**Exercise 8: Implementing the Strategy Pattern**

**Scenario:**

You are developing a payment system where different payment methods (e.g., Credit Card, PayPal) can be selected at runtime. Use the Strategy Pattern to achieve this.

**Steps:**

**1. Create a New Java Project:**

* **Project Name**: StrategyPatternExample
* Use your preferred IDE (like IntelliJ IDEA, Eclipse, or NetBeans) to create a new Java project.

**2. Define Strategy Interface:**

**PaymentStrategy Interface**:

* Create an interface PaymentStrategy with a method pay().

public interface PaymentStrategy {

void pay(double amount);

}

**3. Implement Concrete Strategies:**

**CreditCardPayment Class**:

* Create a class CreditCardPayment that implements PaymentStrategy.

public class CreditCardPayment implements PaymentStrategy {

private String cardNumber;

private String cardHolderName;

private String cvv;

public CreditCardPayment(String cardNumber, String cardHolderName, String cvv) {

this.cardNumber = cardNumber;

this.cardHolderName = cardHolderName;

this.cvv = cvv;

}

@Override

public void pay(double amount) {

System.out.println("Paid " + amount + " using Credit Card.");

}

}

**PayPalPayment Class**:

* Create a class PayPalPayment that implements PaymentStrategy.

public class PayPalPayment implements PaymentStrategy {

private String email;

private String password;

public PayPalPayment(String email, String password) {

this.email = email;

this.password = password;

}

@Override

public void pay(double amount) {

System.out.println("Paid " + amount + " using PayPal.");

}

}

**4. Implement Context Class:**

**PaymentContext Class**:

* Create a class PaymentContext that holds a reference to PaymentStrategy and a method to execute the strategy.

public class PaymentContext {

private PaymentStrategy paymentStrategy;

// Constructor to set the strategy

public PaymentContext(PaymentStrategy paymentStrategy) {

this.paymentStrategy = paymentStrategy;

}

// Method to execute the payment strategy

public void executePayment(double amount) {

paymentStrategy.pay(amount);

}

}

**5. Test the Strategy Implementation:**

**TestStrategyPattern Class**:

* Create a test class to demonstrate selecting and using different payment strategies

public class TestStrategyPattern {

public static void main(String[] args) {

// Select Credit Card payment strategy

PaymentStrategy creditCardPayment = new CreditCardPayment("1234-5678-9012-3456", "John Doe", "123");

PaymentContext context = new PaymentContext(creditCardPayment);

context.executePayment(150.00);

// Select PayPal payment strategy

PaymentStrategy payPalPayment = new PayPalPayment("john.doe@example.com", "password123");

context = new PaymentContext(payPalPayment);

context.executePayment(200.00);

}

}