Experiment-2.4

Student Name: Sanchit Singal
UID: 21BCS1569
Branch: BE-CSE
Section/Group: 606_B
Semester: 6TH
Date of Performance:26/03/2024
Subject Name: Project Based learning with Java
Subject Code: 21CSH-319

1. **Aim:** Create a menu-based Java application with the following options.1. Add an Employee 2. Display All 3.Exit If option 1 is selected, the application should gather details of the employee like employee name, employee id, designation and salary and store it in a file. If option 2 is selected, the application should display all the employee details. If option 3 is selected the application should exit.

2. **Objective:** The objective of Program to learn about concept of File Handling in java and learn about LinkedList, Exception Handling in java.

3. Algorithm/Approach/Code:

```
import java.io.*;
import java.util.LinkedList;
import java.util.Scanner;
class Employee implements Serializable {
private static final long serialVersionUID = 1L;
private String name;
private int id;
private String designation;
private double salary;
public Employee(String name, int id, String designation, double salary) {
```

```
this.name = name; this.id = id; this.designation = designation; this.salary = salary;
}
@Override
public String toString() { return "Employee{" +
"name='" + name + '\'' + ", id=" + id +
", designation='" + designation + '\'' + ", salary=" + salary +
'}';
}
}
public class EmployeeManagement {
private static final String FILE_NAME = "employee_data.dat";
private static LinkedList<Employee> employeeList = new LinkedList<>();
public static void main(String[] args) { loadEmployeeDataFromFile();
Scanner scanner = new Scanner(System.in); boolean running = true;
while (running) { System.out.println("\nSelect an option:"); System.out.println("1. Add an
Employee"); System.out.println("2. Display All"); System.out.println("3. Exit");
int choice = scanner.nextInt();
scanner.nextLine(); // Consume the newline character
switch (choice) { case 1:
addEmployee(scanner); break;
case 2:
displayAllEmployees(); break;
case 3:
saveEmployeeDataToFile(); running = false; System.out.println("Exiting..."); break;
default:
System.out.println("Invalid choice!");
}
}
```

```
scanner.close();

}

private static void addEmployee(Scanner scanner) { System.out.println("Enter Employee
Name:"); String name = scanner.nextLine();

System.out.println("Enter Employee ID:"); int id = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

System.out.println("Enter Employee Designation:"); String designation = scanner.nextLine();

System.out.println("Enter Employee Salary:");

double salary = scanner.nextDouble(); scanner.nextLine(); // Consume the newline character

Employee employee = new Employee(name, id, designation, salary);
employeeList.add(employee);

System.out.println("Employee added successfully.");

}

private static void displayAllEmployees() { if (employeeList.isEmpty()) {

System.out.println("No employees to display.");

} else {

System.out.println("Employee details:"); for (Employee employee : employeeList) {

System.out.println(employee);

}

}

}

private static void loadEmployeeDataFromFile() {

try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(FILE_NAME)))
{

employeeList = (LinkedList<Employee>) ois.readObject(); System.out.println("Employee
data loaded successfully.");

} catch (FileNotFoundException e) {

System.out.println("No previous data found. Starting fresh.");

} catch (IOException | ClassNotFoundException e) { System.out.println("Error loading
employee data from file: " +
```

```
e.getMessage());
}
}
private static void saveEmployeeDataToFile() {
try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
oos.writeObject(employeeList); System.out.println("Employee data saved successfully.");
} catch (IOException e) {
System.out.println("Error saving employee data to file: " + e.getMessage());
}
}
}
```

4. Output:

```
No previous data found. Starting fresh.

Select an option:
1. Add an Employee
2. Display All
3. Exit
1
Enter Employee Name:
Ram
Enter Employee ID:
100
Enter Employee Designation:
SDE
Enter Employee Salary:
45000
Employee added successfully.

Select an option:
1. Add an Employee
2. Display All
3. Exit
1
Enter Employee Name:
Mohan
Enter Employee ID:
101
Enter Employee Designation:
Manager
Enter Employee Salary:
50000
Employee added successfully.
```

```
Select an option:
1. Add an Employee
2. Display All
3. Exit
1
Enter Employee Name:
Karan
Enter Employee ID:
102
Enter Employee Designation:
WebDev
Enter Employee Salary:
45000
Employee added successfully.

Select an option:
1. Add an Employee
2. Display All
3. Exit
2
Employee details:
Employee{name='Ram', id=100, designation='SDE', salary=45000.0}
Employee{name='Mohan', id=101, designation='Manager', salary=50000.0}
Employee{name='Karan', id=102, designation='WebDev', salary=45000.0}

Select an option:
1. Add an Employee
2. Display All
3. Exit
3
Employee data saved successfully.
Exiting...


...Program finished with exit code 0
Press ENTER to exit console.
```

5. Learning Outcomes:

1) Array of list to store multiple string Information
2) Oops concept
3) Switch Statements
4) File Handling