**Exercise-4**

**Implementing the Adapter Pattern with Transaction Recording**

**1. Introduction**

This document outlines the implementation of the Adapter Pattern in a payment processing system, enhanced with transaction recording functionality. The system integrates multiple payment gateways (GPay and Stripe) and provides a unified interface for processing payments, viewing transaction history, and saving records.

**2. Problem Statement**

Develop a payment processing system that:

* Integrates multiple payment gateways with different interfaces
* Provides a unified interface for processing payments
* Records and manages transaction history
* Allows saving transaction records for future reference

**3. Solution Overview**

The solution uses the Adapter Pattern to integrate different payment gateways and adds a TransactionRecord class to manage transaction history. The main components are:

* PaymentProcessor (Target Interface)
* GPayGateway and StripeGateway (Adaptees)
* GPayAdapter and StripeAdapter (Adapters)
* TransactionRecord (for managing transaction history)
* AdapterPatternTest (Client code)

4.Implementation

Link:

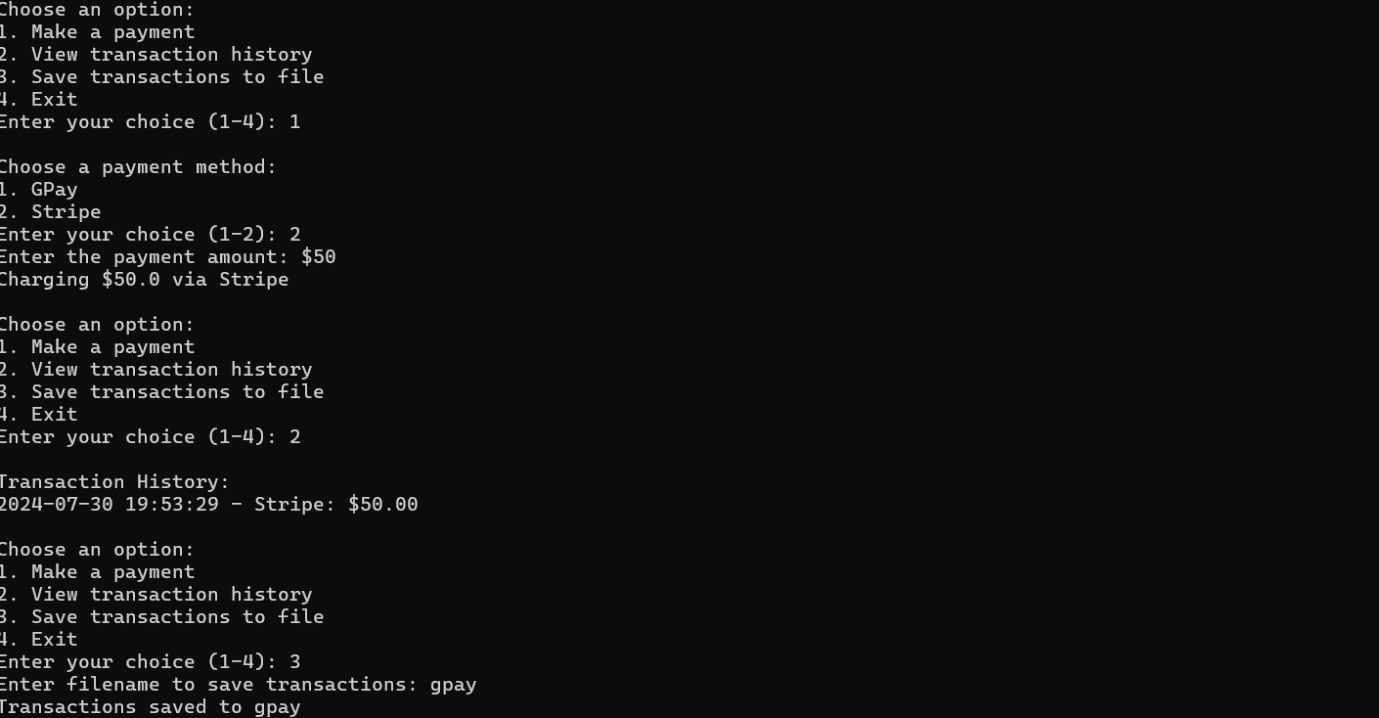
**5. Features and Functionality**

1. **Payment Processing**: Users can choose between GPay and Stripe payment methods and enter payment amounts.
2. **Transaction Recording**: Each successful payment is recorded with a timestamp, payment method, and amount.
3. **View Transaction History**: Users can view a list of all recorded transactions.
4. **Save Transactions**: Transactions can be saved to a file for future reference.

**6. Design Pattern Explanation**

The Adapter Pattern is used to allow the client code (AdapterPatternTest) to work with incompatible interfaces (GPay and Stripe) through a common PaymentProcessor interface. The adapters (GPayAdapter and StripeAdapter) wrap the specific gateway implementations and provide a unified interface for processing payments.

7. Output



**8. Conclusion**

This implementation demonstrates the effective use of the Adapter Pattern in creating a flexible payment processing system. The addition of transaction recording enhances the functionality, providing a more complete and practical example of how design patterns can be applied in real-world scenarios.

This document provides a comprehensive overview of the enhanced Adapter Pattern implementation, including the problem statement, solution overview, code structure, implementation details, running instructions, features, and an explanation of the design pattern used. It serves as a thorough guide for understanding and implementing the project.