ALGONQUIN COLLEGE

CST8288 Assignment #1

In this task, your objective is to apply **UML** and **SOLID principles** to a coding exercise. Your task is to apply SOLID principles to enhance the design and maintainability of the code. This may involve refactoring, implementing interfaces, or creating new classes. Your goal is to improve the codebase by making it more modular, extensible, and in accordance with SOLID principles. The provided code is structured as a **Maven project** and consists of several classes.

1. **Employee.java:** A Plain Old Java Object (POJO) containing attributes related to employees.
2. **EmployeeController.java:** The controller class responsible for processing employee data.
3. **EmployeeService.java:** Interface defining various services related to different types of employees.
4. **Formatter.java:** Interface to format employee data in the desired format
5. **JSONFormatter.java:** An implementation of formatting employee data in JSON format
6. **Main.java:** The main program executes the application.
7. **Rate.java:** Class that has the rates for calculation

Part 1:

1. Examine the **EmployeeController.java** class to identify any violations of the **Single Responsibility Principle**. Adjust the class to adhere to this principle, considering the potential introduction of additional classes to maintain compliance.

2. Evaluate the **EmployeeService.java** interface and implement the **Interface Segregation Principle** to refactor it. Take into account the distinct needs of two employee classes: **PermanentEmployeeImpl.java** and **ContractEmployeeImpl.java**. Ensure that salary, bonus, pension, and renewal date are appropriately calculated based on the employee type. The requirements are as follows:
   a. All employees receive a salary and salary is a positive integer greater than 0.
   b. Total compensation is calculated as **salary + bonus.**
   c. Bonus and Pension are ONLY relevant for Permanent Employees.
   d. Bonus is **2.5%** of base salary and calculated multiplying by the years of service.
   e. Pension contribution is **1%** of base salary.
   f. Renewal date is only applicable for Contract Employees and is calculated 1 year from today.

3. Implement a class named **TextFormatter.java** that implements the **Formatter.java** interface. This class should be responsible for saving the **Employee** object as key-value pairs, such as (name=xxxx, email=xyz@abc.com).

4. Analyze the **saveEmployee(…)** method in the **EmployeeController.java** class to identify how it violates the **Dependency Inversion Principle**. You may want to put the code into a class named **PersistenceService.java** and modify it accordingly to align with the **Dependency Inversion Principle**.

5. Implement **Main.java** class to execute your solution:
   - Instantiate two **Employee** objects and use **PermanentEmployeeImpl.java** and **ContractEmployeeImpl.java** to calculate and populate required data for objects.
   - Utilize **EmployeeController.java** to save both objects in JSON and Text formats, saving the data in files named **json_employee_data.txt** and **text_employee_data.txt**.
   - Do the same thing to output the data to the **Console** instead of saving it to a file.

Part 2:

1. Develop JUnit tests to verify the functionalities of **PermanentEmployeeImpl.java**, **ContractEmployeeImpl.java**, and **PersistenceService.java** classes.

Part 3:

1. Finally, generate a report (limited to 2 pages) comprising a UML Class Diagram, outlining the purpose of each newly introduced class and detailing the application of SOLID principles in your solution.

Lab Deliverables

- This task constitutes 7.5% of your overall grade and due by **June 2, 2024 (Sunday), 11:59 PM.**
- Deliver a comprehensive coding solution.
- Provide the report outlined in part 3.
- You code should include proper commenting and coding practice according to Java standards.
- You must demo your solution during the lab session and submit your code (your project zip file) and report on **Brightspace.** Name your project zip file as ***YourLastname-YourFirstname*-Assignment1.zip**.
- Violating academic integrity or missing the deadline will result in a grade of 0 for your submission.

Individual Assignment

You may verbally discuss the general approach to solving this individual assignment with other students and that is the only extent of collaboration allowed for this assignment. You are not allowed to work together and you are not allowed to share or read each other’s code/deliverables. If your code or any other deliverables for this assignment resemble with those of another student, you will be reported to the Academic Integrity Office for cheating/plagiarism investigation. Please refer to the Academic Integrity policy document (AA48) of the college at https://www.algonquincollege.com/policies/files/2021/09/AA48.pdf .