**Лабораторна робота №4**

**Застосування патернів програмування**

1. Singleton

Клас Authenticator здається одинаком. Патерн забезпечує існування лише одного екземпляра аутентифікатора, що є важливим для правильного управління сесіями користувачів та аутентифікацією по всій системі.

2. Strategy

Payment Service реалізує шаблон стратегії. Різні методи оплати можуть бути інкапсульовані як конкретні стратегії, що дозволяє Payment Service без проблем переключатися між ними.

3. Factory Method

(Створення квитків)

Процес створення Ticket використовує фабричний метод. Клас User або інший керуючий клас може мати фабричний метод для створення екземплярів квитків на основі різних критеріїв (наприклад маршрут, тип вагону).

4. Observer

Notification Service використовує шаблон спостерігача. Він використовується в певних подіях, такі як зміни в розкладі або статусі квитків, і надсилає сповіщення користувачам.

5. Composite

Розклад та Поїзд

Відносини між Schedule і Train використовують паттерн composite. Розклад може складатися з кількох поїздів, що дозволяє управляти та маніпулювати складними розкладами як єдиним об’єктом.

6. State Machine

(Процес реєстрації та входу користувачів)

Діаграми станів для процесів реєстрації та входу користувачів представляють паттерн state machine. Процеси входу користувачів і операторів переходять через різні стани (введення даних, перевірка даних, відправка електронної пошти) на основі валідності введених даних.

**Посилання на GitHub**

https://github.com/shynkuraMaria/ProjectProgmaSystem.git