Міністерство освіти та науки України

Чернівецький національний університет імені

Юрія Федьковича

Звіт лабораторної роботи № 4

з навчальної дисципліни

“Проєктування програмних систем”

На тему:  
“Застосування патернів програмування”

Квасницької Єлизавети Ігорівни

студентки факультету математики та інформатики

Спеціальності “Системний аналіз”

Курсу\_\_\_\_3\_\_\_\_\_\_групи\_\_\_\_\_\_\_\_\_\_\_\_\_\_307\_\_\_\_\_\_\_\_

Чернівці – 2025

Звіт лабораторної роботи № 4

Завдання: Аналізуючи діаграми класів (Лабораторна робота №3), обґрунтувати застосування підібраних для реалізації патернів.

1. Діаграма класів (Class Diagram)

Патерни:

*Facade*

Система має складну логіку (верифікація, документообіг, транзакції). Фасад спростить клієнтський інтерфейс, приховавши деталі реалізації.

*Observer*

Клієнти мають отримувати сповіщення про зміни статусу документів (наприклад, підписання, відхилення).

2. Діаграма станів (Statechart Diagram)

Патерни:

*State*

Процес нотарізації включає чіткі стани (`VerifyingIdentity`, `Signing`, `Rejected`). Патерн State дозволить легко додавати нові стани та змінювати логіку переходів.

*Command*

Переходи між станами (наприклад, "відхилити документ") можуть бути інкапсульовані в об’єкти-команди для відкладених або скасованих дій.

3. Діаграма діяльності (Activity Diagram)

Патерни:

*Chain of Responsibility*

Документи проходять послідовні кроки обробки (перевірка → підпис → архівація). Ланцюжок об’єктів дозволить гнучко налаштовувати послідовність.

*Template Method*

Стандартні нотаріальні дії (наприклад, верифікація) мають загальний алгоритм, але можуть відрізнятися деталями.

4. Діаграма компонентів (Component Diagram)

Патерни:

*Mediator*

Компоненти (клієнтський портал, архів, урядові системи) активні взаємодіють. Посередник запобігає прямому зв’язку між ними, зменшуючи залежності.

*Adapter*

Інтеграція з зовнішніми сервісами (банки, цифрові підписи) вимагає адаптації їхніх API до єдиного інтерфейсу системи.