**Міністерство освіти і науки України**

**Національний університет «Запорізька Політехніка»**

Кафедра програмних засобів

**ПОЯСНЮВАЛЬНА ЗАПИСКА**

з самостійної роботи №2

з дисципліни «Системний аналіз» на тему:  
«ЗАСТОСУВАННЯ СИСТЕМНОГО ПІДХОДУ ПІД ЧАС РОЗРОБЛЕННЯ КОМПОНЕНТНО-ОРІЄНТОВАНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

Варіант №24

Виконав:

студент групи КНТ-112 І. А. Степанов

Прийняв

к.т.н, доцент: В. М. Льовкін

2024

зміст

[зміст 2](#_Toc169281245)

[вступ 10](#_Toc169281246)

[ОСНОВНА ЧАСТИНА 11](#_Toc169281247)

[1.1 Проєктування системи 11](#_Toc169281248)

[1.1.1 Вимоги до програмного забезпечення 11](#_Toc169281249)

[1.1.2 Вхідні і вихідні дані 11](#_Toc169281250)

[1.1.3 Структура бази даних 11](#_Toc169281251)

[1.1.4 Прототип графічного інтерфейсу користувача 12](#_Toc169281252)

[1.1.5 Система об’єктів 12](#_Toc169281253)

[1.1.6 Функціональні моделі основних процесів. 16](#_Toc169281342)

[1.2 Опис програми 17](#_Toc169281350)

[1.2.1 Функціональна схема програмного забезпечення 17](#_Toc169281351)

[1.2.2 Діаграма компонентів 18](#_Toc169281352)

[1.2.3 Опис модулів реалізованого програмного забезпечення з визначенням основних функцій 18](#_Toc169281353)

[1.2.4 Опис роботи з програмним забезпеченням з наведенням усіх необхідних скріншотів 19](#_Toc169281356)

[1.2.5 Опис повідомлень та оброблення критичних ситуацій 22](#_Toc169281364)

[1.3 Управління ризиками програмного забезпечення. 24](#_Toc169281378)

[1.3.1 Аналіз ризиків програмного забезпечення за етапами 24](#_Toc169281379)

[1.3.2 Опис використаних методів управління ризиками 25](#_Toc169281380)

[Висновки 26](#_Toc169281381)

[Додаток А Код Програми 28](#_Toc169281382)

[A1 - Код файлу IAuthenticationService.cs 28](#_Toc169281383)

[A2 - Код файлу IExpenseService.cs 28](#_Toc169281384)

[A3 - Код файлу AuthenticationService.cs 28](#_Toc169281385)

[A4 - Код файлу ExpenseService.cs 30](#_Toc169281386)

[A5 - Код файлу FormAddExpense.cs 34](#_Toc169281387)

[A6 - Код файлу FormAddFriend.cs 35](#_Toc169281388)

[A7 - Код файлу FormMain.cs 35](#_Toc169281389)

[A8 - Код файлу FormSignIn.cs 39](#_Toc169281390)

[A9 - Код файлу FormSignUp.cs 40](#_Toc169281391)

[A10 - Код файлу Program.cs 40](#_Toc169281392)

вступ

Системний підхід до програмування використовується в багатьох наукових дисциплінах, включаючи економіку, де застосовується світо-системний підхід для опису економічної історії.

Хоча системний підхід є фундаментальним для багатьох наукових дисциплін, прогалини знань можуть існувати, особливо у відношенні до нових технологій та методологій, які потребують додаткового дослідження та розвитку.

Основоположниками системного підходу є Л. фон Берталанфі, О. О. Богданов, Г. Саймон, П. Друкер, А. Чандлер. Ці вчені та інші дослідники в галузі системного підходу до програмування є провідними фахівцями у своїх областях.

Світові тенденції включають зростання використання системного підходу в різних галузях, включаючи розробку програмного забезпечення, інженерію, сільське господарство, розробку ігор та фізику. Це свідчить про його важливість та актуальність у сучасному світі.

Актуальність даної роботи та підставу для її виконання полягає в необхідності подальшого дослідження та розвитку системного підходу до програмування, а також у потребі застосування його в різних сферах для розв'язання складних проблем.

Мета роботи полягає в аналізі системного підходу до програмування, його застосуванні в різних галузях, а також у визначенні його впливу на розвиток технологій та науки. Галузь застосування включає різні сфери, від інженерії до сільського господарства, розробки ігор та фізики.

Взаємозв'язок з іншими роботами включає взаємодію з проектами з відкритим кодом, спільнотами розробників та навчальними програмами, що сприяє обміну знаннями та досвідом.

ОСНОВНА ЧАСТИНА

* 1. Проєктування системи

1.1.1 Вимоги до програмного забезпечення

Вимоги до програмного забезпечення включають:

* формування переліку власних витрат користувачами, визначаючи для них назву категорію, вартість;
* аналіз відсоткового розподілу за витратами, визначати найбільші з них, підраховувати загальні витрати за категоріями та загалом за вказаний період часу;
* надання доступу до перегляду власного бюджету деяким користувачам.

1**.1.2 Вхідні і вихідні дані**

Вхідними даними є електронна пошта та пароль користувача. Вихідними даними є таблиця витрат у головному вікні застосунку, яка дозволяє керувати власним бюджетом, дивитись всі витрати, витрати у певний період часу та відсотковий розподіл витрат.

1**.1.3 Структура бази даних**

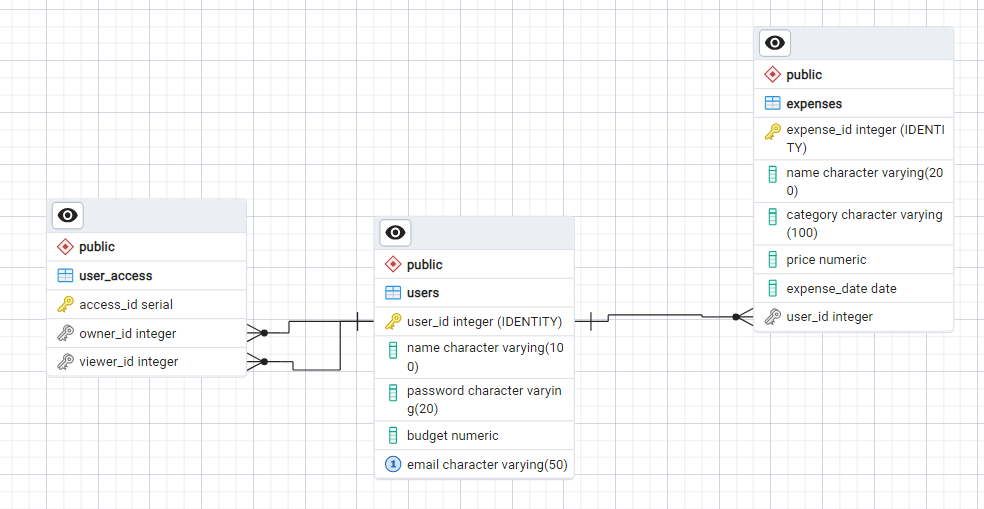


Рисунок 1.1 – Структура бази даних до застосунку в СКБД PostgreSQL

1.1.4 **Прототип графічного інтерфейсу користувача**

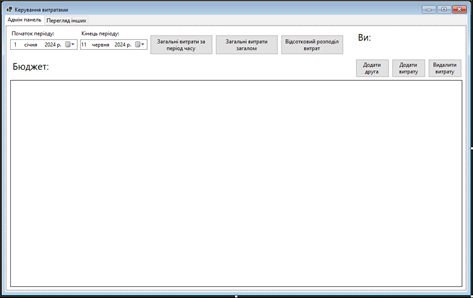


Рисунок 1.2 – Прототип графічного інтерфейсу користувача

1.1.5 **Система об’єктів**

При проєктуванні було розроблено систему об’єктів які в подальшому використовувались у даній предметній області. На наступній таблиці представлено дану систему об’єктів.

Таблиця 1.1 – Система об’єктів та їх атрибутів

|  |  |  |
| --- | --- | --- |
| Об’єкт | Атрибут/метод | Опис |
| User | user\_id | Номер користувача в системі |
| name | Ім’я користувача |
| password | Пароль користувача |
| budget | Бюджет користувача |
| email | Електронна пошта користувача |

Продовження таблиці 1.1

|  |  |  |
| --- | --- | --- |
| Об’єкт | Атрибут/метод | Опис |
| Expense | expense\_id | Номер витрати в ситсемі |
| name | Назва витрати |
| category | Категорія витрати |
| price | Ціна витрати |
| expense\_date | Дата витрати |
| user\_id | Користувач якому належить витрата |
| user\_access | access\_id | Номер доступу в системі |
| owner\_id | Номер користувача який надав доступ до перегляду |
| viewer\_id | Номер користувача якому надали доступ до перегляду |
| IAuthenticationService | SignIn | Вхід в акаунт |
| SignUp | Реєстрація акаунту |
| GetUserName | Отримання імені користувача |
| GetUserId | Отримання ID користувача |

Продовження таблиці 1.1

|  |  |  |
| --- | --- | --- |
| Об’єкт | Атрибут/метод | Опис |
| IExpenseService | LoadCurrentBudget | Завантаження бюджету користувача |
| LoadExpenses | Завантаження витрат користувача |
| GetExpensesByPeriod | Отримання витрат за певним періодом часу |
| GetExpensePercentageByCategory | Отримання відсоткового розподілу витрат користувача |
| AddExpense | Додавання Витрати |
| DeleteExpense | Видалення витрати |
| AddFriend | Додавання друга |
| GetFriends | Отримання списку друзів користувача |
| GetFriendBudget | Отримання бюджету друга |
| AuthenticationService | \_connectionString | Шлях до бази даних |
| SignIn | Вхід в акаунт |

Продовження таблиці 1.1

|  |  |  |
| --- | --- | --- |
| Об’єкт | Атрибут/метод | Опис |
| AuthenticationService | SignUp | Реєстрація акаунту |
| GetUserName | Отримання імені користувача |
| GetUserName | Отримання ID користувача |
| ExpenseService | \_connectionString | Шлях до бази даних |
| LoadCurrentBudget | Завантаження бюджету користувача |
| LoadExpenses | Завантаження витрат користувача |
| GetExpensesByPeriod | Отримання витрат за певним періодом часу |
| GetExpensePercentageByCategory | Отримання відсоткового розподілу витрат користувача |
| AddExpense | Додавання Витрати |
| DeleteExpense | Видалення витрати |
| AddFriend | Додавання друга |

Продовження таблиці 1.1

|  |  |  |
| --- | --- | --- |
| Об’єкт | Атрибут/метод | Опис |
| ExpenseService | GetFriends | Отримання списку друзів користувача |
| GetFriendBudget | Отримання бюджету друга |

На рисунку 1.3 зображено діаграму класів проєкту ExpenseManager.

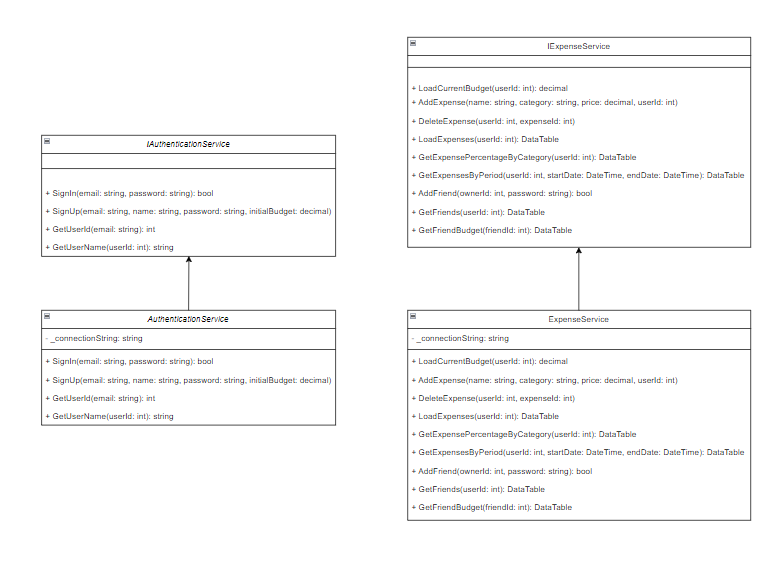


Рисунок 1.3 – Діаграма класів

1.1.6 **Функціональні моделі основних процесів.**

На наступних рисунках приведені функціональні моделі основних процесів проєкту ExpenseManager.

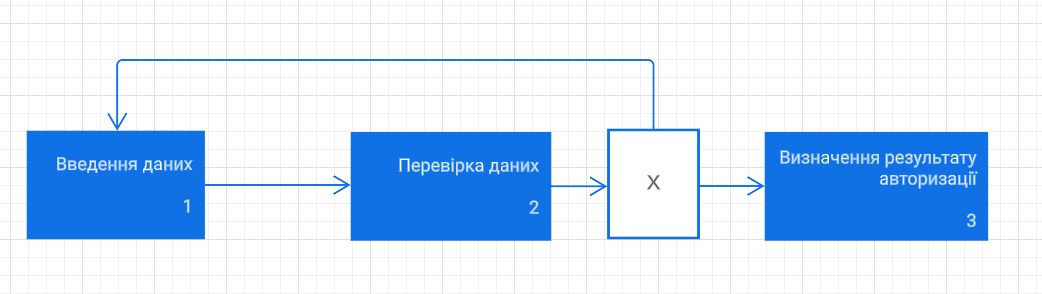


Рисунок 1.4 – Функціональна модель авторизації користувача

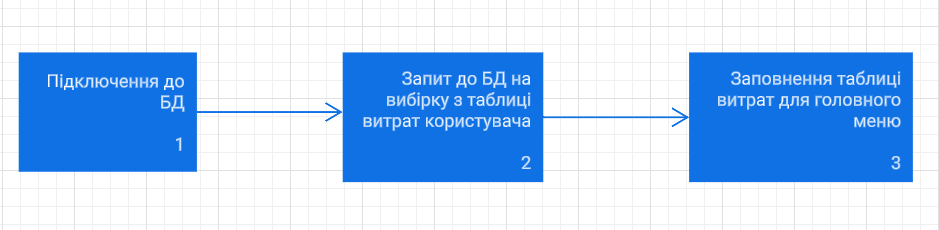


Рисунок 1.5 – Функціональна модель завантаження даних користувача у головній формі

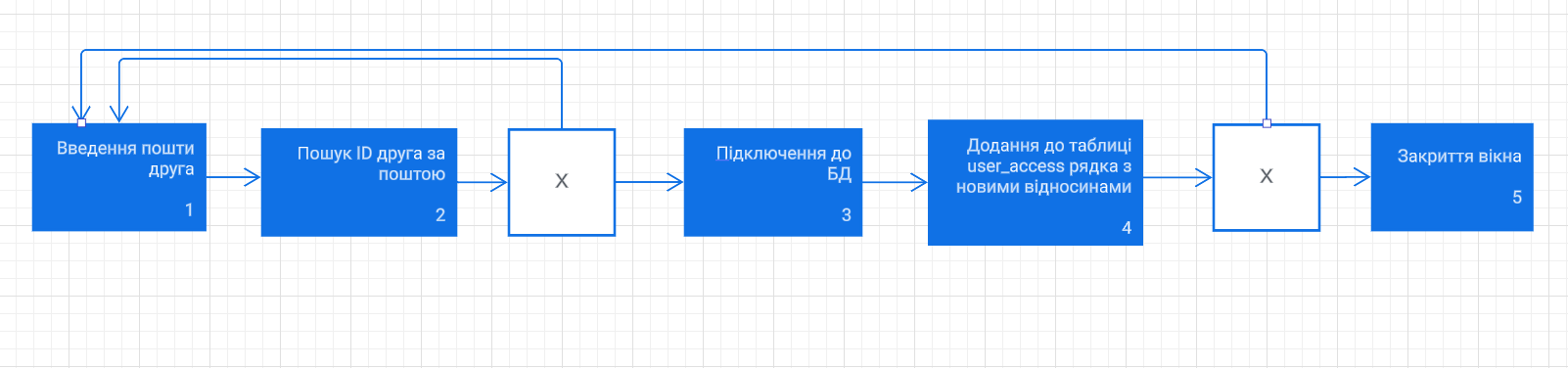


Рисунок 1.6 – Функціональна модель додавання друга(надання доступу до перегляду витрат деякому з користувачів)

* 1. Опис програми

1.2.1 Функціональна схема програмного забезпечення

На рисунку 1.7 зображено функціональну схему застосунку ExpenseManager.

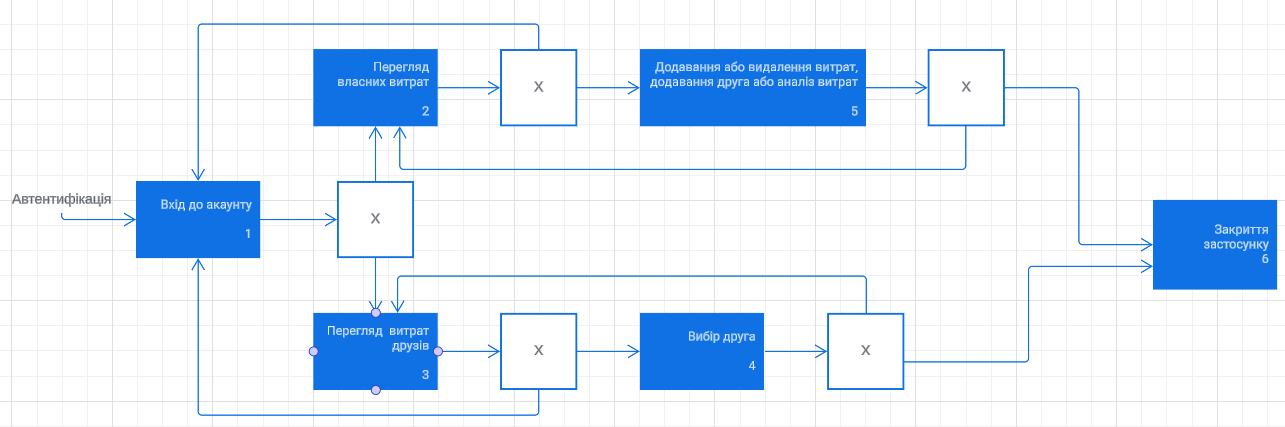


Рисунок 1.7 – Функціональна схема ПЗ ExpenseManager

1.2.2 **Діаграма компонентів**

На рисунку 1.8 зображено діаграму компонентів застосунку ExpenseManager.

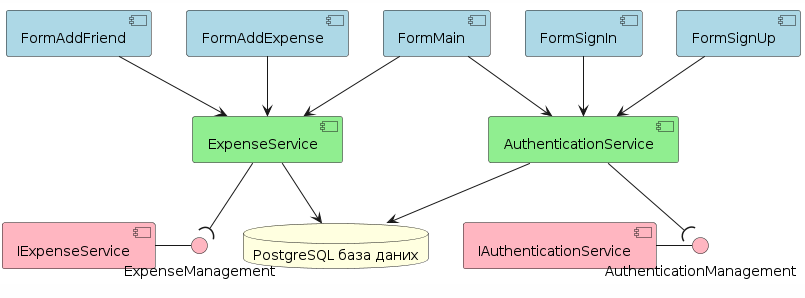


Рисунок 1.8 – Діаграма компонентів застосунку ExpenseManager

1.2.3 **Опис модулів реалізованого програмного забезпечення з визначенням основних функцій**

На наступній таблиці представлено опис модулів реалізованого програмного забезпечення з визначенням основних функцій.

Таблиця 1.2 – Опис модулів реалізованого програмного забезпечення

|  |  |
| --- | --- |
| Модуль | Функції |
| IAuthenticationService | Реєстрація акаунту користувача |

Продовження таблиці 1.2

|  |  |
| --- | --- |
| Модуль | Функції |
| IAuthenticationService | Авторизація користувача |
| Отримання ID та імені користувача |
| AuthenticationService | Реалізація методів IAuthenticationService |
| IExpenseService | Завантаження власного бюджету, витрат |
| Додавання та видалення витрат |
| IExpenseService | Додавання друзів (надання доступу певним користувачам) |
| Отримання відсоткового розподілу та витрат за періодом |
| Отримання списку друзів коритсувача та їх бюджетів |
| ExpenseService | Реалізація методів IExpenseService |

1.2.4 **Опис роботи з програмним забезпеченням з наведенням усіх необхідних скріншотів**

На рисунках 1.9 – 1.14 приведені скріншоти опису роботи з ПЗ ExpesneManager.

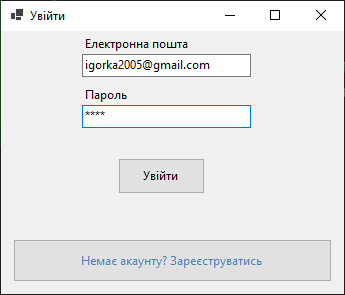


Рисунок 1.9 – Вікно входу до акаунту

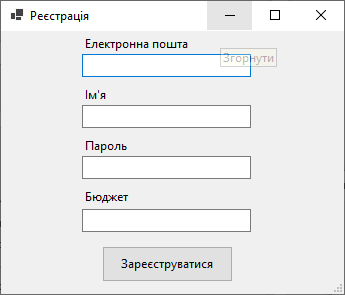


Рисунок 1.10 – Вікно реєстрації акаунту

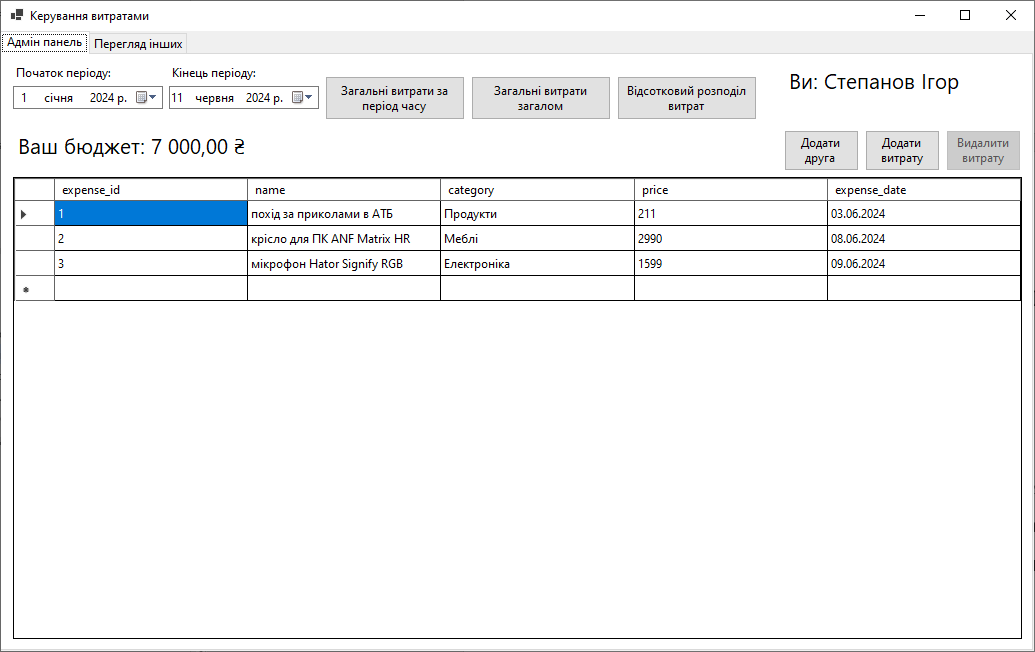


Рисунок 1.11 – Вікно головного меню

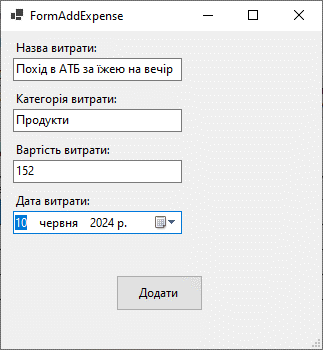


Рисунок 1.12 – Вікно додавання витрати

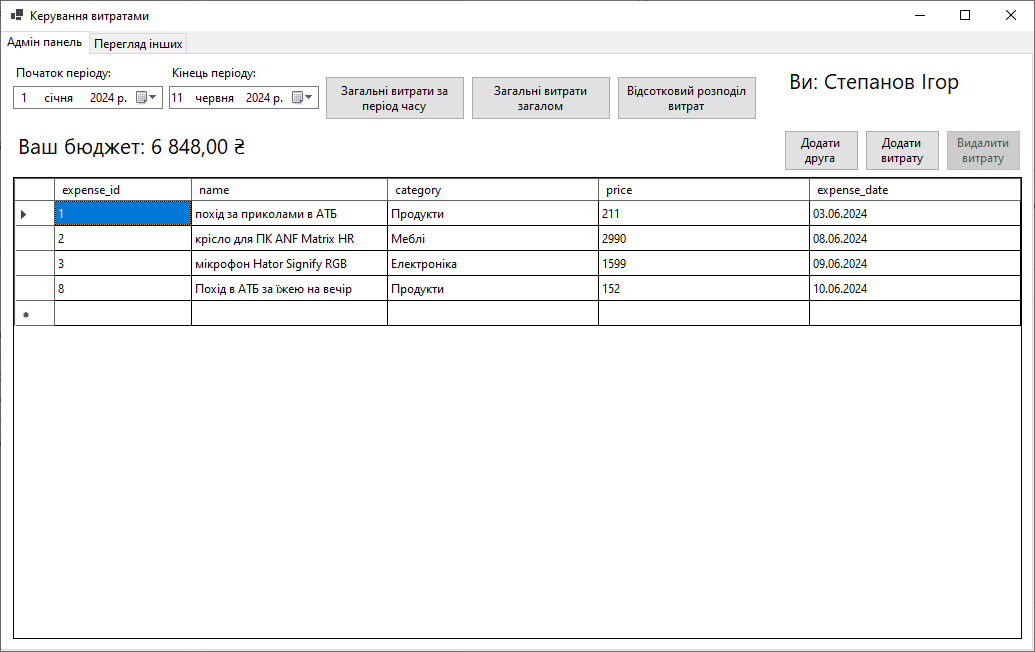


Рисунок 1.13 – Вікно головного меню після додавання витрати

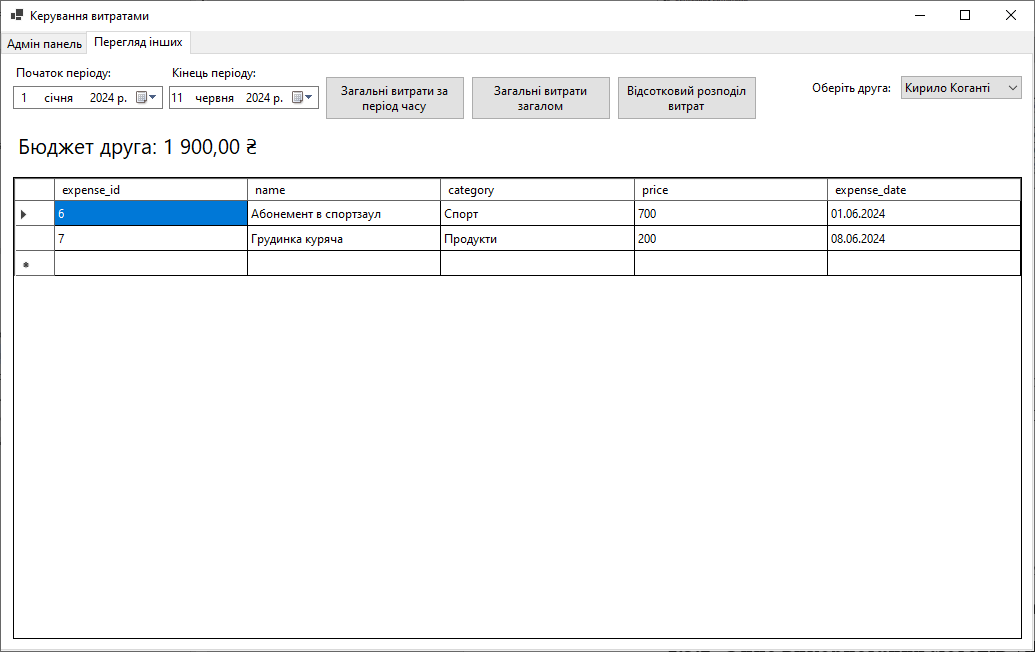


Рисунок 1.14 – Вікно головного меню у вкладці «Перегляд інших» своїх друзів

1.2.5 **Опис повідомлень та оброблення критичних ситуацій**

На рисунках 1.15 – 1.20 приведені основні обробки критичних ситуацій ПЗ ExpenseManager.

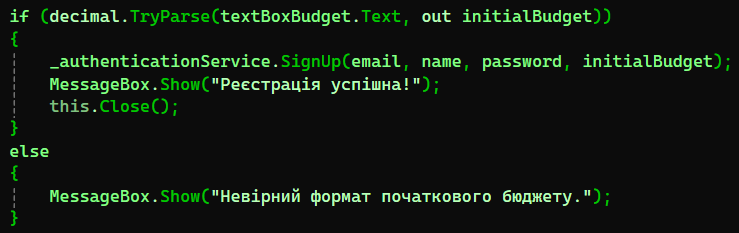


Рисунок 1.15 – Частина коду яка виконує обробку валідації введенного бюджету

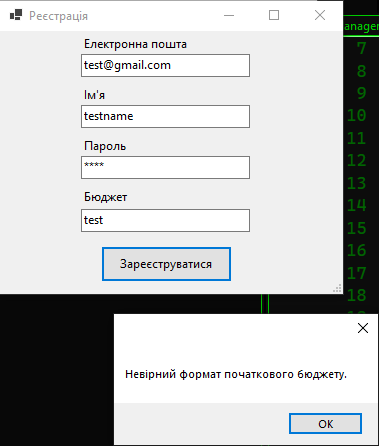


Рисунок 1.16 – Повідомлення обробки валідації введенного бюджету

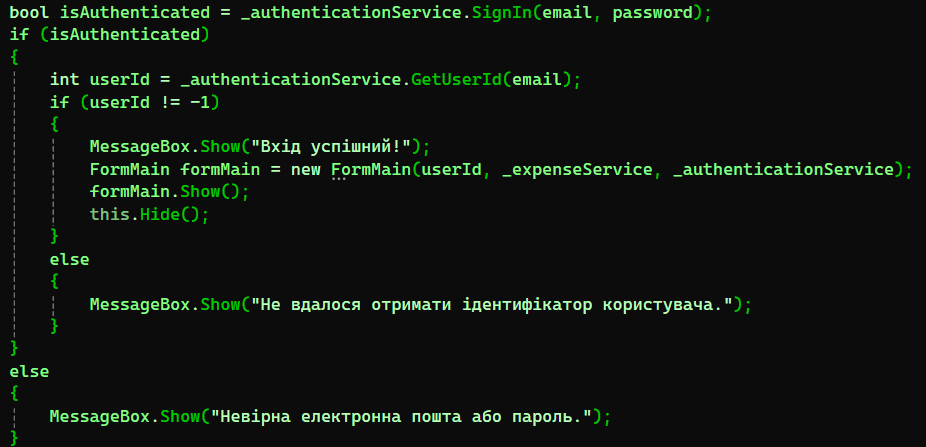


Рисунок 1.17 – Частина коду яка виконує обробку входу в акаунт

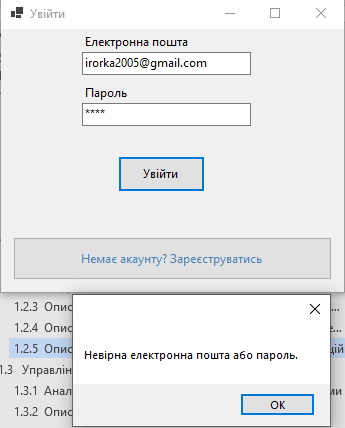


Рисунок 1.18 – Повідомлення обробки входу в акаунт

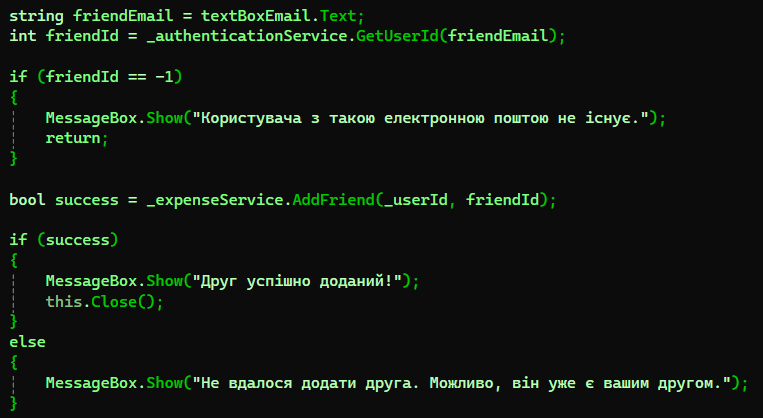


Рисунок 1.19 – Частина коду яка виконує обробку додавання друга

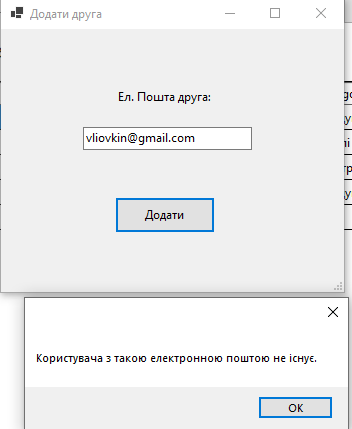


Рисунок 1.20 – Повідомлення обробки додавання друга

* 1. Управління ризиками програмного забезпечення.

1.3.1 Аналіз ризиків програмного забезпечення за етапами

Етап розробки підключення до бази даних: ризик невдачі підключення до БД(ймовірність (до управління): Висока (90%), вплив (до управління): високий (неспроможність функціонування застосунку), управління ризиком: підключення бібліотеки для функціонування застосунку з базою даних PostgreSQL “Npgsql”. Ймовірність (після управління): Низька (10%) вплив (після управління): помірний).

Етап розробки графічного інтерфейсу: ризик неправильного відображення таблиці витрат, переміщення колонок з місце на місце(ймовірність (до управління): Висока (90%), вплив (до управління): помірний (порушення послідовності колонок, естетичне невдоволення), управління ризиком: прописання алгоритму ручного налаштування колонок після натискання на певну кнопку виведення витрат. Ймовірність (після управління): Низька (10%) вплив (після управління): помірний).

Етап розробки запуску програми: ризик продовження роботи застосунку після натискання кнопки закриття вікна застосунку(ймовірність (до управління): Висока (95%), вплив (до управління): високий (неспроможність запуску програми через IDE, витрата ресурсів комп’ютера на дублікати застосунків), управління ризиком: накладення події закривання проблемних вікон застосунку через хрестик закриття з повним завершенням роботи програми. Ймовірність (після управління): Низька (10%) вплив (після управління): низький).

1.3.2 Опис використаних методів управління ризиками

При ризику невдачі підключення до БД було застосовано метод адекватності реакції. Було впроваджено адекватні міри по значущості ризику у вигляді коректного введення шляху до БД та підключення потрібної бібліотеки.

При ризику продовження роботи застосунку після натискання кнопки закриття вікна застосунку та при ризику неправильного відображення таблиці витрат, переміщення колонок з місце на місце. Були впроваджені максимальні ресурси на те, щоб знизити ризик до мінімуму.

Висновки

У ході виконання самостійної роботи було розроблено програмне забезпечення для управління особистими фінансами, яке включає функціонал для обліку витрат, відсоткового розподілу витрат за категоріями, а також можливість перегляду витрат і бюджету друзів. Найважливіші результати включають реалізацію основного функціоналу, такого як авторизація і реєстрація користувачів, облік витрат, додавання, видалення та перегляд витрат, а також отримання звітів за обраний період. Додатково реалізовано можливість перегляду бюджету і витрат друзів. Програма значно спрощує процес обліку особистих фінансів, забезпечуючи високий рівень безпеки даних і зручний інтерфейс користувача, розроблений з використанням WinForms.

Розроблене програмне забезпечення може бути використане як особистий фінансовий менеджер для індивідуального обліку фінансів, сімейний бюджетний менеджер для обліку витрат всієї родини, а також як інструмент для фінансового консультування. Наукова значущість роботи полягає у застосуванні сучасних методів програмування і управління даними, а соціальна значущість – у полегшенні процесу управління особистими фінансами і підвищенні фінансової грамотності користувачів.

Програма забезпечує зручний інтерфейс користувача і надійність роботи. Достовірність результатів обґрунтовується використанням перевірених інструментів і технологій, таких як PostgreSQL для баз даних і WinForms для розробки інтерфейсу. Програма пройшла тестування на різних етапах розробки, що забезпечує її надійність і коректну роботу.

Програмне забезпечення рекомендується для щоденного використання індивідуальними користувачами для обліку особистих фінансів і планування бюджету, для сімейного обліку витрат і бюджету родини, а також для комерційного використання фінансовими консультантами і малими підприємствами. В результаті проведеної роботи створено ефективний інструмент для управління особистими фінансами, який може бути корисним у різних сферах діяльності, сприяючи підвищенню фінансової грамотності і відповідального ставлення до фінансів.

Додаток А Код Програми

A1 - Код файлу IAuthenticationService.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ExpenseManager.Interfaces

{

public interface IAuthenticationService

{

bool SignIn(string email, string password);

void SignUp(string email, string name, string password, decimal initialBudget);

int GetUserId(string email);

string GetUserName(int userId);

}

}

A2 - Код файлу IExpenseService.cs

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ExpenseManager.Interfaces

{

public interface IExpenseService

{

decimal LoadCurrentBudget(int userId);

DataTable LoadExpenses(int userId);

DataTable GetExpensesByPeriod(int userId, DateTime startDate, DateTime endDate);

DataTable GetExpensePercentageByCategory(int userId);

void AddExpense(string name, string category, decimal price, DateTime date, int userId);

void DeleteExpense(int userId, int expenseId);

bool AddFriend(int ownerId, int friendId);

DataTable GetFriends(int userId);

DataTable GetFriendBudget(int friendId);

}

}

A3 - Код файлу AuthenticationService.cs

using ExpenseManager.Interfaces;

using Npgsql;

namespace ExpenseManager.ServiceClasses

{

public class AuthenticationService : IAuthenticationService

{

private readonly string \_connectionString;

public AuthenticationService(string connectionString)

{

\_connectionString = connectionString;

}

public bool SignIn(string email, string password)

{

try

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "SELECT COUNT(1) FROM users WHERE email = @Email AND password = @Password";

cmd.Parameters.AddWithValue("Email", email);

cmd.Parameters.AddWithValue("Password", password);

return (long)cmd.ExecuteScalar() > 0;

}

}

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

return false;

}

}

public int GetUserId(string email)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "SELECT user\_id FROM users WHERE email = @Email";

cmd.Parameters.AddWithValue("Email", email);

var result = cmd.ExecuteScalar();

return result != null ? Convert.ToInt32(result) : -1;

}

}

}

public string GetUserName(int userId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "SELECT name FROM users WHERE user\_id = @UserId";

cmd.Parameters.AddWithValue("UserId", userId);

return (string)cmd.ExecuteScalar();

}

}

}

public void SignUp(string email, string name, string password, decimal initialBudget)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "INSERT INTO users (email, name, password, budget) VALUES (@email, @name, @password, @budget)";

cmd.Parameters.AddWithValue("@email", email);

cmd.Parameters.AddWithValue("@name", name);

cmd.Parameters.AddWithValue("@password", password);

cmd.Parameters.AddWithValue("@budget", initialBudget);

cmd.ExecuteNonQuery();

}

}

}

}

}

A4 - Код файлу ExpenseService.cs

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using ExpenseManager.Interfaces;

using Npgsql;

namespace ExpenseManager.ServiceClasses

{

public class ExpenseService : IExpenseService

{

private readonly string \_connectionString;

public ExpenseService(string connectionString)

{

\_connectionString = connectionString;

}

public decimal LoadCurrentBudget(int userId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "SELECT budget FROM users WHERE user\_id = @UserId";

cmd.Parameters.AddWithValue("UserId", userId);

return (decimal)cmd.ExecuteScalar();

}

}

}

public DataTable LoadExpenses(int userId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = @"

SELECT expense\_id, name, category, price, expense\_date

FROM expenses

WHERE user\_id = @UserId";

cmd.Parameters.AddWithValue("UserId", userId);

using (var adapter = new NpgsqlDataAdapter(cmd))

{

DataTable expensesTable = new DataTable();

adapter.Fill(expensesTable);

return expensesTable;

}

}

}

}

public DataTable GetExpensesByPeriod(int userId, DateTime startDate, DateTime endDate)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = @"

SELECT expense\_id, name, category, price, expense\_date

FROM expenses

WHERE user\_id = @UserId AND expense\_date >= @StartDate AND expense\_date <= @EndDate";

cmd.Parameters.AddWithValue("UserId", userId);

cmd.Parameters.AddWithValue("StartDate", startDate);

cmd.Parameters.AddWithValue("EndDate", endDate);

using (var reader = cmd.ExecuteReader())

{

var dataTable = new DataTable();

dataTable.Load(reader);

return dataTable;

}

}

}

}

public DataTable GetExpensePercentageByCategory(int userId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = @"

SELECT category, SUM(price) AS total,

ROUND((SUM(price) \* 100.0 / (SELECT SUM(price) FROM expenses WHERE user\_id = @UserId)), 2) AS percentage

FROM expenses

WHERE user\_id = @UserId

GROUP BY category";

cmd.Parameters.AddWithValue("UserId", userId);

using (var reader = cmd.ExecuteReader())

{

var dataTable = new DataTable();

dataTable.Load(reader);

return dataTable;

}

}

}

}

public void AddExpense(string name, string category, decimal price, DateTime date, int userId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "INSERT INTO expenses (name, category, price, expense\_date, user\_id) VALUES (@name, @category, @price, @date, @userId)";

cmd.Parameters.AddWithValue("@name", name);

cmd.Parameters.AddWithValue("@category", category);

cmd.Parameters.AddWithValue("@price", price);

cmd.Parameters.AddWithValue("@date", date);

cmd.Parameters.AddWithValue("@userId", userId);

cmd.ExecuteNonQuery();

}

}

}

public void DeleteExpense(int userId, int expenseId)

{

decimal expensePrice = 0;

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "SELECT price FROM expenses WHERE expense\_id = @ExpenseId";

cmd.Parameters.AddWithValue("ExpenseId", expenseId);

expensePrice = (decimal)cmd.ExecuteScalar();

}

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "DELETE FROM expenses WHERE expense\_id = @ExpenseId";

cmd.Parameters.AddWithValue("ExpenseId", expenseId);

cmd.ExecuteNonQuery();

}

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = "UPDATE users SET budget = budget + @ExpensePrice WHERE user\_id = @UserId";

cmd.Parameters.AddWithValue("ExpensePrice", expensePrice);

cmd.Parameters.AddWithValue("UserId", userId);

cmd.ExecuteNonQuery();

}

}

}

public bool AddFriend(int ownerId, int friendId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = @"

INSERT INTO user\_access (owner\_id, viewer\_id)

VALUES (@OwnerId, @ViewerId)

ON CONFLICT (owner\_id, viewer\_id) DO NOTHING";

cmd.Parameters.AddWithValue("OwnerId", ownerId);

cmd.Parameters.AddWithValue("ViewerId", friendId);

int rowsAffected = cmd.ExecuteNonQuery();

return rowsAffected > 0;

}

}

}

public DataTable GetFriends(int userId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = @"

SELECT u.user\_id, u.name

FROM users u

JOIN user\_access ua ON u.user\_id = ua.owner\_id

WHERE ua.viewer\_id = @UserId";

cmd.Parameters.AddWithValue("UserId", userId);

using (var reader = cmd.ExecuteReader())

{

DataTable table = new DataTable();

table.Load(reader);

return table;

}

}

}

}

public DataTable GetFriendBudget(int friendId)

{

using (var conn = new NpgsqlConnection(\_connectionString))

{

conn.Open();

using (var cmd = new NpgsqlCommand())

{

cmd.Connection = conn;

cmd.CommandText = @"

SELECT u.user\_id, u.name, u.budget

FROM users u

WHERE u.user\_id = @FriendId";

cmd.Parameters.AddWithValue("FriendId", friendId);

using (var reader = cmd.ExecuteReader())

{

DataTable table = new DataTable();

table.Load(reader);

return table;

}

}

}

}

}

}

A5 - Код файлу FormAddExpense.cs

using ExpenseManager.Interfaces;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseManager

{

public partial class FormAddExpense : Form

{

private readonly IExpenseService \_expenseService;

private readonly int \_userId;

public FormAddExpense(int userId, IExpenseService expenseService)

{

\_expenseService = expenseService;

\_userId = userId;

InitializeComponent();

}

private void buttonAdd\_Click(object sender, EventArgs e)

{

string name = textBoxNameExpense.Text;

string category = textBoxCategory.Text;

decimal price;

DateTime date = dateTimePickerExpense.Value;

if (decimal.TryParse(textBoxPrice.Text, out price))

{

\_expenseService.AddExpense(name, category, price, date, \_userId);

MessageBox.Show("Витрату додано успішно.");

this.Close();

}

else

{

MessageBox.Show("Некоректна ціна. Будь ласка, введіть правильне значення.");

}

}

}

}

A6 - Код файлу FormAddFriend.cs

using ExpenseManager.Interfaces;

using ExpenseManager.ServiceClasses;

using Microsoft.VisualBasic.ApplicationServices;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ExpenseManager

{

public partial class FormAddFriend : Form

{

private readonly IAuthenticationService \_authenticationService;

private readonly IExpenseService \_expenseService;

private readonly int \_userId;

public FormAddFriend(IAuthenticationService authenticationService, IExpenseService expenseService, int userId)

{

InitializeComponent();

\_authenticationService = authenticationService;

\_expenseService = expenseService;

\_userId = userId;

}

private void buttonAdd\_Click(object sender, EventArgs e)

{

string friendEmail = textBoxEmail.Text;

int friendId = \_authenticationService.GetUserId(friendEmail);

if (friendId == -1)

{

MessageBox.Show("Користувача з такою електронною поштою не існує.");

return;

}

bool success = \_expenseService.AddFriend(\_userId, friendId);

if (success)

{

MessageBox.Show("Друг успішно доданий!");

this.Close();

}

else

{

MessageBox.Show("Не вдалося додати друга. Можливо, він уже є вашим другом.");

}

}

}

}

A7 - Код файлу FormMain.cs

using ExpenseManager.Interfaces;

using System.Data;

using ExpenseManager.ServiceClasses;

namespace ExpenseManager

{

public partial class FormMain : Form

{

private readonly int \_userId;

private readonly IExpenseService \_expenseService;

private readonly IAuthenticationService \_authenticationService;

public FormMain(int userId, IExpenseService expenseService, IAuthenticationService authenticationService)

{

\_userId = userId;

\_expenseService = expenseService;

\_authenticationService = authenticationService;

InitializeComponent();

LoadData();

LoadFriends();

}

private void LoadData()

{

decimal budget = \_expenseService.LoadCurrentBudget(\_userId);

labelBudget.Text = $"Ваш бюджет: {budget:C}";

string userName = \_authenticationService.GetUserName(\_userId);

labelUserName.Text = $"Ви: {userName}";

ExpensesDataGridView.DataSource = \_expenseService.LoadExpenses(\_userId);

ExpensesDataGridView.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

ExpensesDataGridView.ColumnHeadersHeightSizeMode = DataGridViewColumnHeadersHeightSizeMode.AutoSize;

foreach (DataGridViewColumn column in ExpensesDataGridView.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

}

private void buttonPercentageExpenses\_Click(object sender, EventArgs e)

{

DataTable percentagesTable = \_expenseService.GetExpensePercentageByCategory(\_userId);

ExpensesDataGridView.DataSource = percentagesTable;

buttonDeleteExpense.Visible = false;

}

private void buttonPeriodExpenses\_Click(object sender, EventArgs e)

{

DateTime startDate = dateTimePickerAdminFrom.Value;

DateTime endDate = dateTimePickerAdminTo.Value;

var expenses = \_expenseService.GetExpensesByPeriod(\_userId, startDate, endDate);

ExpensesDataGridView.DataSource = expenses;

ExpensesDataGridView.Columns["expense\_id"].DisplayIndex = 0;

ExpensesDataGridView.Columns["name"].DisplayIndex = 1;

ExpensesDataGridView.Columns["category"].DisplayIndex = 2;

ExpensesDataGridView.Columns["price"].DisplayIndex = 3;

ExpensesDataGridView.Columns["expense\_date"].DisplayIndex = 4;

buttonDeleteExpense.Visible = true;

}

private void buttonAllExpenses\_Click(object sender, EventArgs e)

{

ExpensesDataGridView.DataSource = \_expenseService.LoadExpenses(\_userId);

ExpensesDataGridView.Columns["expense\_id"].DisplayIndex = 0;

ExpensesDataGridView.Columns["name"].DisplayIndex = 1;

ExpensesDataGridView.Columns["category"].DisplayIndex = 2;

ExpensesDataGridView.Columns["price"].DisplayIndex = 3;

ExpensesDataGridView.Columns["expense\_date"].DisplayIndex = 4;

buttonDeleteExpense.Visible = true;

}

private void buttonAddExpense\_Click(object sender, EventArgs e)

{

FormAddExpense formAddExpense = new FormAddExpense(\_userId, \_expenseService);

formAddExpense.ShowDialog();

LoadData();

LoadFriends();

}

private void ExpensesDataGridView\_SelectionChanged(object sender, EventArgs e)

{

if (ExpensesDataGridView.SelectedRows.Count == 1)

buttonDeleteExpense.Enabled = true;

else

buttonDeleteExpense.Enabled = false;

}

private void buttonDeleteExpense\_Click(object sender, EventArgs e)

{

if (ExpensesDataGridView.SelectedRows.Count > 0)

{

int expenseId = Convert.ToInt32(ExpensesDataGridView.SelectedRows[0].Cells["expense\_id"].Value);

\_expenseService.DeleteExpense(\_userId, expenseId);

LoadData();

}

else

{

MessageBox.Show("Будь ласка, виберіть витрату для видалення.");

}

}

private void buttonAddFriend\_Click(object sender, EventArgs e)

{

FormAddFriend formAddFriend = new FormAddFriend(\_authenticationService, \_expenseService, \_userId);

formAddFriend.ShowDialog();

}

private void LoadFriends()

{

var friends = \_expenseService.GetFriends(\_userId);

comboBoxFriend.DataSource = friends;

comboBoxFriend.DisplayMember = "name";

comboBoxFriend.ValueMember = "user\_id";

}

private void LoadFriendExpenses(int friendId)

{

DataTable friendExpenses = \_expenseService.LoadExpenses(friendId);

dataGridViewFriend.DataSource = friendExpenses;

dataGridViewFriend.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

dataGridViewFriend.ColumnHeadersHeightSizeMode = DataGridViewColumnHeadersHeightSizeMode.AutoSize;

foreach (DataGridViewColumn column in dataGridViewFriend.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

decimal friendBudget = \_expenseService.LoadCurrentBudget(friendId);

labelFriendBudget.Text = $"Бюджет друга: {friendBudget:C}";

}

private void comboBoxFriend\_SelectedIndexChanged(object sender, EventArgs e)

{

if (comboBoxFriend.SelectedValue != null)

{

int friendId = (int)comboBoxFriend.SelectedValue;

LoadFriendExpenses(friendId);

}

}

private void buttonPeriodExpensesFriend\_Click(object sender, EventArgs e)

{

int friendId = (int)comboBoxFriend.SelectedValue;

DateTime startDate = dateTimePickerViewFrom.Value;

DateTime endDate = dateTimePickerViewTo.Value;

var expenses = \_expenseService.GetExpensesByPeriod(friendId, startDate, endDate);

dataGridViewFriend.DataSource = expenses;

dataGridViewFriend.Columns["expense\_id"].DisplayIndex = 0;

dataGridViewFriend.Columns["name"].DisplayIndex = 1;

dataGridViewFriend.Columns["category"].DisplayIndex = 2;

dataGridViewFriend.Columns["price"].DisplayIndex = 3;

dataGridViewFriend.Columns["expense\_date"].DisplayIndex = 4;

}

private void buttonAllExpensesFriend\_Click(object sender, EventArgs e)

{

int friendId = (int)comboBoxFriend.SelectedValue;

dataGridViewFriend.DataSource = \_expenseService.LoadExpenses(friendId);

dataGridViewFriend.Columns["expense\_id"].DisplayIndex = 0;

dataGridViewFriend.Columns["name"].DisplayIndex = 1;

dataGridViewFriend.Columns["category"].DisplayIndex = 2;

dataGridViewFriend.Columns["price"].DisplayIndex = 3;

dataGridViewFriend.Columns["expense\_date"].DisplayIndex = 4;

}

private void buttonPercentageExpensesFriend\_Click(object sender, EventArgs e)

{

int friendId = (int)comboBoxFriend.SelectedValue;

DataTable friendPercentagesTable = \_expenseService.GetExpensePercentageByCategory(friendId);

dataGridViewFriend.DataSource = friendPercentagesTable;

}

private void FormMain\_FormClosing(object sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

}

}

A8 - Код файлу FormSignIn.cs

using ExpenseManager.Interfaces;

namespace ExpenseManager

{

public partial class FormSignIn : Form

{

private readonly IAuthenticationService \_authenticationService;

private readonly IExpenseService \_expenseService;

public FormSignIn(IAuthenticationService authenticationService, IExpenseService expenseService)

{

\_authenticationService = authenticationService;

\_expenseService = expenseService;

InitializeComponent();

}

private void buttonSignUp\_Click(object sender, EventArgs e)

{

FormSignUp formSignUp = new FormSignUp(\_authenticationService);

formSignUp.ShowDialog();

}

private void buttonSignIn\_Click(object sender, EventArgs e)

{

string email = textBoxEmail.Text;

string password = textBoxPass.Text;

bool isAuthenticated = \_authenticationService.SignIn(email, password);

if (isAuthenticated)

{

int userId = \_authenticationService.GetUserId(email);

if (userId != -1)

{

MessageBox.Show("Вхід успішний!");

FormMain formMain = new FormMain(userId, \_expenseService, \_authenticationService);

formMain.Show();

this.Hide();

}

else

{

MessageBox.Show("Не вдалося отримати ідентифікатор користувача.");

}

}

else

{

MessageBox.Show("Невірна електронна пошта або пароль.");

}

}

private void FormSignIn\_FormClosing(object sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

}

}

A9 - Код файлу FormSignUp.cs

using ExpenseManager.Interfaces;

namespace ExpenseManager

{

public partial class FormSignUp : Form

{

private readonly IAuthenticationService \_authenticationService;

public FormSignUp(IAuthenticationService authenticationService)

{

\_authenticationService = authenticationService;

InitializeComponent();

}

private void buttonSignUp\_Click(object sender, EventArgs e)

{

string email = textBoxEmail.Text;

string name = textBoxName.Text;

string password = textBoxPass.Text;

decimal initialBudget;

if (decimal.TryParse(textBoxBudget.Text, out initialBudget))

{

\_authenticationService.SignUp(email, name, password, initialBudget);

MessageBox.Show("Реєстрація успішна!");

this.Close();

}

else

{

MessageBox.Show("Невірний формат початкового бюджету.");

}

}

}

}

A10 - Код файлу Program.cs

using ExpenseManager.Interfaces;

using ExpenseManager.ServiceClasses;

namespace ExpenseManager

{

internal static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

string connectionString = "Server=26.205.250.8;Port=5432;Database=sa\_lab\_4; User Id = stas; Password = stas2002";

IAuthenticationService authService = new AuthenticationService(connectionString);

IExpenseService expenseService = new ExpenseService(connectionString);

FormSignIn formSignIn = new FormSignIn(authService, expenseService);

Application.Run(formSignIn);

}

}

}