****

**Data Structures with JAVA laboratory**

**A mini project report**

**on**

**EMPLOYEE DATA MANAGEMENT**

**Submitted by**

**1PI11CS154, SARTHAK GUPTA**

**1PI11CS158, SHARATH K**

**1PI11CS165, SHREYAS T**

**1PI11CS181, SYED SAMEER AZHAR**

Department of Computer Science & Engineering

**PES INSTITUTE OF TECHNOLOGY**

*(An Autonomous Institute under VTU Belgaum)*

**100 Feet Road, BSK III Stage, Hosakerehalli, Bengaluru - 560 085**

****

**PES INSTITUTE OF TECHNOLOGY**

*(An Autonomous Institute under VTU Belgaum)*

**100 Feet Road, BSK III Stage, Hosakerehalli, Bengaluru - 560 085**

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the mini project entitled “Employee data management ” has been carried out by

**1PI11CS154, SARTHAK GUPTA**

**1PI11CS158, SHARATH K**

**1PI11CS165, SHREYAS T**

**1PI11CS181, SYED SAMEER AZHAR**

in the partial fulfillment of third semester data structures with JAVA laboratory [ 11CS207].

Signature of the Signature of the

**Lab Incharge Head - CSE**

**Name & Signature of the Examiners:**

Examiner 1:

Examiner 2:

**ACKNOWLEDGEMENT**

I take this opportunity to express my profound gratitude and deep regards to my guide Prof. V.R. Badri Prasad, Ms. Priya Badrinath, for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by him time to time shall carry me a long way in the journey of life on which I am about to embark.

I also take this opportunity to express a deep sense of gratitude to Prof. Dinesh Singh for their cordial support, valuable information and guidance, which helped me in completing this task through various stages.

I am obliged to my group members Sarthak,Sharath,Shreyas and Sameer for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of my assignment.

Lastly, I thank almighty, my parents, brother, sisters and friends for their constant encouragement without which this assignment would not be possible.

**INDEX**

|  |  |  |
| --- | --- | --- |
| **SERIAL NO.** | **TOPIC** | **PAGE NO.** |
| 1. | Abstract | 5 |
| 2. | Introduction | 6 |
| 3. | Objective | 7 |
| 4. | Characteristics of Java, Java Swings & Netbeans | 8 |
| 5. | Design | 11 |
| 6. | Implementation Details | 12 |
| 7. | Flow Chart | 14 |
| 8. | Test Cases & Snapshots | 15 |
| 9. | Conclusion | 17 |
| 10 | Scope for improvement | 17 |
| 11. | Bibliography | 19 |

**ABSTRACT**

We are doing a project which is about storing the data(details) of employees of a company or any other industry.

In this project we are providing many functions to store and manipulate the data of employees such as

\* Storing the details like name of the employee, ID, salary, date of join etc.

\* Adding a new employee details into the existing list.

\* Providing the report of required employees.

\* To provide details of a specific employee.

\* To give the sorted list of employees based on name, ID, salary, date of join.

\* This project also gives the report of employees with the certain specifications.

For this project we are using data structure-linked list to store the details of employees and also we can add details of a new employee to the existing list. We are implementing the method to search for an employee and give details provided his name or ID. In this project we also have a method to sort the list of employees based on name, ID, salary, date of join using different data structures. This project allows user to get the details of certain number of employees from the list. We can also use this project to get the list of employees with certain specifications like number employees whose salary is in the given range.

**INTRODUCTION**

Our project “Employee Data Base Management” is an application which we have created helps to manage the details of employees associated with the company. Since in a company there are many employees who have worked or are working, to manage all their details such as their name, ID, date of join and salary etc., so to manage these data by conventional methods is cumbersome so we have developed this small application which is very helpful nowadays. This application will reduce the workload in generating the report, maintaining the data and updating it as per the user’s requirement.

In our project we have providing two user modes. First is the administrator and guest mode.

In our project the administrator mode gives the user the ability to

* Add a new employee’s details.
* Update the details of an existing employee.
* Delete (fired from the company) the details of a specific employee from the data base.
* Search for the details of an employee based name or his/her ID in both working and non working category.
* Display the list of all employees either in the working or non working category.

In our project the guest mode gives the user the to

* Search for the details of an employee based name or his/her ID in both working and non working category.
* Generate a report as the guest user desires for example generating the report in sorted manner based on name, ID.

**OBJECTIVE**

The main objective behind the development of this project is to create software with the following features:

1. Highly user-friendly
2. Enterprise independent
3. Cross-platform
4. Easy-to-use
5. Tested system to track unnoticed error
6. Data entry restricted to valid domains to avoid errors

**JAVA**

The entire component has been developed using Java technology. Java has been chosen as the platform because of its feature rich nature. The Java Platform provides robust end-to-end solutions for networked applications as well as a trusted standard for embedded applications. So Java was a natural choice for development process.

**Characteristics of Java**

**Object Oriented:**

                    Java is object oriented to the truest sense of the word. Everything in Java is represented as objects. Variables and methods both are encapsulated in objects. Java is the purest object-oriented language.

**Robust**:

1. Java is a very robust language owing to the following features.
2. Excellent exception handling facilities.
3. Memory management relief for the user. User does not have to worry about allocation and deallocation of memory.
4. Strict compile-time and runtime checks for data types.

**Portable and Architecture-neutral (Platform Independent):**

Java is portable and platform independent so much that they satisfy “write once; run anywhere, anytime, forever”. This feature is implemented in the following ways:

1. Compiler generates  machine independent byte-code instructions which can be run on any machine supporting Java Virtual Machine.
2. Size of primitive data type is machine independent.

**Multithreaded:**

1. Programs can do many things simultaneously using different threads.
2. Provides a solution for multiprocess synchronization.
3. Allows the creation of networked and interactive programs.

**Distributed:**

1. Open access  to remote objects by the use of RMI(Remote Method Invocation).
2. Brings a level of abstraction to client/server programming.

Secure:

1. Security is achieved by confining a java program to the java execution      environment and not allowing access to other parts of the user computer.
2. Absence of pointers provide memory related security as encroachment of memory    is avoided Proper measures for prevention of viral infection and malicious intent.

**Dynamic and Extensible**:

1. Facilitates linking in of new classes, objects and methods.
2. Supports native methods (methods written in other languages like C ,C++).
3. Programs carry with them a substantial amount of runtime type information that is used to verify and resolve accesses to objects at run-time.

**High Performance:**

                Just-In-Time (JIT) compilers are used to convert byte-code into native machine code resulting in very high performance. These JIT compilers can be used on a real time, piece by piece demand basis to perform on-the-fly compilation of byte-code into native-code.

**Compilation and Interpretation**

Java programs are implemented as a two-stage system.

Compilation: Source code to byte-code and not machine instructions.

Interpretation: Byte-code to machine code (for any system that supports using JVM)

Thus cross-platform programs can be written.

**JAVA  SWINGS**

"Swing" refers to the new library of GUI controls (buttons, sliders, checkboxes, etc.) that replaces the somewhat weak and inflexible AWT controls.

 The Swing classes eliminate Java's biggest weakness: its relatively primitive user interface toolkit. Java Swing helps you to take full advantage of the Swing classes, providing detailed descriptions of every class and interface in the key Swing packages. It shows you how to use all of the new components, allowing you to build state-of-the-art user interfaces and giving you the context you need to understand what you're doing. It's more than documentation; Java Swing helps you develop code quickly and effectively.

**NETBEANS**

NetBeans is an integrated development environment (IDE) for developing primarily with Java, but also with other languages, in particular PHP, C/C++, and HTML5. It is also an application platform framework for Java desktop applications and others.

The NetBeans IDE is written in Java and can run on Windows, OS X, Linux, Solaris and other platforms supporting a compatible JVM.

The NetBeans Platform allows applications to be developed from a set of modular software components called *modules*.

## NetBeans Platform

The NetBeans Platform is a reusable framework for simplifying the development of Java Swing desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download digitally signed upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

* User interface management (e.g. menus and toolbars)
* User settings management
* Storage management (saving and loading any kind of data)
* Window management
* Wizard framework (supports step-by-step dialogs)
* NetBeans Visual Library
* Integrated development tools

NetBeans IDE is a free, open-source, cross-platform IDE with built-in-support for Java Programming Language

**DESIGN ( DATA STRUCTURE OVERVIEW )**

**LIST DESIGN:**

elist

ENode1 ENode2 ENode3 ……. ENode n

ENode Structure:

**LEFT DATA RIGHT**

Name of the employee

ID of the employee

Designation

Gender

Salary

Date of Join

We are using the node ENode to form a doubly linked list and in each node we are storing the details of one employee.

In each node we are having different fields as follows

* Left- Address of the previous node
* Name-Name of the employee
* ID-Employee’s ID
* Position-Employee’s position in the company/firm
* Gender-Gender of the employee
* Salary-Employee’s annual income
* Date of join-Date on which employee joined the company
* Right-Address of the next node

**IMPLEMENTATION**

* **Class ENode**

In this class we have fields to store the details of employees. In each node we are having different fields as follows

* Left- Address of the previous node
* Name-Name of the employee
* ID-Employee’s ID
* Position-Employee’s position in the company/firm
* Gender-Gender of the employee
* Salary-Employee’s annual income
* Date of join-Date on which employee joined the company
* Right-Address of the next node
* **Class EList**

In this class we have implemented the following methods

* **addEmp** :this method does the job of adding the ENode node of employee details to the list-elist. And also this method is used to add ENode read from the file(“employee.dat”) again to the elist.
* **delEmp** : this method does the job of deleting the node ENode from working employees list elist and calls the method naddEmp and passes the deleted node as parameter.
* **naddEmp** : this method does the job of adding the ENode node of employee details which is deleted from elist to the list-nlist. And also this method is used to add ENode read from file (“nemployee.dat”) again the nlist.
* **update** : this method allows to edit and resave the details of an employee asked by the user.
* **fread** : this method reads the data from both files “employee.dat” and “nemployee.dat” and creates a node and calls addEmp method to form the doubly linked list.
* **display** : this method displays the list of employees in a display window
* **Class ESearch**

In this class we have implemented the following methods

* **nsearch** : this method searches for the entered name and displays the details of that employee. If the name is not found in the list then an error message "The name you entered not found” is printed.
* **idsearch**: this method searches for the entered ID and displays the details of that employee. If the ID is not found in the list then an error message "The ID you entered not found” is printed.
* **Salsearch** : this method gives the list of employees who all have salary in the given range. If no employees in the database are having salary within the entered range then an error message "The salary range you entered not found” is printed
* **dojsearch** : this method gives the list employees who joined the company in the specified range of date. If no employees in the database have joined within the entered range then an error message ”No employees joined during this time” is printed.
* **Class ESort** :

In this class we have implemented the following methods

* **nsort** : this method gives the sorted list of employees based on name. If there the list is empty then the message “NO DATA EXISTS” is printed.
* **idsort** : this method gives the sorted list of employees based on ID. If there the list is empty then the message “NO DATA EXISTS” is printed.
* **Class EFile** :

In this class we have implemented the following methods

* **fwrite1** : this method writes the data of list-elist the list into the file “employee.dat” whenever the list is edited. We are using @ as delimiter.
* **Fwrite2** : this method writes the data of list-nlist the list into the file “nmployee.dat” whenever the list is edited. We are using @ as delimiter.

**FLOW CHART**

**EMPLOYEE DATA MANAGEMENT**

Sorted by name  
Sorted by ID

Between range of

salaries

Between range of

Date of join

GENERATE A REPORT

Based on name

Based on name

Working

Non working-based on name

ADD

Based on ID

Based on name

Working

Working

Non working

This method asks the user to enter the ID of the employee to be updated then searches for that ID and allows user to update the details

This method asks the user to enter the ID of the employee to be deleted then searches for that ID deletes it from the list

**GUEST MODE**

DISPLAY

SEARCH

UPDATE

DELETE

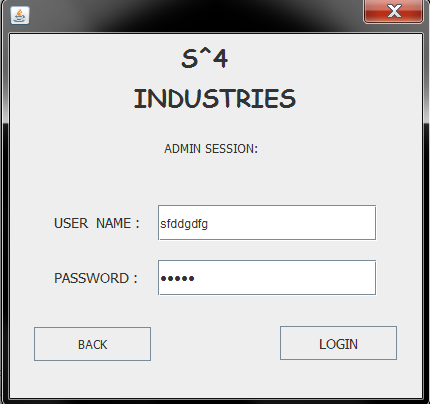
SEARCH

**ADMIN MODE**

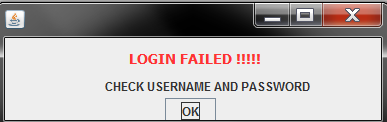
Username Password

**TEST CASES & SNAPSHOTS**

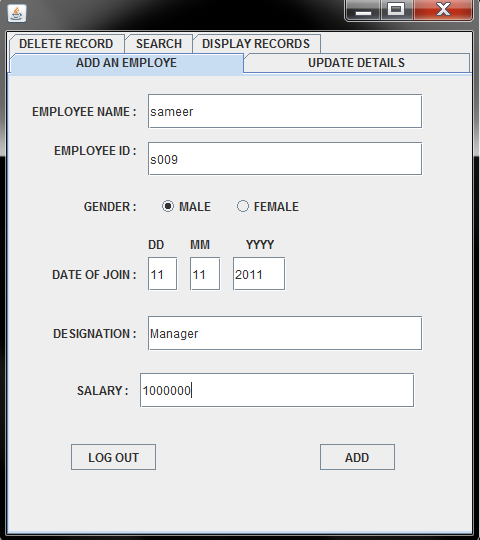
****

****

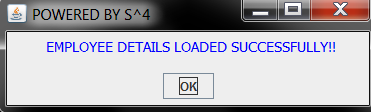
If user enters the invalid password or username then below shown window pops out

****

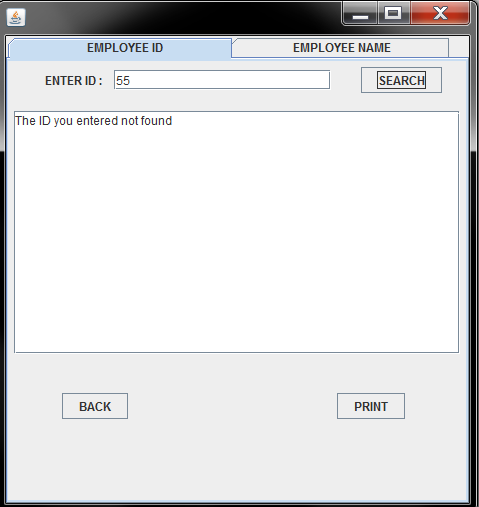
This is the window where the user adds the new employee details.



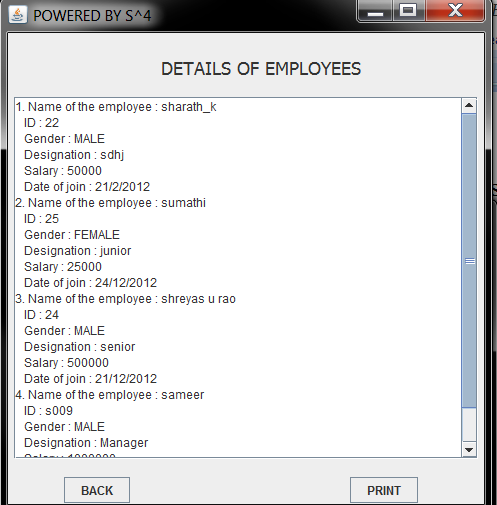
After adding, this window pops out :



If the employee searched is not found the below command is shown :



Displaying list of all working employees:



**CONCLUSION**

This was our project based upon a Employee data base management System. We think this system will give a better way to a shopkeeper to maintain his shop. Though every task is never said to be perfect in this development field even more improvement are possible in this system which we have mentioned below and some more improvement can also be done to give a more user friendly system which can help the shopkeeper more efficiently. We take this opportunity to express our sense of indebtedness and gratitude to all those people who helped us in completing this project. This project has contributed a lot to my knowledge that has proved to be a value addition for me.

**SCOPE FOR IMPROVEMENT**

* All frame objects are create as static hence when we move to next frame and return back to the previous frame the content of the frame will be stored. Therefore we are using reset button everywhere.
* In our project in some of frames we have a button PRINT to get the print out of the report generated but we have not given any action to that button.

**BIBLIOGRAPHY**

**BOOKS REFERRED –**

1. Information Practices – Sumita Arora
2. Java : The Complete Reference – Herbert Schildt
3. Programming with Java – E. Balaguruswamy

**SITES REFERRED –**

1. Wikipedia.org