Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 6

з дисципліни «Теорія розробки ПЗ»

Тема: ШАБЛОНИ «Abstract Factory», «Factory Method», «Memento», «Observer», «Decorator»

Виконав:
студент групи ІА-14
Онуфрійчук Антон
Валерійович

Перевірив: Мягкий Михайло
Юрійович

Дата здачі _____

Захищено з балом _____

Київ, 2023

Тема: ШАБЛОНИ «Abstract Factory», «Factory Method», «Memento», «Observer», «Decorator»

Хід роботи

Завдання.

1. Ознайомитися з короткими теоретичними відомостями.

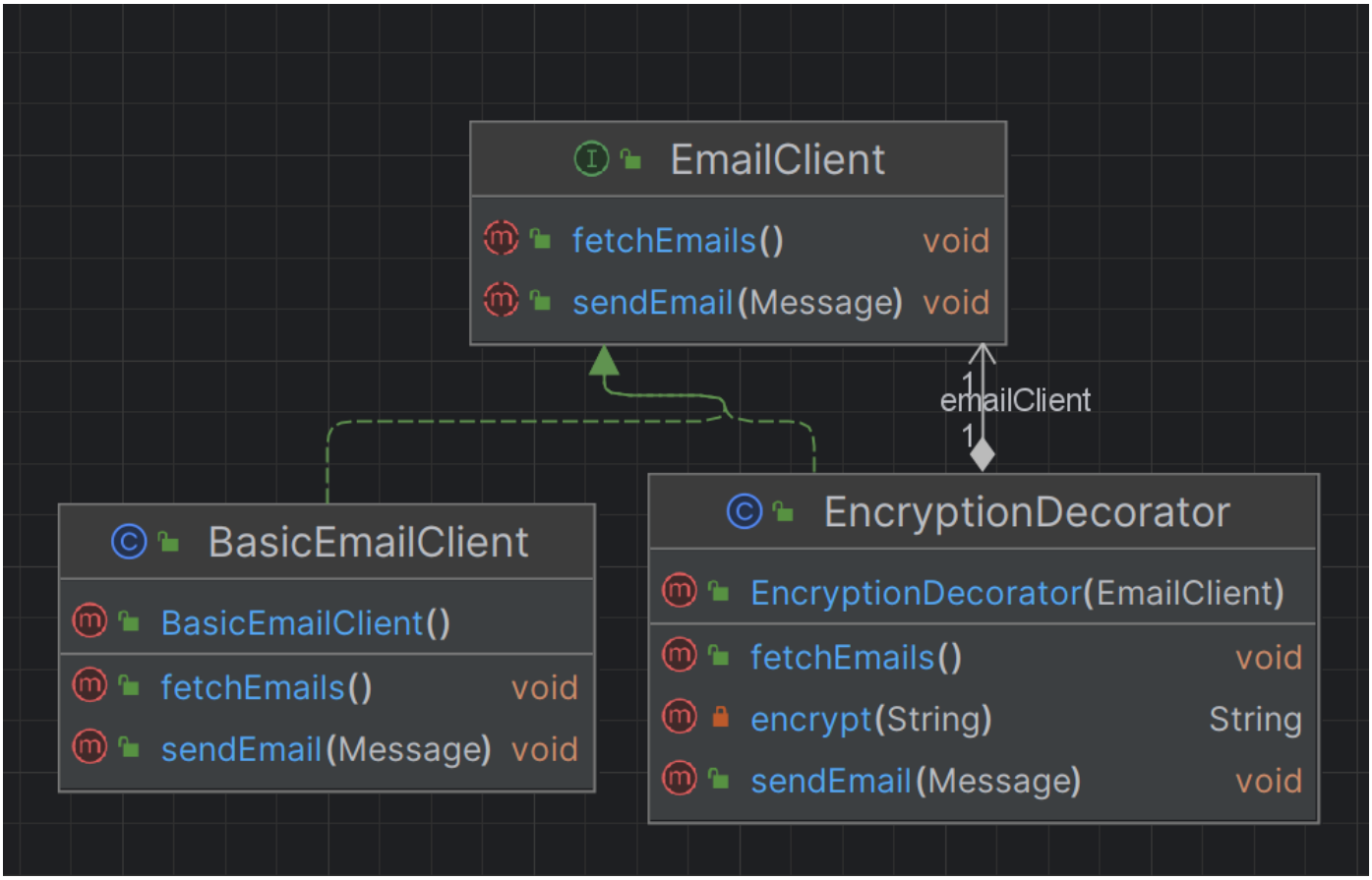
2. Проаналізуйте тему та намалюйте схему прецеденту, що відповідає обраній темі лабораторії.

..15 E-mail клієнт (singleton, builder, decorator, template method, interpreter, SOA)

Поштовий клієнт повинен нагадувати функціонал поштових програм Mozilla Thunderbird, The Bat і т.д. Він повинен сприймати і коректно обробляти

pop3/smtp/imap протоколи, мати функції автонастройки основних поштових провайдерів для України (gmail, ukr.net, i.ua), розділяти повідомлення на папки/категорії/важливість, зберігати чернетки незавершених повідомлень, прикріплювати і обробляти прикріплені файли.

Схема класів



Реалізація шаблону

```java
public class EncryptionDecorator implements EmailClient {
    3 usages
    private EmailClient emailClient;

    1 usage
    public EncryptionDecorator(EmailClient emailClient) {
        this.emailClient = emailClient;
    }

    2 usages
    @Override
    public void sendEmail(Message message) {
        // Логіка для шифрування повідомлення перед відправленням
        String encryptedBody = encrypt(message.getMessage());
        message.setMessage(encryptedBody);
        emailClient.sendEmail(message);
    }

    1 usage
    @Override
    public void fetchEmails() {
        // Логіка для отримання зашифрованих повідомлень
        emailClient.fetchEmails();
    }

    // Метод для шифрування повідомлення
    1 usage
    private String encrypt(String message) {
        return "Encrypted: " + message;
    }
}
```

Приклад використання

```java
public class Main {
    public static void main(String[] args) {
        Message build = Message.builder().withFrom("from").withTo("to").withMessage("message").build(
        // Використання декоратора для додавання шифрування повідомлень
        EmailClient basicEmailClient = new BasicEmailClient();
        EmailClient encryptedEmailClient = new EncryptionDecorator(basicEmailClient);

        encryptedEmailClient.sendEmail(build);

    }
}
```

Висновок: При виконані лабораторної роботи реалізував патерн Decorator.