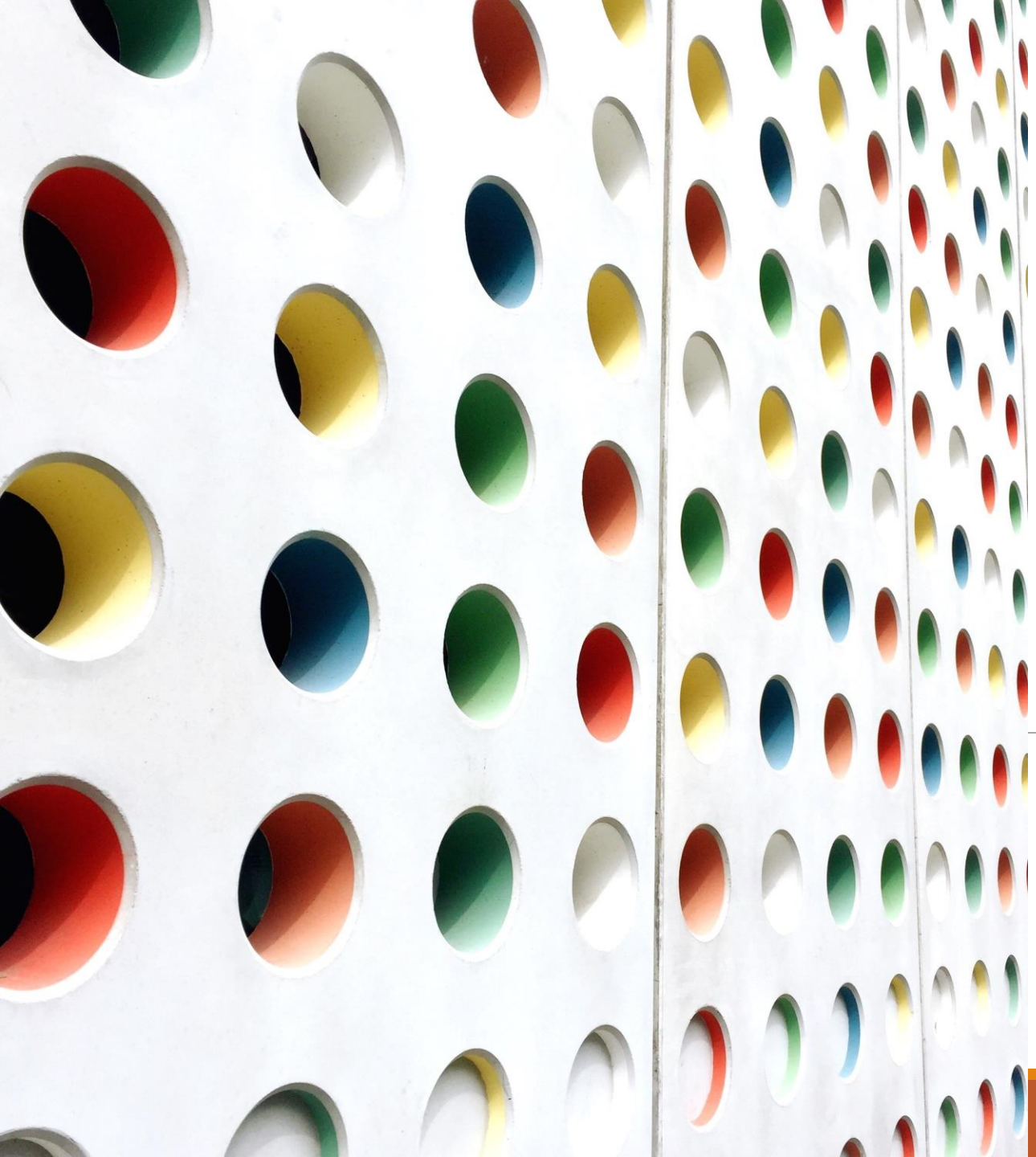


Semester long Assignment

PAYROLL MANAGEMENT SYSTEM

Submitted by :

Himanshu Kakkar
Nikhil Agrawal
Kushagra Richhariya
Tannu Agarwal

INTRODUCTION

- Payroll Management System is basically used to build an application program, that a company uses to manage the records of the employees working in the company.

- Only the administration has the legal rights to work with the system. Employees can only see their current status.

PAYROLL SYSTEM IS USED TO :

- Add Employees
- Remove Employees
- Employees Enquiry

➢ for employees according to their information given to the administration of the company.

Project Objectives

1. Efficiently manage employee data, including addition, removal, and display of information.
2. Validate user input for employee IDs and salaries to maintain data integrity.
3. Implement core functionalities such as adding, removing, and displaying employee information.
4. Handle errors gracefully to ensure a seamless user experience.
5. Implement security measures to protect sensitive employee data.
6. Design the system for scalability to accommodate future growth and workload.

EMPLOYEE MANAGEMENT :

Employee Addition :

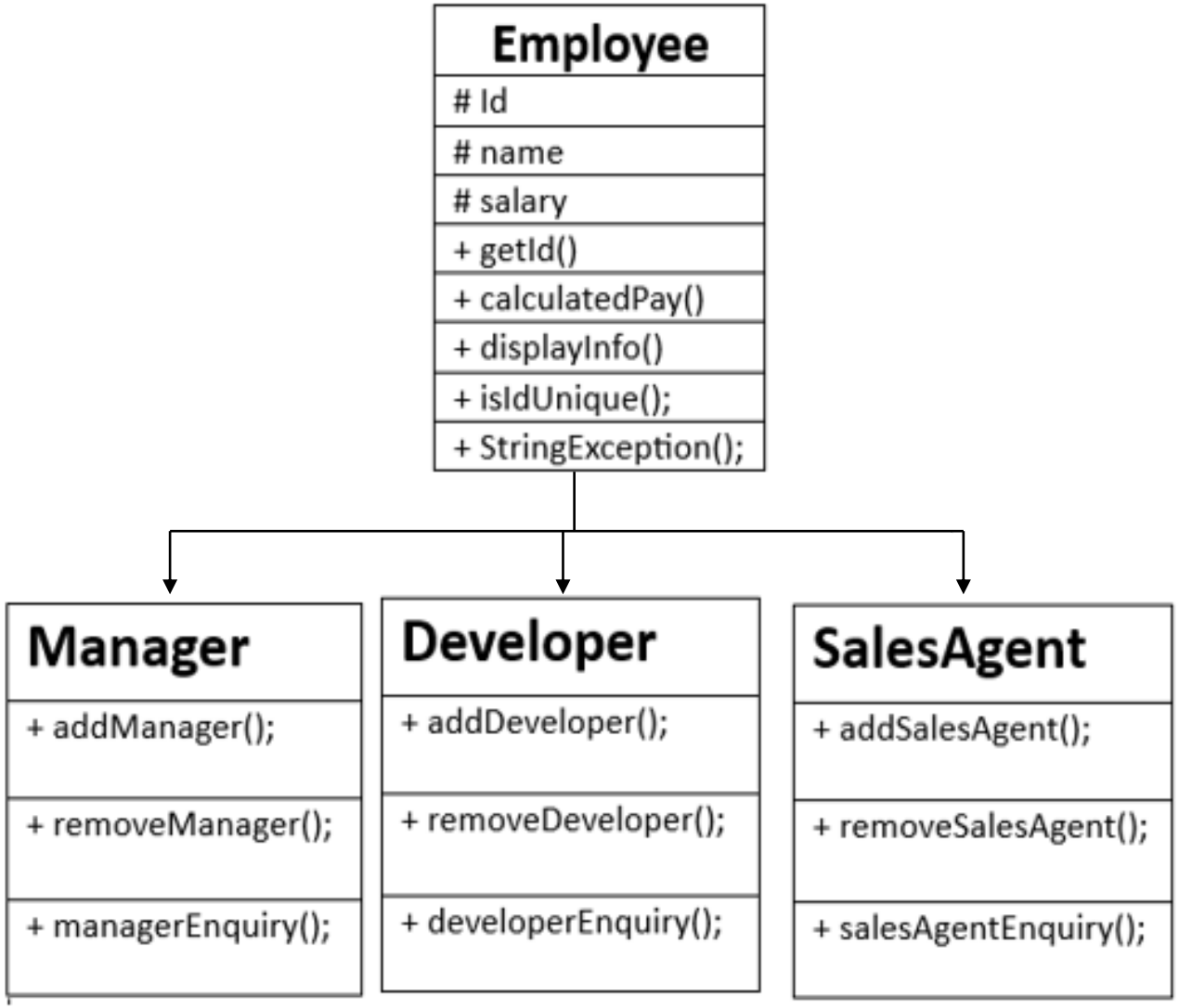
•Utilize appropriate methods (e.g., **addManager**, **addDeveloper**, **addSalesAgent**) based on the type of employee.
•Prompt user input for employee details such as ID, name, and salary.
•Ensure uniqueness of employee IDs to avoid duplication.
•Add the new employee to the respective list (managers, developers, or sales agents).

Employee Removal :

• Implement methods (e.g., **removeManager**, **removeDeveloper**, **removeSalesAgent**) to remove an employee.
•Prompt user input for the ID of the employee to be removed.
•Search the list of employees for the specified ID.
•If found, remove the employee from the list; otherwise, display a message indicating the employee was not found.

Employee Display :

- Implement methods (e.g., **managerEnquiry**, **developerEnquiry**, **salesAgentEnquiry**) to display employee information.
- Iterate through the list of employees of the respective type.
- Display each employee's details such as ID, name, and salary in a formatted manner.



PayrollSystem
- totalAnnualRevenue
- emp("xyz",0.0,0.0);
+ main();
- developerOperations();
- managerOperations();
- SalesAgentOperations();

ShowInfo
- calculateTotalSalary();
- managerEnquiry();
- developerEnquiry();
- salesAgent();
+ showInfo();
// file handling code
- concatenateLists();

ENCAPSULATION :

Implementation in the Code:

1. Methods for Interacting with Object Data:

1.1. Getters and setters methods are provided to interact with the object data indirectly, enabling controlled access to private attributes.

1.2. Getters retrieve the values of attributes, while setters modify them.

2. **Data Hiding using Access Modifiers**

2.1. In the code, encapsulation is achieved through the use of access modifiers such as private and protected.

2.2. Attributes of the Employee class (e.g., id, name, salary) are declared as private to restrict direct access from outside the class.

Example:

```
private int id;
private String name;
private double salary;
```

2. 3. Encapsulation Benefits in the Code:

3.1. By encapsulating attributes and providing controlled access through methods, the code ensures data integrity and prevents unauthorized modification of object state.

3.2. It allows for easy maintenance and modification of the code, as changes to internal implementation details can be made without affecting external users of the class.

INHERITANCE :

Subclasses Inheriting from Superclass :

In the code, subclasses such as Manager, Developer, and SalesAgent inherit from the superclass Employee.

These subclasses inherit attributes (e.g., id, name, salary) and methods (e.g., calculatePay(), displayInfo()) from the Employee class.all types of employees (managers, developers, sales agents) share common attributes such as id, name, and behaviors such as calculatePay().

By inheriting these from the Employee superclass, subclasses avoid redundancy and promote code maintainability.

Example:

```
class Manager extends Employee {
   // Manager-specific attributes and methods
}
```

ERROR HANDLING :

1. **Handling Errors in the System:**
    1. Implement error handling mechanisms to address various types of errors that may occur during system operation.
    2. Identify potential error scenarios such as invalid user inputs, duplicate IDs, or unexpected program behavior.

2. **Strategies for Providing Informative Error Messages:**
    1. Utilize descriptive error messages to clearly communicate the nature of the error to the user.
    2. Include guidance or instructions on how users can resolve the error or provide valid input.
    3. Ensure that error messages are concise, specific, and user-friendly to facilitate understanding and troubleshooting.

3. **Implementation in Code:**
    1. Integrate conditional statements and exception handling mechanisms to detect and respond to errors.
    2. Display informative error messages using print statements or user interface elements.
    3. Provide prompts or suggestions for corrective actions to guide users in resolving errors effectively.

IMPLEMENTATION DETAILS :

Overview of Code Structure and Functionality:

The code implements a Payroll Management System using Java, leveraging object-oriented programming principles.

Key components include classes for managing different types of employees (Manager, Developer, SalesAgent) and core functionalities for employee management.

Description of Key Components:

1. **Employee Class:**

1.1. Serves as the superclass for all types of employees.

1.2. Contains attributes such as id, name, and salary, along with methods for calculating pay and displaying employee information.

Manager Class :

Subclass of Employee representing managers in the organization.

Inherits attributes and methods from the Employee class and may include additional attributes and behaviors specific to managers.

Developer Class:

Subclass of Employee representing developers in the organization.

Inherits attributes and methods from the Employee class and may include additional attributes and behaviors specific to developers.

SalesAgent Class:

Subclass of Employee representing sales agents in the organization.

Inherits attributes and methods from the Employee class and may include additional attributes and behaviors specific to sales agents.

CORE FUNCTIONALITIES :

1. **Adding, Removing, and Displaying Employee Information:**
   - Implement methods (**addManager**, **addDeveloper**, **addSalesAgent**, **removeManager**, **removeDeveloper**, **removeSalesAgent**, **managerEnquiry**, **developerEnquiry**, **salesAgentEnquiry**) for adding, removing, and displaying employee information.
   - Utilize user input to add new employees and prompt for employee IDs, names, and salaries.
   - Verify the uniqueness of employee IDs before adding to ensure data integrity.
   - Display employee information including ID, name, and salary in a formatted manner.
2. **Revenue Calculation:**
   - Calculate the total revenue by summing up the salaries of all employees (managers, developers, and sales agents).
   - Implement a method to calculate the remaining revenue by subtracting the total salary expenses from the total revenue.
   - Ensure accuracy and reliability of revenue calculations to provide meaningful financial insights.

CONCLUSION :

1. **Utilization of OOP Concepts:**

The project extensively utilizes core object-oriented programming (OOP) concepts, including **encapsulation, inheritance, and polymorphism**.

Encapsulation is employed to encapsulate employee data and behaviors within classes, ensuring data security and abstraction.

Inheritance facilitates code reuse and organization by allowing subclasses to inherit attributes and methods from superclasses.

Polymorphism enables the treatment of objects of different types through a common interface, promoting code flexibility and extensibility.

2. Key Features and Functionalities:

The Payroll Management System encompasses a range of features, including employee management (addition, removal, and display of employee information), data validation (ensuring integrity of employee IDs and salaries), core functionalities (calculating total and remaining revenue), and error handling (providing informative error messages).

Thank You…