| **Week 3 Worksheet** |
| --- |

# 

| **Implementing Abstraction in Java** |
| --- |

**Answer the following:**

| 1. How did you implement abstraction in your program? Which classes did you define as abstract or interfaces, and why?   We implemented abstraction by hiding the underlying raw data of the employees by using classes to do the processing. The following classes were defined:   * Payroll Calculation Layer - hides the actual payroll data * Employee Information Management Layer - hides employee PII * Attendance Tracking Abstraction Layer -hides time-tracking data * Benefits Administration Abstraction Layer - hides additional employee compensation details * Leave Management Abstraction Layer - hides available and used leave credits * Taxation and Compliance Abstraction Layer - hides the tax computation of employees * Backup and Restore Layer - hides the whole database content by performing BUR through an interface * Password Management Layer- hides actual user passwords * DB Optimization and maintenance Layer * Reporting Layer - hides the underlying raw data * Audit Layer - hides the underlying raw data * RBAC Layer - hides the privilege assignment for administrators and managers |
| --- |

| 1. What common characteristics and behaviors did you identify for the products and services in the given scenario? How did you represent them in your abstract classes or interfaces?   The abstraction layers uniformly provide interfaces for receiving input and producing output, utilizing the functionality included in the classes and methods. This design supports a uniform and standardized approach throughout the program. |
| --- |

| 1. Explain how you ensured that the abstract methods in your abstract classes or interfaces were implemented correctly in the concrete classes.   Our methods and classes ensure accurate output by methodically processing the correct input. They painstakingly protect raw data integrity, providing only authorized access to pertinent fields. This strategy avoids unauthorized or inadvertent changes to critical raw data, assuring our program's robustness and reliability. |
| --- |

| 1. Discuss the benefits of using abstract classes and interfaces in your program. How did they help define a clear contract for the behavior of products and services?   Abstract classes and interfaces significantly enhance our program:   * Abstraction minimizes the complexity of data display by providing a simplified view. * Enhanced Security: Provides heightened security by limiting access to just relevant details. * Code Reusability: Reduces code duplication while increasing reusability and maintainability.   These benefits work together to create a clear contract for the behavior of our program's products and services, resulting in a more efficient and secure software architecture. |
| --- |