Lab Exercise: Implementing the Strategy Design Pattern

Objective:

The goal of this lab exercise is to practice implementing the Strategy design pattern. You'll create a program that simulates a payment processing system where different payment methods (such as credit card, PayPal, and Bitcoin) can be selected at runtime.

Requirements:

1. Implement the Strategy design pattern.
2. Create three different payment strategies: CreditCardPayment, PayPalPayment, and BitcoinPayment.
3. Allow the payment method to be selected dynamically at runtime.
4. Demonstrate the use of the pattern with a sample client code.

Exercise Steps:

1. **Define the Strategy Interface:**
    - Create an interface `IPaymentStrategy` that declares a method `Pay(amount: decimal): void`.
2. **Implement Concrete Strategies:**
    - Implement the `IPaymentStrategy` interface in three classes: `CreditCardPayment`, `PayPalPayment`, and `BitcoinPayment`.
3. **Create the Context Class:**
    - Create a class `PaymentContext` that will use the `IPaymentStrategy`. It should have a method `SetPaymentStrategy(strategy: IPaymentStrategy): void` to set the payment strategy and a method `ProcessPayment(amount: decimal): void` to process the payment using the selected strategy.
4. **Client Code:**

- Write a client code to demonstrate the dynamic selection of payment strategies and processing payments.