

**Міністерство освіти і науки України  
Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра обчислювальної техніки**

## **Лабораторна робота №3**

з дисципліни

«Сучасні технології розробки WEB-застосувань на платформі Microsoft.NET»

Виконала:

студентка групи ІМ-11

Бащак Ярина Володимирівна

Перевірів:

доц. Крамар Ю. М.

Київ 2023

## Завдання.

### Практична частина:

1. З дотриманням вимог REST-у спроектувати веб-API для обраної(згідно варіанту) доменної області, використовуючи методологію C4 для створення діаграми архітектури системи.
2. Створити ER-діаграму для DAL (Data Access Layer), яка відображатиме структуру бази даних веб-API.
3. Оформити спроектоване рішення у вигляді звіту до лабораторної роботи.

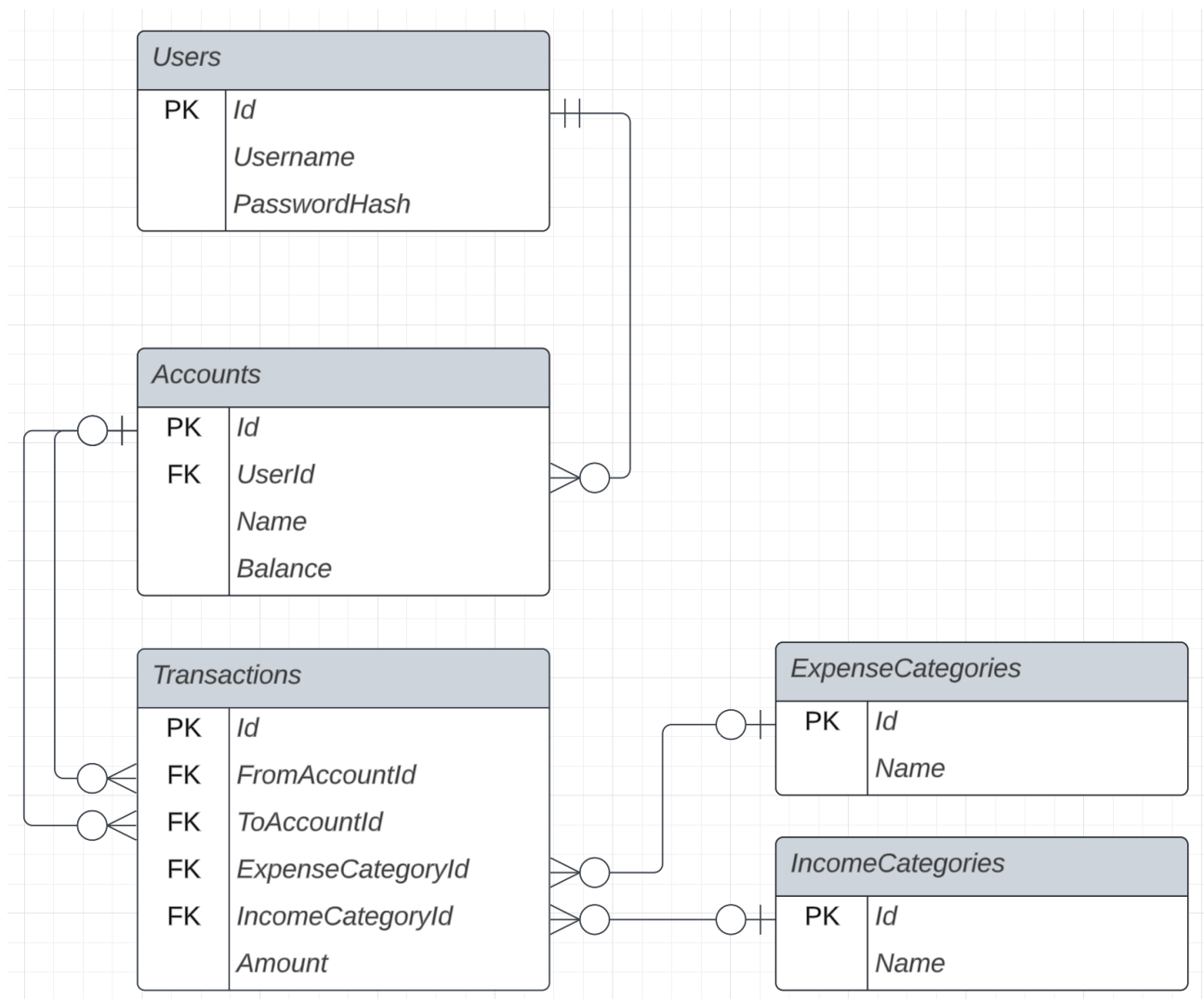
### Документація:

1. Підготувати документацію(звіт до ЛР), яка включатиме опис веб-API, а також структуру бази даних з урахуванням ER-діаграми.

## Мій варіант:

Варіант	Предметна галузь	Функціональні вимоги
4	Гаманець. Керування власним бюджетом та фінансами	<p>1. Власний бюджет складається з декількох рахунків, які поповнюються за заданими статтями прибутку.</p> <p>2. Гроші цих рахунків можуть бути переведені з одного на інший, можуть витратитись за заданими статтями витрат.</p> <p>3. Підсумовуючи витрати та прибутки, можливо отримати інформацію, скільки було витрачено/отримано загалом/за певною статтею по заданому рахунку.</p> <p><b>Функціональні вимоги:</b></p> <p>1. Ведення власного бюджету;</p> <p>2. Отримання звітної інформації по рахунках.</p>

## ER-діаграма.



Таблиця: **Users** – користувачі.

Поля:

- **Id (PK):** INTEGER – ідентифікатор користувача
- **Username:** VARCHAR(50) – ім'я користувача
- **PasswordHash:** VARCHAR(255) – хеш пароля

Таблиця: **Accounts** – рахунки.

Поля:

- **Id (PK):** INTEGER – ідентифікатор рахунку
- **UserId (FK):** INTEGER – ідентифікатор користувача
- **Name:** VARCHAR(50) – назва рахунку
- **Balance:** INTEGER – баланс рахунку

Таблиця: **Transactions** – транзакції (витрати/доходи по категоріях, перекази між рахунками).

Поля:

- Id (PK): INTEGER – ідентифікатор транзакції
- FromAccountId (FK): INTEGER – ідентифікатор рахунку з якого переказують або null, якщо це прибуток не зі свого іншого рахунку
- ToAccountId (FK): INTEGER – ідентифікатор рахунку якому переказують або null, якщо це витрата не на свій інший рахунок
- ExpenseCategoryId (FK): INTEGER – ідентифікатор категорії витрати (має бути вказаний, якщо FromAccountId не null)
- IncomeCategoryId (FK): INTEGER – ідентифікатор категорії прибутку (має бути вказаний, якщо ToAccountId не null)
- Amount: INTEGER – сума транзакції

Таблиця: **ExpenseCatagories** – категорії витрат.

Поля:

- Id (PK): INTEGER – ідентифікатор категорії
- Name: VARCHAR(50) – назва категорії

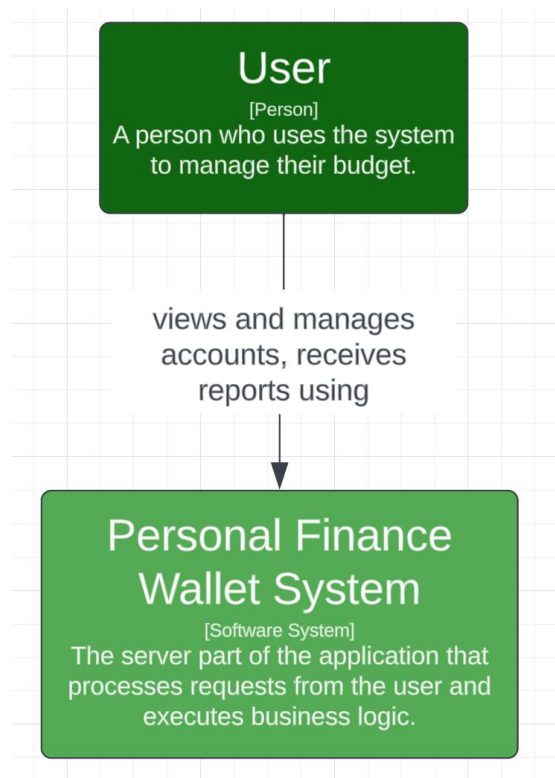
Таблиця: **IncomeCatagories** – категорії прибутку.

Поля:

- Id (PK): INTEGER – ідентифікатор категорії
- Name: VARCHAR(50) – назва категорії

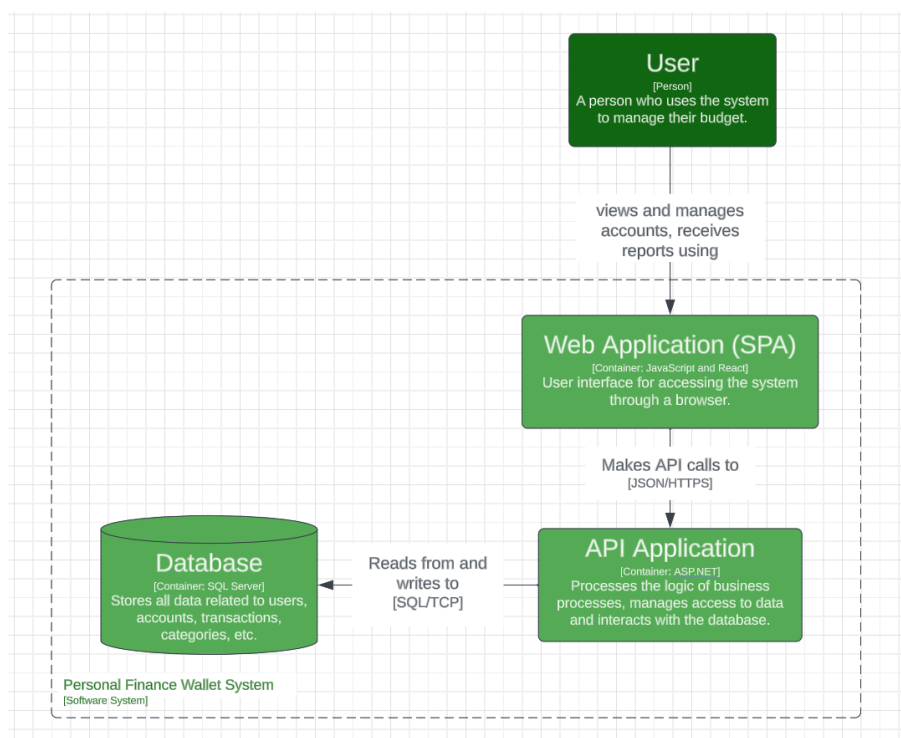
## Діаграма C4.

### Level 1 – System Context Diagram.



Тип користувача тільки один – особа, яка хоче керувати особистим бюджетом. Він взаємодіє з системою і може робити перекази між своїми рахунками, записувати доходи та витрати по категоріях, отримувати звітну інформацію по рахунках.

### Level 2 – Container Diagram.



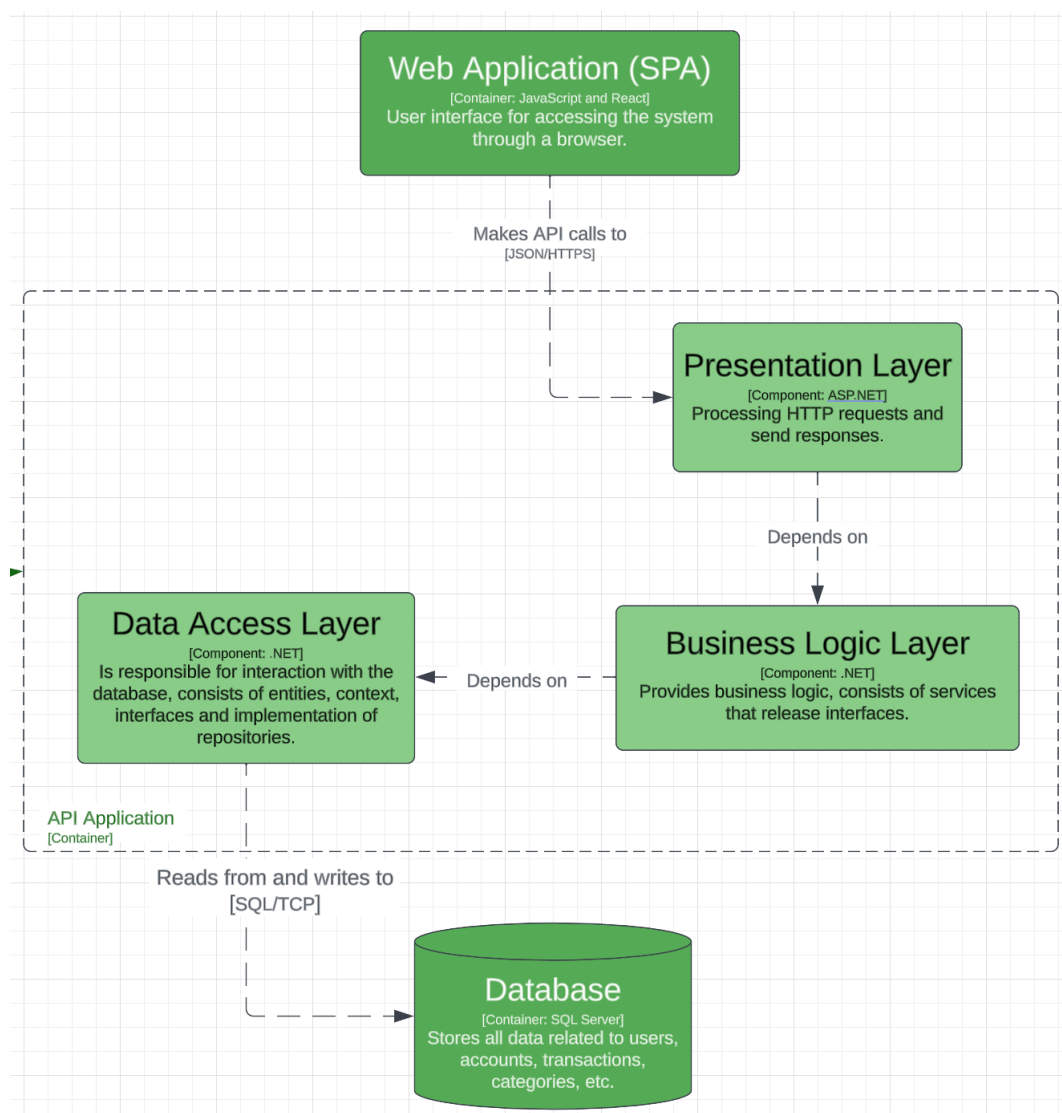
Тут можна виділити 3 основні складові:

- веб-застосунок, з яким взаємодіє користувач;
- серверний застосунок, який відповідає за функціонал на серверній стороні;
- база даних, яка зберігає всю інформацію щодо користувачів, їх рахунків, переказів тощо.

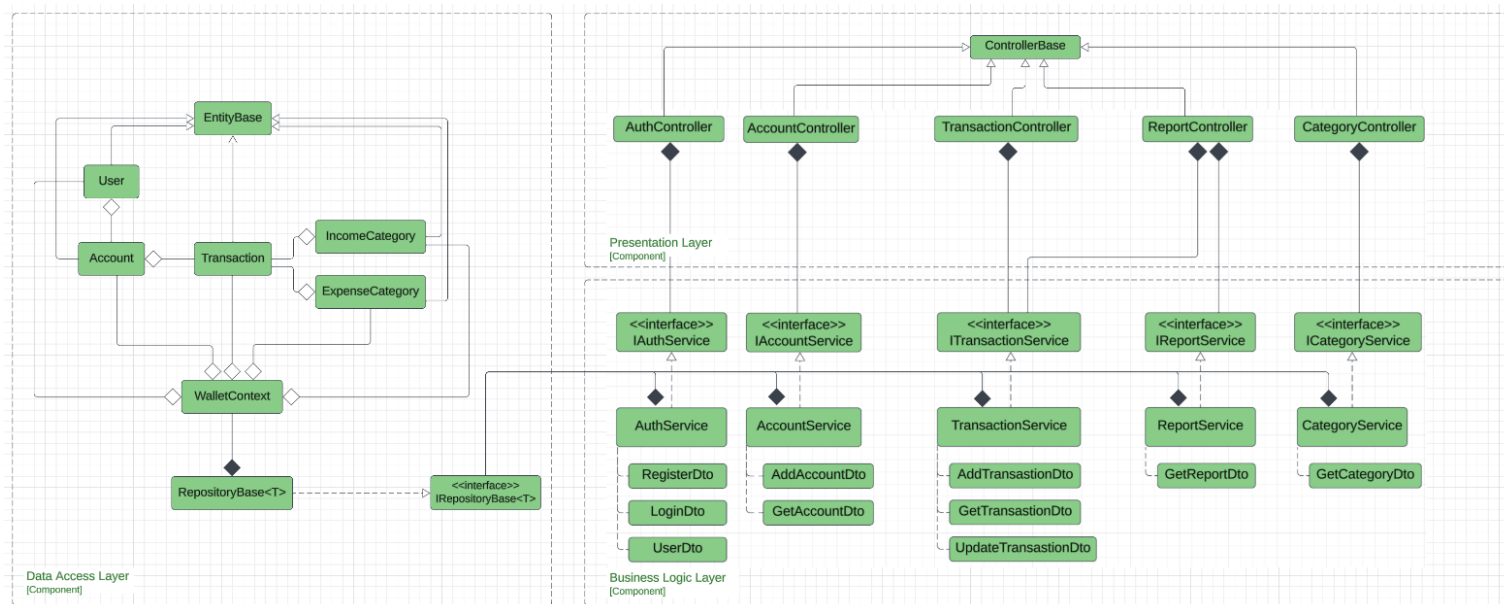
### Level 3 – Component Diagram.

Серверний застосунок можна поділити на такі 3 основні компоненти (шари):

- Presentation Layer – поверхнений шар, який через контролери приймає HTTP-запити і віддає відповідь;
- Business Logic Layer – шар, який виконує логіку додатку, складається з сервісів;
- Persistence Layer – відповідає за взаємодію з базою даних та складається з контексту застосунку, сутностей і репозиторіїв.



## Level 4 – Code Diagram.



На цій діаграмі розширено показано шари 3-го рівня.

Кінцеві точки REST API (Endpoints).

### User


























POST	/api/users/login	📋 🔒 🔗 🔍 ✓
POST	/api/users/register	📋 🔒 🔗 🔍 ✓
PUT	/api/users/{id}	📋 🔒 🔗 🔍 ✓
DELETE	/api/users/{id}	📋 🔒 🔗 🔍 ✓

### Account

POST	/api/accounts	📋 🔒 🔗 🔍 ✓
GET	/api/accounts/{id}	📋 🔒 🔗 🔍 ✓
PUT	/api/accounts/{id}	📋 🔒 🔗 🔍 ✓
DELETE	/api/accounts/{id}	📋 🔒 🔗 🔍 ✓
























## Transaction



POST	/api/transactions	    
GET	/api/transactions/account /{accountId}	    
GET	/api/transactions/{id}	    
PUT	/api/transactions/{id}	    
DELETE	/api/transactions/{id}	    
























## IncomeCategory



POST	/api/incomeCategories	    
GET	/api/incomeCategories	   
GET	/api/incomeCategories/{id}	   
PUT	/api/incomeCategories /{id}	    
DELETE	/api/incomeCategories /{id}	    






## ExpenseCategory



POST	/api/expenseCategories	    
GET	/api/expenseCategories	   
GET	/api/expenseCategories/{id}	   
PUT	/api/expenseCategories /{id}	    
DELETE	/api/expenseCategories /{id}	    

## Report



GET	/api/accounts/{accountId} /summary	    
-----	---------------------------------------	---