



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 №2

із дисципліни *«Технології розроблення програмного забезпечення»*

**Тема: «ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ. СЦЕНАРІЇ
ВАРІАНТІВ ВИКОРИСТАННЯ.
ДІАГРАМИ UML. ДІАГРАМИ
КЛАСІВ. КОНЦЕПТУАЛЬНА МОДЕЛЬ
СИСТЕМИ»**

Виконала:
Студентка групи ІА-24
Сіденко Д.Д.

Перевірив:
Мягкий М.Ю.

Тема лабораторних робіт:

Особиста бухгалтерія (state, prototype, decorator, bridge, flyweight, SOA)

Програма повинна бути наочним засобом для ведення особистих фінансів: витрат і прибутку; з можливістю встановлення періодичних витрат / прибутку (зарплата і орендна плата); введення сканованих чеків з відповідними статтями витрат; побудова статистики; експорт/імпорт в Excel, реляційні джерела даних; різні рахунки; ведення єдиного фонду на всі рахунки (всією сім'єю) - на особливі потреби (ремонт, автомобіль, відпустка); можливість введення вкладів / кредитів для контролю банківських рахунків (звірка нарахованих відсотків з необхідними і т.д.).

Завдання:

1. Ознайомитися з короткими теоретичними відомостями.
2. Проаналізуйте тему та намалюйте схему прецеденту, що відповідає обраній темі лабораторної.
3. Намалюйте діаграму класів для реалізованої частини системи.
4. Виберіть 3 прецеденти і напишіть на їх основі прецеденти.
5. Розробити основні класи і структуру системи баз даних.
6. Класи даних повинні реалізувати шаблон Репозиторію для взаємодії з базою даних.
7. Підготувати звіт про хід виконання лабораторних робіт. Звіт, що подається повинен містити: діаграму прецедентів, діаграму класів системи, вихідні коди класів системи, а також зображення структури бази даних.

Структура файлу:

Крок 1. Аналіз теми та шаблонів проектування.

Крок 2. Створення діаграми прецедентів.

Крок 3. Створення діаграми класів для частини майбутньої системи.

Крок 4. Опис 3 прецедентів.

Крок 5. Представлення схеми таблиць бази даних.

Хід роботи:**Крок 1. Аналіз теми та шаблонів проектування.**

Перед початком роботи над лабораторною роботою, варто проаналізувати надані шаблони проектування та визначитись з варіантами їх використання.

Шаблон	Суть	Варіант застосування
State	Шаблон дозволяє об'єкту змінювати свою поведінку в залежності від свого стану, забезпечуючи динамічну зміну класів.	Стани обробки періодичних платежів ("активні", "призупинені", "скасовані")
Prototype	Дозволяє створювати нові об'єкти копіюванням існуючих. Використовується, коли створення об'єкта є складним або ресурсозатратним.	Створення нових транзакцій на основі існуючих (шаблони регулярних витрат/прибутків)
Decorator	Дозволяє динамічно додавати функціонал до об'єкта без зміни його структури, використовуючи обгортки.	Обгортки для рахунків, які надають інформацію про залишок, кредитний ліміт або спеціальні умови.
Bridge	Дозволяє розділяти абстракцію і реалізацію так, щоб вони могли	Створення гнучкої структури для експорту даних в різні формати (Excel, CSV) чи

	змінюватися незалежно один від одного.	взаємодії з різними джерелами даних (реляційні бази, локальні файли тощо)
Flyweight	Використовується для зменшення споживання пам'яті шляхом спільного використання однакових об'єктів.	Управління великою кількістю однотипних об'єктів, наприклад, транзакцій або витрат, які можуть мати спільні атрибути (категорії, дати).
SOA	Архітектурний стиль, який розбиває програму на незалежні сервіси, що взаємодіють через чітко визначені інтерфейси.	Програма може бути побудована як набір незалежних сервісів для управління фінансами

Крок 2. Створення діаграми прецедентів.

Для створення початкової концептуальної моделі представлення, я взяла за основні функції, що вже були вказані в темі.

Варіанти використання для програми "Особиста бухгалтерія" згідно з діаграмою прецедентів:

1) Додавання фінансової операції:

- Користувач може додати нову витрату до системи, вказавши категорію (наприклад, їжа, транспорт), дату, суму та додаткові коментарі. Ця інформація зберігається в базі даних для подальшого аналізу.

- Користувач може додати новий дохід, вказавши джерело (зарплата, бонуси) та суму. Система автоматично оновлює загальний баланс користувача.

2) Управління періодичними операціями:

- Користувач може створити шаблон для регулярних витрат (наприклад, оренда, комунальні платежі) або доходів (зарплата). Система автоматично додає ці суми до відповідних категорій у визначений період.

3) Введення сканованих чеків:

- Користувач може завантажити зображення чеків, пов'язаних з витратами. Система може зберігати ці зображення і асоціювати їх з відповідними транзакціями, що спростить управління документами.

4) Генерація фінансової статистики:

- Користувач може переглядати фінансові звіти та статистику, включаючи графіки витрат та доходів. Це дозволяє отримати візуальне представлення фінансового стану.

5) Експорт/імпорт даних в Excel:

- Користувач може експортувати свої фінансові дані у форматі Excel для подальшого аналізу або обробки. Це може включати звіти за певний період або всю історію фінансових операцій.

6) Управління рахунками:

- Користувач може додавати, редагувати або видаляти рахунки (дебетові, кредитні). Система повинна відображати баланс кожного рахунку та загальний баланс користувача.

7) Управління фондами:

- Користувач може створювати та управляти фондами (наприклад, фонд для ремонту, автомобіля, відпустки). Система дозволяє користувачу визначати мету фонду, вносити кошти та слідкувати за прогресом накопичення. Користувач може також переглядати, скільки грошей залишилося в фонді, і скільки ще потрібно зібрати для досягнення мети.

8) Керування кредитами/вкладом:

- Користувач може вводити інформацію про свої кредити та вклади, вказуючи умови (відсотки, терміни). Система може надавати звіти про стан кредитів та можливість порівняння з доходами.

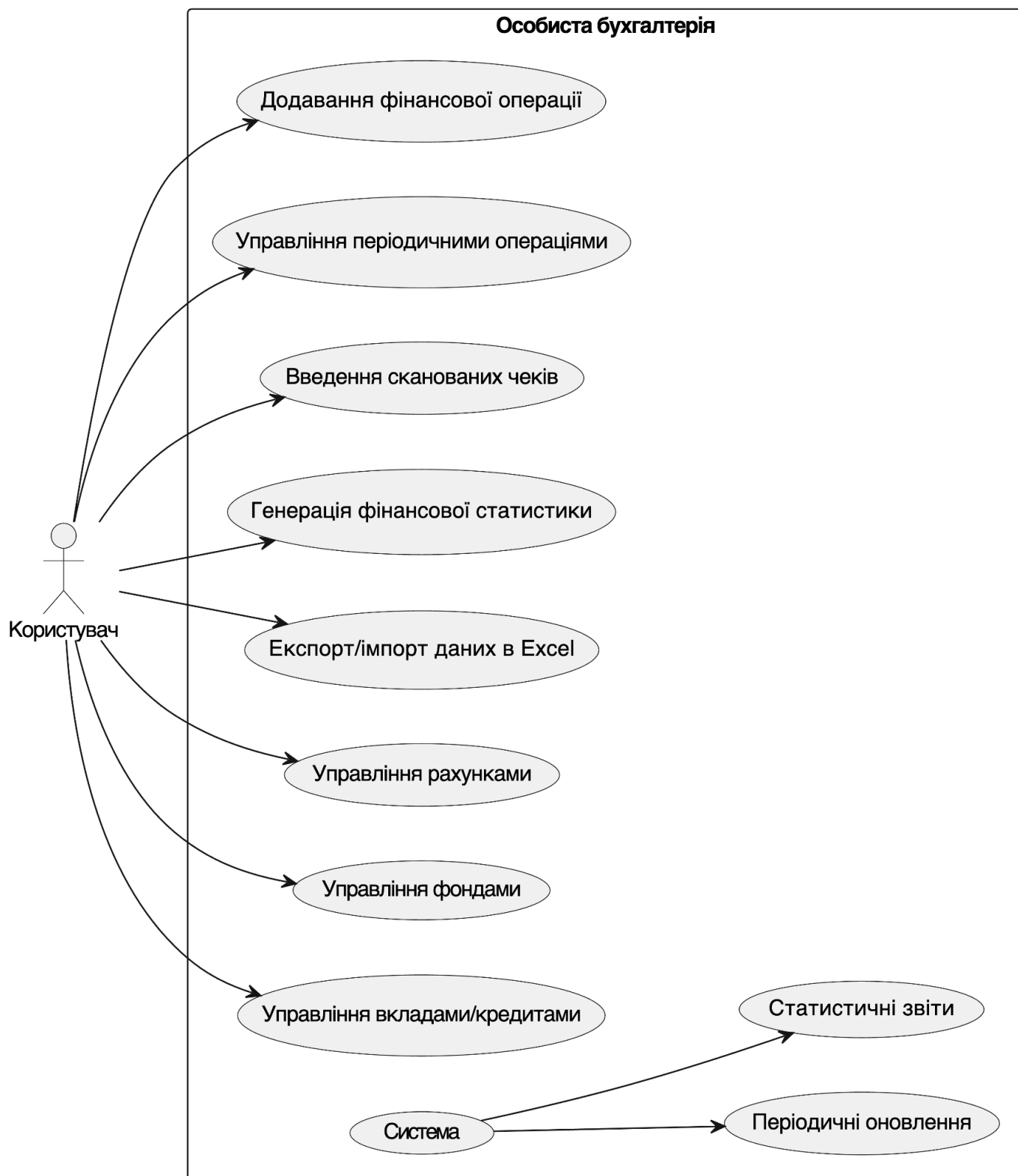


Рис. 1 - Діаграма прецедентів

Крок 3. Створення діаграми класів для частини майбутньої системи.

Опис класів:

Користувач: Основний клас, що представляє користувача програми. Він містить інформацію про ім'я, прізвище та email, а також методи для додавання витрат і прибутків, управління рахунками та перегляду статистики.

Витрата: Клас, що представляє витрату, включаючи категорію, суму, дату та коментар.

Прибуток: Клас, що представляє прибуток, включаючи джерело, суму та дату.

Рахунок: Клас, що представляє рахунок (дебетовий або кредитний) з номером рахунку, типом та балансом. Він також має метод для додавання транзакцій.

Транзакція: Клас, що представляє фінансову транзакцію з сумою, датою та типом (витрата або прибуток).

Фонд: Клас, що представляє фонд для накопичення коштів, включаючи назву, мету та суму. Має методи для внесення коштів і перегляду статусу.

Статистика: Клас для отримання звітів та графіків витрат.

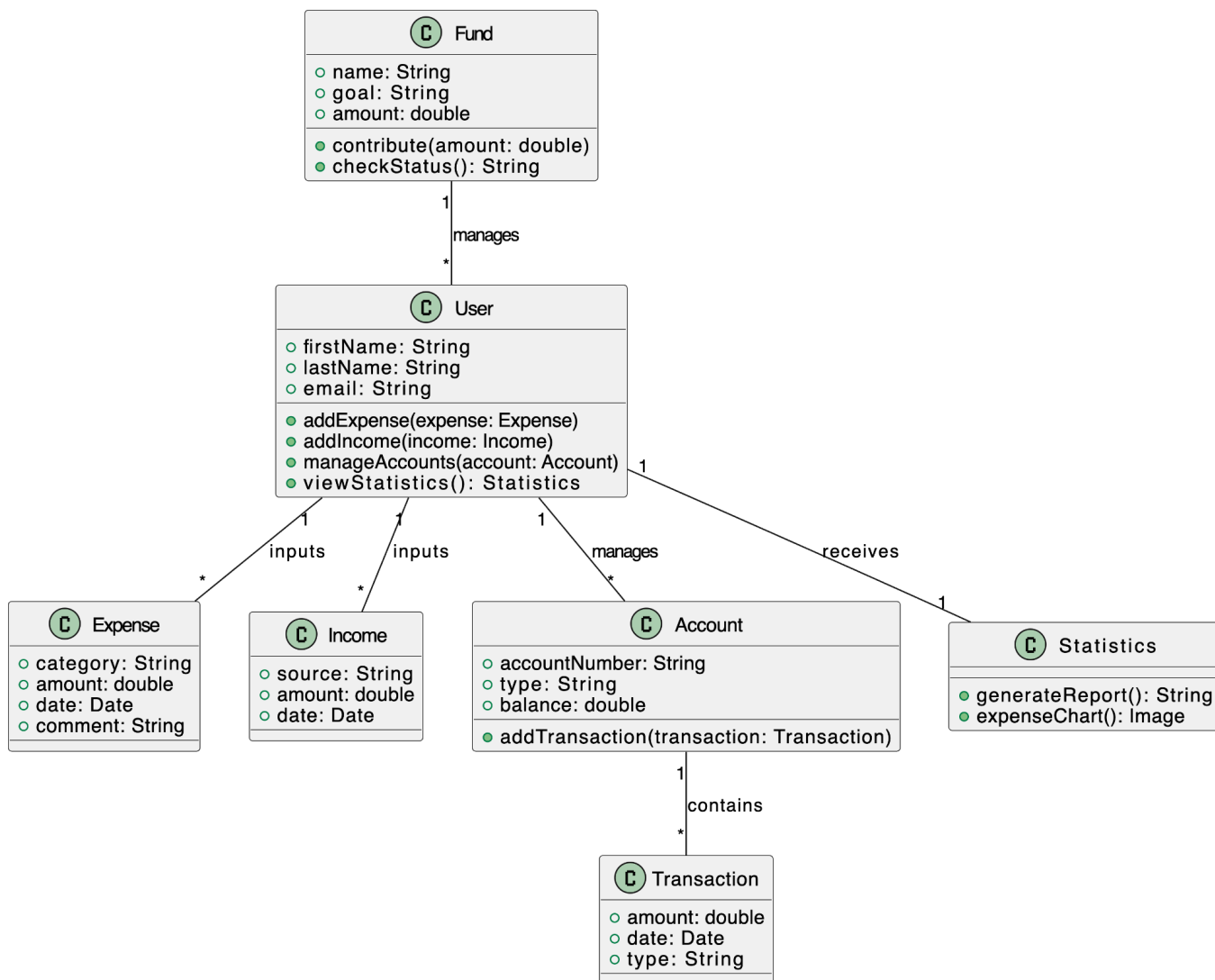


Рис. 2 - Діаграма класів

Крок 4. Опис 3 прецедентів.

1. Прецедент: Додати витрату

Назва: Add Expense

Актори: User

Опис: Користувач може додати нову витрату до системи, заповнивши форму з необхідними даними.

Передумови: Користувач повинен бути автентифікований у системі.

Основний потік:

1. Користувач вибирає опцію "Додати витрату".

2. Система відображає форму для введення інформації про витрату.
3. Користувач заповнює поля: категорія, сума, дата, коментар.
4. Користувач натискає кнопку "Зберегти".
5. Система зберігає витрату у базі даних.
6. Система підтверджує успішне додавання витрати.

Виключення:

4-а. Якщо сума не є числом, система відображає повідомлення про помилку і запитує ввести правильну суму.

2. Прецедент: Переглянути статистику

Назва: View Statistics

Актори: User

Опис: Користувач може переглядати фінансову статистику за обраний період.

Передумови: Користувач повинен бути автентифікований у системі.

Основний потік:

1. Користувач вибирає опцію "Переглянути статистику".
2. Система відображає календар для вибору періоду.
3. Користувач вибирає початкову та кінцеву дати.
4. Користувач натискає кнопку "Показати статистику".
5. Система обробляє запит і формує звіт.
6. Система відображає статистику у формі графіків і таблиць.

Виключення:

3-а. Якщо користувач вибирає неправильні дати (кінцева дата менша за початкову), система відображає повідомлення про помилку.

3. Прецедент: Управління фондами

Назва: Manage Funds

Актори: User

Опис: Користувач може створювати, редагувати або видаляти фонди для

накопичення коштів на конкретні цілі.

Передумови: Користувач повинен бути автентифікований у системі.

Основний потік:

1. Користувач вибирає опцію "Управління фондами".
2. Система відображає список існуючих фондів.
3. Користувач може вибрати один із наступних варіантів:
 - 3-а. Додати новий фонд.
 - 4-а. Система відображає форму для введення деталей фонду (назва, мета, сума).
 - 3-б. Редагувати існуючий фонд.
 - 3-с. Видалити фонд.
4. Користувач заповнює поля і натискає "Зберегти".
5. Система зберігає новий фонд у базі даних і підтверджує успішне додавання.

Виключення:

- 5-а. Якщо одне з обов'язкових полів порожнє, система відображає повідомлення про помилку і запитує ввести дані.

Крок 5. Представлення схеми таблиць бази даних.

Опис зв'язків між таблицями.

Users до Expenses: Один до Багатьох (Один користувач може мати багато витрат)

Users до Incomes: Один до Багатьох (Один користувач може мати багато прибутків)

Users до Accounts: Один до Багатьох (Один користувач може мати багато рахунків)

Accounts до Transactions: Один до Багатьох (Один рахунок може мати багато транзакцій)

Users до Funds: Один до Багатьох (Один користувач може мати багато фондів)

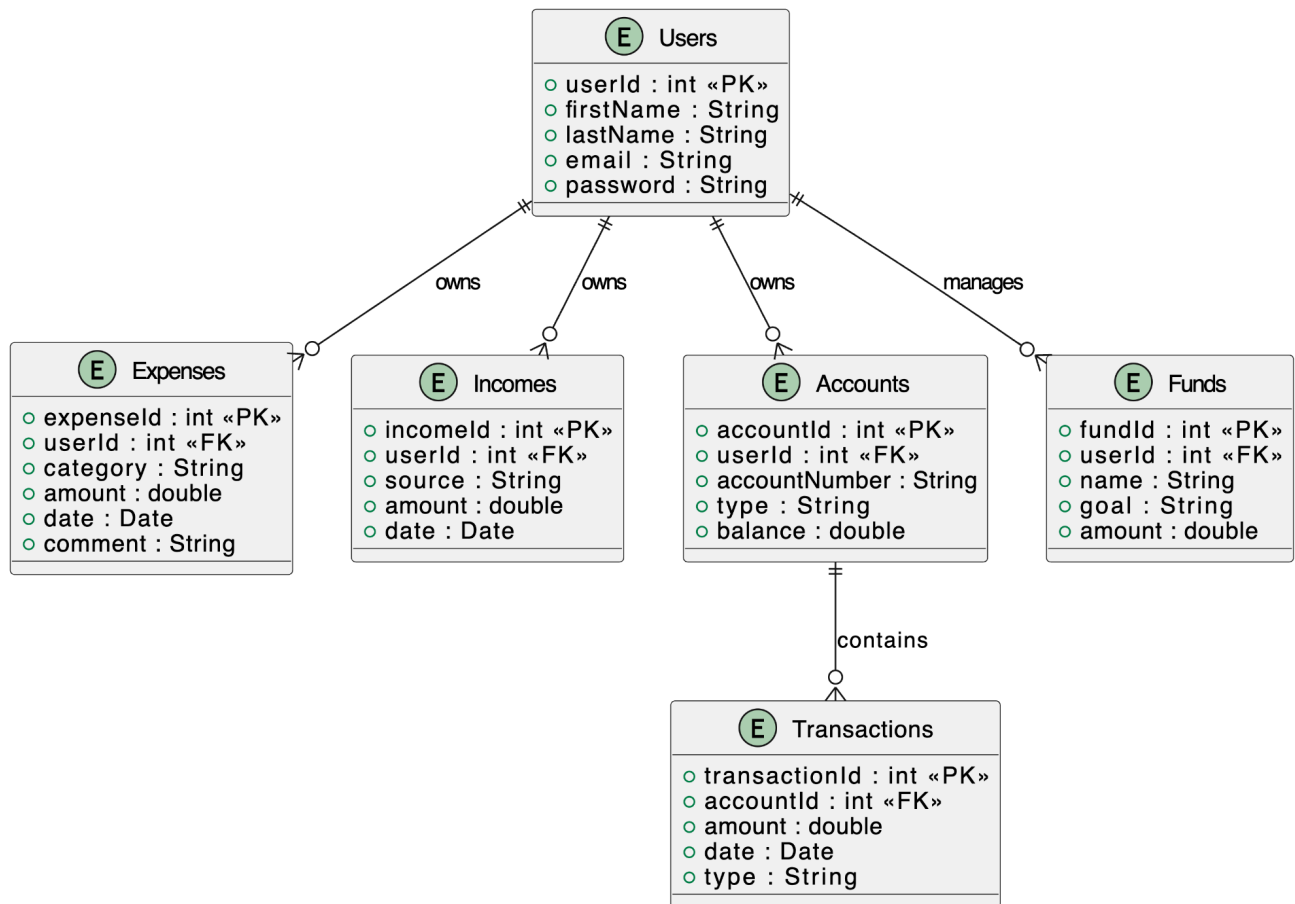


Рис. 3 - Схема таблиц бази даних

Висновок: у даній лабораторній роботі я проаналізувала тему та на основі даних створила діаграму прецедентів та діаграму класів. Також під час виконання даної лабораторної роботи я спроектувала схему бази даних та описала зв'язки між таблицями.