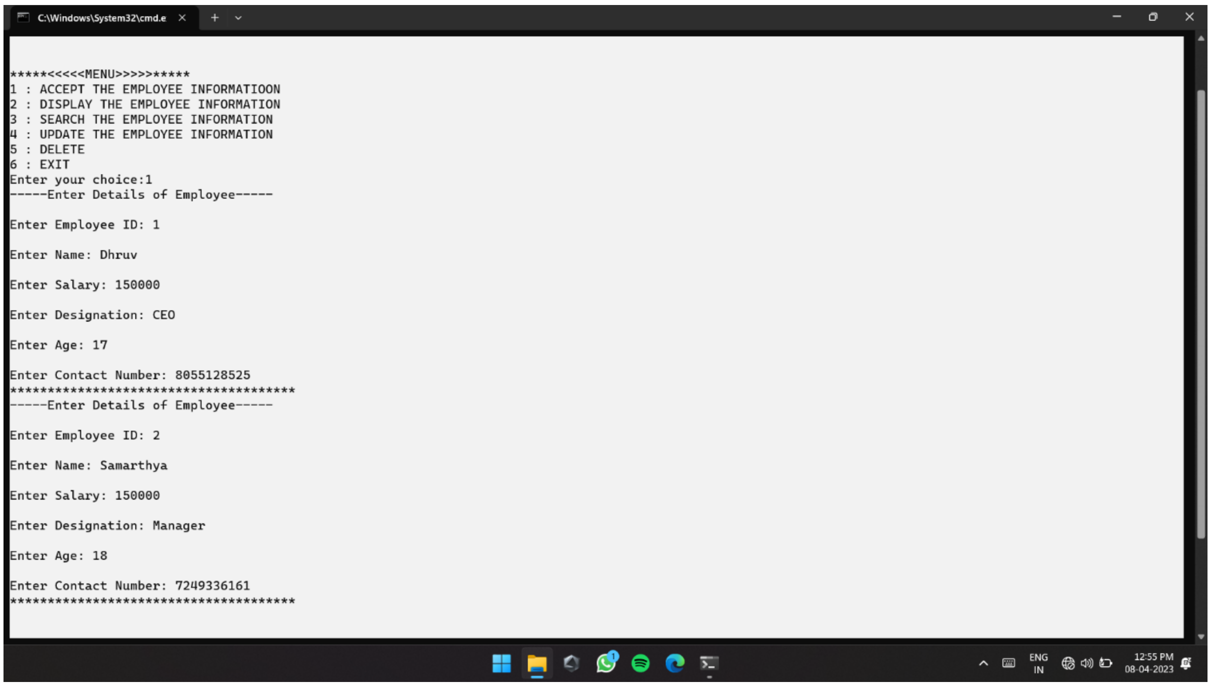
```
                case 4: // Update an employee's information
                    System.out.println("<<<<UPDATE THE DATA>>>>");
                    e[0].update(e,n);
                    break;

                case 5: // Delete an employee's information
                    n= e[0].delete(e, n);
                    break;

                case 6:System.exit(0); // Exit the program
              }
           }
         }
       }
    }
```

5. Actual Resources Required:

Sr. No.	Name of Resource/Material	Specification	Quantity	Remark
1	Computer System	**OS:** Windows 11 (64 bit) **Processor:** Intel i7 11th generation **RAM:** 16GB	1	-
2	Software	jdk 11.0	1	-

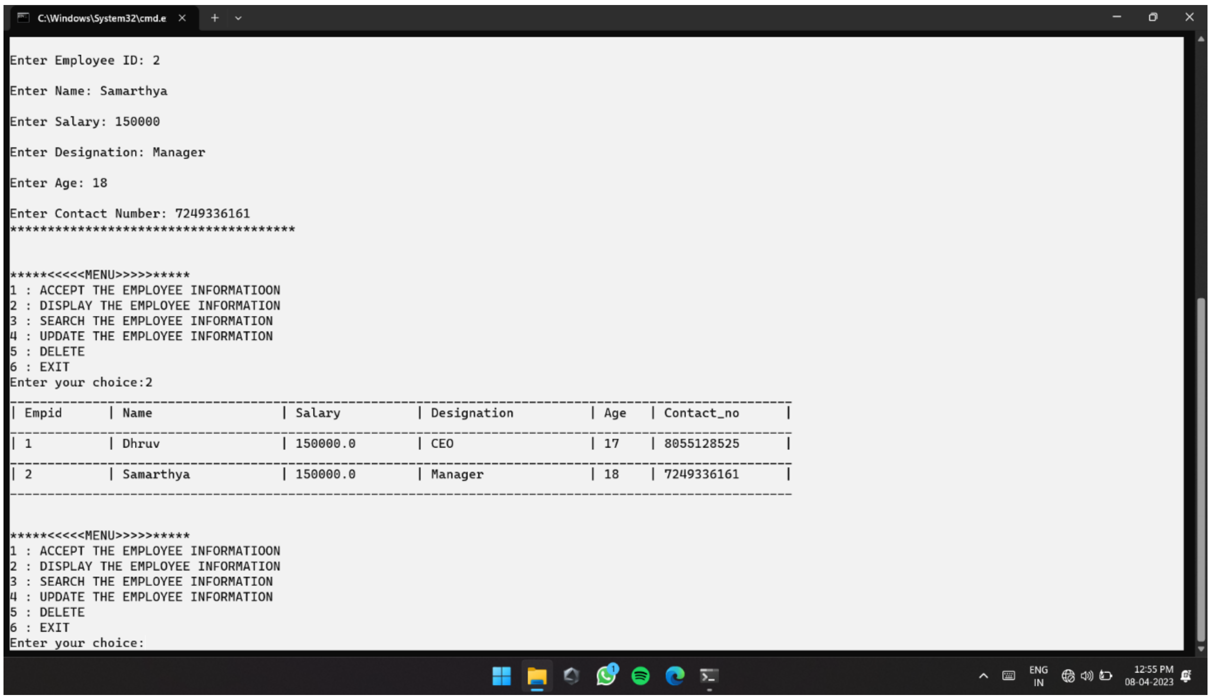
6. Output of Micro-Project:

Case 1: Accepting employee details



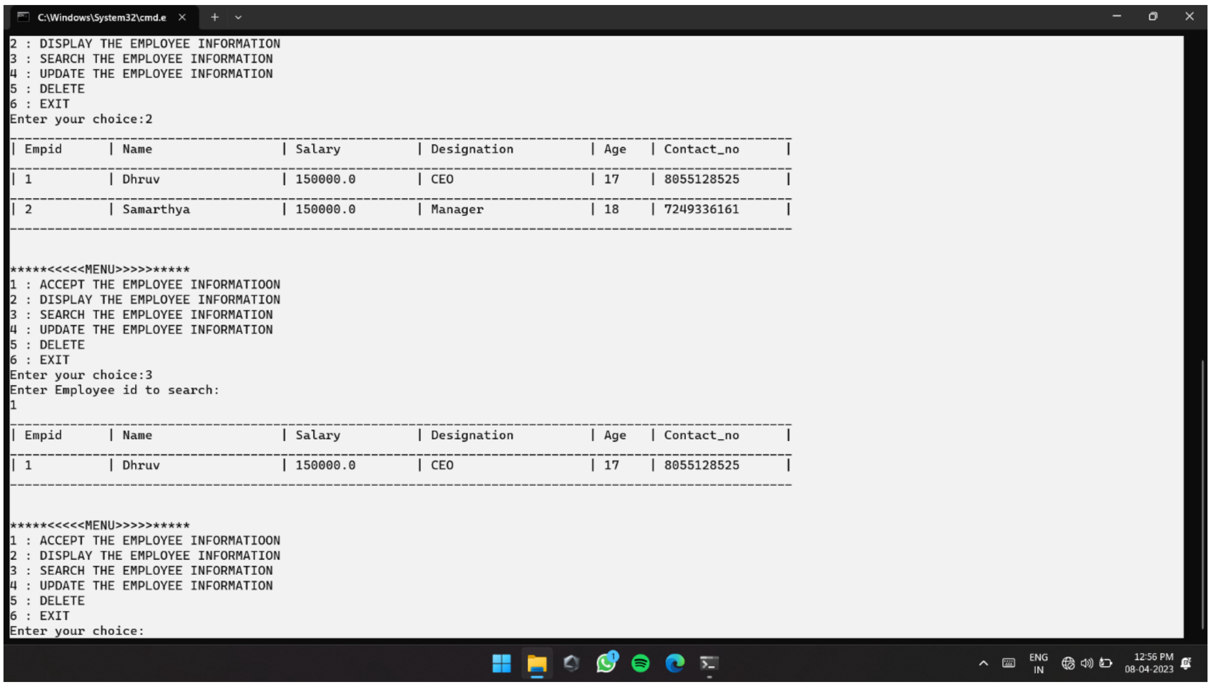
```
*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:1
-----Enter Details of Employee-----

Enter Employee ID: 1

Enter Name: Dhruv

Enter Salary: 150000

Enter Designation: CEO

Enter Age: 17

Enter Contact Number: 8055128525
**************************************
-----Enter Details of Employee-----

Enter Employee ID: 2

Enter Name: Samarthya

Enter Salary: 150000

Enter Designation: Manager

Enter Age: 18

Enter Contact Number: 7249336161
**************************************
```

Case 2: Displaying the information.



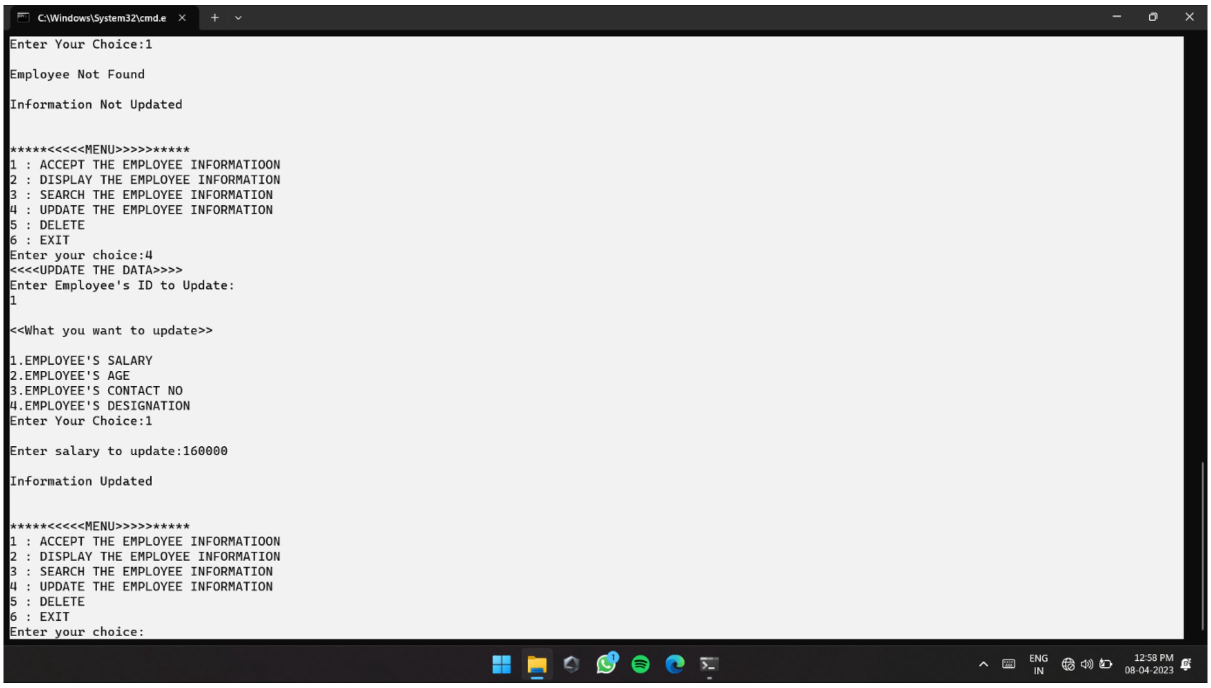
```
Enter Employee ID: 2

Enter Name: Samarthya

Enter Salary: 150000

Enter Designation: Manager

Enter Age: 18

Enter Contact Number: 7249336161
**************************************


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:2
---------------------------------------------------------------------------------------------------------
| Empid      | Name                 | Salary          | Designation          | Age   | Contact_no      |
---------------------------------------------------------------------------------------------------------
| 1          | Dhruv                | 150000.0        | CEO                  | 17    | 8055128525      |
---------------------------------------------------------------------------------------------------------
| 2          | Samarthya            | 150000.0        | Manager              | 18    | 7249336161      |
---------------------------------------------------------------------------------------------------------


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:
```

Case 3: Searching the information



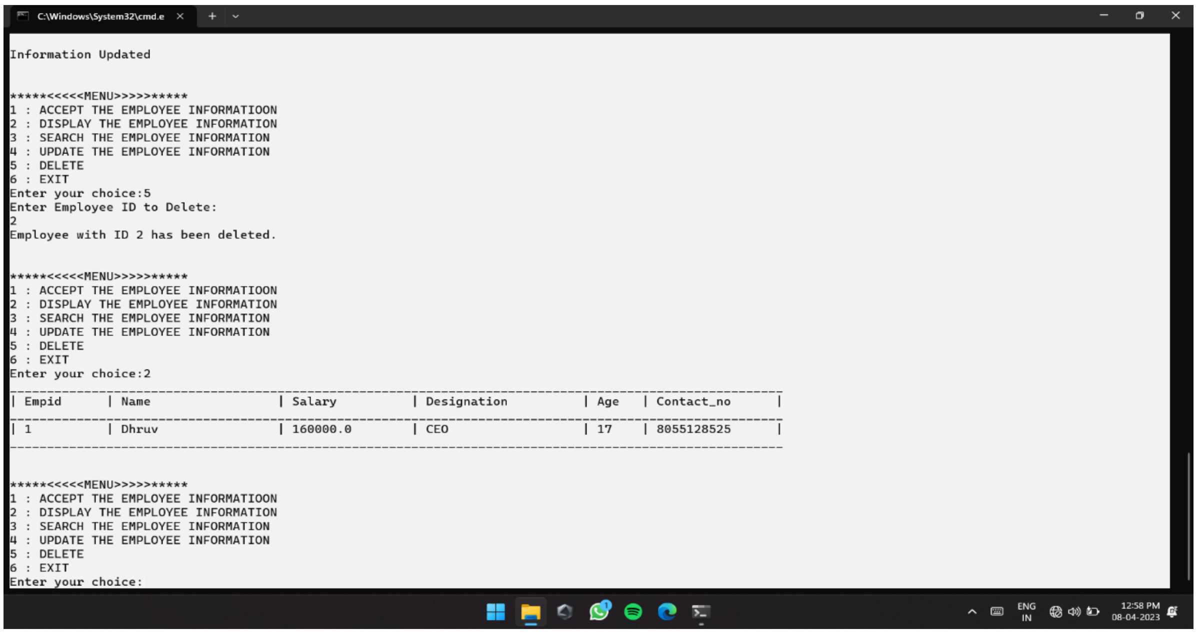
```
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:2
---------------------------------------------------------------------------------------------------------
| Empid       | Name                    | Salary          | Designation          | Age    | Contact_no      |
---------------------------------------------------------------------------------------------------------
| 1           | Dhruv                   | 150000.0        | CEO                  | 17     | 8055128525      |
---------------------------------------------------------------------------------------------------------
| 2           | Samarthya               | 150000.0        | Manager              | 18     | 7249336161      |
---------------------------------------------------------------------------------------------------------


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:3
Enter Employee id to search:
1
---------------------------------------------------------------------------------------------------------
| Empid       | Name                    | Salary          | Designation          | Age    | Contact_no      |
---------------------------------------------------------------------------------------------------------
| 1           | Dhruv                   | 150000.0        | CEO                  | 17     | 8055128525      |
---------------------------------------------------------------------------------------------------------


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:
```

Case 4: Update the information.



```
Enter Your Choice:1

Employee Not Found

Information Not Updated


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:4
<<<<UPDATE THE DATA>>>>
Enter Employee's ID to Update:
1

<<What you want to update>>

1.EMPLOYEE'S SALARY
2.EMPLOYEE'S AGE
3.EMPLOYEE'S CONTACT NO
4.EMPLOYEE'S DESIGNATION
Enter Your Choice:1

Enter salary to update:160000

Information Updated


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:
```

Case 5: Delete the information.



```
Information Updated


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:5
Enter Employee ID to Delete:
2
Employee with ID 2 has been deleted.


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:2
---------------------------------------------------------------------------------------------------------
| Empid       | Name                    | Salary           | Designation             | Age    | Contact_no      |
---------------------------------------------------------------------------------------------------------
| 1           | Dhruv                   | 160000.0         | CEO                     | 17     | 8055128525      |
---------------------------------------------------------------------------------------------------------


*****<<<<<MENU>>>>>*****
1 : ACCEPT THE EMPLOYEE INFORMATIOON
2 : DISPLAY THE EMPLOYEE INFORMATION
3 : SEARCH THE EMPLOYEE INFORMATION
4 : UPDATE THE EMPLOYEE INFORMATION
5 : DELETE
6 : EXIT
Enter your choice:
```

7. Skill Developed:

1) We learned how to define a class and how to create object of class.
2) We learned how to implement the array of object.
3) We learned the concept of menu driven program.
4) We learned how to import package in the program.

8. Applications of this Micro-Project:

1) The employee record management system is use in the offices for managing the information of employee.

9. Area of Future Improvement:

Integration with other systems: For various functions including HR administration, payroll, and time and attendance monitoring, many businesses employee a range of different software systems. These technologies can be integrated with employee management systems to increase productivity and decrease error rates.

Data analysis and reporting: Payroll data, attendance records, and performance indicators are just a few of the types of data that employee management systems can produce. Companies can learn a lot about employee performance, trends, and areas for improvement by integrating powerful data analysis and reporting features.

Security: Sensitive data, including personal and payroll information, is contained in employee management systems. Security needs to be significantly improved as a result. Systems for data backup and recovery, access control, and strong encryption can all aid in preventing data breaches and safeguarding sensitive data.