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

Перевірив:

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 спроєктоване рішення у вигляді звіту до лабораторної роботи.

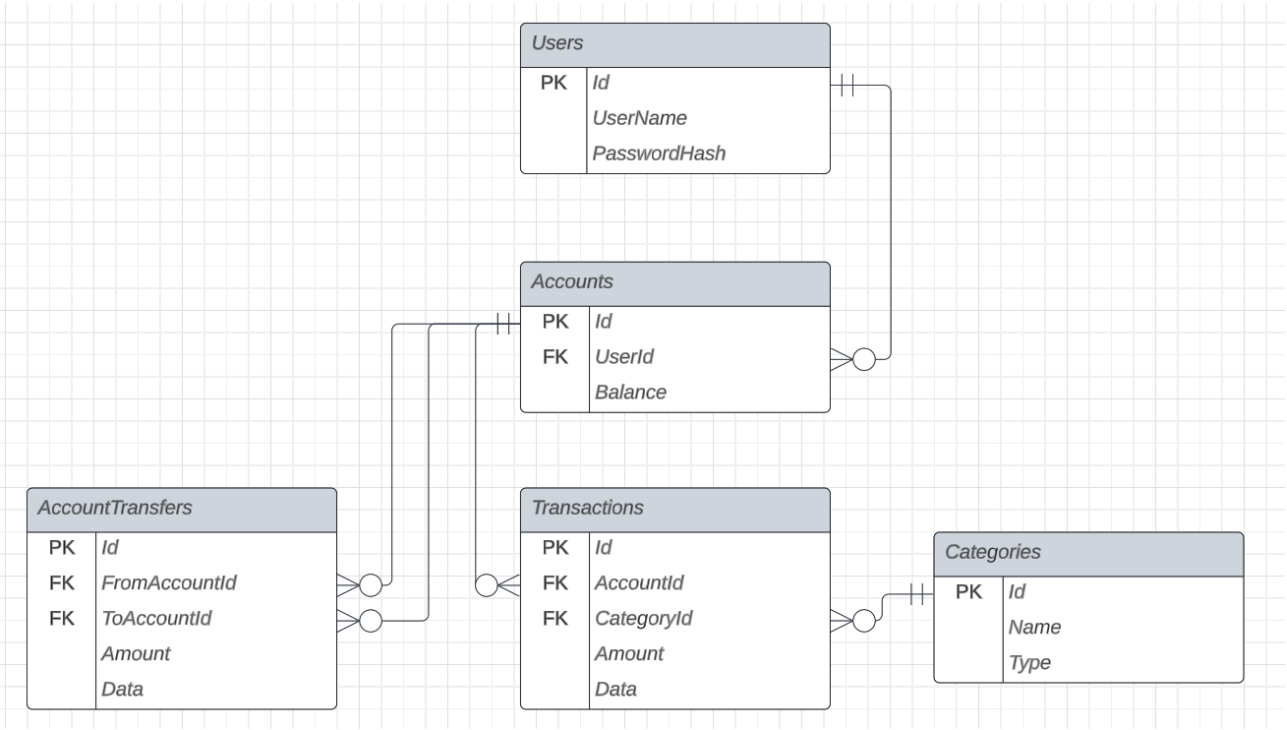
Документація:

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	1. Власний бюджет складається з декількох рахунків, які поповнюються за заданими статтями прибутку. 2. Гроші цих рахунків можуть бути переведені з одного на інший, можуть витрачатись за заданими статтями витрат. 3. Підсумовуючи витрати та прибутки, можливо отримати інформацію, скільки було витрачено/отримано загалом/за певною статтею по заданому рахунку. **Функціональні вимоги:** 1. Ведення власного бюджету; 2. Отримання звітної інформації по рахунках.

ER-діаграма.



Таблиця: **Users** – користувачі.

Поля:

- Id (PK): GUID – ідентифікатор користувача
- UserName: VARCHAR(50) – ім’я користувача
- PasswordHash: VARCHAR(255) – хеш пароля

Таблиця: **Accounts** – рахунки.

Поля:

- Id (PK): GUID – ідентифікатор рахунку
- UserId (FK): GUID – ідентифікатор користувача
- Balance: INTEGER – баланс рахунку

Таблиця: **AccountTransfers** – перекази між рахунками.

Поля:

- Id (PK): GUID – ідентифікатор переказу
- FromAccountId (FK): GUID – ідентифікатор рахунку з якого переказують
- ToAccountId (FK): GUID – ідентифікатор рахунку якому переказують
- Amount: INTEGER – сума переказу
- Data: DATETIME – дата переказу

Таблиця: **Transactions** – транзакції (витрати/доходи по категоріях).

Поля:

- Id (PK): GUID – ідентифікатор транзакції
- AccountId (FK): GUID – ідентифікатор рахунку
- CategoryId (FK): GUID – ідентифікатор категорії
- Amount: INTEGER – сума транзакції
- Data: DATETIME – дата транзакції

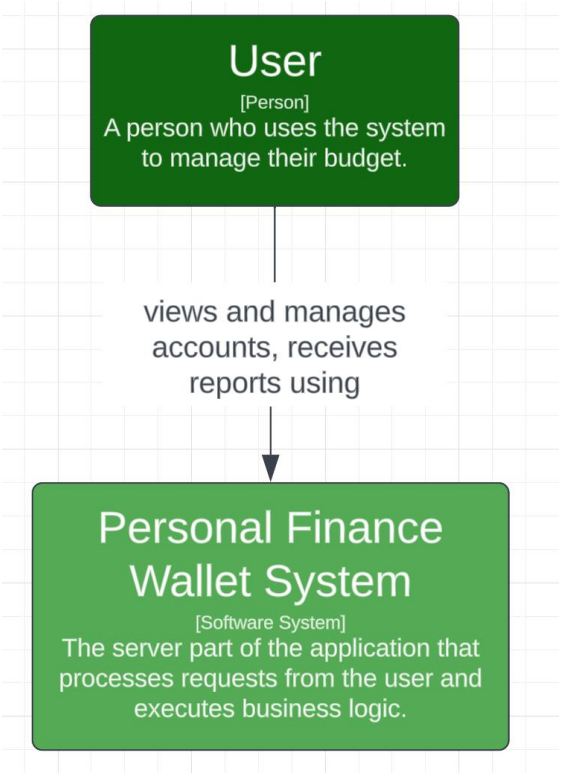
Таблиця: **Catagories** – категорії.

Поля:

- Id (PK): GUID – ідентифікатор категорії
- Name: VARCHAR(50) – назва категорії
- Type: TINYINT – тип категорії (0/1 – витрата/дохід)

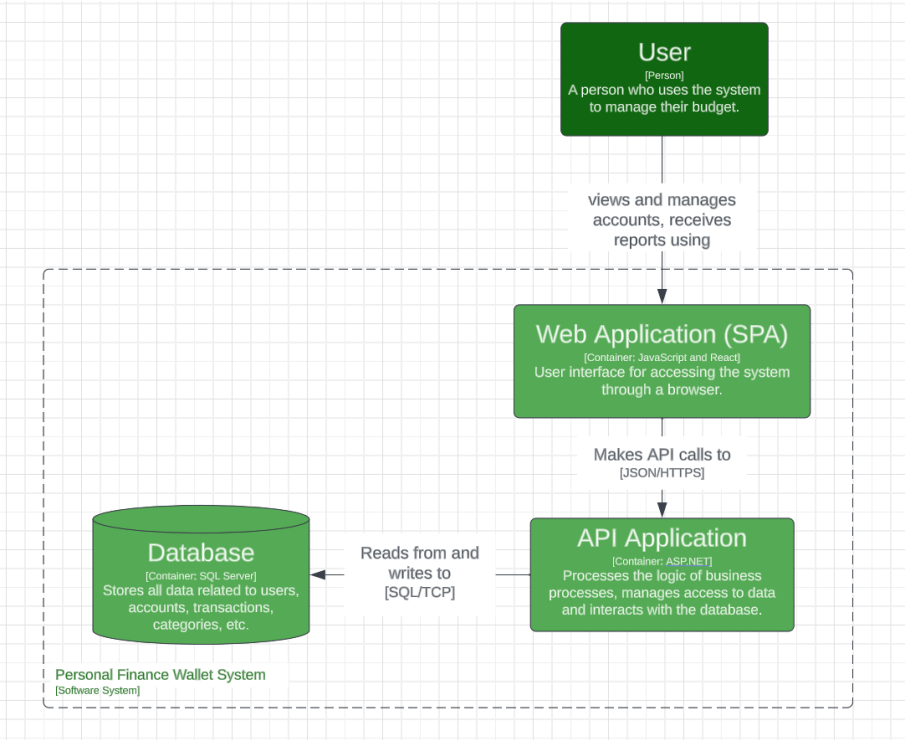
Діаграма C4.

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.



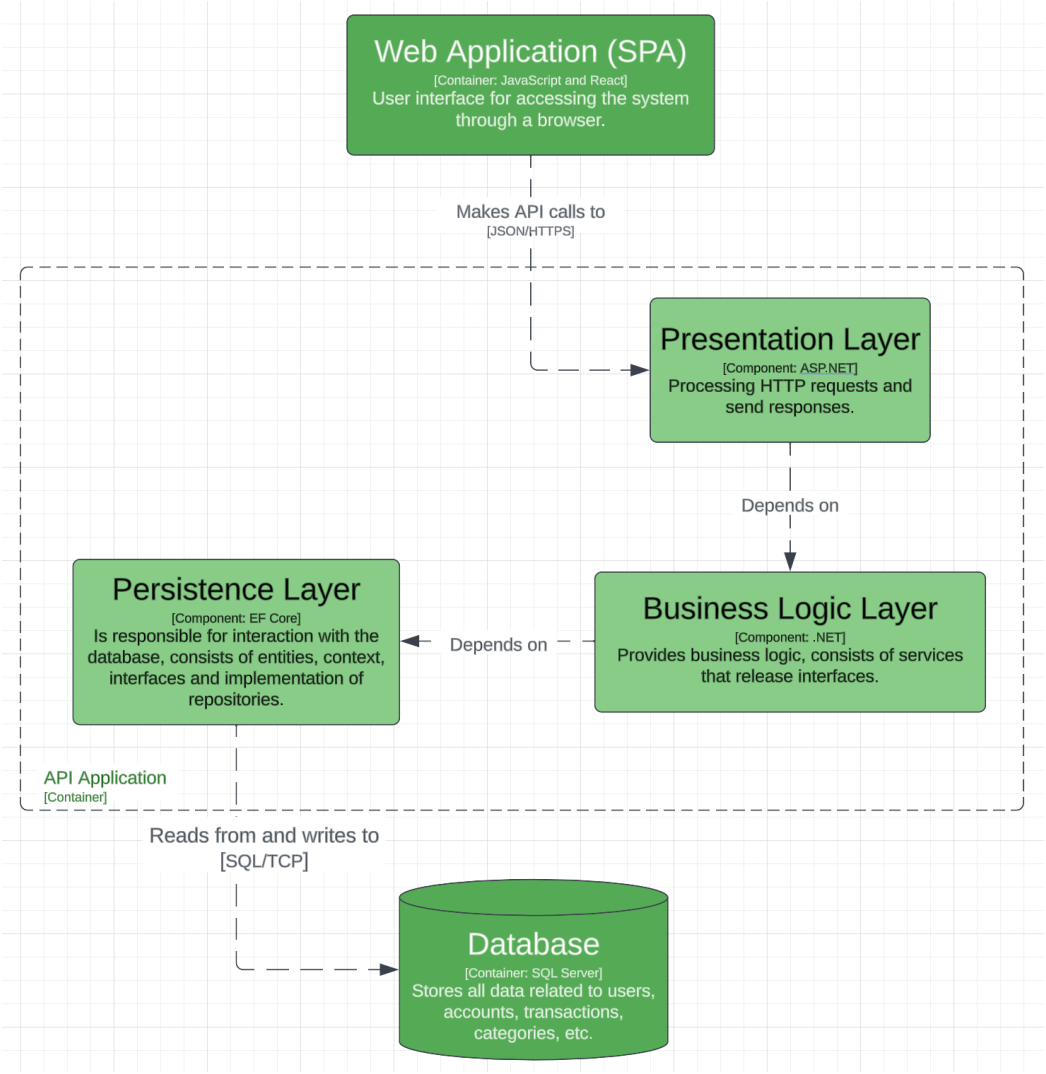
Тут можна виділити 3 основні складові:

- 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.

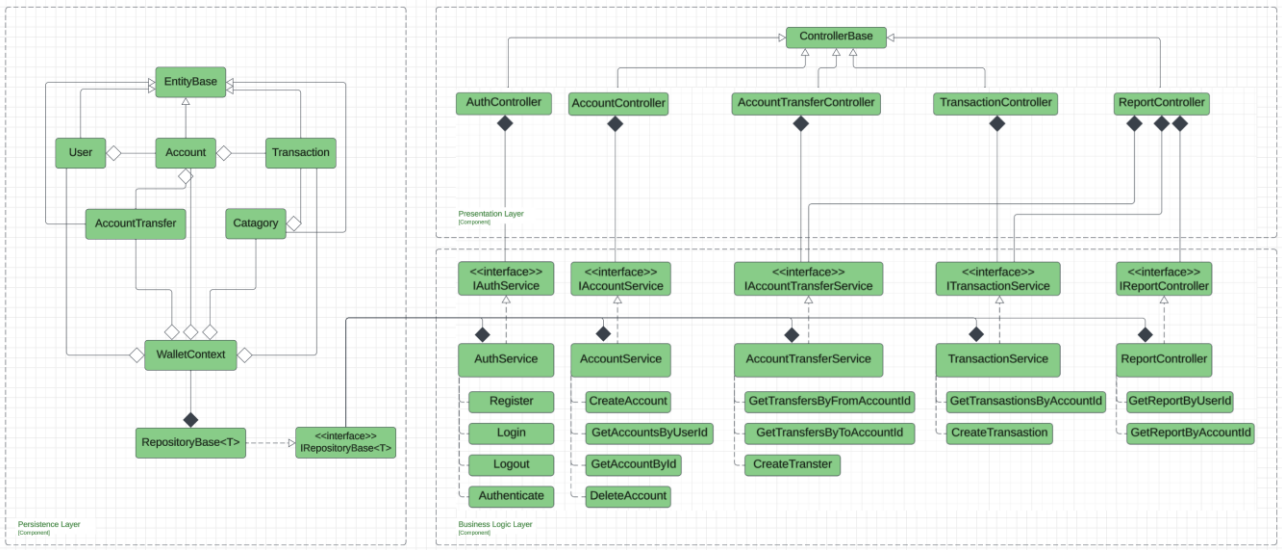
Level 3 – Component Diagram.

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).

Users:

- POST /api/users/register - реєстрація нового користувача.
- POST /api/users/login - логін існуючого користувача.
- GET /api/users/{userId} - отримати інформацію про користувача.

Accounts:

- GET /api/accounts/user/{userId} - отримати список всіх рахунків користувача.
- GET /api/accounts/{accountId} - отримати рахунок.
- POST /api/accounts - створити новий рахунок для користувача.
- DELETE /api/accounts/{accountId} - видалити рахунок.

Transactions:

- POST /api/transactions - виконати нову транзакцію (витрата/дохід).
- GET /api/transactions/user/{userId} - отримати список транзакцій користувача.
- GET /api/transactions/{accountId} - отримати список транзакцій по рахунку.
- DELETE /api/transactions/{transactionId} - видалити транзакцію.

AccountTransfers:

- POST /api/accountTransfers - створити переказ між рахунками користувача.
- GET /api/accountTransfers/from/{accountId} - отримати список переказів, які були здійснені з цього рахунку.
- GET /api/accountTransfers/to/{accountId} - отримати список переказів, які надійшли на цей рахунок.
- DELETE /api/ accountTransfers /{accountTransferId} - видалити переказ.

Accounts:

- GET /api/accounts/user/{userId} - отримати список всіх рахунків користувача.
- GET /api/accounts/{accountId} - отримати рахунок.

Reports:

- GET /api/reports/account/{accountId}/income – отримати звіт по прибутках по заданому рахунку.
- GET /api/reports/account/{accountId}/expenses – отримати звіт по витратах по заданому рахунку.
- GET /api/reports/account/{accountId}/category/{categoryId} – отримати звіт за певною категорією по заданому рахунку.