**Exercise 4: Implementing the Adapter Pattern**

**Scenario:**

You are developing a payment processing system that needs to integrate with multiple third-party payment gateways with different interfaces. Use the Adapter Pattern to achieve this.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **AdapterPatternExample**.
2. **Define Target Interface:**
   * Create an interface **PaymentProcessor** with methods like **processPayment()**.
3. **Implement Adaptee Classes:**
   * Create classes for different payment gateways with their own methods.
4. **Implement the Adapter Class:**
   * Create an adapter class for each payment gateway that implements PaymentProcessor and translates the calls to the gateway-specific methods.
5. **Test the Adapter Implementation:**
   * Create a test class to demonstrate the use of different payment gateways through the adapter.

**CODE:-**

// Target Interface

interface PaymentProcessor {

    void processPayment(double amount);

}

// Adaptee Class 1 – PayPal

class PayPalGateway {

    public void sendPayment(double amount) {

        System.out.println("Processing PayPal payment of ₹" + amount);

    }

}

// Adaptee Class 2 – Stripe

class StripeGateway {

    public void makePayment(double amount) {

        System.out.println("Processing Stripe payment of ₹" + amount);

    }

}

// Adaptee Class 3 – Razorpay

class RazorpayGateway {

    public void doTransaction(double amount) {

        System.out.println("Processing Razorpay payment of ₹" + amount);

    }

}

// Adapter for PayPal

class PayPalAdapter implements PaymentProcessor {

    private PayPalGateway payPalGateway;

    public PayPalAdapter(PayPalGateway payPalGateway) {

        this.payPalGateway = payPalGateway;

    }

    public void processPayment(double amount) {

        payPalGateway.sendPayment(amount);

    }

}

// Adapter for Stripe

class StripeAdapter implements PaymentProcessor {

    private StripeGateway stripeGateway;

    public StripeAdapter(StripeGateway stripeGateway) {

        this.stripeGateway = stripeGateway;

    }

    public void processPayment(double amount) {

        stripeGateway.makePayment(amount);

    }

}

// Adapter for Razorpay

class RazorpayAdapter implements PaymentProcessor {

    private RazorpayGateway razorpayGateway;

    public RazorpayAdapter(RazorpayGateway razorpayGateway) {

        this.razorpayGateway = razorpayGateway;

    }

    public void processPayment(double amount) {

        razorpayGateway.doTransaction(amount);

    }

}

// Test Class

public class AdapterPatternExample {

    public static void main(String[] args) {

        PaymentProcessor paypal = new PayPalAdapter(new PayPalGateway());

        PaymentProcessor stripe = new StripeAdapter(new StripeGateway());

        PaymentProcessor razorpay = new RazorpayAdapter(new RazorpayGateway());

        System.out.println("=== Payment Processing ===");

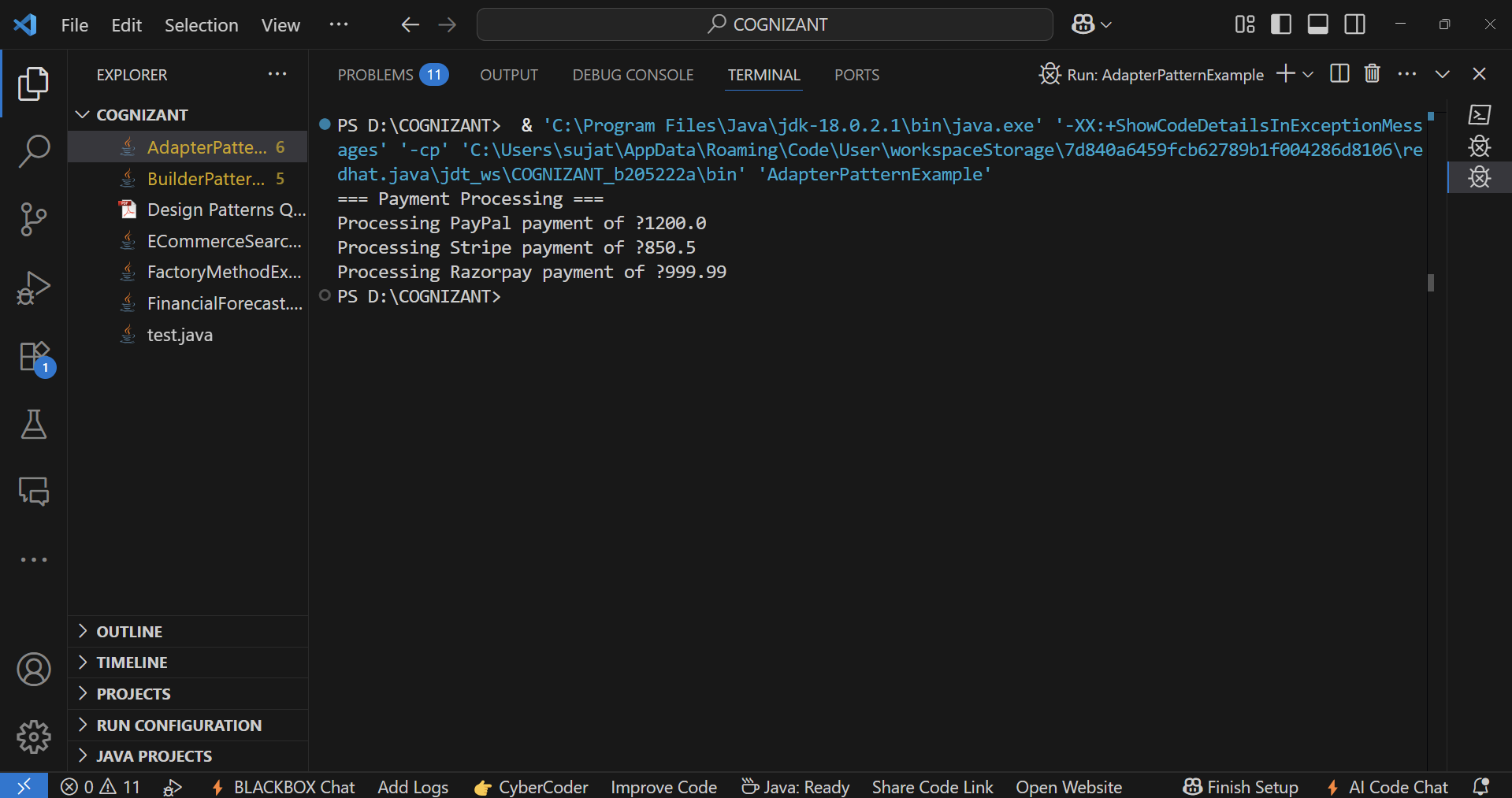
        paypal.processPayment(1200.00);

        stripe.processPayment(850.50);

        razorpay.processPayment(999.99);

    }

}

**OUTPUT:-**