**Курс:** Шаблоны проектирования приложений

**Тема:** Модуль 04 Принципы проектирования

*Баллы:*

Задача: Разработка системы управления заказами для интернет-магазина

**Описание:**

Вам необходимо разработать систему управления заказами для интернет-магазина, которая будет поддерживать добавление новых типов платежей и доставок, отправку уведомлений клиентам и возможность расчета стоимости заказа с учетом различных скидок. При этом следует применять принципы SOLID для создания гибкого и поддерживаемого кода.

**Требования:**

1. Single-Responsibility Principle (SRP): Каждый класс должен иметь одну четко определенную ответственность. Например, классы, отвечающие за обработку платежей, должны заниматься только платежами, а классы, отвечающие за доставку, только доставкой.

2. Open-Closed Principle (OCP): Система должна быть легко расширяема без необходимости изменения существующего кода. Например, для добавления нового типа оплаты или доставки не должно требоваться изменение существующих классов.

3. Liskov Substitution Principle (LSP): Дочерние классы должны корректно заменять родительские классы. Например, класс, реализующий новый способ доставки, должен корректно работать везде, где используется интерфейс доставки.

4. Interface Segregation Principle (ISP): Клиенты не должны зависеть от интерфейсов, которые они не используют. Например, интерфейсы должны быть разделены таким образом, чтобы классы реализовывали только те методы, которые им необходимы.

5. Dependency Inversion Principle (DIP): Модули верхнего уровня должны зависеть от абстракций, а не от конкретных реализаций. Например, система должна зависеть от абстракций оплаты и доставки, а не от конкретных реализаций этих процессов.

**Детали задачи:**

**1. Классы заказа:**

- Создайте класс `Order`, который будет содержать информацию о заказе: товары, их количество, цену, а также способ оплаты и доставки.

- Реализуйте методы для добавления товаров в заказ, расчета общей стоимости заказа с учетом скидок.

**2.Оплата:**

- Создайте интерфейс `IPayment`, который будет определять метод `ProcessPayment(double amount)`.

- Реализуйте несколько классов, которые будут выполнять конкретные типы оплаты: `CreditCardPayment`, `PayPalPayment`, `BankTransferPayment`.

- В системе должна быть возможность легко добавлять новые способы оплаты, используя принцип OCP.

**3. Доставка:**

- Создайте интерфейс `IDelivery`, который будет определять метод `DeliverOrder(Order order)`.

- Реализуйте несколько классов доставки: `CourierDelivery`, `PostDelivery`, `PickUpPointDelivery`.

- В системе должна быть возможность легко добавлять новые способы доставки, используя принцип OCP.

**4. Уведомления:**

- Создайте интерфейс `INotification`, который будет определять метод `SendNotification(string message)`.

- Реализуйте несколько классов уведомлений: `EmailNotification`, `SmsNotification`.

- Внедрите механизм уведомлений для клиентов при изменении статуса заказа.

**5. Расчет стоимости:**

- Создайте класс `DiscountCalculator`, который будет рассчитывать скидки на основе различных правил.

- Расчет скидок должен быть реализован таким образом, чтобы можно было легко добавлять новые типы скидок, не изменяя существующий код (применение OCP).

**6. Пример использования:**

- Создайте пример сценария использования, в котором создается заказ, добавляются товары, выбираются способы оплаты и доставки, рассчитывается стоимость, и отправляется уведомление клиенту.

**Условия:**

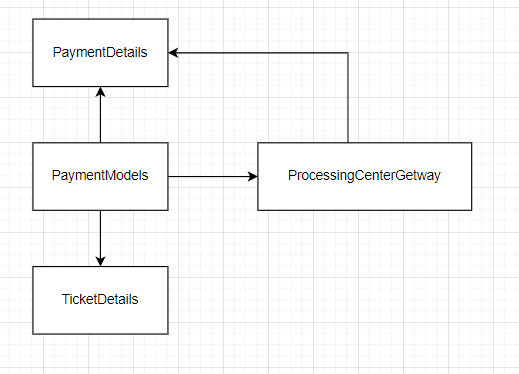
- Применяйте принципы SOLID везде, где это возможно.

- Используйте зависимости через интерфейсы, а не через конкретные классы.

- Обеспечьте возможность легкого добавления новых типов платежей, доставки и уведомлений без изменения существующего кода.

SRP

Пример покупки билетов



Придёте к такому виду

