



Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №4
Технології розроблення програмного забезпечення
Шаблони «singleton», «iterator», «proxy», «state», «strategy»
Варіант 30

Виконав
студент групи ІА-13
Якін С. О.

Перевірив:
Мягкий М.Ю.

Київ 2023

Тема: Шаблони «singleton», «iterator», «proxy», «state», «strategy»

Завдання:

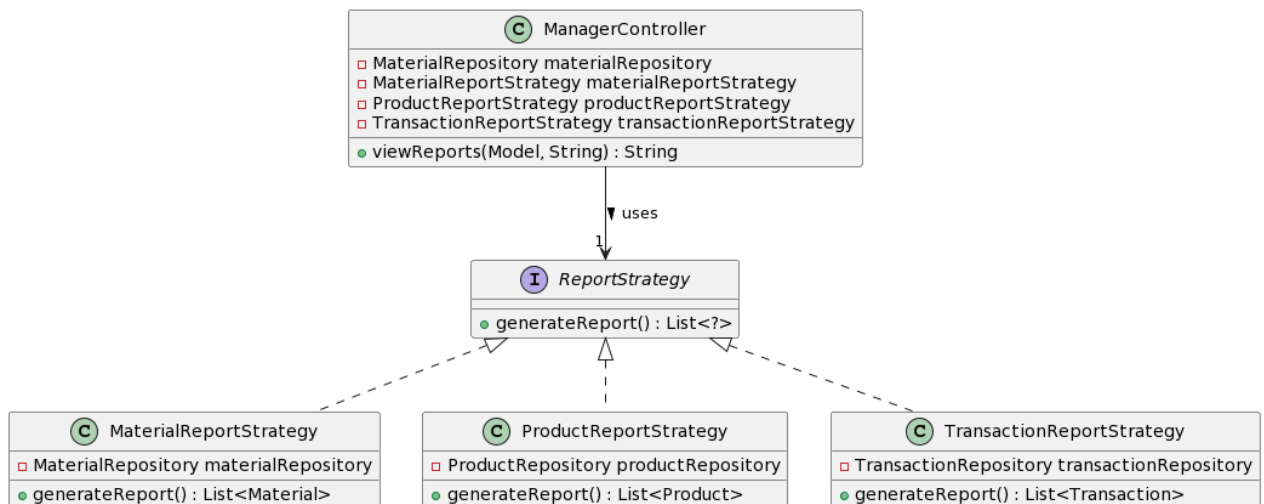
1. Ознайомитися з короткими теоретичними відомостями.
2. Реалізувати частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей.
3. Застосування одного з розглянутих шаблонів при реалізації програми.

Хід роботи:

Тема:

Система складського обліку виробництва.

Згідно мого завдання, було виконано шаблон проектування «Strategy». У якості реалізації, вирішив використовувати його для функціонального перегляду звітів для користувача (Начальника складу).



Як ми бачимо, в даному випадку, **ManagerController** є контекстом, який використовує стратегії. Він володіє посиланням на інтерфейс **ReportStrategy** і може використовувати його різні реалізації.

ReportStrategy є інтерфейсом стратегії, який визначає метод `generateReport`. Цей інтерфейс є основним для різних конкретних стратегій.

MaterialReportStrategy, **ProductReportStrategy**, і **TransactionReportStrategy** є конкретними реалізаціями інтерфейсу стратегії. Кожна з цих класів реалізує метод `generateReport` відповідно до своєї доменної логіки.

Шаблон проектування "Strategy" дозволяє змінювати поведінку об'єкта за допомогою делегування деяких операцій на різні об'єкти-стратегії. У нашому випадку, ManagerController може змінювати спосіб генерації звіту, обираючи відповідну стратегію (MaterialReportStrategy, ProductReportStrategy, або TransactionReportStrategy). Це дозволяє легко розширювати або змінювати логіку генерації звітів без необхідності змінювати код контролера.

Цей шаблон особливо корисний у ситуаціях, де потрібна гнучкість вибору між декількома варіантами поведінки або алгоритмами. Він також сприяє слабкому зв'язку між класами, оскільки контекст не залежить від конкретних реалізацій стратегій, а лише від їх інтерфейсів.

Код паттерну:

ReportStrategy:

```
package com.example.warehouse.services.strategies;

import java.util.List;

9 usages 3 implementations
public interface ReportStrategy {
    1 usage 3 implementations
    List<?> generateReport();
}
```

MaterialReportStrategy:

```
package com.example.warehouse.services.strategies.reports;
import com.example.warehouse.entity.Material;
import com.example.warehouse.repositories.MaterialRepository;
import com.example.warehouse.services.strategies.ReportStrategy;
import lombok.AllArgsConstructor;
import org.springframework.data.domain.Sort;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
@AllArgsConstructor
public class MaterialReportStrategy implements ReportStrategy {
    private final MaterialRepository materialRepository;

    1 usage
    @Override
    public List<Material> generateReport() {
        return materialRepository.findAll(Sort.by(...properties: "id").ascending());
    }
}
```

ProductReportStrategy:

```
package com.example.warehouse.services.strategies.reports;
import ...

@Service
@AllArgsConstructor
public class ProductReportStrategy implements ReportStrategy {
    private final ProductRepository productRepository;

    1 usage
    @Override
    public List<Product> generateReport() {

        return productRepository.findAll();
    }
}
```

TransactionReportStrategy:

```
package com.example.warehouse.services.strategies.reports;

import com.example.warehouse.entity.Transaction;
import com.example.warehouse.repositories.TransactionRepository;
import com.example.warehouse.services.strategies.ReportStrategy;
import lombok.AllArgsConstructor;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
@AllArgsConstructor
public class TransactionReportStrategy implements ReportStrategy {
    private final TransactionRepository transactionRepository;

    1 usage
    @Override
    public List<Transaction> generateReport() { return transactionRepository.findAll(); }
}
```

Приклад використання в контролері:

```

@Autowired
private MaterialReportStrategy materialReportStrategy;
@Autowired
private ProductReportStrategy productReportStrategy;
@Autowired
private TransactionReportStrategy transactionReportStrategy;

± serrb *
@GetMapping("/boss/reports")
private String viewReports(Model model,
                           @RequestParam(value = "reportType", required = false) String reportType) {
    if (reportType == null) {
        reportType = "transaction";
    }
    ReportStrategy reportStrategy;
    switch (reportType) {
        case "material":
            reportStrategy = materialReportStrategy;
            break;
        case "product":
            reportStrategy = productReportStrategy;
            break;
        case "transaction":
        default:
            reportStrategy = transactionReportStrategy;
            break;
    }

    model.addAttribute(attributeName: "report", reportStrategy.generateReport());
    model.addAttribute(attributeName: "reportType", reportType); // Подано
    return "/boss/reports";
}

```

Приклад на сторінці Користувача (Начальник складу):

Reports

localhost:8080/boss/reports?reportType=transaction

GmailYouTubeПеревестиVite AppНовый документ ~...Резюме - Google Д...All Bookmarks

HomeOlya (Logout)

Report

Manage Inventory

Create Recipe

Your Reports

Transaction Report

Material Report

Product Report

ID	Type	Material/Product	Amount	Date/Time	User
1	RECEIVE	Wooden boards	45.0	2023-12-25T07:46:50.783643	Sofia
3	RECEIVE	Aluminum profile	100.0	2023-12-25T08:09:54.964044	Sasha
5	RECEIVE	Wooden boards	25.0	2023-12-30T22:11:04.477726	Sofia
6	RECEIVE	Wooden boards	25.0	2023-12-30T22:13:21.631382	Sofia
7	RECEIVE	Wooden boards	25.0	2023-12-30T22:16:51.679970	Sofia
8	RECEIVE	Wooden boards	25.0	2023-12-30T22:20:49.344002	Sofia
11	RECEIVE	Self-tapping screws	500.0	2024-01-02T12:43:09.510161	Sofia
13	PRODUCTION	Table	2.0	2024-01-05T07:59:56.705680	Sofia
14	PRODUCTION	Door	5.0	2024-01-05T08:04:59.860628	Sofia
16	PRODUCTION	Door	2.0	2024-01-05T17:49:36.637586	Sofia
17	PRODUCTION	Closet	3.0	2024-01-05T22:08:40.961675	Sofia

Reports

localhost:8080/boss/reports?reportType=material

GmailYouTubeПеревестиVite AppНовый документ ~...Резюме - Google Д...All Bookmarks

HomeOlya (Logout)

Report

Manage Inventory

Create Recipe

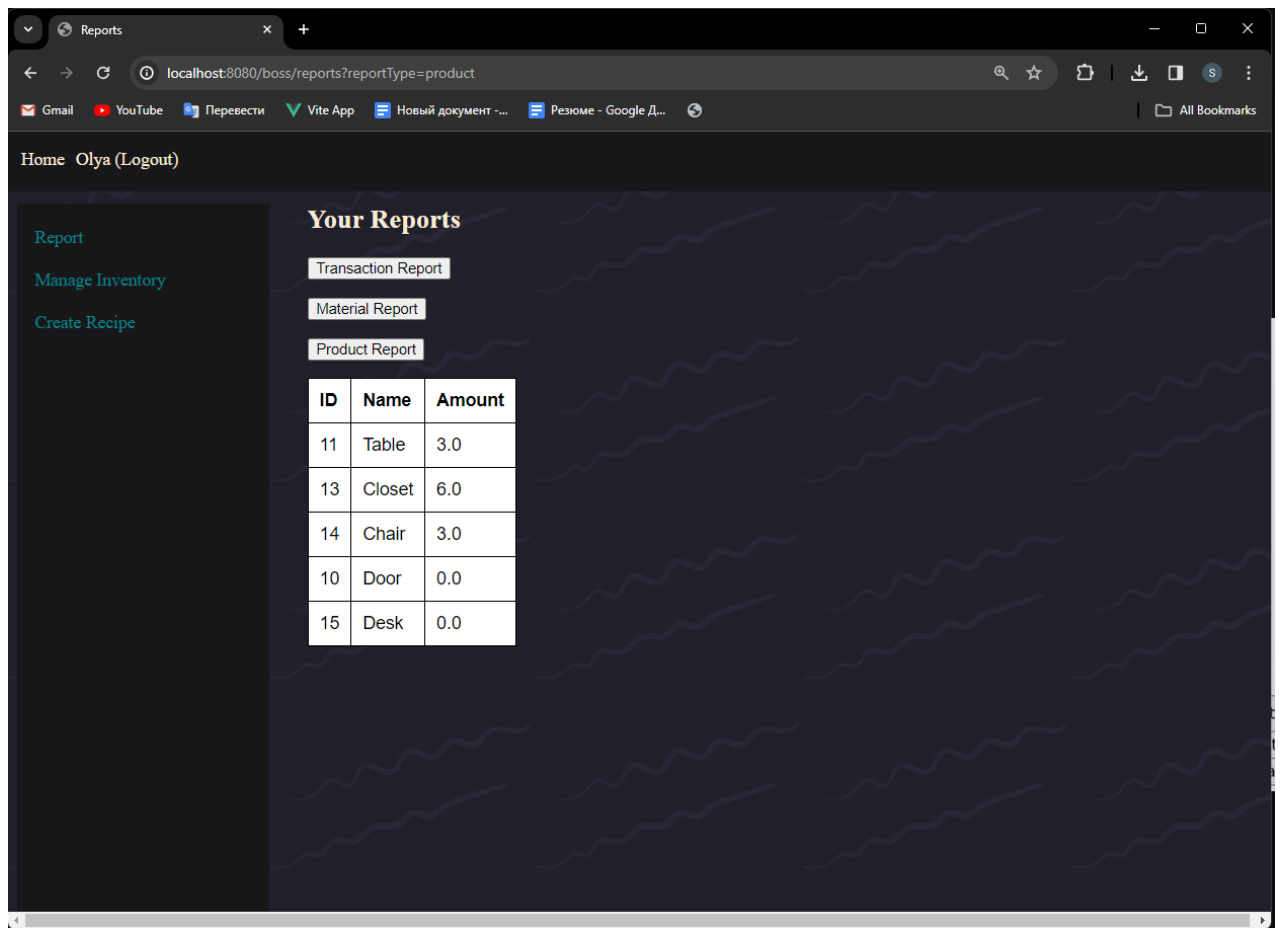
Your Reports

Transaction Report

Material Report

Product Report

ID	Name	Amount	Minimum Stock
1	Wooden boards	414.0	200.0
2	Aluminum profile	124.0	10.0
3	Self-tapping screws	3200.0	500.0
4	Iron strips	108.0	30.0



Висновок: на цій лабораторній роботі я познайомився з такими паттернами, як «singleton», «iterator», «проху», «state», «strategy», розібрався у принципі їх роботи та сфері використання. На практиці спроектував та реалізував паттерн «Strategy» у своєму проєкті.