Design Patterns Description

Name : Facade Design Pattern.

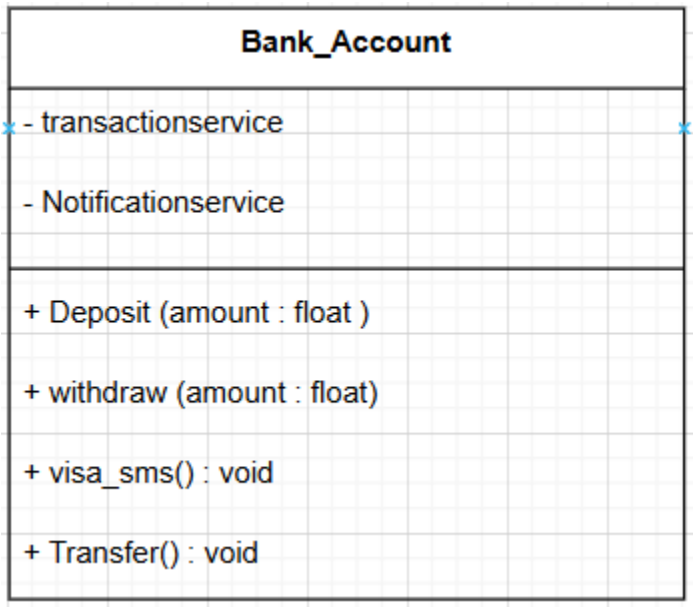
Context : A Bank_Account class that provides a simplified interface to complex banking operations.

Problem : How do you provide a simple, unified interface to a complex subsystem of banking operations without exposing implementation details?

Forces : There must be a way to hide the complexities of the banking system while providing users with easy-to-use methods for common banking operations.

Solution : Create a Bank_Account facade that provides simplified methods like Deposit() and visa_sms() that internally handle the complex interactions with various banking subsystems (transaction processing, notification services, etc.).

References : The Facade pattern is one of the structural patterns introduced by the "Gang of Four" in their influential work on design patterns.



Design Patterns Description

Name : Facade Design Pattern.

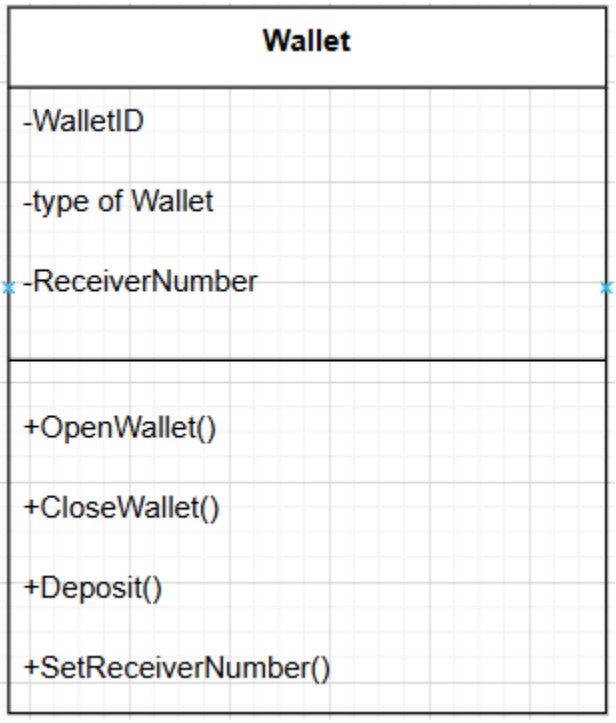
Context : A Bank_Account class that provides a simplified interface to complex banking operations.

Problem : How do you provide a simple, unified interface to a complex subsystem of banking operations without exposing implementation details?

Forces : There must be a way to hide the complexities of the banking system while providing users with easy-to-use methods for common banking operations.

Solution : Create a Bank_Account facade that provides simplified methods like Deposit() and visa_sms() that internally handle the complex interactions with various banking subsystems.

References : The Facade pattern is one of the structural patterns introduced by the "Gang of Four" in their influential work on design patterns.



Name : Strategy Design Pattern.

Context :

-You need to process different types of transactions. -Each transaction type may require different processing algorithms.

Problem : How can you encapsulate various transaction processing behaviors and make them interchangeable without modifying the Transaction class?

Forces : You want to switch between different transaction processing strategies during runtime without complicating the main Transaction class.

Solution : Define a family of notification algorithms, encapsulate each in the Notification interface, and make Transaction use this interface. This allows the Transaction class to vary its notification behavior independently from clients that use it.

References : The Strategy pattern is one of the behavioral patterns described in the book "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides (Gang of Four), published in 1994