PROJECT NO 1:

EMPLOYEE PAYROLL SYSTEM

Name: Sugu Priya S
Reg no : RA2411042020055
CSBS II A

OBJECTIVE

The goal of this project is to provide a Java-based staff payroll management system that facilitates the effective storage, administration, and retrieval of personnel information. By automating payroll data handling and guaranteeing correctness in staff information, including staff ID, name, designation, and compensation, the system aims to minimize human labor. It illustrates how object-oriented concepts in Java can be applied to create practical applications.

Understanding and putting into practice Java file handling and object serialization for long-term data storage is another goal of this project. Staff records can be saved in a binary file and retrieved as needed thanks to the system. Enhancing Java programming abilities, such as exception handling, modular design, and class interaction—all crucial for professional software development—is another goal of this project.

PROBLEM DESCRIPTION

Payroll information for employees is still handled manually or with simple spreadsheet programs in many firms. These ineffective techniques might result in redundant data, inaccurate pay records, and trouble finding personnel information. Additionally, manual methods are more prone to human error and need more time and effort to maintain.

A computerized payroll administration system is required to get around these restrictions. A Java-based system can lessen reliance on manual record keeping, enable fast personnel ID searches, and securely store staff data. In order to solve these problems, this project uses Java and file handling techniques to create a straightforward, dependable, and effective payroll system.

QUESTION:

a. Create Employee class with ID, Name, Basic Salary

b. Add employee details

c. Calculate HRA (20%), DA (10%), Net Salary

d. Store and read employee records using file

e. Search employee by ID

f. Use constructors and method overloading

CLASS DESIGN

The project is organized into several classes, each of which is in charge of a certain task, and it uses an Object-Oriented Programming (OOP) methodology. Code readability, reusability, and maintenance are all enhanced by this modular approach.

1.Class of Staff

In the system, the staff entity is represented by the Staff class. It keeps track of every important aspect about a worker. Staff objects can be written into and read from a binary file thanks to this class's implementation of the Serializable interface.

- Qualities:
  int staffId: The employee's unique identification number
  String name: The employee's name
  String designation: The staff member's job role
  double salary: The staff's compensation
- Techniques:

  Initializing staff details with a parameterized constructor
  Techniques for getting private information with getter and setter
  To present staff information in a legible format, use the toString() method.

2. Class of Payroll

All file operations and staff data processing are under the purview of the Payroll class. It handles staff object storage in a binary file and retrieval as needed.
Features:

- Using ObjectOutputStream to write staff objects into a.bin file
- Reading staff objects using ObjectInputStream
- Searching staff details using staff ID
- Handling file-related exceptions

3. Main Class

The Main class contains the main() method and acts as the entry point of the application. It interacts with the user, accepts input values, and calls appropriate methods from the Payroll class. This class controls the flow of execution of the program.

CODE EXPLANATION

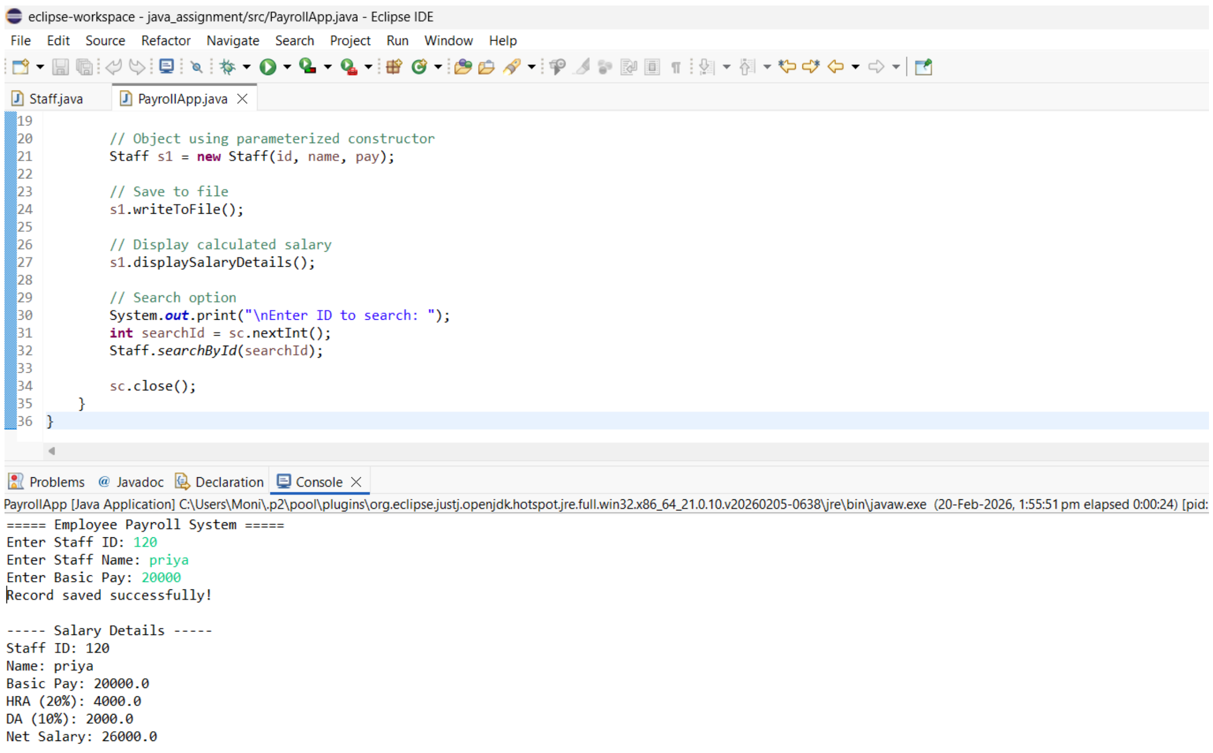
Program execution starts with the Main class, which serves as the user's entry point. Staff record creation and user interaction fall under the purview of the Main class. Employee information, including name, ID, and base pay, can be entered by users. Furthermore, it offers the ability to search personnel records by inputting the staff ID. The data is sent to the Payroll class, which manages all file operations and processing logic, after the user has entered the necessary information.

The template for each employee's record is the Staff class. Fields for storing staff ID, staff name, and base pay are included in every staff object. Methods for determining the staff member's net wage are also covered in the course. Allowances like Dearness Allowance (DA) and House Rent Allowance (HRA) are added to the base income to determine the net salary. The Staff class uses method overloading to enable the computation of net salary using either the user-provided custom rates or the default allowance rates. Additionally, the Staff class has display methods that offer flexibility in output presentation by presenting staff details in both summary and detailed versions.

The application uses file handling and serialization to keep the data permanently while adding staff entries. In particular, serialized staff objects are written to a binary file called staff_data.bin using FileOutputStream and ObjectOutputStream. A Java object can be saved to a file or sent over a network using serialization, which is the act of turning the object into a series of bytes. This guarantees that even after the program ends, all staff records will be accessible. It is also easier to save and retrieve complicated objects when serialization is used for storage rather than manually turning each field into text.

The application uses FileInputStream and ObjectInputStream to read the binary file in order to search staff records. Every serialized object in the file is iterated through by the Payroll class, which deserializes each one and compares its staff ID with the ID that the user entered. The application uses the display methods of the Staff class to show the staff details if a matching record is discovered. Try-catch blocks are used to implement exception handling, which makes sure that unexpected mistakes don't cause the application to crash. These issues can include hitting the end of the file, trying to read from an invalid file, or running into a corrupted file format. Appropriate exception handling guarantees a seamless user experience and strengthens the program's resilience.

INPUT/OUTPUTS

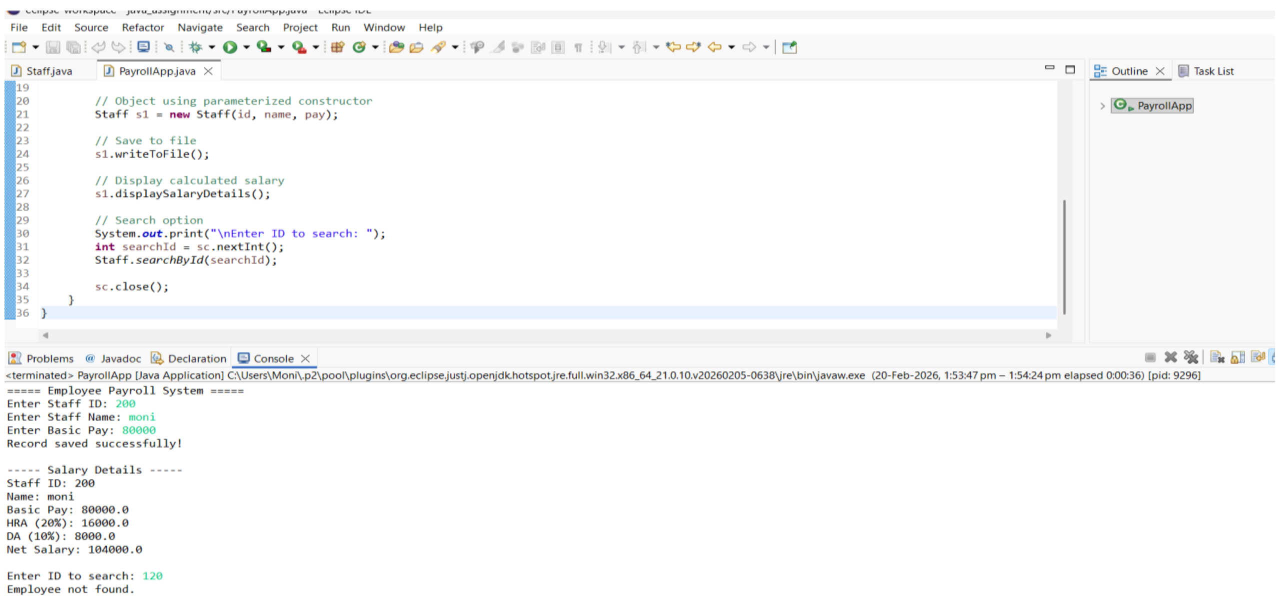


eclipse-workspace - java_assignment/src/PayrollApp.java - Eclipse IDE

```java

        // Object using parameterized constructor
        Staff s1 = new Staff(id, name, pay);

        // Save to file
        s1.writeToFile();

        // Display calculated salary
        s1.displaySalaryDetails();

        // Search option
        System.out.print("\nEnter ID to search: ");
        int searchId = sc.nextInt();
        Staff.searchById(searchId);

        sc.close();
    }
}
```

PayrollApp [Java Application] C:\Users\Moni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.10.v20260205-0638\jre\bin\javaw.exe (20-Feb-2026, 1:55:51 pm elapsed 0:00:24) [pid:

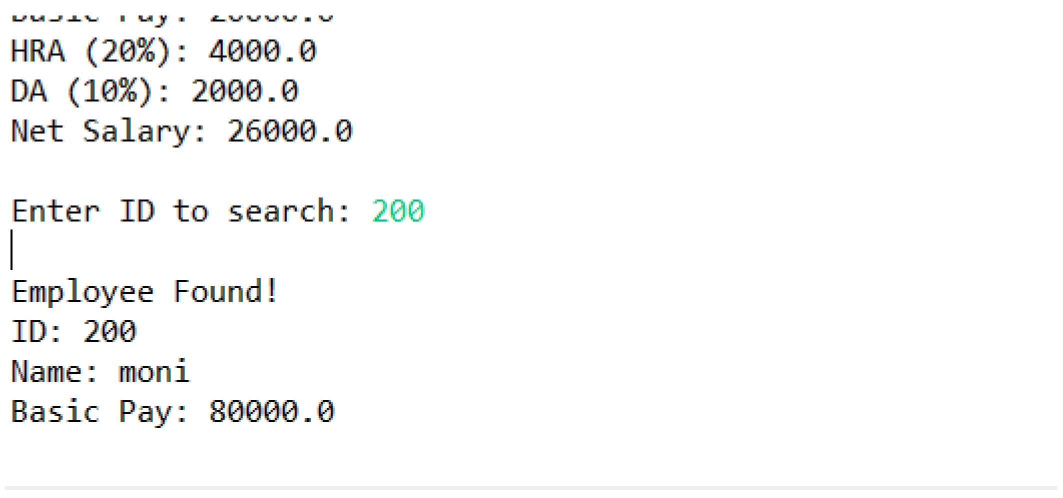
```
===== Employee Payroll System =====
Enter Staff ID: 120
Enter Staff Name: priya
Enter Basic Pay: 20000
Record saved successfully!

----- Salary Details -----
Staff ID: 120
Name: priya
Basic Pay: 20000.0
HRA (20%): 4000.0
DA (10%): 2000.0
Net Salary: 26000.0
```



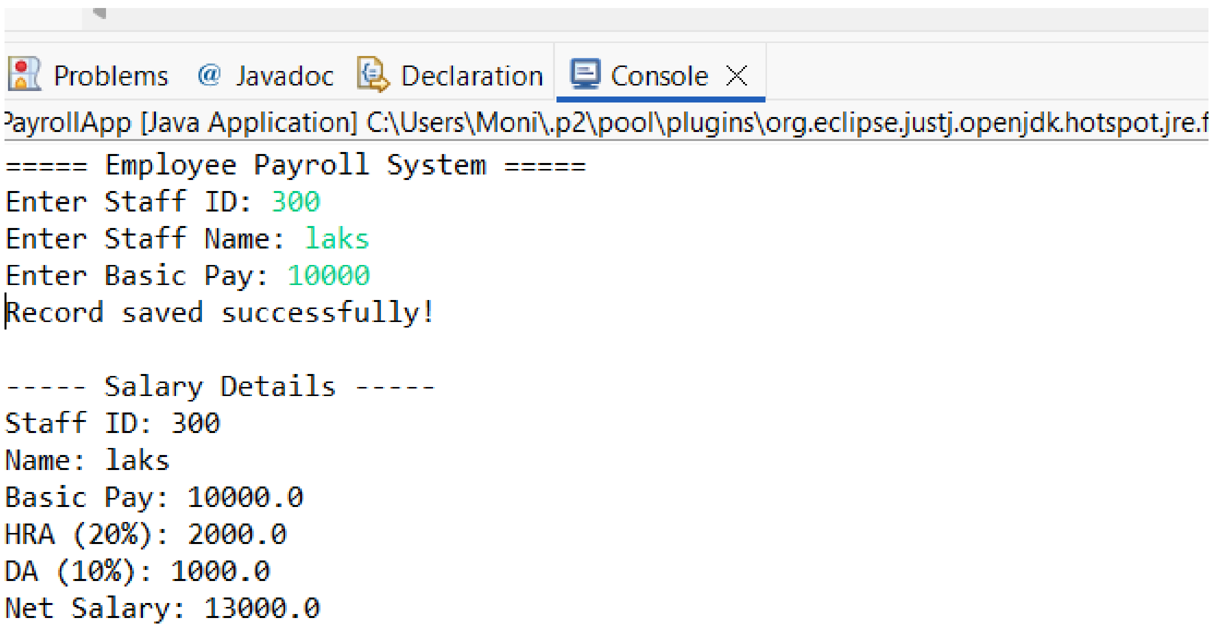
```java

        // Object using parameterized constructor
        Staff s1 = new Staff(id, name, pay);

        // Save to file
        s1.writeToFile();

        // Display calculated salary
        s1.displaySalaryDetails();

        // Search option
        System.out.print("\nEnter ID to search: ");
        int searchId = sc.nextInt();
        Staff.searchById(searchId);

        sc.close();
    }
}
```

<terminated> PayrollApp [Java Application] C:\Users\Moni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.10.v20260205-0638\jre\bin\javaw.exe (20-Feb-2026, 1:53:47 pm – 1:54:24 pm elapsed 0:00:36) [pid: 9296]

```
===== Employee Payroll System =====
Enter Staff ID: 200
Enter Staff Name: moni
Enter Basic Pay: 80000
Record saved successfully!

----- Salary Details -----
Staff ID: 200
Name: moni
Basic Pay: 80000.0
HRA (20%): 16000.0
DA (10%): 8000.0
Net Salary: 104000.0

Enter ID to search: 120
Employee not found.
```



```
HRA (20%): 4000.0
DA (10%): 2000.0
Net Salary: 26000.0

Enter ID to search: 200

Employee Found!
ID: 200
Name: moni
Basic Pay: 80000.0
```



```
ayrollApp [Java Application] C:\Users\Moni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.f
===== Employee Payroll System =====
Enter Staff ID: 300
Enter Staff Name: laks
Enter Basic Pay: 10000
Record saved successfully!

----- Salary Details -----
Staff ID: 300
Name: laks
Basic Pay: 10000.0
HRA (20%): 2000.0
DA (10%): 1000.0
Net Salary: 13000.0
```

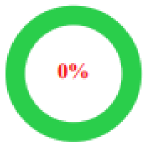
Github rep link : https://github.com/Sugupriya12/Java-assignment-

CONCLUSION

The Staff Payroll Management System using Java successfully meets its objective of automating staff data management. The system efficiently stores and retrieves payroll information using object-oriented design and file handling techniques. It minimizes manual effort, improves accuracy, and provides faster access to staff records.

This project also helped in gaining practical knowledge of Java concepts such as classes, objects, serialization, and exception handling. The system can be further enhanced by adding a graphical user interface (GUI), database connectivity, and advanced payroll features such as tax calculation and attendance tracking, making it suitable for real-world organizational use.

Plagiarism Detection Report by SmallSEOTOOLS



● Plagiarism	0%	● Partial Match	0%
● Exact Match	0%	● Unique	100%

Scan details

Total Words	Total Characters	Plagiarized Sentences	Unique Sentences
1025	7517	0	57 (100%)

#1 100% Unique

PROJECT NO 1:

EMPLOYEE

PAYROLL SYSTEM