**Міністерство освіти і науки України**

**Національний університет “Львівська політехніка”**

**Інститут комп’ютерних наук та інформаційних технологій**

**Кафедра програмного забезпечення**



**Звіт**

Про виконання лабораторної роботи №5

З дисципліни Вступ до Інженерії Програмного Забезпечення

**На тему:**

“Опис предметної області з використанням UML”

**Лектор:**

доц. каф. ПЗ

Левус Є.В.

**Виконав:**

ст. гр. ПЗ-18

Юшкевич А.І.

**Прийняв:**

асис. каф. ПЗ

Вишневський О.К.

« \_\_\_ » \_\_\_ 2023 р.

∑ = \_\_\_\_\_\_\_\_\_\_\_\_

Львів – 2023

**Тема:** опис предметної області з використанням UML.

**Meта:** навчитися створювати обʼєктну модель програмної системи.

**Теоретичні відомості**

17) Приклади для кожного типу сутностей UML:

1. Клас (Class):
2. Приклад: Клас "Студент" з атрибутами, такими як ім'я, номер студента, курс.
3. Об'єкт (Object):
4. Приклад: Об'єкт "Студент 1" зі значеннями атрибутів: ім'я = "Іван", номер студента = 12345, курс = 2.
5. Відношення (Relationship):
6. Приклад: Відношення "Міжнародний студент" між класом "Студент" і класом "Країна", що показує, що студент належить до певної країни.
7. Атрибут (Attribute):
8. Приклад: Атрибут "ім'я" у класі "Студент" для збереження імені студента.
9. Операція (Operation):
10. Приклад: Операція "додати" у класі "Калькулятор", яка виконує додавання двох чисел.
11. Пакет (Package):
12. Приклад: Пакет "Модуль студента" для групування класів, що відносяться до функціональності студента.
13. Компонент (Component):
14. Приклад: Компонент "База даних" для зберігання інформації про студентів.
15. Діаграма (Diagram):
16. Приклад: Діаграма класів, що відображає взаємозв'язки між класами, атрибутами та операціями.
17. Послідовність (Sequence):
18. Приклад: Діаграма послідовності, що показує послідовність викликів методів між об'єктами в процесі виконання програми.
19. Діаграма станів (State):
20. Приклад: Діаграма станів для процесу реєстрації студента, що відображає різні стани, такі як "новий", "зареєстрований", “відрахований".

9) Опис кожного типу сутностей UML:

1. Клас (Class): Класи використовуються для представлення основних будівельних блоків системи. Вони описують структуру, атрибути та поведінку об'єктів.
2. Об'єкт (Object): Об'єкти є конкретними екземплярами класів і мають свої власні значення атрибутів. Вони представляють конкретні об'єкти або екземпляри класів.
3. Відношення (Relationship): Відношення визначають зв'язки між класами або об'єктами. Вони допомагають моделювати залежності, асоціації, агрегації, композиції, спадковість та інші типи зв'язків.
4. Атрибут (Attribute): Атрибути представляють характеристики класів або об'єктів. Вони описують дані, які зберігаються в об'єктах та використовуються для їх ідентифікації, опису та взаємодії з іншими об'єктами.
5. Операція (Operation): Операції описують поведінку класів або об'єктів. Вони представляють методи або функції, які можуть бути викликані для виконання певних дій над об'єктами.
6. Пакет (Package): Пакети дозволяють організувати модель UML на логічні групи. Вони використовуються для групування класів, діаграм та інших елементів моделі.
7. Компонент (Component): Компоненти представляють фізичні або логічні частини системи, що можуть бути розглянуті як самостійні модулі. Вони описують компоненти програмного забезпечення, бібліотеки, модулі, підсистеми тощо.
8. Діаграма (Diagram): Діаграми використовуються для графічного відображення моделей UML. Вони представляють візуальні відображення класів, об'єктів, відношень та інших елементів системи.
9. Послідовність (Sequence): Діаграми послідовності використовуються для моделювання послідовності взаємодії об'єктів в процесі виконання сценаріїв або функціональності системи.
10. Діаграма станів (State): Діаграми станів дозволяють моделювати різні стани, в яких можуть перебувати об'єкти або система в цілому. Вони використовуються для опису поведінки системи з точки зору її станів та переходів між ними.
11. Ці типи сутностей UML допомагають розбити систему на складові частини, описати їх взаємозв'язки та поведінку, та відображати модель графічно за допомогою діаграм.

**Завдання**

Згідно індивідуального варіанту провести аналіз предметної області. Для

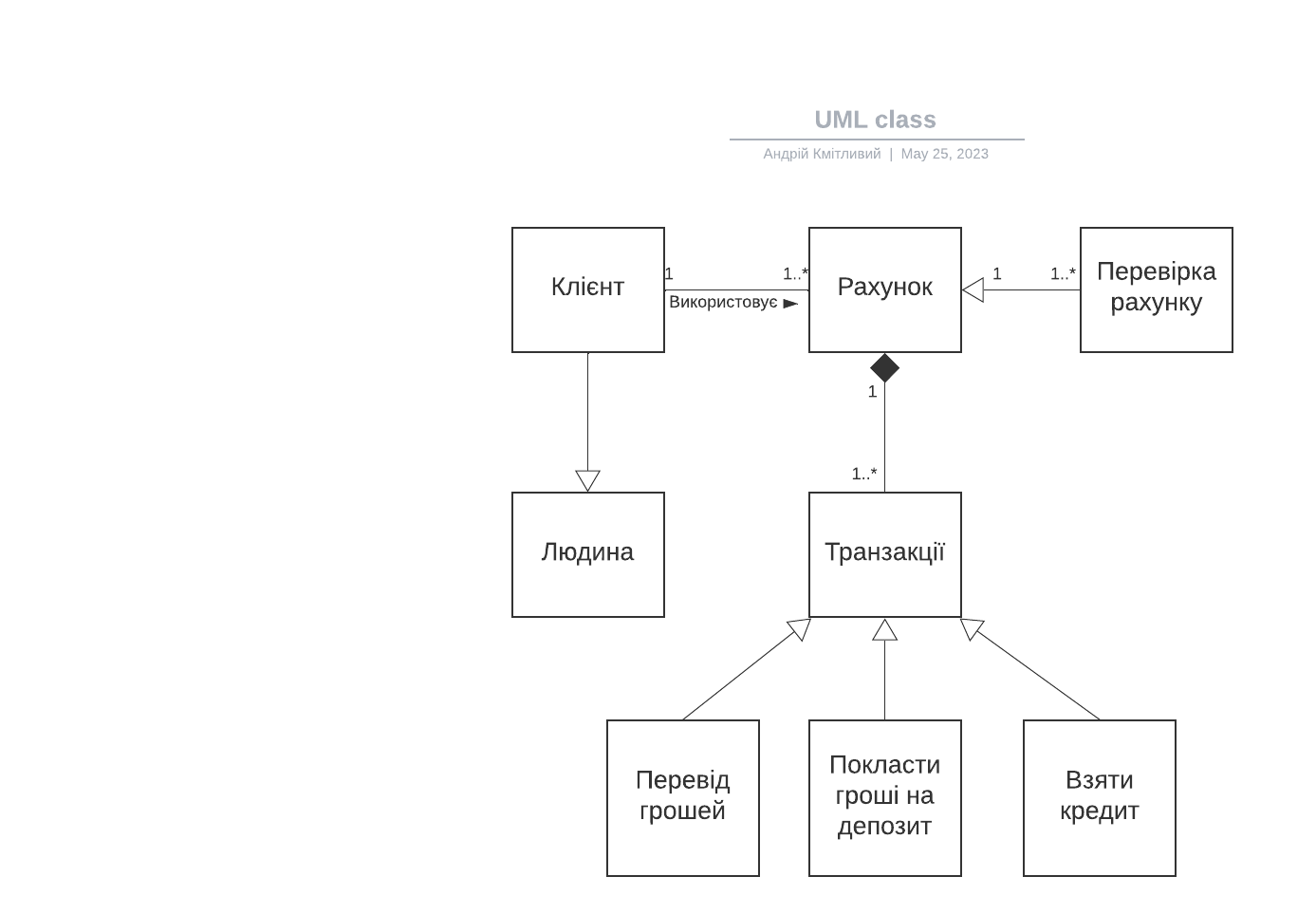
виконання завдання:

* ﻿﻿﻿Скласти словник предметної області.
* ﻿﻿﻿Побудувати UML-діаграму класів на концептуальному рівні засобами програми Visio. Зобразити коментарі на схемі. Вказати відношення між сутностями (узагальнення, звичайна асоціація, агрегація, композиція, залежність) із обовʼязковим зазначенням їх характеристик (кратність, назва асоціації і т.п.).
* ﻿﻿﻿Побудувати UML-діаграму конкретних класів на рівні реалізації засобами програми Visio. Чітко вказати усі поля та методи класів з відповідними модифікаторами доступу, а також усі необхідні відношення між класами.
* ﻿﻿﻿Оформити звіт.

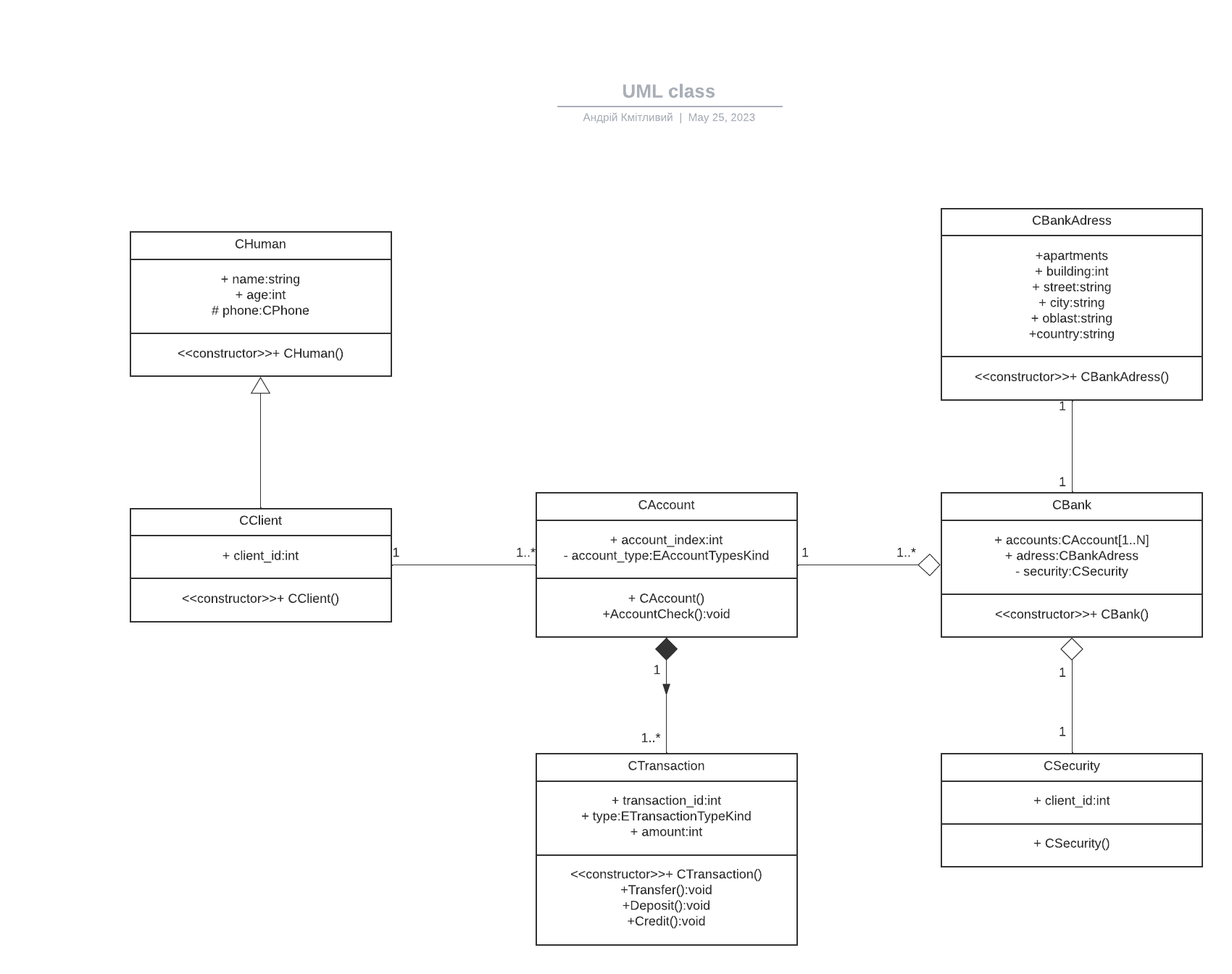
**Словник предметної області**

1. **CBankAdress** — адреса банку; має властивості apartments, building, street, city, oblast, country. Звʼязаний зі сутністю **CBank**.
2. **CSecurity** — безпекови блок; має додаткові властивості client\_id. Звʼязаний зі сутністю **CBank**.
3. **CBank** — власне, банк; має властивості accounts, adress, security. Звʼязаний зі сутностями **CAccount**, **CBankAdress**, **CSecurity** .
4. **CTransaction** — клас транзакцій; має властивості transaction\_id, type, amount; Має операції переказу коштів, дає змогу покласти гроші на депозит та взяти кредит; зв’язана з сутністю **CAccount**.
5. **CAccount** — банківський акаунт, має властивості account\_index, account\_type; зв’язаний із сутностями **CClient**, **CTransaction** та **CBank**.
6. **Human —** людина (базовий клас); має властивості name, age, phone.
7. **Client** —користувач (похідний від класу **Human**); має додаткову властивість client\_id.

**UML-діаграма класів на концептуальному рівні**



**UML-діаграма класів на рівні реалізації**



**Висновок:** На цій лабораторній роботі я навчився створювати обʼєктну модель програмної системи. Зокрема розробляти діаграми класів на концептуальному рівні та рівні реалізації.